

John Macintyre  
Lazaros Iliadis  
Ilias Maglogiannis  
Chrisina Jayne (Eds.)

Communications in Computer and Information Science 1000

# Engineering Applications of Neural Networks

20th International Conference, EANN 2019  
Xersonisos, Crete, Greece, May 24–26, 2019  
Proceedings



Springer

# **Communications in Computer and Information Science**

## **1000**

*Commenced Publication in 2007*

Founding and Former Series Editors:

Phoebe Chen, Alfredo Cuzzocrea, Xiaoyong Du, Orhun Kara, Ting Liu,  
Krishna M. Sivalingam, Dominik Ślęzak, Takashi Washio, and Xiaokang Yang

### **Editorial Board Members**

Simone Diniz Junqueira Barbosa

*Pontifical Catholic University of Rio de Janeiro (PUC-Rio),  
Rio de Janeiro, Brazil*

Joaquim Filipe

*Polytechnic Institute of Setúbal, Setúbal, Portugal*

Ashish Ghosh

*Indian Statistical Institute, Kolkata, India*

Igor Kotenko

*St. Petersburg Institute for Informatics and Automation of the Russian  
Academy of Sciences, St. Petersburg, Russia*

Junsong Yuan

*University at Buffalo, The State University of New York, Buffalo, NY, USA*

Lizhu Zhou

*Tsinghua University, Beijing, China*

More information about this series at <http://www.springer.com/series/7899>

John Macintyre · Lazaros Iliadis ·  
Ilias Maglogiannis · Christina Jayne (Eds.)

# Engineering Applications of Neural Networks

20th International Conference, EANN 2019  
Xersonisos, Crete, Greece, May 24–26, 2019  
Proceedings



Springer

*Editors*

John Macintyre  
David Goldman Informatics Centre  
University of Sunderland  
Sunderland, UK

Ilias Maglogiannis  
University of Piraeus  
Piraeus, Greece

Lazaros Iliadis  
Democritus University of Thrace  
Xanthi, Greece

Chrisina Jayne  
Oxford Brookes University  
Oxford, UK

ISSN 1865-0929                    ISSN 1865-0937 (electronic)  
Communications in Computer and Information Science  
ISBN 978-3-030-20256-9        ISBN 978-3-030-20257-6 (eBook)  
<https://doi.org/10.1007/978-3-030-20257-6>

© Springer Nature Switzerland AG 2019

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

## **EANN 2019 Preface**

It is a fact that (according to Google) in September 2015 the search term “machine learning” (ML) became more popular than the term “artificial intelligence” (AI). According to the Economist, “data is the new oil of the 21st century.” Today, we are living the revolution of deep learning (DEL), convolutional neural networks (CNN), and big data (BD). DEL, ML, and AI can be considered as a set of Russian dolls. DEL is a subset of ML, which is a subset of AI.

In the following years, AI will become more widely available owing to the explosion of cloud computing.

EANN is a mature international scientific conference held in Europe and well established in the scientific area of AI. Its history is long and very successful, following and spreading the evolution of intelligent systems.

The first event was organized in Otaniemi, Finland, in 1995. Since then, it has had a continuous and dynamic presence as a major global but mainly European scientific event. More specifically, it has been organized in Finland, UK, Sweden, Gibraltar, Poland, Italy, Spain, Bulgaria, and Greece. It has always been technically supported by the International Neural Network Society (INNS) and more specifically by the EANN Special Interest Group.

Following a long-standing tradition, this Springer volume belongs to the CCIS Springer series and it contains the papers that were accepted to be presented orally at the 20th EANN 2019 conference and to the First Workshop on Pervasive Intelligence (PEINT). The diverse nature of papers presented demonstrates the vitality of AI algorithms and approaches. It certainly proves the very wide range of neural networks and AI applications as well.

The event was held during May 24–26, 2019, in the Aldemar Knossos Royal five-star hotel in Crete, Greece.

The response of the international scientific community to the EANN 2019 call for papers was more than satisfactory, with 74 papers initially submitted. All papers were peer reviewed by at least two independent academic referees. Where needed, a third referee was consulted to resolve any potential conflicts. A total of 48.6% of the submitted manuscripts (36 papers) were accepted to be published as full papers (12 pages long) in the Springer CCIS proceedings. Owing to the high quality of the submissions, the Program Committee decided that it should additionally accept five more submissions to be published as short papers (10 pages long).

PEINT, which was organized under the framework of EANN 2019, also followed the same review and acceptance ratio rules. More specifically, the workshop accepted four full papers out of nine submissions (44.4%).

The following scientific workshop on timely AI and ANN subjects was organized under the framework of the EANN 2019:

The First Workshop on Pervasive Intelligence (PEINT).

We would like to thank Professor Dimitris Iakovidis and Professor Evangelos Spyrou from the University of Thessaly Greece for their effort in organizing this interesting event.

Pervasive (ubiquitous) computing is a research area whose principle is to embed some kind of computational power (i.e., using microprocessors) into daily life objects, in an effort to make them able to communicate and perform tasks without the need for intense interaction with users. The concept of pervasive computing has recently emerged; a large number of applications such as wearable devices, smart/assistive homes and environments, smart cities, self-driving cars etc. are already part of everyday life. Pervasive computing devices are constantly available and networked, often interconnected with cloud services.

Among the plethora of the domains of application, several user groups such as people with disabilities or elderly persons may benefit the most. Disabled people may use smart devices so that difficulties within their daily life due their disabilities are surpassed. Moreover, elderly people may live in smart environments so that their activities of daily living may be monitored and they may be assisted to continue their lives independently, with minimal human intervention.

This workshop focused on methods and applications for data analysis in smart environments, enabled by AI, including (but not limited to) neural networks. It encouraged the submission of papers addressing concepts and methods related to the processing and analysis of data from multiple sensor modalities, especially high-throughput audio and video. Novel methods and algorithms in this context that cope with specific challenges and open research issues were presented. Experiments on publicly available datasets were also encouraged, to demonstrate the effectiveness of these methods. Application papers were sought that stress the societal impact of the proposed approach.

The First Workshop on Emerging Trends in AI (ETAI) was sponsored by the *Neural Computing and Applications* Springer journal. This was an open workshop without submission of papers.

We are grateful to Professor John Macintyre from the University of Sunderland, UK, for organizing this workshop and for his continuous support of the EANN conference. We wish to thank Professors Lary Medsker and Andrew Starr for their contribution to this very interesting workshop.

AI is going through a new boom period, with exponential growth in the commercialization of research and development, products being introduced into the market with embedded AI, as well as “intelligent systems” of various types. Projections for commercial revenue from AI also show exponential growth; such is the ubiquitous nature of AI in the modern world that members of the public are interacting with intelligent systems or agents every day – even though they often are not aware of it!

This workshop, led by Professor John Macintyre, considered emerging themes in AI, covering not only the technical aspects of where AI is going, but the wider question of ethics, and the potential for future regulatory frameworks for the development, implementation, and operation of intelligent systems and their role in our society.

The workshop format included three short presentations by the keynote speakers, followed by an interactive Q&A session where the panel members and audience engaged in a lively debate on the topics discussed.

The subjects of their presentations were the following:

John Macintyre: “The Future of AI – Existential Threat or New Revolution?”

Andrew Starr: “Practical AI for Practical Problems”

Four keynote speakers were invited to give lectures on timely aspects of artificial neural networks and AI:

1. Professor Plamen Angelov, University of Lancaster, UK: “Empirical Approach: How to Get Fast, Interpretable Deep Learning”
2. Dr. Evangelos Eleftheriou, IBM Fellow, Cloud and Computing Infrastructure, Zurich Research Laboratory Switzerland: “In-memory Computing: Accelerating AI Applications”
3. Dr. John Oommen, Carleton University, Ottawa, Canada: “The Power of Pursuit.” Learning Paradigm in the Partitioning of Data”
4. Professor Panagiotis Papapetrou, Stockholm University, Sweden: “Learning from Electronic Health Records: From temporal Abstraction to Timeseries Interpretability”

A three-hour tutorial on “Automated Machine Learning for Bioinformatics and Computational Biology” was given by Professor Ioannis Tsamardinos (Computer Science Department of University of Crete, co-founder of Gnosis Data Analysis PC, a university spin-off company, and Affiliated Faculty at IACM-FORTH) and Professor Vincenzo Lagani Ilia State University (Tbilisi, Georgia, and Gnosis Data Analysis PC co-founder).

Numerous bioinformaticians, computational biologists, and life scientists in general are applying supervised learning techniques and feature selection in their research work. The tutorial was addressed to this audience intending to shield them against methodological pitfalls, inform them about new methodologies and tools emerging in the field of Auto-ML, and increase their productivity.

The papers accepted for the 20th EANN conference are related to the following thematic topics:

- Deep learning ANN
- Genetic algorithms - optimization
- Constraints modeling
- ANN training algorithms
- Social media intelligent modeling
- Text mining/machine translation
- Fuzzy modeling
- Biomedical and bioinformatics algorithms and systems
- Feature selection
- Emotion recognition
- Hybrid intelligent models
- Classification-pattern recognition

- Intelligent security modeling
- Complex stochastic games
- Unsupervised machine learning
- ANN in industry
- Intelligent clustering
- Convolutional and recurrent ANN
- Recommender systems

The authors of submitted papers came from 19 different countries from all over the globe, namely: Australia, Austria, Brazil, Canada, Czech Republic, Germany, Greece, India, Italy, Japan, Malaysia, Norway, Romania, Russia, South Africa, Spain, Tunisia, the UK, and the USA.

May 2019

John Macintyre  
Lazaros Iliadis  
Ilias Maglogiannis  
Chrisina Jayne

# Organization

## Executive Committee

### General Chairs

John Macintyre

University of Sunderland UK (Dean of the Faculty of Applied Sciences and Pro Vice Chancellor of the University of Sunderland)

Chrisina Jayne

Oxford Brooks University, (Head of the School of Engineering, Computing and Mathematics)  
University of Piraeus, Greece

Ilias Maglogiannis  
(President of the IFIP  
WG12.5)

### Program Chairs

Lazaros Iliadis

Democritus University of Thrace, Greece  
University of the West of England, Bristol, UK

### Workshop Chairs

Christos Makris  
Phivos Mylonas  
Spyros Sioutas

University of Patras, Greece  
Ionian University, Greece  
University of Patras, Greece

### Advisory Chairs

Andreas Stafylopatis  
Georgios Vouros

National Technical University of Athens, Greece  
University of Piraeus, Greece

### Honorary Chairs

Vera Kurkova  
Barbara Hammer

Czech Academy of Sciences, Czech Republic  
Bielefeld University, Germany

### Publication and Publicity Chair

Antonis Papaleonidas

Democritus University of Thrace, Greece

### Program Committee

Michel Aldanondo  
Athanasios Alexiou  
Ioannis Anagnostopoulos  
George Anastassopoulos  
Costin Badica

IMT Mines Albi, France  
NGCEF, Australia  
University of Central Greece, Greece  
Democritus University of Thrace, Greece  
University of Craiova, Romania

Rashid Bakirov	Bournemouth University, UK
Kostas Berberidis	University of Patras, Greece
Nik Bessis	Edge Hill University, UK
Giacomo Boracchi	Politecnico di Milano, Italy
Farah Bouakrif	University of Jijel, Algeria
Antonio Braga	University Federal de Minas Gerais, Brazil
Peter Brida	University of Žilina, Slovakia
Ivo Bukovsky	Czech Technical University of Prague, Czech Republic
George Caridakis	National Technical University of Athens, Greece
Ioannis Chamodrakas	University of Athens, Greece
Ioannis Chochliouros	Hellenic Telecommunications Organization S.A. (OTE), Greece
Dawei Dai	Fudan University, China
Konstantinos Demertzis	Democritus University of Thrace, Greece
Ioannis Dokas	Democritus University of Thrace, Greece
Mauro Gaggero	National Research Council, Italy
Ignazio Gallo	University of Insubria, Italy
Claudio Gallicchio	University of Pisa, Italy
Spiros Georgakopoulos	University of Thessaly, Greece
Giorgio Gnecco	IMT School for Advanced Studies, Italy
Foteini Grivokostopoulou	University of Patras, Greece
Hakan Haberdar	University of Houston, USA
Petr Hajek	University of Pardubice, Czech Republic
Ioannis Hatzilygeroudis	University of Patras, Greece
Jian Hou	Bohai University, China
Lazaros Iliadis	Democritus University of Thrace, Greece
Jacek Kabziński	Lodz University of Technology, Poland
Antonios Kalampakas	American University of the Middle East, Kuwait
Antonios Karatzoglou	Karlsruhe Institute of Technology, Germany
Ioannis Karydis	Ionian University, Greece
Petros Kefalas	University of Sheffield International Faculty, Greece
Katia Lida Kermanidis	Ionian University, Greece
Yiannis Kokkinos	University of Macedonia, Greece
Petia Koprinkova-Hristova	Bulgarian Academy of Sciences, Bulgaria
Paul Krause	University of Surrey, UK
Ondrej Krejcar	University of Hradec Kralove, Czech Republic
Efthyvoulos Kyriacou	Frederick University of Cyprus, Cyprus
Ruggero Labati	Università degli Studi di Milano, Italy
Florin Leon	Technical University of Iasi, Romania
Hongyu Li	Zhongan Tech, China
Aristidis Likas	University of Ioannina, Greece
Spiros Likothanassis	University of Patras, Greece
Doina Logofatu	Frankfurt University of Applied Sciences, Germany
Ilias Maglogiannis	University of Piraeus, Greece
George Magoulas	University of London, Birkbeck College, UK
Mario Malcangi	Università degli Studi di Milano, Italy

Christos Makris	University of Patras, Greece
Francesco Marcelloni	University of Pisa, Italy
Nikolaos Mitianoudis	Democritus University of Thrace, Greece
Haralambos Mouratidis	University of Brighton, UK
Phivos Mylonas	Ionian University, Greece
Giannis Nikolentzos	Ecole Polytechnique, France
Stavros Ntalampiras	University of Milan, Italy
Mihaela Oprea	University Petroleum-Gas of Ploiesti, Romania
Basil Papadopoulos	Democritus University of Thrace, Greece
Antonios Papaleonidas	Democritus University of Thrace, Greece
Isidoros Perikos	University of Patras, Greece
Nicolai Petkov	University of Groningen, The Netherlands
Miltos Petridis	Middlesex University, UK
Elias Pimenidis	University of the West of England, UK
Niko Polatidis	University of the West of England, UK
Juan Qiu	Shanghai Jiao Tong University, China
Rafet Sifa	Fraunhofer IAIS, Germany
Bernardete Ribeiro	University of Coimbra, Portugal
Alexander Ryjov	Lomonosov Moscow State University, Russia
Marcello Sanguineti	University of Genoa, Italy
Simone Scardapane	Sapienza University of Rome, Italy
Andreas Stafylopatis	National Technical University of Athens, Greece
Ioannis Stephanakis	Hellenic Telecommunications Organisation SA, Greece
Ricardo Tanscheit	PUC-Rio, Brazil
Francesco Trovò	Politecnico di Milano, Italy
Nicolas Tsapatsoulis	Cyprus University of Technology, Cyprus
Nikolaos Vassilas	TEI of Athens, Greece
Petra Vidnerová	Czech Academy of Sciences, Czech Republic
Panagiotis Vlamos	Ionian University, Greece
George Vouros	University of Piraeus, Greece
Xin-She Yang	Middlesex University, UK
Shigang Yue	University of Lincoln, UK
Drago Žagar	University of Osijek, Croatia
Dongjie Zhang	University of Chinese Academy of Sciences, Beijing, China
Zhongnan Zhang	Xiamen University, China

## Preface PEINT2019

Pervasive (ubiquitous) computing is a research area whose principle is to embed some kind of computational power, using microprocessors, into daily life objects, in an effort to make them able to communicate and perform tasks without the need for intense interaction with users. The concept of pervasive computing has recently emerged; a large number of applications such as wearable devices, smart/assistive homes and environments, smart cities, self-driving cars etc. are already part of everyday life. Pervasive computing devices are constantly available and networked, often interconnected with cloud services.

Users may benefit from the plethora of application domains of pervasive computing. Several user groups, such as people with disabilities, or elderly persons, may benefit the most. Disabled people may use smart devices so as to surpass difficulties in their daily life. Moreover, elderly people may live in smart environments so that their activities of daily living may be monitored and they may be assisted to continue their lives independently, with minimal human intervention.

The International Workshop on Pervasive Intelligence (PEINT) focuses on methods and applications for data analysis in smart environments, enabled by artificial intelligence, including (but not limited to) neural networks. It encourages research addressing concepts and methods related to the processing and analysis of data from multiple sensor modalities, especially high-throughput audio and video. Novel methods and algorithms in this context should cope with specific challenges and open research issues. Applications should highlight the societal impact of the proposed approaches. Experiments on publicly available datasets are highly encouraged to demonstrate the effectiveness of the proposed methods and applications.

Topics of interest of PEINT Workshop include but are not limited to human action/activity and object recognition; human emotion recognition from audio/visual data; audio/visual methods for affective modeling; natural language processing for behavioral analysis; intelligent optical measurement systems; deep learning for image, audio, and multimodal data analysis; multimodal image fusion; wearable technologies for the disabled; smart/assistive environments; sensor networks for smart environments; dialogue systems; telemedicine; virtual and augmented reality environments; measurements for pervasive systems; decision-making based on multimodal cues; cloud computing for efficient data communications and processing; video coding, processing, and analysis.

A total of four high-quality papers were accepted as full papers (acceptance rate 45%) for PEINT 2019, covering most of the aforementioned topics, with contributions beyond the state of the art, both in terms of methodologies and applications. These include deep learning approaches for recognition of human actions, object detection and fuzzy image fusion approaches in the context of smart pervasive technologies for the visually impaired. Also, a neural network-based parallel coding methodology for

efficient video communications is presented, with applicability to a variety of domains, such as audiovisual cloud services and telemedicine.

## Co-chairs

Dimitris K. Iakovidis      University of Thessaly, Greece  
Evaggelos Spyrou      University of Thessaly, Greece

## Program Committee

Stylianos Asteriadis	University of Maastricht, The Netherlands
Charis Dakolia	University of Thessaly, Greece
Kostas Delibasis	University of Thessaly, Greece
Theodore Giannakopoulos	Behavioural Signals, Greece
Enrique Hortal	University of Maastricht, The Netherlands
Barna Iantovics	Mures University, Romania
Maria Kozyri	University of Thessaly, Greece
Artur Krukowski	Intracom S.A. Telecom Solutions, Greece
Athanasis Loukopoulos	University of Thessaly, Greece
Sara Paiva	Applied Research Centre for Digital Transformation, Portugal
Michalis Papakostas	University of Texas at Arlington, USA
Stavros Perantonis	National Center for Scientific Research Demokritos, Greece

**Acknowledgments.** This workshop was organized in the context of the project ENORASI (Intelligent Audiovisual System Enhancing Cultural Experience and Accessibility), co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE – INNOVATE (project code: T1EDK-02070).

## **Invited Papers**

# Learning from Electronic Health Records: From Temporal Abstractions to Time Series Interpretability

Panagiotis Papapetrou

Department of Computer and Systems Sciences, Stockholm University  
[panagiotis@dsv.su.se](mailto:panagiotis@dsv.su.se)

**Abstract.** The first part of the talk will focus on data mining methods for learning from Electronic Health Records (EHRs), which are typically perceived as big and complex patient data sources. On them, scientists strive to perform predictions on patients' progress, to understand and predict response to therapy, to detect adverse drug effects, and many other learning tasks. Medical researchers are also interested in learning from cohorts of population-based studies and of experiments. Learning tasks include the identification of disease predictors that can lead to new diagnostic tests and the acquisition of insights on interventions. The talk will elaborate on data sources, methods, and case studies in medical mining.

The second part of the talk will tackle the issue of interpretability and explainability of opaque machine learning models, with focus on time series classification. Time series classification has received great attention over the past decade with a wide range of methods focusing on predictive performance by exploiting various types of temporal features. Nonetheless, little emphasis has been placed on interpretability and explainability. This talk will formulate the novel problem of explainable time series tweaking, where, given a time series and an opaque classifier that provides a particular classification decision for the time series, the objective is to find the minimum number of changes to be performed to the given time series so that the classifier changes its decision to another class. Moreover, it will be shown that the problem is NP-hard. Two instantiations of the problem will be presented. The classifier under investigation will be the random shapelet forest classifier. Moreover, two algorithmic solutions for the two problem instantiations will be presented along with simple optimizations, as well as a baseline solution using the nearest neighbor classifier.

# **Empirical Approach: How to Get Fast, Interpretable Deep Learning**

Plamen Angelov

Department of Computing and Communications, University of Lancaster  
[p.angelov@lancaster.ac.uk](mailto:p.angelov@lancaster.ac.uk)

**Abstract.** We are witnessing an explosion of data (streams) being generated and growing exponentially. Nowadays we carry in our pockets Gigabytes of data in the form of USB flash memory sticks, smartphones, smartwatches etc. Extracting useful information and knowledge from these big data streams is of immense importance for the society, economy and science. Deep Learning quickly become a synonymous of a powerful method to enable items and processes with elements of AI in the sense that it makes possible human like performance in recognizing images and speech. However, the currently used methods for deep learning which are based on neural networks (recurrent, belief, etc.) is opaque (not transparent), requires huge amount of training data and computing power (hours of training using GPUs), is offline and its online versions based on reinforcement learning has no proven convergence, does not guarantee same result for the same input (lacks repeatability).

The speaker recently introduced a new concept of empirical approach to machine learning and fuzzy sets and systems, had proven convergence for a class of such models and used the link between neural networks and fuzzy systems (neuro-fuzzy systems are known to have a duality from the radial basis functions (RBF) networks and fuzzy rule based models and having the key property of universal approximation proven for both).

In this talk he will present in a systematic way the basics of the newly introduced Empirical Approach to Machine Learning, Fuzzy Sets and Systems and its applications to problems like anomaly detection, clustering, classification, prediction and control. The major advantages of this new paradigm are the liberation from the restrictive and often unrealistic assumptions and requirements concerning the nature of the data (random, deterministic, fuzzy), the need to formulate and assume a priori the type of distribution models, membership functions, the independence of the individual data observations, their large (theoretically infinite) number, etc.

From a pragmatic point of view, this direct approach from data (streams) to complex, layered model representation is automated fully and leads to very efficient model structures. In addition, the proposed new concept learns in a way similar to the way people learn – it can start from a single example. The reason why the proposed new approach makes this possible is because it is prototype based and non-parametric.

# “In-memory Computing”: Accelerating AI Applications

Evangelos Eleftheriou

IBM Fellow, Cloud and Computing Infrastructure, Zurich Research Laboratory,  
Zurich, Switzerland  
[ele@zurich.ibm.com](mailto:ele@zurich.ibm.com)

**Abstract.** In today’s computing systems based on the conventional von Neumann architecture, there are distinct memory and processing units. Performing computations results in a significant amount of data being moved back and forth between the physically separated memory and processing units. This costs time and energy, and constitutes an inherent performance bottleneck. It is becoming increasingly clear that for application areas such as AI (and indeed cognitive computing in general), we need to transition to computing architectures in which memory and logic coexist in some form. Brain-inspired neuromorphic computing and the fascinating new area of in-memory computing or computational memory are two key non-von Neumann approaches being researched. A critical requirement in these novel computing paradigms is a very-high-density, low-power, variable-state, programmable and non-volatile nanoscale memory device. There are many examples of such nanoscale memory devices in which the information is stored either as charge or as resistance. However, one particular example is phase-change-memory (PCM) devices, which are very well suited to address this need, owing to their multi-level storage capability and potential scalability.

In in-memory computing, the physics of the nanoscale memory devices, as well as the organization of such devices in cross-bar arrays, are exploited to perform certain computational tasks within the memory unit. I will present how computational memories accelerate AI applications and will show small- and large-scale experimental demonstrations that perform high-level computational primitives, such as ultra-low-power inference engines, optimization solvers including compressed sensing and sparse coding, linear solvers and temporal correlation detection. Moreover, I will discuss the efficacy of this approach to efficiently address not only inferencing but also training of deep neural networks. The results show that this co-existence of computation and storage at the nanometer scale could be the enabler for new, ultra-dense, low-power, and massively parallel computing systems. Thus, by augmenting conventional computing systems, in-memory computing could help achieve orders of magnitude improvement in performance and efficiency.

# Contents

## Invited Paper

- The Power of the “Pursuit” Learning Paradigm in the Partitioning of Data . . . . . 3  
*Abdolreza Shirvani and B. John Oommen*

## AI in Energy Management - Industrial Applications

- A Benchmark Framework to Evaluate Energy Disaggregation Solutions . . . . . 19  
*Nikolaos Symeonidis, Christoforos Nalmpantis, and Dimitris Vrakas*
- Application of Deep Learning Long Short-Term Memory in Energy Demand Forecasting . . . . . 31  
*Nameer Al Khafaf, Mahdi Jalili, and Peter Sokolowski*
- Modelling of Compressors in an Industrial CO<sub>2</sub>-Based Operational Cooling System Using ANN for Energy Management Purposes . . . . . 43  
*Sven Myrdahl Opalic, Morten Goodwin, Lei Jiao, Henrik Kofoed Nielsen, and Mohan Lal Kolhe*
- Outlier Detection in Temporal Spatial Log Data Using Autoencoder for Industry 4.0 . . . . . 55  
*Lukas Kaupp, Ulrich Beez, Jens Hülsmann, and Bernhard G. Humm*
- Reservoir Computing Approaches Applied to Energy Management in Industry . . . . . 66  
*Valentina Colla, Ismael Matino, Stefano Dettori, Silvia Cateni, and Ruben Matino*
- Signal2Vec: Time Series Embedding Representation . . . . . 80  
*Christoforos Nalmpantis and Dimitris Vrakas*

## Biomedical - Bioinformatics Modeling

- Classification of Sounds Indicative of Respiratory Diseases . . . . . 93  
*Stavros Ntalampiras and Ilyas Potamitis*
- Eye Disease Prediction from Optical Coherence Tomography Images with Transfer Learning . . . . . 104  
*Arka Bhowmik, Sanjay Kumar, and Neeraj Bhat*
- Severe Asthma Exacerbations Prediction Using Neural Networks . . . . . 115  
*Arthur Silveira, Cristian Muñoz, and Leonardo Mendoza*

A New Generalized Neuron Model Applied to DNA Microarray Classification . . . . .	125
<i>Beatriz A. Garro and Roberto A. Vazquez</i>	
<b>Classification - Learning</b>	
A Hybrid Approach for the Fighting Game AI Challenge: Balancing Case Analysis and Monte Carlo Tree Search for the Ultimate Performance in Unknown Environment . . . . .	139
<i>Lam Gia Thuan, Doina Logofătu, and Costin Badică</i>	
A Probabilistic Graph-Based Method to Improve Recommender System Accuracy . . . . .	151
<i>Nima Joorabloo, Mahdi Jalili, and Yongli Ren</i>	
A Robust Deep Ensemble Classifier for Figurative Language Detection . . . . .	164
<i>Rolandos-Alexandros Potamias, Georgios Siolas, and Andreas Stafylopatis</i>	
Enhanced Feature Selection for Facial Expression Recognition Systems with Genetic Algorithms . . . . .	176
<i>Kennedy Chengeta</i>	
Imaging Time-Series for NILM . . . . .	188
<i>Lamprini Kyrkou, Christoforos Nalmpantis, and Dimitris Vrakas</i>	
Learning Meaningful Sentence Embedding Based on Recursive Auto-encoders . . . . .	197
<i>Amal Bouraoui, Salma Jamoussi, and Abdelmajid Ben Hamadou</i>	
Pruning Extreme Wavelets Learning Machine by Automatic Relevance Determination . . . . .	208
<i>Paulo V. de Campos Souza, Vinicius J. Silva Araujo, Vanessa S. Araujo, Lucas O. Batista, and Augusto J. Guimaraes</i>	
Students' Performance Prediction Model Using Meta-classifier Approach . . . . .	221
<i>Hasniza Hassan, Syahid Anuar, and Nor Bahiah Ahmad</i>	
<b>Deep Learning</b>	
A Deep Network System for Simulated Autonomous Driving Using Behavioral Cloning . . . . .	235
<i>Andreea-Iulia Patachi, Florin Leon, and Doina Logofătu</i>	
A Machine Hearing Framework for Real-Time Streaming Analytics Using Lambda Architecture . . . . .	246
<i>Konstantinos Demertzis, Lazaros Iliidis, and Vardis-Dimitris Anezakis</i>	

Deep Learning and Change Detection for Fall Recognition . . . . .	262
<i>Sotiris K. Tasoulis, Georgios I. Mallis, Spiros V. Georgakopoulos, Aristidis G. Vrahatis, Vassilis P. Plagianakos, and Ilias G. Maglogiannis</i>	
Image Classification Using Deep Neural Networks: Transfer Learning and the Handling of Unknown Images . . . . .	274
<i>Vedang Chauhan, Keyur D. Joshi, and Brian Surgenor</i>	
LEARNAE: Distributed and Resilient Deep Neural Network Training for Heterogeneous Peer to Peer Topologies . . . . .	286
<i>Spyridon Nikolaidis and Ioannis Refanidis</i>	
Predicting Customer Churn Using Artificial Neural Network . . . . .	299
<i>Sanjay Kumar and Manish Kumar</i>	
Virtual Sensor Based on a Deep Learning Approach for Estimating Efficiency in Chillers . . . . .	307
<i>Serafín Alonso, Antonio Morán, Daniel Pérez, Perfecto Reguera, Ignacio Díaz, and Manuel Domínguez</i>	
<b>Deep Learning - Convolutional ANN</b>	
Canonical Correlation Analysis Framework for the Reduction of Test Time in Industrial Manufacturing Quality Tests . . . . .	323
<i>Paul Alexandru Bucur and Philipp Hungerländer</i>	
Convolutional Neural Network for Detection of Building Contours Using Multisource Spatial Data . . . . .	335
<i>George Papadopoulos, Nikolaos Vassilas, and Anastasios Kesidis</i>	
<b>Fuzzy - Vulnerability - Navigation Modeling</b>	
A Meta-metricriteria Approach to Estimate Drought Vulnerability Based on Fuzzy Pattern Recognition . . . . .	349
<i>M. Spiliotis, A. Iglesias, and L. Garrote</i>	
Bioinspired Early Prediction of Earthquakes Inferred by an Evolving Fuzzy Neural Network Paradigm . . . . .	361
<i>Mario Malcangi and Marco Malcangi</i>	
Enhancing Disaster Response for Hazardous Materials Using Emerging Technologies: The Role of AI and a Research Agenda . . . . .	368
<i>Jaziar Radiani, Ioannis Dokas, Kees Boersma, Nadia Saad Noori, Nabil Belbachir, and Stefan Stieglitz</i>	

**Machine Learning Modeling - Optimization**

Evolutionary Optimization on Artificial Neural Networks for Predicting the User's Future Semantic Location . . . . .	379
<i>Antonios Karatzoglou</i>	
Global Minimum Depth in Edwards-Anderson Model . . . . .	391
<i>Iakov Karandashev and Boris Kryzhanovsky</i>	
Imbalanced Datasets Resampling Through Self Organizing Maps and Genetic Algorithms . . . . .	399
<i>Marco Vannucci and Valentina Colla</i>	
Improvement of Routing in Opportunistic Communication Networks of Vehicles by Unsupervised Machine Learning . . . . .	412
<i>Ladislava Smítková Janků and Kateřina Hyniová</i>	
Machine Learning Approach for Drone Perception and Control. . . . .	424
<i>Yograj S. Mandloi and Yoshinobu Inada</i>	

**ML - DL Financial Modeling**

A Deep Dense Neural Network for Bankruptcy Prediction . . . . .	435
<i>Stamatis-Aggelos N. Alexandropoulos, Christos K. Aridas, Sotiris B. Kotsiantis, and Michael N. Vrahatis</i>	
Stock Price Movements Classification Using Machine and Deep Learning Techniques-The Case Study of Indian Stock Market . . . . .	445
<i>Nagaraj Naik and Biju R. Mohan</i>	
Study of Stock Return Predictions Using Recurrent Neural Networks with LSTM . . . . .	453
<i>Nagaraj Naik and Biju R. Mohan</i>	

**Security - Anomaly Detection**

Comparison of Network Intrusion Detection Performance Using Feature Representation . . . . .	463
<i>Daniel Pérez, Serafín Alonso, Antonio Morán, Miguel A. Prada, Juan José Fuertes, and Manuel Domínguez</i>	
Cyber Security Incident Handling, Warning and Response System for the European Critical Information Infrastructures (CyberSANE) . . . . .	476
<i>Spyridon Papastergiou, Haralambos Mouratidis, and Eleni-Maria Kalogeraki</i>	

Fault Diagnosis in Direct Current Electric Motors via an Artificial Neural Network . . . . .	488
<i>Theofanis I. Aravanis, Tryfon-Chrysovalantis I. Aravanis, and Polydoros N. Papadopoulos</i>	
<b>1st PEINT Workshop</b>	
On Predicting Bottlenecks in Wavefront Parallel Video Coding Using Deep Neural Networks . . . . .	501
<i>Natalia Panagou, Panagiotis Oikonomou, Panos K. Papadopoulos, Maria Koziri, Thanasis Loukopoulos, and Dimitris Iakovidis</i>	
Recognizing Human Actions Using 3D Skeletal Information and CNNs. . . . .	511
<i>Antonios Papadakis, Eirini Mathe, Ioannis Vernikos, Apostolos Maniatis, Evangelos Spyrou, and Phivos Mylonas</i>	
Staircase Detection Using a Lightweight Look-Behind Fully Convolutional Neural Network . . . . .	522
<i>Dimitrios E. Diamantis, Dimitra-Christina C. Koutsiou, and Dimitris K. Iakovidis</i>	
Obstacle Detection Based on Generative Adversarial Networks and Fuzzy Sets for Computer-Assisted Navigation . . . . .	533
<i>George Dimas, Charis Ntakolia, and Dimitris K. Iakovidis</i>	
<b>Author Index</b> . . . . .	545

# **Invited Paper**



# The Power of the “Pursuit” Learning Paradigm in the Partitioning of Data

Abdolreza Shirvani<sup>1</sup> and B. John Oommen<sup>1,2(✉)</sup>

<sup>1</sup> School of Computer Science, Carleton University, Ottawa, Canada  
oommen@scs.carleton.ca

<sup>2</sup> Centre for Artificial Intelligence Research, University of Agder, Grimstad, Norway

**Abstract.** Traditional Learning Automata (LA) work with the understanding that the actions are chosen purely based on the “state” in which the machine is. This *modus operandi* completely ignores any estimation of the Random Environment’s (RE’s) (specified as E) reward/penalty probabilities. To take these into consideration, Estimator/Pursuit LA utilize “cheap” estimates of the Environment’s reward probabilities to make them converge by an order of magnitude faster. This concept is quite simply the following: Inexpensive estimates of the reward probabilities can be used to rank the actions. Thereafter, when the action probability vector has to be updated, it is done not on the basis of the Environment’s response alone, but also based on the ranking of these estimates. While this phenomenon has been utilized in the field of LA, until recently, it has not been incorporated into solutions that solve partitioning problems. In this paper (The second author gratefully acknowledges the partial support of NSERC, the Natural Sciences and Engineering Council of Canada), we will submit a complete survey of how the “Pursuit” learning paradigm can be and has been used in Object Partitioning. The results demonstrate that incorporating this paradigm can hasten the partitioning by a order of magnitude.

**Keywords:** Object Partitioning · Learning Automata · Object Migration Automaton · Partitioning-based learning

## 1 Introduction

**The Pursuit Concept in LA:** Absolutely Expedient LA are absorbing and there is always a small probability of them not converging to the best action. Thathachar and Sastry realized this phenomenon and proposed to use Maximum Likelihood Estimators (MLEs) to hasten the LA’s convergence. Such an MLE-based update method would utilize estimates of the reward probabilities in the update equations. At every iteration, the estimated reward vector was also used to update the action probabilities, instead of updating it based only on the RE’s feedback. In this way, the probabilities of choosing the actions with higher reward estimates were increased, and those with lower estimates were significantly reduced, using which they proposed the family of estimator algorithms.

The Pursuit strategy of designing LA is a special derivative of the family of estimator algorithms. Pursuit algorithms “pursue” the currently-known best action, and increase the action probability associated with *this* action. The pursuit concept was first introduced by Thathachar *et al.*, and the corresponding LA was proven to be  $\epsilon$ -optimal. Its discretized version was proposed by Lanc-tot *et al.* in [7], who also discretized the original estimator algorithm. Agache *et al.* [10] then analyzed all the four linear combinations, i.e., the  $L_{RI}$  and  $L_{RP}$  paradigms.

**The Object Partitioning Problem (OPP):** Consider the problem of partitioning a set  $\mathcal{A} = \{A_1, \dots, A_W\}$  of  $W$  physical objects into  $\mathcal{R}$  groups  $\Omega = \{G_1, \dots, G_R\}$ . We assume that the true but unknown state of nature,  $\Omega^*$ , is a partitioning of the set  $\mathcal{A}$  into mutually exclusive and exhaustive subsets  $\{G_1^*, G_2^*, \dots, G_R^*\}$ . The composition of  $\{G_i^*\}$  is unknown, and the elements in the subsets fall together based on some criteria which may be mathematically formulated, or may even be ambiguous. These objects are now presented to a learning algorithm, for example, in pairs or tuples. The goal of the algorithm is to partition  $\mathcal{A}$  into a learned partition,  $\Omega^+$ . The hope is to have  $\Omega^+$  converge to  $\Omega^*$ . In most cases, the underlying partitioning of  $\Omega^*$  is not known, *nor* are the joint access probabilities by which the pairs/tuples of  $\mathcal{A}$  are presented to the learning algorithm known. This problem is known to be NP-hard [9]. Clearly, if we increase the number of objects, the number of partitions increases, and in addition to this quantity, the problem’s complexity grows exponentially. To resolve this, it is possible to explore all partition combinations, use a ranking index, and to thereafter, report the best plausible partition. The goal of the OPP is to identify the *best* or *most likely realizable* partitioning. This requires the AI algorithm to perceive the semantic physical world aspects of the objects, and to then make local decisions based on the best partition in the *abstract* domain [4, 5].

**Real *versus* Abstract Objects:** If there exists a mutual relation between the real objects in the semantic domain  $\mathcal{A}$ , and a domain of abstract objects  $\mathcal{O} = \{O_1, \dots, O_W\}$ , we define the partitioning of  $\mathcal{O}$  in such way that the corresponding partitions of  $\mathcal{O}$  map onto the partitions of the real objects in  $\mathcal{A}$  so as to mimic the state-of-nature. Thus, while we operate on the abstract objects in  $\mathcal{O}$ , the objects in  $\mathcal{A}$  are not necessarily moved because they constitute real-life objects which cannot be easily moved. A special case of the OPP is the Equi-Partitioning Problem (EPP) in which all the partitions are equi-sized.

**The Object Migrating Automation (OMA):** Due to the poor convergence of prior OPP/EPP solutions, they were never utilized in real-life applications. The introduction of an LA-based partitioning algorithm, the OMA, (explained in Sect. 2) made real-life applications possible. The OMA resolved the EPP both efficiently and accurately. This solution is regarded as a benchmark for the EPP. Indeed, since 1986<sup>1</sup>, it has been applied to variety of real-life problems and

---

<sup>1</sup> The bibliography in this paper is necessarily limited. The majority of the present results *very briefly* summarize the results in the Ph.D. thesis of the First Author.

domains which include keyboard optimization, image retrieval, distributed computing, graph partitioning, the constraint satisfaction problems, cryptanalysis, reputation systems, parallel and distributed mapping etc.

**The Intent of this Paper:** Although the “Pursuit” learning paradigm has been utilized in the theory and applications of LA as fundamental *machines*, until recently, it has not been incorporated into solutions that solve partitioning problems. The goal of this paper is to submit a comprehensive survey of how this paradigm can be used in Object Partitioning, and to optimize various versions of the OMA. We also include simulation results on benchmark environments that demonstrate the advantages of incorporating it into the respective machines.

## 2 The Object Migration Automata

The OMA is a fixed structure LA designed to solve the EPP. It is defined as a quintuple with  $R$  actions<sup>2</sup>, each of which represents a specific class, and for every action there exist a fixed number of states,  $N$ . Every abstract object from the set  $\mathcal{O}$  resides in a state identified by a number, and can move from one state to another, or migrate from one group to another. If the abstract object  $O_i$  is in state  $\xi_i$  belonging to a group  $\alpha_k$ , we say that  $O_i$  is assigned to class  $k$ .

If two objects  $O_i$  and  $O_j$  happen to be in the same class and the OMA receives a query  $\langle A_i, A_j \rangle$ , they are jointly rewarded by  $\mathbb{E}$ , the Environment. Otherwise, they will be penalized. We formalize the movements of  $\{O_i\}$  on reward/penalty.

We shall formalize the LA as follows: For every action  $\alpha_k$ , there is a set of states  $\{\phi_{k1}, \dots, \phi_{kN}\}$ , where  $N$  is the fixed depth of the memory, and where  $1 \leq k \leq R$  represents the number of desired classes. We also assume that  $\phi_{k1}$  is the most internal state and that  $\phi_{kN}$  is the boundary state for the corresponding action. The response to the reward and penalty feedback are as follows:

- **Reward:** Given a pair of physical objects presented as a query  $\langle A_i, A_j \rangle$ , if both  $O_i$  and  $O_j$  happen to be in the same class  $\alpha_k$ , the reward scenario is enforced, and they are both moved one step toward the actions’s most internal state,  $\phi_{k1}$ . This is depicted in Figure 3.2 (a) in [11]<sup>3</sup>.
- **Penalty:** If, however, they are in different classes,  $\alpha_k$  and  $\alpha_m$ , (i.e.,  $O_i$ , is in state  $\xi_i$  where  $\xi_i \in \{\phi_{k1}, \dots, \phi_{kN}\}$  and  $O_j$ , is in state  $\xi_j$  where  $\xi_j \in \{\phi_{m1}, \dots, \phi_{mN}\}$ ) they are moved away from  $\phi_{k1}$  and  $\phi_{m1}$  as follows:
  1. If  $\xi_i \neq \phi_{kN}$  and  $\xi_j \neq \phi_{mN}$ , we move  $O_i$  and  $O_j$  one state toward  $\phi_{kN}$  and  $\phi_{mN}$ , respectively, as shown in Figure 3.2 (b) in [11].
  2. If  $\xi_i = \phi_{kN}$  or  $\xi_j = \phi_{mN}$  but not both (i.e., only one of these abstract objects is in the boundry state), the object which is not in the boundary state, say  $O_i$ , is moved towards its boundary state as shown in Figure 3.2 (c) in [11]. Simultaneously, the object that is in the boundary state,  $O_j$ ,

<sup>2</sup> To be consistent with the terminology of LA, we use the terms “action”, “class” and “group” synonymously.

<sup>3</sup> The OMA’s algorithms/figures are in [11], and omitted here in the interest of space.

is moved to the boundary state of  $O_j$ . Since this reallocation will result in an excess of objects in  $\alpha_k$ , we choose one of the objects in  $\alpha_k$  (which is not accessed) and move it to the boundary state of  $\alpha_m$ . In this case, we choose the object nearest to the boundary state of  $\xi_i$ , as shown in Figure 3.2 (c) in [11].

3. If  $\xi_i = \phi_{kN}$  and  $\xi_j = \phi_{mN}$  (both objects are in the boundary states), one object, say  $O_i$ , will be moved to the boundary state of  $\alpha_m$ . Since this reallocation, will again, result in an excess of objects in  $\alpha_m$ , we choose one of the objects in  $\alpha_m$  (which is not accessed) and move it to the boundary state of  $\alpha_k$ . In this case, we choose the object nearest to the boundary state of  $\xi_j$ , as shown in Figure 3.2 (d) in [11].

To asses the partitioning accuracy and the convergence speed of any EPP solution, there must be an “oracle” with a pre-defined number of classes, and with each class containing an equal number of objects. The OMA’s goal is to migrate the objects between its classes, using the incoming queries.  $\mathbb{E}$  is characterized by three parameters: (a)  $W$ , the number of objects, (b)  $R$ , the number of partitions, and (c) a probability ‘ $p$ ’ quantifying how  $\mathbb{E}$  pairs the elements in the query.

Every query presented to the OMA by  $\mathbb{E}$  consists of two objects.  $\mathbb{E}$  randomly selects an initial class with probability  $\frac{1}{R}$ , and it then chooses the first object in the query from it, say,  $q_1$ . The second element of the pair,  $q_2$ , is then chosen with the probability  $p$  from the same class, and with the probability  $(1 - p)$  from one of the other classes uniformly, each of them being chosen with the probability of

**Table 1.** Experimental results for the OMA done for an ensemble of 100 experiments in which we have only included the results from experiments where convergence has occurred.

$W$	$W/R$	$R$	OMAp9	OMAp8	OMAp7
4	2	2	(2, 26)	(2, 36)	(2, 57)
6	2	3	(3, 44)	(4, 62)	(4, 109)
-	3	3	(22, 66)	(20, 88)	(26, 153)
9	3	3	(44, 110)	(43, 144)	(70, 261)
12	2	6	(10, 101)	(12, 146)	(15, 285)
-	3	4	(82, 172)	(84, 228)	(128, 406)
-	4	3	(401, 524)	(252, 405)	(256, 552)
-	6	2	(2240, 2370)	(1151, 1299)	(1053, 1486)
15	3	5	(152, 265)	(155, 325)	(191, 607)
-	5	3	(1854, 2087)	(918, 1136)	(735, 1171)
18	2	9	(17, 167)	(24, 252)	(29, 582)
-	3	6	(180, 319)	(202, 413)	(288, 839)
-	6	3	(5660, 5786)	(1911, 2265)	(1355, 2111)
-	9	2	(11245, 11456)	(6494, 7016)	(3801, 4450)

$\frac{1}{R-1}$ . Thereafter, it chooses a random element from the second class uniformly. We assume that  $\mathbb{E}$  generates an “unending” continuous stream of query pairs.

The results of the simulations are given in Table 1, where in  $OMAp\mathcal{X}$ ,  $\mathcal{X}$  refers to the probability specified above,  $W$ , is the number of objects,  $W/R$  is the number of objects per class, and  $R$  is the number of classes. The results are given as a pair  $(a, b)$  where  $a$  refers to the number of iterations for the OMA to reach the first correct classification and  $b$  refers to the case where the OMA has fully converged. In all experiments, the number of states of the OMA is set to 10. Also, the OMA’s convergence for a single run and for an ensemble of runs display a monotonically decreasing pattern (with time) for the latter.

### 3 Developing the Pursuit Concept: The Environment

In an “un-noisy” Environment, we can denote the actual value of the relation between  $A_i$  and  $A_j$  (for  $k \in \{1, \dots, R\}$ ) by the quantity  $\mu^*(i, j)$ , expressed as:

$$\begin{aligned}\mu^*(i, j) &= P(R_k) \cdot P(A_j|A_i) \cdot P(A_i), \forall i, j \text{ if } \langle A_i, A_j \rangle \in R_K, \\ &= 0 \text{ otherwise,}\end{aligned}$$

where  $P(R_k)$  is the probability that the first element,  $A_i$ , is chosen from the group  $R_k$ , and  $P(A_j|A_i)$  is the conditional probability of choosing  $A_j$ , which is also from  $R_k$ , after  $A_i$  has been chosen. Since  $\mathbb{E}$  chooses the elements of the pairs from the other groups uniformly, with a possible re-numbering operation, the matrix  $\mathcal{M}^* = [\mu^*(i, j)]$  is a *block-diagonal* matrix given by Eq. (1).

$$\mathcal{M}^* = \begin{bmatrix} \mathcal{M}_1^* & \underline{0} & \dots & \underline{0} \\ \underline{0} & \mathcal{M}_2^* & & \vdots \\ \vdots & & \ddots & \vdots \\ \underline{0} & \dots & \dots & \mathcal{M}_R^* \end{bmatrix} \quad (1)$$

where  $\underline{0}$  represents a square matrix containing only 0’s.

**Theorem 1.** *The matrix  $\mathcal{M}_r^*$ , ( $1 \leq r \leq R$ ), is a matrix of probabilities of size  $\frac{W}{R} \times \frac{W}{R}$  possessing the following form:*

$$\mathcal{M}_r^* = \begin{bmatrix} 0 & \frac{R}{W(W-R)} & \dots & \frac{R}{W(W-R)} \\ \frac{R}{W(W-R)} & 0 & \dots & \frac{R}{W(W-R)} \\ \vdots & & \ddots & \vdots \\ \frac{R}{W(W-R)} & \dots & \frac{R}{W(W-R)} & 0 \end{bmatrix} \quad (2)$$

*Proof.* The proof of the theorem is omitted here. It is found in [11].  $\square$

In a real-world scenario where  $\mathbb{E}$  is noisy, i.e., the objects from the different groups can be paired together in a query, the general form for  $\mathcal{M}^*$  is:

$$\mathcal{M}^* = \begin{bmatrix} \mathcal{M}_1^* & \underline{\theta} & \dots & \underline{\theta} \\ \underline{\theta} & \mathcal{M}_2^* & & \vdots \\ \vdots & & \ddots & \vdots \\ \underline{\theta} & \dots & \dots & \mathcal{M}_R^* \end{bmatrix} \quad (3)$$

where  $\underline{\theta}$  and  $\mathcal{M}_r^*$ s are specified as per Eqs. (4) and (5).

**Theorem 2.** *In the presence of noise in  $\mathbb{E}$ , the entries of the pair  $\langle A_i, A_j \rangle$  can be selected from two different distinct classes, and hence the matrix specifying the probabilities of the accesses of the pairs obeys Eq. (3), where:*

$$\underline{\theta} = \theta_o \cdot \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \cdots & \cdots & 1 \end{bmatrix}, \quad (4)$$

$$\mathcal{M}_r^* = \theta_d \cdot \begin{bmatrix} 0 & 1 & \cdots & 1 \\ 1 & 0 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 0 \end{bmatrix}, \quad (5)$$

where,  $0 < \theta_d < 1$  is the coefficient which specifies the accuracy of  $\mathbb{E}$ , and  $\theta_o$  is related to  $\theta_d$  as  $\theta_d = \frac{1-\theta_o(W-\frac{W}{R})}{\frac{W}{R}-1}$ .

*Proof.* The proof of the theorem is omitted here and found in [11].

### 3.1 The Design and Results of the POMA

In a real world scenario, since  $\mathbb{E}$ 's true statistical model is unknown, the expressions in Eqs. (1) and (2) can only be estimated through observing a set of queries. In the presence of noise though, we need to devise a measurable quantity which makes the algorithm capable of recognizing divergent pairs.

Observe that whenever a real query  $\langle A_i, A_j \rangle$  appears, we will be able to obtain a simple ML estimate of how frequently  $A_i$  and  $A_j$  are accessed concurrently. Clearly, by virtue of the Law of Large Numbers, these underlying estimates will converge to the corresponding probabilities of  $\mathbb{E}$  actually containing the elements  $A_i$  and  $A_j$  in the same group. As the number of queries processed become larger, the quantities inside  $\mathcal{M}_i^*$  will become significantly larger than the quantities in each of the  $\underline{\theta}$  matrices. From the plot of these estimates [11], one will observe that the estimates corresponding to the matrix  $\mathcal{M}_i^*$  have much higher values than the off-diagonal entries. This implies that these off-diagonal entries represent *divergent* queries which move the objects away from their accurate partitions.

Intuitively, the pursuit concept for the OPP can be best presented by a matrix of size  $W \times W$  where every entry will capture the same statistical measure about

the stream of the input pairs. For the sake of simplicity, we use a simple averaging and denote this matrix by  $\mathcal{P}$ . Every block represents a pair and the height of the block is set to the frequency count of the reciprocal pair. To obtain the average frequency of each pair, we let the OMA iterate for a sufficient time, say  $J$  iterations, and at every incident we update the value of the matrix  $\mathcal{P}$  respectively. In this way, at the end of the  $J$ -th iteration, we have simply estimated the frequency of each pair. At this point, by observing the values of the matrix, the user can determine an appropriate threshold ( $\tau > 0$ ) to be adopted as the accept or reject policy for any future occurrence of this particular pair of objects. If we permit the algorithm to collect a large enough number of pairs, we see that  $\exists \theta^* \mid \forall \theta_o \leq \theta^*$ , and that  $\forall i, j : \mu_{i,j} \gg \theta_{i,j}^*$ .

If we utilize a user-defined threshold,  $\tau$ , (which is reasonably close to 0), we will be able to compare every estimate to  $\tau$  and make a meaningful decision about the identity of the query. In other words, by merely comparing the estimate to  $\tau$  we can determine whether a query pair  $\langle A_i, A_j \rangle$  should be processed, or quite simply, be ignored. This leads us to algorithm POMA on Page 74 of [11] in which every query which is inferred to be divergent is ignored. Otherwise, one invokes the Reward and Penalty functions of the original OMA algorithm. The issue of determining the parameters of the POMA algorithm are detailed in [11], and omitted here in the interest of space.

In the initial phase of the algorithm, the estimates for the queries are unavailable. Thus, it only makes sense to consider every single query and to process them using the OMA’s Reward and Penalty functions. Since the objects in each class are equally-likely to happen and the classes are equi-probable,  $k \geq \left[ \left( \frac{W}{R} \right)^2 - \frac{W}{R} \right] \times R$ , is chosen as the lower-bound of the number of iterations for any meaningful initialization.

We have compared our results with those presented in [5] and those reported for the original OMA for various values of  $R$  and  $W$ . The number of states in every action was set to 10, and the convergence was expected to have taken place as soon as all the objects in the POMA fell within the last two internal states. The results (specified using the same notation as in Table 1) obtained are outstanding and are summarized in Table 2. The simulation results are based on an ensemble of 100 runs with different uncertainty values, (i.e., values of  $p$ ).

To observe the efficiency of the POMA, consider an easy-to-learn Environment of 6 groups with 2 objects in each group and where  $p = 0.9$ . It took the OMA 599 iterations to converge. As opposed to this, the POMA converged in only 69 iterations, which represents a *ten-fold* improvement. On the other hand, given a difficult-to-learn Environment with 12 objects in 2 groups, the OMA needs 6,506 iterations to converge. The POMA required only 2,112 iterations to converge, which is more than a *three-fold* improvement.

**Table 2.** Experimental results for the POMA approach done for an ensemble of 100 runs.

$W$	$W/R$	$R$	POMAp9	POMAp8	POMAp7
4	2	2	(2, 25)	(3, 30)	(3, 38)
6	2	3	(4, 44)	(4, 52)	(5, 67)
-	3	2	(20, 65)	(22, 77)	(24, 106)
9	3	3	(44, 105)	(70, 148)	(85, 169)
12	2	6	(10, 88)	(12, 103)	(20, 173)
-	3	4	(77, 166)	(105, 205)	(292, 462)
-	4	3	(328, 417)	(228, 372)	(202, 487)
-	6	2	(1563, 1836)	(945, 1091)	(1088, 1395)
15	3	5	(112, 213)	(142, 274)	(179, 315)
-	5	3	(1534, 1655)	(766, 998)	(556, 931)
18	2	9	(20, 151)	(26, 161)	(29, 566)
-	3	6	(245, 410)	(198, 417)	(226, 395)
-	6	3	(3146, 3270)	(2182, 2371)	(1145, 1542)
-	9	2	(5500, 5621)	(5064, 5523)	(4104, 4711)

## 4 Enhanced OMA (EOMA)

The learning of an enhanced LA proposed by Gale *et al.* [5], is based on the same principles of the OMA and leads to the Enhanced OMA (EOMA). They introduced three enhancements to improve its efficiency and speed as below:

- Initial Boundary State Distribution:** All of the objects are initially distributed at the respective boundary states of their respective classes;
- Redefinition of Internal States:** They diminished the vulnerability of the *convergence criterion* of the OMA by redefining the internal state to include “the two innermost states of each class”, rather than a single innermost state.
- Breaking the Deadlock:** The original OMA possesses a deadlock-prone infirmity (please see Section 4.3 of [11]) in which the machine can cycle between two identical configurations by virtue of a sequence of query pairs. Thus is especially evident in noise-free Environments. The EOMA remedies this as follows. Give a query pair of objects  $\langle O_i, O_j \rangle$ , let us assume that  $O_i$ , is in the boundary state, and  $O_j$  is in a non-boundary (internal) state of another class. If there exists an object in the boundary state of the same class, we propose that it gets swapped with the boundary object  $O_j$  to bring both of the queried objects together in the same class. Simultaneously, a non-boundary object has to be moved toward the boundary state of its class. Otherwise, if there is no object in the boundary state of the class that contained  $O_j$ , the algorithm performs identically to the OMA.

The simulation results obtained by introducing all of the three above-mentioned modifications are given in Table 3. In this table, we have reported the results from various Environments, with probabilities  $p = 0.9, 0.8$  and  $0.7$ , where  $p = 0.9$  is the near-optimal Environment. Such an Environment is easy to learn from. On the other hand, for the case where  $p = 0.7$ , we encounter a difficult-to-learn scenario. Our results have also compared our own implementation of the OMA algorithm described in [5] with the EOMA’s simulation results. These are the results displayed in Table 3. All the simulations reported were done on an ensemble of 100 experiments to guarantee statistically stable results. From the table we clearly see that as the value of  $p$  increases, the queries are more informative, and the convergence occurs at a faster rate. As before, the complexity of the classification problem has two criteria which are both observable in the tables, i.e., the number of objects in every group,  $\frac{W}{R}$ , and the number of groups,  $R$ . As the number of objects and groups increase, the problem becomes increasingly complex to solve. The reader will easily observe the advantages gleaned by the above three modifications in the EOMA, by comparing Tables 1 and 3.

**Table 3.** Experimental results for the *Enhanced* OMA (EOMA) done for an ensemble of 100 runs.

$W$	$W/R$	$R$	EOMAp9	EOMAp8	EOMAp7
4	2	2	(2, 26)	(2, 30)	(3, 60)
6	2	3	(4, 46)	(4, 65)	(5, 106)
-	3	2	(6, 50)	(8, 74)	(11, 127)
8	2	4	(6, 64)	(7, 95)	(8, 158)
-	4	2	(14, 75)	(20, 110)	(32, 185)
9	3	3	(18, 91)	(24, 132)	(35, 233)
10	5	2	(8, 85)	(10, 118)	(13, 226)
-	2	5	(25, 106)	(33, 153)	(70, 277)
12	2	6	(10, 102)	(12, 154)	(17, 291)
-	3	4	(43, 136)	(56, 207)	(81, 380)
-	4	3	(54, 150)	(66, 196)	(99, 388)
-	6	2	(40, 133)	(64, 208)	(105, 405)
15	3	5	(65, 187)	(92, 284)	(134, 554)
-	5	3	(75, 191)	(108, 295)	(192, 617)
18	2	9	(19, 170)	(26, 253)	(36, 630)
-	3	6	(106, 258)	(140, 389)	(242, 827)
-	6	3	(114, 255)	(167, 392)	(261, 857)
-	9	2	(112, 246)	(142, 363)	(311, 854)

The convergence of the EOMA with respect to time starts with a large number of objects which are located in random partitions. This number steadily

decreases with time to a very small value. This graph is not monotonic for any given experiment. But from the perspective of an ensemble, the performance is much more monotonic in behavior.

## 5 Enhancing the EOMA with a Pursuit Paradigm

The methodology by which we incorporated the pursuit concept into the OMA required us to formally model noise-free and noisy queries in Sect. 3. However, this is the precise dilemma that the EOMA faces. On the one hand, it would be advantageous, from a partitioning perspective, to have a noise-free Environment. However, it is precisely such noise-free Environments that lead to deadlock situations. Consequently, an attempt to elevate a noisy Environment to become noise-free would only defeat the purpose by exaggerating deadlock scenarios. However, rather than seeking to make the Environment noise-free, we will again accept or reject queries from the incoming stream. To accomplish this, we again apply the same “Pursuit” paradigm, explained below for this specific setting.

To design the PEOMA, as before, we again incorporate the pursuit principle by estimating the Environment’s reward/penalty probabilities. This could, of course, be done based on either a ML or Bayesian methodology. As these estimates become more accurate, we force the LA to converge to the superior actions at the faster rate. In all brevity, the PEOMA utilizes the exact same Pursuit principles explained in Sect. 3.1. Essentially, the EOMA which mitigates the “deadlock” situation is now augmented with the Pursuit concept, and thus:

- The stream of queries is processed using an *estimation* phase;
- The divergent queries are filtered using a *thresholding* phase, that serves as a filter for the above estimates;
- The deadlock scenarios are resolved using the enhancements of the EOMA over the OMA;
- The convergence criterion of making the two most internal states of every group to report convergence, makes the entire process converge even faster.

The experimental results compared the PEOMA with the EOMA, and we were able to show how the PEOMA out-performed the EOMA and the OMA. The results are given in Table 4 which uses the same notation and for the same settings as in the Tables 1 and 2. One can also compare its performance with the results presented in [5] and those reported in Table 3 for various values of  $R$  and  $W$  and in different Environments. The number of states in every action was set to be 10 as in Table 4. The convergence condition was also identical to the one specified in Sect. 4, and was assumed to have taken place when all the objects in the PEOMA fell within the last two internal states. Further, the query probability approximations were updated after receiving every single query.

The performance significance of the PEOMA is, really, not noticeable for easy problems where we had a small number of objects and groups, and where the noise level was low. But this becomes invaluable when we encounter a large number of actions as well as a stream of divergent queries (when  $p$  is “smaller”)

throughout the simulation, especially if we factor in the number of iterations used to obtain an estimate of  $\tau$ . Indeed, the PEOMA’s performance can be up to more than *two* times better than the EOMA. But if we compare the results with the original OMA, the immense performance gain leads to about *forty* times less number of iterations for a complete convergence – which is by no means insignificant. It is fascinating to note that the reduction in the number of iterations required by the PEOMA can again be seen to be a consequence of a Pursuit-like filtering phase in all problem domains.

**Table 4.** Experimental results for the PEOMA approach done for an ensemble of 100 runs.

$W$	$W/R$	$R$	PEOMAp9	PEOMAp8	PEOMAp7
4	2	2	(2, 23)	(2, 37)	(3, 44)
6	2	3	(4, 42)	(4, 52)	(5, 73)
-	3	2	(7, 47)	(8, 62)	(10, 91)
8	2	4	(6, 59)	(6, 76)	(8, 102)
-	4	2	(15, 73)	(23, 100)	(36, 145)
9	3	3	(20, 85)	(24, 110)	(40, 146)
10	2	5	(8, 79)	(10, 102)	(12, 141)
-	5	2	(26, 100)	(36, 140)	(54, 213)
12	2	6	(10, 97)	(12, 129)	(17, 181)
-	3	4	(38, 126)	(55, 165)	(74, 222)
-	4	3	(44, 134)	(58, 165)	(87, 241)
-	6	2	(34, 127)	(60, 182)	(110, 310)
15	3	5	(72, 174)	(88, 228)	(147, 308)
-	5	3	(76, 185)	(105, 249)	(155, 348)
18	2	9	(19, 166)	(26, 218)	(36, 323)
-	3	6	(98, 231)	(139, 310)	(207, 419)
-	6	3	(118, 246)	(162, 328)	(239, 472)
-	9	2	(100, 236)	(133, 330)	(280, 553)

## 6 Cohesiveness in the EPP: The Transitive PEOMA

The first issue that we encounter when we want to advance the field of resolving the EPP is to see if we can use new criteria to identify which objects belong to the same partition. We intend to investigate how this can be inferred without considering the issues that have been analyzed earlier. It is easy to see that all the objects within an underlying partition should be strongly and directly related to each other, and that they should frequently co-appear in the queries. Such

structural patterns are, in turn, based on so-called causal propositions which should lead towards relational “interactions” between the objects themselves. This is the avenue that we now investigate.

Structural relations that are imposed by the Environment can orient the objects towards a uniformity when there is an “interaction” between a pair of objects. Such relations may be “transmitted” through intermediaries even when two objects are not explicitly examined at any given time instant. This interconnection is directly associated with the relational bonds that these objects possess. We shall now investigate whether this property, which already relates *subgroups* and not just pairs, can be quantified by various specific properties that can be extracted from the Environment. They can be seen to be:

1. The frequency of objects co-occurring;
2. The relative frequency of the objects in a pair belonging to distinct partitions;
3. The symmetric property of the queries in any pair presented by  $\mathbb{E}$ ;
4. The reachability of the objects in a partition within the graph representing the set of all objects.

We first formalize the partitioning problem’s symmetry and transitivity properties proven in [11].

**Theorem 3.** *The model of  $\mathbb{E}$  and the solution invoked by any pursuit-based paradigm of the EPP possess the property of symmetry.*

**Theorem 4.** *The model of  $\mathbb{E}$  proposed for the EPP possesses the property of transitivity from a probabilistic perspective.*

Since  $\mathbb{E}$  is transitive, our aim is now to have the LA infer this transitivity and to further enhance the PEOMA. Indeed, if the pursuit matrix is appropriately thresholded, the entries become unity and zero, which allows us to demonstrate transitivity and thus, invoke reward/penalty operations even while the environment is dormant and not generating any new queries. Without going into the explicit details (omitted due to space limitations), this is essentially done by invoking the assertion:  $\forall O_i, O_j, O_k \in W : (O_i \mathcal{R} O_j \wedge O_j \mathcal{R} O_k) \implies O_i \mathcal{R} O_k$ . This leads us to the Transitive PEOMA (TPEOMA). The experimental results for the PEOMA are given in Table 5 for the same settings as in the previous tables.

By way of example, if the TPEOMA is compared with the previously best-reported algorithm, the PEOMA reported in Sect. 5, one can see that the PEOMA can solve the partitioning problem with  $p = 0.9$  and 3 groups with 3 objects in each group, in 85 iterations. For the same problem, the TPEOMA required only 65 iterations to converge. For a difficult-to-learn Environment ( $p = 0.7$ ) and a more complex partitioning problem with 18 objects in 3 groups, the PEOMA needed 472 iterations to converge. The TPEOMA required only 244 iterations to converge, which is nearly *two times* better than the PEOMA. It is certainly the fastest partitioning algorithm reported to date, and its behavior is monotonically decreasing for an ensemble of many experiments. The reader should observe the considerable performance that is gained by a very little additional computational cost. Again, by comparing Tables 4 and 5, one observes that

although the gain is not significant for simple problems and easy Environments, it becomes remarkably high for complex partitioning experiments.

**Table 5.** Experimental results for the TPEOMA approach done for an ensemble of 100 runs.

<i>W</i>	<i>W/R</i>	<i>R</i>	TPEOMAp9	TPEOMAp8	TPEOMAp7
4	2	2	(2,24)	(2,30)	(3,40)
6	2	3	(4,41)	(4,51)	(5,64)
-	3	2	(6,37)	(8,50)	(13,74)
8	2	4	(7,57)	(7,71)	(8,91)
-	4	2	(14,50)	(25,78)	(41,125)
9	3	3	(19,65)	(21,78)	(29,113)
10	2	5	(8,75)	(10,95)	(14,121)
-	5	2	(26,69)	(41,92)	(76,178)
12	2	6	(12,95)	(15,123)	(18,155)
-	3	4	(30,91)	(37,110)	(52,155)
-	4	3	(34,86)	(47,107)	(66,157)
-	6	2	(43,86)	(62,121)	(111,209)
15	3	5	(48,123)	(61,159)	(81,203)
-	5	3	(51,101)	(71,133)	(105,205)
18	2	9	(20,156)	(28,199)	(36,275)
-	3	6	(66,153)	(85,194)	(126,283)
-	6	3	(63,126)	(95,170)	(136,244)
-	9	2	(77,129)	(148,222)	(268,391)

## 7 Conclusions

In this paper we have shown how we can utilize the “Pursuit” concept to enhance solutions to the general problem of partitioning. Unlike traditional Learning Automata (LA), which work with the understanding that the actions are chosen purely based on the “state” in which the machine is, the “Pursuit” concept has been used to estimate the Random Environment’s (RE’s) reward probabilities and to take these into consideration to design Estimator/Pursuit LA. They, utilize “cheap” estimates of the Environment’s reward probabilities to make them converge by an order of magnitude faster. This is achieved by using inexpensive estimates of the reward probabilities to rank the actions. Thereafter, when the action probability vector has to be updated, it is done not on the basis of the Environment’s response alone, but also based on the ranking of these estimates.

In this paper we have shown how the “Pursuit” learning paradigm can be and has been used in Object Partitioning. The results demonstrate that incorporating this paradigm can hasten the partitioning by a order of magnitude. This paper comprehensively describes all the Object Migration Automaton (OMA)-related machines to date, including the Enhanced OMA [5]. It then incorporates the Pursuit paradigm to yield the Pursuit OMA (POMA), the Pursuit Enhanced OMA (PEOMA) and the Pursuit Transitive Enhanced OMA (PTOMA).

Apart from the schemes themselves, the papers reports the experimental results that have been obtained by testing them on benchmark environments.

## References

1. Godsil, C., Royle, G.F.: Algebraic Graph Theory, vol. 207. Springer, New York (2013)
2. Biggs, N.: Algebraic Graph Theory. Cambridge University Press, Cambridge (1993)
3. Fayyoumi, E., Oommen, B.J.: Achieving microaggregation for secure statistical databases using fixed-structure partitioning-based learning automata. *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)* **39**(5), 1192–1205 (2009)
4. Freuder, E.C.: The object partition problem. *Vision Flash, WP-(4)* (1971)
5. Gale, W., Das, S., Yu, C.T.: Improvements to an algorithm for equipartitioning. *IEEE Trans. Comput.* **39**(5), 706–710 (1990)
6. Jobava, A.: Intelligent traffic-aware consolidation of virtual machines in a data center. Master’s thesis, University of Oslo (2015)
7. Lanctot, J.K., Oommen, B.J.: Discretized estimator learning automata. *IEEE Trans. Syst. Man Cybern.* **22**(6), 1473–1483 (1992)
8. Mamaghani, A.S., Mahi, M., Meybodi, M.: A learning automaton based approach for data fragments allocation in distributed database systems. In: 2010 IEEE 10th International Conference on Computer and Information Technology (CIT), pp. 8–12. IEEE (2010)
9. Oommen, B.J., Ma, D.C.Y.: Stochastic automata solutions to the object partitioning problem. Carleton University, School of Computer Science (1986)
10. Oommen, B.J., Agache, M.: Continuous and discretized pursuit learning schemes: various algorithms and their comparison. *IEEE Trans. Syst. Man Cybern. Part B: Cybern.* **31**(3), 277–287 (2001)
11. Shirvani, A.: Novel solutions and applications of the object partitioning problem. Ph.D. thesis, Carleton University, Ottawa, Canada (2018)
12. Yazidi, A., Granmo, O.C., Oommen, B.J.: Service selection in stochastic environments: a learning-automaton based solution. *Appl. Intell.* **36**(3), 617–637 (2012)
13. Amer, A., Oommen, B.J.: A novel framework for self-organizing lists in environments with locality of reference: lists-on-lists. *Comput. J.* **50**(2), 186–196 (2007)

# **AI in Energy Management - Industrial Applications**



# A Benchmark Framework to Evaluate Energy Disaggregation Solutions

Nikolaos Symeonidis, Christoforos Nalmpantis<sup>(✉)</sup>, and Dimitris Vrakas

School of Informatics, Aristotle University of Thessaloniki,

54124 Thessaloniki, Greece

{symeonidn, christofn, dvvakas}@csd.auth.gr

<https://www.csd.auth.gr/en/>

**Abstract.** Energy Disaggregation is the task of decomposing a single meter aggregate energy reading into its appliance level subcomponents. The recent growth of interest in this field has lead to development of many different techniques, among which Artificial Neural Networks have shown remarkable results. In this paper we propose a categorization of experiments that should serve as a benchmark, along with a baseline of results, to efficiently evaluate the most important aspects for this task. Furthermore, using this benchmark we investigate the application of Stacking on five popular ANNs. The models are compared on three metrics and show that Stacking can help improve or ensure performance in certain cases, especially on 2-state devices.

**Keywords:** NILM · Energy disaggregation · Artificial neural networks · Stacked learning · Benchmark

## 1 Introduction

In the modern world, one of the biggest problems humanity is facing is the inadequate management of electrical energy. Overconsumption causes a series of negative phenomena that have both economic and ecological impact, at individual and mass scale. Numerous factors affect this problem, such as the dramatic increase in usage of electrical devices in recent decades, as well as the global population growth. It is apparent that extensive research is required to have better methods of controlling electrical energy consumption. The existence and application of smart meters already contribute in that regard. Energy disaggregation can make further use of those, to enhance load monitoring capabilities.

Energy disaggregation, also known as Non-intrusive load monitoring (NILM), is the task of decomposing an aggregate energy signal into its sub-components, i.e.

---

This work has been funded by the ΕΣΠΑ (2014-2020) Erevno-Dimiourgo-Kainotomo 2018/EPAnEK Program ‘Energy Controlling Voice Enabled Intelligent Smart Home Ecosystem’, General Secretariat for Research and Technology, Ministry of Education, Research and Religious Affairs.

identifying the individual appliances signal from the whole energy consumption of a house. It was first introduced by Hart [4]. By applying NILM methods load monitoring becomes easier and less costly, due to the requirement of only a single meter from which the device level signals can be extracted, in contrast to ILM (intrusive load monitoring) methods where multiple meters are needed per house. Therefore ILM has much bigger economical cost and increased difficulty from installing and configuring the meters, its only advantage being its guaranteed higher accuracy.

Many researchers focus on finding solutions where a model is trained per appliance, having as input the whole house energy data and as output the appliance consumption. Each work has its own test cases defined and investigates a set of metrics upon them. Due to this, many different possible scenarios are created, which complicates the comparison between the proposed and existing solutions. There is a lack of structure to follow when conducting the evaluation of a new model. This creates a necessity for a well-defined set of experiments. A set like this should include a variety of scenarios that cover a wide range of aspects and goals for the model under evaluation. Each scenario is tied to a specific purpose, that reveals if the model is suitable for the case.

In this paper we propose a benchmark framework for the evaluation of NILM solutions. Also five ANNs are combined with the method of Stacking and then evaluated with the proposed framework. This paper is structured as follows: in Sect. 2 some of the most popular neural network solutions, that are important for this study, are reviewed. In Sect. 3 the proposed categorization of experiments is described. Section 4 expands upon the method of Stacking. Section 5 presents the most important results produced from Stacked learning. Section 6 includes conclusions from the experiments and discussion for future work. The implementation of Stacking and the five used ANNs, a detailed spreadsheet containing the defined experiments for each category, as well as baseline results can be found in the following repository <https://github.com/symeonick15/NILM-Stacking>.

## 2 Related Work

There have been many approaches to solving this problem, among which Machine Learning (ML) has taken the lead of research in recent years. ML methods exhibit great generalization capability on unseen environments, without the need of prior information. Initially, Hart proposed a combinatorial optimization method, which suffered from working only with devices that had a finite number of states. Later Factorial Hidden Markov Models (FHMM) have become quite popular and many developed techniques were based on them [1, 7, 15, 19].

The study around NILM has recently turned towards implementing solutions based on artificial neural networks. Deep and convolutional neural networks have become dominant in many fields like Computer Vision [9] and Natural Language Processing [3], and have been used successfully in many problems that include time series data. Their ability to extract features and handling complex data has lead researchers to start developing NILM solutions based on them, outperforming former approaches [2, 5, 10–12, 14, 18].

Kelly and Knottenbelt [5] developed three ANN architectures for the task of Energy Disaggregation. The first was a Denoising Auto Encoder that handles the aggregate signal as a “noisy” series, which filters out the noise (i.e. signals from other devices) to extract the target appliance consumption. The 2nd was a Recurrent Neural Network that used LSTM units (long short-term memory) and had the ability to “remember” previously given inputs to use them for prediction. The last architecture would find the Start time, End time and mean consumption of the first activation in a given window. All three were trained on the UK-DALE dataset, having as input the whole-house aggregate signal and as target the appliance consumption. An FHMM approach and Hart’s combinatorial optimization algorithm were also used for the same experiments. In comparison, ANNs outperformed these two methods.

In another study, Mauch and Yang [12] implemented a different architecture with LSTM units for the Energy Disaggregation task. The proposed deep recurrent network had as input the aggregate energy value at a specific timestamp and as output the appliance consumption at the same timestamp. Among the goals of this network was to automatically extract features from low-frequency data and to generalize well on unseen buildings. REDD dataset was used for train and prediction, for two ON/OFF and one multistate devices. Results were promising both on seen and unseen buildings.

Zhang et al. [18] proposed a different deep convolutional architecture that they named sequence to point. The method took its name from the main idea to predict the value of a single time point, based on a sequence of values in the input that has the time point at its midpoint. The input window used had 600 samples (1 h). The network achieved state of the art results and had great representation power, as it would base its predictions both on the past and the future of a time point.

Krystalakos et al. [10] used Gated Recurrent Units during their experiments to improve current LSTM architectures. To decrease the computational cost and memory demands, LSTM neurons were replaced with GRU, fewer neurons were used and dropout layers were added. Furthermore, a sliding window approach was tested. The network was trained and tested on UK-DALE and compared directly to implementations based on previous architectures, among which a modified seq2point more suitable for online prediction (smaller input windows). The proposed architecture showed promising results, especially on multi-state devices, and it had the same or better results than LSTM while being lighter.

Although there wasn’t a benchmark that was actually followed by these studies to base their experiments, many have used the same or similar structure presented in the work of Kelly and Knottenbelt [5]. Also, most had similarities in their tests, such as having the trained model predicting in an unseen house, aiming to evaluate specific aspects of the models.

In a review of ML approaches for NILM by Nalmpantis and Vrakas [13], it is stated that an objective and direct comparison between different methods is very difficult. The reason is that there are various metrics, many available datasets, different criterias and a variety of methodologies that one can choose

when evaluating NILM solutions. A qualitative and a quantitative analysis is presented with methods for evaluation. Among these are Zeifman's requirements [17], as well "generalization" and "privacy" requirements, along with metrics for each. Also a measure of disaggregation complexity is defined, to evaluate the different environments.

### 3 Purpose

In this paper, we present a categorization of experiments with the purpose to have a unified way of comparing models during research. The scenarios of the proposed categorization include specific train and test cases, each with an explained purpose that explores an aspect of the model, plus a basic set of metrics to evaluate the performance, as both classification and regression. Also, as a reference point it can be further expanded with more cases or have its existing modified, to accommodate additional scenarios, should it be required in a future research, where a more specific goal is examined (e.g. commercial buildings). The proposed benchmark is defined on a set of five appliances, however, the scenarios included can easily be generalized for any device, or even different datasets.

Moreover, the effects of combining several neural networks with the method of Stacking are investigated, by comparing them using the aforementioned taxonomy. Stacking has been used widely in Machine Learning approaches to combine many different models in order to achieve better results than each model individually. The basic idea is to follow this method, in hope of improving the performance of existing neural network architectures, and to see the actual impact it has on them.

### 4 Taxonomy of Experiments

In this section, the proposed categorization of experiments is described, for usage in the evaluation of new Energy Disaggregation models. Specifically, this method is suitable for evaluating techniques (e.g. ANNs) that take an aggregate signal as input and predict a specific appliance consumption. There are four main categories of experiments described for this method and the purpose that each serves. Although these can be expanded and/or modified to suit the goals of individual studies, we also proceed to define the datasets and appliances used in this study.

The two datasets used in this study are Reference Energy Disaggregation Data Set (REDD) [8] and UK-DALE [6]. Both datasets are freely available, support low-frequency data, refer to domestic buildings and have several houses and appliances for testing. They are also two of the most popular datasets in the field of NILM. The target appliances are: fridge, kettle, microwave, washing machine, and dishwasher. These have been used by many researchers because they cover a wide range of appliance types a house may include. For example, the kettle is a simple ON/OFF device, while the dishwasher is multistate with more complex behavior. Furthermore, they constitute most of a building's consumption and appear in most of the houses making a better target for evaluation.

#### 4.1 Category 1: Single Building NILM

This is actually the most simple form of experiment, where training and prediction happen on the data of the same building, at different time ranges. This evaluates the performance of a model on an environment very similar to the one it was trained on. This is a vital prerequisite for a technique. If it has poor results here, it probably won't do better at other experiments as well. Of course, no house stays the same in reality, but that is also a part of the problem. The specific experiment defined in this study for this category includes training and testing on house 1 of UK-DALE. The test set is defined as the year following April 2016, while the rest of the data are available for training. House 1 has the most available data, thus making it suitable for both training and testing.

#### 4.2 Category 2: Single Building Learning and Generalization on Same Dataset

Several experiments may be mapped to this category, one per house. Training happens on one house of a selected dataset, while prediction is applied to the rest houses. The purpose of these experiments is to evaluate the ability of the model to generalize on relatively similar unseen houses when trained on data on one house. Although learning is restricted to only one house, this also shows if the trained model is overfitting on its data, which is another aspect to be evaluated. It needs to be noted that houses here are considered similar (e.g. same city/country), but in reality, even neighboring houses may be totally different in their energy patterns (different appliances, consumption patterns, etc.). House 1 of UK-DALE is selected as the training set here again, while the rest of the houses where the target appliance is present compose the test sets. The first house has a good amount of data for learning, while the rest have a few months each, so they are better for prediction.

#### 4.3 Category 3: Multi Building Learning and Generalization on Same Dataset

In this category, the model is trained with data from more than one, relatively similar houses (same dataset), while prediction is applied on data from a house that was not present during training. The main purpose here is again to evaluate the generalization ability of the method. The second aspect that is being assessed, is the ability of the technique to learn from multiple different sources and combine efficiently the knowledge extracted from them. A model that achieves this will theoretically perform better on new unknown data, due to the variety of data it has learned. This is a similar, but more complex task from the previous category, because a learning algorithm may get “confused”, by the variance of the data. The experiments used for this category are defined for the UK-DALE dataset and follow the same structure as the ones in the study of Kelly and Knottenbelt [5], for tests on unseen buildings. This way a comparison with existing studies becomes easier and more direct.

#### 4.4 Category 4: Generalization to Different Dataset

By expanding the previous experiment with the prediction on a different dataset, the task becomes even more difficult. The houses on which the model is tested are not similar (e.g. different countries) to the ones it was trained on and may present great differences in characteristics such as included appliances, appliance types, energy grid, consumption behavior, etc. So to perform well the model must possess strong generalization capabilities. The difficulty here can get great and unpredictable. However, it is an interesting query, that evaluates an important ability of a NILM solution. The training set is comprised of UK-DALE data, while testing is applied to REDD data. The first has buildings in the UK, while the second is for buildings in USA. The differences between these two can be apparent, making them a suitable choice for this category.

### 5 Artificial Neural Networks with Stacking

In this section, we investigate the application of ensemble learning to existing NILM solutions, to further increase their accuracy. The combined model is evaluated using the aforementioned benchmark method, by comparing it with the individual results of the models that were combined.

As recent research suggests that Artificial Neural Networks fare rather well in the task of Energy Disaggregation, 5 such networks have been selected as the base models for the ensemble. The selected networks are Denoising Auto-Encoder (DAE) [5], Recurrent Neural Network (RNN) [5], Short Seq-2-point (SS2P) [18], GRU Network (GRU) [10], GRU with sliding window (WGRU) [10]. All have shown promising results in previous research, where some were better than the others in certain situations. More info about those ANNs can be found in the Related Work section.

The following is a summary of the ANNs architectures. DAE had a convolutional layer, 3 fully connected layers and one more convolutional as the output, with dropout between them. GRU had 2 convolutional, 2 bidirectional GRU and 2 fully connected layers. RNN had 1 convolutional, 2 bidirectional LSTM and 2 fully connected layers. WGRU had 1 convolutional, 2 bidirectional GRU and 2 fully connected layers, with dropout between the last 4. SS2P had 5 convolutional layers and 2 fully connected ones, with dropout between them. More details can be found at the provided code on github.

#### 5.1 Introduction to Stacking

The basic idea behind stacking is combining several different learners, to achieve better results than each of the base models would achieve individually. It is especially efficient when base learners make different errors. Each algorithm learns a part of the problem, while the final combined model is able to learn a greater space. To achieve this, the training set should further be split into 2 separate parts. The first part, usually much bigger than the other, is used to train each of

the base models. After each model is trained, the second part is given as an input to each model to get their predictions. The predictions generated are then combined into one matrix that will make up the training input of a different learner (usually simpler), the meta-learner, while the target output is left as is (from the second part). That way the stacked model learns from the predictions of the base models, so it can combine them. After that, the stacked model is ready for prediction. The prediction follows a similar procedure, where each base model is given a copy of the input and then their predictions are given as an input to the meta-model to generate the final prediction. Stacking is especially efficient when base learners make different errors.

## 5.2 Implementation

The above procedure was implemented as a 2-step method for the stacking experiments. During the first step, each of the neural networks was trained on a part of the training set. Each trained model was also given the second part of the train set and the test set as input for prediction, to generate the train and test set of the stacked model respectively. The predictions were saved as intermediate files to be reused. In the second step, the prediction for the stack train was loaded and aligned on their timestamps. Then they were scaled and used to fit the selected meta-regressor. In the final phase, the predictions on the test set were loaded in the same way and given as input to the meta-model to generate the final predictions.

The sampling ratio for all data was 6 seconds. Only real data were used, with no synthetic data generation. Code is written in Python. The implementation of the used networks was based on a previous study of Krystalakos et al. [10], which were developed using Keras with Tensorflow backend on GPUs. NILMTK was used for loading and preprocessing of data during the base model training and stacking phase. Scikit-learn was used for the Meta-regressors.

The metrics used for the evaluation were F1, Relative Error in Total Energy (RETE) and Mean Absolute Error (MAE).

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (1)$$

$$RETE = \frac{|E' - E|}{\max(E', E)} \quad (2)$$

$$MAE = \frac{1}{T} \sum |y'_t - y_t| \quad (3)$$

Where  $E'$  is the total predicted energy,  $E$  is the total true energy,  $y'_t$  is the consumption predicted at time  $t$  and  $y_t$  is the true consumption at time  $t$ .

## 6 Results and Discussion

In this section, we present some of the most important results that were observed during the experiments. A complete set of results can be found in the supplied

repository and was not presented here due to its size. Some results are highlighted to indicate that they were the best among those that were tested (best among base and among stacked). The following are the meta-regressors referenced from the result matrices. Ada Boost with Decision Tree of depth 3, 25 estimators, learning rate 0.1 and ‘square’ loss (AB-3d). Ada Boost with Decision Tree, 30 estimators and learning rate 0.5 (AB-30). Ada Boost with Decision Tree, 15 estimators and learning rate 0.5 (AB-15). Multi Layer Perceptron with one hidden layer of 100 neurons (MLP). Decision Tree Regressor of depth 5, 15% min split ratio, 9% min leaf ratio (DT5). Simple Decision Tree Regressor (DT). Gradient Boosting Regressor with 25 estimators and learning rate 0.5 (GB).

**Table 1.** Fridge, Category 1, Train and test on UK-DALE house 1.

MODEL	F1	RETE	MAE
DAE	0.637	0.224	35.81
GRU	<b>0.673</b>	<b>0.130</b>	<b>34.03</b>
RNN	0.662	0.270	34.69
SS2P	0.651	0.215	35.23
WGRU	0.641	0.224	34.30
AB-3d	<b>0.674</b>	0.163	33.54
AB-30	0.635	<b>0.017</b>	<b>26.56</b>

**Table 2.** Fridge, Category 3, Train on houses 1, 2, 4 and test on 5 of UK-DALE.

MODEL	F1	RETE	MAE
DAE	0.514	0.176	<b>47.17</b>
GRU	nan	0.296	53.87
RNN	nan	0.318	53.92
SS2P	nan	<b>0.023</b>	49.18
WGRU	<b>0.569</b>	0.243	51.85
DT5	<b>0.641</b>	<b>0.221</b>	46.71
AB-30	0.525	0.225	<b>44.43</b>

As it can be seen among the ANNs, some results in F1 score are ‘nan’. In those cases the predictions of the model were never above the activation threshold, leading to division by zero. This happens mostly on generalization tasks, proving the difficulty of disaggregating the signal of an unseen building. On the other hand stacking seems to have the advantage of overcoming this problem.

Results for Category 1 experiments of fridge are shown at Table 1. Among base models, GRU was a clear winner. Stacking with AB-3d mostly improved F1, while AB-30 had very good RETE and MAE. AB-3d had short trees, so it could not be as accurate, but managed to hit the activation threshold better. AB-30 has predictions closer to the ground truth values, as can be seen from the Fig. 1, but seems a bit “unstable” (many spikes where it should have continuous values). Maybe by applying a smoothing technique on top of it, it could be further improved. Generally stacking has good results, especially with AB-30, which enhances regression efficiency.

Table 2 shows the results of category 3 experiments for fridge. Here the best RETE was not improved, however, it’s still better than 3 out of 5 of base models. MAE is reduced, while DT5 also has very good F1. As above the short tree (DT5) is better suited for classification, while also is less prone to overfitting. In

general, tree-based models seem to work better with the fridge, probably due to the simple, repetitive nature of its time series.

**Table 3.** Kettle, Category 1, Train and test on UK-DALE house 1.

MODEL	F1	RETE	MAE
DAE	0.496	0.250	15.47
GRU	0.301	0.536	32.00
RNN	0.304	0.461	28.12
SS2P	0.322	<b>0.110</b>	19.95
WGRU	<b>0.582</b>	0.192	<b>10.78</b>
DT	<b>0.740</b>	<b>0.130</b>	<b>8.11</b>
AB-15	<b>0.804</b>	<b>0.161</b>	<b>6.37</b>

**Table 4.** Kettle, Category 3, Train on houses 1, 2, 3, 4 and test on 5 of UK-DALE.

MODEL	F1	RETE	MAE
DAE	0.504	<b>0.055</b>	10.71
GRU	0.104	0.383	24.48
RNN	0.250	0.419	43.58
SS2P	nan	0.593	11.02
WGRU	<b>0.663</b>	0.122	<b>10.16</b>
DT	0.566	0.098	<b>9.98</b>
GB	0.583	<b>0.033</b>	10.44

Table 3 has the results of Category 1 experiments for kettle. F1 is greatly increased with the application of stacking here, especially with AB-15 which also greatly improves MAE metric and has better RETE than 3/5 of base models. However even best RETE from stacking was worse than best from base models. Again trees were the best combiners. In category 2 though, stacking did not do so well. Probably because kettle is a relatively simple device and stacking forces the final model to focus even more on house 1, it is easier to overfit.

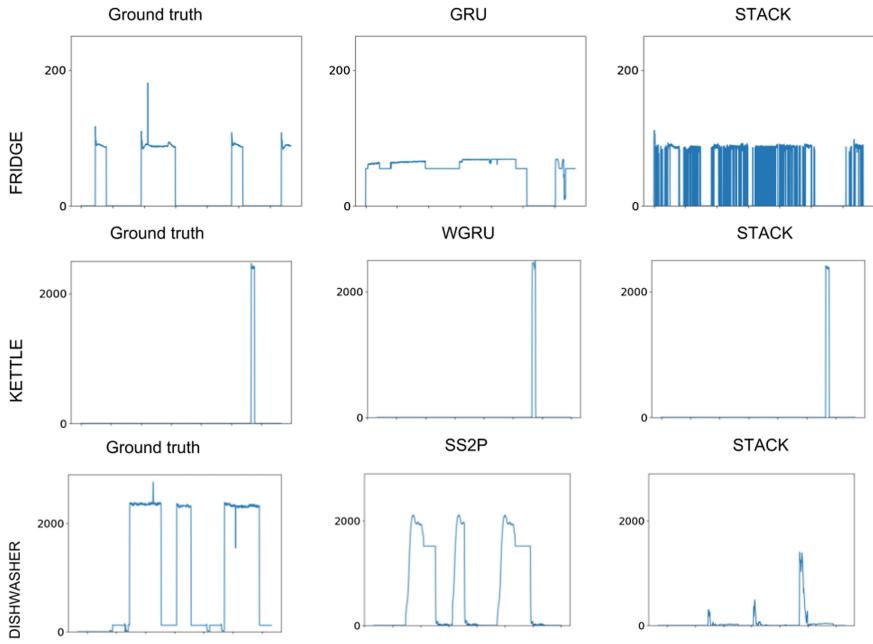
Results of category 3 experiments for kettle are shown at Table 4. Among neural networks DAE and WGRU had the best results. With stacking, GB managed to have the best RETE, about the same MAE and 2nd best F1 after WGRU. It was a fine point between the strong points of the best base models. DT was similar, with little worse on F1 and RETE, but the best MAE overall. A simple tree is again among the best, because of the simplicity of the device's behaviour. Also boosting is susceptible to overfitting, risking generalization capabilities.

**Table 5.** Dishwasher, Category 1, Train and test on UK-DALE house 1.

MODEL	F1	RETE	MAE
DAE	0.109	<b>0.001</b>	43.61
GRU	0.471	0.140	37.06
RNN	0.467	0.072	38.54
SS2P	<b>0.550</b>	0.047	<b>31.01</b>
WGRU	0.468	0.332	31.22
MLP	<b>0.571</b>	0.195	29.46

**Table 6.** Washing machine, Category 3, Train on houses 1, 5 and test on 2 of UK-DALE.

MODEL	F1	RETE	MAE
DAE	0.128	<b>0.566</b>	28.18
GRU	0.156	0.601	32.05
RNN	nan	0.679	33.21
SS2P	0.174	0.745	40.27
WGRU	<b>0.302</b>	<b>0.568</b>	<b>10.55</b>
AB-30	<b>0.324</b>	0.136	12.07



**Fig. 1.** Sample signal plots including: original, predicted from ANN, predicted from Stacking.

Dishwasher category 1 results can be found at Table 5. Stacking here seems to fail at the score of RETE. On the other hand MLP improves the best F1 and the best MAE. AB-3d had an even better MAE, but lost even more on the metrics of F1 and RETE. This time it was not trees, but MLP with 100 neurons that had the best results, suggesting that a more complex model is required for devices such as dishwasher, which have multiple states and complicated behaviour that is time-dependent. Category 2 experiments of the dishwasher had mixed results: F1 would be improved, while the other metrics varied, depending on the house and meta-regressor. In Categories 3 and 4 stacking did not succeed, indicating the requirement of a more appropriate technique.

Results of washing machine category 3 are shown at Table 6. Among the experiments of washing machine, some interesting results were found here. Most meta-models had a great improvement of RETE. AB-30 managed to even increase best F1 score, while keeping a near best MAE.

## 7 Conclusions and Future Work

We have proposed a benchmark to evaluate Energy Disaggregation models and used it to evaluate the application of stacking on existing techniques. Through the experiments, many aspects of the proposed method were explored and stacking showed promising results.

Across the results it appeared that some models were more suited than others in different scenarios. The advantage of stacking was that it could either improve those, or at least find a fine point between them, making it a robust solution. The tested stacked models had good results mostly for disaggregating simple devices (fridge, kettle), especially on same house train-test scenarios. Mainly, Tree based models were the best combiners, while AdaBoosting could further enhance them with the risk of overfitting. This risk was made apparent in generalization experiments (Categories 2–4).

An example scenario that uses stacking could include a weak fast solution that produces online results, which is later combined with the output of other models to produce more accurate final predictions. For generalization tasks improvements seemed less and on complex devices stacking sometimes did not succeed, possibly due to their complicated behaviour that is time dependant, along with the number of states and functions they have. This suggests that other meta-regressors may be more suited, possibly more complicated, time series oriented techniques like Neural Nets, that also have better generalization abilities. Another form of ensemble learning or stacking would also be interesting to test, like meta decision trees [16]. Maybe even another method on top of that could be used to smooth/filter the predicted signal. Regarding the proposed benchmark and the categories of experiments defined, there could also be some other similar scenarios not included here. For example one could have a training set combined from the 2 used datasets (UK-DALE, REDD) and test on both of them or even a third.

## References

1. Aiad, M., Lee, P.H.: Non-intrusive load disaggregation with adaptive estimations of devices main power effects and two-way interactions. *Energy Build.* **130**, 131–139 (2016). <https://doi.org/10.1016/j.enbuild.2016.08.050>. <http://www.sciencedirect.com/science/article/pii/S0378778816307472>
2. Chen, K., Wang, Q., He, Z., Chen, K., Hu, J., He, J.: Convolutional sequence to sequence non-intrusive load monitoring. *J. Eng.* **2018**(17), 1860–1864 (2018). <https://doi.org/10.1049/joe.2018.8352>
3. Graves, A.: Generating sequences with recurrent neural networks. arXiv preprint [arXiv:1308.0850](https://arxiv.org/abs/1308.0850) (2013)
4. Hart, G.W.: Nonintrusive appliance load monitoring. *Proc. IEEE* **80**(12), 1870–1891 (1992)
5. Kelly, J., Knottenbelt, W.: The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes. *Sci. Data* **2**, 150007 (2015). <https://doi.org/10.1038/sdata.2015.7>
6. Kelly, J., Knottenbelt, W.: The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes. *Sci. Data* **2**, 150007 (2015)
7. Kolter, J.Z., Jaakkola, T.: Approximate inference in additive factorial HMMs with application to energy disaggregation, June 2018. <https://doi.org/10.1184/R1/6603563.v1>

8. Kolter, J.Z., Johnson, M.J.: REDD: a public data set for energy disaggregation research. In: Workshop on Data Mining Applications in Sustainability (SIGKDD), San Diego, CA, vol. 25, pp. 59–62 (2011)
9. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
10. Krystalakos, O., Nalmpantis, C., Vrakas, D.: Sliding window approach for online energy disaggregation using artificial neural networks. In: Proceedings of the 10th Hellenic Conference on Artificial Intelligence, SETN 2018, pp. 7:1–7:6. ACM, New York (2018). <https://doi.org/10.1145/3200947.3201011>
11. Lange, H., Bergés, M.: The neural energy decoder: energy disaggregation by combining binary subcomponents (2016)
12. Mauch, L., Yang, B.: A new approach for supervised power disaggregation by using a deep recurrent LSTM network. In: 2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP), pp. 63–67. IEEE (2015)
13. Nalmpantis, C., Vrakas, D.: Machine learning approaches for non-intrusive load monitoring: from qualitative to quantitative comparation. Artif. Intell. Rev. 1–27 (2018)
14. Paradiso, F., Paganelli, F., Giuli, D., Capobianco, S.: Context-based energy disaggregation in smart homes. Future Internet **8**(1) (2016). <https://doi.org/10.3390/fi8010004>. <http://www.mdpi.com/1999-5903/8/1/4>
15. Parson, O., Ghosh, S., Weal, M., Rogers, A.: Non-intrusive load monitoring using prior models of general appliance types. In: Twenty-Sixth AAAI Conference on Artificial Intelligence (2012)
16. Todorovski, L., Džeroski, S.: Combining classifiers with meta decision trees. Mach. Learn. **50**(3), 223–249 (2003)
17. Zeifman, M.: Disaggregation of home energy display data using probabilistic approach. IEEE Trans. Consum. Electron. **58**(1), 23–31 (2012)
18. Zhang, C., Zhong, M., Wang, Z., Goddard, N., Sutton, C.: Sequence-to-point learning with neural networks for non-intrusive load monitoring. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
19. Zhong, M., Goddard, N., Sutton, C.: Signal aggregate constraints in additive factorial HMMs, with application to energy disaggregation. In: Advances in Neural Information Processing Systems, pp. 3590–3598 (2014)



# Application of Deep Learning Long Short-Term Memory in Energy Demand Forecasting

Nameer Al Khafaf<sup>(✉)</sup> , Mahdi Jalili , and Peter Sokolowski

Electrical and Biomedical Engineering, RMIT University, Melbourne, Australia  
nkhaaf@gmail.com,  
{mahdi.jalili,peter.sokolowski}@rmit.edu.au

**Abstract.** The smart metering infrastructure has changed how electricity is measured in both residential and industrial application. The large amount of data collected by smart meter per day provides a huge potential for analytics to support the operation of a smart grid, an example of which is energy demand forecasting. Short term energy forecasting can be used by utilities to assess if any forecasted peak energy demand would have an adverse effect on the power system transmission and distribution infrastructure. It can also help in load scheduling and demand side management. Many techniques have been proposed to forecast time series including Support Vector Machine, Artificial Neural Network and Deep Learning. In this work we use Long Short Term Memory architecture to forecast 3-day ahead energy demand across each month in the year. The results show that 3-day ahead demand can be accurately forecasted with a Mean Absolute Percentage Error of 3.15%. In addition to that, the paper proposes way to quantify the time as a feature to be used in the training phase which is shown to affect the network performance.

**Keywords:** Deep learning · Long Short-Term Memory · Demand forecasting

## 1 Introduction

The introduction of smart meters technology in recent years have shaped the metering infrastructure industry providing a smart, efficient and regular monitoring of energy consumption. Smart meters have been deployed in almost all residential and industrial applications around the world and in Australia giving the potential for metering intelligence and analytics [1, 2]. The smart meter usually captures aggregate of energy consumption in either a 15 or 30 min window creating a historic profile of energy consumption for each user.

Energy consumption is mainly driven by the energy consumer actions and behaviours which are affected by the consumer preferences. Since consumers' preferences are likely to change over time, this introduces uncertainties in the daily energy consumption pattern. Other factors influencing energy consumption include economic situations, climate change, holidays, working days, time periods and social and behavioural aspect [3]. In addition to that electricity demand is on the rise with

increased population and introduction of many appliances such as dish washer and cloth dryer into the household. Another contributor to the increase in energy consumption is climate change which contributes to a spike in energy consumption for approximately 40 h a year or 0.5% according to a report by Powercor Australia [4]. The supply side may or may not be able to handle the spike in demand if it has not been properly planned. One way to manage the spike in demand is to increase electrical assets such as generation, distribution and transmission infrastructure to accommodate the increase in energy consumption. However this increase in asset is not suitable as the cost of installing the electrical infrastructure far outweigh the benefits of meeting the spike in demand as it only happens about 0.5% of the year. Another way of managing the sudden increase in energy consumption is through energy management strategies applied at the demand side [5, 6]. In order to properly and effectively manage peak energy consumption in the short term, an accurate load forecasting is required.

Load forecasting is one of the most important analytics for the smart grid as it provides a prediction of what would be the likely energy demand in the future with a margin of error allowing for a timely decision making [7]. The purpose of demand forecasting depends on the prediction period such as short, medium or long term. Short, medium and long term forecasting are defined as being less than 1 week, between 1 week to one year and more than one year respectively [8]. Short term forecasting can be used as an input into demand side management framework to help better addressing high peak consumption due to specific events such as heat wave or blizzard depending on the season. Medium term forecasting is mostly used for load scheduling and maximizing power distribution and transmission asset utilization. Whereas long term forecasting focuses on identifying time period where the demand will be the lowest to plan for maintenance or shut down for upgrade. Long term is also used to plan for upgrading the power network due to a constant and permanent increase in demand that is not prone to seasonal or daily fluctuations [9].

There are many applications for load forecasting based on statistical and machine learning technologies that have been developed in the literature [10]; time series and artificial neural networks are most common techniques for short term and medium term forecasting [11–13]. Other short term forecasting are based on deep learning Long Short-Term Memory (LSTM) approach which has been proven in [14, 15] to be effective compared to traditional approach. LSTM has also been widely used and proven to be effective in short and medium term forecasting [16, 17].

A recent study by [18] to develop a high precision ANN for load forecasting (DeepEnergy) conclude that the proposed technique exceeds the traditional LSTM network in terms of the Mean Absolute Percentage Error (MAPE) for a 3-day ahead forecast. However in their implementation of LSTM, they did not consider other features to effectively train the LSTM network. Moreover they used data from two different months for training and data from a third month for forecasting. This may affect the prediction accuracy of LSTM as consumption can be different from one month to another. In another study conducted by [2] using dynamic neural network to load forecast which seems to be giving a good accuracy, however similar to the work by [18], the experiment does not consider other features to improve the accuracy of the model rather depend solely on the historic energy consumption.

In this paper we propose a LSTM deep learning to forecast energy demand for clusters of energy users in the short and medium term. The difference between our work and the work in the literature is in the way we pre-process the raw data for training and the use of two more features in addition to the historic energy consumption to improve the forecasting process. In terms of training the LSTM, most works in the literature uses past history of time series to forecast future ones, however in this work we propose a time feature curve to define unique time instance across the week. IN doing this the data during training can be shuffled to avoid over fitting. The first trial is to forecast 3-day ahead energy consumption in each month and the second trial is to forecast 15 days ahead energy consumption in a year. This work is anticipated to be an important step toward developing a LSTM model to accurately forecast peak energy consumption in the short and medium term. The model can be used by utilities to prepare for spike in energy and hence power demand during a heat wave.

## 2 Research Approach

The approach aims at forecasting a 3-day ahead energy consumption of clusters of energy users and accurately predicts peak consumption occurring during the 3 days prediction window. This will allow utilities to take actions to manage the spike in demand if the forecasted demand is higher than the capacity of the supply side. In addition to the 3-day ahead prediction, we attempt to forecast 15 days ahead energy consumption by training the LSTM on larger data. The LSTM we used has the architecture defined in [3] which consists of 3 gate units namely, input, output and forget gate to form one memory cell. Gates control the flow of the energy consumption time series inside the LSTM unit whereas the cell record dependencies between values of the time series. Each gate uses a sigmoid activation function  $\Phi_S$  whereas the cell state uses hyperbolic tangent activation function  $\Phi_T$ .

Given an input time series  $y_t$ , the LSTM has to learn the input weights  $I$ , the recurrent weights  $R$  and the bias  $b$  where  $I$ ,  $R$  and  $b$  are 4-by-1 column vectors with elements correspond to input, output and forget gate and the cell state. The input  $i$ , forget  $f$  and output  $o$  gates at time step  $t$  are written as:

$$i_t = \Phi_S(I_i x_t + R_i h_{t-1} + b_i) \quad (1)$$

$$f_t = \Phi_S(I_f x_t + R_f h_{t-1} + b_f) \quad (2)$$

$$o_t = \Phi_S(I_o x_t + R_o h_{t-1} + b_o) \quad (3)$$

Where  $h_{t-1}$  is the output of the LSTM cell at the previous time step. Similarly, the cell state  $c$  is written as:

$$c_t = \Phi_T(I_c x_t + R_c h_{t-1} + b_c) \quad (4)$$

At each time step, the LSTM uses the time series to compute  $c_t$  and  $h_t$  which are then fed into a regression layer to predict the time series value. We used the root mean square error as a metric performance during training.

### 3 Data Processing

#### 3.1 The Data

The energy consumption of 609 anonymous households in Victoria Australia have been collected by Power Distribution Company using smart meters every 30 min for a full year starting from 9 March 2015 and ending on 8 March 2016. The raw dataset came in 9 files and each file comprised of the energy consumption of all energy consumers in each period of time. Table 1 presents information on the raw dataset used for the research.

**Table 1.** Smart meter energy consumption raw data

Data file no.	Start date	End date	Total no. of sample points
9	9-Mar-15	19-Apr-15	1,227,744
1	20-Apr-15	31-May-15	1,227,744
2	1-Jun-15	11-Jul-15	1,198,512
3	12-Jul-15	22-Aug-15	1,227,744
4	23-Aug-15	2-Oct-15	1,198,512
5	3-Oct-15	14-Nov-15	1,256,976
6	15-Nov-15	27-Dec-15	1,256,976
8	28-Dec-15	6-Feb-16	1,198,512
7	7-Feb-16	8-Mar-16	906,192

Combining all data files into one produces a 3-by-10,698,912 matrix where the first, second and third column corresponds to consumer label, consumption time and energy consumption respectively. The next step is to pre-process the raw data into a feature vector for clustering and then forecasting.

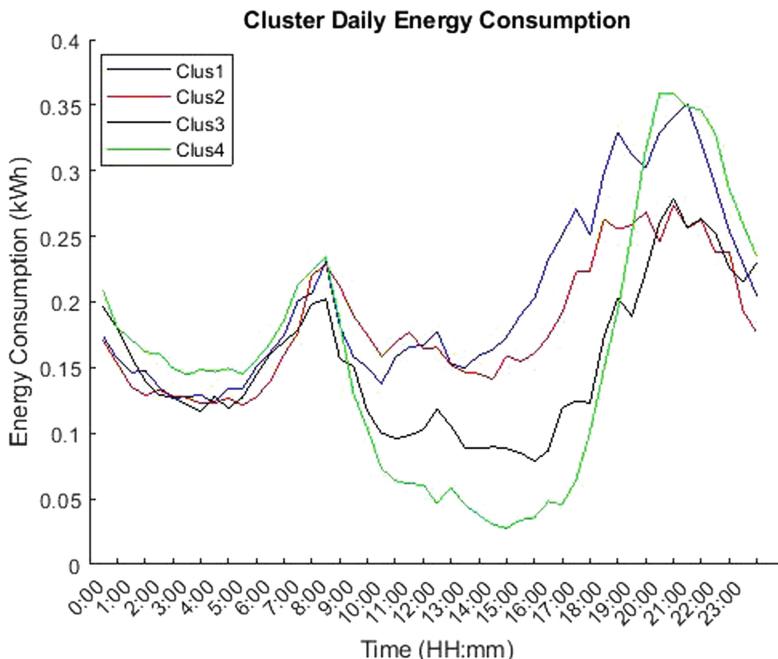
#### 3.2 Data Pre-processing and Analysis

We then reorganize the data into an energy consumption matrix  $A$  17520-by-609 where columns are energy consumptions in ascending order and rows are the energy consumers identification from the dataset. In addition, we create a time column vector  $B$  17520-by-1 to preserve the time instances corresponding to energy consumption.

Few time instances were missing for some of the energy consumers due to either collecting the data days earlier compared to other consumers or after. These time instances were removed from the energy consumption matrix to avoid dealing with empty cells when training the neural network hence reducing the size of Matrix  $A$  to 17496-by-609.

### 3.3 Clustering

Forecasting the load of individual energy consumer is often impracticable in residential sector as each consumer contributes a small proportion to the loading of the transformer or the connection point unless the consumer is a large industrial load which can be treated separately. As a result we employ a metric to cluster the dataset into optimal number of clustering thus reducing the number of loads into manageable time series for forecasting. The metric is based on the eigenvalues of the correlation matrix between the clusters. We use self-organizing maps to group energy consumers from 2 to n clusters and deduce a cluster's representative profile by taking the mean of all energy consumption profiles within the cluster. We then use the metric to decide whether the number of clusters is optimal or not. We found that the optimal number of clusters for this dataset is 4 clusters which are plotted in Fig. 1. The clustering technique is discussed in another paper and is outside the scope of this work.



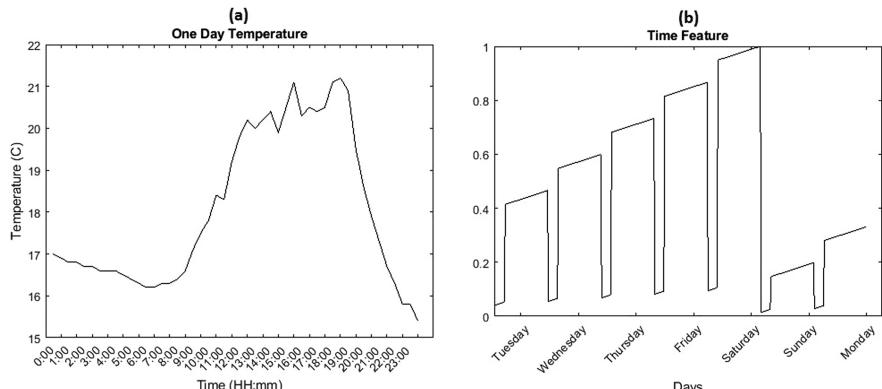
**Fig. 1.** Representative cluster consumption profile of 609 energy consumers within the dataset

### 3.4 Features Definition

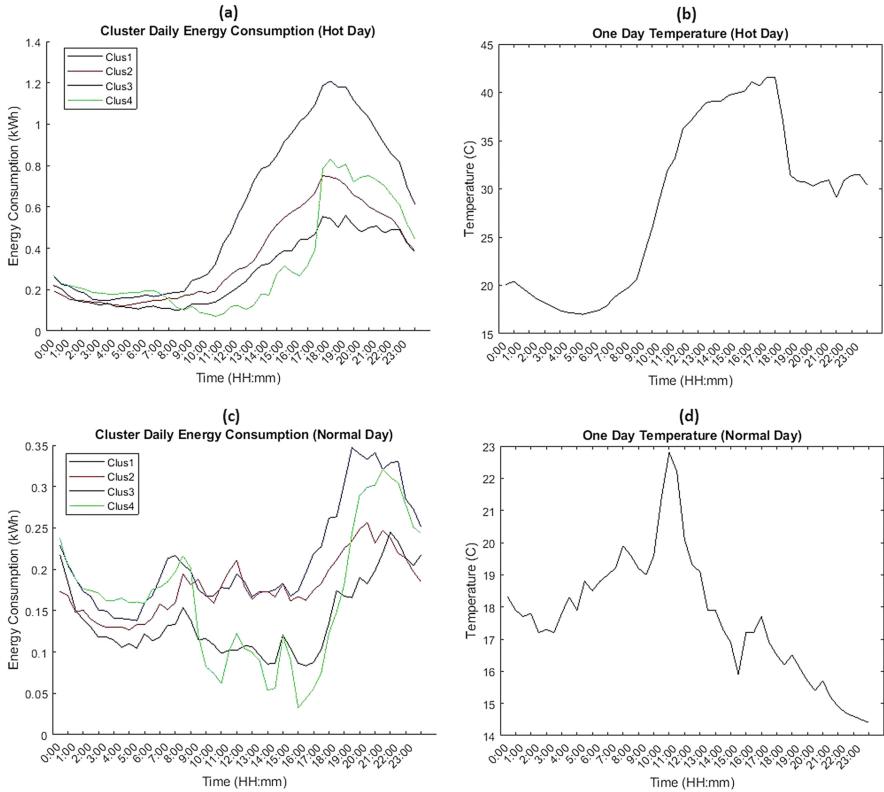
We have identified three features that affect the forecast error and improve predictions namely; historic energy consumption, daily temperature and time of day. Historic energy consumption provides insight into how much energy is being used in a day in each month at specific time of the days. This allows the forecast model to learn from historic consumption to predict future ones.

The daily temperature affects the consumer behavior to use less or more energy depending on how hot or cold the weather is at a specific time instance. Temperatures in the range of 19 to 25 °C are comfortable temperature that would unlikely to influence the energy consumption; however any temperature higher or lower than the comfortable temperature range would likely to influence the energy consumption. Figure 2a shows a typical plot of a temperature profile across the day and Fig. 3 shows how the clusters energy consumption change during a hot day when the temperature is higher than the comfortable temperature compared to a normal day. This is evident from the energy consumption range of 0.1 to 1.2 kWh in Fig. 3a during a hot day compared to the energy consumption range of 0.1 to 0.35 kWh during a normal day in Fig. 3c.

Time affects energy consumer as it represents the instance when a specific energy is consumed. This is usually different for each consumer as the consumption will be affected by daily behavior such as scheduling of regular loads such laundry, dish washing, TV and other electronics devices. Another factor that impact consumption on different is whether the house become non-occupant during a certain time of the day where parent are at work and children at school. This can change from one family to another. The day of the week is also important as the consumer behavior is likely to be different on weekdays compared to weekends. As a result it is vital to propose a time feature where each time instances in a 7 day is unique and each day in a week is unique as week. We construct the feature vector from the time instances where each day is given a number from 1 to 7, where 2–6 represent weekdays and 1 and 7 represent weekends, and each 30-min interval in the day is given a number from 1 to 48 where corresponds to 00:00 and 48 corresponds to 23:00. The time instance then becomes a factor of day and time by appending the time number to the end of the day number. For example, 8:00 am on Tuesday is written as 318 where 3 corresponds to Tuesday and 18 corresponds to 8:00am. We then divide this number by the highest value which is 748 corresponding to Saturday at 23:00 or 11:30 pm to get a time feature value between 0 and 1. Figure 2b is a plot of the time feature profile across the week.



**Fig. 2.** (a) Temperature profile in °C across the day for 0 March 2015; (b) Time feature profile to across the week giving each 30-min time instance a unique number



**Fig. 3.** (a) Clusters energy consumption on a hot day; (b) The temperature across the day for 13 January 2016; (c) Clusters energy consumption on a normal day; (d) The temperature across the day for 16 October 2015 unique number

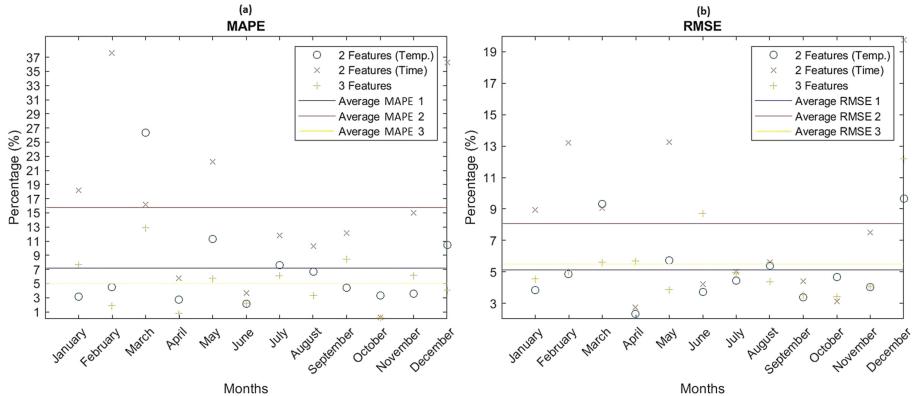
## 4 Experimental Results

We conducted the simulation over three experiments where each experiment tests different hypothesis. The first experiment tests the effectiveness of LSTM short-term forecasting capabilities by comparing performance error of 3-day ahead forecasts for cluster 1 across different months of the year using the historic energy consumption along with either the temperature or the time feature and by using all three features together. The second experiment uses both time and temperature features to forecast energy consumption for clusters 2, 3 and 4 and third experiment aims to forecast 15 days ahead in a year. In all cases the dataset is divided into a ratio of 70, 10 and 10 for training, validation and testing respectively. In the case of 3-day ahead forecasts the

dataset corresponds to daily consumptions in each month, however in the case of 15 days ahead forecast the dataset corresponds to the daily consumption for the entire year.

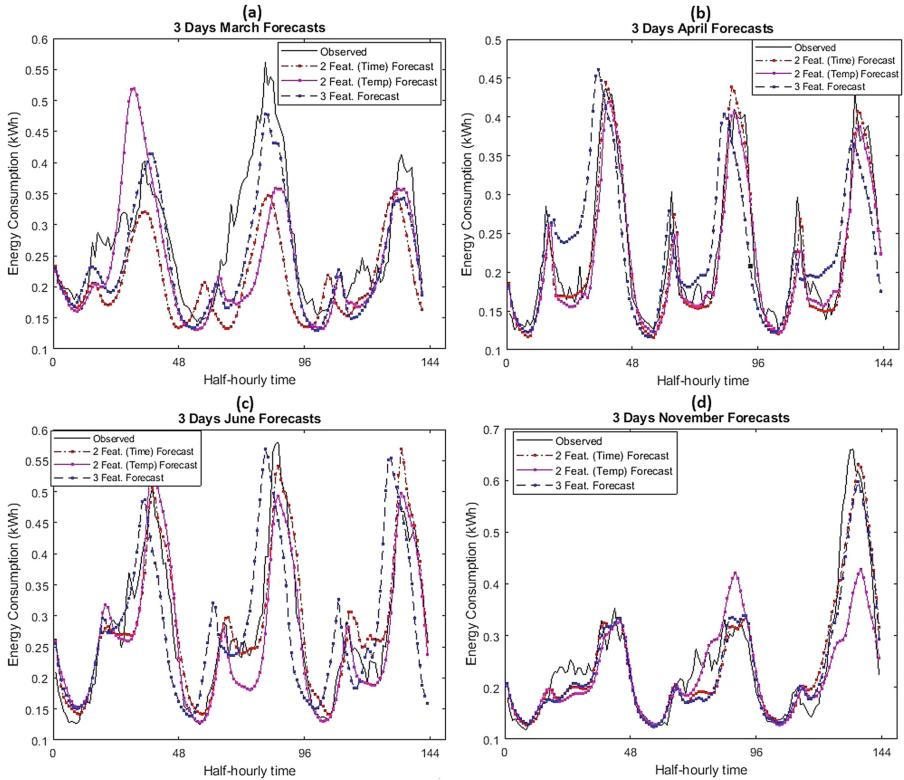
#### 4.1 Experiment 1–3-Day Ahead Forecast (2/3 Features)

We use both the MAPE and the RMSE in testing the performance of the LSTM which is depicted in Fig. 4. It can be seen from the figure that the lowest error in terms of MAPE is achieved by the 3 features forecast in October (Fig. 4a) and the lowest error in terms of RMSE is achieved by the 2 features (temperature feature) forecast in April (Fig. 4b). On average the 3 features forecast across all months is about 5% in terms of MAPE and 5.2% in terms of RMSE. While the average RMSE for 2 features forecast scored slightly lower (less than 0.5%) than the 3 features however the average MAPE for the 3 features is considerably lower (2%) than 2 features.



**Fig. 4.** The mean absolute percentage error (a) and the root mean square error (b) for 3-day ahead prediction for LSTM in each month and the average MAPE across all months for 2 and 3 features

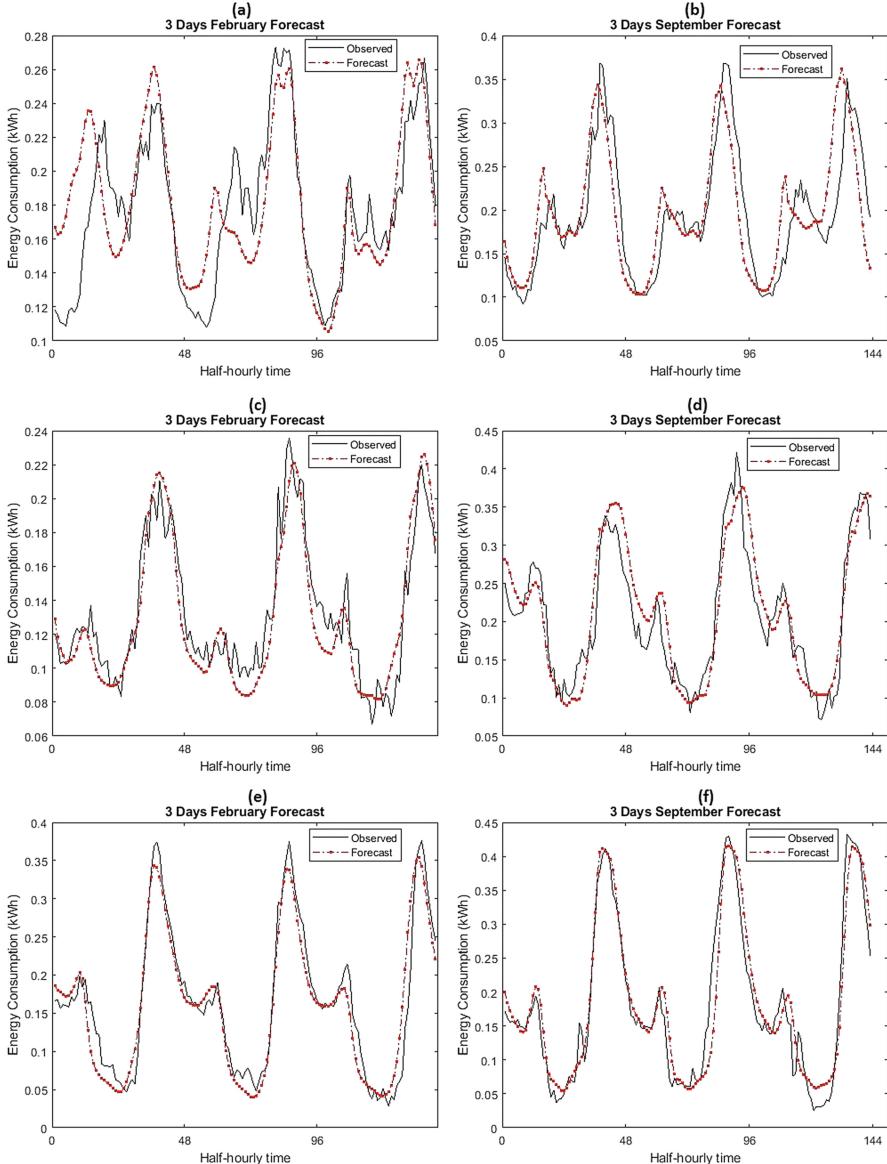
Figure 5 shows the 3-day ahead energy consumption forecasts for April, June, March and November. It can be observed from March forecasts that none of the features used for training were able to accurately predict the second and the third peak consumption. This can be attributed to March raw data being from two different years as shown in Table 1. However the intensities of the peak consumptions in April, June and November are forecasted with acceptable accuracy. Overall the forecasts using the three features are more accurate compared with two features.



**Fig. 5.** 3-day ahead forecasts for 2 and 3 features in the months of March (a), April (b), June (c) and November (d)

#### 4.2 Experiment 2–3-Day Ahead Forecast (3 Features on 3 Remaining Clusters)

In this experiment we use the monthly energy consumption from the dataset to train the LSTM to forecast 3-day ahead for the remaining three clusters and using the 3 features defined in previous section. Figure 6 shows the forecast energy consumption compared with the actual. It can be observed from the figure that the September peak consumption forecast for cluster 4 is the most accurate compared with other clusters in the same month. This may be attributed to fewer variations in the energy consumption across the day for cluster 4. In comparison cluster 2 has more variations across the day and has almost two recorded peak consumptions across the day resulting in higher forecasting error. Overall, the proposed methodology, features and LSTM architecture can be used to accurately forecast 3-day ahead energy consumption of various pattern or shape.

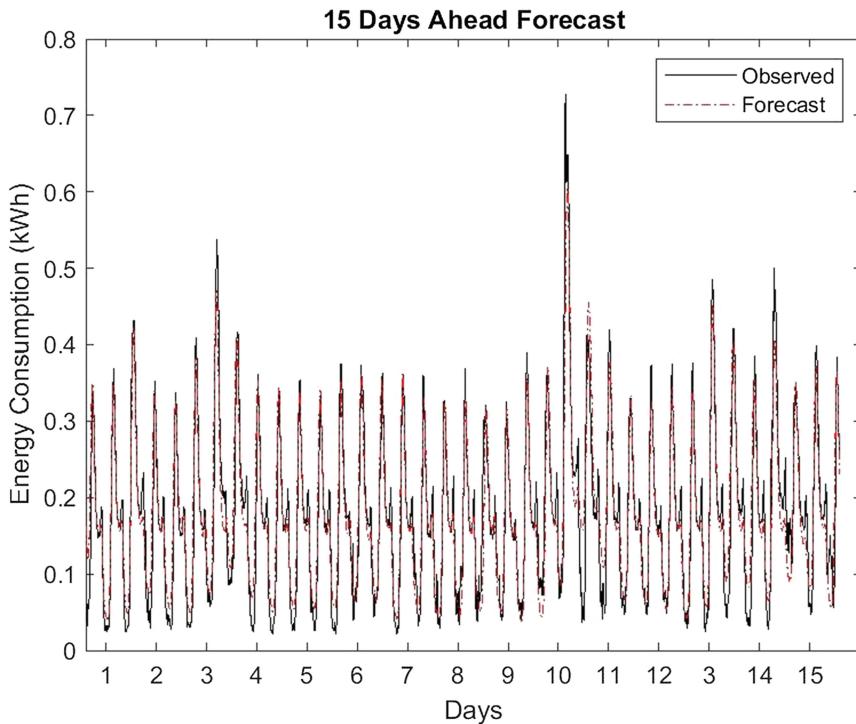


**Fig. 6.** 3-day ahead forecasts using 3 features in the months of February and September for cluster 2 (a) and (b), cluster 3 (c) and (d) and cluster 4 (e) and (f)

#### 4.3 Experiment 3-15-Day Ahead Forecast (3 Features)

In this experiment we use the energy consumption for the whole year to train the LSTM to forecast 15-day ahead energy demand. Figure 7 depicts the forecast energy consumption compared with the actual consumption for cluster 4. It can be noted from the

figure that the energy consumption is forecasted with acceptable forecasting error of 3.7624% and 3.61% in terms of MAPE and RMSE respectively. A 15 days ahead prediction can very useful for utilities to determine whether peak consumption is likely to occur during the 15 days so they can plan for it accordingly.



**Fig. 7.** 15 days ahead energy consumption forecast for cluster 4 with MAPE of 3.7624% and RMSE of 3.61%

## 5 Conclusion and Future Work

This work focuses at the potential of using deep learning LSTM to forecast 3 and 15 days ahead energy demand of different load profiles. The outcome of the research suggests that LSTM is a strong architecture for both short and medium term forecasting. We have also shown that defining effective features improve the forecasting model. As load and energy demand is critical for demand side management, this work can provide utilities with the needed information to make decision on how to better manage peak energy demand with an error of 3.15%. As future works, we plan to improve the LSTM forecasting in short term and accurately forecast 3-month ahead by tweaking the existing architecture as well as develop LSTM deep learning network for long term forecasting.

## References

1. Alahakoon, D., Yu, X.: Smart electricity meter data intelligence for future energy systems: a survey (2016)
2. Wang, Y., Chen, Q., Hong, T., Kang, C.: Review of smart meter data analytics: applications, methodologies, and challenges. *IEEE Trans. Smart Grid* **PP**(99), 1 (2018)
3. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
4. P. A. Citipower: Demand Side Management Strategy (2016). Accessed 1 Mar 2019
5. Barton, J., et al.: The evolution of electricity demand and the role for demand side participation, in buildings and transport. *Energy Policy* **52**(C), 85–102 (2013)
6. Fernández, M.R., García, A.C., Alonso, I.G., Casanova, E.Z.: Using the big data generated by the smart home to improve energy efficiency management. *Energy Effic.* **9**(1), 249–260 (2016)
7. Mirowski, P., Chen, S., Ho, T.K., Yu, C.N.: Demand forecasting in smart grids. *Bell Labs Tech. J.* **18**(4), 135–158 (2014)
8. Raza, M.Q., Khosravi, A.: A review on artificial intelligence based load demand forecasting techniques for smart grid and buildings. *Renew. Sustain. Energy Rev.* **50**, 1352–1372 (2015)
9. Zhao, K., Gan, L., Wang, H., Ye, A.: Application of combination forecast model in the medium and long term power load forecast. *Int. J. Comput. Sci. Issues (IJCSI)* **9**(5), 24 (2012)
10. Alahakoon, D., Yu, X.: Smart electricity meter data intelligence for future energy systems: a survey. *IEEE Trans. Ind. Inform.* **12**(1), 425–436 (2016)
11. Kwang-Ho, K., Hyoung-Sun, Y., Yong-Cheol, K.: Short-term load forecasting for special days in anomalous load conditions using neural networks and fuzzy inference method. *IEEE Trans. Power Syst.* **15**(2), 559–565 (2000)
12. Verdu, S.V., Garcia, M.O., Senabre, C., Marin, A.G., Franco, F.J.G.: Classification, filtering, and identification of electrical customer load patterns through the use of self-organizing maps. *IEEE Trans. Power Syst.* **21**(4), 1672–1682 (2006)
13. Pao, H.-T.: Forecasting electricity market pricing using artificial neural networks. *Energy Convers. Manag.* **48**(3), 907–912 (2007)
14. Hossen, T., Nair, A.S.: Residential load forecasting using deep neural networks (DNN) (2018)
15. Liu, C., Jin, Z., Gu, J., Qiu, C.: Short-term load forecasting using a long short-term memory network. In: 2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe), pp. 1–6 (2017)
16. Han, L., Peng, Y., Li, Y., Yong, B., Zhou, Q., Shu, L.: Enhanced deep networks for short-term and medium-term load forecasting. *IEEE Access* **7**, 4045–4055 (2019)
17. Kong, W., Dong, Z.Y., Jia, Y., Hill, D.J., Xu, Y., Zhang, Y.: Short-term residential load forecasting based on LSTM recurrent neural network. *IEEE Trans. Smart Grid* **10**(1), 841–851 (2019)
18. Kuo, P.-H., Huang, C.-J.: A high precision artificial neural networks model for short-term energy load forecasting. *Energies* **11**(1), 213 (2018)



# Modelling of Compressors in an Industrial CO<sub>2</sub>-Based Operational Cooling System Using ANN for Energy Management Purposes

Sven Myrdahl Opalic<sup>1,2,3</sup>(✉), Morten Goodwin<sup>1,2</sup>, Lei Jiao<sup>1,2</sup>,  
Henrik Kofoed Nielsen<sup>2</sup>, and Mohan Lal Kolhe<sup>2</sup>

<sup>1</sup> Centre for Artificial Intelligence Research,  
University of Agder, 4879 Grimstad, Norway

<sup>2</sup> Faculty of Engineering and Science,  
University of Agder, 4879 Grimstad, Norway

<sup>3</sup> Login Eiendom AS,

Kongens gate 16, 7011 Trondheim, Norway

{sven.opalic,morten.goodwin,lei.jiao,henrik.kofoed.nielsen,  
mohan.l.kolhe}@uia.no

**Abstract.** Large scale cooling installations usually have high energy consumption and fluctuating power demands. There are several ways to control energy consumption and power requirements through intelligent energy and power management, such as utilizing excess heat, thermal energy storage and local renewable energy sources. Intelligent energy and power management in an operational setting is only possible if the time-varying performance of the individual components of the energy system is known. This paper presents an approach to model the compressors in an industrial, operational two-stage cooling system, with CO<sub>2</sub> as the working fluid, located in an advanced food distribution warehouse in Norway. An artificial neural network is adopted to model the compressors using the operational data. The model is trained with cooling medium evaporation and condensation temperature, suction gas temperature and compressor operating frequency, and outputs electrical power load and cooling load. The best results are found by using a single hidden layer with 45 hidden neurons and a hyperbolic tangent activation function trained with the Adam optimizer, with a resulting mean squared error as low as 0.08% for both the training and validation data sets. The trained model will be part of a system implemented in a real-world setting to determine the cooling load, compressor power load, and coefficient of performance. An intelligent energy management system will utilize the model for energy and power optimization of the cooling system by storing energy in a thermal energy storage, using predictions of energy demand and cooling system performance.

**Keywords:** Industrial CO<sub>2</sub>-based cooling system · Artificial neural networks · Modelling of cooling system · Warehouse energy management

## 1 Introduction

It is complex to design and operate an efficient building energy system that incorporates multiple elements of new and emerging technologies [7]. The increase in building-integrated intermittent renewable energy production, local energy storage, and micro-grid solutions provides the building operator with a multitude of options in choosing the optimal operational mode of all the components at any given time. Implementation of an Intelligent Energy Management System (IEMS) is one way to automate this decision-making process in order to reduce the total energy cost for a building [2, 12–15]. An IEMS can be tasked to predict short- and long-term energy demand and local energy production in order to continuously design an optimal schedule for all energy storage options, while also considering energy price fluctuations and peak power tariffs.

For the heating and cooling demands, heat pumps and Cooling Systems (CS) are widely accepted as an efficient way to produce thermal energy, with continual improvements being made to maximize efficiency [3]. In technologically advanced food distribution warehouses, large scale CS represent a large part of the buildings total energy demand. Changing operating conditions, such as weather (including ambient temperature), flow of goods and building occupant behaviour, will continuously impact CS performance [3, 11]. This is especially true for more environmentally friendly working fluids, such as CO<sub>2</sub>, that have made their relatively recent re-entry into the field of refrigeration technology [8, 9, 11]. CO<sub>2</sub>-based large scale multi-stage cooling systems are becoming common as natural refrigerants with low global warming potential and ozone depletion potential replace synthetic refrigerants. One way to increase the efficiency of these systems during operation is to produce thermal energy (heating or cooling) at the most ideal operating conditions and store the energy in a thermal energy storage (TES) for later use [1, 10]. This requires an IEMS that is given accurate energy measurements and individual system performance data. For the CS, this includes cooling load which requires working fluid flow measurement. However, because accurate CO<sub>2</sub> flow measurements are difficult, energy measurement of cooling demand supplied with CO<sub>2</sub> as the working fluid of energy distribution, typically in warehouses with large cooling and freezing storage areas, is usually unavailable. Theoretical calculation of system efficiency, or Coefficient of Performance (COP), is therefore necessary to determine system performance. However, industrially sized CS's are usually unique and built by intellectual property (IP) protected components that limit the system owner and operators options for continual performance evaluation. In many cases, system performance at given operating conditions can only be calculated by the supplier using a proprietary model, but the details of the model itself are not shared. Therefore, openly available alternatives are necessary in order to model the system for performance evaluation purposes to provide reliable input to an IEMS.

In this work, we use an Artificial Neural Network (ANN) to model the freezing stage compressors of an industrial and operational two-stage CO<sub>2</sub>-based CS. ANNs have already shown promising results in performance prediction modelling of heat pump technology [5], but in [5] the training data set consisted of a

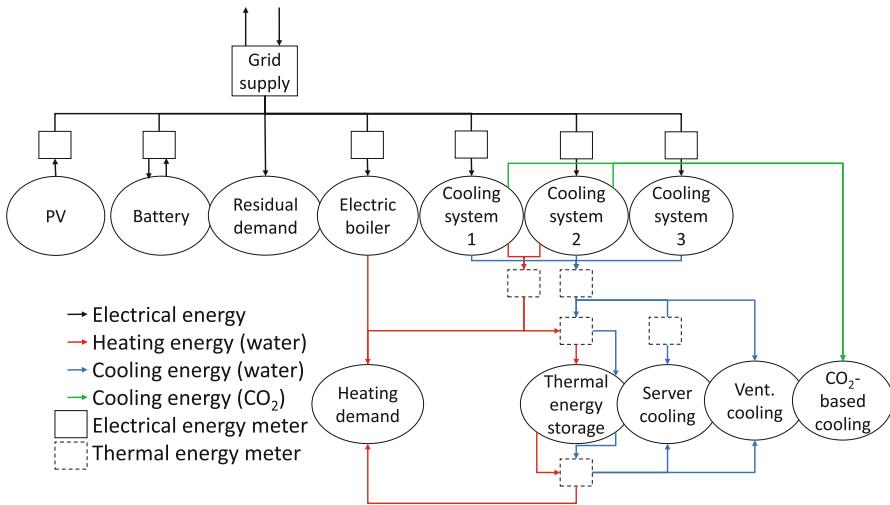
very limited amount of measurements in an experimental setting, where thermal energy and electrical energy input could be measured.

ANNs have the ability to approximate both simple and complex unknown functions that fit the underlying data. Therefore, we apply ANNs on data generated by an openly available online compressor calculation model. The aim is for the ANN to learn the underlying patterns within the data. With ANNs trained on compressor calculation data, the need for extensive knowledge and understanding of refrigeration technology is less critical in development of the model. This is key when it comes to ease of implementation and scalability in real world applications where every system has its own unique aspects that must be taken into account. Since the ANN model is built on freely available information, there is no need to access IP protected empirical data or algorithms, allowing the model to be designed independently of the compressor manufacturer. Lastly, since machine learning models, such as ANNs, can be further trained with new training examples, the parts of the system that are measurable and observable can be used to modify the model to adjust for observable deviations between theory and practice. This fact can also be used to reveal large discrepancies between expected and real performance that should be further investigated.

The remaining sections of this article are organized as follows. Section 2 presents the case-study energy system and cooling system where the ANN CS performance model will be applied. The research setup, various ANN designs and configurations are presented in Sect. 3. Section 4 contains the results and discussion from the various models that were trained. Finally, the article conclusions and suggested future work are presented in Sect. 5.

## 2 System Structure and Configuration

The research in this study is based on existing infrastructure and data from a 27,000 m<sup>2</sup> technologically advanced warehouse for food storage and distribution, located in Sandnes, Norway. The warehouse was completed in 2017 and is currently implementing a commercially available IEMS based on hourly scheduling that uses various machine learning techniques to optimize energy storage, both electrochemical in batteries and thermal utilizing an isolated fire sprinkler basin, for storage of hot or cold water. The IEMS has to predict future electrical and thermal energy demand in order to come up with an optimal scheduling strategy. The main components of the warehouse energy system is a 1 GWp solar Photovoltaic power plant (PV), a 460 kWh storage capacity electrochemical lithium battery bank with two 100 kW inverters, a 300 m<sup>3</sup> water tank for thermal storage, a CO<sub>2</sub>-based large scale CS consisting of two identical two-stage cooling plants and a back-up cooling machine for ventilation air and IT-server cooling, an electrical boiler and accompanying technical infrastructure (HVAC, Lighting etc.). Excess heat from the CS is reclaimed and used to supply the building with heating energy. An overview of the warehouse temperature zones with their respective operating temperatures are listed in Table 1, whereas the main components of the energy system and their inter-dependencies are visualized in Fig. 1 and listed in Table 2.



**Fig. 1.** The case-study warehouse energy system.

**Table 1.** Warehouse cooling floor area and operating temperatures

Area	Size	Operating temperature
Warehouse	27,000 m <sup>2</sup>	-20 °C to +20 °C
Frozen storage area	3,000 m <sup>2</sup>	-20 °C
Cold storage area	3,600 m <sup>2</sup>	+2 °C
Cooled shipping area	3,600 m <sup>2</sup>	+2 °C

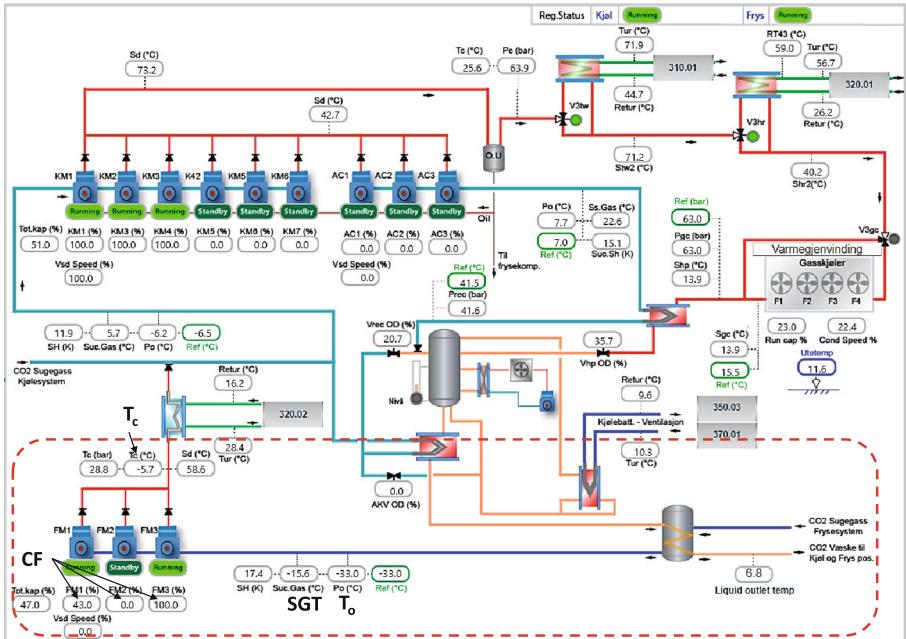
**Table 2.** Energy system components, capacity in [kW<sub>p</sub>], [kW], [kWh], [m<sup>3</sup>] and [kW<sub>thermal</sub>]

Component	Capacity	Unit of measurement
PV - solar power plant	1,000	[kW <sub>p</sub> ]
EB - battery bank	460/200	[kWh/kW]
TES - water tank	300	[m <sup>3</sup> ]
CS - cooling system	1,140	[kW <sub>th</sub> ]
Electrical boiler	495	[kW]

The TES is operated as a Cooling Energy Storage (CES) during spring, summer and fall, and as a Heating Energy Storage (HES) during winter. When ambient air temperature decreases, cooling demand and thereby available excess heat is reduced to the point where excess heat is insufficient to meet the heating energy demand. Therefore, the HES can reduce required heat supply and power load on the electrical boiler by storing available excess heat from the CS.

For the remainder of the year, the CES is used in one of two ways – to store excess energy from the PV-plant when production exceeds demand, or to store

cooling energy produced during optimal operating conditions for the CS. In order to store excess energy from the PV-plant, the electrical energy is converted to thermal energy by the CS and stored in the CES as chilled water in a temperature range between 7 °C and 15 °C. In the evening, when production from the PV-plant is naturally reduced, the CES is discharged by directly supplying cooling energy for ventilation air and IT-servers. Alternatively, the CES can be used to optimize cooling energy production by charging and discharging based on varying operating conditions, such as current and predicted cooling load, and current and predicted ambient air temperature.



**Fig. 2.** The case-study cooling system as visualized in the building management system, courtesy of IWMAC. Freezing compressors and CO<sub>2</sub> distribution to evaporators roughly outlined in the red dashed box, ANN model input values marked in bold (CF in %). (Color figure online)

For the IEMS to make the optimal choice of operating mode for the TES, the performance of the CS must be evaluated both at current and future operating conditions. In this work, ANN models for theoretical calculation of cooling load and compressor power consumption based on available compressor data is explored. The models have been developed for the three identical semi-hermetic reciprocating sub-critical compressors, denoted FM1, FM2 and FM3 within the red dashed box in Fig. 2, but the whole CS will be modelled in future work.

Compressor performance calculation data for the 4CSL-12K compressors was collected from the website of the manufacturer, Bitzer™. Theoretical values were calculated using the following given equation (according to EN12900):

$$y = c_1 + c_2 t_o + c_3 t_c + c_4 t_o^2 + c_5 t_o t_c + c_6 t_c^2 + c_7 t_o^3 + c_8 t_c t_o^2 + c_9 t_o t_c^2 + c_{10} t_c^3 \quad (1)$$

In Eq. (1),  $t_o$  is the evaporating temperature and  $t_c$  is the condensing temperature.  $c_1$  to  $c_{10}$  are constants that are given based on the selected suction gas temperature (SGT), compressor frequency (CF) and subcooling temperature. Four different sets of constants are given that are used to calculate either cooling capacity  $Q$ , power input  $P$ , current  $I$  or mass flow, represented as  $y$  in Eq. (1), for the compressor within its defined and given operating range. Constants for  $Q$  and  $P$  only were collected in 5° steps, from -30 °C to -5 °C for suction gas temperature, and for 5 Hz steps from 70 Hz to 30 Hz for compressor frequency. Since the subcooling temperature was unknown at the time of data extraction, it was set to 2 °K as a reasonable average value provided by the cooling system supplier. A total number of 96 rows of 10 constants were collected and labeled with suction gas temperature and compressor frequency, as well as the compressor evaporation and condensation range at the given operating conditions. Finally,  $P$  [kW] and  $Q$  [kW<sub>th</sub>] were calculated using integers for the compressor evaporation and condensation range, resulting in a data set of approximately 30 000 example values. The COP of the compressor can then calculated by Eq. (2).

$$COP = \frac{Q}{P} \quad (2)$$

### 3 ANN Approach Design and Configurations

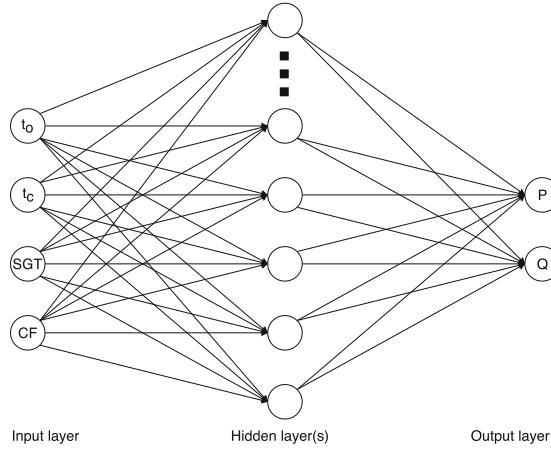
Clearly, from Eq. (1), we understand that the function of the output is in a polynomial form. Although the functions between the inputs and the parameters  $c_i, \forall i \in 1, 2, \dots, 10$  are unknown, Eq. (1) provides important information which can be considered as an indicator of the overall hidden function that our ANN model is approximating. To approach a function that has polynomial features as the overall trend, we believe that a simple ANN with sigmoidal activation functions in the hidden layer is most probably sufficient [4]. Therefore, instead of applying modern deep learning techniques, we start with a neural network structure with one hidden layer (HL) and gradually increase the number of layers and neurons to observe the learning behavior and efficiency. Fully connected ANNs, as shown in Fig. 3 with single and multiple HLs with nonlinear activation functions were trained to predict  $P$  and  $Q$  by forward propagating data from the input neurons To, Tc, SGT and CF through the HLs as shown in Fig. 3. Hyperbolic tangent Eq. (3) (Tanh), tangent sigmoid Eq. (4) (Tansig) from [5], rectified linear unit Eq. (5) (ReLU) and sigmoid Eq. (6) activation functions were tried. Since the Tansig function used in [5] is exponential, the ReLU function was also attempted. An Adam optimizer [6] was used to train the networks until convergence using the Keras early-stop function. During training, the training data set was divided into batches of 100 examples so that the trainable model parameters could be updated after each batch. Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) for a single model, were used as loss functions and model accuracy metrics.

$$\text{Tanh} = \frac{e^z - e^{-z}}{e^z + e^{-z}} \quad (3)$$

$$\text{Tansig} = \frac{2}{(1 + e^{-2z}) - 1} \quad (4)$$

$$\text{ReLU} = \max(0, z) \quad (5)$$

$$\text{Sigmoid} = \frac{e^z}{1 + e^z} \quad (6)$$



**Fig. 3.** Fully connected ANN with four neurons in the input layer and two neurons in the output layer.

The models were programmed in Python 3.6 and the Keras library. Several network configurations were tested by changing the amount of HMs, the amount of neurons and the corresponding activation functions in each HM. Input values were normalized by subtracting the mean and normalizing the variance using Eqs. (7)–(10). The calculated values of  $\mu$  and  $\sigma^2$  for the training data set  $\{X_i\}$  were also applied to the validation data set using Eqs. (8) and (10).

$$\mu = \frac{1}{m} \sum_{i=1}^m X_i \quad (7)$$

$$X_i^{(\mu)} = X_i - \mu \quad (8)$$

$$\sigma^2 = \frac{1}{m} \sum_{i=1}^m \left( X_i^{(\mu)} \right)^2 \quad (9)$$

$$X_i^{(\sigma^2)} = \frac{X_i^{(\mu)}}{\sigma^2} \quad (10)$$

**Table 3.** Results - One hidden layer ANN.

Activation-loss-neurons	Training epochs	Training loss	Validation loss
Tanh-MSE-7	6,184	0.008790	0.008639
Tanh-MSE-10	5,202	0.002355	0.002236
Tanh-MSE-15	4,029	0.001065	0.001047
Tanh-MSE-20	3,242	0.000644	0.000708
Tanh-MSE-25	2,398	0.000558	0.000554
Tanh-MSE-30	2,271	0.000530	0.000518
Tanh-MSE-35	2,620	0.000411	0.000379
Tanh-MSE-40	3,404	0.000367	0.000361
<b>Tanh-MSE-45</b>	<b>2,868</b>	<b>0.000291</b>	<b>0.000273</b>
Tanh-MSE-50	2,987	0.000319	0.000456
Tanh-MSE-55	2,742	0.000297	0.000294
Tanh-MSE-60	3,616	0.000326	0.000309

**Table 4.** Results - Two hidden layers ANN.

Activation-loss-neurons-HL	Training epochs	Training loss	Validation loss
Tanh-MSE-5-2	2,010	0.008039	0.008952
Tanh-MSE-7-2	1,602	0.002343	0.002125
Tanh-MSE-10-2	1,641	0.001450	0.001293
Tanh-MSE-15-2	1,076	0.000880	0.000721
Tanh-MSE-20-2	1,127	0.000762	0.000945
Tanh-MSE-25-2	1,561	0.000483	0.000730
Tanh-MSE-30-2	712	0.000809	0.000797
Tanh-MSE-35-2	683	0.000675	0.000480
Tanh-MSE-40-2	828	0.000632	<b>0.000415</b>
Tanh-MSE-45-2	1,074	<b>0.000541</b>	0.0030
Tanh-MSE-50-2	879	0.000639	0.0013

## 4 Results and Discussion

The data set consists of 29,408 values, divided randomly into a training and validation data set using a 90/10 split. The results for single HL models with a hyperbolic tangent (Eq. (3)) activation function are listed in Table 3.

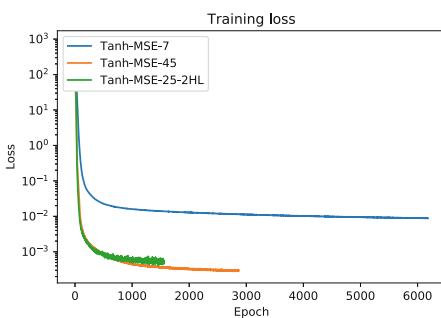
A single HL model with 45 neurons in the HL (Tanh-MSE-45) outperformed all multiple hidden layer models in all tested configurations. This includes models with multiple hidden layers, with both different and identical activation functions across the hidden layers. This result is logical based on the expected polynomial

shape of the hidden ground-truth function. The system complexity is limited, and therefore does not require too many neurons in the hidden layer, as shown in Fig. 4.

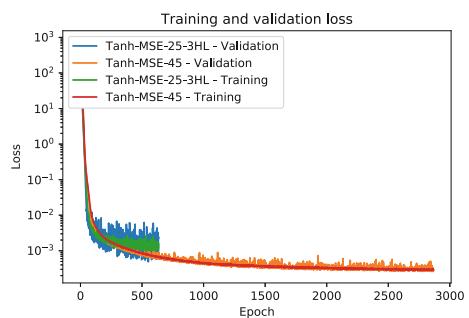
For the single layer models, there was little difference between training and validation error. In contrast, multiple layer models tended towards a higher validation error as well as bigger differences between training and validation, as exemplified in Fig. 5, which is a sign of overfitting the training data. The multiple layer models also tended towards larger variations in loss between every update of the trainable parameters, which is to be expected since there are more parameters being updated after every training batch. This can be clearly observed in Fig. 4 and in Fig. 5 where the fluctuations increase with the amount of trainable parameters. This effect could perhaps be mitigated by tuning training parameters, but because the expected complexity of the hidden function is limited we do not believe that accuracy can be improved by adding more hidden layers. This assumption is supported by the results.

**Table 5.** Results - One seven neuron hidden layer ANN with different activation and loss functions, and the best performing SVR model.

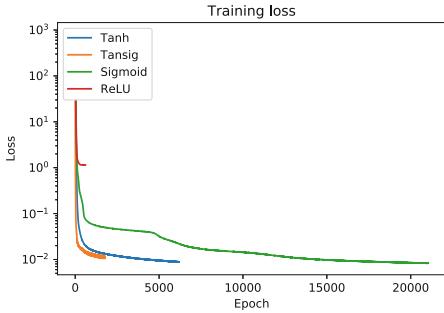
Model	Training epochs	Training loss (as MSE)	Validation loss
Tanh-MSE-7	6,184	0.0088	0.0086
Tansig-RMSE-7	1,763	0.0110	0.0106
Sig-MSE-7	21,043	0.0083	0.0079
ReLU-MSE-7	606	1.1444	1.1697
SVR-C1e10-RBF	-	0.0095	0.0094



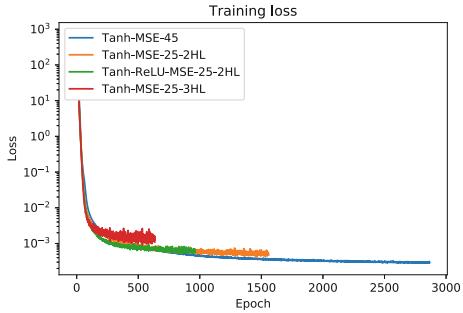
**Fig. 4.** Training loss comparison of the training error between Tanh-MSE-7, Tanh-MSE-45 and Tanh-MSE-25-2HL.



**Fig. 5.** Training loss comparison of the training and validation error between Tanh-MSE-45 and Tanh-MSE-25-3HL.



**Fig. 6.** Training loss comparison of the training error between different activation functions for single HL, seven neuron models.



**Fig. 7.** Training loss comparison of the training error between the best performing models with one (Tanh-MSE-45), two (Tanh-MSE-25-2HL and Tanh-ReLU-MSE-25-2HL) or three HL (Tanh-MSE-25-3HL).

An ANN close to the best performing ANN in [5] (Tansig-RMSE-7 in Table 5) was also trained in order to compare result accuracy with the most similar study found. The architecture of this network consists of a single HL with seven neurons and input values normalized to values between zero and one, using a tangent sigmoid activation function (Eq. (4)) and an RMSE loss function. One HL, seven hidden neuron models were also trained using sigmoid (Eq. (6)) and ReLU (Eq. (5)) activation functions. The tangent sigmoid model converges most quickly among the tested activations, but fluctuates significantly between every batch update. The sigmoid activation is the slowest to converge, but achieves the lowest MSE by a very small margin. This could be explained by the random initiation of parameters and we were not able to duplicate this result multiple times. Several Support Vector Regression (SVR) models were also trained with Sci Kit, using linear, polynomial and gaussian kernels. The best performing SVR, using a very large value for C and the gaussian kernel function is also included in Table 5. Finally, a single HL 7 neuron model using the hyperbolic tangent activation function (Eq. (3)) was trained and yielded the best results, within a reasonable amount of training time, as shown in Table 5 and Fig. 6 which show the comparison of different activation functions. The best performing SVR model result is included in Table 5. The result for the Tanh activation was achievable on multiple training sessions, indicating that the Tanh activation function is a reasonable fit for the training and validation data.

For models with two HL using the hyperbolic tangent (Eq. (3)) activation function, results are displayed in Table 4. None of these models outperformed the Tanh-MSE-45 ANN, but the comparison of training error in Fig. 7 show that the more complex two HL models converge and improve more quickly than the single layer models. This is expected since the amount of trainable parameters increases drastically once extra layers are added, but the fast learning rate quickly abates

and results in fluctuating, indicating that the models are too complex for the underlying data.

Finally, to examine how the Tanh-MSE-45 model performs on completely new data, some example calculations with Bitzer™ software were done using input data that fall in between the values that were used to generate the training and validation data sets. Specifically, instead of 5 °C steps for SGT ∈ −30, −25, … −5, the values −12.5, −17.5 and −22.5 were used. Similarly, values for CF were set to 67, 63, 47, 43, 37 and 33. Table 6 show these results as well as % Squared Error (SE).

**Table 6.** Results - Tanh-MSE-45 model output compared with calculations done with software from Bitzer™.

t <sub>o</sub> /t <sub>c</sub>	SGT	CF	P Bitzer™	P Tanh-MSE-45	Q Bitzer™	Q Tanh-MSE-45	% SE
−35/−5	−12.5	67	13.24	13.25	52.7	52.6	0.03
−35/−5	−12.5	63	12.30	12.30	49.5	49.4	0.04
−35/−5	−17.5	47	8.95	8.96	36.7	36.6	0.03
−35/−5	−17.5	43	8.21	8.22	33.4	33.3	0.08
−35/−5	−22.5	37	7.18	7.18	28.6	28.6	0.01
−35/−5	−22.5	33	6.54	6.54	25.2	25.2	0.01

## 5 Conclusion

With a resulting MSE of 0.08%, we conclusively show that using an ANN to model the compressors in a cooling system is a valid approach that allows quick and quite accurate calculations of cooling load and compressor power. The compressor COP at the given operating conditions can then be calculated. The best result was achieved using a single HL ANN with a hyperbolic tangent activation function. The model was trained with a MSE loss function using the Adam optimizer. For this approach to be valuable to an IEMS, the transcritical compressors that interact directly with TES must also be modelled so that the full system performance can be calculated. For best use of the TES as a HES during winter, the maximal available and reclaimable heat must also be determined. The trained model will be part of a full CS model adopted in a real-world setting and used to determine the cooling load, compressor power load and COP as input to an IEMS for optimization purposes.

## References

1. Arteconi, A., Hewitt, N., Polonara, F.: Domestic demand-side management (DSM): role of heat pumps and thermal energy storage (TES) systems. *Appl. Therm. Eng.* **51**(1), 155–165 (2013)
2. Chen, C., Duan, S., Cai, T., Liu, B., Hu, G.: Smart energy management system for optimal microgrid economic operation. *IET Renew. Power Gener.* **5**(3), 258–267 (2011)
3. Chua, K., Chou, S., Yang, W.: Advances in heat pump systems: a review. *Appl. Energy* **87**(12), 3611–3624 (2010)
4. Cybenko, G.: Approximation by superpositions of a sigmoidal function. *Math. Control, Signals Systems* **2**(4), 303–314 (1989)
5. Esena, H., Inallib, M., Sengurec, A., Esena, M.: Performance prediction of a ground-coupled heat pump system using artificial neural networks. *Expert Syst. Appl.* **35**(4), 1940–1948 (2008)
6. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2014). CoRR abs/1412.6980
7. Manic, M., Amarasinghe, K., Rodriguez-Andina, J.J., Rieger, C.: Intelligent buildings of the future: cyberaware, deep learning powered, and human interacting. *IEEE Ind. Electron. Mag.* **10**(4), 32–49 (2016)
8. Neksa, P.: Co2 heat pump systems. *Int. J. Refrig.* **25**(4), 421–427 (2002)
9. Neksa, P., Rekstad, H., Zakeri, G., Schiefloe, P.A.: Co2-heat pump water heater: characteristics, system design and experimental results. *Int. J. Refrig.* **21**(3), 172–179 (1998)
10. Pardo, N., Montero, A., Martos, J., Urchueguía, J.: Optimization of hybrid - ground coupled and air source - heat pump systems in combination with thermal storage. *Appl. Therm. Eng.* **30**(8), 1073–1077 (2010)
11. Sarkar, J., Bhattacharyya, S., Gopal, M.: Optimization of a transcritical co2 heat pump cycle for simultaneous cooling and heating applications. *Int. J. Refrig.* **27**(8), 830–838 (2004)
12. Remani, T., Jasmin, E.A., Ahamed, T.P.I.: Residential load scheduling with renewable generation in the smart grid: a reinforcement learning approach. *IEEE Syst. J.* **6**(99), 1–12 (2018)
13. Venayagamoorthy, G.K., Sharma, R.K., Gautam, P.K., Ahmadi, A.: Dynamic energy management system for a smart microgrid. *IEEE Trans. Neural Netw. Learn. Syst.* **27**(8), 1643–1656 (2016)
14. Wen, Z., O'Neill, D., Maei, H.: Optimal demand response using device-based reinforcement learning. *IEEE Trans. Smart Grid* **6**(5), 2312–2324 (2015)
15. Zhao, Z., Lee, W.C., Shin, Y., Song, K.B.: An optimal power scheduling method for demand response in home energy management system. *IEEE Trans. Smart Grid* **4**(3), 1391–1400 (2013)



# Outlier Detection in Temporal Spatial Log Data Using Autoencoder for Industry 4.0

Lukas Kaupp<sup>1()</sup>, Ulrich Beez<sup>1</sup>, Jens Hülsmann<sup>2</sup>,  
and Bernhard G. Humm<sup>1</sup>

<sup>1</sup> Hochschule Darmstadt – University of Applied Sciences, Haardtring 100,  
64295 Darmstadt, Germany

{lukas.kaupp, ulrich.beez, bernhard.humm}@h-da.de

<sup>2</sup> ISRA Vision AG, Albert-Einstein-Allee 36-40, 45699 Herten, Germany

jhuelsmann@isravision.com

**Abstract.** Industry is changing rapidly under industry 4.0. The manufacturing process and its cyber-physical systems (CPSs) produce large amounts of data with many relationships and dependencies in the data. Outlier detection and problem solving is difficult in such an environment. We present an unsupervised outlier detection method to find outliers in temporal spatial log data without domain-specific knowledge. Our method is evaluated with real-world unlabeled CPS log data extracted from a quality glass inspection machine used in production. As a measurement metric for success, we set reasonable outlier areas in cooperation with a domain expert. Using our proposed method, we were able to find all known outlier areas. In addition, we found outliers that were not previously known and have been verified as outliers by a domain expert ex post.

**Keywords:** Autoencoder · Cyber-physical system · Outlier detection · Temporal · Spatial · Log data

## 1 Introduction

Cyber-physical systems (CPSs) use in industry is on the rise [1]. CPSs consist of both physical components, e.g. sensors, and cyber components, e.g. measurement software [2]. Each component may produce a massive amounts of log data [3], while also using different log schemas. Log data used in this paper was extracted from a quality glass inspection machine in production within a smart factory environment. Error detection and problem solving in such an environment is a time-consuming task [2–5]. Log schema evolution may even increase diversity.

An automatic outlier detection approach that can mark an outlier in log data can mitigate complexity and speed up problem solving. Supervised machine learning approaches are inappropriate due the lack of labelled log data. We apply an unsupervised autoencoder approach to find outliers in temporal spatial log data. Autoencoders are neural networks, that compress and decompress data using functions that are learned automatically from examples [6], in our case log data. Unsupervised outlier detection methods rely on the fact that outliers are only a small portion of the entire data [7]. This fact can be exploited within the autoencoder neural network. The more

examples of the same type of log data, the better the compression and decompression/reconstruction of the log data will be. As the result, the reconstruction error between input and output of the neural network will be higher in case of an outlier.

We have successfully used an autoencoder approach for automatically detecting outliers in the quality inspection machine environment without any domain-specific knowledge. For this, we employed the mean squared error (MSE) function between the input and output of the autoencoder network to find outliers in the temporal spatial log data. The log data with the greatest MSE has been considered as a potential outlier. With our method, we were able to mark multiple outlier in the provided log data, which have been verified as outliers by a domain expert ex post.

The remainder of this paper is structured as follows. Section 2 introduces a review of the related work done in outlier detection with autoencoders and within CPSs. In Sect. 3 we present our autoencoder approach to find outliers in temporal spatial log data. In Sect. 4 we outline our setup and evaluate our empirical results. Finally, we conclude our work and point to future work.

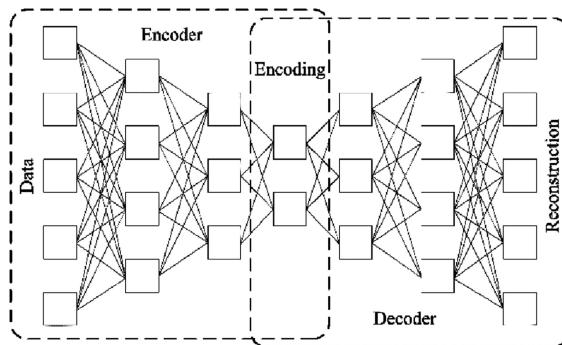
## 2 Related Work

Outlier detection has been extensively studied by the research community and published in comparative books and surveys [7–10]. In this paper, we focus on unsupervised outlier detection in CPSs, especially on outlier detection techniques based on autoencoders that are feasible on log data. As Chen et al. [11] recently stated, autoencoders are appropriate for finding outliers. Several challenges within a CPS environment exist that make outlier detection difficult. Two major challenges are the mixed log data [2] and the highly dimensional feature space [2, 12].

Mixed data consists of numerical, categorical, and temporal attributes [8]. Harada et al. [2] leverage this difficulty by pre-processing the log data of the CPS to enable their proposed Local Outlier Factor (LOF) to process the data. We are aware of mixed data difficulty and pre-process the log data to train an autoencoder neural network. We took the autoencoder approach to face the second challenge, because an autoencoder is able to identify high level correlations in features [6, 13]. Our approach is able to detect outliers using a threshold on reconstruction error [11]. One of the first who applied autoencoder approaches on event log data was Nolle et al. [14]. Nolle et al. [14] used an autoencoder on an artificially generated business process event dataset ( $\sim 50.000$  events) to separate normal traces of data from anomalous ones using a threshold. We apply our autoencoder approach to a different domain, using a different autoencoder configuration and a real-world dataset ( $\sim 1.2$  mio. log lines). However, we facilitate the idea of a threshold on the reconstruction error to classify outliers, as shown in [11, 14, 15], and test this approach on a real-world setting.

### 3 Background

An autoencoder is a neural network that transforms high-dimensional data into low-dimensional data and vice versa [6]. Furthermore, an autoencoder is able to identify high level correlations in features of given inputs [6, 13], which is particularly useful in a temporal spatial log data. Through the identification of high level features by its own, an autoencoder neural network can be used on non-labelled data without any domain specific knowledge. The autoencoder is trained by minimizing the reconstruction error between input and output data. The more often a pattern in input data exists the minor will be the reconstruction error [11, 15].



**Fig. 1.** Autoencoder neural network [16]

Figure 1 shows an autoencoder neuronal network. An autoencoder consists of an encoder neural network and reverse symmetric decoder neural network [6]. Therefore, the autoencoder keeps the dimension of the input data. An autoencoder can be represented by following function:

$$f_1: X \rightarrow \hat{X} \quad (1)$$

The autoencoder function (1) reconstructs the input from an unlabeled dataset  $X = \{V_i \in W^{D \times T}\}$ , where  $V_i = \{v_i^{(t)} \in W^d\}_{t=1}^T$  represents in our case, a multidimensional log line of length  $d \in D$  at Time  $t$  and over a Set  $W = \{d_1, d_2, \dots, d_n | d \in \mathbb{R} \vee d \in K \vee d \in \mathcal{L}\}$ . The log line consists of numerical values  $\mathbb{R}$ , messages from a language  $\mathcal{L}$ , and categorical values  $K \subset \mathcal{L}$ . The output domain  $\hat{X}$  is probabilistic, not exact, reconstruction of the input domain  $X$ . The MSE function that will be minimized in order to maximize reconstruction quality is:

$$f_2: \frac{1}{n} \sum_{i=1}^n (V_i - f_1(V_i, \theta))^2 \quad (2)$$

where  $f_1(\cdot, \theta)$  denotes the prediction  $\hat{X}$  of the autoencoder  $f_1$  using input log line  $V_i$ .

## 4 Our Approach

We explain our approach in four steps. Starting with a definition of our problem domain and the possible outliers that can appear in the environment. Next, we describe our autoencoder configuration and data strategy for training. Finally, we explain our promising outlier detection approach based upon the autoencoder.

### 4.1 Possible Outliers

In a CPS environment the following correlations appear that can contain outliers: time correlation, spatial correlation and functional correlation [17]. Possible outliers within these correlations can be mapped onto the following classes [7, 9]: point outliers, contextual outliers, collective outliers, and pattern outliers. In this paper we are not further considering contextual outliers, because we are not able to detect these kinds of outliers within our inspection machine setting with unlabeled data.

*Point Outliers.* A point outlier is an anomalous data point that differs from a group of points [7]. E.g. all points of set  $X$  can be grouped through a property  $d$ , so  $X_i = \{d \in x_i\}$ . Therefore, a point  $x_i$  with  $d \notin x_i$  is an outlier.

*Collective Outliers.* A collection of points within the data set, that can be seen as outliers [7]. Either the data points are defined outliers by occurrence or data properties.

*Pattern Outliers.* A function with known shape exhibits a pattern that can be used to identify outliers [9]. E.g. A log event stream exists with  $events = \{e1, e2, e3, \dots, e1, e2, e3\}$ , if an event  $e4$  occur within the pattern e.g.  $events = \{e1, e4, e2, e3\}$  a pattern outlier exists.

### 4.2 Dataset Preparation

Our dataset is already defined as a matrix  $W^{DxT}$ . Each log line consists of timestamps, messages, categorical values and numerical values. According to our definition of Set  $W$  we are in need of different strategies for different data values, in order to compute the MSE value of each log line. We state the following conversion rules (3):

*Temporal Values.* We use temporal values to set the order of the sequence within  $W^{DxT}$ . After sorting, the timestamps get deleted to reduce the feature space.

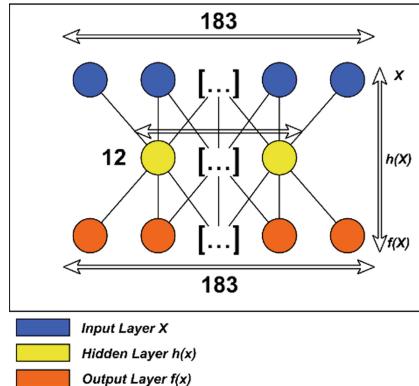
*Categorical Values.* We vectorize all categorical values using one-hot encoding [18]. The encoding process enlarges the spatial size of each log line for each categorical value found in the log data. For log lines with a high number of categorical values, another encoding should be chosen, such as a lower dimensional target embedding [19].

*Numerical Values.* All numerical values get scaled between zero and one, so scaled values will not collide with the vectorized categorical values and achieve balanced inweights between categorical and numerical values.

*Messages.* Each message gets automatically generated by software to ensure human readability and interpretability of each log line. For the purpose of deduplication of information, we prune the messages.

### 4.3 Autoencoder Configuration and Training Phase

We configured our autoencoder to fit the feature space of a log line. Our dataset consists of over 1.2 million log lines covering 2 weeks of operation with 183 values per log line. After the conversion all values are numerical and none of them are undefined.



**Fig. 2.** Autoencoder configuration

As a result, the autoencoder has 183 input neurons and inverse output neurons (see Fig. 2). We employ one hidden layer with 12 neurons between input and output layer. All neurons get initialized by Xavier uniform initializer algorithm [20]. The uniform distribution for weights being in range  $[-x, x]$ , where  $x$  is defined as follows per neuron:

$$x = \sqrt{\frac{6}{(\text{fan in} + \text{fan out})}} \quad (3)$$

The initializer (3) was chosen after successful field trials. We choose Rectified Linear Unit (ReLU) as the activation function for the encoding layer. ReLU is defined as follows [21]:

$$f(x) = \max(0, x) \quad (4)$$

Furthermore, we use the logistic function, a sigmoid function, as the activation function for the output layer. The logistic function is defined as follows:

$$\sigma(x) = \frac{e^x}{1 + e^x} \quad (5)$$

This combination of activation functions achieved a good performance [11]. In addition, we employ the Adam optimizer [22] and use the proposed MSE function (see Eq. 2) as loss function. The autoencoder is trained per log line in 50 epochs with a batch size of 256 shuffled samples. Through training per log line the autoencoder will learn correlations between values, the range of values within columns and the temporal aspect through the order of log lines. After the training phase, we can use the autoencoder in our outlier detection algorithm.

#### 4.4 Outlier Detection Method

Our outlier detection method is based on our already trained autoencoder neural network. Now, we use the autoencoder on all new log lines and predict an output. The output will differ more, be less correct, the less of the same data exists in training data. The hypothesis that outliers are only a minor portion of the training data help in this situation to predict outliers. We apply the MSE function between input and output, the higher the MSE, the higher the probability of an outlier, because the difference in output is caused by the minor training data. For our method we are in need of a domain expert to set a reasonable threshold in order to classify most outliers correct and produce less false-positives. In the following we describe our method in detail using pseudo-code (see Fig. 3).

---

**Method 1:** Proposed outlier detection algorithm

---

```

n = ||WDxT||;
i = 1;
outliers = Array();
Threshold t = 1.121E-4;
while vi = read next log line sample until i == n do
    Δxi = convertAccordingToRules[3](vi); /* according to rules (3) */
    Δei = useTrainedEncoderOn(Δxi);
    Δdi = useTrainedDecoderOn(Δei);
    Δmsei = MSE(Δxi, Δdi); /* according to equation (2) */
    if Δmsei > t then
        outliers.add(Δvi);
    end
    i = i + 1;
end

```

---

**Fig. 3.** Outlier detection method

Each log line gets extracted ( $v_i$ ), transformed ( $\Delta x_i$ ) and used as input for the autoencoder. Furthermore, the encoded log line ( $\Delta e_i$ ) is used to reproduce the input ( $\Delta d_i$ ). In addition, the MSE function is applied to  $\Delta x_i$  and  $\Delta d_i$ . As a result, the  $\Delta mse_i$

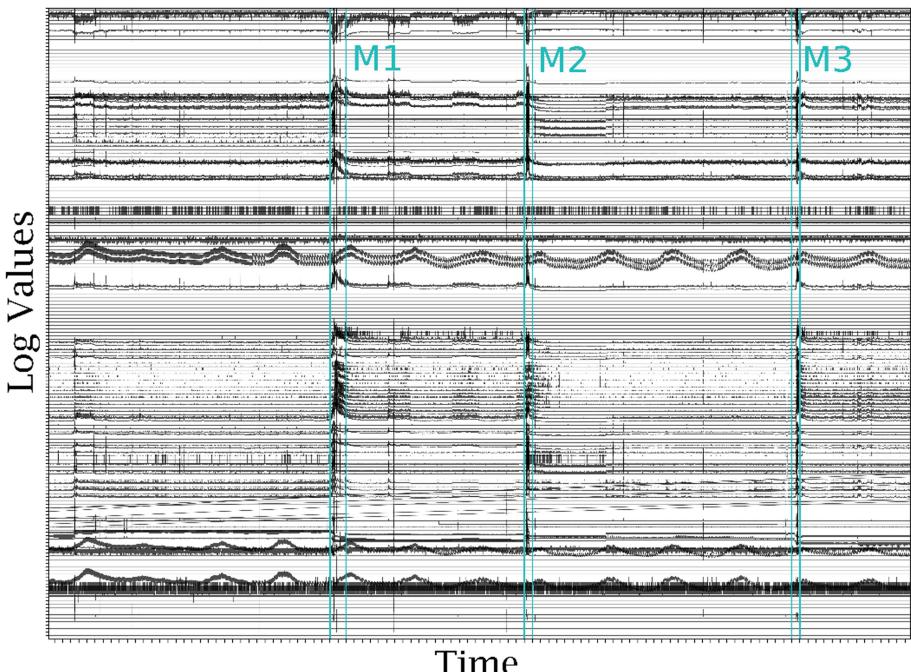
can be compared to threshold  $t$ . Everything larger than the threshold is considered as an outlier. Threshold  $t$  is defined using field trials with a domain expert knowing the CPS environment and reasonable outlier shapes in advance.

In order to suit other cases of log data, e.g. more columns, the autoencoder configuration for the input and output neurons must be adapted to the new column size. After the adaptation, the autoencoder needs to be trained on the new log data. Succeeding this process, it is again possible to use our proposed outlier detection method (see Fig. 3), the only change that has to be done is to set a new reasonable threshold.

## 5 Experiments and Results

We present our experimental results in this section. The experiments are done without any domain-specific knowledge. In order to verify our results on unlabeled log data, we conducted an interview with a domain expert *ex ante*.

The presented log data was captured on a glass inspection machine in production. A conveyor delivers a thin glass layer towards the glass inspection machine, which detects glass defects and rates material quality. A glass inspection machine consists of cameras, lights, multiple sensors, hardware and software modules, which produce log data. Figures 4 and 5 show the log files, after the dataset preparation step is done (see Sect. 4.2).

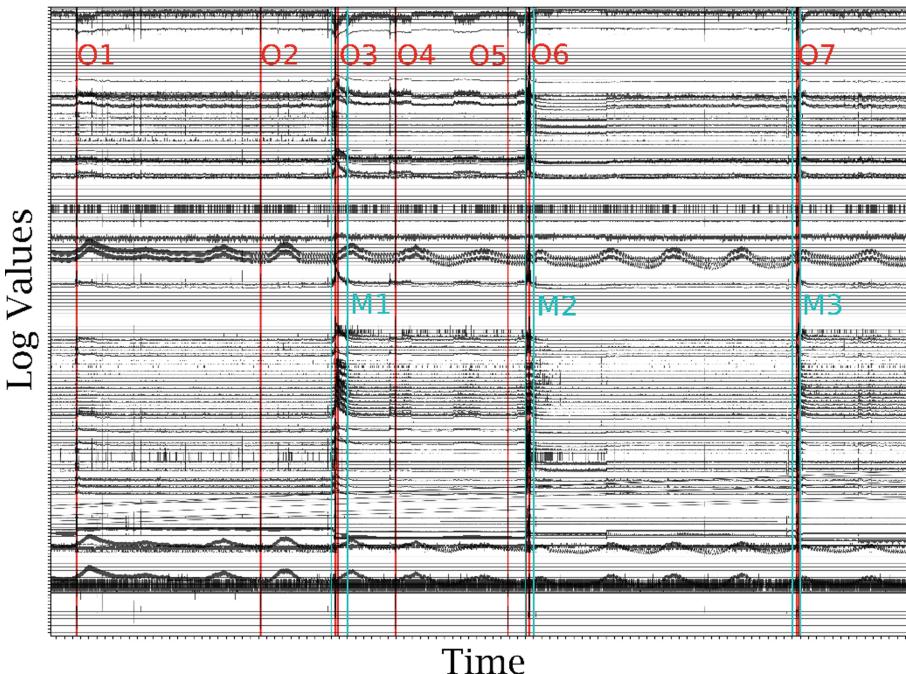


**Fig. 4.** Log-plotting: two weeks of operation

In Fig. 4, the reasonable outliers areas [M1-3] correspond with a change of material, through a new glass layer. M1-3 are marked cyan in Fig. 4. Figure 4 represents the converted log data in a single chart. All 183 columns are shifted along y axes to separate the graphs and enable manual visual recognition of peaks in log data. A portion of this data is used for training the autoencoder neural network.

In the marked areas [M1-3] the material had changed during operation and the inspection machine had to recalibrate its algorithms. Also, the peaks in the marked areas in Fig. 4 can be seen and detected by hand. Those outliers appear if something as small as the thickness of the inspected layer changed. As a test-case, our algorithm has to detect the three marked areas. The domain expert is interested in all outliers that can be detected in addition.

We present our results in Fig. 5. Outliers, that exceed the threshold (see Fig. 3), are marked in red. We name the outliers from left to right in order [O1-7]. Our proposed algorithm (see Fig. 3) is able to find the aforementioned peaks in log data, which correspond to changes of material. In addition, our algorithm has found even more outliers that a domain expert would not consider as outliers ex ante, which were verified as outliers ex post. Outliers O3, O6-7 match the marked areas, therefore our algorithm passed the test-case.



**Fig. 5.** Log-plotting: outliers detected (Color figure online)

Outliers O1-2 and O4-5 represent outliers that were unknown upfront. O1 and O4 were identified as a slight change of thickness in the inspection layer ex post and are valid outliers. O2 and O5 appear out of order and not directly connected physical visual outliers. A possible explanation is, that O2 and O5 can be seen as indicators for the marked outlier areas M1 and M2. This remains unverified for the moment, a study on more log data is planned.

Through the automatic outlier detection process we are able to mark outliers in log data. The marked log line can be mapped backwards to the original log line, which includes the timestamp. Therefore, the expert is enabled to do a more in-depth analysis of the log data around that timestamp. So, our method mitigate complexity and speed up problem solving.

## 6 Conclusion and Future Work

In this paper we presented an outlier detection method for temporal spatial log data that can be used without domain-specific knowledge. We present conversion rules that can be applied to log data independent of the application domain. Furthermore, we presented an autoencoder configuration. The configuration was found during field trials on the log data. In the end, we were able to find outliers in temporal spatial log data with our presented algorithm.

We found the outlier areas that were formerly known by the domain expert. In addition, we were able to find four additional outliers. Two outliers were verified as valid outliers ex post. The other outliers are in question; a study is planned to prove if the outliers can be seen as indicators for a major outlier peak. Another study on the autoencoder configuration will be conducted in order to prove if the configuration is suitable in other domains and on other log data. Through our study we were able to help a domain expert to identify more outliers in temporal spatial log data that were not considered before in a pre-analysis of the log data.

We have shown that autoencoder can be successfully used as an outlier detection method on log data. In further studies we plan to also incorporate contextual outlier detection using semi-supervised methods. A limitation of this study is that our method relies on the fact that outliers are only a portion of the whole log data, otherwise our autoencoder can reconstruct outliers with a minor reconstruction error than normal log data leading to a failure of our method. Another limitation is that we were only provided with log data of one production machine and get feedback only from one domain expert. We are aware of a potential bias and try to minimize and verify our results with more log data and more domain experts in future. Nevertheless, our method can be used as an additional tool for quick inspection in temporal spatial log data. In future this can speed up problem solving in a complex highly integrated connected CPS environment and reduce the time span of cost-intensive downtimes and non-functional CPSs.

**Acknowledgements.** This study takes place within the project ProDok 4.0, funded by the German Ministry of Education and Research (BMBF) within the framework of the Services 2010 action plan under funding no. 02K14A110. Executive steering committee is the Karlsruher

Institut für Technologie - Karlsruhe Institute of Technology (KIT). Project partners are KUKA AG, ISRA VISION AG, dictaJet Ingenieurgesellschaft mbH and Hochschule Darmstadt - University of Applied Sciences. Glass inspection machine (Type FS5D) logs provided by ISRA VISION AG. All rights on example log data remains solely to ISRA VISION AG. Usage must be requested.

## References

1. Fu, Y., Zhu, J., Gao, S.: CPS information security risk evaluation system based on Petri Net. In: DSC 2017, 2017 IEEE Second International Conference on Data Science in Cyberspace, Proceedings, 26–29 June 2017, Shenzhen, China, pp. 541–548. IEEE, Piscataway (2017)
2. Harada, Y., Yamagata, Y., Mizuno, O., Choi, E.-H.: Log-based anomaly detection of CPS using a statistical method. In: 8th IEEE International Workshop on Empirical Software Engineering in Practice, IWESEP 2017, Proceedings, 13 March 2017, Tokyo, Japan, pp. 1–6. Conference Publishing Services, IEEE Computer Society, Los Alamitos, California, Washington, Tokyo (2017)
3. Nguyen, H., Cai, C., Chen, F.: Automatic classification of traffic incident's severity using machine learning approaches. *IET Intell. Transp. Syst.* **11**, 615–623 (2017)
4. Hasani, Z.: Robust anomaly detection algorithms for real-time big data. Comparison of algorithms. In: Stojanović, R. (ed.) 2017 6th Mediterranean Conference on Embedded Computing (MECO). Including ECYPS 2017, Proceedings: Research Monograph, Bar, Montenegro, 11th–15th June 2017, pp. 1–6. IEEE, Piscataway (2017)
5. Lu, X., Nagelkerke, M., van de Wiel, D., Fahland, D.: Discovering interacting artifacts from ERP systems. *IEEE Trans. Serv. Comput.* **8**, 861–873 (2015)
6. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. *Science* **313**, 504–507 (2006)
7. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection. *ACM Comput. Surv.* **41**, 1–58 (2009)
8. Aggarwal, C.C.: Outlier Analysis. Springer, Cham (2017). <https://doi.org/10.1007/978-3-319-47578-3>
9. Ibidunmoye, O., Hernández-Rodríguez, F., Elmroth, E.: Performance anomaly detection and bottleneck identification. *ACM Comput. Surv.* **48**, 1–35 (2015)
10. Mehrotra, K.G., Mohan, C.K., Huang, H.: Anomaly Detection Principles and Algorithms. Springer, Cham (2017). <https://doi.org/10.1007/978-3-319-67526-8>
11. Chen, J., Sathe, S., Aggarwal, C., Turaga, D.: Outlier detection with autoencoder ensembles. In: Chawla, N., Wang, W. (eds.) Proceedings of the 2017 SIAM International Conference on Data Mining, pp. 90–98. Society for Industrial and Applied Mathematics, Philadelphia (2017)
12. Tang, L.-A., et al.: Trustworthiness analysis of sensor data in cyber-physical systems. *J. Comput. Syst. Sci.* **79**, 383–401 (2013)
13. Protopapadakis, E., Voulodimos, A., Doulamis, A., Doulamis, N., Dres, D., Bimpas, M.: Stacked autoencoders for outlier detection in over-the-horizon radar signals. *Comput. Intell. Neurosci.* **2017**, 5891417 (2017)
14. Nolle, T., Seeliger, A., Mühlhäuser, M.: Unsupervised anomaly detection in noisy business process event logs using denoising autoencoders. In: Calders, T., Ceci, M., Malerba, D. (eds.) Discovery Science: 19th International Conference, DS 2016, Bari, Italy, October 19–21, 2016, Proceedings, pp. 442–456. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-46307-0\\_28](https://doi.org/10.1007/978-3-319-46307-0_28)

15. Lu, W., et al.: Unsupervised sequential outlier detection with deep architectures. *IEEE Trans. Image Process.* **26**, 4321–4330 (2017)
16. Xiong, Y., Zuo, R.: Recognition of geochemical anomalies using a deep autoencoder network. *Comput. Geosci.* **86**, 75–82 (2016)
17. Sebestyen, G., Hangan, A.: Anomaly detection techniques in cyber-physical systems. *Acta Univ. Sapientiae Inform.* **9**, 116 (2017)
18. Qu, Y., et al.: Product-based neural networks for user response prediction over multi-field categorical data (2018)
19. Rodríguez, P., Bautista, M.A., González, J., Escalera, S.: Beyond one-hot encoding: lower dimensional target embedding. *Image Vis. Comput.* **75**, 21–31 (2018)
20. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS 2010). Society for Artificial Intelligence and Statistics (2010)
21. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th International Conference on International Conference on Machine Learning, pp. 807–814. Omnipress, USA (2010)
22. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2014)



# Reservoir Computing Approaches Applied to Energy Management in Industry

Valentina Colla<sup>(✉)</sup>, Ismael Matino, Stefano Dettori, Silvia Cateni,  
and Ruben Matino

TeCIP Institute, ICT-COISP Center, Scuola Superiore Sant'Anna, Pisa, Italy  
[valentina.colla@santannapisa.it](mailto:valentina.colla@santannapisa.it)

**Abstract.** Echo-State Neural Networks represent a very efficient solution for modelling of dynamic systems, thanks to their particular structure, which allows faithful reproduction of the behavior of the system to model with a usually limited computational burden for a training phase. This aspect favors the deployment of Echo-State Neural networks in the industrial field. In this paper, a novel application of such approach is proposed for the modelling of industrial processes. The developed models are part of a complex system for optimizing the exploitation of process off-gases in an integrated steelwork. Two models are presented and discussed, where both shallow Echo-State Neural Networks and Deep Echo State Neural networks are applied. The achieved results are presented and discussed, by comparing advantages and drawbacks of both approaches.

**Keywords:** Echo-State Neural Networks · Modelling · Off-gas management · Energy management · Steel industry

## 1 Introduction

In the last decade, the environmental consciousness of the society increased and the environmental regulation became more severe in most European countries by pressurizing industries and communities toward energy and resource efficiency. The concept of “resource” includes primary raw materials and energy, but also by-products, wastes, wastewater and off-gases.

In particular, the off-gases produced during the different industrial processes such as the integrated steelmaking cycle are often very valuable, as they are a source of energy, which can replace natural gas for the production of heat, electric energy or steam. In integrated steelworks, these gases come from the main production steps, i.e. coke production (coke ovens), pig iron production (Blast Furnace - BF) and pig iron conversion to steel through oxygen insufflation (Basic Oxygen Furnaces - BOF). Currently these gases are internally exploited but their management is often non-optimal due to technical and process-related constraints as well as to discrepancies in the plant production scheduling. The poor coordination in management of gas production and consumption can lead to either over-production or under-production of off-gases. In the first condition, the off-gases are stored in gasholders until they are full and afterwards they are flared (or, in some facilities, the production is stopped), with

consequent emissions and economic losses. In the second case, the off-gases cannot satisfy the internal demand and natural gas is exploited, with consequent consumption of a primary source and associated costs. These issues are enhanced in intermittent processes, such as the BOF.

The off-gases production is unavoidable, but their wastage can be reduced through the development of solutions allowing their optimal management and exploitation, with consequent reduction of economic and environmental impacts, of CO<sub>2</sub> emissions and of primary resources exploitation. In the past, some Decision Support Systems (DSS) were developed to support plant technicians in the off-gas management by also incorporating sophisticated models computing the amount and energy content of the off gasses produced in a the integrated steelmaking cycle (the steelmaking route which produces steels from virgin primary raw material, i.e. basically iron ore and coke) [1–3]. However, this approach provides off-line indications, considers only limited number of constraints, neglect power generation systems as gas users and does not consider dynamic prediction of process gas production and use. The ongoing EU-funded GASNET project aims at filling this gap through the development of a library of models to forecast the production of main off-gases in terms of volume flowrate and intrinsic gas energy and the related consumptions by the main consumers processes in the time horizon of two hours with a sampling rate of 1 min. Some of these models exploit Echo State Neural Networks (ESN) either in their standard or in their “deep” version (DESN).

Some solutions can be found in literature, which exploit back propagation Neural Networks (NN) [4], improved least squares Support Vector Machine (SVM) [5] or multiple linear regression model [6] to forecast the Blast Furnace Gas (BFG) generation. However, none of them provides information about the energetic value of the gases, as the CO and H<sub>2</sub> content are not provided and thus the Net Calorific Value (NCV) of the gas cannot be computed. On the other hand, no literature works are available concerning the prediction of BOF Gas (BOFG). This gas has a higher NCV with respect to BFG and, thus, is more interesting from recovery point of view. However, the converter process is more difficult to model, being highly dynamic. On this subject, ESNs prove their capability to model efficiently complex dynamic processes, being also characterized by a cost-efficient training procedure, which is a fundamental element for the future deployment of the system in an industrial context.

In the proposed application, the design of the ESN-based models was supported by a deep analysis and careful selection of the input data, which was fundamental in order to select the most relevant variables and eliminate useless or redundant data.

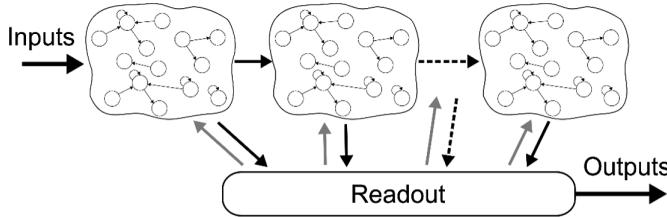
The Paper is organized as follows: Sect. 2 provides some theoretical background on ESN, Sect. 3 describes the necessary data pre-processing stages, Sect. 4 provide details on the developed models, while Sect. 5 presents and discusses the models performances. Finally Sect. 6 provides some concluding remarks and hints for future work.

## 2 Theoretical Background on Echo-State Neural Networks

Between reservoir computing architectures, ESN is well known as a very effective approach to model complex nonlinear time dependencies. The ESN structure consists of a reservoir layer, the only recurrent part of the NN, and a readout output layer.

The main idea that characterizes the ESN approach is the possibility to draw on a rich set of dynamics, generated by the reservoir, to make regression on a target, through supervised algorithms that, unlike other recurrent neural network (RNN) structure, do not use iterative routines such as back-propagation through time (BPTT) that suffer from several numerical issues, such as exploding or vanishing gradients [7]. In general, RNN are considered universal “approximators” [8], but in time-critical applications it is important to overcome the computational training costs and potentially slow convergence. In literature, several works show the efficiency of ESN, starting from the work of Jaeger [9, 10], up to the recent research results that demonstrated, from a mathematical point of view, that ESN are universal uniform approximants [11].

A further leap forward in the study of increasingly performing architectures consisted in the development of the DESN-based approach [12, 13]. The main idea is the exploitation of a deep layered version of  $N$  reservoirs connected in series, as shown in Fig. 1, which can result in a great accuracy improvement as demonstrated by several deep learning architectures and applications [14]. ESNs have become a special case of DESN with only one reservoir layer. By starting from this assumption, it is possible to generalize the mathematical background focusing on the DESN equations.



**Fig. 1.** Architecture of a Deep Echo-State Network.

Moreover, the reservoir layers generate dynamics starting from an exciting input vector  $\mathbf{u}(k)$  composed of  $n_l$  features. The state of the entire reservoir system is composed of the state of each reservoir, as follows:

$$\mathbf{x}(t) = [\mathbf{x}_1(t) \quad \cdots \quad \mathbf{x}_N(t)] \quad (1)$$

Without losing the generality, we can think that each reservoir comprises of  $n$  neurons. The state of the first reservoir is computed as:

$$\mathbf{x}_1(t) = f_x(\mathbf{W}_{in_1}\mathbf{u}(t) + \mathbf{W}_{r_1}\mathbf{x}_1(t-1) + \mathbf{W}_{f_1}\mathbf{y}(t-1) + \mathbf{v}_1(t)) \quad (2)$$

Where  $f_x$  is a nonlinear function (in general a tanh function), the input matrix  $\mathbf{W}_{in_i}$  is a  $n \times n_I$  matrix, the reservoir matrix  $\mathbf{W}_r$  a  $n \times n$  matrix, the feedback matrix  $\mathbf{W}_{f_i}$  a  $n \times n_y$  matrix,  $\mathbf{y}$  is the output of the ESN and  $\mathbf{v}$  is a small amplitude white noise.

The state of  $i$ -th reservoir layer is computed starting from the state of the  $(i - 1)$ -th layer as:

$$\mathbf{x}_i(t) = f_x(\mathbf{W}_{in_i} \mathbf{x}_{i-1}(t) + \mathbf{W}_r \mathbf{x}_i(t-1) + \mathbf{W}_{f_i} \mathbf{y}(t-1) + \mathbf{v}(t)) \quad (3)$$

The output of the readout layer is computed as:

$$\mathbf{y}(t) = f_y(\mathbf{W}_y \mathbf{x}(t)) \quad (4)$$

where  $f_y$  is the neuron function, in general linear,  $\mathbf{W}_y$  a  $n_y \times N$  matrix.

One of the most important characteristics of the DESN lies in the effectiveness of the training concept, whose strength aims at eliminating iterative calculation procedures by splitting training in two distinct and different sequential phases:

- **The Reservoir system initialization:** in order to guarantee generalization, the reservoir system must verify a series of properties that can be summarized as follows: (I) The reservoir state has to be contractive and guarantee stability properties; (II) It has to generate a sufficiently rich set of dynamics, among which we can choose those that best characterize the system we want to model.  
More in detail, as regards the first point, in the first phase each reservoir layer matrix  $\mathbf{W}_{r_i}$  is initialized with a sparse randomized  $\mathbf{W}_{r_i}^{rand}$  with elements in the range  $[-1, 1]$ . The matrix sparsity defines the percent number of synapse connections within the reservoir and, in general, can be set within the range 1–5%. In order to satisfy stability properties, the overall reservoir system has to be characterized by the state “contractivity”, state forgetting and input forgetting properties, which are summarized in the so-called Echo State Property (ESP). The ESP has been widely studied in several works in literature, starting from seminal papers of Jaeger [9, 15], up to the definition of stability for the case of DESN [12], in which necessary and sufficient conditions are defined and discussed. A necessary condition to guarantee the ESP can be summarized as follows:

$$\rho_g = \max(\rho(\mathbf{W}_{r_i})) < 1, i = 1, 2, \dots, N \quad (5)$$

where  $\rho(\mathbf{W}_{r_i})$  is the spectral radius of each reservoir matrix. It is important to note that the condition is only sufficient and ESP must be verified empirically for the designed network, but in general in the majority of the case studies, the necessary condition is also empirically sufficient.

In order to verify ESP, once  $\mathbf{W}_{r_i}$  is initialized has to be scaled to obtain the desired spectral radius  $\rho_i$ , as follow:

$$\mathbf{W}_{r_i} = \mathbf{W}_{r_i}^{rand} \frac{\rho_i}{\rho(\mathbf{W}_{r_i}^{rand})} \quad (6)$$

Each reservoir input matrix  $\mathbf{W}_{in_1} \cdots \mathbf{W}_{in_N}$  is initialized randomly with weights in the range  $[-1, 1]$ , and then scaled by a factor  $K_i$ , known as **input scaling**.

To verify the second requirement, firstly, the reservoir system must have a sufficiently large number of neurons, secondly both number of layers and synapse connections within the reservoir system must be ad hoc designed. Several heuristics and algorithms for the selection of these hyperparameters are based on cross-validation techniques, or other less costly techniques from a computational point of view. Some interesting examples are described in [16], where several hints are suggested to set the hyperparameters in a shallow ESN, in [17], where a Bayesian optimization is used to determine a good set of hyperparameters, and in [18]. Here spectral analysis is exploited to determine the number of layers of the DESNs and Intrinsic Plasticity is used to increase the richness of nonlinear dynamics within each reservoir. Once each reservoir layer has been initialized and verifies ESP, the weights are left untrained and remain stationary.

- **Readout Training:** the linear readout is trained with the objective of minimizing the 2-norm of the training dataset error, as follows:

$$\min \|\mathbf{W}_y \mathbf{X} - \mathbf{Y}_{target}\|_2$$

where  $\mathbf{X} = [\mathbf{x}(1) \cdots \mathbf{x}(T_{end})]$  and  $\mathbf{Y}_{target} = [\mathbf{y}(1) \cdots \mathbf{y}(T_{end})]$  are respectively the state and target collection matrixes. In particular,  $\mathbf{X}$  is calculated starting from the initialized reservoir system by using Eqs. 2 and 3. In the case of the presence of output feedback ( $\mathbf{W}_{fi} \neq 0$ ), the teacher forcing is a possible solution to compute the state evolution for each time step.

The minimization problem can be solved with a ridge regression algorithm that implies the regularization of a simple pseudo-inversion, as follows:

$$\mathbf{W}_y = \mathbf{Y}_{target} \mathbf{X}^T (\mathbf{X} \mathbf{X}^T + \lambda \mathbf{I})^{-1} \quad (7)$$

### 3 Data Pre-processing

The number of collected variables, which are somehow related to the off-gas generation is huge (in the order of hundreds). In order to develop effective models, the careful selection of the input variables for each model is fundamental: to this aim, plant staff experience and process knowledge need to be exploited together with appropriate pre-processing analysis including outliers' removal and feature space reduction.

#### 3.1 Outlier Removal

The detection and removal of outliers is an important pre-processing step, which allows selection of reliable data for models development. Outliers are samples that can be labelled as anomalous due to different causes (e.g. wrong measurements, faulty sensors).

Traditional outlier detection methods can be classified in four categories: distribution-based, distance-based, density based and clustering-based. Distribution-based method is usually exploited in the statistic application, it defines an outlier as a sample that does not fit in a reasonable way with a standard distribution. The most popular distribution-based approach is the Grubbs test [19], which adopts the Normal distribution function. Distance-based approach was presented by Knorr and Ng considering as outlier “*An object  $x$  in a dataset  $T$  is a  $DB(p, D)$ -outlier if at least fraction  $p$  of the objects in  $T$  lie at a distance greater than  $D$  from  $x$* ” [20], distance approach presume to fix a definition of distance. Clustering-based approach considers as an outlier a sample, which does not belong to any cluster after an appropriate clustering operation. Finally density-based method, assigns to each sample data a “degree of outliersness” called *Local Outlier Factor (LOF)*. The LOF specifies how a data point is isolated with respect to his neighborhood. In this application, an approach which combines 3 different outlier detection approaches (distribution-base, distance-based and clustering-based) through a FIS is exploited. The density-based method has been excluded due to its high computational cost in case of a large amount of data.

In the present application, in order to detect the outliers in the available dataset, a fuzzy-logic based method is exploited [21, 22]. This method, which combines the advantages of the traditional approaches by avoiding their drawbacks, is more efficient than the single traditional approach. Three features are computed through the following conventional techniques: Distribution based method based on Grubbs algorithm, distance based method is determined by computing the Mahalanobis distance [23] among each instance and the rest of the data, clustering based method is evaluated by applying the fuzzy C-means technique and specifies the degree of membership of each data to the allocated cluster. The three features are fed as inputs to a Mandami FIS and the output provides a “degree of outliersness”. Finally, a threshold is added to identify if each data point can be considered as an outlier or not.

Once identified, the outliers need to be replaced by appropriate values, especially when dealing with time-series data. In literature, several techniques are proposed to this purpose; in this work, a sample classified as outlier is replaced by a sample, which maintain the previous consistent value.

### 3.2 Feature Space Reduction

Feature space reduction consists in converting a wide dataset into a more reduced one by holding most of the significant information content of the initial dataset. In this analysis, the reduction is reached through the following two main operations:

- elimination of highly correlated variables for removing redundancy;
- selection of the input variables which mostly affect the considered target.

The selection of the most relevant variables representing by a subset of informative and uncorrelated variables improves the prediction accuracy.

Redundant variables can be detected through the dominating set algorithm, a method derived from the graph theory [24]. The dominating set algorithm identifies the highly correlated input variables using the definition of graph where the vertexes are associated to the variables: the links among such vertexes are established among

variables, which indicate a linear correlation coefficient greater than 0.95 with a  $p$  value lower than 0.05. Afterwards, the algorithm selects the minimum dominating set of the built graph, according to the following definition: “*Given a graph  $G = (V, E)$ , a dominating set is defined as a subset  $V' \subseteq V$  such that every vertex not in  $V'$  is adjacent to at least one member of  $V'$ .*” Redundant variables are also defined as those variables whose corresponding vertexes of the graph are not included in the minimal size dominating set.

After elimination of the redundant variables, a Variable Selection (VS) algorithm is applied on the remaining ones in order to select only the variables, which actually affect the considered process, as, if irrelevant input variables are included, the system performances can decrease. VS approaches, which are commonly applied in Machine Learning (ML), can be classified into 3 main categories:

- filter methods, which select variables considering the relation between input and output of the system and independently from the learning algorithm;
- wrapper methods, which use the learning machine as a black box by selecting variables on the basis on their predictive influence.
- embedded methods, which implement VS within the training process. Embedded approaches show comparable performance to wrapper-based ones, but need ad-hoc designed learning systems.

Generally, when dealing with large datasets filter methods are preferable, as they are less computational cumbersome. However, if the dataset has a reasonable dimension, wrapper and embedded methods are preferable, as they reach a higher accuracy with respect to filter approaches, as they take into account the ML model.

The ideal (in terms of search space) VS method is the analysis of all combinations of potential input variables, the so-called exhaustive search or brute force method. This approach belongs to the wrapper class but it is not viable when a significant number of variables are involved, being its computational time complexity exponential. When the number of potential input variables is significant, evolutionary approaches can be an appropriate compromise allowing to cover a large combination of variables. In this application an efficient approach based on Genetic Algorithms (GAs), is applied [25, 26]. Each variable subset is characterized by a binary chromosome of the GA (each gene corresponds to an input variable and its value is 1, if the associating variable is included in the considered subset, 0 otherwise). The GA creates a population of several chromosomes and estimates their goodness by evaluating a *fitness-function*, which corresponds to the model performance when trained by exploiting the selected input variables subset. The best chromosomes are exploited, through GA *crossover* and *mutation* procedures, to create a new population: *crossover* generates the son chromosome by randomly selecting the genes values from the two parents; *mutation* creates new individuals by randomly switching the binary value of a gene of the considered chromosome. When a new population is generated, the evaluation and reproduction process is repeated until a stop condition, which, in this application, consists of the achievement of a fixed number of iterations or of a plateau for the fitness function.

Redundancy analysis and VS allowed creating a subset of significant variables.

## 4 Developed Models

Hereby we present some exemplar models of industrial processes, which show the effectiveness of the proposed approach in industrial applications. In particular, the BOF process has been studied in depth with several modelling objective:

- **Model 1:** prediction of the BOFG production;
- **Model 2:** prediction of the steam production by recovering the BOFG heat.

An ESN-based model is proposed forecasting the recoverable BOFG (i.e. volume and heating power) and providing a qualitative idea of flareable and total BOFG volume flows and heating powers with a frequency of 1 min. in a time horizon of 2 h.

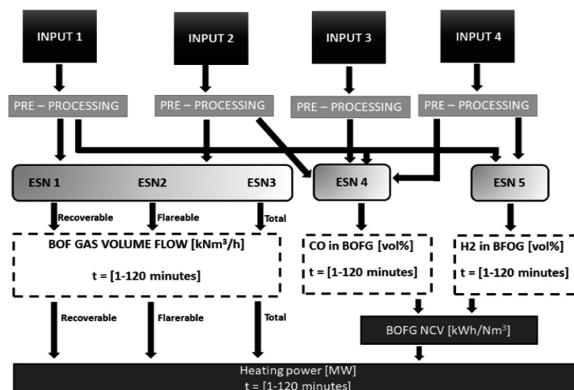
The qualitative feature of the estimate of flareable and total BOFG volume flows and heating powers is affected by the performance of data associated to the flared gas, which is not comparable to the performance of the data associated to the recovered gas and does not allow a good performance training of the model for these outputs. Moreover, in order to ensure proper exploitation of BOFG gas, the knowledge of the behavior of recoverable gas with respect to the flareable gas (part of gas part that need to be flared due to process and security constraints) is more interesting.

The model includes five different ESNs that require to be separately tuned in order to independently forecast recoverable, flareable and total BOFG, CO and H<sub>2</sub> contents of the gas. Each network is specialized and, thus, the computational burden in the training phase is moderate. The prediction of the CO and H<sub>2</sub> content in the BOFG is exploited to estimate the NCV according to the following equation:

$$NCV = \frac{(0.0789 \cdot CO + 0.0672 \cdot H_2) \cdot 1000}{22.4} \text{ kWh/Nm}^3 \quad (8)$$

The achieved NCV value and the BOFG volume allow computing the heating power.

The general scheme of the model is shown in Fig. 2.



**Fig. 2.** Scheme of the model developed to predict BOFG production

The inputs of the model, which have been selected through VS and redundancy analysis, are reported in Table 1 with the related cluster and other useful information. They mostly refer to materials and Oxygen which are fed to the BOF. The position of the movable skirt is also considered, which is an operating variable affecting the air feed of the process.

Most of the inputs are common among the different ESNs, while some inputs associated to the charged materials are fed only to some ESNs.

The variable named “*BOF Plant Unit States*” is a sort of Boolean representation of the scheduling of the process: when it hold a unitary value, the converter is in operation (start of the blowing of the oxygen), otherwise oxygen blowing is not occurring.

The second model forecasts the total steam production related to the scheduling of the BOF process. The model has a forecasting period of 2 h ahead, starting from the most correlated variables. The list of inputs is shown on Table 2. These variables affect the transfer of the heat which is exploited for steam production. The dynamic of the steam production is modeled through the use of the novel DESN-based architecture.

**Table 1.** Input variables of the ESNs for the BOFG production forecasting (freq. 1 min,  $t = 0$  stands for current value,  $t = 0 \dots 120$  stands for all values in the next 2 h).

Variable	Meas. unit	Input cluster	Note
Position of moveable skirt	mm	Input 1	$t = 0$
Blowed Oxygen Volume Flow	kNm <sup>3</sup> /h	Input 1	$t = 0$
Recovered BOFG Vol. Flow	kNm <sup>3</sup> /h	Input 1	$t = 0$
Flared BOFG Volume Flow	kNm <sup>3</sup> /h	Input 1	$t = 0$
BOF Plant Unit State	Boolean	Input 1	$t = 0 \dots 120$
Charged Liquid Pig iron	t	Input 1	$t = 0 \dots 120$
Charged Lime	t	Input 2	$t = 0 \dots 120$
CO content in BOFG	vol.%,	Input 3	$t = 0$
H <sub>2</sub> content in BOFG	vol.%,	Input 4	$t = 0$
Recycled briquetted sinter	t	Input 4	$t = 0 \dots 120$

**Table 2.** Input variables for the prediction of steam production in BOG process.

Variable	Measurement unit	Note
Position of moveable skirt	mm	$t = 0$
Blowed Oxygen Volume Flow	kNm <sup>3</sup> /h	$t = 0$
Recovered BOFG Volume Flow	kNm <sup>3</sup> /h	$t = 0$
Flared BOFG Volume Flow	kNm <sup>3</sup> /h	$t = 0$
BOF Plant Unit State	Boolean	$t = 0 \dots 120$
Production of steam in the BOF boiler	t/h	$t = 0$

Before the training and the simulation of the models, outliers are removed.

All the presented ESN and DESN-based architectures are designed according to a common approach, which can be summarized as follows:

- reservoir units connection is low, with a sparsity between 2–4%;
- reservoir and readout activation functions are, respectively, the hyperbolic tangent and a linear function;
- the tuned hyper-parameters are: number of neurons of reservoir, spectral radius, scaling of input and of teacher (i.e. the targets of the training phase), noise level, input and teacher shifting, regularization parameter for the ridge regression of the readout (training phase). The mentioned hyper-parameters have been heuristically tuned with the objective of minimizing the Normal Root Mean Square Error (*NRMSE*) on the validation dataset, defined as:

$$\text{NRMSE} = 100 \frac{\sqrt{\frac{1}{N_s} \sum_{i=1}^{N_s} (\tilde{y}_i - y_i)^2}}{\max(y) - \min(y)} \quad (8)$$

where  $N_s$  is the number of the samples of the dataset,  $y_i$  and  $\tilde{y}_i$  are respectively the  $i$ -th sample of the target and the predicted output.

- The training algorithm is an offline ridge regression of the readout;

## 5 Experimental Results

### 5.1 Available Data and Performance Index

All models have been trained and validated by using real data coming from the industrial facilities. Large datasets have been acquired with a sampling time of 1 min and split into training and validation sets, holding, respectively, 70% and 30% of the available data. In the case of the first model (BOFG production) the dataset is composed of 450000 data coming from 10 months of production. For the second model the dataset is composed of 252000 data associated to 6 months of production.

In order to evaluate the performances and find the best solution for modeling different behaviors of the process, both shallow and DESN architecture have been tested. NRMSE and Training time are considered for efficiency assessment.

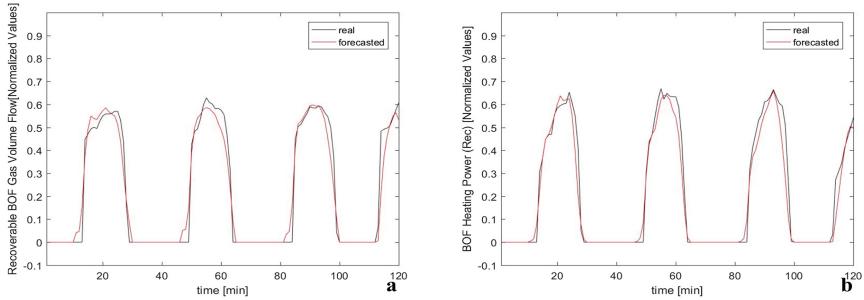
### 5.2 Numerical Results

The validation results are shown in Table 3. For confidentiality constraints, all results and graphs are normalized with respect to their ranges.

**Table 3.** Validation results of the developed models

Model	Approach	Output variables	NRMSE
Model 1	ESN	BOFG recoverable volume flow	7 ÷ 14%
	ESN	BOFG heating power	6 ÷ 13%
Model 2	DESN	Steam Production in the BOF boilers	3.2 ÷ 6%

In the case of BOFG, the prediction performance is very satisfactory in both cases (i.e. shallow and DESNs): the BOFG model allows faithfully following the amount of recoverable gas and associated heating power. This last quantity is evaluated by exploiting the predicted CO and H<sub>2</sub> contents in the gas that are estimated by the model with a good performance. The forecast of flareable gas is less precise but the qualitative behavior is followed in a satisfactory way. Finally, the model provides an intermediate accuracy for the prediction of the total amount of BOFG volume flow and heating power, which indicates the predicted sum of the recoverable and flareable BOFG. Figure 3 depicts the prediction of the most important outputs (i.e. recoverable BOFG and related heating power) by using shallow ESN and shows the model capability to follow the intermittent behavior of the process under consideration without any delay.



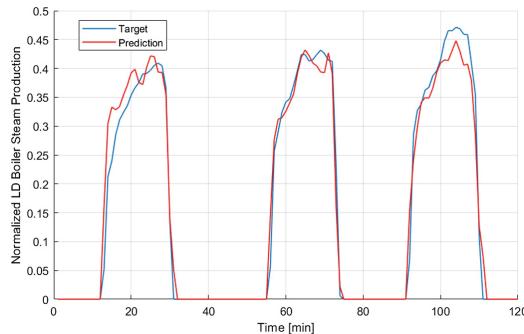
**Fig. 3.** Comparison between real (black) and forecasted (red) values by the model based on shallow ESN of recoverable BOFG gas: (a) volume flow; (b) heating power. (Color figure online)

The suitability of this model is reflected in the NRMSE that lies in the range 7  $\div$  14% for the recoverable gas volume flow and 6  $\div$  13% for the related heating power; the lowest value of the error corresponds to values which are closer in time. However, the error values are in part distorted, as in some applications the estimated recoverable BOFG is dissimilar from the actually recovered portion. This fact could occur when there is a specific situation, in which the gas was recoverable but for some reasons (e.g. the gasholder was full or for other unusual conditions) the operators decided to flare it anyway. Consequently, the error measures are not totally representative of the accuracy of the models, which are considered appropriate for forecasting in a quantitative way the gas amount and its intrinsic energy for recovery purposes.

Using DESNs in the BOFG model provides an almost identical value of the NRMSE. In both cases, the training time is reasonable but the DESN required almost 75% of the training time with respect to the shallow ESNs. On the other hand, the DESN computational burden in this case study is higher than the shallow ESN one.

Also the validation results for the second model are very encouraging. An example of the prediction of future 120 min of BOF Boiler Steam Production is shown in Fig. 4: the trained model allows to predict the variable of interest for 2 h ahead, with an accuracy in the range 3.2  $\div$  6% in the case of the DESN-based approach and

$3.5 \div 6\%$  in the case of shallow ESN. The error is low in the first few minutes of prediction and, as it is normally expected, it tends to increase, as we want to predict the phenomenon in the distant future. Both shallow ESN and DESN show very good performances also for the prediction of 2 h ahead, but in this case the DESN-based model proves to be the best solution both in terms of accuracy and computation time.



**Fig. 4.** DESN-based prediction of BOF Steam production: real (blue) vs. forecasted values (red). (Color figure online)

## 6 Conclusions

An application of ESNs in their shallow and Deep versions to the modelling of industrial processes is presented and discussed, by proving their effectiveness and excellent capability to reproduce the behavior of dynamic systems. Both the standard ESN and the DESN provide good results in terms of forecasting accuracy: in some cases DESNs provides slightly better performances, but their most relevant advantage in the present application is represented by the shorter training time, which is achieved at the price of a more relevant computational burden. Therefore, the final choice between the two approaches is driven not only by the target performance accuracy, but by the available time for model training and by the available computational resources.

Ongoing and future work concerns the deployment of the proposed modelling approach in areal industrial context within a Decision Support System integrating different process models and aimed at providing suggestion for the management of off-gases.

**Acknowledgments.** The work described in the present paper was developed within the project entitled “Optimization of the management of the process gases network within the integrated steelworks - GASNET” (Contract No. RFSR-CT-2015-00029) and received funding from the Research Fund for Coal and Steel of the European Union, which is gratefully acknowledged. The sole responsibility of the issues treated in the present paper lies with the authors; the Union is not responsible for any use that may be made of the information contained therein.

## References

1. Porzio, G.F., et al.: Reducing the energy consumption and CO<sub>2</sub> emissions of energy intensive industries through decision support systems—an example of application to the steel industry. *Appl. Energy* **112**, 818–833 (2013)
2. Porzio, G.F., et al.: Process integration in energy and carbon intensive industries: an example of exploitation of optimization techniques and decision support. *Appl. Therm. Eng.* **70**(2), 1148–1155 (2014)
3. Porzio, G.F., Nastasi, G., Colla, V., Vannucci, M., Branca, T.A.: Comparison of multi-objective optimization techniques applied to off-gas management within an integrated steelwork. *Appl. Energy* **136**, 1085–1097 (2014)
4. Zhang, Q., Gu, Y.L., Ti, W., Cai, J.J.: Supply and demand forecasting of blast furnace gas based on artificial neural network in iron and steel works. *Adv. Mat. Res.* **443**, 183–188 (2012)
5. Yang, L., He, K., Zhao, X., Lv, Z.: The prediction for output of blast furnace gas based on genetic algorithm and LSSVM. In: IEEE 9th Conference on Industrial Electronics and Applications, pp. 1493–1498 (2015)
6. Zhao, J., Wang, W., Liu, Y., Pedrycz, W.: A two-stage online prediction method for a blast furnace gas system and its application. *IEEE Trans. Control Syst. Tech.* **19**(3), 507–520 (2011)
7. Pascanu, R., Mikolov, T., Bengio, Y.: On the difficulty of training recurrent neural networks. In: International Conference on Machine Learning, pp. 1310–1318, February 2013
8. Schäfer, A.M., Zimmermann, H.G.: Recurrent neural networks are universal approximators. *Int. J. Neural Syst.* **17**(04), 253–263 (2007)
9. Jaeger, H.: The, “echo state” approach to analysing and training recurrent neural networks—with an erratum note. Bonn Ger.: Ger. Natl. Res. Cent. Inf. Technol. GMD Tech. Rep. **148** (34), 13 (2001)
10. Jaeger, H., Haas, H.: Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science* **304**(5667), 78–80 (2004)
11. Grigoryeva, L., Ortega, J.P.: Echo state network are universal. *Neural Netw.* **108**, 495–508 (2018)
12. Gallicchio, C., Micheli, A., Pedrelli, L.: Deep reservoir computing: a critical experimental analysis. *Neurocomputing* **268**, 87–99 (2017)
13. Gallicchio, C., Micheli, A.: Echo state property of deep reservoir computing networks. *Cogn. Comput.* **9**(3), 337–350 (2017)
14. Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y.: Deep Learning, vol. 1. MIT Press, Cambridge (2016)
15. Yıldız, I.B., Jaeger, H., Kiebel, S.J.: Re-visiting the echo state property. *Neural Netw.* **35**, 1–9 (2012)
16. Lukoševičius, M.: A practical guide to applying echo state networks. In: Montavon, G., Orr, G.B., Müller, K.-R. (eds.) *Neural Networks: Tricks of the Trade*. LNCS, vol. 7700, pp. 659–686. Springer, Heidelberg (2012). [https://doi.org/10.1007/978-3-642-35289-8\\_36](https://doi.org/10.1007/978-3-642-35289-8_36)
17. Maat, J.R., Gianniotis, N., Protopapas, P.: Efficient optimization of echo state networks for time series datasets. In: International Joint Conference on Neural Networks (IJCNN), pp. 1–7 (2018)
18. Gallicchio, C., Micheli, A., Pedrelli, L.: Design of deep echo state networks. *Neural Netw.* **108**, 33–47 (2018)
19. Grubbs, F.E.: Procedures for detecting outlying observation sin samples. *Technometrics* **11**, 1–21 (1969)

20. Knorr, E.M., Ng, R.: Algorithms for mining distance-based outliers in large datasets. In: Proceedings of VLDB, pp. 392–403 (2003)
21. Cateni, S., Colla, V., Nastasi, G.: A multivariate fuzzy system applied for outliers detection. *J. Intell. Fuzzy Syst.* **24**(4), 889–903 (2013)
22. Cateni, S., Colla, V., Vannucci, M.: A fuzzy logic-based method for outliers detection. In: Proceedings of the IASTED International Conference on Artificial Intelligence and Applications, AIA 2007, pp. 561–566 (2007)
23. Mahalanobis, P.C.: On the generalized distance in statistics. *Proc. Natl. Inst. Sci. India* **4**, 9–55 (1936)
24. Grandoni, F.: A note on the complexity of minimum dominating set. *J. Discrete Algorithms* **4**(2), 209–214 (2006)
25. Cateni, S., Colla, V., Vannucci, M.: General purpose input variables extraction: a genetic algorithm based procedure GIVE a GAP. In: 9th International Conference on Intelligent Systems Design and Applications, ISDA 2009, pp. 1278–1283 (2009)
26. Cateni, S., Colla, V., Vannucci, M.: A genetic algorithm-based approach for selecting input variables and setting relevant network parameters of a SOM-based classifier. *Int. J. Simul. Syst. Sci. Technol.* **12**(2), 30–37 (2011)



# Signal2Vec: Time Series Embedding Representation

Christoforos Nalmpantis<sup>(✉)</sup> and Dimitris Vrakas

School of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece  
[{christofn,dvrakas}@csd.auth.gr](mailto:{christofn,dvrakas}@csd.auth.gr)  
<https://www.csd.auth.gr/en/>

**Abstract.** The rise of Internet-of-Things (IoT) and the exponential increase of devices using sensors, has lead to an increasing interest in data mining of time series. In this context, several representation methods have been proposed. Signal2vec is a novel framework, which can represent any time-series in a vector space. It is unsupervised, computationally efficient, scalable and generic. The framework is evaluated via a theoretical analysis and real world applications, with a focus on energy data. The experimental results are compared against a baseline using raw data and two other popular representations, SAX and PAA. Signal2vec is superior not only in terms of performance, but also in efficiency, due to dimensionality reduction.

**Keywords:** Time series · Data mining · Representations · Time series classification · Energy embeddings · Non intrusive load monitoring

## 1 Introduction

Time series is a sequence of data in time order, with values in continuous space. The order can be irrelevant to time, but it is still important. This type of data has always attracted the interest of scientists in a vast range of areas such as speech recognition, finance, physics, biology etc. Some common tasks involving time series are: motif discovery, forecasting, source separation, subsequence matching, anomaly detection and segmentation.

In time series problems, regardless the approach, the performance of the solution is heavily affected by the representation of the data. The categories of representations can be classified into data adaptive, non-data adaptive, model-based

---

This work has been funded by the ΕΣΠΑ (2014–2020) Erevno-Dimiourgo-Kainotomo 2018/EPAnEK Program 'Energy Controlling Voice Enabled Intelligent Smart Home Ecosystem', General Secretariat for Research and Technology, Ministry of Education, Research and Religious Affairs.

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

and data dictated. The first one includes techniques such as Adaptive Piecewise Constant Approximation [13], Singular Value Decomposition [15], Symbolic Natural Language [23], Symbolic Aggregate ApproXimation [16]. Approaches, which belong to the second representation, are: Discrete Wavelet Transform [5], spectral DFT [9], Piecewise Aggregate Approximation [14] and Indexable Piecewise Linear Approximation [6]. Model based representations are based on statistics such as Markov Models and Hidden Markov Model [19] and Auto-Regressive Moving Average [7]. Finally, the most popular data dictated approach is Clipped [24].

In this paper a novel framework, named Signal2vec, is introduced. A similar approach has been proposed by Nalmpantis et al. [20], where a model called Energy2vec is used to create a hyperspace of energy embeddings. Energy2vec is binded to the energy domain, is supervised and its applicability is limited. On the other hand, Signal2vec is a general, unsupervised model and is applicable in any time series. It is inspired by Word2vec [17] which builds a vector space, maintaining semantic and syntactic relations of the original words. Word2vec has been applied on numerous textual or discrete sequences such as recommendation systems [2, 22], ranking of sets of entities [4, 10], biology [1] and others [26]. Signal2vec is the first attempt, that extends Word2vec applicability on any sequential data in continuous space.

The benefits and the drawbacks of the framework are discussed extensively through a theoretical analysis. The framework is validated with experiments in two different tasks: classification and single source separation. Both, the analysis and the experiments are based on energy data.

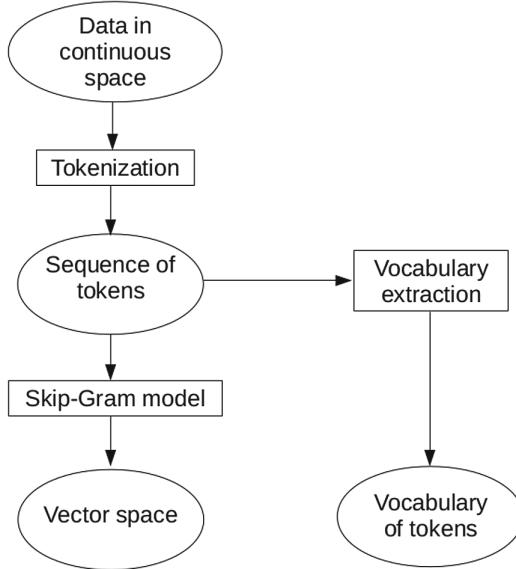
## 2 Signal2Vec

Signal2vec consists of two main steps: tokenization and skip-gram model. The former one is a discretization process, transforming a continuous time series into tokens. The latter one transforms the sequence of tokens into embeddings. Figure 1 illustrates the steps of the framework.

### 2.1 Tokenization

Unsupervised tokenization is an abstract and scalable approach in order to discretize a time series. It can be applied on different domains and it can fit different variations of data in the same domain. It is completed in two main steps: token extraction and token assignment. The first step can be achieved by a clustering algorithm. The second one uses classification, in order to transform a continuous time series to a sequence of tokens.

At the current implementation, k-means is used to define the tokens, the number of which is not known upfront. The best number of tokens, which is also the number of clusters, is found by using silhouette score, within a desirable range of values. Silhouette score shows which objects lie well within their cluster [25] and the desirable range of values can be defined empirically. In energy domain this range can be estimated by calculating the number of possible energy states



**Fig. 1.** Signal2Vec framework.

using complexity of power draws [8]. Next, the classifier k-nearest neighbors is trained to map values of the signal to tokens. The classifier can be used to tokenize time series from the same domain.

The algorithm is evaluated using household energy data. Tokens represent the energy states of each appliance, because token extraction is applied on the submetered data. Then, sequences of tokens are created for each appliance. The final sequence is a concatenation of the appliance specific sequences and corresponds to the aggregated signal. In problems like power disaggregation, both token extraction and token assignment would be applied directly on the aggregated signal, because the submetered data are supposed to be unknown. The analysis that follows is based on submetered data, in order to present meaningful tokens, that correspond to appliances states.

## 2.2 Skip-Gram

Signal2vec is based on word2vec, which uses either skip-gram model or continuous bag-of-words (CBOW). Skip-gram predicts the words around the target word and CBOW predicts the target word given its neighbours. Both methods can be applied having minor differences on the results. For consistency, skip-gram is selected for all the experiments.

Following tokenization, a time series is mapped to a sequence of tokens. This sequence is now called a corpus. If the tokens are not abstract states and reflect real-world conditions of the time series, then the corpus is a human description of the total signal. The collection of the tokens consists a vocabulary. In order

to apply the skip-gram model, a context is also defined as the window to the left and to the right of the target token. The objective of the algorithm is to predict the context given a specific token. The architecture is a shallow neural network with one hidden layer and is trained with pairs of tokens. One token is the target and the other one belongs to the context. The network trains the weights of the hidden layer from the frequencies each pairing shows up. A more formal definition of the objective function is defined next.

Let  $t_{kn_1}, t_{kn_2}, \dots, t_{kn_T}$  be a sequence of  $T$  training tokens. Then the objective function tries to maximize the average log probability according to the formula:

$$\frac{1}{T} \sum_{t=1}^T \sum_{-c \leq i \leq c, i \neq 0} \log p(t_{kn_{t+i}} | t_{kn_t}), \quad (1)$$

where  $c$  is the training context.

The order of the tokens in a context doesn't affect the result of the algorithm, which depends mainly on the frequency of the tokens. In word2vec model this is mentioned by Mikolov et al. [18] as a limitation of the model, because it cannot capture rules that dictate the order of words in a sentence. In time series usually there aren't any syntactic rules and there is no difference if a token is before or after its neighbors.

The network is trained computing Noise Contrastive Estimation (NCE) [11] loss function. The optimizer is the Adagrad with learning rate 0.001. The size of each embedding is 300 and the window is 6 tokens. The data come from House 1 of UK-DALE dataset [12] during the year 2014.

### 3 Evaluation

#### 3.1 Data and Tools

The source of the data is the UK-DALE dataset, which includes both the aggregated and the individual power consumption of the appliances in a house. House 1 is selected, because it has the most devices of all the houses. The preferred programming language is Python. The tool named NILMTK [3] is used for accessing the database and preprocessing the energy data. In order to distribute computation to many CPU cores and a GPU, the skip-gram model is developed in Tensorflow. Tensorflow comes along with a suite of visualization tools, called Tensorboard. It is used to plot diagrams of the model, visualize the embeddings and evaluate the model. Tensorboard's visualization tool uses PCA and TSNE in order to plot the embedding space in three or two dimensions. The evaluation of the embedding space is mainly done with tensorboard's similarity tool, which supports both cosine and euclidean distance.

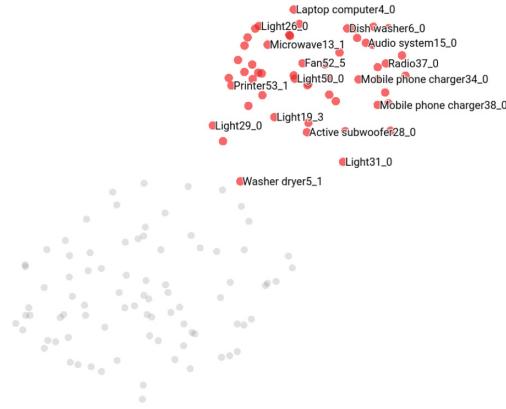
#### 3.2 Analysis of the Learned Representations

Signal2vec is a framework which transforms a time series to a continuous vector space. In order to understand the intuition of a signal's geometrical representation, an evaluation process is presented, focusing on household energy data.

In unsupervised tokenization the tokens are defined in an abstract mathematical way. Tokenizing the aggregated energy signal is helpful solving real world problems, without any insight about the nature of the tokens. In this evaluation the method is applied on the individual signals, extracting multiple states per appliance. No window is used and tokens are mapped in high resolution, close to the sampling rate. In order to get a physical image of the energy behavior, diagrams depicting the frequency of each energy state are generated.

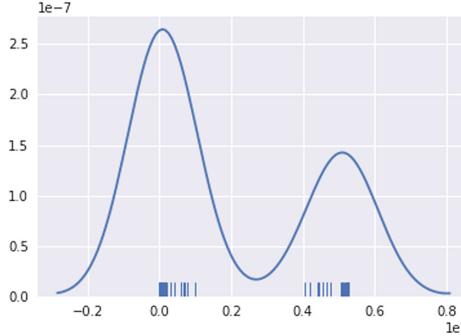
The name of the states is labeled by the name of each appliance, followed by a number. Thus, the name of an energy state doesn't directly reveal the real world functionality, but only which appliance it belongs to. Also, regarding the zero state of an appliance, it is no more distinguishable from the other states, as it is just another extra label with arbitrary number. A diagram of an appliance which is ON continuously would be the same with a diagram of an appliance being continuously OFF. To avoid any ambiguity, energy plots from raw data are used, as well.

Tensorboard is used to visualize and explore the embedding space. Dimensionality reduction is achieved by means of PCA. The geometry of the space has two distinct groups of points. One group represents high frequency states and the other one low frequency states. The same separation based on frequency is seen in word embeddings, where words follow Zipf's law.



**Fig. 2.** TSNE: The embeddings are grouped in two clusters.

Another useful tool, which comes with Tensorboard, is a clustering analysis using TSNE. The majority of the combinations of the values of perplexity and learning rate give clear separation of the two groups that have been identified with PCA. Figure 2 shows an example of the TSNE algorithm with perplexity 5 and learning rate 10. The two clusters are consistent even when trying different settings of the algorithm. Only the shape is changing when the perplexity number is much bigger. For example, perplexity 40 gives a circle, when projecting the



**Fig. 3.** Frequency distribution of tokens.

embeddings in 2D and the two groups are gathered in two different semicircles. In 3D the clusters are more clear because the two semicircles are separated as two distinct shapes.

The similarity tool is used to find vectors that are close to each other. Both Euclidean and cosine distance metrics are used, although there isn't any significant difference in the results. For each similarity search, the respective plots of the input vector and its five closest ones are compared. The results show that embeddings with small distance in the geometrical space, have similar frequency diagrams. The results are very robust in terms of distinguishing high and low frequency energy states. Almost all the cases of similarity searches give neighbor vectors, which correspond to the same category of frequency. On the other hand, vectors belonging to the same cluster cannot be distinguished and no characteristics are found to justify the results of a similarity search.

Figure 3 depicts the frequency distribution of tokens, derived from unsupervised tokenization. There are two central points, forming two normal distributions. Comparing the frequency distributions and the geometric properties of the embedding spaces, there is a connection between the tokens and the embeddings. Skip-gram, transferred the properties of the sequence of tokens to a multidimensional space. Assuming that sequences of tokens can be translated to frequency of appliance usage, which in turn implies human behaviour and habits, it can be concluded that the constructed vector space encapsulates the energy profile of the house.

## 4 Real World Applications

In this proposal two real world experiments are presented, classification and energy disaggregation. The first one examines the capabilities of the proposed framework to classify signals from different appliances efficiently. The second one is a simple approach on how a single source separation problem, such as energy disaggregation, can be solved using Signal2vec. The data are from UK-DALE house 1 during the year 2014. The difference is that now the vectors are

produced from the aggregated energy signal and they are used to transform any other energy time series.

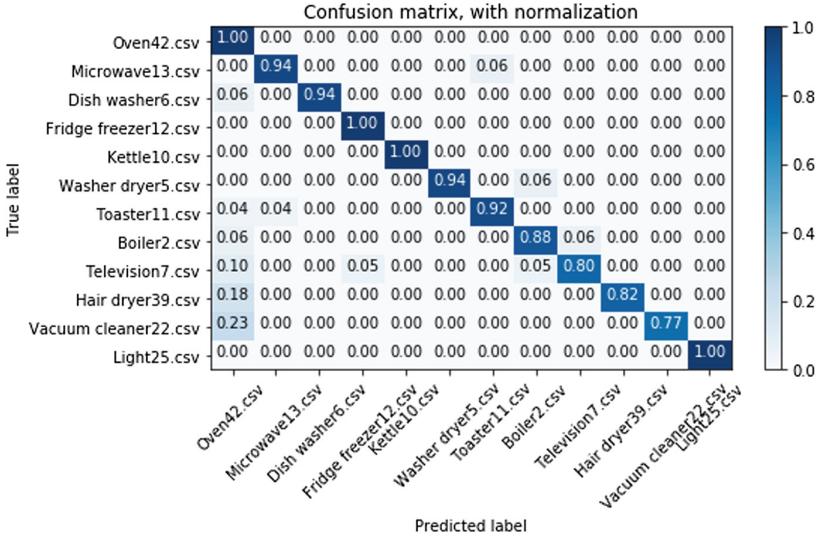
#### 4.1 Multiclass Classification of Appliances' Energy Consumption

The first experiment is a multiclass classification problem, identifying 12 different appliances: oven, microwave, dish washer, fridge freezer, kettle, washer dryer, toaster, boiler, television, hair dryer, vacuum cleaner and light. The classifier is a random forest with 200 estimators. The dataset of different labeled signals is created as follows. Firstly, the submetered data of 12 appliances are converted to sequences of 300 dimensional vectors, using Signal2vec. Next, the data are broken into smaller pieces with fixed length, corresponding to a specific time period. For example during 9 months with data sampled every 6s, a time period of 1 day and 12 appliances would give approximately 4188 labeled sequences of vectors. Then the average vector of each 1 day length time series is calculated. The average vector is the input to the classifier and the label is one of the 12 appliances. The metric that is used is mainly macro f1-score. The robustness of the results was validated using a k-fold cross-validation, after the initial data had been randomly shuffled. The parameter k is chosen as  $k = 3$ , because for larger values the test data sample was not statistically representative of the broader dataset.

The results vary depending on the size of each time series. The smaller the time period is, the worse the results are. This can be explained because in periods smaller than a day some devices are not used at all. Indeed, for devices that are used daily, such as fridge freezer, the classifier could recognize the appliance successfully even with a time period of 4 h with individual f1-score 0.95. The respective macro f1-score for the 12 appliances was 0.39. Consequently, the experiments for 12 appliances are meaningful for time period greater than a day. Another approach would be to have an extra label for time periods during which no device is on, known as zero state, but this is left for future work.

The framework is robust when the appliances are increased. For example using 7 day length time series for the cases of 6, 12 and 17 appliances the macro f1-score is 0.97 ( $+/- 0.019$ ), 0.94 ( $+/- 0.015$ ) and 0.79 ( $+/- 0.015$ ) respectively. Figure 4 shows a confusion matrix summarizing the classification results of 12 appliances. It is worth mentioning that most of the misclassifications concern the oven. This is not a surprise because an oven's energy consumption heavily depends on the way it is used.

Table 1 compares Signal2vec against raw data and two other representations, SAX [16] and PAA [14]. The results involve time series with sizes 1, 5 and 7 days, using 3 cross validation. Other classifiers have also been tested. Regardless the representation, random forest showed the best results. SAX and PAA have been tuned to get the best possible f scores. Overall Signal2vec is superior, not only giving the best results, but also because the dimension of the final representative vector is independent of the length of the time series. Signal2vec is surpassed by PAA, only for short time series such as 1 day size. A possible explanation is that temporal patterns are more important during short periods, because a device is



**Fig. 4.** Confusion matrix of 12 classes using Signal2vec.

used rarely. When calculating the average vector any temporal patterns are lost, whereas in PAA they are maintained. For future experiments more sophisticated methods can be evaluated e.g. weighted average vector. Finally, it is notable that for larger time series the results for raw data are getting worse. This is explained by the dimensionality explosion, which signal2vec faces by compressing all the information into a single vector of constant dimensions.

**Table 1.** Macro f1 score for 12 appliances.

TS size	Signal2vec	Raw	SAX	PAA
1 day	0.730	0.730	0.682	<b>0.814</b>
3 days	<b>0.878</b>	0.732	0.748	0.869
5 days	<b>0.930</b>	0.681	0.767	0.897
7 days	<b>0.942</b>	0.693	0.766	0.920

## 4.2 Energy Disaggregation

Many machine learning approaches, including deep learning, have been proposed for the problem of energy disaggregation [21]. The majority of them use one model for each appliance, because they underperform when trying to disaggregate many devices. The current experiment tackles energy disaggregation as a multilabel classification problem. The input data come from the main aggregated energy signal, following the same procedure as in the first experiment. The whole

time series during 2014 is segmented into fixed windows. Signal2vec is applied to each window and finally the average vectors are calculated. Each representative vector is the input to a multilayer perceptron classifier with one hidden layer and 100 neurons. The fixed windows that have been tested correspond to 4, 8, 12 and 24 h. The labels are the appliances that were ON at least one time during the fixed time period. The results are very encouraging, especially when comparing the performance of the same model identifying 6 and 12 appliances. Table 2 presents the results in details. Additional experiments need to be implemented for comparison with other models.

**Table 2.** Macro f1 score of energy disaggregation.

TS size	6 Appliances	12 Appliances
4 h	0.534 ( $\pm 0.029$ )	0.512 ( $\pm 0.008$ )
8 h	0.654 ( $\pm 0.005$ )	0.599 ( $\pm 0.037$ )
12 h	0.722 ( $\pm 0.015$ )	0.678 ( $\pm 0.043$ )
24 h	0.868 ( $\pm 0.032$ )	0.757 ( $\pm 0.045$ )

## 5 Conclusion

Signal2vec is a computationally efficient model, which transforms the data of a time series into a vector space. The trained embeddings are easily reusable and maintain some of the properties of the original data. It is unsupervised, requires no domain knowledge, is scalable and applicable in different areas e.g. speech recognition, finance, health, IoT.

Specifically, in energy domain, it is the first time the energy profile of a building is mapped to a multidimensional space. Once the embeddings are trained, they can be reused either by incorporating them in a neural network or by other models as features. Two different classification problems have been showcased, with application in real world problems. The first one classifies different categories of signals, coming from different sources. The second one, energy disaggregation, is a single source separation problem. Both experiments showed very promising results, reducing a time series of thousands values to a 300 dimensional vector.

Further research is suggested to be conducted to improve Signal2vec, show experimental results in other real world problems and compare against state-of-the-art methods.

## References

1. Asgari, E., Mofrad, M.R.: Continuous distributed representation of biological sequences for deep proteomics and genomics. *PloS one* **10**(11), e0141287 (2015)
2. Barkan, O., Koenigstein, N.: Item2Vec: neural item embedding for collaborative filtering. In: 2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP), pp. 1–6. IEEE (2016)
3. Batra, N., et al.: NILMTK: an open source toolkit for non-intrusive load monitoring. In: Proceedings of the 5th International Conference on Future Energy Systems, pp. 265–276. ACM (2014)
4. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in Neural Information Processing Systems, pp. 2787–2795 (2013)
5. Chan, K.P., Fu, A.W.C.: Efficient time series matching by wavelets. In: Proceedings of the 15th International Conference on Data Engineering, 1999, pp. 126–133. IEEE (1999)
6. Chen, Q., Chen, L., Lian, X., Liu, Y., Yu, J.X.: Indexable PLA for efficient similarity search. In: Proceedings of the 33rd International Conference on Very Large Data Bases, pp. 435–446. VLDB Endowment (2007)
7. Corduas, M., Piccolo, D.: Time series clustering and classification by the autoregressive metric. *Comput. Stat. Data Anal.* **52**(4), 1860–1872 (2008)
8. Egarter, D., Pöchacker, M., Elmenreich, W.: Complexity of power draws for load disaggregation (2015). arXiv preprint [arXiv:1501.02954](https://arxiv.org/abs/1501.02954)
9. Faloutsos, C., Ranganathan, M., Manolopoulos, Y.: Fast subsequence matching in time-series databases, vol. 23. ACM (1994)
10. Garcia-Duran, A., Bordes, A., Usunier, N.: Composing relationships with translations. Ph.D. thesis, CNRS, Heudiasyc (2015)
11. Gutmann, M.U., Hyvärinen, A.: Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *J. Mach. Learn. Res.* **13**(Feb), 307–361 (2012)
12. Kelly, J., Knottenbelt, W.: The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes. *Sci. Data* **2**, 150007 (2015)
13. Keogh, E., Chakrabarti, K., Pazzani, M., Mehrotra, S.: Locally adaptive dimensionality reduction for indexing large time series databases. *ACM Sigmod Rec.* **30**(2), 151–162 (2001)
14. Keogh, E.J., Pazzani, M.J.: A simple dimensionality reduction technique for fast similarity search in large time series databases. In: Terano, T., Liu, H., Chen, A.L.P. (eds.) PAKDD 2000. LNCS (LNAI), vol. 1805, pp. 122–133. Springer, Heidelberg (2000). [https://doi.org/10.1007/3-540-45571-X\\_14](https://doi.org/10.1007/3-540-45571-X_14)
15. Korn, F., Jagadish, H.V., Faloutsos, C.: Efficiently supporting ad hoc queries in large datasets of time sequences. In: ACM Sigmod Record, vol. 26, pp. 289–300. ACM (1997)
16. Lin, J., Keogh, E., Wei, L., Lonardi, S.: Experiencing SAX: a novel symbolic representation of time series. *Data Min. Knowl. Disc.* **15**(2), 107–144 (2007)
17. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space (2013). CoRR abs/1301.3781. <http://arxiv.org/abs/1301.3781>
18. 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)

19. Minnen, D., Isbell, C.L., Essa, I., Starner, T.: Discovering multivariate motifs using subsequence density estimation and greedy mixture learning. In: Proceedings of the National Conference on Artificial Intelligence, vol. 22, p. 615. AAAI Press; MIT Press, Menlo Park, Cambridge, London (1999, 2007)
20. Nalmpantis, C., Krystalakos, O., Vrakas, D.: Energy profile representation in vector space. In: 10th Hellenic Conference on Artificial Intelligence SETN 2018. ACM (2018)
21. Nalmpantis, C., Vrakas, D.: Machine learning approaches for non-intrusive load monitoring: from qualitative to quantitative comparation. Artif. Intell. Rev. 1–27 (2018)
22. Ozsoy, M.G.: From word embeddings to item recommendation (2016). arXiv preprint [arXiv:1601.01356](https://arxiv.org/abs/1601.01356)
23. Portet, F., et al.: Automatic generation of textual summaries from neonatal intensive care data. Artif. Intell. **173**(7–8), 789–816 (2009)
24. Ratanamahatana, C., Keogh, E., Bagnall, A.J., Lonardi, S.: A novel bit level time series representation with implication of similarity search and clustering. In: Ho, T.B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS (LNAI), vol. 3518, pp. 771–777. Springer, Heidelberg (2005). [https://doi.org/10.1007/11430919\\_90](https://doi.org/10.1007/11430919_90)
25. Rousseeuw, P.J.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math. **20**, 53–65 (1987)
26. Wu, L., Fisch, A., Chopra, S., Adams, K., Bordes, A., Weston, J.: Starspace: Embed all the things (2017)! arXiv preprint [arXiv:1709.03856](https://arxiv.org/abs/1709.03856)

# **Biomedical - Bioinformatics Modeling**



# Classification of Sounds Indicative of Respiratory Diseases

Stavros Ntalampiras<sup>1</sup> and Ilyas Potamitis<sup>2</sup>(✉)

<sup>1</sup> University of Milan, Milan, Italy  
stavros.ntalampiras@unimi.it

<sup>2</sup> Technological Educational Institute of Crete, Rethymno, Greece  
potamitis@staff.teicrete.gr

**Abstract.** This work presents a system achieving classification of respiratory sounds directly related to various diseases of the human respiratory system, such as asthma, COPD, and pneumonia. We designed a feature set based on wavelet packet analysis characterizing data coming from four sound classes, i.e. *crack*, *wheeze*, *normal*, *crack+wheeze*. Subsequently, the captured temporal patterns are learned by hidden Markov models (HMMs). Finally, classification is achieved via a directed acyclic graph scheme limiting the problem space while based on decisions made by the available HMMs. Thorough experiments following a well-established protocol demonstrate the efficacy of the proposed solution.

**Keywords:** Respiratory sound classification · Acoustic signal processing · Respiratory diseases

## 1 Introduction

It is widely accepted that diseases of the human respiratory system, such as asthma, COPD, and pneumonia are associated with distinctive acoustic patterns [13]. This is due to the abnormalities they cause in the airway path. Typically, a medical expert is able to correctly identify such patterns (e.g. by means of a stethoscope) and subsequently propose the corresponding treatment. However, this process relies both on the availability of an expert as well as their degree of expertise. Thus, the need to automatize the diagnosis process has arose in the last years igniting the development of such algorithms.

Even though during the last decade there has been a significant amount of research in this direction, a standardized way to compare the existing solutions is yet to appear. Systematic reviews of the state of the art are available in [13, 17]. A great variety of temporal, spectral and wavelet features along with generative and non-generative classifiers have been employed in the literature. Such a review is beyond the scope of the present article; we rather focus on a recent standardized attempt approaching this problem. More in detail, the challenge organized within the International Conference on Biomedical Health Informatics in 2017 provides

a dataset characterizing the properties of the classes of interest as well as an experimental protocol allowing the extraction of comparable results.

So far, two solutions employing the challenge’s experimental protocol stand out. The first one [4] uses hidden Markov models fed with mel-frequency cepstral coefficients. The second [19] employs non-linear spectral features along with a support vector machine with a radial basis function kernel.

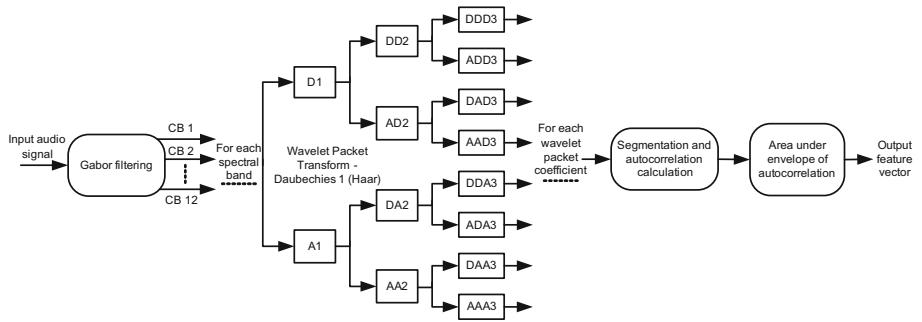
This work builds on the existing findings and proposes the usage of features derived from the wavelet domain, the distribution of which is learned by a directed acyclic graph scheme composed of hidden Markov models. More precisely, we designed a three-level wavelet packet band-based analysis component able to capture the behavior of the involved sound events within various spectral bands. Subsequently, we construct a directed acyclic graph limiting the problem space into a series of binary classifiers, each one relying on a pair of hidden Markov models. On top of that, we provide a solution to the topological ordering of such a graph. Experimental results using the protocol of the ICBHI 2017 challenge demonstrate the efficacy of the proposed approach.

The rest of this work is organized as follows: Sect. 2 describes the wavelet packet feature extraction module, while Sect. 3 explains the graph-based classification scheme. Section 4 details the employed dataset, the parameterization of the proposed approach as well as the achieved results and how these compare to existing solutions. Finally, in Sect. 5 we draw our conclusions.

## 2 The Feature Set

This section introduces the usage of band-based multiresolution analysis for automated respiratory sound classification. Lately, digital signal processing using wavelets has become a common tool in various research areas with heterogeneous needs. Such cases refer to enhancement of biological signals [11, 15], geophysics applications like tropical convection, dispersion of ocean waves, etc. [21], speech/music discrimination [12], emotion prediction [10], farm monitoring [7], voice activity detection [2], moving vehicle classification [8], audio fingerprinting [1], generalized sound recognition [6], to name but a few.

The fundamental property of the Fourier transform is the usage of sinusoids with infinite duration. Whereas sinusoids functions are smooth and predictable, wavelets tend to be irregular and asymmetric. The main advantage of the wavelet transform is that it can analyze at many different frequencies time series characterized by non-stationary power. They comprise a dynamic windowing technique which can treat with different precision low and high frequency information. The first step of the wavelet packet analysis is the choice of the original (or mother) wavelet and by utilizing this function, the transformation breaks up the signal into shifted and scaled versions of it. In this paper we utilized Daubechies 1 (or Haar) function as the original wavelet while its optimal choice will be a subject of our future work. Unlike discrete wavelet transform (DWT), when wavelet packets (WP) are employed both low and high frequencies coefficients are kept. In our case the DWT is applied three subsequent times and consists of three-stage filtering of the audio signals as we can see in Fig. 1.



**Fig. 1.** The block diagram of proposed feature extraction module. Audio signals are filtered and each spectral band is analyzed by a 3-level wavelet packet transform. After segmenting and computing the area under the autocorrelation envelope, we obtain the feature vector.

**Table 1.** The frequency limits used for perceptual wavelet packet integration analysis.

Band ID	Lower (Hz)	Center (Hz)	Upper (Hz)
1	0	125	250
2	250	375	500
3	500	625	750
4	750	875	1000
5	1000	1125	1250
6	1250	1375	1500
7	1500	1625	1750
8	1750	1875	2000
9	2000	2250	2500
10	2500	2750	3000
11	3000	3250	3500
12	3500	3750	4000

The idea behind the specific set is the production of a vector that provides a complete analysis of the audio signal across different spectral areas while they are approximated by WP. We should also take into account that respiratory signals do not distribute their energy across the spectrum in a homogeneous way. Thus, a fine partitioning of the spectrum could offer relevant distinctive information. Based on this observation, we designed a band-based signal analysis with the frequency ranges denoted in Table 1. Such a division is achieved by Gabor bandpass filters. Subsequently, three-level wavelet packets are extracted out of each spectral band. The specific level is able to provide detailed information regarding the signal characteristics at a specific band. Downsampling is applied on each coefficient at each stage in order not to end up having the double

amount of data, as Nyquist theorem requests. The wavelet coefficients are then segmented and the autocorrelation envelope area is computed and normalized by half the segment size.  $M$  normalized integration parameters are calculated for each frame, where  $M$  is the total number of the frequency bands multiplied by the number of the wavelet coefficients. This series of parameters comprises the WP-Integration feature vector and the entire calculation process is depicted in Fig. 1.

WP-Integration parameters reflect upon the degree of variability of a specific wavelet coefficient within a frequency band. Since the audio signals we try to classify exhibit great differences among these bands, we decided to utilize the normalized autocorrelation envelope area.

### 3 The Classification Scheme

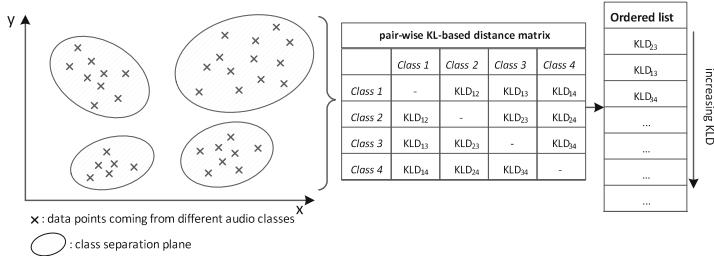
The proposed framework relies on the Directed Acyclic Graph (DAG) logic [9], i.e. the classification scheme is a graph denoted as  $\mathcal{G} = \{N, L\}$ , where  $N = \{n_1, \dots, n_m\}$  represents the nodes and  $L = \{l_1, \dots, l_p\}$  the links associating the nodes. Each node in  $N$  is responsible for a binary classification task conducted via a set of HMMs which fit well the specifications of audio pattern recognition tasks, thus the DAG-HMM notation.

The motivation behind creating such a graph-based classification system is that in this way, one is able to limit the problem space and design classification algorithms for two mutually-exclusive classes than having to deal with the entirety of the different classes at the same time. Essentially, the proposed methodology breaks down any  $m$ -class classification problem into a series of 2-class classification problems.

DAGs can be seen as a generalization of the class of Decision Trees, while the redundancies and repetitions that may occur in different branches of the tree can be observed more efficiently since different decision paths might be merged. In addition, DAGs are able to collect and conduct a series of tasks in an ordered manner, subject to constraints that certain tasks must be performed earlier than others. The sequential execution of tasks is particularly important and directly related to the efficacy with which the overall task is addressed [22].

The DAG-HMM architecture used in this paper includes  $m(m - 1)/2$  nodes ( $m$  being the total number of classes) each one associated with a two-class classification problem. The connections between the different nodes in  $\mathcal{G}$  have only one orientation without any kind of loop(s). As a result, each node of a such a so-called *rooted* DAG has either 0 or 2 leaving arcs.

The principal issue associated with the design of every DAG is the *topological ordering*, i.e. ordering the nodes in a way that the starting endpoints of every edge occur earlier than the corresponding ending endpoints. In the following, we describe how such a topological ordering is discovered based on the Kullback-Leibler divergence.



**Fig. 2.** Determination of the topological ordering.

### 3.1 Determining the Topological Ordering of the DAG-HMM

Naturally, one would expect that the performance of the DAG-HMM depends on the order in which the different classification tasks are conducted. This was also evident from early experiments. This observation motivated the construction of the DAG-HMM so that “simple” tasks are executed earlier in the graph. In other words, these are placed in the top nodes of the DAG-HMM, in a way that classes responsible for a high amount of misclassifications are discarded early in the graph operation. In order to get an early indication of the degree of difficulty of a classification task, we employed the metric representing the distance of the involved classes in the probabilistic space, i.e. the Kullback-Leibler Divergence (KLD) between per-class GMMs in the feature space. The basic motivation is to place early in the DAG-HMM tasks concerning the classification of classes with large KLD, as they could be completed with high accuracy. The scheme determining the topological ordering is illustrated in Fig. 2.

The KLD between two  $J$ -dimensional probability distributions  $A$  and  $B$  is defined as [20]:

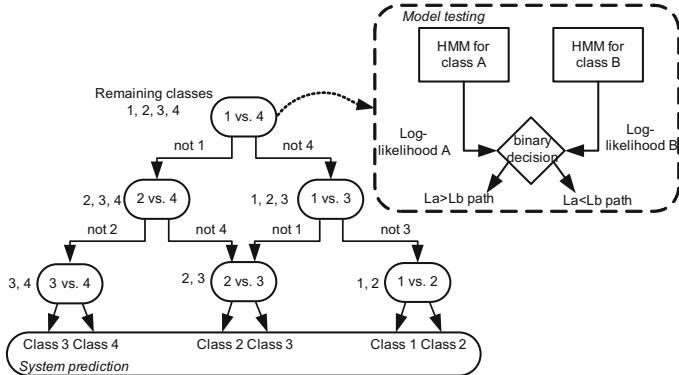
$$D(A||B) = \int_{R^J} p(X|A) \log \frac{p(X|A)}{p(X|B)} dx \quad (1)$$

KLD provides an indication of how distant two models are in the probabilistic space. It is important to note that KLD as given in Eq. 1 comprises an asymmetric quantity. The symmetrical form can be inferred by simply adding the integrals in both directions, i.e.

$$D_s(A||B) = D(A||B) + D(B||A) \quad (2)$$

In the special case where both  $A$  and  $B$  are Gaussian mixture models KLD can be defined as follows:

$$KLD(A||B) = \int A(x) \log \frac{B(x)}{A(x)} dx \quad (3)$$



**Fig. 3.** An example of a DAG-HMM addressing a problem with four classes.

Unfortunately, there is not a closed-form solution for Eq. 3, thus we employed the empirical mean as follows

$$KLD(A||B) \approx \frac{1}{n} \sum_{i=1}^n \log \frac{B(x_i)}{A(x_i)} \quad (4)$$

given that the number of Monte Carlo draws is sufficiently large. During our experiments we set  $n = 2000$ .

It should be noted the KLD between HMMs was not used since computing distances between HMMs of unequal lengths, which might be common in this work as HMMs representing different classes might have different number of states, can be significantly more computationally demanding without a corresponding gain in modeling accuracy [5, 23].

After computing the KLD for the different pairs of classes, i.e. reach the second stage depicted in Fig. 2, the KLD distances are sorted in a decreasing manner. This way the topological ordering of the DAG-HMM is revealed, placing the classification tasks of low difficulty on its top. Each node removes a class from the candidate list until there is only one class left, which comprises the DAG-HMM prediction. The distance matrix elements could be seen as early performance indicators of the task carried out by the corresponding node. The proposed topological ordering places tasks likely to produce misclassifications at the bottom of the graph. This process outputs a *unique* solution for the topological sorting problem, as it is usually met in the graph theory literature [3].

### 3.2 The DAG-HMM Operation

The operation of the proposed DAG-HMM scheme is the following: after extracting the features of the unknown audio signal, the first/root node is activated. More precisely, the feature sequence is fed to the HMMs, which produce two log-likelihoods showing the degree of resemblance between the training data of

each HMM and the unknown one. These are compared and the graph flow continues on the larger log-likelihood path. It should be stressed out that the HMMs are optimized (in terms of number of states and Gaussian components) so that they address the task of each node optimally. That said, it is possible that a specific class is represented by HMMs with different parameters when it comes to different nodes of the DAG-HMM.

An example of a DAG-HMM addressing a problem with four classes is illustrated in Fig. 3. The remaining classes for testing are mentioned beside each node. Digging inside each node, Fig. 3 also shows the HMM-based sound classifier responsible for activating the path of the maximum log-likelihood.

The operation of the DAG-HMM may be parallelized with that of investigating a list of classes, where each level eliminates one class from the list. More in detail, in the beginning the list includes all the potential audio classes. At each node the feature sequence is matched against the respective HMMs and the model with the lowest log-likelihood is erased from the list, while the DAG-HMM proceeds to the part of the topology without the discarded class. This process terminates when only one class remains in the list, which comprises the system's prediction. Hence, in case the problem deals with  $m$  different classes, the DAG's decision will be made after the evaluation of  $m - 1$  nodes.

## 4 Experiments

This section explains (*a*) the dataset, (*b*) the parameterization of the proposed solution for classification of respiratory sounds, and (*c*) finally presents and analyses the achieved results.

### 4.1 Dataset

The respiratory sound database comes from the challenge organized within the International Conference on Biomedical Health Informatics in 2017 and it is publicly available<sup>1</sup>. The recordings span over several years. The database has a total duration of 5.5 h and contains 6898 respiratory cycles, of which 1864 contain crackles, 886 contain wheezes, and 506 contain both crackles and wheezes, in 920 annotated audio samples coming from 126 subjects.

The cycles were annotated by respiratory experts as including *crackles*, *wheezes*, a *combination* of them, or *no adventitious* respiratory sounds. The recordings were collected using heterogeneous equipment and their duration ranges from 10 s to 90 s. In addition, noise levels in some respiration cycles is high, representing very well, real life conditions. Finally, training and testing data are already defined by the challenge organization committee. More information regarding the dataset is available in [18].

---

<sup>1</sup> <https://bhichallenge.med.auth.gr/>.

**Table 2.** The recognition rates for the proposed and contrasted methods.

Approach	Recognition rate (%)
HMMs+MFCCs [4]	39.5
Non-linear spectral features+SVM [19]	49.8
DAG-HMM + WP-Integration	<b>50.1</b>

## 4.2 System Parameterization

The low-level feature extraction window is 30 ms with 10 ms overlap between subsequent windows, while the Daubechies mother wavelet was selected. The HMMs of each node are optimized in terms of number of states and nodes following the Expectation-Maximization and Baum Welch algorithms [14]. As the considered sound events are characterized by a distinct time evolution, we employed HMMs with left-right topology, i.e. only left to right states transitions are permitted. Moreover, the distribution of each state is approximated by a Gaussian mixture model of diagonal covariance, which may be equally effective to a full one at a much lower computational cost [16].

The maximum number of  $k$ -means iterations for cluster initialization was set to 50 while the Baum-Welch algorithm used to estimate the transition matrix was bounded to 25 iterations with a threshold of 0.001 between subsequent iterations. The number of explored states ranges from 3 to 7 while the number of Gaussian components used to build the GMM belongs to the {2, 4, 8, 16, 32, 64, 128, 256, and 512} set. The final parameters were selected based on the maximum recognition rate criterion. The machine learning package Torch<sup>2</sup> was used to construct and evaluate GMMs and HMMs.

## 4.3 Results

Table 2 depicts the rates achieved by two contrasted approaches as well as the proposed one. We observe that the solution based on DAG-HMM fed with PWP-Integration feature set achieved the highest recognition rate which is equal to 50.1%. Interestingly the inferred topological order suggested the execution of classification tasks with the following order: (a) crack+wheeze vs. normal, (b) normal vs. wheeze, (c) crack vs. crack+wheeze, (d) crack vs. wheeze, (e) crack+wheeze vs. wheeze, and (f) crack vs. normal.

Towards a more detailed picture of its classification capabilities, Table 3 tabulates the confusion matrix. As we can see, the class identified with the highest accuracy is the *wheeze* one with 64.5% and second is the *normal* one with 63%. On the contrary, *crack* sound events were the most misclassified ones with the respective rate being 36.7%.

Even though the achieved rate is the highest one reported in the literature, it is still far from satisfactory. Interestingly, when samples from *crack*,

---

<sup>2</sup> Freely available at <http://torch.ch/>.

**Table 3.** The confusion matrix (in %) achieved by the proposed approach. The average recognition rate is 50.1%.

Presented	Responded			
	crack	crack+wheeze	normal	wheeze
crack	36.7	3.1	58.5	1.6
crack+wheeze	3.1	38.4	57.9	0.6
normal	32.4	3.3	63	1.3
wheeze	0.5	3.2	31.8	64.5

*crack+wheeze*, and *wheeze* are misclassified, they are identified as *normal* at most cases. This indicates that the patterns exhibited by the data coming from the *normal* class are similar to all other classes. A potential solution to this problem is collecting more data from the *non-normal* classes, such that the differences are highlighted.

## 5 Conclusions

This work explained a graph-based classification scheme encompassing wavelet analysis and temporal modeling. Such an approach was able to surpass solutions existing in the literature. Nonetheless, the achieved classification rates highlight the fact that automated respiratory sound analysis systems are not yet ready to assist medical experts. Towards improving the current performance, in the future we intent to pursue the following directions: (a) augment the *non-normal* part of the dataset, (b) employ a combination of spectral and wavelet features, and (c) include a discriminative classifier, possibly forming a synergistic framework.

**Acknowledgment.** This research was funded by the ELKE TEI Crete funds related to the domestic project: Bioacoustic applications, number 80680.

## References

1. Baluja, S., Covell, M.: Waveprint: efficient wavelet-based audio fingerprinting. Pattern Recogn. **41**(11), 3467–3480 (2008). <https://doi.org/10.1016/j.patcog.2008.05.006>. <http://www.sciencedirect.com/science/article/pii/S0031320308001702>
2. Chen, S.H., Wu, H.T., Chen, C.H., Ruan, J.C., Truong, T.: Robust voice activity detection algorithm based on the perceptual wavelet packet transform. In: 2005 International Symposium on Intelligent Signal Processing and Communication Systems. IEEE (2005). <https://doi.org/10.1109/ispacs.2005.1595342>
3. Cook, S.A.: A taxonomy of problems with fast parallel algorithms. Inf. Control **64**(1), 2–22 (1985). [https://doi.org/10.1016/S0019-9958\(85\)80041-3](https://doi.org/10.1016/S0019-9958(85)80041-3). <http://www.sciencedirect.com/science/article/pii/S0019995885800413>. International Conference on Foundations of Computation Theory

4. Jakovljević, N., Lončar-Turukalo, T.: Hidden Markov model based respiratory sound classification. In: Maglavera, N., Chouvarda, I., de Carvalho, P. (eds.) Precision Medicine Powered by pHealth and Connected Health. IP, vol. 66, pp. 39–43. Springer, Singapore (2018). [https://doi.org/10.1007/978-981-10-7419-6\\_7](https://doi.org/10.1007/978-981-10-7419-6_7)
5. Liu, P., Soong, F.K., Zhou, J.L.: Divergence-based similarity measure for spoken document retrieval. In: 2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP 2007, vol. 4, pp. IV-89–IV-92, April 2007. <https://doi.org/10.1109/ICASSP.2007.367170>
6. Ntalampiras, S.: A novel holistic modeling approach for generalized sound recognition. *IEEE Sig. Process. Lett.* **20**(2), 185–188 (2013). <https://doi.org/10.1109/LSP.2013.2237902>
7. Ntalampiras, S.: A classification scheme based on directed acyclic graphs for acoustic farm monitoring. In: 2018 23rd Conference of Open Innovations Association (FRUCT), pp. 276–282, November 2018. <https://doi.org/10.23919/FRUCT.2018.8588077>
8. Ntalampiras, S.: Moving vehicle classification using wireless acoustic sensor networks. *IEEE Trans. Merg. Top. Comput. Intell.* **2**(2), 129–138 (2018). <https://doi.org/10.1109/TETCI.2017.2783340>
9. Ntalampiras, S.: Directed acyclic graphs for content based sound, musical genre, and speech emotion classification. *J. New Music Res.* **43**(2), 173–182 (2014). <https://doi.org/10.1080/09298215.2013.859709>
10. Ntalampiras, S.: A transfer learning framework for predicting the emotional content of generalized sound events. *J. Acoust. Soc. Am.* **141**(3), 1694–1701 (2017). <https://doi.org/10.1121/1.4977749>
11. Ntalampiras, S.: Bird species identification via transfer learning from music genres. *Ecol. Inf.* **44**, 76–81 (2018). <https://doi.org/10.1016/j.ecoinf.2018.01.006>. <https://www.sciencedirect.com/science/article/pii/S1574954117302467>
12. Ntalampiras, S., Fakotakis, N.: Speech/music discrimination based on discrete wavelet transform. In: Darzentas, J., Vouros, G.A., Vosinakis, S., Arnellos, A. (eds.) SETN 2008. LNCS (LNAI), vol. 5138, pp. 205–211. Springer, Heidelberg (2008). [https://doi.org/10.1007/978-3-540-87881-0\\_19](https://doi.org/10.1007/978-3-540-87881-0_19)
13. Pramono, R.X.A., Bowyer, S., Rodriguez-Villegas, E.: Automatic adventitious respiratory sound analysis: a systematic review. *PloS One* **12**(5), e0177926 (2017). <https://doi.org/10.1371/journal.pone.0177926>
14. Rabiner, L.R.: A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* **77**(2), 257–286 (1989). <https://doi.org/10.1109/5.18626>
15. Ren, Y., Johnson, M.T., Tao, J.: Perceptually motivated wavelet packet transform for bioacoustic signal enhancement. *J. Acoust. Soc. Am.* **124**(1), 316–327 (2008). <https://doi.org/10.1121/1.2932070>
16. Reynolds, D.A., Rose, R.C.: Robust text-independent speaker identification using Gaussian mixture speaker models. *IEEE Trans. Speech Audio Process.* **3**(1), 72–83 (1995). <https://doi.org/10.1109/89.365379>
17. Rizal, A., Hidayat, R., Nugroho, H.A.: Signal domain in respiratory sound analysis: methods, application and future development. *J. Comput. Sci.* **11**(10), 1005–1016 (2015). <https://doi.org/10.3844/jcssp.2015.1005.1016>
18. Rocha, B.M., et al.: A respiratory sound database for the development of automated classification. In: Maglavera, N., Chouvarda, I., de Carvalho, P. (eds.) Precision Medicine Powered by pHealth and Connected Health. IP, vol. 66, pp. 33–37. Springer, Singapore (2018). [https://doi.org/10.1007/978-981-10-7419-6\\_6](https://doi.org/10.1007/978-981-10-7419-6_6)

19. Serbes, G., Ulukaya, S., Kahya, Y.P.: An automated lung sound preprocessing and classification system based on spectral analysis methods. In: Maglaveras, N., Chouvarda, I., de Carvalho, P. (eds.) Precision Medicine Powered by pHealth and Connected Health. IP, vol. 66, pp. 45–49. Springer, Singapore (2018). [https://doi.org/10.1007/978-981-10-7419-6\\_8](https://doi.org/10.1007/978-981-10-7419-6_8)
20. Taylor, P.: The target cost formulation in unit selection speech synthesis. In: Ninth International Conference on Spoken Language Processing, ICSLP, INTERSPEECH 2006, Pittsburgh, PA, USA, 17–21 September 2006 (2006). [http://www.isca-speech.org/archive/interspeech\\_2006/i06\\_1455.html](http://www.isca-speech.org/archive/interspeech_2006/i06_1455.html)
21. Torrence, C., Compo, G.P.: A practical guide to wavelet analysis. Bull. Am. Meteorol. Soc. **79**, 61–78 (1998)
22. VanderWeele, T.J., Robins, J.M.: Signed directed acyclic graphs for causal inference. J. Roy. Stat. Soc.: Ser. B (Stat. Method.) **72**(1), 111–127 (2010). <https://doi.org/10.1111/j.1467-9868.2009.00728.x>
23. Zhao, Y., Zhang, C., Soong, F.K., Chu, M., Xiao, X.: Measuring attribute dissimilarity with HMM KL-divergence for speech synthesis, 6 p. (2007)



# Eye Disease Prediction from Optical Coherence Tomography Images with Transfer Learning

Arka Bhowmik<sup>(✉)</sup>, Sanjay Kumar<sup>(✉)</sup>, and Neeraj Bhat<sup>(✉)</sup>

Department of Computer Science and Engineering, Delhi Technological University, New Delhi, India

arka.bhowmik95@gmail.com, sanjay.kumar@dtu.ac.in,  
sanjaykr6090@gmail.com, neerajbhat98@gmail.com,

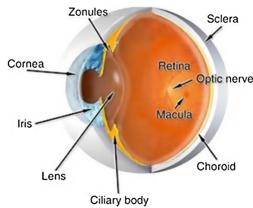
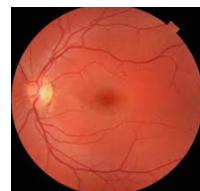
**Abstract.** Optical Coherence Tomography (OCT) of the human eye are used by optometrists to analyze and detect various age-related eye abnormalities like Choroidal Neovascularization, Drusen (CNV), Diabetic Macular Odeama (DME), Drusen. Detecting these diseases are quite challenging and requires hours of analysis by experts, as their symptoms are somewhat similar. We have used transfer learning with VGG16 and Inception V3 models which are state of the art CNN models. Our solution enables us to predict the disease by analyzing the image through a convolutional neural network (CNN) trained using transfer learning. Proposed approach achieves a commendable accuracy of 94% on the testing data and 99.94% on training dataset with just 4000 units of data, whereas to the best of our knowledge other researchers have achieved similar accuracies using a substantially larger (almost 10 times) dataset.

**Keywords:** Deep learning · Transfer learning · Convolutional Neural Networks · Artificial intelligence · Bioinformatics · Age-related macular degeneration · Choroidal Neovascularization · Pneumonia · Diabetic Retinopathy · Diabetic Macular Edema · Optical Coherence Tomography

## 1 Introduction

OCT is used to obtain structural images of the human eye, which is otherwise not visible to the naked eye. It has revolutionized the evaluation for treatment of many eye diseases, some of which also may lead to blinding. OCT, performed by looking inside the machine, keeping still, captures high resolution images of the back of the eye which can be used for retina scanning (A retinal scan uses retina blood vessel patterns of a person's eye). Figures 1 and 2 shows parts of a normal eye and its retina respectively.

DRUSEN: Drusen [2] (Fig. 3) predominantly occurs due to aging, when yellow extracellular particles accumulate between the Bruch's membrane and retinal pigment of the eye. Drusen can lead to clogging of the transport system, which may prevent cone cells (responsible for color vision) from receiving enough oxygen and gradually lead to their death. Drusen leads to less vivid colors and makes one's central vision blurry.

**Fig. 1.** Parts of eye**Fig. 2.** Retina

CNV: Choroidal Neovascularization [3] (Fig. 4) occurs due to the formation of new blood vessels near the choroid. Defects in the Bruch's membrane located in the innermost part of choroid, extreme myopia, excessive vascular endothelial growths are causes of CNV. CNV causes distortion in central vision and pressure can be felt at the back of the eye.

DME: Diabetic Macular Edema [4] (Fig. 5) occurs mainly in diabetic patients, resulting in blurred vision due to the macula beginning to fill with fluid. The cone cells are impaired of being able to sense light and hence results in blurred vision. DME occurs when the blood vessels at the back start widening.

Using a subset of 4000 images from a large dataset of 85000 OCT scans, we trained 2 deep CNN models, using Keras library in python, to get maximum average validation accuracy of 94%.

**Fig. 3.** Drusen**Fig. 4.** CNV**Fig. 5.** DME

## 1.1 Convolutional Neural Networks

Convolutional Neural Networks [5] are essentially deep feed-forward networks containing a series of convolutional and pooling layers which specialize in classifying image data specifically. Initial layers are not fully connected as image data dimensions are too large. Input layer takes in images of a reduced dimension like  $224 \times 224$  (here each pixel value in RGB is one input and the total input is of the dimension of the total number of pixels) and for the output, we have a fully connected layer followed by some normalization, producing a single 1-dimensional vector, with greater depth.

### 1.1.1 VGG16

VGG16 is a state-of-the-art CNN model introduced by Karen Simonyan, Andrew Zisserman [6], for the ImageNet 2014 competition. It comprises of 23 layers, and is

available for public use along with the pretrained weights (trained in ImageNet which is a source of over a million images), to classify objects belonging to 1000 classes. This model architecture thus has the outermost layer generating 1000 values for the 1000 classes of objects, for a given input image of dimension no less than (32,32,3). It has 138,357,544 trainable parameters and its size is approx. 528 MB.

### 1.1.2 Inception V3

Similar to VGG16, Inception V3 [7] is also a state of the art, very deep (159 layer) CNN model for Image classification, and 23,851,784 trainable parameters. It supports images of dimension no less than (75,75,3). Size of InceptionV3 is 92 MB.

## 1.2 Transfer Learning

Transfer Learning [8] is a process of reusing pre-trained models of state-of-the-art networks to classify data of a new domain, which saves the users a lot of processing time by avoiding training of data from scratch. Training state of the art CNNs from scratch normally would have taken at least 2 to 3 weeks on high end systems. Usually in transfer learning the initial weights of the pretrained networks are unaltered since they are used to distinguish basic features like colors and shapes. The outermost layers have trainable weights which are altered based on the new data. The following specifies the training practices based on respective requirements:

- (i) ‘New dataset is small and similar to original dataset’: Changing core elements of the weights of a complicated network for small dataset leads to overfitting. It is wise not to alter the whole model, in this case. Also, the data being similar, it is useless to alter the initial layers and only higher-level features may be relevant to change. It is better to train a linear classifier on top of our present architecture in this case.
- (ii) ‘New dataset is large and similar to the original dataset’: Having more data will not overfit the network and thus we can train the upper layers of the network without any difficulties.
- (iii) ‘New dataset is small but very different from the original dataset’: For such cases, it is better to reduce our model off a few layers and train a simple SVM classifier for such cases to prevent overfitting and easily classify our data.
- (iv) ‘New dataset is large and very different from the original dataset’: In such cases, we can afford to train our classifier from scratch as data is large. We might require to train the model from the starting layers too. Also, we can initially try to get good accuracies by initially just training the upper layers and if that fails, try training from lower layers. Our problem statement belongs to this case.

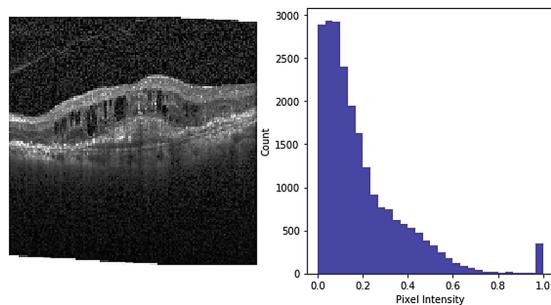
The remainder of this paper is organized as follows. Section 2 surveys related works about medical image analysis and related to our research. Section 3 introduces our dataset and visualization. In Sect. 4, we describe our CNN based proposed model. Improved results obtained are depicted in Sect. 5. Finally, we presented conclusions in Sect. 6.

## 2 Related Work

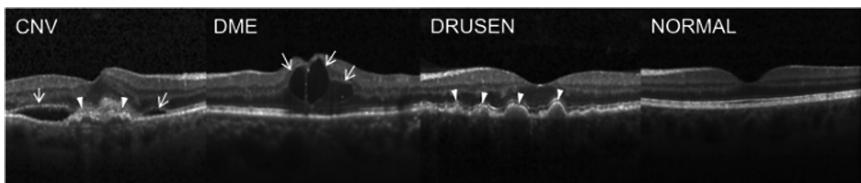
Medical image analysis needs a strong dataset of diverse images. The significance of creating a dataset from patients of diverse age and races are to avoid bias in prediction process. Also, the evaluation of learning procedure of a model being trained, gives us a clear understanding of the volume and impact of medical data, which is being fed into the training model. Many such key points of consideration have been pointed out in Marleen's article [9]. A recent work [10] studies the formation of Myocardial Infarction by processing cardiac magnetic resonance images in a region based CNN, to locate left ventricle and then uses a stacked auto encoder-decoder model to assess the local motion information of the sequences. The model achieved 87.6% accuracy in prediction of infarct locations. Shie et al. in [11] have used a transfer learning approach for Otitis Media images. They have extracted encoded features from the Otitis Media images using transfer learning on deep CNN, AlexNet and then with those extracted features, they have trained an SVM classifier on top of AlexNet to get an average accuracy of 88.5%. Treader et al. in [12] describes utilizing a deep CNN in TensorFlow using an SD-OCT dataset of 1112 exudative AMD eye and normal eye scans to achieve an accuracy of 99.7%. In [13] Phillip Prahs describes the use of transfer learning on a dataset of 1.8 lakh OCT images to detect Antivascular endothelial growth with a 95.5% accuracy. Comparing to [12] and [13], our model classifies over a larger variety of eye diseases. Kermany et al. [14] have performed a research work similar to our paper by using transfer learning on OCT images and also chest X-ray images, using VGG16 network, reaching high accuracies. They achieved maximum accuracy of 96.6% training over 85000 OCT images which they had collected over a period of 4 years. We have used the same dataset, but demonstrated that our model achieves similarly high accuracies by using a significantly smaller part of the dataset, hence significantly reducing training and increasing availability and performance.

## 3 Dataset and Visualization

The dataset consists of exactly 84495 grayscale images taken by the OCT device and labelled in the 4 categories of NORMAL, CNV, DME and DRUSEN by experienced optometrists. The dataset has been downloaded from Kaggle and it was created originally by [15] "Shiley Eye Institute of the University of California San Diego, the California Retinal Research Foundation, Medical Center Ophthalmology Associates, the Shanghai First People's Hospital, and Beijing Tongren Eye Center" over a period of 4 years, starting from July 2013. Every sample has been assessed by professionals through multiple levels of expertise. Following Fig. 6 shows the pixel intensities of a random sample (here CNV) from the dataset. Figure 7, from [15] shows the comparative study of the OCT scans for our classes.

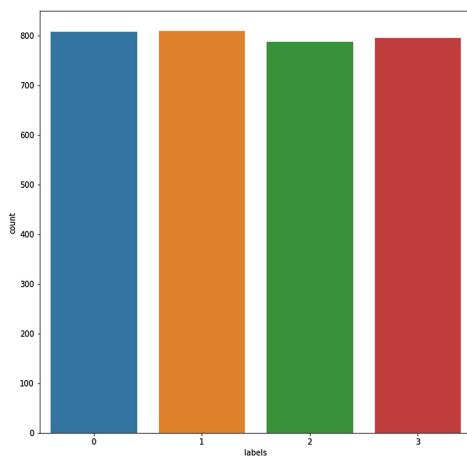


**Fig. 6.** Pixel intensities of an OCT scan sample



**Fig. 7.** Various eye diseases

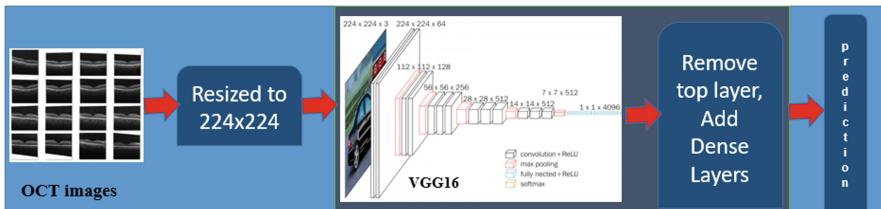
We have taken a subset of the dataset (exactly 4000 samples, 1000 per class) so that the training time is less. Following figure shows frequency of NORMAL, CNV, DME, DRUSEN in a random sample batch of size 728, used for training. Its distribution is shown in Fig. 8.



**Fig. 8.** Even distribution of our dataset

## 4 Implementation

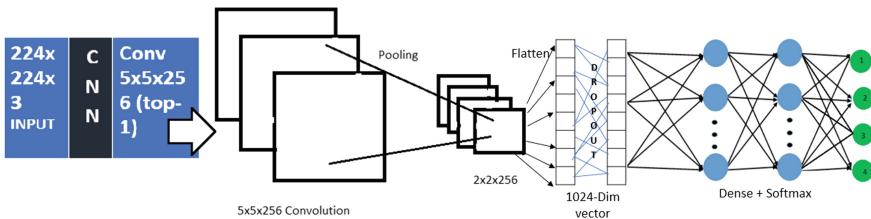
For our scenario we have used the VGG16 and InceptionV3 model for transfer learning using the pretrained ImageNet weights. The models are available through the Keras library along with the weights. The outermost fully connected layer was removed leaving a convolutional layer with an output dimension of [5,5,256]. We add to this a pooling layer of size  $2 \times 2$  to scale down the feature size, followed by a flattening layer and a Dense (perceptron) layer. The Dense layer with a SoftMax activation function takes in a 1024 size vector from the flattening module and outputs a vector of size 4. Initially, in the pre-processing stage, the images have been resized to  $224 \times 224$  resolution and reshaped into NumPy arrays (resulting dimension [224,224,3]). The data labels have been categorized into 4 labels, 1, 2, 3, 4 and coded into their 1-hot vector representations. We froze the layers of the original model so that they were not updated in the training. Only our layers were trained. We used both Adam and RMSProp to optimize the training process, with a learning rate of 0.00005. Adam [16], as expected performed slightly better. For evaluation of the model's loss, we have based it on categorical cross entropy. A loss function is a measure of how bad the model predicted for that epoch and usually is measured by mean squared error. We ran the training for 50 epochs on a dataset of size 4000 with equally balanced data from all classes and train and test split of 0.2. The system was tested on both VGG16 and InceptionV3 models. In about 12 epochs, the Inception V3 model had a faster learning rate for small dataset, than the VGG16, without the dropout layer, although VGG16 performed better overall. Although our scenario was that of type 4 as mentioned in Sect. 1.2, fortunately, we did not need to train the whole network from scratch to get good results. Training the uppermost layer was sufficient. Figure 9 describes the whole training process. Adding a dropout [17] layer with dropout rate of 0.5, before the final layer actually yielded better results than the previous model.



**Fig. 9.** Overview of the whole process

Dropout layers are used for regularization by “dropping out” the contribution of a few randomly selected neurons during training. This helps other active neurons to contribute more to the features which were otherwise assigned to be determined by the currently deactivated neurons, thus helping them generalize the handling of data better. This helps in building multiple independent representations of the subject being learned by the network. Using a dropout of 0.5, we achieved 1 to 2 percent improvements in accuracies than our base model. Figure 10 describes the layers of the VGG16 CNN

model we have altered, for this research work. The CNN in the figure corresponds to a normal VGG16 with the top layer removed.



**Fig. 10.** Detailed description of our CNN model

## 5 Results

The proposed network was trained on a device with Nvidia k80 12 GB GPU and 32 GB RAM, running on Intel Xeon processor. Testing with our network with a training set of size 3200 and validation of 800 samples produces 94% accuracy. It is noteworthy that such high accuracy is achieved using a transfer learning model on a dataset which is comparatively so small. Studying other approaches for the same problem, it is seen that, increasing the data obviously enhances the accuracy only by a small percentage. But it takes considerably larger amount of time to train in such cases. Our model takes a running time of 41 s per epoch (one epoch running over all 4000 data records). Here is a comparative study of similar models with substantially more data and our model with lesser data (Table 1).

**Table 1.** Comparative performance of related models

Approach	Training dataset size (#images)	Accuracy
Kermany 2018 best model [12]	80000	96.6%
Our model	4000	94%
Human performance	Medical training	90–99%

Brief description of evaluation metrics used: Precision and Recall. [18] are measures of the amount of data used and the amount of which was beneficial for the training. Precision is a ratio of the amount of useful results, to the total retrieved results. Recall is the ratio of the fraction of relevant results we received, to the total number of relevant instances.

$$\text{Precision } P = \frac{TP}{TP + FP} \quad (1)$$

$$\text{Recall } R = \frac{TP}{TP + FN} \quad (2)$$

$$F = 2 * ((\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})) \quad (3)$$

Where TP = True Positive (predicted values = actual values and both are positive), FP = False Positive or Type 1 error (Predicted values to be positive when they were actually false) FN = False Negative (Predicted values to be negative but they were false) Finally, the F score is a measure of how close Precision and Recall are (it is their Harmonic Mean).

**VGG16 Results:** The average accuracy was 91.6% without the dropout layer, over 12 epochs on a 1000 size dataset (Fig. 10) and with dropout layer, and 4000 size dataset, accuracy boosted to 94%, in about 33 epochs, after which the process converged by early stopping. The precision, recall and f-scores for each class are shown in the figure.

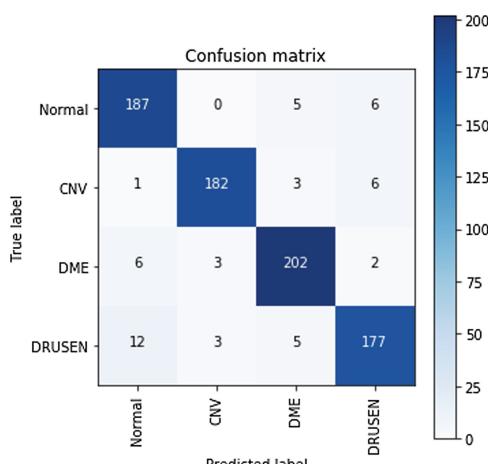
The support tag signifies the number of validation samples for each class. The average recorded accuracy on the training dataset, however, was 99.4%.

**Inception V3 Results:** It recorded average prediction accuracy of 92.6% without the dropout layer using a 1000 size dataset and no further improvement was noticed when using a larger dataset of 3200 training samples (Fig. 11).

	precision	recall	f1-score	support		precision	recall	f1-score	support
Normal	0.86	1.00	0.92	42	Normal	0.91	0.94	0.93	198
CNV	0.91	0.95	0.93	42	CNV	0.97	0.95	0.96	192
DME	1.00	0.81	0.89	42	DME	0.94	0.95	0.94	213
DRUSEN	0.93	0.98	0.92	42	DRUSEN	0.93	0.90	0.91	197
avg / total	0.92	0.92	0.92	168	avg / total	0.94	0.94	0.94	800

**Fig. 11.** Results without and with Dropout layer respectively

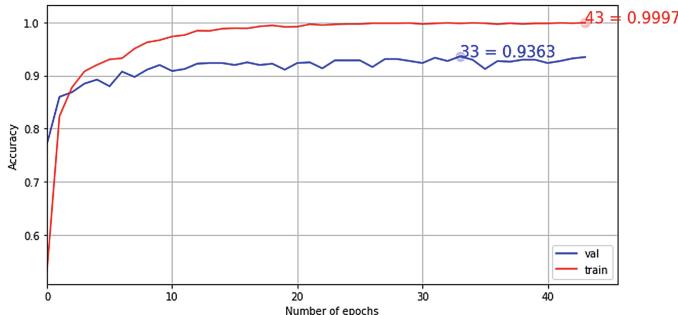
Confusion matrices are best for visualizing multiclass-classification performance where a one-versus-all comparison of predictions are made. Figure 12 shows the confusion matrix for the VGG16 based model. Rows denote the ground truth labels



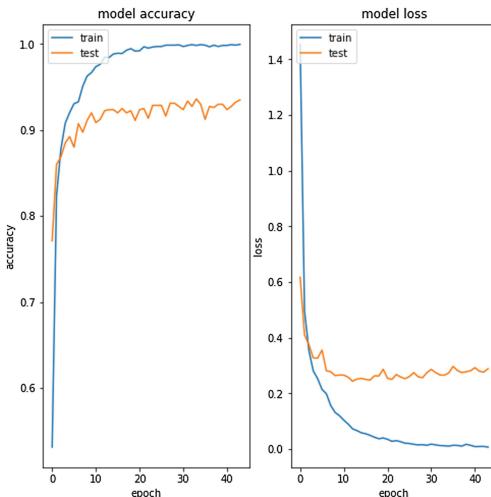
**Fig. 12.** Confusion matrix for VGG16 based model

while columns denote the predicted labels. The dark tiles show the number of current label predictions while the light tiles denote incorrect predictions. It gives us a clearer overview of the model's performance. From the results we notice that differentiating between Normal eye and Drusen eye has been more challenging than other disease predictions.

Figures 13 and 14 show the learning performance of our VGG16-based model throughout the training process. Our model provides excellent results, from the 10<sup>th</sup> epoch itself. After the 10<sup>th</sup> epoch, the model starts to converge at a slower rate, ultimately reaching a training accuracy of 99.7 at the 40<sup>th</sup> epoch, and the model stops automatically by early stopping [19] regularization.



**Fig. 13.** Learning curve of VGG16 based model



**Fig. 14.** Loss curve over 50 epochs

## 6 Conclusion

Our proposed model proves to be highly efficient in computation, preserving quality of prediction on a significantly smaller dataset and showcases that sometimes even smaller datasets of around 4000 images are enough to get good accuracies for medical experiments, especially when the outputs are of a smaller number of classes. It also highlights the importance and benefits of using transfer learning in CNN models to get the best out of the state-of-the-art models without complex hardware equipment. Thus, this is a very economically feasible approach for analyzing medical datasets for disease prediction. Similar models can be used for other image datasets also, to get fast and accurate results, by changing a few parameters in the model. There is a lot of room for further experimentation with our model also. We might achieve greater accuracy by using more training samples on our data. Also, instead of using another dense CNN layer on top of the model, we can train a simpler classifier like multiclass SVM, when we have very few output classes. SVMs might perform better than the convolutional layer in such cases, even with the little amount of data we have trained with. Also to enhance the usefulness of the dataset in further studies regarding disease formation, we can use semantic segmentation. Semantic segmentation [20] focusses on every entity instead of just focusing on a few pixels to predict objects. For example, it can detect the number of people in an image if the image is labelled to have contain a human image. Hence semantic segmentation can help detect nanoscale changes [21] in the retinal layers which in turn may signify the process of formation of such diseases. Semantic segmentation is a new concept that can find great usefulness as far as biological studies are concerned.

## References

1. Brezinski, M.E., Fujimoto, J.G.: Optical coherence tomography: high-resolution imaging in nontransparent tissue. *IEEE J. Sel. Top. Quantum Electron.* **5**(4), 1185–1192 (1999)
2. Hunter, A.A., Chin, E.K., Almeida, D.R., Telander, D.G.: Drusen imaging: a review. *J. Clin. Exp. Ophthalmol.* **5**(327), 2 (2014)
3. Amaro, M.H., Holler, A.B.: Age-related macular degeneration with choroidal neovascularization in the setting of pre-existing geographic atrophy and ranibizumab treatment. Analysis of a case series and revision paper. *Revista Brasileira de Oftalmologia* **71**(6), 407–411 (2012)
4. Bressler, N., et al.: Optimizing management of diabetic macular edema in Hong Kong: a collaborative position paper. *Hong Kong J. Ophthalmol.* **21**(2), 59–64 (2017)
5. Kumar, S., Kumar, M.: A study on the image detection using convolution neural networks and TensorFlow. In: 2018 International Conference on Inventive Research in Computing Applications (ICIRCA), pp. 1080–1083. IEEE (2018)
6. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint [arXiv:1409.1556](https://arxiv.org/abs/1409.1556) (2014)
7. Szegedy, C., et al.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
8. Pan, S.J., Yang, Q.: A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **22**(10), 1345–1359 (2010)

9. de Bruijne, M.: Machine learning approaches in medical image analysis: from detection to diagnosis (2016)
10. Chen, M., Fang, L., Zhuang, Q., Liu, H.: Deep learning assessment of myocardial infarction from MR image sequences. *IEEE Access* **7**, 5438–5446 (2019)
11. Shie, C.-K., Chuang, C.-H., Chou, C.-N., Wu, M.-H., Chang, E.Y.: Transfer representation learning for medical image analysis. In: IEEE Engineering in Medicine and Biology Society, Conference, pp. 711–714 (2015)
12. Treder, M., Lauermann, J.L., Eter, N.: Automated detection of exudative age-related macular degeneration in spectral domain optical coherence tomography using deep learning. *Graefe's Arch. Clin. Exp. Ophthalmol.* **256**(2), 259–265 (2018)
13. Prahs, P., et al.: OCT-based deep learning algorithm for the evaluation of treatment indication with anti-vascular endothelial growth factor medications. *Graefe's Arch. Clin. Exp. Ophthalmol.* **256**(1), 91–98 (2018)
14. Kermany, D.S., et al.: Identifying medical diagnoses and treatable diseases by image-based deep learning. *Cell* **172**(5), 1122–1131 (2018)
15. Cell. [http://www.cell.com/cell/fulltext/S0092-8674\(18\)30154-5](http://www.cell.com/cell/fulltext/S0092-8674(18)30154-5)
16. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv: [1412.6980](https://arxiv.org/abs/1412.6980) (2014)
17. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**(1), 1929–1958 (2014)
18. Towards Datascience. <https://towardsdatascience.com>
19. Prechelt, L.: Early stopping - but when? In: Orr, Genevieve B., Müller, K.-R. (eds.) *Neural Networks: Tricks of the Trade*. LNCS, vol. 1524, pp. 55–69. Springer, Heidelberg (1998). [https://doi.org/10.1007/3-540-49430-8\\_3](https://doi.org/10.1007/3-540-49430-8_3)
20. 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)
21. Van Engelen, A., et al.: Multi-center MRI carotid plaque component segmentation using feature normalization and transfer learning. *IEEE Trans. Med. Imaging* **34**(6), 1294–1305 (2015)



# Severe Asthma Exacerbations Prediction Using Neural Networks

Arthur Silveira<sup>(✉)</sup> ID, Cristian Muñoz ID, and Leonardo Mendoza

Pontifícia Universidade Católica, Rio de Janeiro, RJ 22451-900, Brazil  
arthur.silveirasc@gmail.com

**Abstract.** This work introduces a classification model using neural networks on the severity for asthma exacerbations [1] from patients who seek the Brazilian healthcare system in order to get a first treatment of their condition. Healthcare specialists, who work on related databases, usually have access to the information about whether these events need a critical handling or not. However, in many situations, these databases present missing data, which usually demands a manual evaluation of this data and, therefore, time.

Hence, the aim of this work is to automate the classification process on the asthma emergency cases – by training and testing neural networks – and compare its performance with other classifiers. The results will be part of the analysis and assessments routines performed by specialists of a healthcare company.

**Keywords:** Deep learning · Healthcare · Neural network

## 1 Background and Motivation

Asthma exacerbations and the general disease mishandling are one of the main sources of hospitalizations, emergency treatment and patients' life quality worsening, causing, in some cases, irreversible pulmonary obstruction after a few years of its poor controlling [2]. Emergency treatment and hospitalizations, after a primary care, can be assessed under several perspectives from related databases, such as the type of treatment performed, or the medication applied.

In order to evaluate these cases, databases are managed and analyzed, especially under the perspective if the assistance in the hospital resulted in an emergency handling or not.

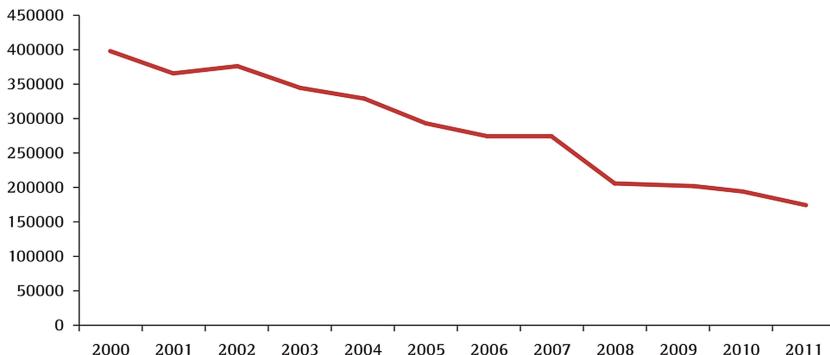
Despite its importance, this information is missing in some cases, affecting the quality of the assessments developed by specialists who usually have access to it. It's critical to them, for example, to understand which age or sex is typically involved in emergency cases of asthma exacerbations or even what drugs are being managed.

Hence, this work aims to train a neural network in order to automate the emergency cases prediction and to avoid a long manual analysis of it. Besides, we intend to evaluate its accuracy and calculate other performance metrics – such as the model loss function – to better understand how precise the neural network is and compare it with other machine learning algorithms.

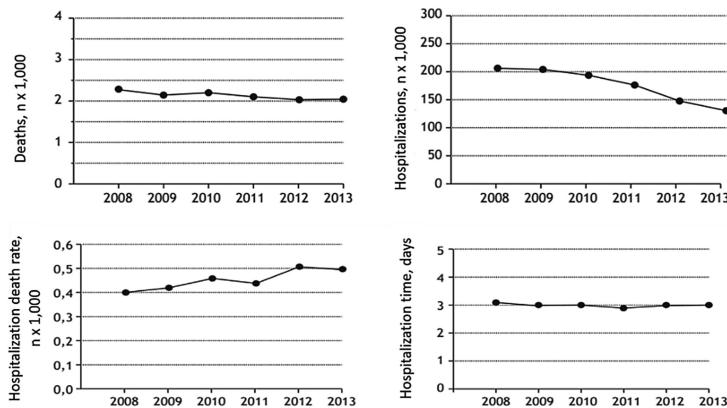
## 2 The Costs of Asthma in Brazil

Asthma is a chronic inflammatory disease associated to a hyper-responsivity of the respiratory tract, which induces to recurrent episodes of thoracic oppression and cough, particularly at night or early morning. These episodes are consequence of a generalized respiratory tract obstruction, which can be reversed naturally or by treatment [3].

It's a common chronic condition in children and adults, being a problem, which affects, in Brazil, roughly 20 million people [4]. The asthma hospitalization rate of people with more than 20 years in the public healthcare system declined 49% between 2000 and 2010 and, in 2011, it was recorded by the DATASUS 160,000 hospitalizations among all ages, a figure that has put asthma as the fourth major cause of critical care in healthcare units (see Fig. 1). Despite this decline, the mortality rate between hospitalized patients (i.e., emergency cases) increased approximately to 25% between 2008 and 2013 (see Fig. 2) and represented a cost to the public healthcare system of US \$ 170 million (see Fig. 3).



**Fig. 1.** Hospitalizations caused by asthma, between January 2000 and December 2011

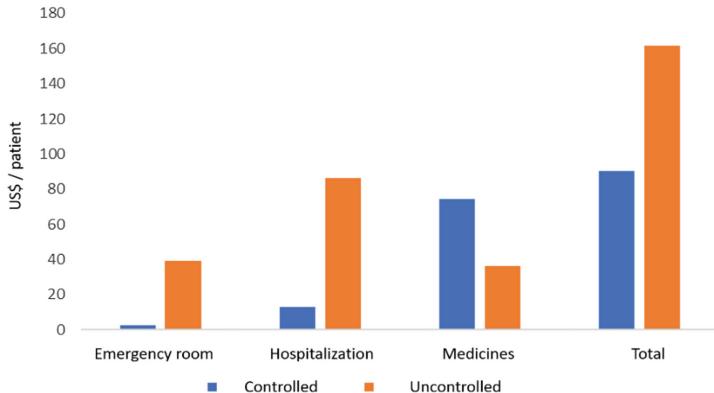


**Fig. 2.** Asthma mortality and hospitalization landscape in Brazilian public healthcare system

Year	Hospitalization, n	Cost, USD	Average Cost, USD
2008	205.276	30.195.020,86	147,09
2009	203.649	32.708.217,35	160,61
2010	193.017	31.165.431,83	161,46
2011	175.955	28.720.338,89	163,23
2012	146.559	24.153.812,17	164,81
2013	129.728	21.490.888,95	165,66
Total	1.054.184	168.433.710,05	160,48

**Fig. 3.** Total asthma hospitalizations and their costs in Brazilian public healthcare system

To be more precise regarding the asthma expenses, in 2006 [5], a study was conducted with controlled asthma patients – i.e., people with daytime symptoms manifestations or nighttime manifestations twice a week– and uncontrolled ones – i.e., people with daytime symptoms manifestations more than twice a week, or with nighttime manifestations in two consecutive nights, or even using a relieve medication more than twice a week. In this report, the direct costs of an uncontrolled patient were US\$ 39,15 for the use of emergency rooms, US\$ 86,30 for the hospitalization procedures and US\$ 36,20 for the medicine handling, whereas the costs for a controlled patient was US\$ 2,70 for the emergency rooms, US\$ 12,88 for the hospitalization procedures and US\$ 74,50 for the drug application (see Fig. 4). Thus, regarding the indirect costs, the days lost at school or work were larger in the uncontrolled group (54 vs. 30 school days, 48 vs. 12 work days).



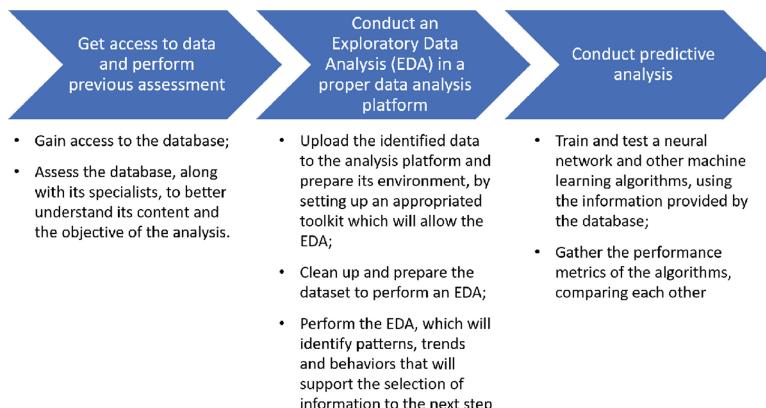
**Fig. 4.** Direct costs distribution between controlled and uncontrolled asthma patients

Hence, the study showed that patients with uncontrolled asthma have a bigger financial impact when compared to the controlled ones. The bigger costs proportion are related to the use of emergency rooms and hospitalizations, which imply that there are opportunities to cost reduction, disease controlling (by using medicines) and hospitalizations decrease. Analysis which try to understand the exacerbations behavior that induce emergency handle, from a series of characteristics analyzed during the patient primary care, may improve the creation of better strategies to control the disease and diminish its weight on the public or private healthcare budget in Brazil. By offering information and structured treatment to the patients, a bigger control of the condition, a cost reduction and a better well-being can be achieved.

### 3 Methodology

#### 3.1 A Basic Deep Learning Workflow

The general process of the project follows the commons steps of a basic deep learning project (see Fig. 5):



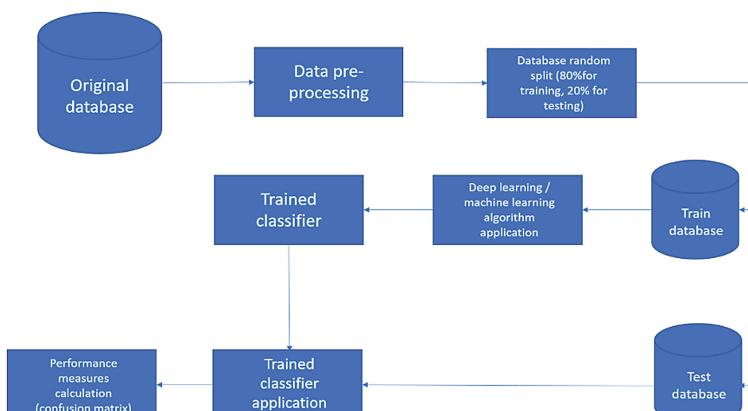
**Fig. 5.** The three-stage process to conduct this study

It's important to remind that each activity listed above breaks down into other essential ones, such as:

- Clean up and prepare the dataset to perform an EDA:
  - Exclude unnecessary variables and not available (N/A) data;
- Perform the EDA, which will identify patterns, trends and behaviors that will support the selection of information to the next step:
  - Examine database balance according the input and output variables;
  - Visualize database general behavior, capturing patterns from it;
  - Select the variables that will be used to train the classifier algorithm.
- Train and test a neural network and other machine learning algorithms, using the information provided by the database:
  - Split a train and test database samples in order to perform these tasks;
  - Train and test a neural network, along with other classifiers, to predict the type of care, using relevant variables identified in the EDA.

### 3.2 Classification Methods

Classification problems in deep learning and machine learning are dedicated to study how it's possible to automatically learn from past behavior to make precise predictions about the future [6]. In this field of study, the objective is to approximate a mapping function  $f$ , from input variables  $X$ , in order to predict values of an output variable  $y$  [7]. Usually, in such cases, a sample of a database is already labeled (classified) – in our case, telling us if a patient's uncontrolled asthma event should be treated as an emergency or not – and used to train a classification algorithm. Afterwards, the trained model is applied on the remaining sample – the test data – and accuracy and other performance metrics are measured, by comparing the predicted values with the true ones (see Fig. 6).



**Fig. 6.** A detailed view of the predictive analysis

### 3.3 System Architecture

#### 3.3.1 The Database

The databases used for this work were accessed just by one of the authors, in password secured computer, with absolutely no type of sharing with the others and third parties. Besides, it was impossible to access any identifiable patient information and the data only included unidentified variables, with no means to know the identity of the patient.

#### 3.3.2 The Neural Network Architecture

In general, there are three main categories of neural networks architecture: *feed-forward*, *recurrent* and *symmetric* [9]. In this research, a multilayered Perceptron recurrent neural network will be applied, with 112 neurons each, since the total dimension of the input values – label variable included – was 56. After each hidden layer, a dropout and a ReLU activation function were established. In the output layer, one neuron – since it's a binary classification problem –, with a sigmoidal activation function, was included.

Regarding evaluation metrics, besides the confusion matrix, a cross-entropy method was used as loss function (below, its mathematical representation [10]), due to its good performance with sigmoidal activation functions:

$$L = -\frac{1}{n} \sum_{i=1}^n (y^{(i)} * \log(\hat{y}^{(i)}) + (1 - y^{(i)}) * \log(1 - \hat{y}^{(i)})) \quad (1)$$

### 3.4 Other Classification Methods Employed

Below, some other classification methods employed whose accuracy were compared with the neural network:

- *Specialists' classification*: the database specialists created their own manual method, in which some variables were observed, and the final emergency label was given;
- *Naïve Bayes classifier*: a supervised learning approach in which the Bayes' theorem is applied, with its naïve assumption that the input variables are independent;
- *Decision tree*: another supervised learning method, based on a non-parametrical heuristic, in order to predict output values, by setting up simple decision rules extracted from the database understanding;
- *Random forest*: widely used in classification and regression problems, this method builds multiples decision trees and collect the results in order to create a more accurate algorithm;
- *Support Vector Machines (SVM)*: this algorithm builds a hyperplane in a multidimensional space used to separate different classes iteratively, by minimizing its error. There are different ways to construct the hyperplane (called *kernels*) and here we employed the *linear*, *polynomial* and *Gaussian* methods.

## 4 Implementation

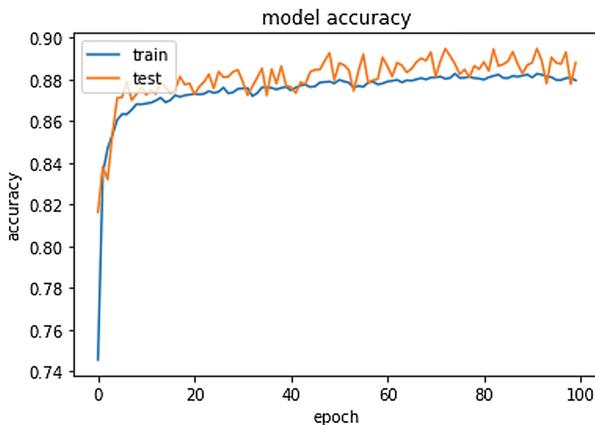
### 4.1 Exploratory Data Analysis (EDA), Pre-processing, Training and Test

An EDA is crucial to uncover patterns in the database and can be extremely helpful for the neural network classification performance. For example, other variables can be created from the assessment of the original ones – in order to make the analysis more insightful. In this research, two variables were created and helped in the whole process.

After the EDA, the final variables were selected and passed through a preprocessing and an one-hot encoding [11] processes in order to enable the data usage by the model. The separation between a train (with 72.946 entries – 80% of the database) and a test samples (with 18.237 entries – 20% of the database) occurred at the end.

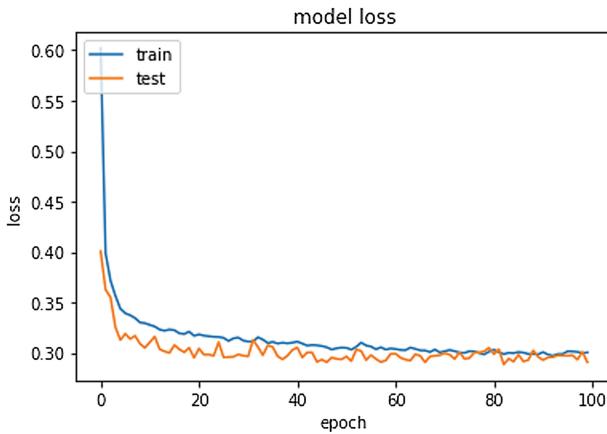
## 5 Results

After 100 epochs, these are the results of our neural network model (Figs. 7, 8 and 9).



**Fig. 7.** Model accuracy behavior

		Predicted	
		Non-emergency	Emergency
Actual	Non-emergency	6,638 (36,4%)	930 (5,1%)
	Emergency	986 (5,4%)	9,683 (53,1%)

**Fig. 8.** Confusion matrix**Fig. 9.** Loss function behavior

The neural network achieved a maximum accuracy of 89,5% (epoch 73) for the test sample, ranking as the third best accuracy between all the classification methods, behind only the specialists' classification and the decision tree method (see Fig. 10. Accuracy level comparison table).

Model	Accuracy
Specialists'	91,2%
Decision tree	89,8%
Neural network	89,5%
Random forest	89,2%
Gaussian SVM	83,0%
Linear SVM	76,2%
Naive Bayes	70,3%
Polynomial SVM	58,4%

**Fig. 10.** Accuracy level comparison table

## 6 Conclusion

Regarding the confusion matrix, a few points need to be noticed: how the algorithm managed to classify better true emergency cases than non-emergency ones (91% vs. 87%). This is probably caused by a slightly higher number of emergency cases in the dataset, which may have leaded to an algorithm specialization. Secondly, it's important to say how the low level of true negatives observed – compared to the other methods assessed – is important to healthcare machine learning applications: saying a patient isn't ill, when she indeed needs treatment, can cost lives and limit the application in real life.

It's noticed, as well, that the specialists' method produced a higher accuracy level. Yet, given the time it took to be completed (roughly six weeks), a trained neural network may yield to an automatized and less time-consuming result.

Finally, new databases, which don't contain the emergency classification, may have it, by applying our trained neural network. Automating this process, new emergency asthma cases can be prevented, by studying its behavior and possible severity diagnoses, leading to more assertive medicine campaigns for pharmaceutical companies, better public healthcare policies and a decrease in hospitalization spending in Brazil.

## References

1. BMJ Best Practices. <https://bestpractice.bmj.com/topics/pt-br/45>. Accessed 02 Dec 2018
2. Jornal Brasileiro de Pneumologia. [http://www.scielo.br/scielo.php?script=sci\\_arttext&pid=S1806-37132006001100002](http://www.scielo.br/scielo.php?script=sci_arttext&pid=S1806-37132006001100002). Accessed 02 Dec 2018
3. Global Initiative for Asthma. [www.ginasthma.org](http://www.ginasthma.org). Accessed 02 Dec 2018
4. Jornal de Pediatria. <http://www.scielo.br/pdf/jped/v82n5/v82n5a06>. Accessed 02 Dec 2018

5. Brazilian Journal of Medical and Biological Research. <http://www.scielo.br/pdf/bjmbr/v40n7/6613.pdf>. Accessed 03 Dec 2018
6. Princeton University. <https://www.cs.princeton.edu/~schapire/talks/picasso-minicourse.pdf>. Accessed 10 Dec 2018
7. Towards Data Science. <https://towardsdatascience.com/machine-learning-classifiers-a5cc4e1b0623>. Accessed 10 Dec 2018
8. Inteligência Computacional Aplicada PUC-Rio. <http://www.ica.ele.puc-rio.br/Arquivos/monografias/TCC%20-%20Daniel%20Ielpo%20Previs%C3%A3o%20de%20Indicadores%20de%20Qualidade%20de%20Servi%C3%A7os%20em%20Rede%20Banda%20Larga%20Fixa%20atrav%C3%A9s%20de%20Redes%20Neurais%20Articiais.pdf>. Accessed 02 Jan 2019
9. Deep Learning Book. <http://deeplearningbook.com.br/a-arquitetura-das-redes-neurais>. Accessed 02 Jan 2019
10. Github. [https://isaacchanghau.github.io/post/loss\\_functions](https://isaacchanghau.github.io/post/loss_functions). Accessed 02 Jan 2019
11. Hackernoon. <https://hackernoon.com/what-is-one-hot-encoding-why-and-when-do-you-have-to-use-it-e3c6186d008f>. Accessed 03 Jan 2019



# A New Generalized Neuron Model Applied to DNA Microarray Classification

Beatriz A. Garro<sup>1</sup> and Roberto A. Vazquez<sup>2(✉)</sup>

<sup>1</sup> IIMAS-UNAM, Ciudad Universitaria, México, D.F., Mexico  
[beatriz.garro@iimas.unam.mx](mailto:beatriz.garro@iimas.unam.mx)

<sup>2</sup> Intelligent Systems Group, Facultad de Ingeniería, Universidad La Salle México,  
Benjamin Franklin 47, Condesa, 06140 México, D.F., Mexico  
[ravem@lasallistas.org.mx](mailto:ravem@lasallistas.org.mx)

**Abstract.** The DNA Microarray classification played an important role in bioinformatics and medicine area. By means of the genetic expressions obtained from a DNA microarrays, it is possible to identify which genes are correlated to a particular disease, in order solve different tasks such as tumor detection, best treatment selection, etc. In the last years, several computational intelligence techniques have been proposed to identify different groups of genes associated with a particular disease; one popular example is the application of artificial neural networks (ANN). The main disadvantage of using this technique is that ANN require a representative number of samples to provide acceptable results. However, the enormous quantity of genes and the few samples available for any disease, demand the use of more robust artificial neural models, capable of providing acceptable results using few samples during the learning process. In this research, we described a new type of generalized neuron model (GNM) applied to the DNA microarray classification task. The proposed methodology selects the set of genes that better describe the disease applying the artificial bee colony algorithm; after that, the GNM is trained using the discovered genes by means of a differential evolution algorithm. Finally, the accuracy of the proposed methodology is evaluated classifying two types of cancer using DNA microarrays: the acute lymphocytic leukemia and the acute myeloid leukemia.

**Keywords:** Generalized neurons • Bioinformatics • DNA microarrays

## 1 Introduction

DNA microarrays have been popularized in medicine field due to this technology is capable of analyzing the expression level of millions of genes at the same time. By means of the analysis, it is possible to perform several tasks such as diagnose diseases, identify different tumors, select the best treatment for a specific patient to resist illness, among others. However, the analysis of this expressions requires of robust computational and statistical technique to obtain the most relevant information from the DNA microarray. Particularly, the Artificial Intelligence

Community has applied several new technique for obtaining this valuable information. A branch of AI, called pattern classification focuses on the identification of different classes or groups analyzing the information contained in the samples; this samples could be associated with a particular disease, making possible the identification different types of cancer. The DNA microarray has an enormous quantity of genes to be analyzed and the samples available are few, this implies that the computational intelligent technique must be capable of learning with few samples in order to be applied in a DNA classification task.

Artificial neural networks (ANN) are computational models that have been applied in different tasks such as pattern recognition, particularly to the classification of DNA microarrays. For example, the authors in [1], describe how an ANN can be used to identify recurrence of cancer after prostatectomy in terms of predictive genes. Other authors, such as in [2], apply an ANN for cancer classification using the singular value decomposition (SVD). In [3], the authors diagnose disease categories of a kind of cancer and in [4], the authors focus their research in mass spectrometry. ANN ensembles based on sample filtering algorithm is designed for separating the wrongly labeled DNA microarrays from the training set and used to construct one more ANN just for the wrong samples classified [5].

The previous papers have in common that the authors first perform a dimensionality reduction in order to select the most representative genes because many of them are irrelevant. If there is not a carefully selection of genes, the consequences can be reflected in the low performance obtained in the classification or prediction a disease. For example, in [6], a selection of genes is performed in terms of the gene ranking based on the significance level, after that, the information obtained during the dimensionality reduction process is used to train an ANN and then classify cancer tumors. Other examples are described in [7] and [8], where the authors apply different swarm intelligence techniques for the dimensionality reduction process and then training an ANN in order to classify the DNA microarrays.

However in order to get acceptable results with an ANN it is necessary to adjust several parameters during its design. The number of synaptic weights, the kind of transfer functions and the number of inputs that will be transformed in the system are important features that directly impact in accuracy during the solution of the problem. For this reason, authors in [9] optimize the ANN design for classification problems.

The generalized neural network (GNN) is other kind of ANN that was developed by [10] with the aim to reduce the design of an artificial neural network without many connections, with a good performance compared with the classic ANN and easy to implement in hardware to solve real time problems. Some works applied GNN to solve approximation functions [11], for computing density estimations [12, 13], prediction and recently DNA classification problems [14]. Basically, the generalized neuron is composed of three neurons, although the results obtained with these model are highly acceptable, if the problem requires to build a generalized network, this network will be composed of three times more

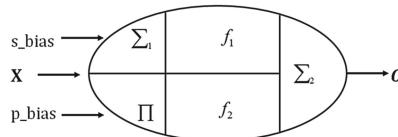
neurons than a classical feed forward network. In that sense, it is still necessary to propose a neuron model that preserves the advantages of the GNN and at the same time reduce the number of neurons used in the model.

In this work, we propose a new generalized neuron model as a generalization of the model proposed by [10], allowing to select automatically different transfer functions, aggregation functions as well as the operators associated with the integration of the input pattern and synaptic weight, all these features in only one neuron. In order to apply the generalized neuron to a DNA microarray classification problem, firstly, the gene dimensionality is reduced using artificial bee colony (ABC) algorithm, then the data is used to train the GNN by means of a differential evolution (DE) algorithm where the main parameters of the model are evolved.

This work comprises five chapters: section one is a brief introduction. Then, the concept of generalized neuron as well as the new generalized model is presented. In addition, the propose methodology is detailed in section three followed by the experimental results in section four. Finally, the conclusions of this work are presented in section five.

## 2 A New Generalized Neurons (GN) Model

The Classical Generalized Neuron (CGN) is a computational model for non-linear function approximation, probability density estimation and classification [10]. This model has been applied in a wide range of problems like classification, forecasting and regression. The CGN comprises two transfer functions (sigmoid and gaussian) connected and integrated with the output neuron, few synaptic weights, and two aggregation functions (sum and product), see Fig. 1. The number of synaptic weights is less than a MLP.



**Fig. 1.** CGN structure.

To compute the output of the CGN, the input patterns,  $\mathbf{x} \in \mathbb{R}^N$  are mapped with the corresponding synaptic weights  $\mathbf{w} \in \mathbb{R}^N$  of each neuron ( $O_\Sigma$  and  $O_\Pi$ ) using the corresponding aggregation functions (sum and product), then the result are evaluated by the transfer function (sigmoidal for sum function and gaussian for product function). Finally, the output neuron integrates the obtained results of the first two neurons. In addition, another input called *bias* is considered.

The output neural using *sum* aggregation function is represented by Eq. 1.

$$O_{\Sigma} = f_1(S_{CGN}) = \frac{1}{1 + \exp(-\lambda_s \cdot S_{CGN})} \quad (1)$$

In this case,  $S_{CGN} = \sum W_{\Sigma_i} X_i + X_{0\Sigma}$ , where  $\lambda_s$  is the gain factor,  $W_{\Sigma}$  represent the input weights and  $X_{0\Sigma}$  is the sum-bias.

The output neuron using *product* aggregation function is represented by Eq. 2.

$$O_{\Pi} = f_2(P_{CGN}) = \exp(-\lambda_p \cdot (P_{CGN})^2) \quad (2)$$

Also, for this case,  $P_{CGN} = \prod W_{\Pi_i} X_i \cdot X_{0\Pi}$ , where  $\lambda_p$  also is the gain factor,  $W_{\Pi}$  represent the input weights and  $X_{0\Pi}$  is the product-bias.

Finally, the total output of the CGN is represented by Eq. 3.

$$O_{CGN} = W \cdot O_{\Sigma} + (1 - W) \cdot O_{\Pi} \quad (3)$$

In order to propose a more simple generalized neuron model, we analyze the main parts of a perceptron (Eq. 4), a morphological perceptron and (Eq. 5) and some terms of a polynomial neural network (Eq. 6).

$$y_{cp} = f([\sum_{i=1}^n x_i \cdot w_i] + b) \quad (4)$$

$$y_{mp} = f([\vee_{i=1}^n x_i + w_i] + b) \quad (5)$$

$$y_{pp} = f([\sum_{i=1}^n w_i x_i^2 + w_i x_i] + b) \quad (6)$$

As can be observed, these expressions contain common elements in the structure of the neural model: the transfer function ( $f$ ), the aggregation operator ( $\Sigma$  and  $\vee$ ) and the integration operator ( $\oint$ ,  $+$  and  $\cdot$ ). In that sense, we could propose a generalized neuron model in terms of this three main parts as described in Eq. 7:  $T_f$  (transfer function),  $A_o$  (aggregation operator) and  $I_o$  (integration operator).

$$y_{gnm} = T_f([A_o \sum_{i=1}^n I_o(x_i, w_i)] + b) \quad (7)$$

where the transfer function can be any function selected from the set of transfer functions such as

$$T_f = \{tansig, logsig, hardlim, radbas, purelin\} \quad (8)$$

the aggregation function can be selected from the set of the aggregation functions such as summation, multiplication, maximum and minimum

$$A_o = \{\sum, \prod, \vee, \wedge\} \quad (9)$$

and the integration operator that allow the interaction between input pattern ( $x$ ) and synaptic weights ( $w$ ) can be selected from a set of three main operators

$$I_o = \{\oplus, \odot, \otimes\} \quad (10)$$

defined as

$$\oplus (x, w) = x + w \quad (11)$$

$$\odot (x, w) = x \cdot w \quad (12)$$

$$\otimes (x, w) = wx^2 + wx \quad (13)$$

### 3 Methodology for DNA Classification

The methodology presented in this article, for performing a DNA classification task, is based on the method described in [8]. The methodology is divided in two main steps: one devoted to select the set of genes that best describe the DNA microarray, and other focused on training the new generalized neuron for improving the accuracy of the classification task.

Both phases apply different bio-inspired algorithms for achieving better results. Bio-inspired algorithm are based on biological processes of some species looking for their survival such as the food search task or natural selection process [15]. These kind of algorithms try to find a set of possible solutions distributed in a search space that optimize a fitness function. These solutions change in terms of different operators that help to improve the solution of the problem. The solutions are evaluated using a fitness function until the best solution is found.

The first step performs a dimensionality reduction of the DNA microarray, reducing the number of features and selecting the set of genes that best represent a specific disease. In order to achieve that, the artificial bee colony (ABC) algorithm is applied in combination with the Euclidean distance classifier.

Once selected the set of genes that best describe the disease, the second step uses this information for training an artificial neural network (ANN) or any classification device. Any type of ANN could be applied after reducing the dimension of the DNA microarray. However, in order to avoid the complex task of designing an ANN, we decide to use a simple generalized neural network (GNN) (composed of two neurons) trained with the differential evolution (DE) algorithm as described in [14].

A deep description of the two steps are described in the next subsections.

#### 3.1 Dimensionality Reduction

According to [8], the selection of the best set of genes could be defined in terms of an optimization problem. Given a set of  $p$  DNA microarrays  $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_p\}$ ,  $\mathbf{x}_i \in \mathbb{R}^n$ ,  $i = 1, \dots, p$  and its corresponding disease  $\mathbf{d} = \{d_1, \dots, d_p\}$  associated to each DNA microarray. The number of disease is defined by  $d_i \in \{1, \dots, K\}$  and  $K$ . The aim is to find a subset of genes of the DNA microarray data  $G \in \{0, 1\}^n$  such that a fitness function defined by  $\min(F(\mathbf{X}|_G, \mathbf{d}))$  is minimized.

The dimensionality reduction solution is represented with a subset of genes defined by an array  $I \in \mathbb{IR}^n$ . Each individual  $I_q, q = 1, \dots, NB$  is binarized using the Eq. 14 with a threshold level  $th$ . This threshold select the best set of genes defined as  $G^k = T_{th}(I^k), k = 1, \dots, n$ ; whose component is set to 1, indicates that this gene will be selected to make up the subset of genes.

$$T_{th}(x) = \begin{cases} 0, & x < th \\ 1, & x \geq th \end{cases}. \quad (14)$$

The artificial bee colony (ABC) algorithm is based on the metaphor of the bees foraging behavior [16]. The population of  $NB$  bees  $\mathbf{x}_i \in \mathbb{IR}^n, i = 1, \dots, NB$  represented by the position of the food sources (possible solutions) are distributed in a search space. Three classes of bees are used to achieve the convergence near to the optimal solution that represents the most relevant genes from the DNA microarray.

To evaluate the solutions found by the ABC algorithm and determine which is the best solution, it is necessary to define a fitness function. The aptitude of an individual is calculated with a fitness function that measures how many samples have been wrongly predicted in terms of the classification error function (CER). This fitness function is defined in Eq. 15.

$$F(\mathbf{X}|_G, \mathbf{d}) = \frac{\sum_{i=1}^p \left( \left| \arg \min_{k=1}^K (D(\mathbf{x}_i|_G, \mathbf{c}^k|_G)) - d_i \right| \right)}{p} \quad (15)$$

where  $p$  is the total number of gene expressions to be classified,  $D$  is a distance measure,  $K$  is the number of classes,  $\mathbf{c}$  is the center of each category,  $\arg \min$  provides the class to which the input pattern belongs in terms of the distance classifier and  $d_i$  is the expected class.

In [8], the authors comment that different distance measures could be applied to classify the gene expression samples. In this research, we adopt the Euclidean distance.

### 3.2 Pattern Classification Using the New Generalized Neuron Model

Once selected the set of features that best describes the disease, the next step is to train the generalized neural network (GNN) composed of two generalized neurons, one neuron for each class defined in the problem. For determining the class to which the input pattern belongs, the generalized neurons enter into a competition stage where the neuron with the highest output is set to 1 and the remaining neurons are set to 0; the neuron set with 1 determines the class to which the input pattern belongs.

The inputs of the GNN is feed with the genes previously selected using the ABC algorithm. Before starting to train the GNN, the dataset with the best genes was partitioned into two datasets: training and testing subsets. After that, the GNN was trained with the differential evolution (DE) algorithm.

The differential evolution (DE) algorithm is a optimization technique proposed in [17]. It has few parameters (CR (crossover rate) and F (mutation rate)) and it converges to the optimum faster than others evolutionary techniques. The solution population is represented by vectors of real numbers.

The solutions (individuals), generated with DE algorithm, codify the GNN structure in terms of the synaptic weights ( $w$ ), bias ( $b$ ) as well as the transfer function ( $T_f$ ), aggregation operator ( $A_o$ ) and integration operator ( $I_o$ ) defined in previous section. Particularly, in this paper the set of transfer function was composed of the *tansig*, *logsig*, *hardlim*, *hardlims*, *netinv*, *poslin*, *radbas*, *satlin*, *satlins*, *tribas*, *purelin* functions. Each solution was evaluated with the fitness function defined in Eq. 16.

$$F(\mathbf{X}|_G, \mathbf{d}) = \frac{\sum_{i=1}^p (|GNN(\mathbf{x}_i|_G) - d_i|)}{p} \quad (16)$$

where  $p$  is the total number of gene expressions to be classified and  $GNN$  is the output of the generalized neuron.

Once trained the GNN, we proceed to evaluate its generalization capabilities using the testing subset. To measure the accuracy of the GNN, we computed the classification performance also by means of Eq. 16.

## 4 Experimental Results

In this section, we analyze the experimental results obtained with the proposed methodology to determine its accuracy. For that purpose, the methodology was applied to classify two type of cancer: the acute lymphocytic leukemia and the acute myeloid leukemia.

In [18], the authors demonstrate successful classification between ALL and AML leukemia using DNA microarrays. The *Leukemia benchmark ALL-AML* dataset contains measurements corresponding to ALL and AML samples from Bone Marrow and Peripheral Blood. It is composed of samples for training (27 ALL and 11 AML) and 34 samples for testing (20 ALL and 14 AML) where each sample contains information of 7129 gene expressions.

For selecting the best set of genes, we adopt the protocol described in [8], performing 30 experiments for different values of  $th$  and using the next parameters for the ABC algorithm: population size ( $NB = 40$ ), maximum number of cycles  $MNC = 2000$ , limit  $l = 100$  and food sources  $NB/2$ .

After concluding the experimental results, we observed that the best result were obtained with a threshold set to  $th = 0.3$ . By using this parameters, the ABC algorithm select three genes for performing the classification task, achieving an average accuracy of 74.6% during the testing stage, see Table 1. According to the experimental results obtained during the dimensionality reduction stage, the best genes found by the methodology were SLC17A2 Solute carrier family 17 (L13258\_at), MLC gene (M22919\_rna2\_at) and FBN2 Fibrillin 2 (U03272\_at), achieving a maximum accuracy of 88.2% with the samples for testing.

**Table 1.** Best behavior of the proposed methodology using a Euclidean distance classifier for ALL-AML dataset.

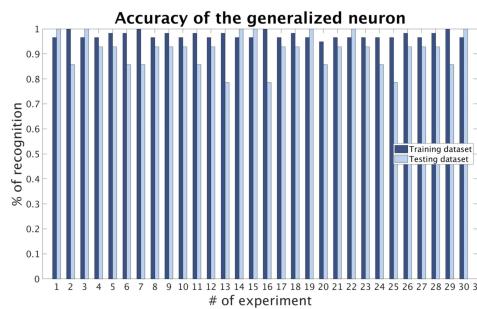
th	Average accuracy		Ave. # of genes	Ave. # of iter.	Best accuracy		# of genes	# of iter.
	Tr. cl.	Te. cl.			Tr. cl.	Te. cl.		
0.3	$0.992 \pm 0.01$	$0.746 \pm 0.07$	1506.7	724.7	1.000	0.882	3	253

Tr. cl. = Training classification rate, Te. cl. = Testing classification rate.

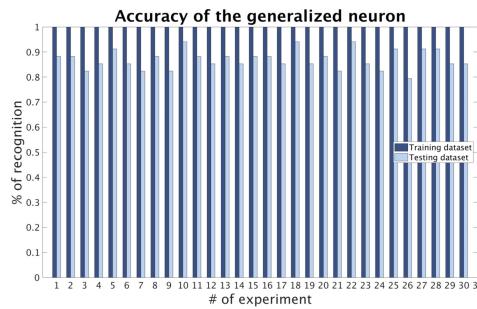
Once the best set of genes was found, the information was used to train a generalized neural network (GNN) composed of two generalized neurons. By using these three genes combined with the methodology for training the GNN, we expect to improve the results achieved with the Euclidean distance classifier. For training the GNN, we set the parameters of differential evolution (DE) algorithm as follows: crossover  $CR = 0.9$ , mutation rate  $F = 0.8$ , number of population members  $NP = 50$  and 1000 generations. In order to statistically validate and compare the results obtained with the designed ANN, 30 experiments were performed. In each experiment, the dataset was partitioned following two different criteria. Original partition (O) where data is partitioned as in the original dataset. Random partition (R) where 80% of the samples were selected randomly from the original dataset to construct the training dataset and the remaining for the testing dataset.

In addition, the experimental results obtained with the GNN were compared against the results obtained with a classical generalized neuron (CGN) following the methodology described in [14] and a feed forward neural network (MLP) with the next architecture: input layer with three linear neurons, hidden layer with five sigmoidal neurons and output layer with two sigmoidal neurons. For determining the class to which the input pattern belongs, the neurons of the output layer enter into a competition stage where the neuron with the highest output is set to 1 and the remaining neurons are set to 0; the neuron set with 1 determines the class to which the input pattern belongs. The proposed MLP was trained according to the next parameters: the learning rate was set to 0.1. The number of epoch for the training phase was set to 5000, and the goal error was set to 0. The Levenberg-Marquardt algorithm was used during the training phase of the MLP (Figs. 2 and 3).

Table 2 shows the results for Leukemia problem using the proposed MLP, the CGN and the proposed GNN. From this results, we can observed that the MLP, CGN as well as the GNN provides better results compared to those obtained with the euclidean distance classifier. In average, the CGN and GNN provides better results than the MLP, achieving an accuracy of 93.3% and 91.4%, respectively, during the testing phase with the random partition. Although, the is slightly decreasing in the accuracy using the GNN, for the case of the original partition the GNN provides the best results (87.1%). The best average accuracy provided by the proposed methodology was of 100% using the random partitions. On



**Fig. 2.** Experimental results using the original partition.



**Fig. 3.** Experimental results using random partitions.

the other hand, the best accuracy obtained with the proposed methodology was 94.1%, using the original partition.

Finally, in Table 3, we compare the results against those obtained by other authors using a support vector machine (SVM). In general, we observed that the proposed methodology performs better than those revised from the literature.

These experimental results suggest that the GNN could be adopted as an alternative technique for performing the classification of DNA microarrays. Compared against the CGN and MLP, the GNN provide similar results but it is

**Table 2.** Best and worst accuracy obtained with the GN and MLP.

P	Type NN	Average accuracy		Best accuracy		Worst accuracy		# of genes
		Tr. cl.	Te. cl.	Tr. cl.	Te. cl.	Tr. cl.	Te. cl.	
O	MLP	$0.819 \pm 0.213$	$0.740 \pm 0.170$	1.000	0.941	0.289	0.412	3
R	MLP	$0.844 \pm 0.185$	$0.833 \pm 0.182$	0.983	1.000	0.328	0.357	3
O	CGN	$1.000 \pm 0.000$	$0.868 \pm 0.020$	1.000	0.941	1.000	0.853	3
R	CGN	$0.970 \pm 0.008$	$0.933 \pm 0.065$	0.983	1.000	0.966	0.786	3
O	GNN	$1.000 \pm 0.000$	$0.871 \pm 0.038$	1.000	0.941	1.000	0.794	3
R	GNN	$0.974 \pm 0.013$	$0.914 \pm 0.066$	1.000	1.000	0.948	0.786	3

Tr. cl. = Training classification rate, Te. cl. = Testing classification rate.

**Table 3.** Comparison against other techniques

Classification technique	Selection technique	# of genes	% of accuracy	References
GNN	ABC	3	1.0000	Proposed methodology
CGN	ABC	3	1.0000	[14]
MLP	ABC	3	1.0000	[8]
SVM	PSO	10	1.0000	[19]
SVM	PLS-RFE	16	1.0000	[20]
SVM	GBC	3	0.9583	[21]

important to mention that less neurons are required compared against MLP. On the other hand, the new generalized neuron model is less complex than the classical generalized neuron.

## 5 Conclusions

During the first stage, a dimensionality reduction over the ALL-AML dataset was successfully applied in order to select the set of genes that best describe the leukemia cancer using the ABC algorithm. Once discover the best set of genes, we evaluated the accuracy of a simple distance classifier, obtaining an accuracy highly acceptable. Nonetheless, in the second stage, we tried to improve the results using a generalized neural network (GNN).

In the second stage, we evaluated the performance of the GNN during the detection of the two type of leukemia cancer. The GNN was trained using the set of genes discovered by the proposed methodology during the first stage. The obtained results, show that the differential evolution algorithm is an excellent technique for training a GNN. The accuracy achieved with the GNN was compared against the accuracy of a feedforward neural network (FNN) as well as the classical generalized neuron (CGN). Through several experiments, we observed that the GNN as well as the CGN and FNN obtained better results compared to those reached with the distance classifier.

On the other hand, the experimental results showed that GNN achieved a better performance than the results obtained with the FNN and similar to those obtained with the CGN. Finally, we concluded that the GNN trained with the proposed methodology is capable of detecting, predicting and classifying a disease with an acceptable accuracy.

**Acknowledgment.** The authors would like to thank Universidad La Salle México for the economic support under grant number NEC-10/18.

## References

1. Peterson, L., et al.: Artificial neural network analysis of DNA microarray-based prostate cancer recurrence. In: 2005 Proceedings of the 2005 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, CIBCB 2005, pp. 1–8, November 2005
2. Huynh, H.T., Kim, J.J., Won, Y.: Classification study on DNA micro array with feed forward neural network trained by singular value decomposition. *Int. J. Bio-Sci. Bio-Technol.* **1**, 17–24 (2009)
3. Khan, J., et al.: Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nat. Med.* **7**(6), 673–679 (2001)
4. Lancashire, L.J., Lemetre, C., Ball, G.R.: An introduction to artificial neural networks in bioinformatics application to complex microarray and mass spectrometry datasets in cancer studies. *Briefings Bioinform.* **10**(3), 315–329 (2009)
5. Chen, W., Lu, H., Wang, M., Fang, C.: Gene expression data classification using artificial neural network ensembles based on samples filtering. In: 2009 International Conference on Artificial Intelligence and Computational Intelligence, AICI 2009, vol. 1, pp. 626–628, November 2009
6. Peterson, L.E., Coleman, M.A.: Comparison of gene identification based on artificial neural network pre-processing with k-means cluster and principal component analysis. In: Bloch, I., Petrosino, A., Tettamanzi, A.G.B. (eds.) WILF 2005. LNCS (LNAI), vol. 3849, pp. 267–276. Springer, Heidelberg (2006). [https://doi.org/10.1007/11676935\\_33](https://doi.org/10.1007/11676935_33)
7. Garro, B.A., Vazquez, R.A., Rodríguez, K.: Classification of DNA microarrays using artificial bee colony (ABC) algorithm. In: Tan, Y., Shi, Y., Coello, C.A.C. (eds.) ICSI 2014. LNCS, vol. 8794, pp. 207–214. Springer, Cham (2014). [https://doi.org/10.1007/978-3-319-11857-4\\_24](https://doi.org/10.1007/978-3-319-11857-4_24)
8. Garro, B.A., Rodríguez, K., Vázquez, R.A.: Classification of DNA microarrays using artificial neural networks and ABC algorithm. *Appl. Soft Comput.* **38**, 548–560 (2016)
9. Garro, B.A., Vázquez, R.A.: Designing artificial neural networks using particle swarm optimization algorithms. *Comp. Int. and Neurosc.* **2015**, 369298:1–369298:20 (2015)
10. Kulkarni, R.V., Venayagamoorthy, G.K.: Generalized neuron: feedforward and recurrent architectures. *Neural Netw.* **22**(7), 1011–1017 (2009)
11. Rizwan, M., Jamil, M., Kothari, D.: Generalized neural network approach for global solar energy estimation in India. *IEEE Trans. Sustain. Energy* **3**(3), 576–584 (2012)
12. Kiran, R., Venayagamoorthy, G.K., Palaniswami, M.: Density estimation using a generalized neuron. In: 9th International Conference on Information Fusion, FUSION 2006, Florence, Italy, 10–13 July 2006, pp. 1–7. IEEE (2006)
13. Kiran, R., Jetti, S.R., Venayagamoorthy, G.K.: Online training of a generalized neuron with particle swarm optimization. In: Proceedings of the International Joint Conference on Neural Networks, IJCNN 2006, Part of the IEEE World Congress on Computational Intelligence, WCCI 2006, Vancouver, BC, Canada, 16–21 July 2006, pp. 5088–5095. IEEE (2006)
14. Garro, B.A., Rodríguez, K., Vázquez, R.A.: Generalized neurons and its application in DNA microarray classification. In: IEEE Congress on Evolutionary Computation, CEC 2016, Vancouver, BC, Canada, 24–29 July 2016, pp. 3110–3115. IEEE (2016)

15. George, G., Raimond, K.: Article: a survey on optimization algorithms for optimizing the numerical functions. *Int. J. Comput. Appl.* **61**(6), 41–46 (2013). Full text available
16. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical report, Computer Engineering Department, Engineering Faculty, Erciyes University (2005)
17. Storn, R., Price, K.: Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical report (1995)
18. Golub, T.R., et al.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science* **286**(5439), 531–537 (1999)
19. Sahu, B., Mishra, D.: A novel feature selection algorithm using particle swarm optimization for cancer microarray data. *Proc. Eng.* **38**(0), 27–31 (2012). International Conference on Modelling Optimization and Computing
20. Wang, A., An, N., Chen, G., Li, L., Alterovitz, G.: Improving PLSRFE based gene selection for microarray data classification. *Comput. Biol. Med.* **62**, 14–24 (2015)
21. Alshamlan, H.M., Badr, G.H., Alohal, Y.A.: Genetic bee colony (GBC) algorithm: a new gene selection method for microarray cancer classification. *Comput. Biol. Chem.* **56**, 49–60 (2015)

## **Classification - Learning**



# A Hybrid Approach for the Fighting Game AI Challenge: Balancing Case Analysis and Monte Carlo Tree Search for the Ultimate Performance in Unknown Environment

Lam Gia Thuan<sup>1</sup>(✉), Doina Logofătu<sup>2</sup>(✉), and Costin Badică<sup>3</sup>(✉)

<sup>1</sup> Faculty of Engineering, Vietnamese-Germany University,  
Le Lai, Thu Dau Mot, Vietnam

cs2014\_thuan.1g@student.vgu.edu.vn

<sup>2</sup> Department of Computer Science and Engineering,  
Frankfurt University of Applied Sciences, Nibelungenpl. 1,  
60318 Frankfurt, Germany

logofatu@fb2.fra-uas.de

<sup>3</sup> Department of Computer Sciences and Information Technology,  
University of Craiova, 200285 Craiova, Romania  
cbadica@software.ucv.ro

**Abstract.** The challenging nature of the Fighting Game AI Challenge originates from the short instant of response time which is a typical requirement in real-time fighting games. Handling such real-time constraint requires either tremendous computing power or a clever algorithm design. The former is uncontrollable by the participants, as for the latter, the competition has received a variety of submissions, ranging from the naivest case analysis approach to those using highly advanced computing techniques such as Genetic Algorithms (GA), Reinforcement Learning (RL) or Monte Carlo Tree Search (MCTS), but none could provide a stable solution, especially in the LUD division, where the environment setting is unknown in advance. Our study presents our submission to this challenge in which we designed a winning solution in the LUD division which, for the first time, stably outperformed all players in all competition categories. Our results demonstrate that a proper blend of case analysis and advanced algorithms could result in an ultimate performance.

**Keywords:** Fighting Game AI Challenge · Real-time fighting game · LUD division · Case analysis · MCTS

## 1 Introduction

Advances in computing have given rise to computer game complexity, to the extent that human ability is no longer sufficient to handle the vast number of game states. Human gamers will soon be dominated by computer programs due to the emergence of powerful techniques such as Monte Carlo Tree Search [1] and Machine Learning [2] in which the programs can learn the solution with little human intervention. Aided by modern computer power, these algorithms have demonstrated a capacity beyond that of

human experts with numerous examples such as Deep Blue [3] or AlphaGo [4], but does that imply that human guidance is completely irrelevant in this day and age?

The success of programs like AlphaGo was partially thanks to the nature of the game, which was chess, a turn-based game, in which modern computers are given sufficient time to process. However, in a real-time environment where programs must respond reasonably well within a short instance of time, current processing power proves insufficient since modern game states can grow exponentially while processors are increasingly closer to their physical limit [5].

This paper presents our submission to the Fighting Game AI challenge [6] 2018, a challenging AI competition that aims at solving the general real-time fighting problem. Our research focuses on the design of an algorithm that can handle the vast number of game states in a reasonable way given our average computational power. Our algorithm is a proper blend between a generic case analysis approach with the winning MCTS algorithm. Our results show that by blending some human wisdom with MCTS, the resulting performance is superior to all in existence.

## 2 The Fighting Game AI Challenge

### 2.1 Overview

The Fighting Game AI Challenge is initiated by the Intelligent Computer Entertainment Lab of the Ritsumeikan University to promote AI research towards general fighting games, a classical class of games, in which players compete against each other until only one remains using techniques resembling those in martial arts. Starting since 2013, the competition has been well-received by scholars around the world and achieved a high reputation among the most prestigious academic conferences worldwide, including the IEEE Conference on Computational Intelligence and Games (CIG). CIG 2018 marked another successful milestone of the challenge with the emergence of numerous interesting solutions, among which was our submission, the first that could perform well and stably in the most challenging LUD division in all competition categories.

### 2.2 Organization and Rules

The competition consists of 3 divisions – ZEN, GARNET and LUD, each is further subdivided into 2 categories: STANDARD (participants fighting each other) and SPEED-RUNNING (participants defeating the organizer's program in the shortest amount of time). Each player is given a set of actions, from which one must be selected within a fixed unit of time, namely frame. The player with the most HP remained after at most 3600 frames is the winner of the game. In the STANDARD category, the player that wins the most games, is the final winner, while in SPEED-RUNNING, the final winner is the player that wins against a common opponent in the least number of frames.

### 2.3 Our Focus - The LUD Division

The LUD division distinguishes itself from ZEN and GARNET by hiding all action data in advance. An action is characterized by a number of parameters, a subset of which can be shown in Table 1. These parameters are vital for memorized methods such as Q-Learning or if-else approach, in which participants can train their program for a long time before the contest and store learned experiences into files that will be retrieved during the playtime, which is a realistic approach for the first 2 divisions in that they have all information published prior to the competition. Whilst for the LUD division, such essential information is only provided at the start of the game as a step to make it closer to the general fighting problem, which, at the same time, prevents memorized approaches from emerging victorious. That is where self-learning methods such as MCTS reign considering that they have no need for prior training.

**Table 1.** A sample subset of action parameters.

Parameter name	Description
Frame number	The number of frames required to perform
Horizontal speed	The horizontal velocity
Vertical speed	The vertical velocity
Hit area	The impact area as a rectangle
State	The state of opponent after getting hit

MCTS is, unfortunately, not without its limitations. No prior training implies the need for self-learning during the playtime which is not favorable in a real-time environment. Albeit the majority of the submissions were dominated by MCTS, only those who could manage to reduce its time-consuming nature could become victors. The most successful example being Eita Aoki [7]. By discovering a winning heuristic, he reduces the use of MCTS to the minimum possible and became the 3-time consecutive winner of this challenge. Nonetheless, even his solution is unable to conquer the LUD division since his heuristic could not be formalized without prior information.

Our submission proposed the first stable winning solution to the challenging LUD division. Albeit missing a detailed case analysis made us stay only in the second place in the first 2 divisions, our contribution is still of paramount importance since LUD is, among the three, the closest division to the general real-time fighting problem.

## 3 Previous Work

The first period between 2013 and 2015 could be regarded as the Dark Ages in the history of the competition in that submissions were entirely populated with variants of the if-else approach with little to no sign of new directions. Regardless, there were numerous interesting heuristics, as shown in Table 2, which are still relevant until now. The initial popularity of these if-else variants is understandable since advanced techniques require a certain level of design and implementation skill and algorithm design

is too hard of a field even for experienced experts, making them unpopular among participants, most of whom are students or young professionals. Case analysis with heuristics is much easier to implement and hence was the only technique in use in the first year of the competition. Subsequent years saw more advanced techniques such as Reinforcement Learning or Fuzzy Logic, but their implementation was still too simplistic to produce satisfiable results. Worse came to worst, they were even losing against the naïve and much-simpler-to-implement if-else approaches.

**Table 2.** Competition notable approaches in 2013–2015 period.

Year	Approach
2013	Position Analysis + Random Limited Choices
	Position Analysis + Fixed Choices + Defending/Escaping Heuristic
2014	Simple Reinforcement Learning for memorizing states
	Case Analysis + Opponent Modelling
	Simple Fuzzy Logic
2015	Position Analysis + Random Limited Choices + Runaway Case
	Simple Machine Learning for memorizing states

The initial popularity of these if-else variants is understandable since advanced techniques require a certain level of design and implementation skill and algorithm design is too hard of a field even for experienced experts, making them unpopular among participants, most of whom are students or young professionals. Case analysis with heuristics is much easier to implement and hence was the only technique in use in the first year of the competition. Subsequent years saw more advanced techniques such as Reinforcement Learning or Fuzzy Logic, but their implementation was still too simplistic to produce satisfiable results. Worse came to worst, they were even losing against the naïve and much-simpler-to-implement if-else approaches.

In 2016, the organizers introduced MCTS into the competition as a sample, which marked a new standard for submissions and resulted in the emergence of more sophisticated solutions. Some of the most successful participants can be mentioned as follows:

- Eita Aoki for combining his winning unbreakable corner heuristics with MCTS which overwhelmingly dominated the first 2 divisions: LUD and GARNET.
- Youssouf Ismail Cherifi for combining his consecutive strike heuristics with MCTS.
- Man-Je Kim and Kyung-Joong Kim for the first evolutionary algorithm solution in combination with MCTS [8].

In addition, many submissions reimplemented algorithms that had been seen in previous years, but with much more mature implementation. Albeit they are unable to compete with MCTS and evolutionary algorithms, they have demonstrated their true worth by at least outperforming simplistic case analysis approaches.

Despite all these advances, the LUD division remained unmanageable until our participation. In 2018, we proposed a solution that outperformed all others in LUD divisions in all categories and became the first stable winning solution to this division.

## 4 The Winning Solution to the LUD Division

Careful analysis of the previous solutions leads us to the conclusion that a successful solution is the one that is based on MCTS but does not abuse it, as illustrated in the following pseudocode:

```
Action findBestAction (GameState state) {
    Action action = intelligentSearch(state)
    if (action != null)
        action = MCTS(state)
    return action
}
```

In detail, a successful solution should implement an intelligent search method in addition to MCTS and prioritize it whenever possible.

For the first 2 divisions, ZEN and GARNET, the distinction between submitted MCTS-based solutions usually lies in the former and almost all participants employed the standard implementation of MCTS provided by the organizer. Unfortunately, none could provide a similar solution for the LUD division, since the lack of data prevented participants from formalizing a working heuristic. Our study concludes that such a model is still applicable to the LUD division as long as the heuristic is made *generic*.

### 4.1 Adaptive Intelligent Search

This is the winning aspect of our solution but also the simplest one in that our analysis is truly generic - almost independent of actual values, which is why it works well in this unknown context. The search for the best action in our solution depends on the following factors: potential, preselection, and upper distance analysis.

**Potential.** To analyze each action's potential, it is of paramount importance to analyze parameters outside those mentioned in Table 1, including the startup time and the resulting state of the player under attack. The latter is more prioritized than the former in our submission since startup time does not vary much among different actions.

*Startup Time.* The complete process for performing an attack consists of multiple periods, including a startup, active, recovery and canceling period, as visualized in Fig. 1.

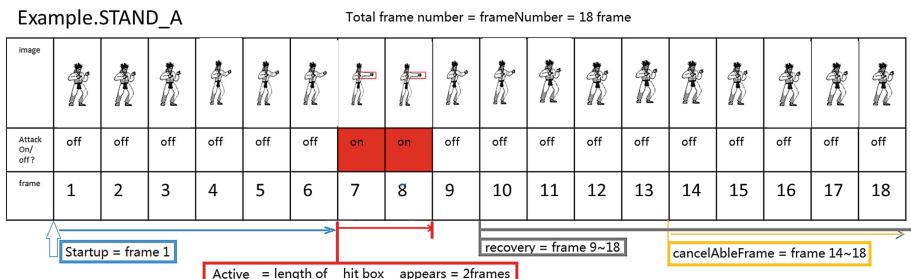
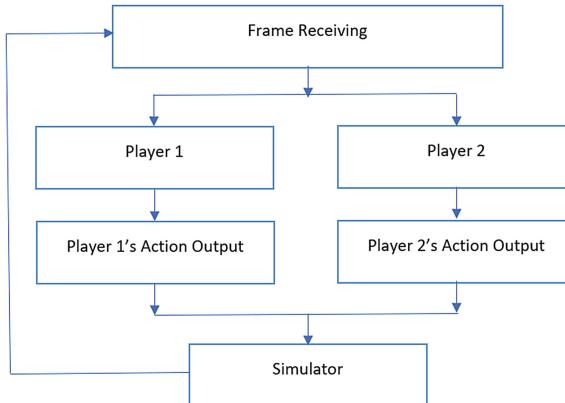


Fig. 1. Motion illustration (The Fighting Game AI Challenge).

Among which, we believe that startup is the most imperative period. The rationale behind this decision is due to the concurrent nature of the problem as illustrated in Fig. 2.



**Fig. 2.** Game processing flow.

As illustrated in Fig. 2, players perform their respective action at the same time, implying the possibility that one player can be put under attack even before he or she is ready to act. Thus, the faster the attack can start, the less likely our player will end up in a disadvantageous situation, which was why this factor should be considered as a priority to determine the potential of the action.

*Resulting State of the Player Under Attack.* Nonetheless, fast startup alone does not guarantee an excellent action since the opponent still has a chance to revenge in the next attack. A promising attack should be one that opens more opportunity for future attacks. In further details, it means that a good attack should lead the opponent into a vulnerable state in which it is difficult or even impossible for him or her to block the next one, forming a series of consecutive moves, namely combo [9]. In this particular challenge, the most known vulnerable state is the DOWN state – a state where the opponent is completely knocked out and became unable to retaliate. Hence actions that can result in the opponent’s DOWN state will be our first-class priority.

**Preselection.** A further improvement is the preselection of a minimum set of actions based on their potential for speed improvement during the playtime. After that, an additional reselection to further minimize the selected list will be conducted based on random game simulation and decide which actions to choose according to its respective total damage for the opponent during the random games. An illustration can be shown as follows.

```

List<Action> preselection(Int round1Size, Int round2Size)
{
    List<Action> firstRound = getAllActions()
        .sortedAscBy (action -> action.getPotential())
        .toList()
        .subList(0, round1Size)

    Map<Action, Int> damageByAction =
        simulateRandomGamesAndRecord(firstRound)

    List<Action> secondRound = firstRound
        .sortedDescBy (action -> damageByAction[action])
        .toList()

    return secondRound
}

```

**Upper Distance Analysis.** Attacking is important, but another indispensable aspect of victory in a fighting game is the control of player movement, which usually replies on actual values. In order to make it generic to work in the LUD division, we define a much simpler strategy: attack or defend. In attack mode, our player will choose actions that get us as close to the opponent as possible. In defend mode, we will just choose actions that run away from the opponent. The threshold to determine whether we should attack or defend depends solely on the difference between our current HP and the opponent HP as illustrated in the pseudocode below. This will be executed before all to filter irrelevant actions before selection.

```

List<Action> upperDistanceAnalysis() {
    if (myHP - opponentHP >= UPPER_LIMIT) {
        return defendActions() // actions that runs away
    } else {
        return attackActions() // actions that get us closer
    }
}

```

The threshold is set to be an upper bound value, which can be *any high enough* value that if it is met, we are certainly going to win regardless of what happens afterward. Since such value is independent of the action data, it can be chosen generically.

In addition to the above factors, each action will only be considered for selection if the current amount of energy permits it. Actions that require too much energy will be filtered beforehand. Last, but not least, a simple minimax algorithm will be used to select those with the same priority and each action will only be selected if it can produce a positive impact on the opponent. The priority to select an action will be first, in those chosen in the preselection, followed by those with the largest potential.

## 4.2 A Brief Overview of Our Optimized MCTS

As stated in Subsect. 4.1, the core of our contribution is the intelligent search, not the MCTS since almost all MCTS-based solutions were just reusing the provided sample MCTS code with little to no improvement. Moreover, the sample provided implementation is a very standard one, hence an extensive description will not be included in our paper. Further information can be found at [10]. Instead, in this section, we will just briefly review the key points of MCTS used in our program and briefly introduce our optimization.

**Monte Carlo Tree Search.** MCTS is a simulation-based search algorithm. Like other algorithms, first it will search for the known *best* node, then it will extend the search tree from there. The best node is chosen in a way that balances both the depth (exploitation) of the search tree and breadth (exploration) of opportunities and the way to achieve that is to use the famous Upper Confidence Bound (UCB) formula. The full form of UCB used for computing the potential of each node in our solution is as follows.

$$\overline{W} + c \times \sqrt{\frac{\log_2 N_p}{N_c}}, \quad (1)$$

in which  $\overline{W}$  is the average number of wins,  $N_p$  is the number of visits in the direct parent node and  $N_c$  is the number of visits in the current node.  $c = \sqrt{2}$  is the exploration parameter that is used to balances between exploitation ( $\overline{W}$ ) and exploration ( $\sqrt{\frac{\log_2 N_p}{N_c}}$ ).

In the context of our solution, each node is a game state. The root node is the current game state and all others are virtual nodes that will be created by game simulation. Each new node is created from the current node when we try to perform a different action. The tree branch will go deep until we can decide if the current state is a win or loss and then we can recursively recompute all information such as the number of wins and the number of visits from the current state up to root. The entire process is repeated until time runs out (1 frame =  $\frac{1}{60}$  s). The selected action is one that leads to the most visited direct child from root. The complete procedure can be summarized as follows.

```

Action MCTS(Node rootNode) {
    Node node = rootNode
    while (node.isNotEndState()) {
        if (node.areAllActionsConsidered()) {
            node = findBestChildNodeByUCB(node)
        } else {
            Action action = node.pickAnyActionNotConsidered()
            node = createNode(node, action)
        }
    }
    while (node != null) {
        updateNumberOfWins(node)
        updateNumberOfVisits(node)
        node = node.getParent()
    }

    return findMostVisitedChild(rootNode).getAction()
}

```

**Optimization.** Our improvement lies in the reduction in memory, which eventually leads to speed improvement as in the case similar to why insertion sort is faster than quick-sort for small lists of elements [11]. Each node in a standard implementation needs to keep track of both the actions that we have considered (simulated) and actions we have not, implying that we may require two arrays for storing them. That approach is used by most people since it is both intuitive and simple to implement. In our submission, we avoid the creation of the additional array by keeping track of only the number of unused actions. Every newly used action will be swapped in  $O(1)$  with the first action that is not yet used. By utilizing this trick, we manage to reduce the memory consumption by half and since the number of possible actions after various filters in Subsect. 4.1 is relatively small, the performance is significantly enhanced. Unfortunately, this is not key to our success due to the choice of our programming language which will be explained in Sect. 5.

## 5 Evaluation

### 5.1 Performance

The first priority to evaluate a solution is naturally its performance. Figure 3 illustrates the improvement of our solution before and after mixing our case analysis approach.

The table on the left of Fig. 3 is our mid-term result against 5 other competitors in which we did not introduce the intelligent search. Albeit we did not have high expectation, it was still shocking that we were at the bottom of ranking in the LUD

	Value	Ranking		Ranking	Value
<b>STANDARD</b>	0	6/6	→	1/9	39
<b>Win Count</b>			→	<b>STANDARD</b>	<b>Win Count</b>
<b>SPEED-RUNNING</b>	70	6/6		1/9	52.72
<b>Time</b>	seconds			<b>SPEED-RUNNING</b>	<b>Time</b>

**Fig. 3.** Performance before and after applying our case analysis approach.

division at the beginning and even worse, we were knocked out by every other participant in every single game. However painful it was, it served well for us as a crucial step to highlight the brilliantness of our improvement as shown in the second table – our final result against 8 other participants, in which we climb straight to the top from the bottom. Being on the top of the ranking in both STANDARD and SPEED-RUNNING leagues demonstrate our performance.

## 5.2 Stability

The next key factor to evaluate our solution is stability which can be illustrated in Table 3, showing instability among other participants.

**Table 3.** Ranking instability.

Solution	STANDARD	SPEED-RUNNING	Ranking change
Our solution	1	1	No change
Thunder	2	5	-3
SampleMctsAi	3	3	No change
MogakuMono	4	2	+2
JayBot_GM	5	4	+1
SimpleAI	6	7	-1
UtalFighter	7	6	+1
MultiHeadAI	8	8	No change
BCB	9	8	+1

As can be seen from Table 3, most participants either perform too unstably or badly. Since SampleMctsAi belongs to the organizers, it is counted, implying that ours is the only stable solution that well-performed (first place for all).

## 5.3 Remarks About MCTS Optimization

As mentioned in the previous section, our solution is not only with one improvement – the intelligent search, but it also includes an optimization in MCTS. This subsection answers the question: Does it improve anything and why is it not key to the success of our solution?

The problem originates from our choice of programming language for implementing this solution - Kotlin [12], a promising programing language on JVM, while

other participants all use Java, a much more mature language. In our submission, we chose Kotlin for its expressiveness and beauty, but it was unexpected that Kotlin compiler is still too young to generate bytecode of the same quality as Java compiler. In our experiment after the contest, we translated the Java programs into equivalent Kotlin code and were surprised to see that the Kotlin version lost every single match despite the algorithm in use is the same. Therefore, even though our MCTS implementation is of higher quality compared to others, it is invisible from the competition results.

Does our optimization really work? In our experiment, we compared MCTS programs with and without the optimization. With our optimization, it was still on the losing side but could handle approximately 1 out of every 3 matches instead of failing every single time as the program without optimization. Regardless, this result demonstrates that without our case analysis, we would have lost also in LUD division and that our intelligent search is truly effective.

## 6 Conclusion

In this study, we have contributed a promising solution for the general fighting problem via our submission to the Fighting Game AI Challenge 2018. Our solution is combination of case analysis and Monte Carlo Tree Search that produces stable and remarkable results in the LUD division, the division closest to the general real-time fighting game. Unfortunately, we were unable to make a significant optimization in the MCTS and still rely on human wisdom to generate the heuristics for the intelligent search which, at the same time, opens a new direction for our future research.

One possible direction for the future is to automate the generation of generic heuristics and the second is to further optimize MCTS for solving this challenge. One promising possibility for the former is the use of Deep Learning and Neural Networks to learn and formalize the heuristics before the competition, but the challenge of such approaches is to construct a model that can ensure the genericity of the resulting heuristics, meaning that it must be independent of actual values. Another approach to handle the latter is to introduce memorization into MCTS as illustrated in the AlphaGo Zero paper [13], which may significantly improve both the quality and performance of the search tree. The downside is that such an approach may be incompatible since the performance of the Google machines running AlphaGo Zero is still largely outperforming that of normal computers. Regardless, these are all promising directions, in which research towards real-time fighting games can evolve.

## References

1. Chaslot, G., Bakkes, S., Szita, I., Spronck, P.: Monte-carlo tree search: a new framework for game AI. In: AIIDE. The AAAI Press (2008)
2. Nork, B., Lengert, G.D., Litschel, R.U., Ahmad, N., Lam, G.T., Logofătu, D.: Machine learning with the pong game: a case study. In: Pimenidis, E., Jayne, C. (eds.) EANN 2018. CCIS, vol. 893, pp. 106–117. Springer, Cham (2018). [https://doi.org/10.1007/978-3-319-98204-5\\_9](https://doi.org/10.1007/978-3-319-98204-5_9)

3. Campbell, M., Hoane, J., Hsu, F.: Deep blue. *Artif. Intell.* **134**(1–2), 57–83 (2002)
4. Silver, D., et al.: Mastering the game of Go with deep neural networks and tree search. *Nature* **529**(7587), 484–489 (2016)
5. Are processors pushing up against the limits of physics? <https://arstechnica.com/science/2014/08/are-processors-pushing-up-against-the-limits-of-physics>. Accessed 21 Feb 2019
6. Fighting Game AI Competition. <http://www.ice.ci.ritsumei.ac.jp/~ftgaic/index-1.html>. Accessed 21 Feb 2019
7. Fighting Game AI Competition. <https://www.slideshare.net/ftgaic/2018-fighting-game-ai-competition?ref=http://www.ice.ci.ritsumei.ac.jp/~ftgaic/index-R.html>. Accessed 21 Feb 2019
8. Kim, M.J., Ahn, C.W.: Hybrid fighting game AI using a genetic algorithm and Monte Carlo tree search. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO 2018, Kyoto, Japan (2018)
9. Zuin, G., Macedo, Y., Chaimowicz, L., Pappa, G.: Discovering combos in fighting games with evolutionary algorithms. In: GECCO 2016, Denver, CO, USA, pp. 277–284 (2016)
10. James, S., Konidaris, G., Rosman, B.: An analysis of monte carlo tree search. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI 2017) (2017)
11. Heineman, G., Pollice, G., Selkow, S.: Algorithms in a Nutshell. O'Reilly Media, Sebastopol (2016)
12. Jangid, M.: Kotlin – the unrivaled android programming language lineage. *Imperial J. Interdisc. Res.* **3**, 256–259 (2017)
13. Silver, D., et al.: Mastering the game of Go without human knowledge. *Nature* **550**, 354 (2017)



# A Probabilistic Graph-Based Method to Improve Recommender System Accuracy

Nima Joorabloo<sup>1</sup>(✉) , Mahdi Jalili<sup>1</sup> , and Yongli Ren<sup>2</sup>

<sup>1</sup> School of Engineering, RMIT University, Melbourne, VIC, Australia  
S3624411@student.rmit.edu.au,  
mahdi.jalili@rmit.edu.au

<sup>2</sup> School of Science, RMIT University, Melbourne, VIC, Australia  
yongli.ren@rmit.edu.au

**Abstract.** The last two decades have seen a surge of data on the Web which causes overwhelming users with huge amount of information. Recommender systems (RSs) help users to efficiently find desirable items among a pool of items. RSs often rely on collaborating filtering (CF), where history of transactions are analyzed in order to recommend items. High accuracy, and low time and implementation complexity are most important factors for evaluating the performance algorithms which current methods have the shortage of all or some of them. In this paper, a probabilistic graph-based recommender system (PGB) is proposed based on graph theory and Markov chain with improved accuracy and low complexity. In the proposed method, selecting each item for recommendation is conditioned by considering recommended items in the previous steps. This approach uses a probabilistic model to consider the items which are likely to be preferred by users in the future. Experimental results performed on two real-world datasets including MovieLens and Jester, demonstrate that the proposed method significantly outperforms several traditional and state-of-the-art recommender systems.

**Keywords:** Recommender system · Collaborative filtering · Markov chain

## 1 Introduction

Today, one of the major problems of the online shops is that users are often confused when deciding what to choose among a huge number of items. RSs have been proposed in order to help users to find the most suitable item according to their preferences [1]. Generally, RSs are classified into content-based methods (CB), collaborative filtering (CF), and hybrid methods. In CB approaches, the system recommends items based on available content information on the users and items [2]. CF is a widely used approach in RSs which focuses on similarity values between the users (or items). CF uses an information filtering technique based on the user's previous rating/purchase history to offer items which are aligned with the taste of the target user [3]. Hybrid RSs combine CF and CB approach to obtain improved performance [4]. The existing research papers in the field of RSs have mainly considered movie recommendation topic [5, 6]. There is also a rich literature on other topics, such as e-commerce [7], books [8], documents [9],

music [10], television programs [11], applications in markets [12], e-learning [13], and Web search [14]. There are various metrics in the literature for evaluating the performance of RS algorithms [15].

Accuracy is one of the most important evaluation metrics, and most of studies in RSs evaluation criteria have focused on the accuracy [16]. Evaluation of RSs can be performed in an offline or online manner [17]. In an offline analysis, part of ratings in the dataset are hidden from the recommender algorithm as a test set and the RS algorithm uses the rest of data (training set) to predict new ratings or rank for unseen items. Offline evaluation methods are fast, but we cannot elicit the real taste of users regarding the recommended items. Likewise, online evaluations are conducted in a live environment to observe users' behavior and tracking their acts [18]. In addition, in comparison with offline methods, conducting an online evaluation, if possible, is more costly and time consuming, and this leads most of the research works to offline evaluation. Over the last decade, many research studies have focused on proposing new approaches to improve performance of recommenders. Although paying attention to existing evaluation measures are important to have a good RS, to implement RS in real world, we have to take into the account some considerations, such as simplicity of implementation and reasonable run time.

In this paper we propose an accurate probabilistic recommendation system based on graph theory to generate accurate recommendations with reasonable run-time in comparison with some traditional and state-of-the-art algorithms. PGB algorithm transforms ratings in the first step to *like* and *dislike*, and by so doing, it helps us to apply ratings on traditional Markov model and makes calculations simpler. Also, PGB lets us to model history of the ratings in a comprehensive and compact graph (system-state graph) using Markov model idea that helps us to recommend items without referring to raw ratings anymore; PGB generates system-state graph only one time at the start of algorithm, and then use it to recommend items. In the next step, we make decision based on system-state graph and travers through it for each user to find most probable items which will be liked by him/her in the future. One of the most significant advantages of PGB is its flexibility against updating dataset. In case of adding new ratings, system-state graph can be modified only by updating respective weights and it doesn't need to make it from scratch; Thus, proposed method is suitable for systems that have a lot of change in a short period of time.

## 2 Related Works

The last two decades witnessed much attention in network science, where graph theory and data mining meet [19]. A number of applications have been proposed to model behaviour of users with graph theory or related theories [20, 21]. Several works have applied Markov models in the context of RSs, where a Markov chain model makes recommendations based on previous actions. Rendle et al. proposed Factorized Personalized Markov Chains (FPMC) to combines Matrix Factorization and Markov Chain to model personalized sequential behavior [22]. His method improved by Cheng et al. by changing factorizing transition matrix into two latent and low-rank sub-matrices [23]. Shani et al. modelled the RS using Markov decision processes

(MDP) [24]. In [25] a context-aware approach to query suggestion were proposed by He et al. Sahoo et al. proposed a new collaborative filtering algorithm based on Hidden Markov Model that outperformed traditional CF techniques, especially when consumers' preferences are changing [26]. A graph-based algorithm was introduced by Yang et al. to first discover the topics of interest for each user, and then make a topic-aware Markov model to learn the navigation patterns for each user [27]. Although Markov model has been used in many real-world applications, it has some restrictions in RS field, namely it is affected by sparsity and neglecting users' ratings on items. Most of the proposed algorithms only consider the sequence of purchase/rating [6]. Ratings can be so important for getting more accurate result. On the other hand, some of Markov-based algorithms consider the probability of transition between states based on users' history. In this work, we propose a method that considers the probability of selecting items by the target user in future.

### 3 Probabilistic Graph-Based Recommendation Method

In this section, we first explain Markov model as the base idea for the proposed method, and then discuss details of the proposed method. The proposed method includes two main steps: (i) transforming the ratings and creating system-state graph using Markov chain to model users' ratings history, and (ii) applying a probabilistic model on the generated graph to recommend items.

#### 3.1 Markov Models

Traditional recommenders like CF constructs the recommendation list based on the preferences of a group of users that are similar to the active user, but Markov chain considers information about ratings' sequence. Let's consider we have a dataset of ratings with a set of users  $U$  and set of items  $I$ .  $S_u = \langle i_1, i_2, \dots, i_m \rangle$  represents state of user  $u$ , which denotes that target user  $u$  has rated  $m$  items in a sequential manner. Our goal is to predict  $i_{m+1}, \dots, i_{m+k}$  that are likely to be preferred by the user in the future, where  $k$  is the number of the recommended items. System-state graph is created by Markov model where each item can be assumed as a node of a graph and the sequence of ratings can be modelled as edges between them. To predict the probability of purchasing item  $i_{m+1}$  by a user who have already purchased  $i_1, i_2, \dots, i_m$  in the past, we need to define a transition function. In fact, this function counts the frequency of  $\langle i_1, i_2, \dots, i_m \rangle$  and  $\langle i_1, i_2, \dots, i_m, i_{m+1} \rangle$  sequences in the dataset to obtain the probability of changing  $\langle i_1, i_2, \dots, i_m \rangle$  to  $\langle i_1, i_2, \dots, i_m, i_{m+1} \rangle$  sequence which is obtained from following equation.

$$TF(\langle i_1, i_2, \dots, i_m \rangle, \langle i_1, i_2, \dots, i_m, i_{m+1} \rangle) = \frac{N(\langle i_1, i_2, \dots, i_m, i_{m+1} \rangle)}{N(\langle i_1, i_2, \dots, i_m \rangle)}, \quad (1)$$

where  $N(\langle i_1, i_2, \dots, i_m, i_{m+1} \rangle)$  is the number of users who have this sequence in their ratings' history. We use this idea to propose our aggregated system-state graph.

### 3.2 Generating System-State Graph

We define the system-state graph different from classic Markov model. First, we define a threshold  $T$  to transform users' ratings to *like* if the rating is higher than or equal to  $T$ , and *dislike* if the rating is lower than  $T$ . This transformation helps us to apply the effects of the ratings in Markov model, simplify the problem and decrease the amount of calculations. With this assumption, items can have only two states: *like* and *dislike*, denoted by  $s_{i,l}$  and  $s_{i,d}$ , respectively. We model the ratings as a graph with  $s_{i,l}$  and  $s_{i,d}$  being the nodes and co-occurrence of items in the users' states  $S_u$  the edges. Let's denote this graph by  $G$ , which is an undirected graph. Assume that the ratings dataset contains  $N$  items, then  $G$  has  $2N$  nodes due to having a *like* node and a *dislike* node for each item. The weight of connection between two nodes  $s_{i,l}$  and  $s_{j,d}$  is defined as the number of users who *like* item  $i$  and at the same time *dislike* item  $j$ . Figure 1 shows an example how the ratings are transformed.

User	Item	Rate
$u_1$	$i_1$	1
	$i_2$	2
	$i_3$	5
	$i_6$	5
$u_2$	$i_1$	3
	$i_2$	3
	$i_6$	5
	$i_8$	1

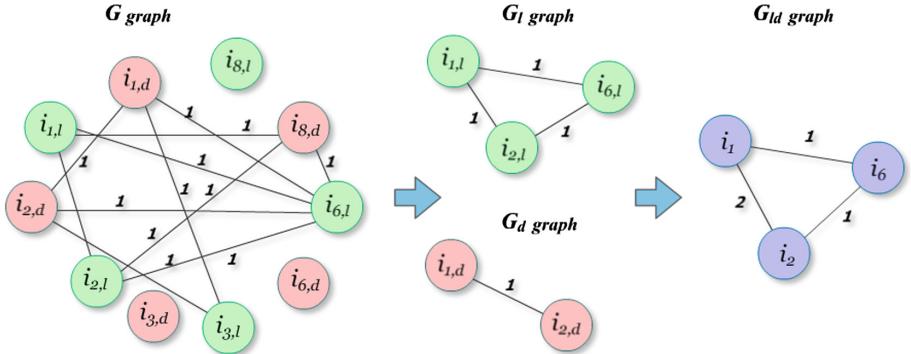
Transforming with  $T=2.5$

User	Item	Rate
$u_1$	$i_1$	dislike
	$i_2$	dislike
	$i_3$	like
	$i_6$	like
$u_2$	$i_1$	like
	$i_2$	like
	$i_6$	like
	$i_8$	dislike

**Fig. 1.** Transforming user rates to *like* and *dislike*

In the Next step, we extract two subgraphs,  $G_l$  and  $G_d$  from  $G$  that only contain relation between *like* and *dislike* nodes, respectively. Both  $G_l$  and  $G_d$  graphs show the correlation of items from different aspects;  $G_l$  shows the similarity of two items based on the number of likes they received together while  $G_d$  show the similarity of items based on dislike they received together. For example, in Fig. 1, user  $u_1$  dislikes  $i_1$  and  $i_2$ , and user  $u_2$  likes  $i_1$  and  $i_2$ . It shows  $i_1$  and  $i_2$  have a similar behaviour and get like or get dislike at same time in users' ratings history.

Since  $G_l$  and  $G_d$  have similar concepts, we merge them and make a new graph that shows the correlation between all items based on users' rating history. users' rating history; this graph is denoted by  $G_{ld}$ . To merge these graphs, we unify  $s_{i,l}$  and  $s_{i,d}$  as a single node and aggregate their edges along with their weights. The resulted graph is the system-state graph, which items in dataset and their correlations make its nodes and weights, respectively. Ultimately,  $G_{ld}$  is used for the recommendation purpose. In the next step, we traverse in  $G_{ld}$  to find items that are likely to be rated as *like* for the target user in the future. Figure 2 shows the process of generating  $G_l$ ,  $G_d$  and  $G_{ld}$  according to



**Fig. 2.** Generating G graph for users in Fig. 1

dataset in Fig. 1. There are some differences between our proposed method and classic Markov model in creating system-state graph: (i) we consider ratings in system-state graph, (ii) we don't consider the sequence of ratings (iii) we ignore part of unimportant relations when extract  $G_l$  and  $G_d$  from  $G$  to make graph compact.

### 3.3 Recommendation

If  $S_{l_u}$  represents the set of items that are rated as *like* by  $u$ , the aim is to find  $k$  items with strongest correlation with  $S_{l_u}$  in  $G_{ld}$ , and recommend them to  $u$ . The main idea is that if we find recommendations based on the users' rating history, we can assume that users are likely to be like this recommendation. Then, we add this recommendation to  $S_{l_u}$  to consider it as rating history of the target user. In other words, each time we recommend a new item, we update  $S_{l_u}$  for target user  $u$  with the items recommended until that step. Suppose that  $Rec$  is a function that generate the recommendation list  $R_u = <r_1, \dots, r_k>$  where  $k$  is the number of recommendations; the  $m$ th recommendation  $r_m$  for target user  $u$  is obtained as follow:

$$r_m = Rec(\overline{S_{l_u}}), \quad (2)$$

where  $\overline{S_{l_u}}$  is updated  $S_{l_u}$  which is obtained from union of  $S_{l_u}$  and previous recommended items as follow:

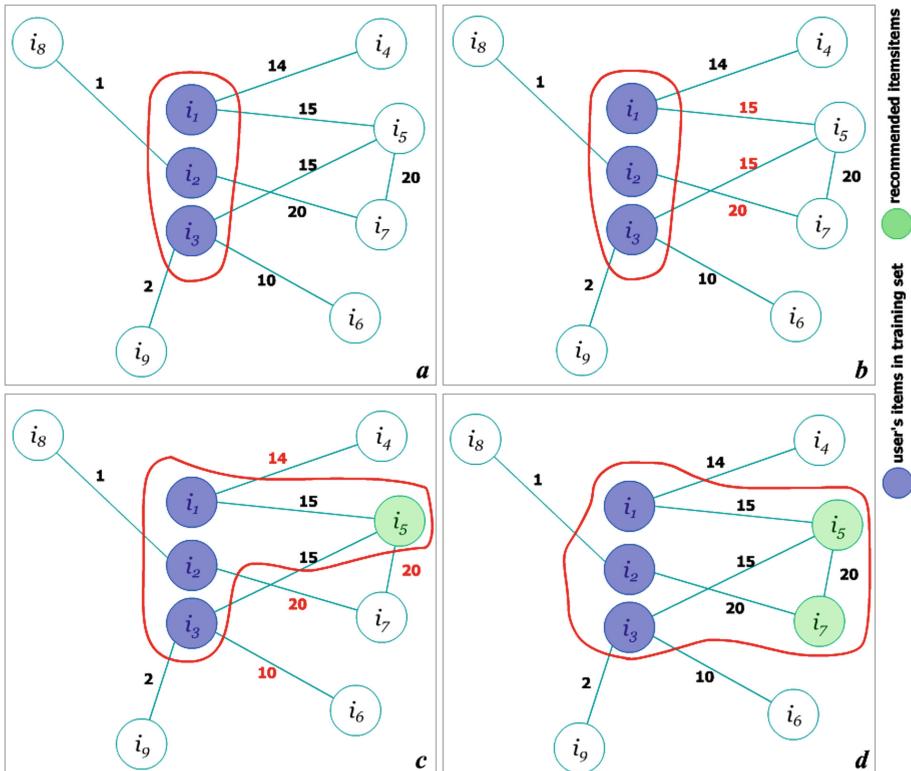
$$\overline{S_{l_u}} = S_{l_u} \cup R_u^{m-1}, \quad (3)$$

where  $R_u^{m-1}$  is the situation of  $R_u$  after adding the  $(m-1)$ th recommendation. The pseudo-code of the above function is given in Algorithm 1.

1. Input:  $S_{l_u}$ ,  $G_{ld}$ ,  $k$  //user-state, system-state graph, number of recommendations
2. Output:  $R_u$  // final recommendation list
3.  $NI_u = \{\}$ ,  $NW_u = \{\}$ ,  $R_u = \{\}$  // Initializing variables
4.  $\overline{S_{l_u}} = S_{l_u} \cup R_u^0$
5. For  $m = 1:k$
6. For each item  $j$  in  $\overline{S_{l_u}}$
7.      $max_j$  = connected item with highest weight to item  $j$  in  $G_{ld}$
8.     Add  $max_j$  to  $NI_u$  set
9.     Add weight between  $j$  and  $max_j$  to  $NW_u$  set
10. End for;
11. remove duplicate item in  $NI_u$  and Aggregate related weights in  $NW_u$
12.  $r_m$  = find item with maximum weight in  $NI_u$
13.  $R_u^m = R_u^{m-1} \cup r_m$
14.  $\overline{S_{l_u}} = S_{l_u} \cup R_u^m$
15. End for;

**Algorithm 1.** Pseudo-code of the proposed *Rec* function

To recommend item  $r_1$  to target user  $u$ , our *Rec* function finds  $S_{l_u}$  items in  $G_{ld}$  and then find connected nodes with the highest weights to them; We denote these items and their related weights by  $NI_u$  and  $NW_u$ , respectively. Since we may have repetitive nodes in  $NI_u$ , *Rec* removes the duplicates in  $NI_u$  and aggregates the related weights in  $NW_u$ . The items present in  $NI_u$  have the strongest correlation with  $S_{l_u}$  items. In other words, weights in  $NW_u$  show the probability of occurring  $NI_u$  and  $S_{l_u}$  items together in the ratings history. We select the item with the highest weight in  $NI_u$  as the first item for the recommendation. In fact, we assume all  $S_{l_u}$  items as a single node in graph and select the node in  $G_{ld}$  that has the highest correlation with  $S_{l_u}$ ; Fig. 3 shows how  $r_1$  and  $r_2$  are selected. Figure 3(a) shows  $G_{ld}$  where purple nodes and the red line around them shows  $S_{l_u}$ . In Fig. 3(b), the algorithm finds connected nodes to  $S_{l_u}$  items with highest weights which are depicted with red colour. In (c),  $i_5$  with the highest aggregated weight is selected as a first recommendation and added to  $S_{l_u}$  to update it for next round of recommendation. For recommending the next item, we assume that  $u$  like  $r_1$ , and then find  $r_2$  based on this assumption. This means that we select  $r_2$  only if  $r_1$  is preferred by the target user. To this end, we add  $r_1$  to  $S_{l_u}$  and repeat the same process with updated  $S_{l_u}$ . Figure 3(c, d) shows the process of selecting  $r_2$ . The proposed approach reveals hidden correlations between items in system-state graph and helps users to find more neighbours when they have few numbers of items in  $S_{l_u}$ , which ultimately leads to have better precision in spars datasets.



**Fig. 3.** Steps to obtain recommendation from system-state graph. (a) initial state of the target user in ratings dataset. (b) finding items with highest connected weight for each item in  $S_{u_l}$ . (c) select  $i_5$  as first recommended item and add it to initial set. (d) repeat b and c process based on updated  $S_{u_l}$  and selecting  $i_7$  as second recommendation. (Color figure online)

To make the approach clearer, imagine that  $S_{u_l} = \langle i_1, i_2, i_3 \rangle$  for target user  $u$ .  $G_{ld}$  and the process of selecting the first and second recommendation items are depicted in Fig. 3. The strongest connection between  $i_1, i_2$  and  $i_3$  with other nodes in  $G_{ld}$  are  $i_5, i_7$  and  $i_5$  respectively. Figure 4 shows the process of creating  $NI_u$  and  $NW_u$  based on  $G_{ld}$ . After aggregating weights, the item with the highest weight is selected as the recommendation, which is  $i_5$  in this example.

$S_{u_l}$	$NI_u$	$NW_u$
$i_1$	$i_5$	15
$i_2$	$i_7$	20
$i_3$	$i_5$	15

Aggregating weights

$S_{u_l}$	$NI_u$	$NW_u$
$i_1, i_3$	$i_5$	30
$i_2$	$i_7$	20

**Fig. 4.** Creating  $NI_u$  and  $NW_u$  and get the recommendation after aggregating weights

For top- $k$  recommendation problem, we repeat the above introduced process for  $k$  times, which is a simple process. Markov model often struggles with some problems, such as determining the size of state and sparsity of the dataset, which is often the case for many real datasets. Markov model only considers the previously rated items in the recommendation process and ignores valuable users' rating information. Small chain number can make another problem for Markov model, as small chains cannot accurately represent the taste of users. The proposed method aims at solving these problems, and our experiments in the next section reveals its effectiveness to the state-of-the-art recommendation methods.

## 4 Experimental Results

In this section we compare the proposed algorithm with a number of classical and state-of-the-art algorithms. Since our algorithm is a ranking method, we use the evaluation metrics, which are proper for this type of methods.

### 4.1 Evaluation Metrics

#### Precision

$P_u(N)$  is precision for a list of recommended items to user  $u$  and is defined as the percentage of relevant items to user  $u$  in their list of recommendation. Relevant items to target user are those rated as *like* by the user. Precision of a system with  $N$  users,  $P(N)$ , is calculated as:

$$P(N) = \frac{\sum_{u \in testSet} P_u(N)}{N} \quad (4)$$

#### Recall

Recall is among the most frequently used metrics of information retrieval field. Recall for a target user  $u$  is denoted by  $Recall_u(N)$  which is the proportion of relevant items to all items and Recall of a system with  $N$  users,  $Recall(N)$ , is calculated as:

$$Recall(N) = \frac{\sum_{u \in testSet} Recall_u(N)}{N} \quad (5)$$

#### Normalized Discounted Cumulative Gain (NDCG)

Discounted cumulative gain (DCG) measures the ranking quality of the recommended items. DCG increases when relevant items are placed higher in the list. It is obtained as follows:

$$DCG = rel_1 + \sum_{i=2}^{|R|} \frac{rel_i}{\log_2(i+1)} \quad (6)$$

where  $R$  is the recommendation list and  $rel_i$  shows the relative item in position  $i$ , which can be zero or one if a relevant item recommended in  $i$ th position in the recommended list or an irrelevant item is placed there, respectively. Normalized DCG (NDCG) is determined by calculating DCG and dividing it by the ideal DCG in which the recommended items are perfectly ranked:

$$IDCG = 1 + \sum_{i=2}^{|S_i|} \frac{1}{\log_2(i+1)} \quad (7)$$

## F1 SCORE

Since precision and recall are inversely correlated, it is needed to consider both of them when evaluating different algorithms. Since precision and recall are dependent on the number of recommended items, researchers have often used  $F1$  score, as a combination of precision and recall.  $F1$  is calculated as follow:

$$F1 = \frac{2 * P(N) * Recall(N)}{P(N) + Recall(N)} \quad (8)$$

## 4.2 Datasets

In this paper, we employ two well-known datasets, including MovieLens-100K and Jester, to evaluate performance of our method. The density of Jester dataset is about 10 times more than MovieLens dataset that helps us to compare the performance of the proposed algorithm in spars and dense datasets. MovieLens-100 K is a movie dataset with 943 users, 1682 items and 100,000 ratings. Jester is ratings of users to set of jokes. In this work we use a sample of the original dataset with 3000 users, 100 jokes and 165,536 ratings. The ratings in MovieLens and Jester are on a scale of 1 to 5 and -10 to +10, respectively. For all benchmarks we use the same train and test sets for recommending 10 items. Threshold  $T$  to transform data to *like* and *dislike* is set to 2.5 for MovieLens dataset and 0 for Jester dataset.

## 4.3 Results

In order to generate the result for comparison, we have used the Librec library in Java. The proposed method is developed in Matlab and compared with AspectModel [28], BPOissMF [29], EALS [17], ListRankMF [30], RankSGD [31], RankALS [32], WBPR [33], CLIMF [34], UserKNN, BUCM [35], ItemKNN, IMULT [36], and GRAD [37].

The result in Tables 1 and 2 report the performance of the algorithms in terms of different evaluation metrics over MovieLens and Jester datasets, respectively. The proposed algorithm performs better than other algorithms in terms of precision, recall, NDCG and F1 evaluation metrics in both datasets. While it is the fastest algorithm in Jester, it has the fourth fastest runtime in MovieLens, where AspectModel and ListRankMF are the fastest, and the second fastest algorithms, respectively. The performance of other algorithms differs across the datasets. While UserKNN is the second top-performer in MovieLens (after the proposed algorithm), in the other dataset,

**Table 1.** Performance of algorithms on MovieLens dataset. The best result for each metric is shown in boldface, while the second best result is shown in underlined boldface.

	Precision	Recall	NDCG	F1	Time(ms)
GBP	<b>0.3478</b>	<b>0.1475</b>	<b>0.3949</b>	<b>0.207149</b>	9500
AspectModel	0.228862	0.091245	0.247767	0.130473	<b>3148</b>
BPoissMF	0.019919	0.006309	0.015464	0.009583	16172
EALS	0.174187	0.07951	0.187264	0.109182	64966
ListRankMF	0.10122	0.047671	0.108673	0.064816	<b>4282</b>
RankSGD	0.261179	0.11386	0.292407	0.158586	13282
RankALS	0.175203	0.070117	0.189135	0.100153	572904
WBPR	0.14939	0.06305	0.152679	0.088674	104072
CLIMF	0.004065	0.00849	0.003427	0.001465	3696650
UserKNN	<b>0.305691</b>	<b>0.129918</b>	<b>0.33227</b>	<b>0.182341</b>	21259
BUCM	0.057927	0.020219	0.05347	0.029976	6368
ItemKNN	0.030081	0.012945	0.029488	0.0181	22388
IMULT	0.1842	0.0657	0.2068	0.096854	3402569
GRAD	0.0774	0.0429	0.0914	0.055203	19302

**Table 2.** Performance of algorithms on Jester dataset. The best result for each metric is shown in boldface, while the second best result is shown in underlined boldface.

	Precision	Recall	NDCG	F1	Time(ms)
GBP	<b>0.7331</b>	<b>0.7443</b>	<b>0.823</b>	<b>0.738658</b>	<b>11880</b>
AspectModel	<b>0.553032</b>	<b>0.527585</b>	0.628423	<b>0.540009</b>	24522
BPoissMF	0.186442	0.174734	0.214315	0.180398	32608
EALS	0.238501	0.217747	0.265346	0.227652	47041
ListRankMF	0.36413	0.341146	0.433668	0.352264	15783
RankSGD	0.381922	0.366335	0.445621	0.373967	<b>13709</b>
RankALS	0.323398	0.309093	0.384412	0.316084	103244
WBPR	0.406522	0.375251	0.43215	0.390261	71296
CLIMF	0.115103	0.103977	0.085998	0.109257	1602445
UserKNN	0.446568	0.41683	0.482625	0.431187	26096
BUCM	0.307723	0.288875	0.343151	0.298002	14850
ItemKNN	0.136041	0.123463	0.127869	0.129447	36769
IMULT	0.562	0.4438	0.6301	0.495955	985245
GRAD	0.5133	0.4081	<b>0.6318</b>	0.454694	24834

AspectModel is the second top-performer in terms of precision, recall and F1 and GRAD has the second best performance for NDCG. The result shows that proposed algorithm is not so sensitive about increasing number of users and by increasing 300% in number of users, time of recommendation increases only 20%. In addition, while most of algorithms have a big change in evaluation ranking by changing the dataset, GBP almost shows dataset-independent behaviour in comparison with other algorithms.

## 5 Conclusion

In this paper, we introduced a probabilistic graph-based method to obtain accurate recommender systems. The proposed method that called PGB, uses classic Markov model idea to makes a system-state graph based on users' ratings history, and then traverses the graph to predict items which are likely to be preferred by uses in the future. Selecting each item for recommendation is conditioned by considering recommended items in the previous steps. This approach uses a probabilistic model to consider the items which are likely to be preferred by users in the future. Experimental results performed on two real-world datasets including MovieLens and Jester, demonstrate that the proposed method significantly outperforms several traditional and state-of-the-art recommender systems in terms of precision, recall, NDCG and F1.

## References

1. Quan, T.K., Fuyuki, I., Shinichi, H.: Improving accuracy of recommender system by clustering items based on stability of user similarity. In: CIMCA 2006. IEEE (2006)
2. Pazzani, M.J., Billsus, D.: Content-based recommendation systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) *The Adaptive Web. LNCS*, vol. 4321, pp. 325–341. Springer, Heidelberg (2007). [https://doi.org/10.1007/978-3-540-72079-9\\_10](https://doi.org/10.1007/978-3-540-72079-9_10)
3. Bobadilla, J., et al.: Recommender systems survey. *Knowl.-Based Syst.* **46**, 109–132 (2013)
4. Burke, R.: Hybrid recommender systems: survey and experiments. *User Model. User-Adap. Inter.* **12**(4), 331–370 (2002)
5. Winoto, P., Tang, T.Y.: The role of user mood in movie recommendations. *Expert Syst. Appl.* **37**(8), 6086–6092 (2010)
6. Javari, A., Jalili, M.: A probabilistic model to resolve diversity–accuracy challenge of recommendation systems. *Knowl. Inf. Syst.* **44**(3), 609–627 (2015)
7. Castro-Schez, J.J., et al.: A highly adaptive recommender system based on fuzzy logic for B2C e-commerce portals. *Expert Syst. Appl.* **38**(3), 2441–2454 (2011)
8. Núñez-Valdés, E.R., et al.: Implicit feedback techniques on recommender systems applied to electronic books. *Comput. Hum. Behav.* **28**(4), 1186–1193 (2012)
9. Porcel, C., et al.: A hybrid recommender system for the selective dissemination of research resources in a technology transfer office. *Inf. Sci.* **184**(1), 1–19 (2012)
10. Tan, S., et al.: Using rich social media information for music recommendation via hypergraph model. *ACM Trans. Multimedia Comput. Commun. Appl. (TOMM)* **7**(1), 22 (2011)
11. Barragáns-Martínez, A.B., et al.: A hybrid content-based and item-based collaborative filtering approach to recommend TV programs enhanced with singular value decomposition. *Inf. Sci.* **180**(22), 4290–4311 (2010)
12. Costa-Montenegro, E., Barragáns-Martínez, A.B., Rey-López, M.: Which App? A recommender system of applications in markets: implementation of the service for monitoring users' interaction. *Expert Syst. Appl.* **39**(10), 9367–9375 (2012)
13. Bobadilla, J., Serradilla, F., Hernando, A.: Collaborative filtering adapted to recommender systems of e-learning. *Knowl.-Based Syst.* **22**(4), 261–265 (2009)
14. McNally, K., et al.: A case study of collaboration and reputation in social web search. *ACM Trans. Intell. Syst. Technol. (TIST)* **3**(1), 4 (2011)

15. Jalili, M., et al.: Evaluating collaborative filtering recommender algorithms: a survey. *IEEE Access* **6**, 74003–74024 (2018)
16. Li, X., Wang, H., Yan, X.: Accurate recommendation based on opinion mining. In: Sun, H., Yang, C.-Y., Lin, C.-W., Pan, J.-S., Snasel, V., Abraham, A. (eds.) *Genetic and Evolutionary Computing*. AISC, vol. 329, pp. 399–408. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-12286-1\\_41](https://doi.org/10.1007/978-3-319-12286-1_41)
17. He, X., et al.: Fast matrix factorization for online recommendation with implicit feedback. In: *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM (2016)
18. Santos, B.S., et al.: Integrating user studies into computer graphics-related courses. *IEEE Comput. Graph. Appl.* **31**(5), 14–17 (2011)
19. Deo, N.: *Graph Theory with Applications to Engineering and Computer Science*. Courier Dover Publications, Mineola (2017)
20. Augustyniak, P., Ślusarczyk, G.: Graph-based representation of behavior in detection and prediction of daily living activities. *Comput. Biol. Med.* **95**, 261–270 (2018)
21. Mobasher, B.: Data mining for web personalization. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) *The Adaptive Web*. LNCS, vol. 4321, pp. 90–135. Springer, Heidelberg (2007). [https://doi.org/10.1007/978-3-540-72079-9\\_3](https://doi.org/10.1007/978-3-540-72079-9_3)
22. Rendle, S., Freudenthaler, C., Schmidt-Thieme, L.: Factorizing personalized Markov chains for next-basket recommendation. In: *Proceedings of the 19th International Conference on World Wide Web*. ACM (2010)
23. Cheng, C., et al.: Where you like to go next: successive point-of-interest recommendation. In: *Twenty-Third International Joint Conference on Artificial Intelligence* (2013)
24. Shani, G., Heckerman, D., Brafman, R.I.: An MDP-based recommender system. *J. Mach. Learn. Res.* **6**(Sep), 1265–1295 (2005)
25. He, Q., et al.: Web query recommendation via sequential query prediction. In: *2009 IEEE 25th International Conference on Data Engineering*. IEEE (2009)
26. Sahoo, N., Singh, P.V., Mukhopadhyay, T.: A hidden Markov model for collaborative filtering. In: *Management Information Systems Quarterly*, Forthcoming (2010)
27. Yang, Q., et al.: Personalizing web page recommendation via collaborative filtering and topic-aware markov model. In: *2010 IEEE International Conference on Data Mining*. IEEE (2010)
28. Hofmann, T., Puzicha, J.: Latent class models for collaborative filtering. In: *IJCAI* (1999)
29. Salakhutdinov, R., Mnih, A.: Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In: *Proceedings of the 25th International Conference on Machine Learning*. ACM (2008)
30. Shi, Y., Larson, M., Hanjalic, A.: List-wise learning to rank with matrix factorization for collaborative filtering. In: *Proceedings of the Fourth ACM Conference on Recommender Systems*. ACM (2010)
31. Töscher, A., Jahrer, M.: Collaborative filtering ensemble for ranking. *J. Mach. Learn. Res. W&CP* **18**, 61–74 (2012)
32. Takács, G., Tikk, D.: Alternating least squares for personalized ranking. In: *Proceedings of the Sixth ACM Conference on Recommender Systems*. ACM (2012)
33. Gantner, Z., et al.: Personalized ranking for non-uniformly sampled items. In: *Proceedings of KDD Cup 2011* (2012)
34. Shi, Y., et al.: CLiMF: learning to maximize reciprocal rank with collaborative less-is-more filtering. In: *Proceedings of the Sixth ACM Conference on Recommender Systems*. ACM (2012)

35. Barbieri, N., et al.: Modeling item selection and relevance for accurate recommendations: a Bayesian approach. In: Proceedings of the Fifth ACM Conference on Recommender Systems. ACM (2011)
36. Ranjbar, M., et al.: An imputation-based matrix factorization method for improving accuracy of collaborative filtering systems. *Eng. Appl. Artif. Intell.* **46**, 58–66 (2015)
37. Lin, C.-J.: Projected gradient methods for nonnegative matrix factorization. *Neural Comput.* **19**(10), 2756–2779 (2007)



# A Robust Deep Ensemble Classifier for Figurative Language Detection

Rolandos-Alexandros Potamias<sup>(✉)</sup>, Georgios Siolas, and Andreas Stafylopatis

Artificial Intelligence and Learning Systems Laboratory,

School of Electrical and Computer Engineering,

National Technical University of Athens, Athens, Greece

[rolpotamias@gmail.com](mailto:rolpotamias@gmail.com), [gsiolas@islab.ntua.gr](mailto:gsiolas@islab.ntua.gr), [andreas@cs.ntua.gr](mailto:andreas@cs.ntua.gr)

**Abstract.** Recognition and classification of Figurative Language (FL) is an open problem of Sentiment Analysis in the broader field of Natural Language Processing (NLP) due to the contradictory meaning contained in phrases with metaphorical content. The problem itself contains three interrelated FL recognition tasks: sarcasm, irony and metaphor which, in the present paper, are dealt with advanced Deep Learning (DL) techniques. First, we introduce a data prepossessing framework towards efficient data representation formats so that to optimize the respective inputs to the DL models. In addition, special features are extracted in order to characterize the syntactic, expressive, emotional and temper content reflected in the respective social media text references. These features aim to capture aspects of the social network user's writing method. Finally, features are fed to a robust, Deep Ensemble Soft Classifier (DESC) which is based on the combination of different DL techniques. Using three different benchmark datasets (one of them containing various FL forms) we conclude that the DESC model achieves a very good performance, worthy of comparison with relevant methodologies and state-of-the-art technologies in the challenging field of FL recognition.

**Keywords:** Sentiment Analysis · Natural Language Processing · Figurative Language · Sarcasm · Irony · Deep Learning · Ensemble classifier

## 1 Introduction

Figurative language (FL), as a linguistic phenomenon, refers to the contradiction between the literal and non-literal meaning of a sentence. Detection of FL is an open problem of computational linguistics mostly because of the difference or distance between the form and the actual meaning of a sentence. In the present paper, we address sentiment analysis of FL as well as tracking three most common types of FL, irony (Ancient Greek: εἰρωνεία), sarcasm (σαρκασμός) and metaphor (μεταφορά). Below are three indicative FL examples:

- *I just love being ignored (irony)*
- *So many useless classes, great to be student (sarcasm)*
- *You're about as genuine as a used car salesman. (metaphor)*

There are two types of irony, situational and verbal, depending on its use. Verbal irony is defined as the FL phenomenon in which the author denotes exactly the opposite of what he/she means. Aristotle himself described irony as an “refined insult” highlighting its trenchant use. In contrast with irony, sarcasm cannot be easily defined. Both Oxford<sup>1</sup> and Merriam Webster<sup>2</sup> Dictionaries do not disserve sarcasm from irony, describing sarcasm as “the use of irony to mock or convey contempt”. Sarcastic comments tend to be more condescending in contrast with ironic ones that mostly denote humour. On the other hand, metaphor, a super set of irony and sarcasm, is a figure of speech that when taken in its literal sense makes no or little meaning, with its underlying meaning to be still easily understood. These definitions of FL forms imply the ambiguity separating literal and non-literal comments.

Despite that all forms of FL are well studied linguistic phenomena [24], computational approaches fail to identify the polarity of them within a text. The influence of FL in sentiment classification emerged on SemEval-2014 Sentiment Analysis task [22]. Results show that Natural Language Processing (NLP) systems effective in most other tasks see their performance drop when dealing with figurative forms of language. Thus, methods capable of detecting, separating and classifying forms of FL would be valuable building blocks for a system that could ultimately provide a full-spectrum sentiment analysis of natural language.

In this paper we present a new method for FL detection. The novelty of our approach is founded on: (a) the integration of different deep leaning architectures (LSTMs, DNN); (b) the introduction of an ensemble between different neural classifiers; and (c) the combination of different representations including word-embeddings and engineered features.

## 2 Literature Review

Despite all forms of FL have been studied independently by the Machine Learning community, none of the proposed systems have been tested on more than one problem. Related work on the impact of FL in sentence sentiment classification problems are usually categorized with respect to their subject: irony, sarcasm detection and sentiment analysis of figurative language. Many researchers tend to treat sarcasm and irony as an identical phenomenon in their works but we will investigate each subject separately.

### 2.1 Irony and Sarcasm Detection

Irony detection was first explored by Reyes [19, 20]. Indeed, his work carried out the first comprehensive studies on FL from a computer science perspective.

---

<sup>1</sup> <https://en.oxforddictionaries.com/definition/sarcasm>.

<sup>2</sup> <https://www.merriam-webster.com/dictionary/sarcasm>.

He approached irony as a linguistic phenomenon, undistinguished from sarcasm, that contains something unexpected or contradictory. Decision trees were used to classify features indicating unexpectedness like emotional words, contradictory terms and punctuation. A similar process was followed by Barbieri [4], where he performed sarcasm detection in topics such as politics, education and humor. Data was collected from Twitter using the hashtags #sarcasm, #politics, #education, #humour. He calculates a measure of the unexpectedness and possible ambiguities, based on words that are mainly used in spoken language based on the American National Corpus Frequency Data<sup>3</sup> as well as the tweets morphology. Data classification is performed using Random Forests and Decision Trees. Buschmeir [5] used Logistic Regression to recognize irony and focused on the unexpectedness factor which is considered as an emotional imbalance between words in the text. In another direction, an attentive deep learning approach was implemented by Huang [13] using Google's pre-trained Word2vec Embeddings [16]. A pattern-based method to detect irony was proposed by Carvalho [6], using n-grams and combinations of adverbs and acronyms that indicate humor, as features. A context approach was proposed by Wallace [25], to indicate irony on Reddit posts using previous and following comments. On *Semantic Evaluation Workshop-2018 task ‘Irony Detection in English Tweets’* [12] four teams showcased remarkable results. Team named *Thu-Ngn* used a fully-connected Long short-term memory (LSTM) with pretrained Word Embeddings and a combination of syntactic and emotional features. An unweighted average between character and word level bidirectional LSTMs was proposed by the Ntua-Slp team, using an attention layer to determine irony. A majority ensemble classifier consisting of Linear Regression and SVM was proposed by WLV, using Word-Emoji embeddings as well as features highlighting the tweet's emotional background. Finally, the NLPRL-IITBHU team proposed features based on the opposition and disharmony of words. Ghosh [10] claims that the key for sarcasm detection is the contradiction between words within the same tweet. He implements a SVM kernel function, based on similarity measure of sarcastic tweets as well as word2vec embeddings. Davidov's [7] semi-supervised pattern-based approach used attributes like words frequency and content words to classify them using k-Nearest Neighbours. On the other hand, Ibáñez [11] proposed a different procedure considering sarcasm as a binary classification task, against positive-negative comments. Features are both lexical, extracted using WordNet-Affect and LWIC, and so-called “pragmatic factors”, reflecting users sentiment. A behavioral approach was implemented by Rajadesingan [18], recording user's background information. In this work both behavioral and sentiment contrast features fed an SVM classifier. Word Embeddings combined with (Convolutional Neural Networks) CNN-LSTM units were used by Kumar [15] and Ghosh and Veale [9] resulting state-of-the-art performance.

---

<sup>3</sup> <http://www.anc.org/data/anc-second-release/frequency-data/>.

## 2.2 Sentiment Analysis on Figurative Language

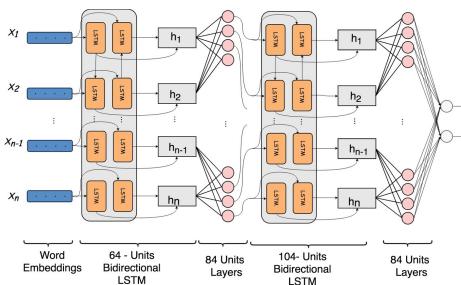
The Semantic Evaluation Workshop-2015 [8] proposed a joint task to evaluate the impact of FL in sentiment analysis on ironic, sarcastic and metaphorical data from tweets. The ClaC team exploited four lexicons to extract attributes as well as syntactic features to identify sentiment polarity. The UPF team used regression to classify features extracted using lexicons such as SentiWordNet and DepecheMood. The LLT-PolyU team used semi-supervised Regression and Decision Trees, given uni-gram and bi-gram models. Possible word contradiction at short distances was taken into account. A SVM-based classifier was used by the Elirf team, given n-gram and Tf-idf features. In addition, lexicons such as Affin, Pattern and Jeffrey10 were also utilized. Finally, the LT3 team used an ensemble Regression and SVM semi-supervised classifier. The features consisted mainly of a range of lexical data combined with WordNet and DBpedia11.

## 3 A Deep Learning Architecture for Figurative Language Recognition

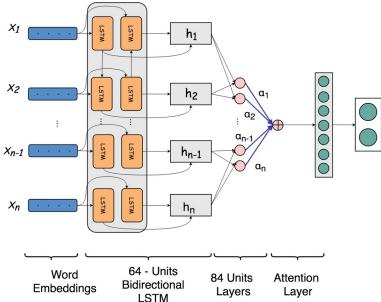
Our work proposes an ensemble architecture consisting of three deep models detailed in the present section, a BiLSTM, an AttentionLSTM and a Dense NN. Both BiLSTM and AttentionLSTM are sequential architectures comprising LSTM cells, fed with pre-trained GloVe Word Embeddings [17]. The DNN model is fed with several features extracted from each tweet, such as uni-grams, bi-grams Tf-Idf representations of text combined with syntactic, demonstrative, sentiment, mood and readability features to track FL. The proposed robust ensemble classifier (DESC) uses soft classification techniques.

### 3.1 BiLSTM

LSTM cell architectures tend to perform significantly better than regular neural networks in exploiting sequential data. An ordered input vector  $\mathbf{x} = (x_1, x_2, \dots, x_n)$  is mapped to an output vector  $\mathbf{y} = (y_1, y_2, \dots, y_n)$  throughout computing hidden state vector  $\mathbf{h} = (h_1, h_2, \dots, h_n)$  repetitively. Cell  $i$  uses information from previous cells in time, creating a context feature, in the neural network architecture. However, sequential data, as text, speech or frame series often require knowledge of both past and future context. Bidirectional LSTMs solve this problem by exploiting input data, summarizing both the past and future context of each time sample. They assign two hidden states for the same time sample calculated in both directions in order to feed the output layer. The forward hidden sequence  $\overrightarrow{\mathbf{h}}$  is calculated from reading input  $x_1$  to  $x_n$  while on the other hand the backward hidden sequence  $\overleftarrow{\mathbf{h}}$  reads input from  $x_n$  to  $x_1$ . The total hidden state determines the time sample  $i$  by concatenating both forward and backward hidden states, that is  $h_i = \overrightarrow{h_i} || \overleftarrow{h_i}$ . In our work, we implement a deep two-layered bidirectional LSTM (BiLSTM) stacked with a dense layer between them as shown in Fig. 1.



**Fig. 1.** BiLSTM architecture consisting of two bidirectional LSTM layers, LeakyReLU activated and fully connected with a dense layer.



**Fig. 2.** AttentionLSTM architecture consisting of one bidirectional LSTM layer, LeakyReLU activated and followed by an attention layer

### 3.2 AttentionLSTM

Additionally, a mechanism focusing on the most significant time sample is added on top of the regular LSTMs. This mechanism is called attention layer, was introduced by [3] and maps hidden states according their importance. In other words, an attention layer assigns a value  $\alpha_i$  to each state, which is called the attention factor, and is represented by a so-called attentive vector  $r$ :

$$r_t = \tanh(W_h h_t + b_t) \quad (1)$$

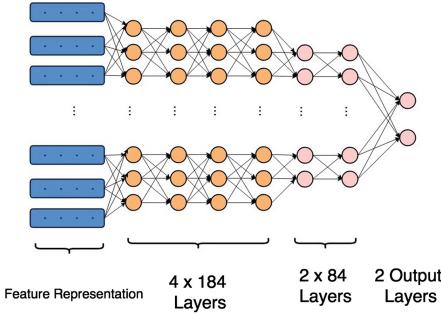
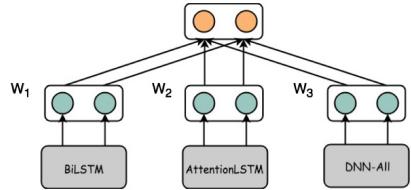
$$a_t = \text{softmax}(r_t) = \frac{e^{r_t}}{\sum_{j=0}^T e^{r_j}}, \quad \sum_{t=1}^n a_t = 1 \quad (2)$$

$$s = \sum_{t=0}^T a_t h_t \quad (3)$$

where  $W_h$  and  $b_t$  are the LSTM model weights, optimized during training. As FL detection often demands focus on the sentiment contrast between words we implement an architecture based on an attentive LSTM layer (AttentionLSTM) as shown in Fig. 2. Finally, a dense softmax activation layer is applied to the  $s$  feature vector representation of the tweet for the classification step.

### 3.3 Dense Neural Network

Finally, we implemented a dense fully connected deep neural network consisting of six layers. ReLU activation is used on every neuron and a dense vector representation of the features is presented the input as shown in Fig. 3. Detailed feature representations are presented in Sect. 4.2.

**Fig. 3.** DNN architecture**Fig. 4.** DESC: Deep Ensemble Soft Classifier architecture

### 3.4 Deep Ensemble Soft Classifier-DESC

In order to capture FL in Twitter we use an ensemble technique, combining different classifiers. All three classifiers described above use disparate features at textual and word level. The ensemble method makes prediction based on weighted confidence scores from the three classifiers. The classifiers impact to the final classification is determined through cross validation during training phase:

$$w_i = \frac{e^{F_1^i}}{\sum_{i=1}^3 e^{F_1^i}}, \quad i = 1, 2, 3 \quad (4)$$

where  $F_1^i$  stands for  $F_1$  score of classifier  $i$ . The final prediction is made by combining the confidence scores of all three classifiers confidence scores multiplied with a scaling factor  $w_i$ .

$$o_j = \arg \max_c \sum_{i=1}^3 w_i * \vec{p}_i^j, \quad \vec{p}_i^j \in \mathbb{R}^c \quad (5)$$

where  $j$  denotes the sample  $j$  of data set,  $\vec{p}_i^j$  the confidence score predicted by classifier  $i$  for sample  $j$  and  $c$  is the number of classes.

## 4 Experimental Setup

### 4.1 Datasets

Detecting FL, in addition to its inherent difficulty as a problem, is a particularly difficult task since there is not much benchmark data available. To examine DESC robustness and reliability, we investigate three different Twitter-based datasets. First, we detect ironic comments on Twitter using SemEval-2018's 'Detecting Irony in English Twitter' balanced dataset [12], consisting of 3834 train and 784, gold standard, test data. In addition, we utilized an imbalanced dataset, containing sarcastic tweets, compiled by Riloff et al. [21]. Riloff's dataset

is a high-quality dataset as indicated by its high Cohen's kappa score and consists of 2278 tweets, only 506 of them being sarcastic. Finally, to evaluate our model's performance on sentiment analysis we acquired a dataset containing all forms of figurative language as well as their sentiment polarity, as proposed on SemEval's 2015 Task-11 [8]. This dataset is also composed by figurative tweets and contains overall 8000 training and 4000 test tweets. Each tweet sample is ranked on a 11-point scale according to their sentiment polarity, ranging from -5 (negative sentiment polarity, for tweets with critical and ironic meanings) to +5 (positive sentiment polarity, for tweets with very upbeat meanings).

## 4.2 Feature Engineering

As already mentioned, figurative language detection is a challenging NLP task, requiring complex attributes hopefully capturing non literal use of language. Every figure of speech tend to relate with linguistic patterns which discriminate it from other ones. By the same reasoning, we claim that if we could capture all aspects and structural differences between literal and figurative language we could enhance classification predictions. As demonstrated in Fig. 5, simple patterns such as sentiment polarity and average word length per tweet could denote non literal figures of speech. Thus, we try to pursue the user's intention using syntactic, semantic and emotional language structure. We combine unigram and bi-gram models with 44 features extracted from text. These features are fed to the DNN and can be categorized into four major groups.

- **Syntactic Features (12):** This group contains features indicative of the user's syntactic usage habits, and consists of the frequency of the Part-Of-Speech(POS) tags.
- **Demonstrative Features (8):** Attributes implying user's expression, such as: the number of words and emojis used, average length of words in the tweet, frequency of punctuation marks, duplicate letters that may express user's emphasis, as well as the frequency of polysyllabic words.
- **Sentiment Features (12):** As argued before, FL can be defined as sentiment contrast between words in short distance. This necessitated the use of many lexicons, in order to obtain a sentiment estimation in word level, and then estimate the overall tweet positive, negative and contradictory sentiment, as shown in Eq. 6. For this purpose, we calculate the tweet's average positive and negative sentiment using SentiWordNet [2], VADER [14], Afinn [1], DepecheMood [23], as well as their positive-negative variance by the following formulas:

$$S_i^{pos} = \frac{1}{n_i} \sum_{j=1}^{n_i} w_{i,j}^{pos}, \quad S_i^{neg} = \frac{1}{n_i} \sum_{j=1}^{n_i} w_{i,j}^{neg}, \quad S_i^{contrast} = S_i^{pos} - S_i^{neg} \quad (6)$$

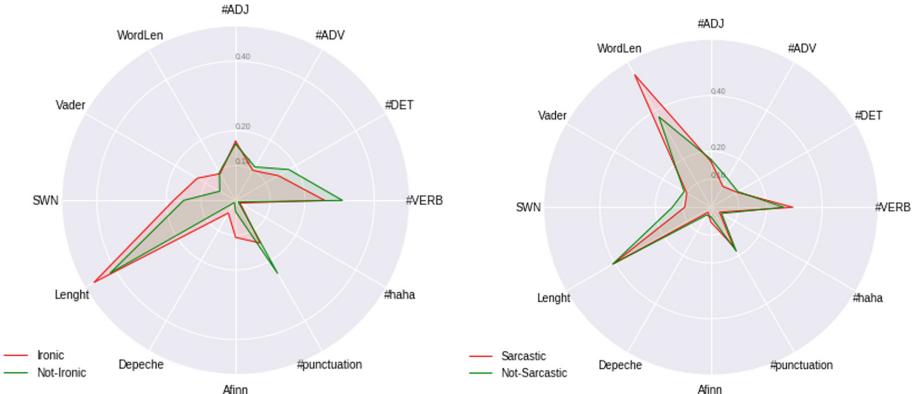
- **Mood Features (8):** An additional lexicon, DepecheMood, was used to annotate words indicating the user's temper. Mood features describe user's happiness, sadness, annoyance, inspiration, fear, indifference, anger and amusement and can help us to detect non literal quotes.

- **Readability Features (4):** Finally, we claim that all FL forms tend to have different readability scores than literally ones. Thus, we implement four metrics measuring text readability using well known readability scores. First, we enumerate words not present on Dale Chall’s list and afterwards we calculate three readability scores:

$$\text{Dale Chall} = 0.1579 \left( \frac{\text{difficult words}}{\text{words}} \times 100 \right) + 0.0496 \left( \frac{\text{words}}{\text{sentences}} \right) \quad (7)$$

$$\text{Flesch} = 206.835 - 1.015 \left( \frac{\text{total words}}{\text{total sentences}} \right) - 84.6 \left( \frac{\text{total syllables}}{\text{total words}} \right) \quad (8)$$

$$\text{Gunning Fog} = 0.4 \left[ \left( \frac{\text{words}}{\text{sentences}} \right) + 100 \left( \frac{\text{complex words}}{\text{words}} \right) \right] \quad (9)$$



**Fig. 5.** Structural differences between irony and sarcasm

## 5 Experimental Results

We compared the proposed DESC method against a variety of classifiers and different feature combinations. We experimented with different combinations of Tf-Idf and the features presented in Sect. 4.2 using a deep neural network architecture and also an SVM, frequently used by the teams participating in SemEval. As illustrated in Table 1, DESC outperforms all baseline classifiers with consistent (over all metrics) performance figures, a fact that is indicative for the robustness of the proposed approach. In addition, we can observe the stability that DESC demonstrates on all datasets, including unbalanced and multiclass-classification problems. The abbreviations used in the table for the features are unigrams, bigrams along with feature Feature Vector (FeatVec) (2inp), unigrams (uni) only, Tf-idf (Tfidf), the feature set of Sect. 4.2 (FeatVec) as well the concatenation of all the features (All).

**Table 1.** Comparison with baseline classifiers (various set-ups of DNN, SVM, AttLSTM, and BiLSTM) for the tasks of (a) irony and sarcasm detection (binary), and (b) sentiment polarity detection (eleven ordered values, -5 for “very negative” to 5 for “very positive”) - bold figures indicate superior performance.

(a) Irony & Sarcasm detection												(b) Sentiment polarity detection					
Irony SemVal-2018-Task 3.A[12]						Sarcasm Riloff [21]						Average					
System	Acc	Pre	Rec	F1	AUC	Acc	Pre	Rec	F1	AUC	Acc	Pre	Rec	F1	AUC	COS	MSE
DNN-2inp	0.63	0.64	0.62	0.63	0.65	0.82	0.78	0.81	0.79	0.84	0.73	0.71	0.72	0.71	0.75	0.602	4.230
DNN-Tfidf	0.65	0.65	0.63	0.64	0.70	0.83	0.81	0.83	0.80	0.72	0.74	0.73	0.73	0.72	0.71	0.710	3.170
DNN-uni	0.65	0.68	0.65	0.65	0.72	0.79	0.78	0.81	0.79	0.74	0.72	0.73	0.73	0.72	0.73	0.690	8,430
DNN-All	0.66	0.69	0.66	0.67	0.75	0.83	0.81	0.83	0.81	0.82	0.75	0.75	0.75	0.74	0.79	0.789	2,790
DNN-FeatVec	0.64	0.65	0.65	0.65	0.71	0.81	0.81	0.83	0.80	0.72	0.73	0.73	0.74	0.73	0.71	0.680	3,230
SVM-Tfidf	0.65	0.68	0.65	0.66	0.70	0.82	0.80	0.82	0.80	0.80	0.74	0.74	0.74	0.73	0.75	0.720	2,890
SVM-FeatVec	0.59	0.59	0.59	0.59	0.60	0.82	0.73	0.81	0.75	0.76	0.71	0.66	0.70	0.67	0.68	0.700	3,390
SVM-All	0.66	0.69	0.66	0.67	0.75	0.83	0.81	0.83	0.81	0.81	0.75	0.75	0.75	0.74	0.78	0.723	2,810
AttentionLSTM	0.71	0.70	0.71	0.70	0.75	0.85	0.83	0.85	0.83	0.84	0.78	0.77	0.78	0.77	0.80	0.749	2,860
BiLSTM	0.71	0.71	0.71	0.70	0.76	0.85	0.85	0.85	0.85	0.85	0.78	0.78	0.78	0.78	0.81	0.704	3,220
<b>DESC</b>	<b>0.74</b>	<b>0.73</b>	<b>0.73</b>	<b>0.73</b>	<b>0.78</b>	<b>0.87</b>	<b>0.87</b>	<b>0.86</b>	<b>0.87</b>	<b>0.86</b>	<b>0.81</b>	<b>0.80</b>	<b>0.80</b>	<b>0.82</b>	<b>0.820</b>	<b>2,480</b>	

We tested DESC against the performance scores of all models submitted and published in SemEval-2015 [8] Sentiment Analysis task. DESC achieves 0.82 in cosine similarity whereas the winning team [26] obtained 0.758; ranked also on 4th position regarding the MSE measure. At the same time, ClAC and UPF teams are the only one to obtain a better MSE value than DESC, as illustrated in Table 4. Further evidence of the robustness of DESC across all metrics on both Irony [12] and Sarcasm [21] detection is illustrated in Tables 2 and 3. Specifically, DESC outperforms all submissions on [12] regarding F1 measure; a fact that it is indicative for the balance achieved between precision and recall figures which also satisfies the desired property for low false-positives regarding the task of automated twitter classification. In addition, DESC’s performance is highly

**Table 2.** Systems comparison on SemEval-2018-Task 3-A<sup>a</sup> ironic dataset.

Submission	Acc	Pre	Rec	F1
THU_NGN	0.73	0.63	0.80	0.71
NTUA-SLP	0.73	0.65	0.69	0.67
WLV	0.64	0.53	<b>0.84</b>	0.65
rangwani_harsh	0.66	0.55	0.79	0.65
NIHRIO, NCL	0.70	0.61	0.69	0.65
<b>DESC</b>	<b>0.74</b>	<b>0.73</b>	0.73	<b>0.73</b>

<sup>a</sup>Officially submitted results reported in [12]. From the total of 43 submissions only the ones that exhibit performance figures higher than the average of all teams for all four metrics are included

**Table 3.** Systems comparison on Riloff’s imbalanced sarcastic dataset.

System submission	Pre	Rec	F <sub>1</sub>
Riloff		0.44	0.62
Ghosh&Veale	<b>0.88</b>	<b>0.88</b>	<b>0.88</b>
<b>DESC</b>	0.87	0.86	0.87

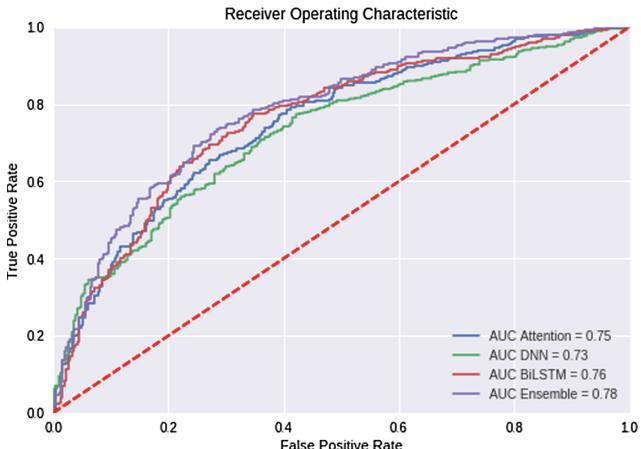
**Table 4.** Systems comparison on Sentiment analysis SemEval-2015-Task 11<sup>a</sup> according to task's official metrics, Mean Squared Error and Cosine similarity.

System submission	Cosine	MSE
Naive Bayes	0,39	5,672
MaxEnt	0,43	5,450
Decision Tree	0,55	4,065
CLaC	0,76	<b>2,117</b>
UPF	0,71	2,458
LLT_PolyU	0,69	2,600
LT3	0,66	2,913
ValenTo	0,63	2,999
PRHLT	0,62	3,023
CPH	0,62	3,078
elirf	0,66	3,096
<b>DESC</b>	<b>0,82</b>	2,480

<sup>a</sup>Officially submitted results reported in [8]. From the total of 15 submissions only the ones that exhibit higher performance than the average over both metrics are included. The first three entries refer to reported values for three baseline classifiers

increased compared to Riloff's initial proposal [21] and is very close to Ghosh and Veale [9].

Finally, combining all three AttentionLSTM, BiLSTM and DNN-all classifiers in an Ensemble model we can detect Irony with increased confidence as illustrated in Fig. 6.

**Fig. 6.** ROC-AUC curve for three ensemble classifiers on SemEval's-2018 Irony detection task.

## 6 Conclusion and Future Work

In this paper, we propose an ensemble method for sentiment classification and Figurative Language tracking with non contextual information, in short texts segments like tweets. The proposed method is based on a combination of word and sentence based features and outperforms published SemEval models in most metrics and on all datasets. In addition, we introduce a feature level approach using a variety of author’s sentiment, mood and expressiveness to detect FL usage. Beyond that, DESC shows improved AUC performance at the difficult task of irony detection.

DESC model could be even more accurate if we enhance past-present diversion since sarcastic utterances tend to include more time-related contradictions as Rajadesingan [18] denoted. Beside time and sentiment contradictions a behavioral approach could be used to analyze change’s in user’s stylistic patterns such as average tweet length, user’s word style familiarity or even user’s sarcastic intention. Another key improvement could be obtained by measuring the gap between written and informal spoken style using the ANC Frequency Data corpus as Barbieri [4] proposed. In addition, Linguistic Inquiry and Word Count (LIWC) could be used to extract vectorized lingual patterns indicating FL. Finally, the approach presented in this paper is based solely on data from Twitter’s short texts. A future FL system could become even more effective if trained and tuned using texts including FL from more sources.

## References

1. Nielsen, F.A.: A new ANEW: evaluation of a word list for sentiment analysis in microblogs. arXiv e-prints, March 2011
2. Baccianella, S., Esuli, A., Sebastiani, F.: SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining, vol. 10, January 2010
3. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. CoRR abs/1409.0473 (2014)
4. Barbieri, F., Saggin, H.: Modelling Irony in Twitter. In: EACL (2014)
5. Buschmeier, K., Cimiano, P., Klinger, R.: An Impact Analysis of Features in a Classification Approach to Irony Detection in Product Reviews, January 2014. <https://doi.org/10.3115/v1/W14-2608>
6. Carvalho, P., Sarmento, L., Silva, M., Oliveira, E.: Clues for detecting irony in user-generated contents: oh... It’s “so easy”; -. In: International Conference on Information and Knowledge Management, Proceedings (2009)
7. Davidov, D., Tsur, O., Rappoport, A.: Semi-supervised recognition of sarcastic sentences in Twitter and Amazon. In: Proceedings of the Fourteenth Conference on Computational Natural Language Learning, CoNLL 2010, Stroudsburg, PA, USA, pp. 107–116. Association for Computational Linguistics (2010)
8. Ghosh, A., et al.: SemEval-2015 Task 11: Sentiment Analysis of Figurative Language in Twitter (2015)
9. Ghosh, A., Veale, T.: Fracking sarcasm using neural network. In: Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, pp. 161–169 (2016)

10. Ghosh, D., Guo, W., Muresan, S.: Sarcastic or not: word embeddings to predict the literal or sarcastic meaning of words. In: EMNLP (2015)
11. González-Ibáñez, R.I., Muresan, S., Wacholder, N.: Identifying sarcasm in Twitter: a closer look. In: ACL (2011)
12. Hee, C.V., Lefever, E., Hoste, V.: SemEval-2018 task 3: irony detection in English Tweets. In: SemEval@NAACL-HLT (2018)
13. Huang, Y.-H., Huang, H.-H., Chen, H.-H.: Irony detection with attentive recurrent neural networks. In: Jose, J.M., et al. (eds.) ECIR 2017. LNCS, vol. 10193, pp. 534–540. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-56608-5\\_45](https://doi.org/10.1007/978-3-319-56608-5_45)
14. Hutto, C.J., Gilbert, E.: VADER: a parsimonious rule-based model for sentiment analysis of social media text. In: ICWSM (2014)
15. Kumar, L., Somani, A., Bhattacharyya, P.: “Having 2 hours to write a paper is fun!”: Detecting Sarcasm in Numerical Portions of Text. arXiv e-prints, September 2017
16. Mikolov, T., Yih, W.T., Zweig, G.: Linguistic regularities in continuous space word representations. In: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 746–751 (2013)
17. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: EMNLP, vol. 14, pp. 1532–1543 (2014)
18. Rajadesigan, A., Zafarani, R., Liu, H.: Sarcasm detection on Twitter: a behavioral modeling approach. In: WSDM (2015)
19. Reyes, A., Rosso, P., Buscaldi, D.: From humor recognition to irony detection: the figurative language of social media. Data Knowl. Eng. **74**, 1–12 (2012)
20. Reyes, A., Rosso, P., Veale, T.: A multidimensional approach for detecting irony in Twitter. Lang. Resour. Eval. **47**(1), 239–268 (2013)
21. Riloff, E., Qadir, A., Surve, P., De Silva, L., Gilbert, N., Huang, R.: Sarcasm as contrast between a positive sentiment and negative situation. In: EMNLP 2013–2013 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, pp. 704–714. Association for Computational Linguistics (ACL) (2013)
22. Rosenthal, S., Ritter, A., Nakov, P., Stoyanov, V.: SemEval-2014 Task 9: Sentiment Analysis in Twitter, January 2014. <https://doi.org/10.3115/v1/S14-2009>
23. Staiano, J., Guerini, M.: DepecheMood: a Lexicon for Emotion Analysis from Crowd-Annotated News. arXiv e-prints, May 2014
24. Gibbs, W.: R.: On the psycholinguistics of sarcasm, vol. 115, March 1986. <https://doi.org/10.1037/0096-3445.115.1.3>
25. Wallace, B.C., Choe, D.K., Charniak, E.: Sparse, contextually informed models for irony detection: exploiting user communities, entities and sentiment. In: ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics (ACL), Proceedings of the Conference, vol. 1 (2015)
26. Özdemir, C., Bergler, S.: CLaC-SentiPipe: SemEval2015 subtasks 10 B, E, and task 11. In: Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), Denver, Colorado, pp. 479–485. Association for Computational Linguistics, June 2015



# Enhanced Feature Selection for Facial Expression Recognition Systems with Genetic Algorithms

Kennedy Chengeta<sup>(✉)</sup>

Department of Mathematics, Statistics and Computer Science,  
University of KwaZulu Natal, Westville Campus, Durban, South Africa  
[216073421@ukzn.ac.za](mailto:216073421@ukzn.ac.za)  
<http://kaributechs.com>

**Abstract.** Humans use voice and facial expressions to infer their state of emotions. Key expressions include being happy, angry, sad and neutral. The expressions accounts for a third of non-verbal communication. The study presents an efficient method to identify facial expressions in images based on artificial neural networks enhanced by genetic algorithms. We use Viola Jones for facial detections and PCA, a statistical method to reduce the dimensionality and extract the features with a local variant of local binary patterns called CS-LBP for static images which is a local algorithm that reduces feature set by comparing symmetrical pixels halving the feature set. The features are then optimally selected using genetic algorithm before classification using artificial neural networks. It is also crucial to note that with these emotions being natural reactions, recognition of feature selection and edge detection from the images can increase accuracy and reduce the error rate. This can be achieved by removing unimportant information from the facial images. The genetic algorithm (GA) chooses a subset of image features based on a reduced-dimensional dataset. The study proposes local binary pattern variant central symmetric local directional pattern (CS-LDP), central symmetric LBP (CS-LBP) and artificial neural networks aided by genetic algorithms for feature selection. The study used the Japanese Female Facial Expression, JAFFE database. The approach outperformed other traditional approaches and proved that with added feature selection and optimization the processing time is reduced and accuracy improved.

**Keywords:** Local Directional and Binary patterns ·  
Genetic algorithms

## 1 Introduction

Emotion recognition with facial expressions is used in child therapy, patients with autism, marketing product responsiveness, security, internet gaming, accounting and educational activities [4, 5, 11, 20, 23]. The key expressions include, sadness, joy, neutral, surprise, fear and anger [2, 18]. FER is also widely used digital identification and surveillance and remote access control systems [5, 6, 14]. Measuring

of such expression includes feature extraction using computing techniques like geometric, component or feature based as well as deep learning approaches. Local binary patterns and local directional patterns, both local algorithms have been used successfully to identify all the different expressions [3, 9, 14, 15] achieving great accuracy [5].

The study uses a variant of the local based approach algorithms to do feature extraction. The FER process involves facial expression detection, preprocessing, feature extraction, feature selection and feature classification. The study aims to improve the performance by using genetic algorithms for feature selection and a combined classifier of neural networks and support vector machines. The preprocessing is done using Gabor filters and histogram equalization [16]. The study uses genetic algorithm and PCA to reduce the feature set as well a combined classifier of neural networks and support vector machines in 3 steps. In the first step, pre-processing involves use of histogram equalization and edge detection. In the second step, features are extracted with a local binary pattern variant the central symmetric local binary pattern (CS-LBP) algorithm. We use PCA, a statistical method to reduce the dimensionality. Optimized feature selection is done using genetic algorithm and the emotions are classified with support vector machines and a neural networks weighted classifier.

## 2 Literature Review

Facial expression recognition has been achieved using local feature extraction algorithms and holistic algorithms like LBP, LDP and Gray Level Co-occurrence Matrix (GLCM) algorithms [2, 4, 14, 21, 24]. Feature selection and reduction has been applied using PCA as well as genetic algorithm though less studies have been applied in FER on the latter [22]. Preprocessing uses gray level transformations, normalization and histogram equalization has been widely researched [17, 20, 22]. The classification of facial expressions using neural networks and support vector machines has also been widely proven though a combination of the two has been less applied in the field [14, 23]. The section reviews the preprocessing and feature extraction stages of the FER process.

### 2.1 Gray Level Transformations, Normalization and Histogram Equalization

A histogram of a facial image is denoted by the frequency of image gray scale levels ranging from 0 to L-1 based on discrete function  $p$  of  $r,k$  for the  $k$ th gray level,  $n_k$  is equal to image pixels and  $N$  is the sum total of pixels and  $k$  takes values from 0 to  $L-1$  [27, 28]. Histogram normalization is used in facial expression recognition to alter the pixel intensity range. Normalization or contrast/histogram stretching is used to find the difference between maximum and minimum pixel intensity [17, 20, 22, 28]. The facial images are normalized into standard ranges between 0 and 255 to a zero mean and variance of 1.

$$g(x, y) = \frac{f(x, y) - f_{min}}{f - max - f_{min}} * 2^{bpp} \quad g(x, y) = \frac{f(x, y) - 0}{255 - 0} * 255 \quad (1)$$

The normalization reduces light in too bright areas of the facial image as shown in Eq. 1 where  $f(x, y)$  is equal to the value of each pixel intensity. Histogram equalization improves the image's global contrast by adjusting the facial image intensity by calculation of the cumulative distributive function [13,28]. Equalization is done to map the image distribution from a source histogram to a new histogram which has a much wider and uniformly distributed intensity values. The gray level image pixels are transformed by intensity values into a uniformly distributed histogram. Gray scale normalization will compensate the effects of color and light [22,28]. The hue image is shown as

$$H = \cos^{-1} \left( \frac{\frac{1}{2}[(x - q) + (x - y)]}{(x - q)(x - q) + (x - y)(q - y)} \right)^2 \quad (2)$$

This Gamma Intensity Correction (GIC) transformer is used to adjust the overall image brightness. RGB normalization involves all pixels being scaled by a given factor and subdividing with the total of 3 color components to elminate color effects [22,28].

$$(x_{norm}, q_{norm}, y_{norm}) = \left( \frac{x}{x + q + y}, \frac{q}{x + q + y}, \frac{y}{x + q + y} \right) \quad (3)$$

## 2.2 Feature Extraction with LBP Variants

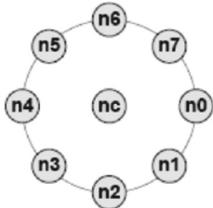
Different LBP variants were successfully used in facial expression recognition that include, TLBP or Ternary Local Binary Patterns, Over-Complete Local Binary Patterns (OCLBP) and ELBP or elliptical local binary patterns and rotational local binary patterns [9,16,17]. The basic LBP is based on a central pixel value and non-center pixels differentials and then taking the binary values of 0 or 1 only [2,5,6,17]. The local binary LBP M, V operator is represented mathematically as follows:

$$LBP(M, V)(c_y, d_y) = \sum_{K=1}^{x=0} q(p_y - p_c)2^x. \quad (4)$$

The neighborhood is depicted as an m-bit binary string leading to x unique values for the local binary pattern code [17,19,20,22].

**Central Symmetric Local Binary Patterns (CS-LBP).** The basic local binary pattern is not resilient when flat images are used and the feature vector is long with 256 dimensions. The central symmetric local binary pattern combines advantages of SIFT descriptors, and LBP [17,19,20,22] (Fig. 1).

$$\begin{aligned} LBP &= k(n_0 - n_c) + k(n_0 - n_c)2 + k(n_0 - n_c)4 + k(n_0 - n_c)8 \\ &\quad + k(n_0 - n_c)16 + k(n_0 - n_c)32 + k(n_0 - n_c)64 + k(n_0 - n_c)128 \\ CS\_LBP &= k(n_0 - n_4) + k(n_1 - n_5)2 + k(n_3 - n_6)4 + k(n_4 - n_7)8 \end{aligned}$$



**Fig. 1.** CS LBP

$$CS - LBP_{r,n} = \sum_{i=0}^{n-1} s(z_i - z_{i+(n/2)})2^i \quad (5)$$

where  $z_i$  and  $z_{i+(n/2)}$  are the values of neighborhood pixels [19]. The algorithm rather than comparing the gray levels of the pixels to the center pixels compares the symmetric pairs of the pixels. The gray level variances between the pixels symmetrically opposite means that the number of patterns produced is much less.

### 2.3 Feature Selection and Optimization with Genetic Algorithms

Genetic algorithms and neural networks have been used in facial expression recognition with great success [6, 17, 25]. The genetic algorithms are classified as a class of evolutionary algorithms based on Darwin's theory of evolution that resolve search and optimization problems [25]. They have been used with success in search problems and optimization scenarios in FER systems (facial expression recognition) [25, 26]. The algorithm reduces processing time and also selects the best features from a sample set [6, 25]. Genetic algorithms a form of adaptive heuristic algorithm based on a genetic analogy and chromosome behavior in defined populations [25]. They select the fittest features in an image set for faster performance and remove unwanted features to increase accuracy. The algorithm use past information to focus the search into a region likely to give better performance. It uses mutation, selection and crossover as a means of population regeneration [25, 26]. The algorithm begins with a set of facial expression images and these are evolved into a generation of fittest images selected for the next selection. This is to resolve problems in facial expression recognition where images are taken in different conditions of light and angle as well as noise [25, 26]. The chromosome represents the features. Another set of evolutionary algorithm includes sequential search algorithms where features are added or removed from using sequential backward search methods (Fig. 2).

Parent 1	1 1 0 1   0 0 1 0 1   1 0 1 1 1 0 0
Parent 2	0 0 1 1   1 0 1 1 0   1 0 0 1 0 1 1
Child	1 1 0 1 1 0 1 1 0 1 0 1 1 1 0 0

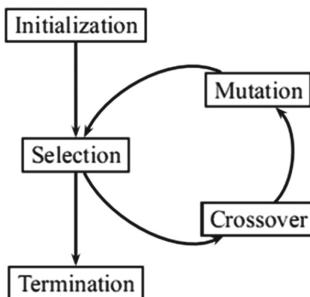
**Fig. 2.** Genetic algorithm mutation

The fitness function optimizes the objective function. The population selected once its satisfies the fitness, undergoes mutation or crossover for them to be selected for next iteration [1, 22, 25]. This string is analogous to the chromosome. The genetic algorithm ensures faster processing due to a reduced feature and search space. The algorithm is based on the given flow below [26]:

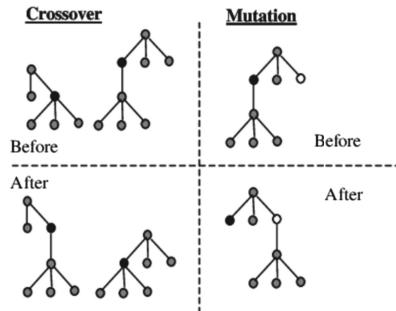
## Genetic Algorithm

1. Selection of the base population of facial expression images is done
2. The fitness of each image in the image set population is calculated
3. The process is repeated till complete based on time, fitness and other factors
4. The best facial expression images are chosen for reproduction
5. The new individuals through crossover and mutation then produce offspring
6. The fitness of the new features is determined
7. The least fit features are then replaced by new individuals in the population

**Selection, Mutation, Fitness Function and CrossOver.** The selection operator selects the fittest offspring likely to survive into the future generation. Crossover includes mixing at least 2 parents to generate a different offspring [25, 26]. The crossover involves binary digit crossovers of the parents.



**Fig. 3.** Genetic algorithm flow [25]

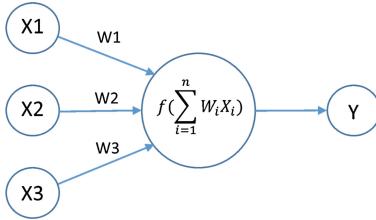


**Fig. 4.** Mutation and Cross over [26]

Mutation is used to preserve the population diversity. The fitness function is used to determine the fitness level for the given chromosomes [26]. The function measures the difference between the facial and base images. The chromosomes selected are recombined using crossover and mutation as shown in Figs. 3 and 4 [1, 22, 25, 26].

## 2.4 Classification with ANNs or Artificial Neural Networks

ANNs are derived from biological neural networks based on a biological neuron comprised of cell bodies, axons (connected via synapses) as well as dendrites and have been widely applied in facial expression [6, 10, 12]. Data is transmitted from one neuron to another using electrical signals [1, 6]. The layers include the input, hidden and output layers. The nodes are interconnected to one another where the input layer nodes are followed by the hidden layer [10, 11]. Key examples include Feedforward Neural Networks and Back Propagation Neural Networks (BPNN) are algorithms inspired by biological neural networks [1, 6]. The neural network is tuned for number of layers, number of nodes as well the learning rates.



**Fig. 5.** Artificial neural network

The activation rule of the perceptron is [1, 6] given an  $n$ -dimensional input  $\mathbf{x} = (x_1, \dots, x_n)$ , the weighted sum of each dimension of the input  $x_i$  and its associated weight  $w_i$  [6] is computed as

$$z = \sum_{i=1}^n (w_i \cdot x_i) + b, \quad (6)$$

where the weight  $w_i$  is often referred to as *preactivation* and to simplify the notation the bias term  $b$  will often be replaced by an equivalent input term  $x_0 = 1$  weighted by  $w_0 = b$ . This is classified as the dot product of a given weight vector  $\mathbf{w}_x = (w_1, \dots, w_n)$  with input vector  $\mathbf{x}$  [6, 17, 22]. The result of this first affine transformation is then passed through a step function as shown in Eq. (7) and Fig. 5.

$$\text{weight} = \text{weight} + \text{learning\_rate} * (\text{expected} - \text{predicted}) * x \quad (7)$$

**Feedforward Neural Networks/Multilayer Perceptron (MLP).** These are simple forms of Artificial Neural Networks comprised of 3 types of layers namely input, hidden as well as the output layer [1, 6]. The neural network is a fully-connected feedforward network with at least one hidden layer followed by a nonlinearity  $\sigma$  more smoother perceptron defined as *activation function* [6]

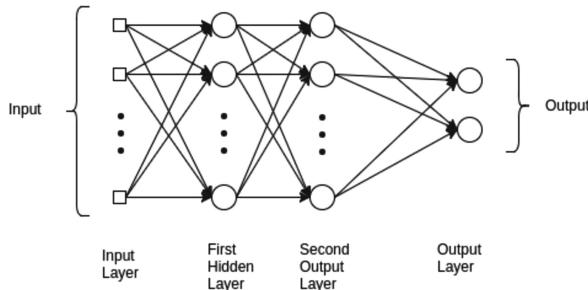
$$\mathbf{z}^{(l)} = \mathbf{W}^{(l)} \cdot \mathbf{a}^{(l-1)}, \quad \mathbf{a}^{(l)} = \sigma(\mathbf{z}^{(l)}). \quad (8)$$

Each layer computes an affine transformation. An MLP (form of Artificial Neural Network) is a class of feedforward artificial neural network. MLPs with one hidden layer act as a continuous function [1, 6] (Fig. 6).

**Backpropagation.** The neural network addresses problems that the computations on the activation functions are non linear by allowing partial derivatives of the errors. The last layer's contribution to errors is determined and consequent splits propagate backwards the errors assigning a weight to each blame [1, 6]. An optimization algorithm is then used to alter the weights to minimize the given errors. Backpropagation with mean square error/MSE with output  $y$  is used to generate the cost using the following metric where the scalar is  $y$  and vector  $y$  with 3 layers of affine transformations, The summation is done of a dataset  $X$  with  $M$  samples given the expected output  $y$  [1, 6]. The algorithm is based on calculating the gradient descent search in a given weight space. The error fraction is based on each weight as shown in Eqs. 9 and 10.

$$E_{mse} = \frac{1}{M} \sum_{\mathcal{D}} \frac{1}{2} \|y - \hat{y}\|_2, \quad (9)$$

$$\log(\hat{y}_j) = \log \left( \frac{e^{z_j}}{\sum_{i=1}^n e^{z_i}} \right) = \log(e^{z_j}) - \log \left( \sum_{i=1}^n e^{z_i} \right) = z_j - \log \left( \sum_{i=1}^n e^{z_i} \right) \quad (10)$$



**Fig. 6.** Feedforward neural networks/Multilayer perceptron (MLP) [6, 17]

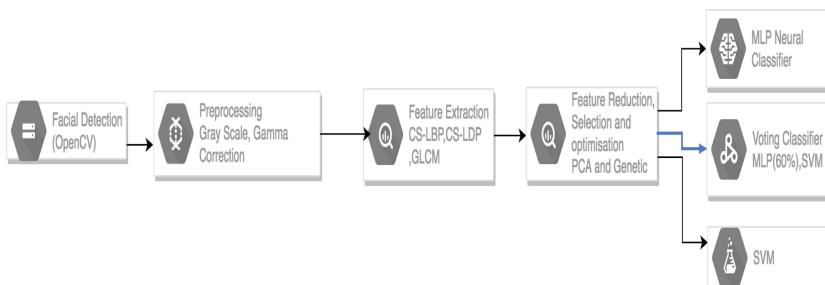
The middle layers use a tanh/sigmoid activation function and the softmax top layer is based on a ‘LogisticRegression’ class). Multilayer perceptrons are based on adjusting weights and biases during training to reduce errors [1, 6]. On the other hand, backpropagation links the weights and bias to the error using y root mean squared error (RMSE). *Deep Belief Network (DBN)* algorithms have been used in facial expression recognition through a 2-layer DBN architecture based on a stack of restricted Boltzmann machines (RBMs) [1, 6].

### 3 Implementation

The facial expression recognition system includes detection of the facial images, preprocessing, feature extraction, feature selection using genetic algorithm and lastly classification of the given feature vector histograms to generate the emotions happy, sad, angry, surprise as well as fear. The implementation is shown in the next diagram from facial detection, preprocessing, feature extraction and selection or reduction to classification.

**Facial Expression Detection.** For our training images, we use the Viola-Jones Haar Cascade Method implemented in OpenCV.

```
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')
```



The *Viola-Jones mouth detector* was used to detect the mouth features and uses the mouth position against the nose and eyes. The *single eye detection* uses 2 triangles to detect both eyes as left and right. The *nose detection tool* is used to detect the nose [7, 13, 17].

### 3.1 Pre-processing

Preprocessing involved the application of histogram equalization to enhance the image quality. The images were also converted to gray scale. The effects of angle, distance, lighting were also eliminated by using the Gamma intensity correction and logarithmic transformation on the given images. The RGB to Gray function is implemented using a python Matlab function. The color information is discarded to improve speed and there is no loss of accuracy. The images are transformed from the RGB to grey level space using the following *RGB to GRAY* [7, 13, 17]:

$$Y = 0,3R + 0,59G + 0,11B \quad (11)$$

### 3.2 Facial Expression Databases

The dataset used is the JAFFE datasets. This dataset has 213 images with 7 facial expressions made up of six basic and the neutral expression taken from 10 models of Japanese descent and the emotions include the following  
['angry', 'fear', 'disgust', 'surprise', 'neutral', 'joy', 'sadness']

### 3.3 Feature Extraction

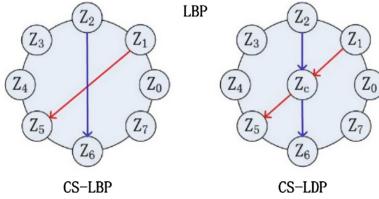
The feature extraction was done with LBP, Central Symmetric LBP [5]. These local extractors were also compared with a holistic algorithm namely GLCM. The feature selection involved using the genetic algorithm to select the healthier feature vectors. The other variant considered was the local directional pattern where the histogram was represented as below [9].

$$LDP_h(\sigma) = \sum_{M}^{r=0} \sum_{N}^{r=0} f(LDP_k(r, c), \sigma). \quad (12)$$

The central symmetric LBP is shown in the diagrammatic form below [16, 17] (Fig. 7).

**Dimensional Reduction and Feature Selection** are done using principal component analysis (PCA) and genetic algorithms. For a PCA the optimal hyperplane is shown as below [1, 5, 22] for mean  $m$ , the training sample is computed as The covariance matrix  $X$  based on given training samples is

$$X = \sum_{k=1}^n (x_k - m) (x_k - m)^T \quad (13)$$



**Fig. 7.** Center-symmetric local binary pattern sequence [19]

The genetic algorithm selected feature vectors based on 15%, 18%, 25%, 28% and 30% and 35% of the total feature vectors selecting the most fittest feature vectors. The best accuracy came when the ratio of 0.28 was used hence the classification was done on the basis of 28% of the total feature vectors. for scenarios with PCA alone, 66.66% of top-most eigenvectors are selected.

### 3.4 Classification

Various classification methods were used namely, support vector machines, weighted voting classifier of support vector machines and neural networks. Cross validation was also applied to ensure there is reduced overfitting [6, 7, 13, 17]. The classification was implemented using python scipy, jupyter and numpy packages on a Mac OS operating system.

$$eclf = \text{voting\_classifier}(\text{clfs} = [\text{clf}_a, \text{clf}_b], \text{weights} = [0.6, 0.4]) \quad (14)$$

**Support Vector Machine Classifiers** were based on a hyperplane to split dual classes and were represented by the equation below [1, 13, 15]:

$$v^T m_q + k = 0, \quad v^T, m_q + k1 \quad y_q = +1, \quad (15)$$

The SVM classifier uses the RBF kernel for classification and the *Multilayer Perceptron (MLP)* was implemented with the following where soft voting classifier was used

```
MLPClassifier(activation = 'relu', alpha = 0.0001,
               batch_size = 'auto', beta_1 = 0.9,
               beta_2 = 0.999, early_stopping = False, epsilon = 1e-08,
               hidden_layer_sizes = (30, 30, 30), learning_rate = 'constant',
               learning_rate_init = 0.001, max_iter = 200, momentum = 0.9,
               nesterovs_momentum = True, power_t = 0.5, random_state = None,
               shuffle = True, solver = 'adam', tol = 0.0001, validation_fraction = 0.1,
               eclf = VotingClassifier(estimators
               = [('mlp', clf1), ('svm', clf2)], voting = 'hard', weights = [40, 60])
```

## 4 Facial Recognition and Expression Results

The experiment used various algorithms namely, basic LBP, local direction pattern or LDP, GLCM a holistic feature expression extractor, CS-LBP and CSLBP with genetic algorithm. The study ran experiments with the above algorithms on the JAFFE dataset. The classification used involved running against a support vector machine alone, artificial neural network alone and a weighted classifier of 60% of neural networks and 40% support vector machines.

**Table 1.** Facial expression classification accuracy on 213 JAFFE images

Algorithm selected	SVM(RBF)	MLP	SVM(RBF)+MLP
CS-LBP	0.912	0.932	0.942
LBP	0.899	0.902	0.931
LDP	0.902	0.932	0.944
HOG	0.79	0.821	0.859
GLCM	0.847	0.842	0.869
CS-LBP, Genetic	0.932	0.934	0.966
CS-LBP, PCA	0.911	0.928	0.951
CS-LDP, Genetic	0.904	0.935	0.989

Table 1 shows the accuracy levels of the SVM, MLP, and weighted classifier of 60:40 ratio of support vector machine and MLP neural network. The weighted classifier showed better classification accuracy than the base algorithms alone. The CS-LBP algorithm and CS-LDP algorithms showed greater accuracy than the base LBP and LDP algorithms. When combined with the genetic algorithm for feature optimization and selection the accuracy levels was higher than the CS-LBP alone and GLCM a holistic algorithm. The results for the Genetic algorithm with CS-LBP were based on a 28% reduced feature selection using the genetic algorithm and recorded higher classification results than CS-LBP alone due to the reduced set of features and removal of noisy features. The CS-LDP algorithm with genetic algorithm recorded the highest accuracy due to advantages of having an filtering edge detector known as kirsch filter as part of the algorithm with a classifier of 0.989. The processing time on the scenarios where genetic algorithms were used for CS-LBP and CS-LDP all recorded processing times between 15–20% faster than the traditional LBP and CS-LBP alone.

## 5 Conclusion

The study proposed a hybrid approach of Central Symmetric Local Binary Patterns, a variant of local binary patterns, genetic algorithm to enhance feature selection and a weighted classifier of artificial neural networks and support vector

machines. The different emotions measured namely fear, disgust, anger, happiness and sadness are recognized with better accuracy than basic local binary patterns or using just support vector machines or neural networks as the classifiers. The databases trained included the Japanese Female Facial Expression (JAFFE) database. The accuracy of almost 97% was achieved on small datasets which is better than the traditional algorithms as shown in Table 1. The use of genetic algorithm and central symmetric LBP which both reduce the feature set by selecting the healthy feature vectors and using symmetrical differences of pixels respectively improves accuracy of facial expression recognition and reduces processing time. The Central Symmetric Local Directional Pattern combined with genetic algorithm also adds the advantage of filtering unwanted edges through its kirsch edge detector.

## References

1. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, San Francisco (2001)
2. Aung, M.S., et al.: The automatic detection of chronic pain-related expression: requirements, challenges and the multimodal EmoPain dataset (2015)
3. Pavithra, P., Ganesh, A.B.: Detection of human facial behavioral expression using image processing. *ICTACT J. Image Video Process.* **1**, 162–165 (2011)
4. Nurzynska, K., Smolka, B.: Smiling and neutral facial display recognition with the local binary patterns operator. *J. Med. Imaging Health Inf.* **5**(6), 1374–1382 (2015)
5. Calder, A.J., Burton, A.M., Miller, P., Young, A.W., Akamatsu, S.: A principal component analysis of facial expressions. *Vis. Res.* **41**(9), 1179–1208 (2001)
6. Padgett, C., Cottrell, G.W.: Representing face images for emotion classification. In: *Advances in Neural Information Processing Systems*, pp. 894–900 (1997)
7. Viola, P., Jones, M.J.: Robust real-time face detection. *Int. J. Comput. Vision* **57**, 137–154 (2004)
8. Zhao, X., Zhang, S.: Facial expression recognition based on local binary patterns and kernel discriminant isomap. *Sensors* **11**(10), 9573–9588 (2011)
9. Rivera, A.R., Castillo, R., Chae, O.: Local directional number pattern for face analysis: face and expression recognition. *IEEE Trans. Image Process.* **22**, 1740–1752 (2013)
10. Alizadeh, S., Fazel, A.: Convolutional Neural Networks for Facial Expression Recognition [arXiv:1704.06756](https://arxiv.org/abs/1704.06756), June 2017
11. Chen, L., Xi, M.: Local binary pattern network: a deep learning approach for face recognition. In: *2016 IEEE International Conference on Image Processing (ICIP)* (2016)
12. Uddin, M.Z., Khaksar, W., Torresen, J.: Facial expression recognition using salient features and convolutional neural network. *IEEE Access* **5**, 26146–26161 (2017). <https://doi.org/10.1109/ACCESS.2017.2777003>
13. Viola, P., Jones, M.J.: Robust real-time face detection. *Int. J. Comput. Vis.* **57**, 137–154 (2004)
14. Valenti, R., Sebe, N., Gevers, T.: Facial expression recognition: a fully integrated approach. In: *14th International Conference of Image Analysis and Processing, Modena*, pp. 125–130 (2007). <https://doi.org/10.1109/ICIAPW.2007.25>

15. Mattivi, R., Shao, L.: Human action recognition using LBP-TOP as sparse spatio-temporal feature descriptor. In: Jiang, X., Petkov, N. (eds.) CAIP 2009. LNCS, vol. 5702, pp. 740–747. Springer, Heidelberg (2009). [https://doi.org/10.1007/978-3-642-03767-2\\_90](https://doi.org/10.1007/978-3-642-03767-2_90)
16. Ravi Kumar, Y.B., Ravi Kumar, C.N.: Local binary pattern: an improved LBP to extract nonuniform LBP patterns with Gabor filter to increase the rate of face similarity. In: ICCCCIP, Mysore, pp. 1–5 (2016)
17. Pietikinen, M., Hadid, A., Zhao, G., Ahonen, T.: Computer Vision Using Local Binary Patterns. Springer, London (2011). <https://doi.org/10.1007/978-0-85729-748-8>
18. Ekman, P., Friesen, W.V.: The repertoire of nonverbal behavior: categories, origins, usage, and coding. *Semiotica* **1**, 49–98 (1969)
19. Rami, H., Hamri, M., Masmoudi, L.: Objects tracking in images sequence using center-symmetric local binary pattern (CS-LBP). *Int. J. Comput. Appl. Technol. Res.* **2**(5), 504–508 (2013)
20. Nakashima, Y., Kuroki, Y.: SIFT feature point selection by using image segmentation. In: International Symposium on Intelligent Signal Processing and Communication Systems, Xiamen, pp. 275–280 (2017)
21. Thakare, V.S., Patil, N.N.: Classification of texture using gray level co-occurrence matrix and self-organizing map. 2014 International Conference on Electronic Systems. Signal Processing and Computing Technologies, pp. 350–355. IEEE, Washington (2014)
22. Theodoridis, S., Theodoridis, S., Koutroumbas, K.: Pattern Recognition, 4th edn (2008). ISBN 9781597492720, 9780080949123
23. Dikkers, H., Spaans, M., Datcu, D., Novak, M., Rothkrantz, L.: Facial recognition system for driver vigilance monitoring. In: 2004 IEEE International Conference on Systems, Man and Cybernetics, vol. 4, pp. 3787–3792 (2004)
24. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Washington, DC, USA (2005)
25. Boubenna, H., Lee, D.: Feature selection for facial emotion recognition based on genetic algorithm. In: 2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery, Changsha, pp. 511–517 (2016)
26. Satone, M., Kharate, G.: Feature selection using genetic algorithm for face recognition based on PCA, wavelet and SVM. *Int. J. Electr. Eng. Inf.* **6**(1), 39 (2014)
27. Nafchi, H.Z., Ayatollahi, S.M.: A set of criteria for face detection preprocessing. *Procedia Comput. Sci.* **13**, 162–170 (2012)
28. Han, H., Shan, S., Qing, L., Chen, X., Gao, W.: Lighting aware preprocessing for face recognition across varying illumination. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6312, pp. 308–321. Springer, Heidelberg (2010). [https://doi.org/10.1007/978-3-642-15552-9\\_23](https://doi.org/10.1007/978-3-642-15552-9_23)



# Imaging Time-Series for NILM

Lamprini Kyrkou, Christoforos Nalmpantis<sup>(✉)</sup>, and Dimitris Vrakas

School of Informatics, Aristotle University of Thessaloniki,  
54124 Thessaloniki, Greece

{lakyrkou, christofn, dvrakas}@csd.auth.gr

<https://www.csd.auth.gr/en/>

**Abstract.** Non Intrusive Load Monitoring is the field that encompasses energy disaggregation and appliance detection. In recent years, Deep Neural Networks have improved the classification performance, using the standard data representation that most datasets provide; that being low-frequency or high-frequency data. In this paper, we explore the NILM problem from the scope of transfer learning. We propose a way of changing the feature space with the use of an image representation of the low-frequency data from UK-Dale and REDD datasets and the pretrained Convolutional Neural Network VGG16. We then train some basic classifiers and use the metric F1 score to test the performance of this representation. Multiple tests are performed to test the adaptability of the models to unseen houses and different datasets. We find that the performance is on par and in some cases outperforms that of popular deep NN algorithms.

**Keywords:** NILM · Energy disaggregation · Transfer learning · Artificial neural networks

## 1 Introduction

Energy demands have risen greatly in the past 40 years. More and more electrical devices are becoming essential in a household; computers, tablets, cell phones etc. That, of course, means that more energy is being spent. Therefore a need arises for efficient monitoring of the energy being consumed. With the progress of technology, it has become possible to monitor the energy consumption of a house with the use of smart meters. The idea is to apply a smart meter in each appliance of the house and have real-time information about energy consumption. The application of smart meters on the appliance level is still quite costly and therefore other ways have to be explored.

A more cost-efficient way of monitoring power consumption of a house would be to have to install only one meter per household, which will monitor the total

---

This work has been funded by the ΕΣΠΑ (2014–2020) Erevno-Dimiotorgo-Kainotomo 2018/EPAnEK Program ‘Energy Controlling Voice Enabled Intelligent Smart Home Ecosystem’, General Secretariat for Research and Technology, Ministry of Education, Research and Religious Affairs.

energy being consumed. The real world application of this is possible with the help of algorithms that are able to infer the total energy signal to the device level sub-signals that compose it. This is also known as an energy disaggregation task and the field that studies it is Non-Intrusive Load Monitoring (NILM).

The most popular NILM approaches in machine learning are producing models whose input for the training is high or low-frequency data of the mains of a house and the labels are the true energy consumption values of a chosen appliance. Usually, both the training data and labels are chronologically arranged and one of the goals of the chosen model is to find correlations between the aggregated signal and the appliance signal, given a certain time frame. In NILM, appliances within a household often change and get replaced by others as time passes. Therefore, there are many occasions where a model trained to classify an appliance in a certain time frame of a house cannot predict it accurately on a different time frame. That being said, it is worthwhile investigating whether the application of a transfer learning technique can be used to classify electrical appliances (ON/OFF state) given the aggregated power signal of a house.

Transfer learning [3, 15] does not rely on training and test data being in the same feature space or even having the same distribution, in contrast to the more classical approaches of machine learning techniques. It can be beneficial in problems where there is a shortage of data, such as an accurate classifier or regressor is not able to be trained, or there is a need for a more general model; a model that generalizes in different distributions. There are several ways transfer learning can be achieved. In this work, we focus on feature representation. The idea is to encode the existing knowledge to another feature space. E.g, encode a time series to an image representation. This approach, to our knowledge, has not been previously explored in the existing literature concerning NILM problems. It has been however applied to other time series classification tasks and has been proven to yield good results.

In this study, the idea involves transformation of low-frequency data (1 Hz data), from popular datasets UK-Dale [9] and REDD [11] to images using Wang and Oates' [20] time series to image algorithm, which uses Gramian Angular Field Matrices (GAF), and afterwards changing the feature space with the help of the pretrained Convolutional Neural Network VGG16 image classification algorithm [19] to vectors. The final step is, feeding those vectors, accompanied with their respective device labels, to a classification algorithm, such as a decision tree algorithm, and training it to be able to recognize whether the appliance is ON or OFF.

In the following sections, the process of the feature transformation and transference is analyzed, as well as a set of experiments for the appliance “fridge” of the UK-Dale and REDD datasets is compared directly to previous implementations.

## 2 Related Work

Hart [6] was the first to work on this problem and used combinatorial techniques in order to monitor changes on the appliances states of a household given the aggregated signal. Since then there have been many approaches to the problem. Factorial Hidden Markov Models (FHMM) solutions [1, 10, 16, 17] have been the leading implementations the past two decades, while Deep learning Artificial Neural Network (ANN) architectures have become popular in the last decade [2, 8, 12, 13, 21]. Nalmpantis and Vrakas [14] in their review describe some of the most important recent Machine Learning approaches for the problem of NILM. The approaches are compared in detail, presenting a qualitative and quantitative analysis.

De Baets et al. [4] at their work represent the  $V_i$  trajectory of appliances as images and train a Siamese Neural network from which a new feature space is derived. This new representation is the input of the DBSCAN algorithm that ultimately is able to recognize appliances in a household that are left unlabeled. They use the high-frequency data from the datasets PLAID [5] and WHITED [7]. Their approach seems to be successful in recognizing unknown appliances in a household.

Wang et al. [20] propose a novel way of transforming time series into images, in order to test whether this new representation of the data improves classification results. The images are constructed by transforming a time series to its polar coordinate representation. For this the time series used are scaled either to the interval  $[-1, 1]$  or  $[0, 1]$ . The rescaled time series  $\tilde{X}$  can then be represented as its polar coordinates. This can be achieved by using the value of the time series and the time stamp to encode as the angular cosine and radius respectively. The equation is defined as:

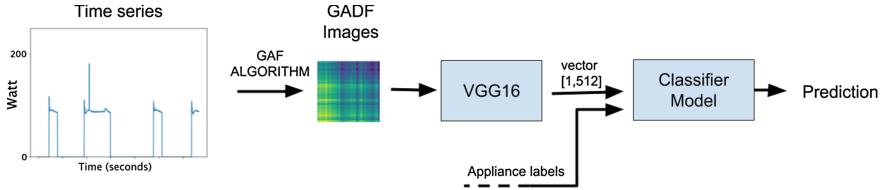
$$\begin{cases} \phi = \arccos(\tilde{x}_i, -1 \leq \tilde{x}_i \leq 1, \tilde{x}_i \in \tilde{X}) \\ r = \frac{t_i}{N}, t_i \in N \end{cases} \quad (1)$$

The next step is constructing a Gramian Angular Summation/Difference Fields (GASF/GADF) matrix which contains the correlation between elements of different timestamps. GASF and GADF are defined as follows:

$$GASF = [\cos(\phi_i + \phi_j)] = \tilde{X}' \cdot \tilde{X} - \sqrt{I - \tilde{X}^2}' \cdot \sqrt{I - \tilde{X}^2} \quad (2)$$

$$GADF = [\cos(\phi_i - \phi_j)] = \sqrt{I - \tilde{X}^2}' \cdot \tilde{X} - \tilde{X}' \cdot \sqrt{I - \tilde{X}^2} \quad (3)$$

In their study they test the above representation on multiple datasets. They use a Tiled Convolutional Neural Network for extracting features. For the classification task they use a Denoising Auto-Encoder. The results are promising as the Mean Squared Error metric is reduced by a maximum of 48% compared to approaches that use raw data.



**Fig. 1.** Proposed training/prediction procedure.

### 3 Implementation

#### 3.1 Data Preprocessing

The process begins with the transformation of the mains aggregated energy signal of a house to multiple vectors of length 64 that will, in turn, become the input of GAF algorithm. In this study only the GADF form is used. The vector size corresponds to 6.4 min of data and is chosen so that the time frame is not considerably large, but at the same time is not too small which would cause substantial delays in most machines when producing the image files. Moreover, the time frame chosen cannot be bigger than 6.4 min because of computational restrictions. The nilmtk framework for Python 2.7 is used in order to preprocess the data and construct the input files for the GAF algorithm. The set sampling rate for both the mains data and the appliance data is set to 6 s. Once the data has been preprocessed the GAF algorithm can begin generating the images. The output of this algorithm are multiple images in .png format and  $100 \times 100$  pixel size. PAA smoothing is applied on them. The images are also tied with same length vectors of labels for the appliance “fridge”, for which the experiments are conducted. The whole procedure can be seen in Fig. 1.

#### 3.2 Data Extraction

The next step concerns the transference to another input space by using the VGG16 pre-trained model in Keras, Python to categorize the images produced by the GAF algorithm in the previous step. VGG16 is an image classification Convolutional Neural Network (CNN) which is trained for 1000 classes of the ImageNet database [18]. It has been proven very successful on image classification tasks. Its input consists of a 3D tensor that represents the image. Because the model has already been trained there is no need to re-train it with our images. Instead, each image passes through the network and its prediction/output is the new data space we wanted. Therefore, each image is inferred to 512 categories vectors. After each image for a selected house and time period has been processed, an array of dimensions [number of images  $\times$  512] is constructed. This will be the input for the training of the classification algorithms that are tested in this study.

### 3.3 Classification

The labels of each image are averaged and simplified to 1 or 0, depending on whether the averaged number surpasses the threshold of the appliance. If the appliance was on an ON state, during the 6.4 min the image represents, the label for it becomes 1. If the appliance was deactivated (and therefore on an OFF state) during the time period of the image, the label for it is 0. Alternatively, one could choose the maximum value of the label vector as the representative point on whether the appliance is in use or not.

For the classification task, the algorithms that were chosen belong to Python's Scikit-learn library. Those are the following: (1) Multi-Layered Perceptron Classifier (MLPC), for which several hidden layer sizes were tested and were concluded to an optimal size of 50. (2) AdaBoost Classifier, with Decision Tree Classifier learners (ABDTC), that each developed to a maximum depth of 2. 1000 estimators were used for the training of this meta-classifier. (3) AdaBoost Classifier (ABC), with its default parameters and 1000 estimators.

Note that some of the above models are trained on UK-Dale's dataset data for house 1, while others for several houses of the REDD dataset. Moreover, the training data and labels are always shuffled before the training of an algorithm begins and a small portion of it is left out as a validation set, while a different set is put aside as the test set. The test set is the one which is used for evaluation of the models, while the validation set is used for fine tuning the models' parameters.

### 3.4 Evaluation Metrics

The metric that is used to evaluate the models is F1 score:

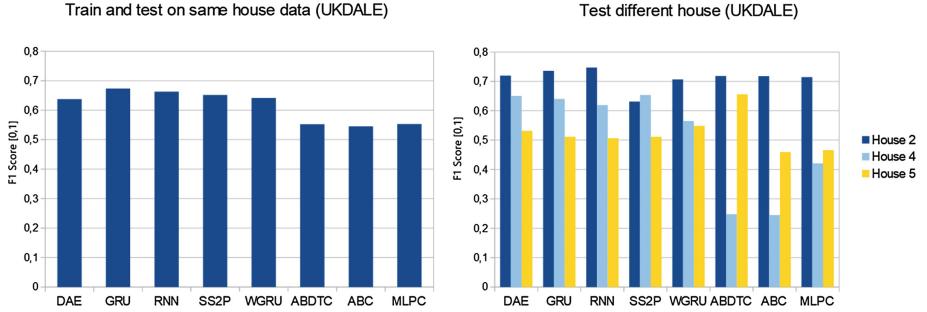
$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

F1 score is better than accuracy for evaluating models as it is a function of precision and recall that calculated a balance point of the two. It is a very useful metric for problems that suffer from class imbalance as it can make that apparent.

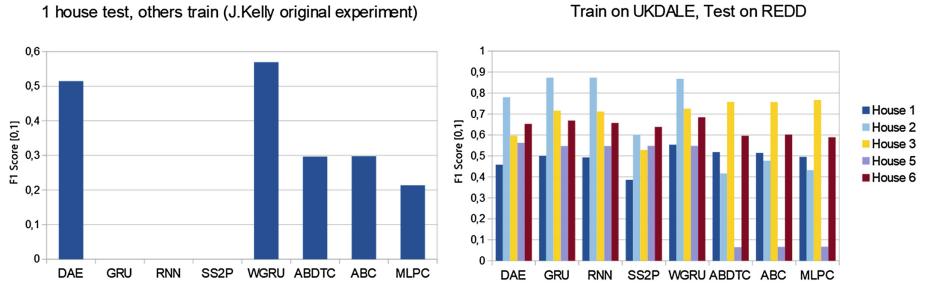
### 3.5 Specifications

All experiments can be reproduced by our implementation code in our GitHub repository: <https://github.com/LampriniKyrk/Imaging-NILM-time-series>. The hardware that is used for all of the model training and testing is an AMD Ryzen 5 1600x processor, an AMD R7 260x GPU and 8 GB ddr4 RAM. Note that the image production takes a significantly long time, even when running in multiple threads, as the hard disk serves as the bottleneck.

On the experiment section, it is defined in which house and time period data each algorithm is trained on and what the test set for each experiment entitled.



**Fig. 2.** The bar chart on the left shows the results for all the models that are trained for the data of house 1 of UK-Dale and tested on data from the same house. The bar chart on the right shows the results for the same models as the left but the data tested belong to the unseen houses 2, 4 and 5 of the UK-Dale dataset.



**Fig. 3.** The bar chart on the left shows the results for all the models that are trained for the data of house 1, 2 and 4 of UK-Dale and tested on data from house 5 of UK-Dale. The bar chart on the right shows the results for the models that are trained for the data of house 1 of UK-Dale dataset and tested for the houses of the REDD dataset.

## 4 Experiments and Results

In this section, we discuss the experiments and results of our novel NILM solution and compare it directly to previous work. The proposed method is compared with some popular ANN architectures as they were implemented in a previous work of Krystalakos et al. [12]. The following networks were used for this task: Gated Recurrent Units Network (GRU) [12], Recurrent Neural Network (RNN) [8], Windowed GRU (WGRU) [12], Short-Sequence2Point (SS2P) [21], and Denoising Auto-Encoder (DAE) [8]. The comparison between the aforementioned ANN architectures and our approach is direct, meaning the training and testing of the models are done for the exact same time periods and houses.

The ANN architectures that were used as a base for the comparison follow. GRU consists of two convolutional layers, followed by two bidirectional GRU and two fully connected layers. RNN consists of one convolutional layer, followed by

two bidirectional LSTM layers and two fully connected layers. WGRU consists of one convolutional layer and two bidirectional GRU layers who are followed by two fully connected layers. The last four layers have dropout between them. SS2P consists of a total of seven layers: five convolutional layers followed by two fully connected layers. All layers of the SS2P architecture have dropout between them. DAE consists of a convolutional layer, three fully connected layers followed by one convolutional output layer. All layers of the DAE architecture have dropout between them.

We train multiple models based on the new feature space on the task of predicting ON/OFF states of the appliance “fridge”. In the following sub-sections, the results are presented, categorized by experiment type.

In the first experiments category, all models are being trained on house 1 of the UK-Dale dataset and the dates 1/4/2013 to 1/4/2014. The test also occurs for the data of the same house, though the time frame is 1/4/2016 to 1/4/2017. As it can be seen in Fig. 2 the proposed approach gives comparable results with those from the ANN models. This experiment uses the F1 metric and evaluates how well the algorithm performs on data of the same house.

The second category of experiments uses the same trained models as above but the test occurs on different houses of the same dataset. In these experiments, we test how well the models generalize on unseen houses of the same dataset. The metric we use for this evaluation is F1 score. Our approach seems to be on par with the comparative models for the houses 2 and 5. Testing on house 5 the AdaBoost classifier with low depth decision trees (ABDTC) seems to outdo all of the ANN models. On house 4 our models are not as accurate as the ANN models, which could be due to the fact that house 1 and house 4 differ greatly based on the number of appliances each house has. House 1 has a total of 54 appliances, while house 4 has only 6.

In the third experiment category, the models are trained with data from multiple houses of the UK-Dale dataset. The houses used for the training data are 1, 2 and 4. From house 1 only the time frame from 1/4/2013 to 1/4/2014 is used for the training, as it has the most data out of all the houses and therefore if all where to be used a memory error could occur. This test evaluates if it is possible for a model to learn from multiple houses and be able to predict accurately on an unseen house. Some ANN models were not able to converge. Our models have a significantly lower F1 score from the best ANN, which is the Windowed GRU model.

The last experiment category tests how well a model trained in UK-Dale generalizes for the data of the REDD dataset. That way, if a model performed well on the unseen houses of the different dataset it is safe to say that it has the potential to be accurate on data regardless of the country of origin. Note that UK-Dale is a UK based dataset while REDD is a US-based dataset. As it is shown in Fig. 3 most models, whether they’re ANN’s or the different input space approaches, perform well on most houses. Our approach is on par with the ANN’s on most houses and on house 3 it outperforms them.

## 5 Conclusions and Future Work

Transferring of NILM data to a different feature space shows promise as the results seem on par with most of the modern NILM approaches. It certainly has room for improvement as the algorithms we use for classification are not state of the art nor are configured in the best way. Furthermore, the accuracy of the approach could potentially be improved if smaller time windows were to be used (e.g 3 min or less), as the problem that arises from bigger time windows is that we don't always get a good estimation of the true appliance state. Moreover, the approach could be explored for regression and multi-label tasks, in addition to the classification approach that is researched in this study. Closing, we believe that continuing this research by combining the new feature space with state of the art algorithms, it is possible to achieve even better results.

## References

1. Aiad, M., Lee, P.H.: Non-intrusive load disaggregation with adaptive estimations of devices main power effects and two-way interactions. *Energy Build.* **130**, 131–139 (2016). <https://doi.org/10.1016/j.enbuild.2016.08.050>. <http://www.sciencedirect.com/science/article/pii/S0378778816307472>
2. Chen, K., Wang, Q., He, Z., Chen, K., Hu, J., He, J.: Convolutional sequence to sequence non-intrusive load monitoring. *J. Eng.* **2018**(17), 1860–1864 (2018). <https://doi.org/10.1049/joe.2018.8352>
3. Dai, W., Chen, Y., Xue, G.R., Yang, Q., Yu, Y.: Translated learning: transfer learning across different feature spaces. In: *Advances in Neural Information Processing Systems*, pp. 353–360 (2009)
4. De Baets, L., Develder, C., Dhaene, T., Deschrijver, D.: Detection of unidentified appliances in non-intrusive load monitoring using siamese neural networks. *Int. J. Electr. Power Energy Syst.* **104**, 645–653 (2019)
5. Gao, J., Giri, S., Kara, E.C., Bergés, M.: PLAID: a public dataset of high-resolution electrical appliance measurements for load identification research: demo abstract. In: *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, pp. 198–199. ACM (2014)
6. Hart, G.W.: Nonintrusive appliance load monitoring. *Proc. IEEE* **80**(12), 1870–1891 (1992)
7. Kahl, M., Haq, A.U., Kriechbaumer, T., Jacobsen, H.A.: Whited-a worldwide household and industry transient energy data set. In: *3rd International Workshop on Non-Intrusive Load Monitoring* (2016)
8. Kelly, J., Knottenbelt, W.: The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes. *Sci. Data* **2**, 150007 (2015). <https://doi.org/10.1038/sdata.2015.7>
9. Kelly, J., Knottenbelt, W.: The UK-dale dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes. *Sci. Data* **2**, 150007 (2015)
10. Kolter, J.Z., Jaakkola, T.: Approximate Inference in Additive Factorial HMMs with Application to Energy Disaggregation, June 2018. <https://doi.org/10.1184/R1/6603563.v1>

11. Kolter, J.Z., Johnson, M.J.: REDD: a public data set for energy disaggregation research. In: Workshop on Data Mining Applications in Sustainability (SIGKDD), San Diego, CA, vol. 25, pp. 59–62 (2011)
12. Krystalakos, O., Nalmpantis, C., Vrakas, D.: Sliding window approach for online energy disaggregation using artificial neural networks. In: Proceedings of the 10th Hellenic Conference on Artificial Intelligence, SETN 2018, pp. 7:1–7:6. ACM, New York (2018). <http://doi.acm.org/10.1145/3200947.3201011>
13. Lange, H., Bergés, M.: The neural energy decoder: energy disaggregation by combining binary subcomponents (2016)
14. Nalmpantis, C., Vrakas, D.: Machine learning approaches for non-intrusive load monitoring: from qualitative to quantitative comparation. Artif. Intell. Rev. 1–27 (2018)
15. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. Knowl. Data Eng. **22**(10), 1345–1359 (2010)
16. Paradiso, F., Paganelli, F., Giuli, D., Capobianco, S.: Context-based energy disaggregation in smart homes. Future Internet **8**(1) (2016). <https://doi.org/10.3390/fi8010004>. <http://www.mdpi.com/1999-5903/8/1/4>
17. Parson, O., Ghosh, S., Weal, M., Rogers, A.: Non-intrusive load monitoring using prior models of general appliance types. In: Twenty-Sixth AAAI Conference on Artificial Intelligence (2012)
18. Russakovsky, O., et al.: Imagenet large scale visual recognition challenge. Int. J. Comput. Vis. **115**(3), 211–252 (2015)
19. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint <arXiv:1409.1556> (2014)
20. Wang, Z., Oates, T.: Imaging time-series to improve classification and imputation. In: Twenty-Fourth International Joint Conference on Artificial Intelligence (2015)
21. Zhang, C., Zhong, M., Wang, Z., Goddard, N., Sutton, C.: Sequence-to-point learning with neural networks for non-intrusive load monitoring. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)



# Learning Meaningful Sentence Embedding Based on Recursive Auto-encoders

Amal Bouraoui<sup>(✉)</sup>, Salma Jamoussi, and Abdelmajid Ben Hamadou

Multimedia InfoRmation Systems and Advanced Computing Laboratory,  
MIRACL-Sfax University, Technopole of Sfax: Av.Tunis Km 10,  
B.P. 242, 3021 Sfax, Tunisia  
[amal.bouraoui@gmail.com](mailto:amal.bouraoui@gmail.com)

**Abstract.** Learning meaningful representations for different granularities of texts is a challenging and on-going area of research in natural language processing. Recently, neural sentence modeling that learns continuous valued vector representations for sentences in a low dimensional latent semantic space has gained increasing attention. In this work, we propose a novel method to learn meaning representation for variable-sized sentence based on recursive auto-encoders. The key difference between our model and others is that we embed the sentence meaning while jointly learning evolved word representation in unsupervised manner and without using any parse or dependency tree. Our deep compositional model is not only able to construct meaningful sentence representation but also to keep pace with the words meanings evolving. We evaluate our obtained embeddings on semantic similarity task. The experimental results show the effectiveness of our proposed model and demonstrate that it can achieve a competitive performance without any feature engineering.

**Keywords:** Sentence meaning embeddings · Word embeddings · Recursive auto-encoders · Semantic similarity

## 1 Introduction

Embedding the meaning of a sentence into a vector space is a challenging and on going area of research in natural language processing as expected to be very useful for several natural language tasks. Neural word embedding technique has recently received a lot of attention and has proved to be a powerful tool for modeling semantic relations between individual words. Almost all word embedding models are based on the distributional hypothesis [1] which states that words that occur in the same contexts tend to have similar meaning. The context is usually defined as the words which precede and follow the target word within some fixed window in most word embedding models with various architectures. Word2vec [2] is a well-known word embedding model that received substantial success in mapping words with similar syntactic or semantic meaning to vectors close to each other.

Recently, there has been a growing interest in addressing the meaning of sentences representations. Simply averaging or summing word embeddings of all words in a longer pieces of text (sentences or documents) has surprisingly proven to be a strong baseline or feature across some tasks. However, these methods are poor ways of describing the distribution of word embeddings across a semantic space. In fact, the meaning of a word is affected not only by its adjacent words but also by the rules to combine them. As each context word in general does not contribute equally to the meaning of the target word, it would be desirable to capture the mutual interaction of distributed word vectors by a means of a compositional model. This direction can provide an opportunity for detecting more meaningful embedding. In other hand, most sentence embedding methods depend heavily on the domain of the dataset. They may also require a large textual corpus such as the skip-thought model [5], extensive pre-training and hyper-parameters tuning. Consequently, it may be hard to used them as universal embeddings across domains.

In this work, we propose a novel deep compositional method to learn meaning representation for variable-sized sentences based on recursive auto-encoders. Our motivation here comes from distributional hypothesis that the meaning of a word changes with new context. So, we think that evolving semantics enables us to capture the true meaning of the words in different usage contexts. We believe also that the global meaning of a sentence can affect the meaning of words within it. Consequently the evolved meaning of words can help constructing more meaningful sentence embedding. Besides, we would that our model build general-purpose embedding which can be used across domain. To this end, we suggest embedding the sentence meaning while jointly learning evolved word representation in unsupervised manner without involving a parse tree structure. We also introduce a new method to construct representation for out-of-vocabulary words. We evaluate our obtained embeddings on semantic similarity task. The experimental results show that our proposed model can produce meaningful embeddings that may perform well in practice.

The rest of this paper is organized as follows: In Sect. 2, we review some related work. Section 3 introduces our proposed method for learning meaningful sentence embedding. Thereafter, in Sect. 4, we describe our experimental setting, report the obtained results and compare them to some state-of-art works. Finally, we conclude this paper and provide some future works.

## 2 Related Work

Distributed representations of words, better known as word embedding, have garnered great success and popularity in NLP. It consists in mapping words into latent, dense, low-dimensional and real valued, vector representations that reflect as possible contextual, semantical and syntactical information among words. In this line, [2] propose word2vec model to embed words in a vector space such that words which share similar semantic or syntactic aspects are close to one another in this space. It offers two architectures: Continuous bag of words (CBOW)

architecture and skip-gram architecture. While skip-gram architecture predicts a context of a given words, the CBOW architecture predicts a word in a given context. [3] propose the Glove model that is a count-based model which learns by constructing a co-occurrence matrix that basically count how frequently a word appears in a context. Based on word2vec, [4] propose the FastText model taking into account sub-word, character-level information. Instead of learning a representation for a word, FastText learns a representation for all character n-grams in a word, thereby jointly optimizing semantic representations of character n-grams and full words.

Recent work has tried to compute meaningful and useful distributed representations of sentences. [7] trains a simple CNN with one layer of convolution on top of word2vec and a max-pooling operation over the entire sentence for each feature map. The author suggests that the pre-trained vectors are universal feature extractors that can be utilized for various classification. For further improvements, he modifies the architecture by having multiple channels to allow the use of both pre-trained and task-specific vectors. [6] propose modeling sentences by a dynamic convolutional neural network (DCNN). Their model is characterized by including dynamic k-max pooling to select the most active features. It was built on the concept of time delay neural network. This combination allowed the features to be extracted independently of their position in the sentence. [8] show that, with an annotated training corpus such Natural Language Inference (NLI), LSTM-based sentence encoder can capture useful features that are transferable to a various text classification tasks. [5] generalize the idea of word2vec word embeddings to model sentences. They exploit a sentence-level distributional hypothesis: it constructs a distribution over the words in the surrounding sentences given the current sentence. Their model is implemented in the encoder-decoder setting using LSTM networks. [9] propose to build general-purpose sentence encoder by learning from a joint objective of classification, machine translation, parse tree generation and unsupervised skip-thought tasks. [10] propose an unfolding recursive auto-encoders with dynamic pooling and constituency tree to learn composition functions of word embeddings to phrase embeddings and eventually sentence embeddings in a binarized constituent parse. [12] propose the tree-structured long short-term memory networks (Tree-LSTMs) for modeling sentences, which composes its states from an input vector and the hidden states of arbitrarily many child units. [13, 14] assume that trees are predicted based on explicit tree-bank annotations jointly with the treated task. The model proposed by [13] combines parsing and interpretation within a single tree-sequence hybrid model by integrating tree-structured sentence interpretation into the linear sequential structure of a shift-reduce parser. Meanwhile, the model proposed in [14] is used only for parsing and generation. It was a generative probabilistic model of sentences that explicitly models nested, hierarchical relationships among words and phrases. [15] consider a denoising auto-encoder model where noise is introduced in a sentence by deleting words and swapping bi-grams and the decoder is required to reconstruct the original sentence. [16] use unfolding recursive auto-encoders to learn feature vectors for

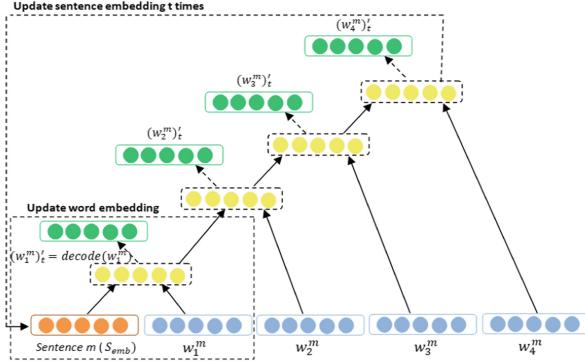
phrases in syntactical tree of the sentence. To compare two sentences, they use a similarity matrix which dimensions are proportional to the size of the two sentences. Since the similarity matrix generated to compare two sentences has varying dimension due to different sentence lengths, a dynamic pooling layer is used to map it to a matrix of fixed dimension. The resulting matrix is used to calculate the similarity scores between the two sentences.

### 3 Proposed Method

Most of previous methods focus on the syntactic aspects of a sentence by the means of parse dependency tree or use immediately neighboring sentences to learn sentence embedding. However, we are more interested in constructing representations that capture semantics of variable-sized sentences based on word meaning evolution and the meaning of the current sentence. We believe that not only word embeddings contain and express the semantics of the sentences in the vector spaces. But also the global meaning of sentences in which a word is used can help determining the intended meaning of this word. In fact, words can have more than one meanings. We believe that a change in the global meaning of a sentence where a word appears is a good indicator of a change in the meaning of a word. So, influenced the word meaning by the global meaning of the sentence contains it might be relevant to model its new semantic meaning. On other hand, initializing the word when it appears by its old meaning can help detecting new meanings for same words. That can provides an understanding of what the word means at that point.

Motivated by the aforementioned, we propose a new method to learn and embed the sentence meaning while jointly learning evolved word representation in unsupervised manner based on recursive auto-encoders. Traditional recursive auto-encoders are based on using recursive structure of parse trees, starting from the leaves and proceeding recursively in a bottom-up fashion until the root of the parse tree is reached. Our model differs from traditional recursive auto-encoders at three points. The first, we take into account words order and combine neighbors words recursively and sequentially. Second, we don't rely on a parse tree structure. Third, we jointly learn sentences and words meaning using their evolved representations. We construct the meaning of sentences and words iteratively and recursively with update of word embedding at each new input. We include the representation meaning of a sentence to construct the representation meaning of their words, where the sentence meaning is represented by an embedding computed based on a recursive auto-encoders model updated  $t$  times. The meaning of words is represented by an embedding computed from the decoding function. The evolved word embedding is fed as input to our model to construct sentence meaning. In fact, the word meaning is influenced not only by the meaning of the previous words but also by the meaning of the sentence where it occurs. Our model is illustrated in Fig. 1.

Given a sentence  $s = [w_1, w_2, \dots, w_i, \dots, w_j, \dots, w_n]$  with length  $n$ , our model regards it as a sequence of ordered words and we apply the auto-encoder recursively (as shown in Fig. 2). Among the sequence, our model combines each pair



**Fig. 1.** Illustration of the proposed model.

of sibling nodes into a potential parent node. It takes the first pair of neighboring vectors, the sentence embedding vector and the first word vector within the sentence, defines them as potential children of a sentence  $(c_1, c_2) = (s_{emb}, w_1)$ , concatenates and gives them as input to the auto-encoder. Then, we save the potential parent node  $p_1$ , the right reconstructed children which represent the reconstructed vector embedding of the current word of the given sentence. Therefore, the network is shifted by one position and takes as input vectors  $(c_1, c_2) = (p_1, w_2)$  and again computes a potential parent node and reconstructed vectors for potential child. This process repeats until it hits the last pair of vectors in the sentence:  $(c_1, c_2) = (p_{n-1}, w_n)$ . Each potential parent  $p_i$  is calculated using Eq. 1.

$$p_i = f(W_\phi^i [c_1, c_2] + b_\phi^i) \quad (1)$$

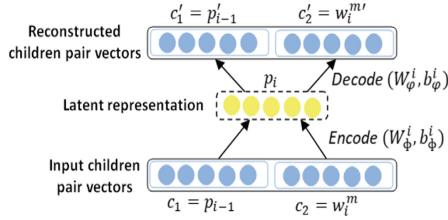
where  $W_\phi^i \in \Re^{d \times 2d}$  is an encoding weight matrix ( $2d$  is the number of input units),  $c_1$  and  $c_2$  are the  $d$ -dimensional vector representations corresponding to every child in the pair,  $b_\phi^i$  is the encoding bias vector and  $f$  is a non-linear activation function. The learned parent vector  $p_i$  is also a  $d$ -dimensional vector. The resulting representation  $p_i$  is then mapped back to a reconstructed vectors of original children pair vectors in order to obtain more informative and abstract representation of children meaning by performing the following evaluation:

$$\left[ \overset{' }{c_1}, \overset{' }{c_2} \right] = f(W_\varphi^i p_i + b_\varphi^i) \quad (2)$$

This function is parametrized by  $W_\varphi^i$  which is a decoding weight matrix and  $b_\varphi^i$  which is a decoding bias vector.

In Fig. 2, we present an example of computing a parent node.

We assume that sentence encodes the general semantic direction of their individual words and each word contributes to construct more significant meaning of sentence embedding. So, we used the latent representations computed by auto-encoder recursively for computing the meaning of the whole sentence. The final latent representation is interpreted as a representation of the sentence. This



**Fig. 2.** Example of computing a parent node in our model.

representation will be used as input for a next training for our recursive auto-encoders. The decoder part will be used as initialization for words when they occurred within a sentence that will later be trained. This process was repeated  $t$  times.

We explored two strategies to initialize the sentence vector included as a child in our model input: random real-valued initialization and weighted average of vectors of words constituent a sentence. We also suggest a new method to handle out-of-vocabulary words. The main idea consists at determining for each out-of-vocabulary word within a test sentence, all training sentences containing this word. From the constructed set of sentences, we select one that has the maximum number of common words. The out-of-vocabulary word will be initialized by the average of these words. In fact, we design three model variants considering the different methods of sentence embedding and out-of-vocabulary initialization.

The first variant (model-v1) consists at initializing the sentence representation and out-of-vocabulary words randomly with each component sampled from uniform distribution. The second variant (model-v2) employs the suggested method of handling out-of-vocabulary words and random initialization for sentence representation input. We furthermore test if our model does better when the initialized sentence embedding be a weighted average of word vectors obtained by word2vec (model-v3). We constructed the initial sentence embedding by weighting each word embedding within a sentence by its *idf* and therefore averaging the resulted embeddings. We treat each sentence as a document and calculate the *idf* weight for word  $w$  as follows:

$$idf_w = \log \frac{N}{1 + n_w} \quad (3)$$

where  $N$  is the total number of sentences and  $n_w$  is the number of sentences in which the word  $w$  occurs.

## 4 Experiments

### 4.1 Setting

We build our model using the framework Tensorflow and keras for python. We adopt the Adam optimizer [17] to tune the network parameters. As objective

function, we apply mean square error (MSE). It is calculated between the original input  $(p_{i-1}, w_i)$  and its reconstructed vector  $(p'_{i-1}, w'_i)$  of each auto-encoder bloc of our recursive model using Eq. 4.

$$MSE(p_i) = \left\| \begin{bmatrix} p'_{i-1} \\ w'_i \end{bmatrix} - \begin{bmatrix} p_{i-1} \\ w_i \end{bmatrix} \right\|^2 \quad (4)$$

We initialize the word embeddings using word2vec tool. We use 200 as dimensional size for embeddings. To measure the performance of our model, we utilize two metrics the Pearson Correlation coefficient, denoted by  $r$ , and the Spearman's Rank Correlation coefficient, denoted by  $p$ . We performed the semantic similarity task in an unsupervised framework by calculating the cosine similarity between two sentence vectors.

## 4.2 Results

In this section, we present the results for the experiments we conducted with the setup explained in the previous section.

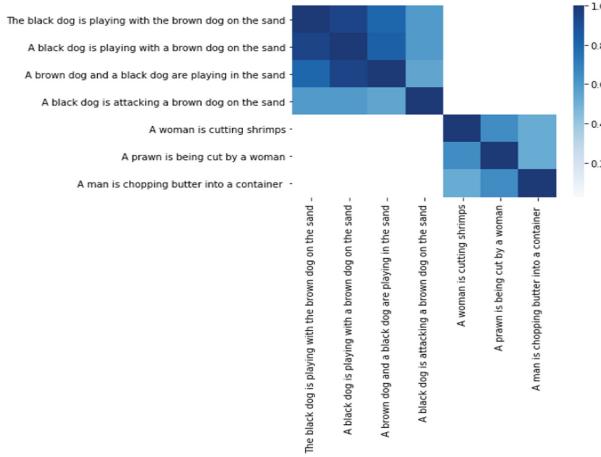
In Table 1, we show the results of our model when trained on SICK and SMTeuropal datasets. We also include obtained results with two tested methods for embedding sentences. *Idf*-avg-w2v method explicated above and Avg-w2v method where we construct representations of variable-length sentences by averaging the embeddings of all words within a sentence obtained by word2vec model.

**Table 1.** Experimental results on sentence similarity dataset.

Method	SICK		SMTeuropal	
	Pearson's r	Spearman's p	Pearson's r	Spearman's p
Avg-w2v	0.6502	0.5503	0.3638	0.4766
Idf-Avg-w2v	0.6585	0.5585	0.3698	0.4674
Our model <sub>v1</sub>	0.8381	0.7917	0.4517	0.5410
Our model <sub>v2</sub>	0.8462	0.7976	0.4576	0.5439
Our model <sub>v3</sub>	0.8592	0.8012	0.4694	0.5457

We can see that overall, our model variants achieve a better performance compared with our baselines simple averaging and *idf*-weighted averaging. The simple averaging is the poorly performed method. That can be explicated by the fact that taking the average of word embedding in a sentence tends to give too much weight to words that are quite irrelevant semantically. We can find that our embeddings yield more semantic information especially for our model-v3 which proves to be the more promising variant. This indicates that our assumption of considering evolving words meaning for modeling sentence meaning was beneficial for learning sentence meaning as well as in learning better word meaning.

In Fig. 3, we illustrate the semantic similarity predicted scores obtained by our model for some sentences from SICK test corpus.



**Fig. 3.** Heatmap of example predictions from the SICK set test.

We can find from Fig. 3 that when two sentences have a lot of word overlap, and have little differences in key semantic roles (A black dog is attacking a brown dog on the sand and A black dog is playing with a brown dog on the sand), our model tends to make smaller error and it seems to be good at capturing semantics similarity. When sentence pairs differ in meaning and have few words in common (A woman is cutting shrimps and A prawn is being cut by a woman), our model can detect the semantic differences. Thus, we can say that our model is able to give us satisfying predictive semantic similarity scores.

Table 2 displays the results of some state-of-art methods on SICK dataset and our results.

The four first results represent high level of traditional methods that are released by SemEval 2014. It can be seen that the result of our model outperforms the aforementioned results, according to pearson and spearman correlations. The four last models represent four well known state-of-art models; recursive auto-encoder models proposed by [11] and recurrent models based on parse-tree proposed by [12]. It can be seen that our model-v3 perform well against others state-of-art sentence embedding. Thus we can say that our model is promising and having competitive results comparing it with traditional methods and state-of-art deep networks.

Table 3 shows the Pearson correlation of our model against some state-of-art methods on SMTEuroparl dataset.

Comparing our result to three best results on Semeval task and some state-of-art results on Smteuroparl dataset further shows that our model provide

**Table 2.** Experimental results for different methods on SICK dataset.

Method	Pearson's r	Spearman's p
Illinois-LH [20]	0.7993	0.7538
UNAL-NLP [21]	0.8070	0.7489
Meaning factory [18]	0.8268	0.7721
ECNU [19]	0.8414	-
DT-RNN [11]	0.7923	0.7319
SDT-RNN [11]	0.7900	0.7304
CT-LSTMs [12]	0.8582	0.7966
Our model <sub>v3</sub>	0.8592	0.8012

**Table 3.** Pearson results for different methods on SMTeuropal dataset.

Method	Pearson's r
[22]	0.4203
[24]	0.3612
[23]	0.5280
RNN	0.409
LSTM	0.443
[25]	0.450
Our model <sub>v3</sub>	0.4694

competitive results. For example, our model is better than the model of [24] by 0.1082 and that of [25] by 0.0194.

In contrast to other methods which are trained using external knowledge information such as wordnet or parse trees, our unsupervised method uses only word embeddings and recursive auto-encoder without external information. For example, the ECNU model [19] uses four learning methods, WordNet and additional corpus; The meaning factory model [18] also utilizes three different knowledge base including WordNet. [23] also use external resources. These additional corpus consist of rich semantic information which is quite helpful for representing sentence meaning. Furthermore, previous works use pre-training on a much larger corpus which could introduce prior knowledge of the dataset. We would note that the output of our neural network is a vector that uses to compute the score of sentence similarity. In other works, the score of sentence similarity is directly computed by the neural network using linear regression. In addition, our model yields not only meaningful sentence embedding but also evolving word meaning embedding. So, obtained competitive results with our unsupervised model is good and we can conclude that using evolved word meaning improve modeling sentence meaning. Also, incorporating sentence representation as global meaning improves the word semantic embedding.

## 5 Conclusion

In this work, we address the problem of learning sentence representation that capture semantics for variable-length sentence in unsupervised manner. To do so, we propose a novel method based on recursive auto-encoders. Our model incorporates sentence embedding with embeddings of words that contain without using parse tree structure. We take word meaning evolution into consideration. That's every sentence will be embedded as a vector representing its semantic meaning using the evolved embedding of words meaning within it. We also propose a method addressing the problem of unknown words. This method ameliorates results comparing to randomly embedding unknown words. Conducted experiments on semantic similarity task show that proposed method for learning meaningful sentence embeddings gives competitive results. In the future, we plan to introduce attention mechanism to enhance more semantic representations of sentences.

## References

1. Harris, Z.S.: Distributional structure. *Word* **10**(2–3), 146–162 (1954)
2. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: ICLR (2013)
3. 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, 25–29 October, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, pp. 1532–1543 (2014)
4. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguist.* **5**, 135–146 (2017)
5. Kiros, R., Zhu, Y., Salakhutdinov, R., Zemel, R.S., Torralba, A., Urtasun, R., Fidler, S.: Skip-thought vectors. In: NIPS (2015)
6. 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)
7. 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)
8. Conneau, A., Kiela, D., Schwenk, H., Barrault, L., Bordes, A.: Supervised learning of universal sentence representations from natural language inference data. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (2017)
9. Subramanian, S., Trischler, A., Bengio, Y., Pal, C.J.: Learning general purpose distributed sentence representations via large scale multi-task learning. In: International Conference on Learning Representations (2018)
10. Socher, R., Huang, E.H., Pennin, J., Manning, C., Ng, A.Y.: Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In: Advances in Neural Information Processing Systems, USA, pp. 801–809 (2011)
11. Socher, R., Karpathy, A., Le, Q.V., Manning, C.D., Ng, A.Y.: Grounded compositional semantics for finding and describing images with sentences. *Trans. Assoc. Comput. Linguist.* **2**, 207–218 (2014)

12. 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)
13. Bowman, S.R., Gauthier, J., Rastogi, A., Gupta, R., Manning, C.D., Potts, C.: Fast unified model for parsing and sentence understanding. ACL (2016)
14. Dyer, C., Kuncoro, A., Ballesteros, M., Smith, N.A.: Recurrent neural network grammars. In: NAACL, San Diego, California, 12–17 June 2016, pp. 199–209 (2016)
15. Hill, F., Cho, K., Korhonen, A.: Learning distributed representations of sentences from unlabelled data. In: NAACL-HLT, San Diego, California, 12–17 June 2016, pp. 1367–1377 (2016)
16. Grover, J., Mitra, P.: Sentence alignment using unfolding recursive autoencoders. In: Proceedings of the 10th Workshop on Building and Using Comparable Corpora ACL, Vancouver, Canada, 3 August 2017, pp. 16–20 (2017)
17. Kingma, D.P., Ba, J.L.: Adam: a method for stochastic optimization. In: International Conference on Learning Representations, pp. 1–13 (2015)
18. Bjerva, J., Bos, J., Van der Goot, R., Nissim, M.: The meaning factory: formal semantics for recognizing textual entailment and determining semantic similarity. In: SemEval COLING, pp. 642–646 (2014)
19. Zhao, J., Zhu, T., Lan, M.: ECNU: one stone two birds: ensemble of heterogenous measures for semantic relatedness and textual entailment. In: SemEval COLING, pp. 271–277 (2014)
20. Lai, A., Hockenmaier, J.: Illinois-LH: a denotational and distributional approach to semantics. In: International Workshop on Semantic Evaluation, Dublin, Ireland (2014)
21. Jimenez, S., Dueñas, G., Baquero, J., Gelbukh, A.: UNAL-NLP: combining soft cardinality features for semantic textual similarity, relatedness and entailment. In: Proceedings of the 8th International Workshop on Semantic Evaluation (SemeVal), Dublin, Ireland (2014)
22. Banea, C., Hassan, S., Mohler, M., Mihalcea, R.: UNT: a supervised synergistic approach to semantic text similarity. In: Proceedings of SemEval 2012, pp. 635–642 (2012)
23. Bar, D., Biemann, C., Gurevych, I., Zesch, T.: UKP: computing semantic textual similarity by combining multiple content similarity measures. In: Proceedings of SemEval 2012, pp. 435–440 (2012)
24. Saric, R., Glavas, G., Karan, M., Snajder, J., Dalbelo, B.: TakeLab: systems for measuring semantic text similarity. In: Proceedings of SemEval 2012, Montreal, Canada, pp. 441–448 (2012)
25. Kenter, T., Borisov, A., Rijke, M.D.: Siamese CBOW: optimizing word embeddings for sentence representations. CoRR abs/1606.04640 (2016)



# Pruning Extreme Wavelets Learning Machine by Automatic Relevance Determination

Paulo V. de Campos Souza<sup>1,2</sup> , Vinicius J. Silva Araujo<sup>2</sup> , Vanessa S. Araujo<sup>2</sup> , Lucas O. Batista<sup>2</sup>, and Augusto J. Guimaraes<sup>2</sup>

<sup>1</sup> CEFET-MG, Av. Amazonas, 5253, Nova Suica, Belo Horizonte, MG, Brazil  
`goldenpaul@informatica.esp.ufmg.br`

<sup>2</sup> Faculty UNA Betim, Av. Gov. Valadares, 640 - Centro, Betim, MG, Brazil

**Abstract.** Extreme learning machines are used for various contexts in artificial intelligence, such as for classifying patterns, performing time series prediction and regression problems, and being a more viable solution for training hidden layer weights to determine values of the learning model. However, the essence, the model determines that these weights should be determined randomly, and the Moore Penrose pseudoinverse will define only the weights that will act in the output layer. Random weights make this learning a black box because there is no relationship between the hidden layer weights and the problem data. This paper proposes the initialization of weights and bias in the hidden layer through the Wavelets transform that allows the two parameters, previously initialized at random, to be more representative about the problem domain, allowing the frequency range of the input patterns of the network to aid in the definition of weights of the ELM hidden layer. To assist in the representativeness of the data, a technique of selection of characteristics based on automatic relevance determination will be applied to the selection of the most characteristic dimensions of the problem. To compose the network structure, activation functions of the type rectified linear unit, also called ReLU, were used. The proposed model was submitted to the classification test of binary patterns in real classes, and the results show that the proposition of this work assists in bringing better accuracy to the classification results, and thus can be considered a feasible proposition to the training of neural networks that use extreme learning machine.

**Keywords:** Extreme learning machine · Wavelets · ReLU · Automatic relevance determination · Pattern classification

## 1 Introduction

The Extreme Learning Machine [10] is a learning algorithm for hidden layer single feedforward networks - SLFNs with low computational complexity and

ensuring fast convergence and good generalization performance. The random definition of the parameters of the hidden layer and the estimation of the weights of output via ordinary least squares allow to carry out the training in a single iteration, but it makes the ELM algorithm a kind of black box, where the hidden layer has little interpretability about the problem. In situations where the amount of neurons in the hidden layer is high, and many attributes are estimated at random, overfitting may occur, allowing the model to be susceptible to the dataset used and lose its generalization capacity. The original formulation of the ELM algorithm does not establish a methodology for defining the structure of SLFN. The hidden layer dimension is usually inferred by trial and error, and the weights are randomly defined. Although it is a more efficient methodology than backpropagation concerning the update of hidden layer weights, the random definition makes it difficult to understand the real importance and relevance of hidden layer weights to the problem. If the weights of the hidden layer were defined using a more consistent pattern, there would probably be ways to interpret the architecture of an ELM-trained SLFN more coherently. The studies performed with the wavelet transform [6], despite being widely used in signal-related problems, can identify a frequency range of the data in the input patterns of a neural network, thus creating weights that follow the activation of the patterns submitted to the model. This paper proposes to create a training methodology for SLFN networks based on an extreme learning machine, where the weights and bias of the hidden layer will not be defined randomly but characterized according to the frequency domain of the input information submitted to the model using the concepts of the transform of wavelets proposed by [6]. All the functions of activation of the internal neurons will be of type ReLU [16], seeking to treat in a simplified way the relevance of the weights on the inputs of the model. Recent studies show that rectified linear unit activation functions can improve the predictive capacity of intelligent models due to the pure nature of treating the attributes involved in neural network training [12]. Therefore it is necessary to verify the impacts of the ReLU activation function with weights and bias defined by the frequency of the input data. The paper is organized as follows: Sect. 2 presents the central concepts that guide the research, such as definitions of ELM, Wavelets and activation functions. In Sect. 3 will be presented the steps and concepts related to the proposed methodology for generating the ELM hidden layer weights based on the wavelet transform so that neurons based on ReLU type activation functions can perform binary pattern classification. Section 4 will present the methodology used in tests, including the bases and algorithms used for performing binary classification of patterns. Finally in Sect. 5 will be presented the conclusions of the paper.

## 2 Literature Review

### 2.1 Extreme Learning Machine

The ELM (Extreme Learning Machine) is a learning method developed for hidden layer feedforward neural networks (SLNFs) in 2006, where its main contri-

bution is in the simple adjustment of parameters in training, avoiding successive evaluations and updates of internal parameters of the network [10].

Because it has its weights that bind the hidden layer with the estimated output layer in a single step through the pseudo-inverse concepts of the array of neuron activations functions present in the hidden layer of the model [10].

Given  $N$  arbitrary training samples with  $m$  features and associated outputs, composing pairs of type  $(\mathbf{x}_i, y_i)_{i=1}^N \in \mathbb{R}^{m \times c}$ , where  $c$  is the dimension of  $\mathbf{x}_i$ , the model representing a SLFN with  $l$  hidden nodes, activation functions  $f(\cdot)$ . The ELM learning algorithm, can be defined as follows: [3, 10, 17]:

$$y_i = \sum_{j=1}^l h_j f(\mathbf{w}_j, x_i, b_j) \quad i = 1, \dots, N. \quad (1)$$

where  $\mathbf{w}_j$  is the weight vector of the connections between the  $m$  inputs and the hidden  $j$ -th neuron,  $h_j$  is the vector of weights of the connections between the  $j$ -th hidden neuron and the neurons of the network output, and  $b_j$  is the bias of the  $j$ -hidden neuron. For the ELM,  $f(\cdot)$  is the activation function applied to the scalar product of the input vector with the hidden weights  $w_k$  that are defined at random. The function  $f(\cdot)$  can be for more types (ex. sigmoidal) [10, 17]. With the model defined in (1), we can write  $\mathbf{y}$  as  $\mathbf{H} * \beta$ , where  $\beta$  is the vector of weights of the output layer,  $\mathbf{y}$  is the vector of outputs.  $\mathbf{H}$  is determined to be [3, 10, 17]:

$$\mathbf{H} = \begin{bmatrix} f(w_1, x_1 + b_1) & \dots & f(w_m, x_1 + b_l) \\ f(w_1, x_2 + b_1) & \dots & f(w_m, x_2 + b_l) \\ f(w_1, x_n + b_1) & \dots & f(w_m, x_n + b_l) \end{bmatrix}_{N \times l} \quad (2)$$

The columns of the matrix  $\mathbf{G}$ , defined in (2), correspond to the outputs of the hidden neurons of the SLFN with respect to the input  $\mathbf{X} = [x_1, x_2, \dots, x_N]_{m \times N}^T$ . The ELM implements a random initialization of the weights of the hidden layer (based on a numerical range any),  $w_k$ . Then, the weights of the output layer are obtained through the pseudo inverse [9] according to the expression [10, 17]:

$$\beta = \mathbf{H}^+ \mathbf{y} \quad (3)$$

$H^+$  is the Moore-Penrose pseudo Inverse of  $\mathbf{H}$ , which is the minimum norm of the least squares solution for the output weights.

This approach is efficient for updating weights, but when the amount of neurons in the hidden layer is disproportionate to the amount needed to solve the problems, the model can suffer from overfitting, thus generating problems in its capacity to generalize the data, since the model will be dependent on the training data and will not be able to perform intelligent tasks with new data sets. Some researchers approach the random definition in the hidden layers of intelligent models as an approach that does not contribute to the understanding of the model [21], since this type of approach does not bring relevant characteristics of the analyzed data.

## 2.2 Wavelets

Wavelet is a function capable of decomposing and representing another function described in the time domain so that we can analyze this other function in different frequency and time scales. In Fourier analysis, we can only identify information about the frequency domain, but we can not know “when” in time these frequencies that we study happen; Meanwhile, in wavelet analysis, we can also extract information from the function in the time domain. The detailing of the frequency domain analysis decreases as time resolution increases, and it is impossible to increase the detail in one domain without decreasing it in the other. Using wavelet analysis, you can choose the best combination of details for an established goal. Adapting this concept to the artificial neural networks, the adjustment of the detail provided by the wavelets is an element capable of providing a generalization of network recognition. In this work, the discrete wavelet transform will be adopted. This type of methodology is much used in data compression. In order to calculate the discrete wavelet transform it is through the filter bank application where the filter determined by the coefficients  $h = \{h_n\}_{n \in \mathbb{Z}}$  corresponds to a high pass filter and the filter  $g = \{g_n\}_{n \in \mathbb{Z}}$  to a low pass filter. Each of these coefficients in the discrete wavelet transform is tabulated. Emphasis is given to the use of the operator  $(\downarrow 2)$  is the sub-sampling operator. This operator applied to a discrete function reduces its number of elements in half, allowing the procedure to be faster and more precise. The filters  $h$  and  $g$  are linear operators, which can be applied to the input  $x$  as a convolution:

$$c(n) = \sum_k g(k)x(n - k) = g * x \quad (4)$$

$$d(n) = \sum_k h(k)x(n - k) = h * x \quad (5)$$

where the signal  $c(n)$  is known as approximation and the signal  $d(n)$  as detail [6]. The decomposition with the filter decomposes the signal into only two frequency bands. We can chain a series of filter banks by using the sub-sampling operation to provide the division of the sampling frequency by 2 to each new filter bank threaded.

## 2.3 Related Work

The extreme learning machines have been the target of several recent types of research, mainly in regression models, binary classification and multiclass. However, the studies concerning the incorporation of wavelet techniques into its structure are still in the initial stage. The work of [8] wavelets are used in kernel functions to compose the structures of the activation functions of the network neurons, similar to the work of [4] where wavelets processed the inputs and submitted to functions continuous activation by non-constant and limited parts. Subsequently, the feature selection method uses non-zero wavelet parameters that are used to initialize conversion and network expansion parameters. Finally,

the ELM-based training method is used. The concepts of wavelets and ELM are also used in composite functions with differential evolution and finally a parameter initialization with dual wavelet-based activation functions, in addition to a combination of Morlet wavelets function and inverse hyperbolic sine function with initialization of weights and bias through a heuristic procedure [11]. Other works that use the ELM to train models along with the concepts of wavelets [1, 5, 7, 14]. Several works use the ReLU function and ELM to solve problems, mainly related to deep learning [13, 18, 24]. The main advantage of using the ReLU function over other activation functions is that it does not activate all neurons at the same time. This means that if for the ReLU function and the input is negative, it will be converted to zero and the neuron will not be activated allowing at the same time, only some neurons are activated, making the network sparse, efficient and easy for the computational processing of answers.

The determination of the number of neurons has also been the target of several academic works. Some approaches use incremental methodology (start with a low number of neurons and gradually increase), and others use pruning techniques that allow the network to start with a high number of neurons and techniques pruning the neurons less relevant. One of the most prominent pruning works in neural network architectures that use ELM was proposed by [19] which uses statistical techniques to perform the pruning of neurons. Affinity matrix techniques, data density, probabilistic and other statistical evaluations are used in [3, 15, 22]. This work differs from other authors' proposals due to the use of wavelet functions to define weights and bias and the technique of pruning based on Bayesian techniques.

## 2.4 Automatic Relevance Determination

Bayesian techniques have several applications in science, including to determine the degree of relevance of a variable to an analyzed context. However, to find the best results within a relevance problem of a neural network, these techniques may take a long time due to the convergence problem than some Bayesian probability-based techniques have. The estimation problem in intelligent models with a large number of resources is fundamentally ill-posed. A useful unintelligible penalty emerges from a dual space view of sparse Bayesian learning, which is based on the notion of automatic relevance determination (ARD) that solves this problem by regularizing the solution space using a parameterized prior distribution data-dependent prior distribution that effectively eliminates redundant or superfluous features [20]. [23] gives the canonical form of this problem

$$\min_{\mathbf{x}} \|\mathbf{x}\|_0, \quad \text{s.t. } \mathbf{y} = \Phi \mathbf{x} \quad (6)$$

where  $\Phi \in \mathbb{R}^{n \times m}$  is a matrix whose columns  $\phi_i$  represent an overcomplete basis (i.e.,  $\text{rank}(\Phi) = n$  and  $m > n$ ),  $\mathbf{x} \in \mathbb{R}^m$  is a vector of unknown coefficients to be learned, and  $\mathbf{y}$  is the signal vector. The cost function being minimized represents the  $l_0$  norm of  $\mathbf{x}$ , which is a count of the nonzero elements in  $\mathbf{x}$  [23]. If measurement noise or modeling errors are present, we instead solve the alternative problem for this way:

$$\min_x \|y - \Phi x\|_2^2 + \lambda \|x\|_0, \quad \lambda > 0 \quad (7)$$

SBL assumes a Gaussian likelihood function  $p(y|x) = \mathcal{N}(y|\Phi x, \lambda I)$ , consistent with the data fit term from (6). The basic ARD prior incorporated by SBL is  $p(x; \gamma) = \mathcal{N}(x; 0, \text{diag}[\gamma])$ , where  $\gamma \in \mathbb{R}_+^m$  is a vector of  $m$  non-negative hyperparameters governing the prior variance of each unknown coefficient. These hyperparameters are estimated from the data by first marginalizing over the coefficients  $\mathbf{x}$  and then performing what is commonly referred to as evidence maximization or type-II maximum likelihood, this is equivalent to minimizing [23]:

$$\mathcal{L}(\gamma) \triangleq -\log \int p(y|x)p(x; \gamma)dx = -\log p(y; \gamma) \equiv \log |\Sigma_y| + y^T \Sigma_y^{-1} y \quad (8)$$

where  $\Sigma_y \triangleq \lambda I + \Phi \Gamma \Phi^T$  and  $\Gamma \triangleq \text{diag}[\gamma]$ . Once some  $\gamma_* = \arg \min_\gamma \mathcal{L}(\gamma)$  is computed, an estimate of the unknown coefficients can be obtained by setting  $x_{\text{SBL}}$  to the posterior mean computed using  $\gamma_*$  [23]

$$x_{\text{SBL}} = \mathbb{E}[x|y; \gamma_*] = \Gamma_* \Phi^T \Sigma_{y*}^{-1} y. \quad (9)$$

Note that if any  $\gamma_{*,i} = 0$ , as often occurs during the learning process, then  $x_{\text{SBL},i} = 0$  and the corresponding dictionary column is effectively pruned from the model. The resulting  $x_{\text{SBL}}$  is therefore sparse, with nonzero elements corresponding with the “relevant” basis vectors [23].

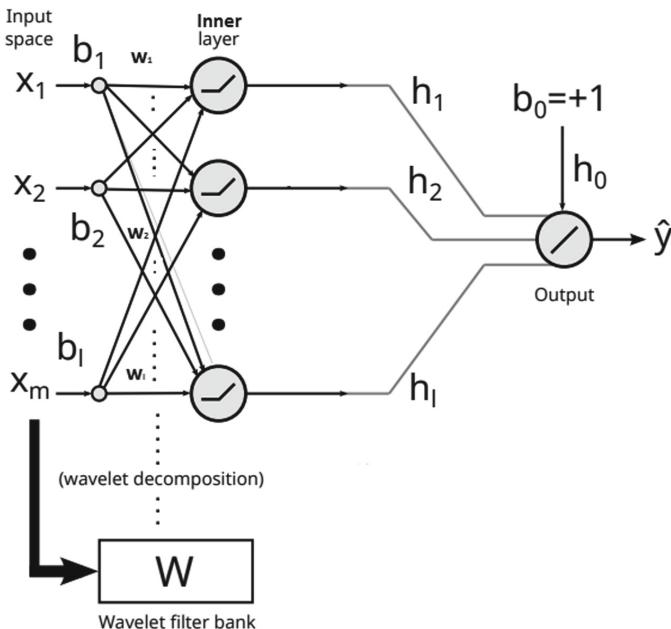
### 3 Pruning Extreme Wavelet Learning Machine and ReLU

#### 3.1 Network Architecture

The architecture of the proposed model follows the assumptions widely used in the literature, where a Single Layer Feed Forward Network (SLFN) has a certain amount of hidden neurons, and these neurons have weight and value of bias calculated through wavelet functions. The number of neurons in the hidden layer varies according to the values chosen by the network architecture, and the training is performed through an extreme learning machine. A single output neuron carries the binary responses of the network. All neurons involved in the architecture of this network are of the ReLU type. The Fig. 1 shows the architecture explained in this topic.

#### 3.2 Proposition to Update Hidden Layer Weights and Bias Using Wavelets

For the hidden layer of the SLFN, training will be performed with each output of the filters of each level of the Wavelet transform, thus allowing to update the



**Fig. 1.** Single Layer Feed Forward Network architecture

weights and bias, which by the original definition of the ELM should be randomly assigned, assigning them the corresponding values of the output of the Wavelet filters. Thus, the training of the neural network can happen differently from the traditional ELM algorithm, the characteristic of the input data of the model will allow in a single step the definition of values for weights and bias. Initially, the wavelet transform is applied to the input  $\mathbf{X}$  of the model resulting in a vector  $\psi_1$ . This vector is then passed to a detail removal function that eliminates the high frequencies (if any) of the vector  $\psi_1$  by adjusting the number of neurons  $l$  in the hidden layer so that the training can be done resulting in a vector  $\phi_1$ . In the hidden layer of this architecture, the initial vector has  $l$  values. After the application of the Wavelet transform, the resulting vector still with  $l$  elements but part of this vector is responsible for the high frequencies (detail), and the other part is responsible for the low frequencies (approximation).

Consider that, for the ELM example, the initial data vector has ten dimensions. After applying the wavelet transform, the resulting vector continues with ten elements, but part of that vector is responsible for the high frequencies (detail), and the other part is responsible for the low frequencies (approximation). When operating  $\text{RemoveDetails}(\psi_1)$  only the  $r$  vector values are used (for example six). In this way, we have two vectors: a vector of ten items (first layer input) and another vector of six characteristics (first layer output). From this six-element vector, the values responsible for the approximation are assigned to the bias and the detail value to the weights of the neurons of the first layer (choice

made at random). In this paper, the high values of the filter will be assigned to the weights of the neurons, and the low values of the filter will be allocated to the bias. The option occurred without any apparent criteria because there are no factors that prove that the inverse would not work either. This procedure ensures that the same amount of weights and bias that would be randomly assigned are provided based on the Wavelet transform, allowing these two parameters to be based on the characteristics of the dataset submitted to the model.

---

**Algorithm 1.** ELM training with filter bank *Wavelets* for weight  $w$  and bias  $b$

---

```

 $\psi_1 \leftarrow \text{Wavelet}(\text{input});$ 
 $\phi_1 \leftarrow \text{RemoveDetails}(\psi_1);$ 
 $\text{Train}(in = \text{input}, out = \phi_1);$ 

```

---

### 3.3 Use of Activation Functions of Type Rectified Linear Activation (ReLU) in the Hidden Layer Neurons and the SLFN Output Layer

The activation functions introduce a nonlinear component in the neural networks so that they can learn more than linear relationships between the dependent and independent variables. The activation functions are essential to give artificial neural networks representative capacity by introducing a nonlinearity component. In order to identify more efficient functions to act as activation functions the work [16] defined the rectified linear activation (ReLU). This function is defined by:

$$\text{ReLU}(x) = \max(0, x) \quad (10)$$

ReLU is the nonlinear activation function more widely used when designing neural networks today. It has as main advantage the use of a smaller number of neurons because it does not activate all the neurons at the same time. This means that if the input is unimpressive to the model, the ReLU function will convert its heat to zero and the neuron will not be activated. In the same time, only a few neurons are activated, making the network sparse and efficient and easy for computing [16].

## 4 Using Automatic Relevance Determination (ARD) to Selected Best Neurons

The following algorithm presents the steps performed for the pruning of unnecessary neurons.

**Algorithm 2.** Pruning algorithm for Extreme Learning Machines

---

```

1-CREATE Initiate hidden neurons,  $k$  (in general, the same quantity of training
and test samples).
2-Assign values to  $w_j$  and  $b_j$  using wavelet transform proposed by Alg. 1
for all  $N \neq 0$  do
    3-Compute  $H$  2
end for
4- Choose the best neurons using sparse Bayesian learning pruning methods to
accomplish the adequacy of the ELM architecture. At this point, we used two
main variables (vector  $\mathbf{H}$  Eq.(2) and  $\mathbf{Y}$  for training) to compare ARD.
5-Select  $L_p$  relevance neurons using ARD. (based on (9)). After collecting the
ranking of the values, pruning of unnecessary information is performed creating
new  $\mathbf{H}$ .
6- Estimate the output layer weights using Eq. (3)
7- Calculate output hidden layer using Eq. (1).

```

---

#### 4.1 Assumptions and Initial Test Configurations

In this section, we will discuss the classification tests for the model proposed in this paper. In order to perform the tests, real and synthetic bases were chosen, seeking to verify if the accuracy of the proposed model surpasses the traditional techniques that work in ELM. The information tables present information about the tests, presenting factors such as the percentage of samples destined for the training and testing of the neural networks. All the tests with the algorithms involved were done randomly so that tendencies that could interfere in the evaluations of the results are avoided. The proposed WR-ELM model was compared with the state of the art of pruning in ELM (OP-ELM) [19] and a recent model that uses the Matthew coefficient to prune less relevant neurons (CM-ELM) [3]. In the models compared in the test were used as activation function in neurons is the sigmoid, weights were used in the randomly defined hidden layer. A total of 30 experiments were done with the three models submitted to all test bases. In all tests and all models, the number of primary neurons was the same number of samples in the dataset. The samples were shuffled with each test to demonstrate the real capacity of the models. In the results tables are presented percentage values for the classification tests, accompanied by the standard deviation found in the 30 repetitions. The expected pattern obtained all responses with the response obtained. Finally, AUC is also highlighted for classification tests. The outputs expected in the test were set to 0 and 1 to meet ReLU responses. Therefore all bases used had their outputs converted to zero and one. Accuracy is the primary test result. It is given as a percentage and compares the response obtained by the model with the expected response. When the two are equal, a hit unit is added. In the end, the total of hits obtained by the model is taken and divided by the total number of samples destined for the test in order to obtain the accuracy of the model.

## 4.2 Dataset Used in the Tests

The following tables identify the settings applied in the tests. In Table 1 the real databases extracted from [2] for classification problems.

**Table 1.** Dataset used in the experiments

Dataset	Init.	Feature	Train	Test	Neurons (l)
Haberman	HAB	3	214	92	306
Transfusion	TRA	4	523	225	748
Mammographic	MAM	5	581	249	830
Liver Disorder	LIV	6	242	103	345
Diabetes	DIA	8	538	230	768
Heart	HEA	13	189	81	270
Spam	SPA	57	3221	1380	4601
Sonar	SON	60	146	62	208

## 4.3 Binary Pattern Classification Tests

The following are the pattern classification results for the real dataset. Also unique are the tables that present the accuracy and final neurons after pruning (Tables 2 and 3).

**Table 2.** Accuracies of the model in the tests performed.

Dataset	OP-ELM	CM-ELM	WR-ELM
HAB	<b>70.21 (2.42)</b>	70.07 (3.32)	68.50 (4.42)
TRA	78.71 (2.07)	78.37 (2.19)	<b>79.14 (2.16)</b>
MAM	82.24 (2.28)	82.10 (2.11)	<b>83.45 (2.29)</b>
LIV	63.86 (5.57)	66.68 (4.41)	<b>67.16 (4.28)</b>
DIA	75.03 (2.67)	69.20 (2.56)	<b>75.10 (2.48)</b>
HEA	79.72 (4.09)	76.40 (11.12)	<b>79.88 (3.77)</b>
SPA	86.21 (1.33)	<b>88.13 (1.43)</b>	84.05 (1.33)
SON	<b>78.59 (4.48)</b>	67.53 (0.78)	76.59 (4.51)

In the execution of real datasets, it was verified that the model proposed obtained superior results of accuracy in five of the datasets in the test. In the other bases, the proposed model maintained a difference within the standard deviation in two of the three bases, showing that the approach is statistically equivalent to the original ELM training propositions, adding a more direct relationship with the database for the determination of the weights in the hidden

**Table 3.** Number of final neurons

Dataset	TN-ELM	SG-ELM	WR-ELM
HAB	27.18 (2,09)	20.16 (1,12)	<b>18.26 (0.89)</b>
TRA	112.43 (9.21)	86.71 (21.15)	<b>29.19 (7.87)</b>
MAM	115.67 (19.15)	129.91 (67.12)	<b>92.87 (36.42)</b>
LIV	32.42 (10.16)	65.91 (9.01)	<b>19.18 (7.12)</b>
DIA	112.86 (32.54)	77.62 (18.51)	<b>44.09 (12.78)</b>
HEA	72.19 (10.19)	82.99 (42.18)	<b>38.26 (17.10)</b>
SPA	187.76 (61.29)	176.99 (79.09)	<b>106.87 (51.18)</b>
SON	42.96 (18.76)	33.85 (16.01)	<b>28.52 (10.19)</b>

layer. It is also noticed that the model proposed in the paper has the best answers with the least average number of neurons. This demonstrates that the pruning technique acted efficiently in choosing the most significant neurons.

## 5 Conclusion

We can verify that the definition of weights and bias in a random way is satisfactory, but the results obtained with the determination of the wavelet functions vary according to the input samples in conjunction with ReLU activation functions the effects of making networks that use ELM as a basis for your most efficient and accurate training. The obtained results demonstrate that the processing capacity is better to neutralize the levels of neurons in its structure. This method can be extended to solving complex problems with large databases and having a large number of dimensions. Other work can be performed for linear regression problems using other activation functions derived from ReLU and other actual databases commonly shared by the machine learning community. Future works may investigate whether the wavelet transform can work on models with more than one hidden layer, facilitating the assignment of weights in the following layers based on the responses obtained in the hidden layer.

## References

1. Avci, E., Coteli, R.: A new automatic target recognition system based on wavelet extreme learning machine. *Expert Syst. Appl.* **39**(16), 12340–12348 (2012)
2. Bache, K., Lichman, M.: UCI machine learning repository (2013)
3. de Campos Souza, P.V., Araujo, V.S., Guimaraes, A.J., Araujo, V.J.S., Rezende, T.S.: Method of pruning the hidden layer of the extreme learning machine based on correlation coefficient. In: 2018 IEEE Latin American Conference on Computational Intelligence (LA-CCI), pp. 1–6, November 2018. <https://doi.org/10.1109/LA-CCI.2018.8625247>
4. Cao, J., Lin, Z., Huang, G.B.: Composite function wavelet neural networks with extreme learning machine. *Neurocomputing* **73**(7–9), 1405–1416 (2010)

5. Chacko, B.P., Krishnan, V.V., Raju, G., Anto, P.B.: Handwritten character recognition using wavelet energy and extreme learning machine. *Int. J. Mach. Learn. Cybern.* **3**(2), 149–161 (2012)
6. Daubechies, I.: The wavelet transform, time-frequency localization and signal analysis. *IEEE Trans. Inf. Theory* **36**(5), 961–1005 (1990)
7. Deo, R.C., Tiwari, M.K., Adamowski, J.F., Quilty, J.M.: Forecasting effective drought index using a wavelet extreme learning machine (W-ELM) model. *Stochast. Environ. Res. Risk Assess.* **31**(5), 1211–1240 (2017)
8. Ding, S., Zhang, J., Xu, X., Zhang, Y.: A wavelet extreme learning machine. *Neural Comput. Appl.* **27**(4), 1033–1040 (2016)
9. Golub, G., Kahan, W.: Calculating the singular values and pseudo-inverse of a matrix. *J. Soc. Ind. Appl. Math. Ser. B: Numer. Anal.* **2**(2), 205–224 (1965)
10. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: theory and applications. *Neurocomputing* **70**(1–3), 489–501 (2006)
11. Javed, K., Gouriveau, R., Zerhouni, N.: SW-ELM: a summation wavelet extreme learning machine algorithm with a priori parameter initialization. *Neurocomputing* **123**, 299–307 (2014)
12. Karlik, B., Olgac, A.V.: Performance analysis of various activation functions in generalized MLP architectures of neural networks. *Int. J. Artif. Intell. Expert Syst.* **1**(4), 111–122 (2011)
13. Kuang, Y., Wu, Q., Shao, J., Wu, J., Wu, X.: Extreme learning machine classification method for lower limb movement recognition. *Cluster Comput.* **20**(4), 3051–3059 (2017)
14. Li, B., Cheng, C.: Monthly discharge forecasting using wavelet neural networks with extreme learning machine. *Sci. China Technol. Sci.* **57**(12), 2441–2452 (2014)
15. Li, R., Wang, X., Lei, L., Song, Y.:  $l_{\cdot}\{21\}$ -norm based loss function and regularization extreme learning machine. *IEEE Access* **7**, 6575–6586 (2019)
16. Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: Proceedings of ICML, vol. 30, p. 3 (2013)
17. Martínez-Martínez, J.M., Escandell-Montero, P., Soria-Olivas, E., Martín-Guerrero, J.D., Magdalena-Benedito, R., Gómez-Sanchis, J.: Regularized extreme learning machine for regression problems. *Neurocomputing* **74**(17), 3716–3721 (2011)
18. McDonnell, M.D., Tissera, M.D., Vladusich, T., Van Schaik, A., Tapson, J.: Fast, simple and accurate handwritten digit classification by training shallow neural network classifiers with the ‘extreme learning machine’ algorithm. *PLoS ONE* **10**(8), e0134254 (2015)
19. Miche, Y., Sorjamaa, A., Bas, P., Simula, O., Jutten, C., Lendasse, A.: OP-ELM: optimally pruned extreme learning machine. *IEEE Trans. Neural Netw.* **21**(1), 158–162 (2010)
20. Neal, R.M.: Bayesian Learning for Neural Networks, vol. 118. Springer, Heidelberg (2012)
21. Peck, C.C., Sheiner, L.B., Nichols, A.I.: The problem of choosing weights in non-linear regression analysis of pharmacokinetic data. *Drug Metab. Rev.* **15**(1–2), 133–148 (1984)
22. Pinto, D., Lemos, A.P., Braga, A.P., Horizonte, B., Gerais-Brazil, M.: An affinity matrix approach for structure selection of extreme learning machines. In: Proceedings, p. 343. Presses universitaires de Louvain (2015)

23. Wipf, D.P., Nagarajan, S.S.: A new view of automatic relevance determination. In: Platt, J.C., Koller, D., Singer, Y., Roweis, S.T. (eds.) Advances in Neural Information Processing Systems 20, pp. 1625–1632. Curran Associates, Inc. (2008). <http://papers.nips.cc/paper/3372-a-new-view-of-automatic-relevance-determination.pdf>
24. Zeng, Y., Xu, X., Fang, Y., Zhao, K.: Traffic sign recognition using deep convolutional networks and extreme learning machine. In: He, X., et al. (eds.) IScIDE 2015. LNCS, vol. 9242, pp. 272–280. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-23989-7\\_28](https://doi.org/10.1007/978-3-319-23989-7_28)



# Students' Performance Prediction Model Using Meta-classifier Approach

Hasniza Hassan<sup>1</sup>, Syahid Anuar<sup>1()</sup>, and Nor Bahiah Ahmad<sup>2</sup>

<sup>1</sup> Universiti Teknologi Malaysia, 81310 Johor Bahru, Johor, Malaysia  
niezal212@gmail.com, syahid.anuar@utm.my

<sup>2</sup> Universiti Teknologi Malaysia, 54100 Kuala Lumpur, Malaysia  
bahiah@utm.my

**Abstract.** Students' performance is vitally important at all stages of education, particularly for Higher Education Institutions. One of the most important issues is to improve the performance and quality of students enrolled. The initial symptom of at-risks' students need to be observed and earlier preventive measures are required to be carried out so as to determine the cause of students' dropout rate. Hence, the purpose of this research is to identify factors influencing students' performance using educational data mining techniques. In order to achieve this, data from different sources is employed into a single platform for pre-processing and modelling. The design of the study is divided into 6 different phases (data collection, data integration, data pre-processing such as cleaning, normalization, and transformation, feature selection, patterns extraction and model optimization as well as evaluation. The datasets were collected from a students' information system and e-learning system from a public university in Malaysia, while sample data from the Faculty of Engineering were used accordingly. This study also employed the use of academic, demographical, economical and behaviour e-learning features, in which 8 different group models were developed using 3 base-classifiers; Decision Tree, Artificial Neural Network and Support Vector Machine, and 5 multi-classifiers; Random Forest, Bagging, AdaBoost, Stacking and Majority Vote classifier. Finally, the highest accuracy of the classifier model was optimized. At the end, new Students' Performance Prediction Model was developed. The result proves that combination demographics with behaviour using a meta-classifier model with optimized hyper parameter produced better accuracy to predict students' performance.

**Keywords:** Student performance prediction · Educational data mining · E-learning · Higher education · Classification · Ensemble model

## 1 Introduction

### 1.1 A Subsection Sample

In the educational field, data mining or also known as Knowledge Discovery in Databases (KDD) has been widely applied to solve many educational problems. It is used to discover potential original information from massive numbers of data.

Educational data mining is employed to discover unique kind of data from educational databases, which is then used to understand the students.

In achieving its goal, educational data mining exploits multiple level of hierarchy in educational data. This is done with integration of methods from machine learning and data mining literature (Baker 2010). A statistical method, machine learning and data mining are 3 components that have been used widely in educational data mining to mine varieties of educational data. As such, the use of educational data mining method has been employed to develop and discover a unique type of data in educational settings and gain better understanding on how students learn (Romero and Ventura 2010).

There are many techniques being proposed to do prediction and analysis of students in previous studies. The most frequently used technique is data mining (Shahiri et al. 2015) which is a process of discovering hidden knowledge from databases or data warehouse. It is also the process of extracting useful information from a massive number of databases. Nowadays, techniques of data mining are widely used in many fields, including education to predict future situations.

However, the learning of analytical and educational data mining is new field of study in education. Learning analytic is a process of analyzing massive number of educational data. This involves the prediction of forecasting future performance and recognizing the risk. Educational data mining is a process that entails analyzing and understanding students' data using student performance prediction (Kavitha and Raj 2017). Despite many studies about learning analytic in higher education institutions within the last several years, it still remains an emerging field of education (Nunn et al. 2016).

This is because new exploration and finding about students' learning behaviour and factor contributing to students' performance need to be done for public benefit. However, determining hidden knowledge from students' data may require identifying some important elements, such as parameters, data mining techniques and tools to develop accurate models (Ahmad et al. 2015).

As such, there are big gaps regarding the size and amount of data required in educational data mining research. According to Márquez-Vera et al. (2016), only few studies on education dropout rates had been conducted, and most of them used statistical methods, rather than data mining techniques. Xu et al. (2017) said that there are still lack of studies about predicting student's performance in completing undergraduate programs, particularly about students' background and courses. In addition, courses are not informative to make accurate predictions and evolving progress.

Therefore, this study combines features of students' academic, demographical, economical and e-learning behaviour factors that contribute to higher accuracy to form a unique Students' Performance Prediction, using the Meta-Classifier Model. This study is guided by the following research questions, which need to be answered:

1. What are the most important features that influence students' performance?
2. Is data processing able to manage, and improve the classifiers used in building prediction model?
3. What is the most significant combination of classifiers to build students' performance prediction model?

4. Do fine tuning hyper parameters help in improving ensemble classifier performance?
5. How is the accuracy when combining 3 categories of data?

This study is organized as follows:

Section 2 explains about the methodology, which involves data collection and integration, pre-processing, tools, machine learning classifiers as well as the technique employed in the experiment. On the other hand, Sect. 3 explains about experimental process, machine learning classifiers and the technique used in gaining generalized and optimum results. Section 4 presents the comparison classifier results and propose a prediction model using combined classifiers. Finally, Sect. 5 concludes the study and suggests future research opportunity.

## 2 Methodology

### 2.1 Data Collection and Integration

This empirical study is carried out by collecting secondary data from student information system and a public university in Malaysia. This study only used 4413 rows of students' data of the Faculty of Engineering. The data content of students in this study includes academic information, demographics, economic and students' behaviour of using online learning.

The 2 types of datasets collected from 2 sources in this study are as follows:

1. Dataset from student information system
2. Data from e-learning (logfile)

Data set collected stored in MYSQL database. There are different tables created for each dataset and a new table was formed to consolidate all important data. The table contains 43 features of students from student information system and another 7 columns were created to insert the students' data from e-learning log file. E-learning features are based on the highest ranking from the log file, while query involves the use of MySQL. Student information system and e-learning data were consolidated into the new table for the experiment. Handling massive unnecessary data to retrieve important data was the most tedious process in this study. Finally, to consolidate the log data from e-learning, simple web apps were developed using PHP programming tool.

### 2.2 Experiment Setup

For this study, platform and software that were employed are as below:

1. PHP 7, Apache 2.4
2. Database: MySQL 5
3. Jupyter Notebook 5.7.0
4. Python 3.7.7
5. Windows 10 platform operating system with 8 GB RAM, Corei7, SSD.

The process of providing the correct data is important as data will reflect the modelling accuracy and give optimum result when used to predict future data. In data

mining, this step is called pre-processing data and is executed after the data have been integrated. In pre-processing using Python, 5 steps are taken as follow:

1. Data cleaning, normalize and transform
2. Data reduction
3. Modify categorical data to numerical
4. Data scaling
5. Feature selection

### **2.3 Standard Grading**

In this study, standard grading methods from the public university is employed as guidance in grouping the class label to multi-class. According to Marbouti et al. (2016), looking from the educational perspective, misidentifying student who achieves a C grade is better than D grade, as the grade C student has big potential to fail. Therefore, the 3 categories used in this study are; B- with CGPA below 2.67 as LOW, B- with CGPA 2.67 and above as MODERATE, while grade A with CGPA above 3.67 is considered EXCELLENT. These 3 categories of classes that are based on CGPA were created using transformation technique as explained in Sect. 3.

Conversely, in predicting students' performance, grade (CGPA, GPA) remains the most influential features to determine students' survival. Many researchers (Natek and Zwilling 2014; Ahmad et al. 2015; Iam-On and Boongoen 2017; Kondo et al. 2017; Fernandes et al. 2018) have used these features, as they are the benchmark to indicate future education and career mobility (Shahiri et al. 2015). Studies by Barhamzaid and Alleyne (2018) also found three factors that give impact on student's performance in the first accounting course under political conflict, which are CGPA, high school grade, and high school branch.

### **2.4 Machine Learning Classifier Technique**

Classifier is a machine learning algorithm that uses classification technique. There are many base-classifiers in machine learning, such as Decision Tree, Artificial Neural Network, Support Vector Machine, Naïve Bayes and many more.

Decision Tree is one of the commonly used machine learning methods, as it is simple and easy to understand. It has good speed and non-linear capability in training and testing. It is also the best in handling noise through its information gain process (Adejo et al. 2018).

ANN is a machine learning method that solves problem like a human brain, works as neurons cell in the human brain, and capable to detect all possible connections between features, as well as easily recognizing non-linear relationship. It also can handle a small number of data (Amrieh et al. 2016).

SVM is one of machine learning methods that can manage to analyse and recognize patterns in data for both regression and classification. It is also known as binary classifier, by managing to learn various combinations of features with less computational complexity as it supports Kernel's function. Thus, using SVM, the problem with overfitting can be solved (Salini et al. 2018).

Artificial Neural Network and Decision Tree are two data mining methods that are highly used by researchers for predicting students' performance (Shahiri et al. 2015). Therefore, these two base classifiers, together with Support Vector Machine are employed in this study. Conversely, 5 ensemble classifiers; RF, Bagging, AdaBoost, Stacking and Ensemble Vote are employed and compared.

Ensemble meta-classifier is a combination of two or more base-classifiers technique used in developing better machine learning models and solve classification problems (Polikar et al. 2008). It is a type of supervised learning technique that combines multiple weak learners to produce a strong vital learner. Ensemble learner is also known as multiple classifier system, that is based on multiple learning models which has been used to solve problem like the classification problem (Gudivada et al. 2016).

## 2.5 Performance Metric

Performance Metric plays an important role to gain optimal result during training classifier (Hossin and Sulaiman 2015). This study adopts few performance evaluation metrics including accuracy, confusion metric, precision, recall, F-measures and classification error. Multi-class confusion metric has also been used as the output for multi-class label. There are 4 label features used by most performance metrics. These are; (i) True Positive (TP), when data are actually positive, (ii) True Negative (TN) is instance when the number is negative and are predicted to be negative, (ii) False Positive (FP), is when it is actually negative but are predicted to be positive for false group instances and (iv) False Negative (FN) is when it is actually positive but predicted to be positive (AL-Malaise et al. 2014). Accuracy, precision, recall and F-measure use all these label features to calculate results based on the following formula:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

$$\text{Precision} = \frac{TP}{FP + TP} \quad (2)$$

$$\text{Recall} = \frac{TP}{FN + TP} \quad (3)$$

$$F\text{measure} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4)$$

## 3 Pre-processing and Modeling

### 3.1 Pre-processing

The first step of preprocessing is to remove data that are obviously not useful for data mining. By using Python pre-processing, some features have been dropped from the datasets, such as an ID table for features, codes, field of studies, session and year. From

43 features of student information system data, 17 are IDs features and 13 features are not relevant. Therefore, only remaining features were considered for the next step.

After irrelevant data were removed, the next step was cleaning noises and outlier. This study uses statistical methods to replace noise data to mean value. Characteristics of noisy or incomplete data such as missing value, empty column and outlier were used to replace and improve the quality of those data sets.

From all selected features, the data were categorized into 3 groups (demographical and economical, academic background features as well as behaviour features respectively). All these features were employed in the next step to rank the features using an algorithm. The features description as in Table 1:

**Table 1.** Features description

Category	Feature	Description
Academic background features	Study_method	Coursework/research
	Program	Programme of study
	CGPA	Cumulative Grade Point Average
	Year_intake	Year intake
	Education_mode	Part time/Full time
Demographical/socio-economic features	Family_income	Range of family income
	Student_status	Student status
	Scholarship	Name of Scholarship
	Gender	Student Gender
	Age	Student Age
	Nationality	Student Nationality
	Disability	Disability status
Behaviour features	Login	Count no of login
	Course_viewed	Count course viewed
	Resource_viewed	Count resources viewed
	Forum_viewed	Count forum viewed
	Course_submitted	Count course submitted
	Assignment_viewed	Count assignment viewed
	Assignment_submitted	Count assignment submitted

### 3.2 Feature Selection

Random Forest Regressor is an algorithm for unsupervised learning that has been used to select feature's importance. This step is used to identify a list of features that may contribute to high accuracy in the prediction model. 13 features importance used to run classifiers. Course\_viewed is at the highest ranking, while taraf\_warga (nationality) is at the lowest rank.

3 classes for label was produced when transforming the CGPA data based on a standard grading as explained in Sect. 2.3. However, there are issues of multiclass

problem, where most of the metric developed are not for multiclass problem. As such, it is suggested that future development of new metric or choosing a suitable metric should take this problem into consideration (Hossin and Sulaiman 2015).

### 3.3 Modeling

After pre-processing, the data were divided into 2 categories (train and test) with ratio 70:30. Thus, to model the data, cross-validation was used with k-fold = 10 for each model. The best way to evaluate the model is using future data. However, in the preliminary stage, cross-validation was most used to evaluate the model.

## 4 Result and Discussion

Table 2 shows the experimental result of 8 different bases and meta-classifiers models. The accuracy and precision for all models are lower than 80%, but the recall for each model are mostly greater than 90%. The rule of thumb for precision and the recall is that the higher the precision and recall, the better is the classifier, while the opposite is applicable to RMSE and classification error (Adejo et al. 2018).

In Table 2, model 1, 2 and 3 are trained base-classifiers models with 3 categories of data (student information system, e-learning and combination of student information system and e-learning). Model 4, 5 and 6 use meta-classifiers with 3 categories of data. The meta-classifier in these 3 models then employed the use of the same combination of base-classifier which is Decision Tree. Model 7 is Stacked Classification and it uses the combination of Random Forest (RF), Artificial Neural Network (ANN) and Support Vector Machine (SVM). Finally, model 8 is the ensemble vote meta-classifier model which is similar to that of model 7 but uses the combination rule majority vote with optimization hyperparameter.

Among the classifier used, the Decision Tree model gives the lowest accuracy when trained with 3 categories of data, while a combination of voting meta-classifier gives the highest accuracy when trained using combination of system and e-learning data. It seems that, voting meta-classifier has bright expectation that it is able to achieve better accuracy by tuning the hyperparameter in future work.

However, there are things that are required to be taken into consideration, like pre-processing of noise data as well as multi-balance problem that needs improvement. From the 3 categories data, combination of student information system (SIS) and behaviour e-learning (EL) data produced the highest accuracy result among all. 6 most common evaluation metrics used in this study are; Accuracy, Confusion Metric, Precision, Recall, F-Score and Classification Error.

Studies done by Amrieh et al. (2016), Marbouti et al. (2016), Iam-On and Boon-goen (2017), Adejo et al. (2018), Beemer et al (2018), Kostopoulos et al. (2018), Salini et al. (2018), as well as Wanjau and Muketha (2018) developed students' performance prediction model using ensemble technique and compared the performance with variety of single learning model. The results proved that ensemble techniques produce tremendously better performance prediction of accuracy.

**Table 2.** Comparing different classifiers using different features group

Model	Algo	Features	Accuracy (%)/10 KFV	RMSE (Wg)	Precision (Wg)	Recall (Wg)	F-score (Wg)	Classification error (%)
Model 1: Base-Classifier (BC I)	DT	SIS	69.73	98.88	0.73	0.96	0.83	28.98
	DT	EL	61.37	109.68	0.75	0.76	0.76	37.89
	DT	SIS + EL	65.98	106.96	0.75	0.77	0.76	35.82
Model 2: Base-Classifier (BC II)	ANN	SIS	69.73	98.55	0.73	0.96	0.83	28.58
	ANN	EL	74.44	93.67	0.75	0.97	0.84	26.08
	ANN	SIS + EL	71.25	95.18	0.76	0.90	0.83	27.54
Model 3: Base-Classifier (BC III)	SVM	SIS	71.73	98.02	0.72	1	0.84	28.26
	SVM	EL	72.77	96.68	0.57	0.11	0.18	27.07
	SVM	SIS + EL	71.73	98.11	0.70	0.72	0.63	27.94
Model 4: Meta-Classifier (MC I)-RF	RF	SIS	71.01	98.02	0.73	0.97	0.83	28.26
	RF	EL	73.64	98.31	0.74	0.95	0.83	27.86
	RF	SIS + EL	74.12	92.42	0.73	0.75	0.68	25.23
Model 5: Meta-Classifier (MC II)- Bagging	BGG	SIS	69.81	98.55	0.73	0.96	0.83	28.58
	BGG	EL	72.68	102.28	0.74	0.91	0.82	29.85
	BGG	SIS + EL	73.08	93.79	0.76	0.94	0.84	25.87
Model 6: Meta-Classifier (MC III)- Boosting	BOO	SIS	70.87	95.27	0.74	0.98	0.84	26.56
	BOO	EL	68.55	102.59	0.74	0.90	0.81	30.25
	BOO	SIS + EL	65.35	106.25	0.75	0.78	0.77	35.27
Model 7: Meta-Classifier (MC IV)- Stacking	ANN, RF & SVM	SIS	71.8	98.83	0.73	0.97	0.83	28.66
		EL	71.4	98.15	0.74	0.95	0.65	27.78
		SIS + EL	72.8	96.8	0.76	0.90	0.83	27
Model 8: Meta-Classifier (MC V) – Majority Vote	ANN, RF & SVM	SIS	70.29	97.7	0.73	0.97	0.83	28.1
		EL	72.69	98.15	0.74	0.95	0.83	27.78
		SIS + EL	75.56	94.89	0.76	0.95	0.84	26.03

According to Tamhane and Appleton (2014) in their preliminary study, behaviour features are not strong predictor for students' performance. As such, further work is needed to be carried out to investigate how the behaviour of data is collected, transform and interpreted. However, Amrieh et al. (2016) proved that behaviour features contribute to improve accuracy in their proposed model. This is because they discovered that there is a strong relationship between behaviour and academic features. Moreover, Nam et al. (2017) and Zollanvari et al. (2017) also discovered similar finding by saying that behaviour features in mining students' performance data might increase performance accuracy in the students' prediction performance model. This is the reason the current study employed behaviour features from e-learning together with demographic, academic and socioeconomic features using multiple ensemble classifiers to discover which is the best meta-classifier that performs well and suitable for the collected data to predict students' performance.

Even though many studies discovered pedagogical factors and students' performance issues, only a few research concentrated on specific areas (Anoopkumar and Rahman 2018). On the other hand, Fernandes et al. (2018) study showed that grades and absence are two features that are mostly relevant to predict the end of year academic outcome of student performance. They also stated that for demographic features, neighbour, schools and age are 3 potential indicators influencing the academic success or failure.

The influencing factor of students' dropout rate has become a good study subject in the past many years. Hence, student's behaviour during the enrolment of a course, and institutional characteristics are two factors area that have been discovered to increase the chances of completing many researches (Lopez Guarin et al. 2015).

## 5 Conclusion and Future Work

In the education field, academic is the core component where students' academic achievement needs to be improved by all higher educational institutions. Students' academic information, profile and LMS are 2 types of students' data that contain beneficial information and tremendously useful information that could be interpreted as knowledge. However, there is still lack of study to discover patterns and build students' performance prediction model using ensemble techniques even though the technique is proven to give higher accuracy.

In this study, students' performance prediction model using ensemble meta-classifiers to predict students' performance proved to produce high accuracy. Thus, 3 base-classifiers and 5 meta-classifiers were developed and compared. It is also proven that the combination of data of student information system and students' behaviour from e-learning produce the highest result when trained using Majority Vote Ensemble Classifier.

Future step of this study would be to investigate best feature selection methods to handle the massive number of data and optimize model by fine tuning hyperparameters to get optimum result.

**Acknowledgements.** The authors are grateful to Research Management Centre (RMC), Universiti Teknologi Malaysia (UTM) for the financial support under Tier 2 Research University Grant (Q.K130000.2638.14J88).

## References

- Adejo, O.W., Connolly, T., Adejo, O.W., Connolly, T.: ensemble approach Predicting student academic performance using multi-model heterogeneous ensemble approach. *J. Appl. Res. High. Educ.* **10**(1), 61–75 (2018)
- Ahmad, F., Ismail, N.H., Aziz, A.A.: The prediction of students' academic performance using classification data mining techniques. *Appl. Math. Sci.* **9**(129), 6415–6426 (2015)
- AL-Malaise, A., Malibari, A., Alkhozae, M.: Students performance prediction system using multi agent data mining technique. *Int. J. Data Min. Knowl. Process* (2014)

- Amrieh, E.A., Hamtini, T., Aljarah, I.: Mining educational data to predict student's academic performance using ensemble methods. *Int. J. Database Theor. Appl.* **9**(8), 119–136 (2016)
- Anoopkumar, M., Zubair Rahman, A.M.J.Md.: Model of tuned J48 classification and analysis of performance prediction in educational data mining. *Int. J. Appl. Eng. Res.* **13**(20), 14717–14727 (2018). ISSN 0973-4562
- Baker, S.J.R.: Data mining for education. *Int. Encycl. Educ.* (2010)
- Barhamzaid, Z.A.A., Alleyne, A.: Factors affecting student performance in the first accounting course in diploma program under political conflict. *J. Educ. Prac.* **9**(24), 144–154 (2018)
- Beemer, J., Spoon, K., He, L., Fan, J., Levine, R.A.: Ensemble learning for estimating individualized treatment effects in student success studies. *Int. J. Artif. Intell. Educ.* **28**(3), 315–335 (2018)
- Fernandes, E., Holanda, M., Victorino, M., Borges, V., Carvalho, R., Van Erven, G.: Educational data mining: predictive analysis of academic performance of public school students in the capital of Brazil. *J. Bus. Res.* **94**, 335–343 (2018)
- Gudivada, V.N., Irfan, M.T., Fathi, E., Rao, D.L.: Cognitive Analytics: Going Beyond Big Data Analytics and Machine Learning. *Handbook of Statistics*, 1st edn. Elsevier B.V (2016)
- Hossin, M., Sulaiman, M.N.: A review on evaluation metrics for data classification evaluations. *Int. J. Data Min. Knowl. Process (IJDKP)* **5**(2), 1–11 (2015)
- Iam-On, N., Boongoen, T.: Improved student dropout prediction in Thai University using ensemble of mixed-type data clusterings. *Int. J. Mach. Learn. Cyber.* **8**(2), 497–510 (2017)
- Kavitha, G., Raj, L.: Educational data mining and learning analytics educational assistance for teaching and learning. *Int. J. Comput. Organ. Trends* **41**(1), 21–25 (2017)
- Kondo, N., Okubo, M., Hatanaka, T.: Early detection of at-risk students using machine learning based on LMS log data. In: 2017 6th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI), pp. 198–201 (2017)
- Kostopoulos, G., Livieris, I.E., Kotsiantis, S., Tampakas, V.: CST-voting: a semi-supervised ensemble method for classification problems. *J. Intell. Fuzzy Syst.* **35**(1), 99–109 (2018)
- Lopez Guarin, C.E., Guzman, E.L., Gonzalez, F.A.: A model to predict low academic performance at a specific enrollment using data mining. *Revista Iberoamericana de Tecnologías del Aprendizaje* **10**(3), 119–125 (2015)
- Marbouti, F., Diefes-Dux, H.A., Madhavan, K.: Models for early prediction of at-risk students in a course using standards-based grading. *Comput. Educ.* **103**, 1–15 (2016)
- Márquez-Vera, C., Cano, A., Romero, C., Noaman, A.Y.M., Mousa Fardoun, H., Ventura, S.: Early dropout prediction using data mining: a case study with high school students. *Expert Syst.* **33**(1), 107–124 (2016)
- Nam, S.J., Frishkoff, G., Collins-Thompson, K.: Predicting students' disengaged behaviors in an online meaning-generation task. *IEEE Trans. Learn. Technol.* **13****82**, 1–14 (2017)
- Natek, S., Zwilling, M.: Student data mining solution-knowledge management system related to higher education institutions. *Expert Syst. Appl.* **41**(14), 6400–6407 (2014)
- Nunn, S., Avella, J.T., Kanai, T., Kebritchi, M.: Learning analytics methods, benefits, and challenges in higher education: a systematic literature review. *Online Learn.* **20**(2), 13–29 (2016)
- Polikar, R., et al.: An ensemble based data fusion approach for early diagnosis of Alzheimer's disease. *Inf. Fusion* **9**(1), 83–95 (2008)
- Romero, C., Ventura, S.: Educational data mining: a review of the state of the art. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **40**(6), 601–618 (2010)
- Salini, A., Jeyapriya, U., College, S.M., College, S.M.: A majority vote based ensemble classifier for predicting students academic performance. *Int. J. Pure Appl. Math.* **118**(24), 1–11 (2018)

- Shahiri, A.M., Husain, W., Rashid, N.A.: A review on predicting student's performance using data mining techniques. In: 2015 3rd Information Systems International Conference, pp. 414–422 (2015)
- Tamhane, A., Appleton, J.: Predicting student risks through longitudinal analysis. In: KDD, pp. 1544–1552 (2014)
- Wanjau, S.K., Muketha, G.M.: Improving student enrollment prediction using ensemble classifiers. *Int. J. Comput. Appl. Technol. Res.* **7**(03), 122–128 (2018)
- Xu, J., Moon, K.H., Van Der Schaar, M.: A machine learning approach for tracking and predicting student performance in degree programs. *IEEE J. Sel. Top. Signal Process.* **11**(5), 742–753 (2017)
- Zollanvari, A., Kizilirmak, R.C., Kho, Y.H., Hernandez-Torrano, D.: Predicting students' GPA and developing intervention strategies based on self-regulatory learning behaviors. *IEEE Access* **5**, 23792–23802 (2017)

# **Deep Learning**



# A Deep Network System for Simulated Autonomous Driving Using Behavioral Cloning

Andreea-Iulia Patachi<sup>1</sup>, Florin Leon<sup>1()</sup>, and Doina Logofătu<sup>2</sup>

<sup>1</sup> Department of Computer Science and Engineering,

“Gheorghe Asachi” Technical University of Iași, Iași, Romania

patachiandreea@yahoo.com, florin.leon@tuiasi.ro

<sup>2</sup> Faculty of Computer Science and Engineering,

Frankfurt University of Applied Sciences, Frankfurt, Germany

logofatu@fb2.fra-uas.de

**Abstract.** This paper studies the performance of a convolutional neural network (CNN) trained to learn the behavior of a vehicle using data from a simulator that allows real-time information gathering from vehicle chassis, machine position and speed. The network uses information from the front-facing, right and right cameras, the car’s position on the lane and its speed. This approach proves to be quite effective: with a minimum of driving time taken directly from proper driving simulations in the form of a game, the system learns to drive on a marked strip road. The network automatically learns the internal representations of the necessary processing steps, such as the detection of useful road features, required speed, and track position. Different types of activation functions are used, and it is noticed that the exponential linear unit (ELU) activation function leads to improved learning compared to other activation functions.

**Keywords:** Deep learning · Convolutional neural network · Behavioral cloning · Autonomous driving · Simulator

## 1 Introduction

Autonomous car technology is already being developed by many companies on different types of vehicles. The complete driverless system is still at an advanced testing phase, but partially automated systems have been around in the automotive industry for the last few years.

Using simulation, one can create realistic situations to train neural networks. There are virtual simulation platforms available on the market have been used for many years for driver assistance. Various functionalities such as auto emergency braking, cruise control, are typically tested using virtual simulation platforms. With the ongoing race to deploy fully autonomous cars, virtual simulation platforms are gaining popularity, now more than ever. Most car producers have disclosed the use of virtual platforms for modeling the environment and testing their systems. They are made more robust by varying the simulation dynamics such as weather, lighting, or object behavior. Simulators such as *CarMaker*, *Udacity-self driving car* and *CARLA* are capable to create the desired environment with vehicles, traffic signs, pedestrians, different types of sensors and weather conditions.

Behavioral cloning is a method in which human abilities can be transferred into a computer program. Since the human subject performs a task, its actions are recorded with the situation that has generated the action. A log of these records is used as an input to a learning system that can eventually reproduce the desired behavior. This approach can be used to construct automatic control systems for complicated tasks, for which classical control theory is difficult to apply or even incomplete.

We organize our paper as follows. In Sect. 2 presents a selection of related work about autonomous driving research and applications. In Sect. 3 the learning process of the present model is detailed, from data collection and processing to the architecture of the convolutional neural network and individual neuron characteristics such as activation functions. Section 4 describes some experimental results, while Sect. 5 contains the conclusions of our work.

## 2 Related Work

Convolutional neural networks (CNNs) [1] have revolutionized pattern recognition [2]; prior to large-scale adoption of CNNs, most pattern recognition projects were accomplished using an initial stage of hand-crafted component extraction followed by a classifier. The progress of CNNs is that features are learned automatically from training examples. A CNN is particularly effective in image recognition tasks because the convolution operation captures the 2D nature of images. Also, by applying the convolution kernels to examine an entire image, fewer parameters need to be learned in comparison with the total number of operations [3]. While CNNs with learned features have been in commercial use for over twenty years [4], their adoption has exploded in the last few years because of two recent developments. First, large, labeled data sets such as the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [5] have become accessible for training and validation [3]. Second, CNN learning algorithms have been implemented on the parallel graphics processing units (GPUs) which accelerate learning [3].

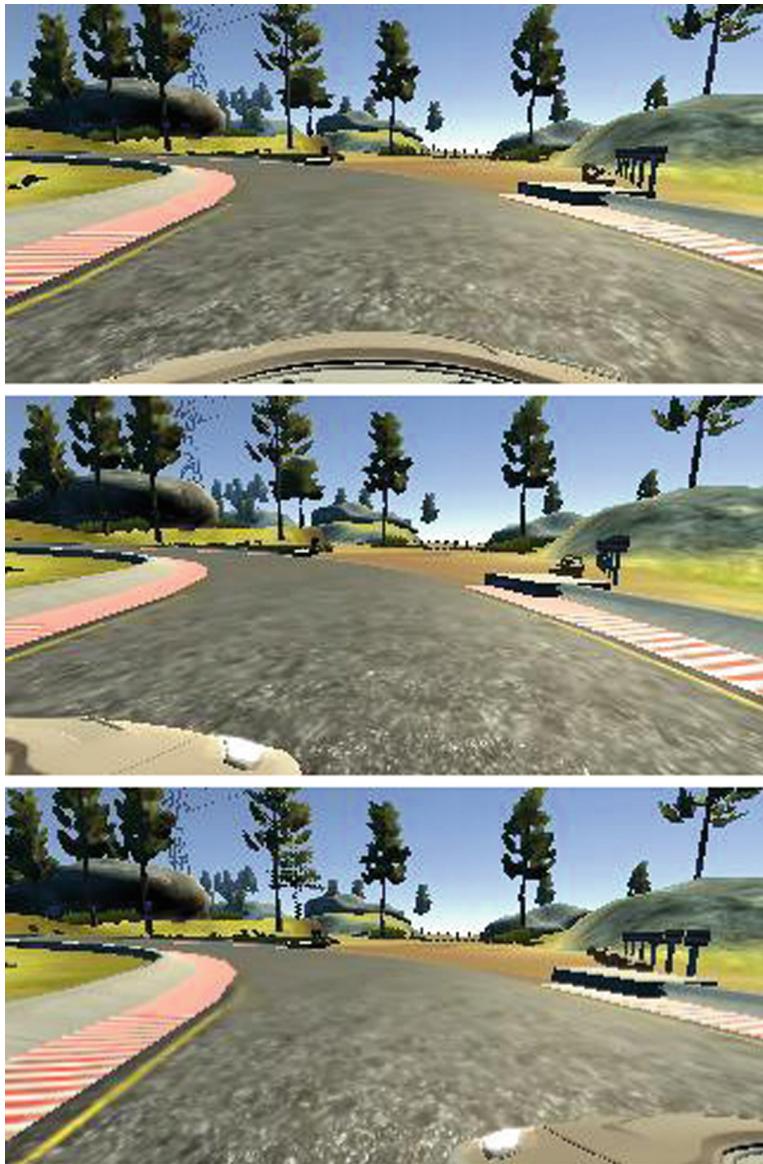
DARPA Autonomous Vehicle (DAVE) [6] demonstrated the potential of end-to-end learning and was used to justify starting the DARPA Learning Applied to Ground Robots (LAGR) program [7]. However, DAVE's achievement was not reliable enough to support a full alternative to more modular approaches to off-road driving: its average space between crashes was about 20 m in complicated environments. Recently, a new application has been started at NVIDIA, which aims to build on DAVE and create a strong system for driving on public roads. The basic motivation for this project is to avoid the need to identify specific human-designated features, such as lane markings, guardrails, or other cars, and to prevent having to create a collection of “if-then-else” rules, based on the perception of these features [3].

DAVE-2 [6] was inspired by the pioneering work of Pomerleau [8] who built the Autonomous Land Vehicle in a Neural Network (ALVINN) system in 1989. It proves that an end-to-end trained neural network can steer a vehicle on public roads [3].

Many major companies are involved in developing self-driving cars. Among those who are currently testing such vehicles on public roads, one can mention, among others, Google, Tesla, Toyota, BMW, Nissan, Ford [9].

### 3 Simulated Driving Using Human Behavioral Cloning

In this paper, we build a CNN that goes beyond pattern recognition and learns the behavior of a vehicle.



**Fig. 1.** Pictures from center, left and right cameras

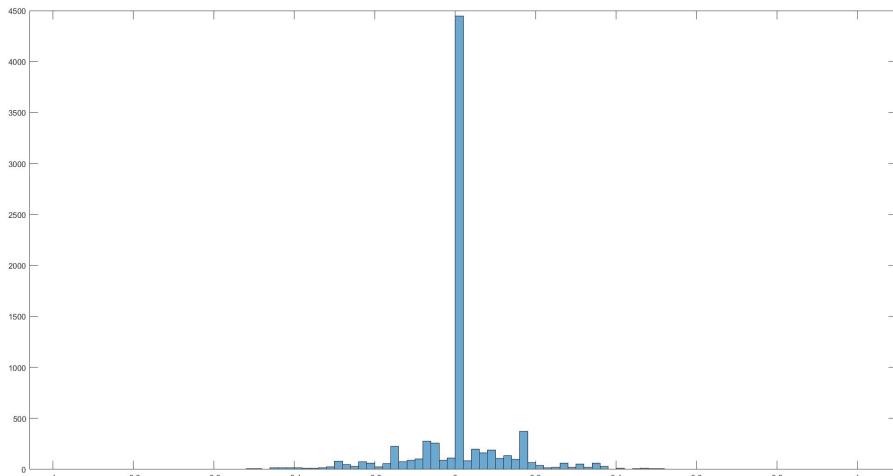
### 3.1 Data Collection

Training data was collected by driving the car on the flat road track in the simulator. The performance of the network can then be checked by letting the car drive autonomously on the same or other tracks. In Fig. 1, one can see some pictures from the different cameras on the vehicle while driving.

### 3.2 Unbalanced Data

During the driving of the vehicle on the road, most often the steering angle is very close to zero, and this can clearly be observed in the training data. In Fig. 2, one can see the histogram of steering angles that were registered while driving the vehicle around the path and staying as close as possible to the middle of the roadway. This is all the data that was collected for training the final model, by driving 4–5 times.

A major problem is the left/right distortion that is caused by driving the vehicle around the track in one direction. This can be solved by rotating each recorded image and its equivalent steering angle. More problematic is the bias of straight driving: there are rare cases when the car drives straight even if the road is curved, and then it makes a sudden turn (i.e. a high steering angle) to remain on the road. One possible way to solve this issue would be to let the vehicle drift to the edge of the road and get back before a crash occurs. Another solution for this would be to test events with extreme angles more often than small angle events. However, because they happen extremely rarely, it may be necessary to collect a large number of training data for the model, in order to avoid overfitting. The best decision was to simulate all recovery events. For training, the vehicle was driven as easily as possible in the middle of the road. The underlying motivation was always to achieve the ideal steering angle.



**Fig. 2.** Histogram of steering angles: the range  $[-1, 1]$

### 3.3 Data Augmentation

For training, the images from all three cameras, front, left and right, were used. The images taken from the side cameras are similar to the parallel translations of the vehicle. To take into account the out of the center situations, it was necessary to adjust the steering angle for the images taken from the side cameras as follows: when the distance from the far side camera to the center of the road was greater than 1.2 m, the car should smoothly turn and return to the middle of the road within the next 20 m. Thus, the steering angle correction should be about  $1.2/20 = 0.06$  radians (using  $\tan(\alpha) \approx \alpha$ ). This proved to be a powerful means to make the car avoid the edges of the road.

The images from the cameras include a part of the car hood. In the next step, the images were cropped in order to remove it. In order to increase the size of the training set, from the resulting image, 3 new images were generated by randomly choosing smaller areas along the  $x$ -axis:  $160 \times 320$  from the original  $160 \times 380$ . This process is



**Fig. 3.** Correction of the steering angle: from  $-2.8^\circ$  to  $-3.4^\circ$



**Fig. 4.** Correction of the steering angle: from  $-0.4^\circ$  to  $-1^\circ$

equivalent to a translation, and the steering angle needs to be adjusted accordingly to this change (Fig. 3). This has resulted in curved track pieces that appear as often in the training set as straight parts (Fig. 4). Finally, each image was horizontally flipped to make both left and right turns to appear as often as possible. Also, the brightness was adjusted at random.

### 3.4 Model Architecture and Training

The final model architecture consisted of a convolution neural network with the following layers and layer sizes, written in the notation typically used to describe this kind of architecture:

- Cropping from  $160 \times 320 \times 3$ ;
- Normalized input planes  $100 \times 320 \times 3$ ;
- $3@100 \times 320$  Convolution  $5 \times 5$  and Maxpooling  $3 \times 3$ ;
- $24@32 \times 106$  Convolution  $5 \times 5$  and Maxpooling  $2 \times 2$ ;
- $32@14 \times 51$  Convolution  $5 \times 5$  and Maxpooling  $2 \times 2$ ;
- $48@5 \times 24$  Convolution  $3 \times 3$  and Maxpooling  $1 \times 1$ ;
- $64@3 \times 22$  Convolution  $3 \times 3$  and Maxpooling  $1 \times 1$ ;
- $64@1 \times 20$  Flatten 100;
- Flatten 50;
- Flatten 10;
- 1 real-valued output:  $1@1 \times 1$ .

The network architecture contains 9 layers, including a normalization layer, 5 convolutional layers and 3 fully connected layers. The first layer of the network performs image normalization. Performing normalization in the network allows the normalization scheme to be altered with the network architecture and to be accelerated via GPU processing.

The convolutional layers were designed to perform feature extraction and were chosen empirically through a series of experiments that varied layer configurations. We use strided convolutions in the first three convolutional layers with a  $3 \times 3/2 \times 2$  stride and a  $5 \times 5$  kernel and a non-strided convolution with a  $3 \times 3$  kernel size in the last two convolutional layers. We follow the five convolutional layers with three fully connected layers leading to an output value.

This is close to the NVIDIA DAVE-2 [3]. The main difference is the stride (3, 3) which allows to reduce the bigger input of our CNN  $100 \times 320$  to the given  $32 \times 106$  in an easy way. For this CNN we used the exponential linear unit (*ELU*) activation instead of a simple rectified linear unit (*ReLU*), because the area of nonlinearity is higher. The L2 regularizer parameter was set to 0.0001, in order to minimize the prediction error.

For the final results, we also studied different activation functions to see what is more appropriate to this problem, as explained in the following section.

### 3.5 Activation Functions

Presently, one of the most popular activation functions for deep neural networks is the rectified linear unit (*ReLU*), which was first recommended for restricted Boltzmann machines [1] and then used with success for neural networks [2]. The *ReLU* activation function is:

$$f_{ReLU}(x) = \max(0, x). \quad (1)$$

Apart from producing sparse codes, the main advantage of *ReLU* is that it does not have the vanishing gradient problem [4, 10] since the derivative for positive values is not contractive [2]. On the other hand, *ReLU* is non-negative and for that reason, have a mean activation greater than zero [11].

The hyperbolic tangent activation function (*tanh*) non-linearity compresses the input in the range  $(-1, 1)$ :

$$f_{tanh}(x) = (1 - e^{-2x}) / (1 + e^{-2x}). \quad (2)$$

It determines an output which is zero-centered. So, large negative values are mapped to negative outputs, and near zero-valued inputs are mapped to near zero outputs. The gradients for *Tanh* are high than those of the unipolar sigmoid, but it also experiences the vanishing gradient problem.

Another activation function that we used is the scaled exponential linear unit (*SELU*):

$$f_{SELU}(x) = \begin{cases} \lambda \cdot \alpha \cdot (e^x - 1), & x \leq 0 \\ \lambda \cdot x, & x > 0 \end{cases}, \quad (3)$$

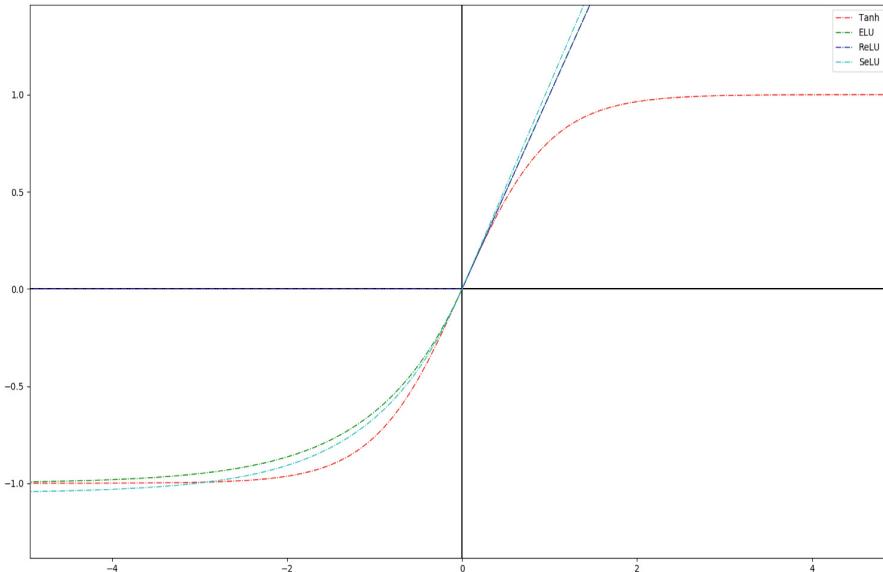
where  $\lambda = 1.0507$  and  $\alpha = 1.67326$  [12]. It has self-normalizing properties because the activations that are close to zero mean and unit variance, propagated through many network layers, will also converge towards zero mean and unit variance. This makes the learning highly robust and allows to train networks with many layers [13].

Because the learning can be made faster by centering the activations at zero, the exponential linear unit (*ELU*) also uses the activation function to achieve mean zero, which accelerates learning in deep neural networks and contributes to better learning performance:

$$f_{ELU}(x) = \begin{cases} \alpha \cdot (e^x - 1), & x \leq 0 \\ x, & x > 0 \end{cases}. \quad (4)$$

The hyperparameter  $\alpha$  controls the value to which an ELU saturates for negative net inputs. In the simplest case,  $\alpha = 1$  [11]. Like *ReLU* (with its variants such as *Leaky ReLU* or *Parametrized ReLU*), *ELU* relieves the vanishing gradient problem by using the identity for positive values.

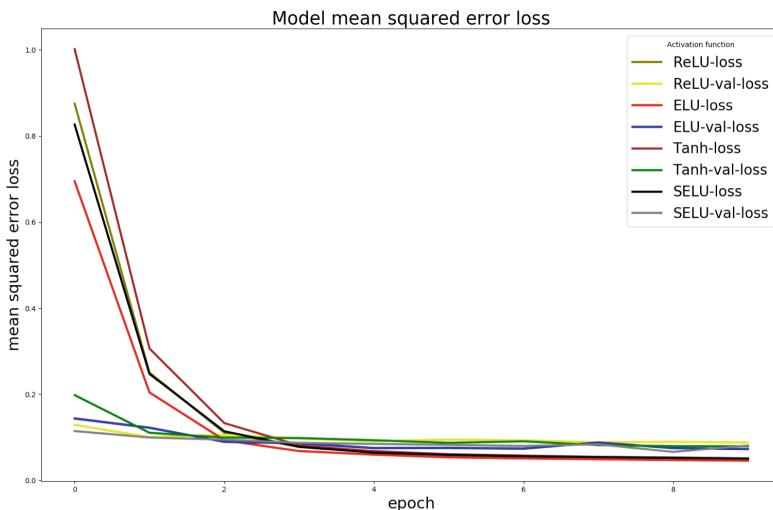
These activation functions are graphically displayed in Fig. 5.



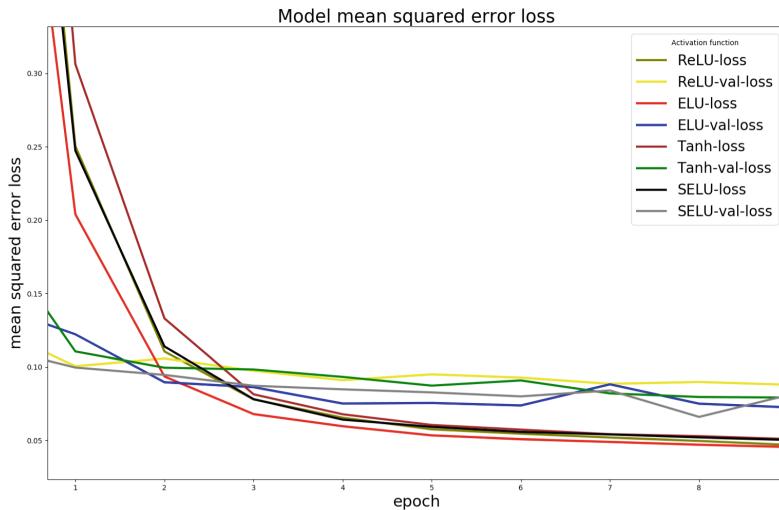
**Fig. 5.** Activation functions

## 4 Experimental Results

The results of behavioral cloning for the simulated driving scenario, implemented with a CNN using different activation functions are presented in Figs. 6 and 7. They are generally similar, but on closer inspection, one can notice that the *ELU* activation function leads to a smaller mean squared error loss both for the training and for the validation data.

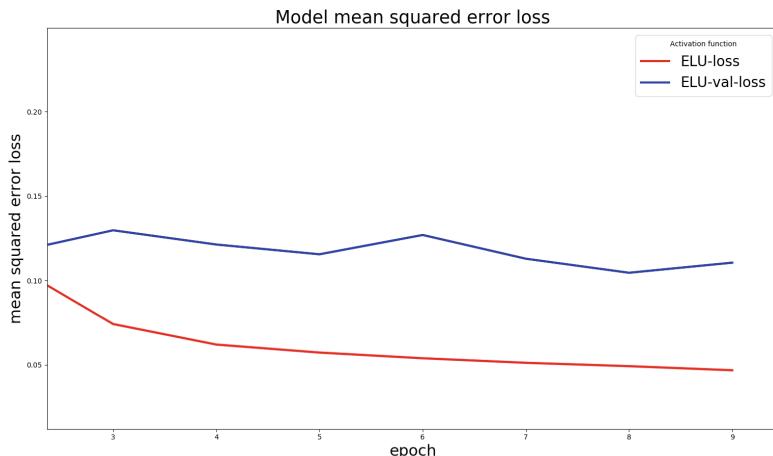


**Fig. 6.** Training mean squared error loss and data validation mean squared error loss

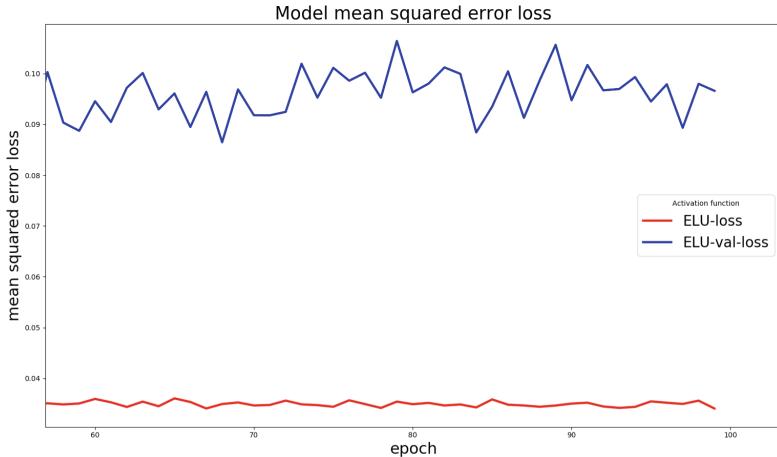


**Fig. 7.** Zoom on the relevant area of mean squared error loss

As shown in the legends, the “[activation-function]-loss” represents the loss obtained for the training set, while “[activation-function]-val-loss” represents the loss obtained for the validation set. An explanation for the best performance achieved by the *ELU* activation function is that, on the one hand, compared to *ReLU*, *ELU* has a non-zero gradient in the negative region, and on the other hand, compared to *Tanh*, *ELU* does not saturate its gradient in the farther positive region.



**Fig. 8.** The mean squared error loss obtained for the *ELU* activation function



**Fig. 9.** The evolution of the mean squared error loss for a greater number of training epochs

When considering only the behavior of the *ELU* activation function, Fig. 8 shows a clearer comparison between the results on the training and validation data. As expected, the performance for the training data is better, but overall the two values are quite close, and this signifies that the model has good generalization capabilities.

Figure 9 shows the evolution of the mean squared error loss for a greater number of training epochs, i.e. 100 instead of 10. One can see that there are variations in the training and validation losses, but the values remain generally stable and the performance improvement is not great compared to the scenario with only 10 epochs. Therefore, we can state that the behavioral cloning model can achieve good performance in a very small number of training epochs.

## 5 Conclusions

In this paper, we have experimentally shown that CNNs are capable to learn the entire task of lane and road following without manual decomposition into road or lane marking detection, path planning and control. A small amount of driving training data was sufficient to train the vehicle to operate on a road. The CNN can learn meaningful road features from a very sparse training signals, such as steering and speed. The system discovers, for example, to detect the outline of a road without the need for specific labels during training. The best performance is obtained by using the exponential linear unit (*ELU*) activation function.

As a future direction of research, the system can be improved such that it could learn to drive faster on difficult roads. To this end, further effort needs to be dedicated to the refinement of the network architecture and the proper selection of the training data.

In the current scenario, the car is a single agent in its environment and the main goal is to drive as close to the middle of the road as possible. The next endeavors should also

address the situations with more traffic participants and the introduction of corrective actions before a possible accident.

**Acknowledgements.** This work was funded in part by Continental Automotive Romania SRL Iași.

## References

1. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Furnkranz, J., Joachims, T. (eds.) Proceedings of the 27th International Conference on Machine Learning (ICML10), pp. 807–814 (2010)
2. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: Gordon, G., Dunson, D., Dudk, M. (eds.) JMLR W&CP: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2011), vol. 15, pp. 315–323 (2011)
3. Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B.: End to end learning for self-driving cars (2016). <https://arxiv.org/abs/1604.07316>. Accessed 24 Mar 2019
4. Hochreiter, S.: The vanishing gradient problem during learning recurrent neural nets and problem solutions. Int. J. Uncertainty Fuzziness Knowl.-Based Syst. **6**(2), 107–116 (1998)
5. Russakovsky, O., Deng, J., Su, H., Krause, J.: ImageNet large scale visual recognition challenge (ILSVRC). Int. J. Comput. Vis. **115**(3), 211–252 (2014)
6. Net-Scale Technologies, Inc.: Autonomous off-road vehicle control using end-to-end learning. Final technical report (2004). <http://net-scale.com/doc/net-scale-dave-report.pdf>. Accessed 24 Mar 2019
7. LeCun, Y.A., Bottou, L., Orr, G.B., Müller, K.-R.: Efficient backprop. In: Montavon, G., Orr, G.B., Müller, K.-R. (eds.) Neural Networks: Tricks of the Trade. LNCS, vol. 7700, pp. 9–48. Springer, Heidelberg (2012). [https://doi.org/10.1007/978-3-642-35289-8\\_3](https://doi.org/10.1007/978-3-642-35289-8_3)
8. Pomerleau, D.A.: ALVINN: an autonomous land vehicle in a neural network (1989). <http://papers.nips.cc/paper/95-alvinn-an-autonomous-land-vehicle-in-a-neural-network.pdf>. Accessed 24 Mar 2019
9. Matthews, K.: Top Article for 2018 - Here Are All the Companies Testing Autonomous Cars In 2018 (2018). <https://www.roboticstomorrow.com/article/2018/03/top-article-for-2018-here-are-all-the-companies-testing-autonomous-cars-in-2018/11592>. Accessed 24 Mar 2019
10. Hochreiter, S., Schmidhuber, J.: Feature extraction through LOCOCODE. Neural Comput. **11**(3), 679–714 (1999)
11. Clevert, D.-A., Unterthiner, T., Mayr, A., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (ELUs). In: ICLR2016 (2016)
12. Klambauer, G., Unterthiner, T., Mayr, A., Hochreiter, S.: Self-normalizing neural networks. In: Advances in Neural Information Processing Systems (2017). <https://arxiv.org/abs/1706.02515>. Accessed 24 Mar 2019
13. Pedamonti, D.: Comparison of non-linear activation functions for deep neural networks on MNIST classification task (2018). <https://arxiv.org/abs/1804.02763>. Accessed 24 Mar 2019



# A Machine Hearing Framework for Real-Time Streaming Analytics Using Lambda Architecture

Konstantinos Demertzis<sup>1(✉)</sup>, Lazaros Iliadis<sup>1</sup>,  
and Vardis-Dimitris Anezakis<sup>2</sup>

<sup>1</sup> School of Engineering, Department of Civil Engineering,  
Faculty of Mathematics Programming and General Courses,  
Democritus University of Thrace, Kimmeria, Xanthi, Greece

[kdemertz@fmenr.duth.gr](mailto:kdemertz@fmenr.duth.gr), [liliadis@civil.duth.gr](mailto:liliadis@civil.duth.gr)

<sup>2</sup> School of Agriculture and Forestry, Department of Forestry and Management  
of the Environment and Natural Resources,  
Democritus University of Thrace, Orestiada, Greece  
[danezaki@fmenr.duth.gr](mailto:danezaki@fmenr.duth.gr)

**Abstract.** Disruptions to the earth's biosphere and to the natural environment stemming from the indiscreet human activity, have caused serious environmental problems which are tantamount to an extended and prolonged ecological crisis. Climate change is clearly reflected in the increase of the global average air and ocean temperatures, in the excessive melting of snow-ice, and in the rise of the global average sea level. One of the most serious impacts of climate change is the complex interaction of species in relation to their corresponding climatic survival factors, which favors the spread of invasive species (INSP). These species constitute a very serious and rapidly deteriorating threat to the natural biodiversity of the native environment, but also to the flora, fauna, and even to the local human population. This research proposes a Machine Hearing (MH) framework for real-time streaming analytics, employing Lambda Architecture (LARC). The hybrid modeling effort is based on timely and advanced Computational Intelligence (COIN) approaches. The Framework for Lambda Architecture Machine Hearing (FLAME\_H) uses a combination of batch and streaming data. The FLAME\_H applies the EL\_GROSEMARI (Extreme Learning Graph Regularized Online Sequential Multilayer Multiencoder Algorithm) to classify the batch data and the Adaptive Random Forest (ARF) in order to control the data streams in real time. The aim of the proposed framework is the intelligent identification and classification of invasive alien species, based on the sounds they produce. This would contribute to the protection of biodiversity and biosecurity in a certain area.

**Keywords:** Invasive species · Machine Hearing · Lambda Architecture · Adaptive random forests · Deep learning · Extreme Learning Machine · Streaming data

## 1 Introduction

Endemism can be sustained by natural hurdles such as rivers, oceans, mountains, deserts and climatic conditions [1]. Due to increasing and continuous climate change, coupled with globalization and the development of international trade and tourism, these natural barriers are becoming increasingly inefficient. As a result, species, particularly marine ones, are able to travel long distances to other biotopes where they become alien or, in many cases, even expansive species [2]. The process of recognizing INSP is a critical step for the adoption and implementation of a specific policy to tackle, eradicate, control and/or contain these species. Given that these species are usually unknown in their new environment, it is extremely difficult, complex and dangerous to identify and securely isolate them. It should be emphasized that neither the large differences in morphology nor the significant similarities reflect the affinity of biological organisms [3]. The need for thorough and fully valid identification of these species is very serious in the case of planning their response programs, as the recognition process depends on a multitude of required information and on a continuous monitoring of the current situation. Searching for novel methods of resolving or analyzing phenomena related to the potential impacts of climate change, such as methods of identifying invasive species, are key research priorities of high importance.

On the other hand, Machine Hearing (MAHE) is a scientific branch of artificial intelligence that attempts to reproduce the sense of hearing algorithmically [4]. It is related (in theoretical and practical level) to the design and development of data analysis systems. MAHE data are obtained from digital sound recorders or by appropriate sensors. Audio signal analysis is related to knowledge mining and it aims in the classification, segmentation, or automated retrieval [5]. In general, the process initially involves the extraction of certain features which must be able to differentiate their values according to the content and structure of the respective signals. After the determination of the audio features that characterize the sound signal, a pattern recognition approach is employed [6]. The algorithmic resolution of a MAHE classification problem requires a high availability of resources. In this case, we have to examine the temporal complexity of the algorithm, the memory availability as a function of its input data, as well as individual analysis should be performed related to other resources as appropriate (e.g. how many parallel processors are needed for a parallel solution of the problem). This is a *Big Data* (BDA) problem, as data extracted from audio clips, require big storage space for their clear computer comprehension.

The need for Big Data management and analysis such as the MAHE problems, has re-established the prototyping architectures of BDA [7]. Lambda architecture (LAR) is the most important one. It has been designed to handle massive amounts of data using the batch and streaming processing methods. This approach attempts to balance latency, throughput, and fault tolerance using the batch process to provide complete and accurate views of historical data. At the same time, it uses real time data stream processing to provide views of new inputs [8]. The two projection outputs can be joined before the final data presentation or the final decision.

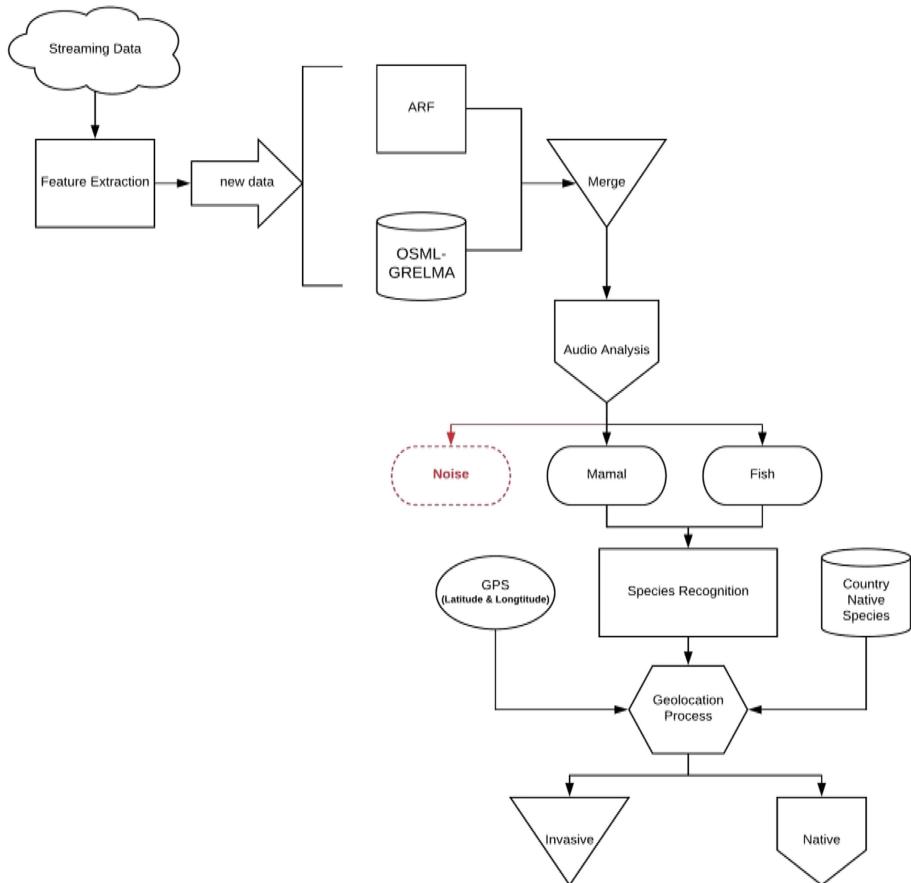
## 2 The Proposed FLAME\_H

This paper proposes the development of the FLAME\_H (Framework of Lambda Architecture for Machine Hearing) which performs real-time streaming analytics. It is an advanced hybrid computational intelligence approach. The FLAME\_H, employs an innovative version of the Lambda architecture, which optimally combines batch and streaming data, to safely perform classification. FLAME\_H is a MAHE system, that performs real-time audio streaming analysis and classifies audio data sets to identify patterns. At the same time, it adopts different biosecurity policies for each resulting category. It is a very important method of locating invasive species, and an innovative approach of recording biofouling. At the same time, it can be considered a key tool in security policy mechanisms as it allows for safe and cost-effective assessment of their behavior and disclosure of the damage caused by their activities.

Batch data (BADA) is usually a set of data that is collected during some processes for a specific time period and it characterizes and identifies species. Their processing by conventional data mining (DAM) or Machine Learning (ML) methods, considers that they are available and can be accessed simultaneously without any limitation in terms of their processing or analysis time. It should be noted that this data is susceptible to noise, their classification process has a significant cost, and they require serious hardware infrastructure for safe storage and general handling. On the contrary, due to their reliance on strict time constraints and their more general availability, they are selected for detailed and specialized data processing techniques that can lead to multiple levels of revelation of the hidden knowledge they may contain. The growing field of real-world applications produces streaming data (SDA) at an increasing rate, requiring large-scale and real-time processing. SDA such as audio data analysis and data generated in dynamic environments, leads to one of the most robust research areas of DAM. It is the ML application on data streams for pattern recognition under dynamic displacement and feedback environments. In general, FLAME\_H is an intelligent hybrid ML system, that employs a special version of the LARC architecture and the Deep Learning EL\_GROSEMARI algorithm (for the batch data classification) combined with the ARF approach in order to control data streams in real time.

Figure 1 presents an overall description of the algorithm.

In the first phase of the algorithm, the appropriate features are derived from the *Sea Audio* data stream (*audio feature extraction*). The data are then provided as input and they are controlled in parallel by the two learning algorithms. This is done aiming to minimize the likelihood of misleading and to achieve high reliability classification. The decision process merges the results of the two algorithms offering advantage to the ones obtained by the EL\_GROSEMARI. The decision concerns whether it is a “sound” coming from an invasive species fish. If the sound is described as noise that comes from a usually human sea-related process, then it is rejected and there is no further development in the process. If the sound comes from a species of fish or mammal and once this species is identified, the coordinates are taken from the GPS and assigned to the country where they belong. Then a check is made on whether the species identified is native to this country, otherwise it is recorded as an invasive species. Listings with indigenous and invasive species were extracted from the Invasive Species Compendium



**Fig. 1.** Structure of the FLAME\_H

(<http://www.cabi.org/isc/>) [9], the most valid and comprehensive database on the issue, world-wide. The Geolocation process is presented below:

---

**Algorithm 1.** Geolocation Process

---

**Input:**

```

Recognized_Species;
Country;
Country_Native_Species;
1: Read Recognized_Species, Country, Country_Native_Species;
2:   for i=1 to Country_Native_Species [max] do
3:     if Country_Native_Species [i]= Recognized_Species then
4:       Recognized_Species=Native_Species
5:     else
6:       Recognized_Species=Invasive_Species
7:     end if
8:   end
9: end
  
```

**Output:**

Species Identity;

---

Lambda architecture was employed, as in multifactorial problems of high complexity of large data sets such as the one under consideration, the results of the forecast are multi-variable, especially with respect to analysis and integration of data flows.v. This architecture employs a serious *Reactive* strategy to deal with invasive species. The combination of two different algorithms facilitates the sorting process, making each classifier more robust, and it accelerates the convergence of the generic multiple model, which is less noisy than any single one [10]. Thus, this approach offers generalization and avoids overfitting which is one of the basic targets in Machine Learning.

### 3 Literature Review

Invasive alien species are a result of generalized climate change and they constitute a serious and rapidly worsening threat to natural biodiversity in Europe. European Union spends at least 12 billion Euros per year on control of IAS and disasters they cause. Also, the risk to public health should not be overlooked as these species may be toxic, such as "*Lagocephalus*" fish, which contains "*Tetrodotoxin*" a very dangerous substance, capable of causing serious health problems, even death in the consumer [11, 12]. The significance of the hybrid innovative intelligent approaches (Machine Learning Algorithms) for identifying IAS and their separation from indigenous ones has been developed by recent researches [13, 14]. Soft computing techniques are capable to model and detect cyber security threats [15, 16] and they also offer optimization mechanisms in order to produce reliable results.

Hinton et al. [17] had proposed methods and applications of DL. Through a series of new learning architectures and algorithms, domains such as object recognition [18] and machine translation [19, 20] have been transformed; deep learning methods are now the state-of-the-art in object, speech and audio recognition. In particular, deep learning has been the driving force behind large leaps in accuracy and model robustness in audio related domains like audio sensing [21]. Alom et al. [22] applied the Cellular Simultaneous Recurrent Networks (CSRNs) to generate initial filters of CNNs for features extraction and Regularized Extreme Learning Machines (RELM) for classification. Experiments were conducted on three popular datasets for object recognition (such as face, pedestrian, and car) to evaluate the performance of the proposed system. Zhang, et al. [23], proposed an object recognition algorithm which did not depend on human experts to design features for fish species classification, but constructed efficient features automatically. Results from experiments showed that the proposed method obtained an average of 98.9% classification accuracy with a standard deviation of 0.96% with a dataset composed of 8 fish species and a total of 1049 images. Also, DL has been the driving force behind large leaps in accuracy and model robustness in audio related domains like speech recognition. Moreover Han et al. [24] proposed to utilize DNNs to extract high level features from raw data and show that they are effective for speech emotion recognition. Finally, Zhao et al. [25] proposed a new method for automated field recording analysis with improved automated segmentation and robust bird species classification by a Gaussian Mixture Model.

## 4 Algorithms

### 4.1 Extreme Learning Machines for Batch Data Algorithms

An ELM is a Single-Hidden Layer Feed Forward Neural Network (SLFFNN) [26] with  $N$  hidden neurons, randomly selected input weights and random values of bias in the hidden layer, while the weights at its output are calculated with a single multiplication of vector matrix [27]. For an ELM using SLFFNN and random representation of hidden neurons, input data is mapped to a random  $L$ -dimensional space with a discrete training set  $N$ , where  $(x_i, t_i), i \in \llbracket 1, N \rrbracket$  with  $x_i \in R^d$  and  $t_i \in R^c$ . The specification output of the network is the following:

$$f_L(x) = \sum_{i=1}^L \beta_i h_i(x) = h(x)\beta \quad i \in \llbracket 1, N \rrbracket \quad (1)$$

Vector matrix  $\beta = [\beta_1, \dots, \beta_L]^T$  is the output of the weight vector matrix connecting hidden and output nodes. On the other hand,  $h(x) = [g_1(x), \dots, g_L(x)]$  is the output of the hidden nodes for input  $x$ , and  $g_i(x)$  is the output of the  $i$ th neuron. Based on a training set  $\{(x_i, t_i)\}_{i=1}^N$ , an ELM can solve the Learning Problem  $H\beta = T$ , where  $T = [t_1, \dots, t_N]^T$  are the target labels and the output vector matrix of the Hidden Layer  $H$  is the following:

$$H(\omega_j, b_j, x_i) = \begin{bmatrix} g(\omega_1 x_1 + b_1) & \cdots & g(\omega_L x_1 + b_L) \\ \vdots & \ddots & \vdots \\ g(\omega_1 x_N + b_1) & \cdots & g(\omega_L x_N + b_L) \end{bmatrix}_{N \times l} \quad (2)$$

The input weight vector matrix of the hidden layer  $\omega$  (before training) and the bias vectors  $b$  are created randomly in the interval  $[-1, 1]$ , with

$$\omega_j = [\omega_{j1}, \omega_{j2}, \dots, \omega_{jm}]^T \text{ and } \beta_j = [\beta_{j1}, \beta_{j2}, \dots, \beta_{jm}]^T \quad (3)$$

The output weight vector matrix of the hidden layer  $H$  is calculated by the use of the Activation function in the training dataset, based on the following function:

$$H = g(\omega x + b) \quad (4)$$

The output weights  $\beta$  can be estimated by using function:

$$\beta = \left( \frac{I}{C} + H^T H \right)^{-1} H^T X \quad (5)$$

where  $H = [h_1, \dots, h_N]$  is the output vector matrix of the hidden layer and  $X = [x_1, \dots, x_N]$  the input vector matrix of the hidden layer. Indeed,  $\beta$  can be calculated by the following general relation:

$$\beta = H^+ T \quad (6)$$

where  $H^+$  is the generalized inverse vector matrix Moore-Penrose for matrix  $H$ . This approach is employing ELM with Gaussian Radial Basis Function kernel  $K(u, v) = \exp(-\gamma \|u - v\|^2)$ . The size  $k$  of the hidden layer are 20 neurons. Subsequently assigned random input weights  $w_i$  and biases  $b_i$ ,  $i = 1, \dots, N$ . To calculate the hidden layer output matrix  $H$  we have used the function (7):

$$H = \begin{bmatrix} h(x_1) \\ \vdots \\ h(x_N) \end{bmatrix} = \begin{bmatrix} h_1(x_1) & \cdots & h_L(x_1) \\ \vdots & & \vdots \\ h_1(x_N) & \cdots & h_L(x_N) \end{bmatrix} \quad (7)$$

Where  $h(x) = [h_1(x), \dots, h_L(x)]$  is the output (row) vector of the hidden layer with respect to the input  $x$ .  $h(x)$  actually maps the data from the D-dimensional input space to the L-dimensional hidden-layer feature space (ELM feature space)  $H$ . Thus,  $h(x)$  is indeed a feature mapping. ELM is to minimize the training error as well as the norm of the output weights:

$$\text{Minimize: } \|H\beta - T\|^2 \text{ and } \|\beta\| \quad (8)$$

where  $H$  is the hidden-layer output matrix of the function (7).

Minimization of the norm of the output weights  $\|\beta\|$  is actually achieved by maximizing the distance of the separating margins of the two different classes in the ELM feature space  $2/\|\beta\|$ . To calculate the output weights  $\beta$  we used function (9):

$$\beta = (\frac{I}{C} + H^T H)^{-1} H^T T \quad (9)$$

where the value of  $C$  (a positive constant) and the value of  $T$  are obtained from the *Function Approximation of SLFFNs* with additive neurons:

$$t_i = [t_{i1}, t_{i2}, \dots, t_{im}]^T \in \mathbb{R}^m, T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix} \quad (10)$$

Considering and combining the features of ELM presented above, we introduce and propose a new Deep architecture by creating an Online Learning Multilayer Graph Regularized Extreme Learning Machine Auto-Encoder (OSML-GRELMA). This is a multi-layered neural network model that receives successive OL data streams and uses the unsupervised GRELMA algorithm as a basic building block in which the output of each level is used as inputs to the next one [28].

An autoencoder is an artificial neural network used for unsupervised learning of efficient coding. The aim of an autoencoder is to learn a representation (encoding) for a set of data, but with the output layer having the same number of nodes as the input layer, and with the purpose of reconstructing its own inputs (instead of predicting target value  $Y$  given inputs  $X$ ). The Algorithm 2 is described below [28]:

**Algorithm 2.** GRELMA Algorithm for Clustering [28]

---

Input: Data  $\{X\} = \{x_i\}_{i=1}^N$  the number of hidden neurons  $n_h$ , the penalty coefficient  $\kappa$  and  $\lambda$   
 Output: The cluster results.  
 Step 1: Initialize an ELM of  $n_h$  hidden neurons with random input weights and biases.  
 Step 2: If  $n_h \leq N$  Compute the output weights  $\beta$  by equation  $\beta^* = (I_{n_h} + H^T CH + \lambda H^T LH)^{-1} H^T CX$   
 Else Compute the output weights  $\beta$  by equation  $\beta^* = H^T (I_N + CHH^T + \lambda LHH^T)^{-1} CX$   
 Step 3:  $X_{new} = X\beta^*$   
 Step 4: Treat each row of  $X_{new}$  as a point and cluster the  $N$  points into  $K$  clusters using the k-means algorithm.

---

The overall function of the OSML-GRELMA is presented in the following algorithm.

**Algorithm 3.** OSML-GRELMA Algorithm for Classification

---

Input: A small initial training set  $N = \{(x_i, t_i) | x_i \in R^n, t_i \in R^m, i=1, \dots, \bar{N}\}$   
 The model depth:  $m$ ;  
 The number of hidden nodes in the each GRELMA:  $n_{h1}, n_{h2}, \dots, n_{hm}$ ;  
 The new activation function:  $h_{new}$ .  
 Output: The classification results of the  $M$  data.  
 Phase 1 (BPh)  
 Initialize  $X_j = X_{train}$   
 For  $i = 1: m$ :  
 Assign arbitrary input weight  $w_i$  and bias  $b_i, i=1$ , of the  $i_{th}$  layer GRELMA by some random numbers;  
 Calculate the initial hidden layer output matrix  $H_0 = [h_1, \dots, h_{\bar{N}}]^T$ , where  $h_i = [h_{new}(w_1 \cdot x_i + b_1), \dots, h_{new}(w_{\bar{N}} \cdot x_i + b_{\bar{N}})]^T$ ,  $i = 1, \dots, \bar{N}$ , where  $h_{new}$  activation function.  
 Train the output weights  $\beta$  of the  $i_{th}$  layer GRELMA;  
 Estimate the initial output weight  $\beta^{(0)} = M_0 H_0^T T_0$ , where  $M_0 = (H_0^T H_0)^{-1}$  and  $T_0 = [t_1, \dots, t_{\bar{N}}]^T$ .  
 Set  $k = 0$ .  
 Compute the outputs  $X_i + I = h_{new}(X\beta_i^T)$ .  
 Phase 2 (SLPh)  
 The essentials step of this phase for each further coming observation  $(x_i, t_1)$ , where  $x_i \in R^n, t_i \in R^m$  and  $i = \bar{N} + 1, \bar{N} + 2, \bar{N} + 3$ , described as follow:  
 Calculate the hidden layer output vector  $h_{(k+1)} = [h_{new}(w_1 \cdot x_i + b_1), \dots, h_{new}(w_{\bar{N}} \cdot x_i + b_{\bar{N}})]^T$   
 Calculate latest output weight  $\beta^{(k+1)}$  by the algorithm  $\hat{\beta} = (H^T H)^{-1} H^T T$  which is called the Recursive Least-Squares (RLS) algorithm.  
 Set  $k = k + 1$   
 End For  
 Map  $X_{m+1}$ , the output of the  $m_{th}$  layer, to the output layer.  
 Compute the classification results by using the above trained OSML-GRELMA model.

---

The main objective and training success of the proposed OSML-GRELMA approach is based on evolutionary identification of the underlying structure of the input data flows to produce the final model. It basically uses the knowledge of labelled data to investigate the distribution of the input data, aiming at enhancing the outcome of the learning process using an adaptive scheme. In this sense, it includes procedures that approach unsupervised learning, where inputs come from the same marginal distribution or follow a common cluster structure.

## 4.2 Adaptive Random Forests for Streaming Data

As it can be seen, data flow management and especially knowledge extraction with ML algorithms from these flows are unlikely to be performed by applying iterations over input data. Accordingly, adapting the Random Forest algorithm [29] to streaming data, depends on a suitable accumulation process that is partially achieved by a bootstrap method and at the same time by limiting each decision to divide the sheets into a subset

of attributes. This is achieved with the modification of the base tree algorithm, by effectively reducing the set of features examined for further separation into random subsets of magnitude  $m$ , where  $m < M$  ( $M$  corresponds to the total number of attributes that are examined in each case). In the non-streaming bagging, each of the  $n$ -base models is trained in a  $Z$ -sized bootstrap sample created by random samples with replacement from the original training kit. Every bootstrapped sample contains an original training snapshot  $K$ , where  $P(K = k)$  follows a binomial distribution. For large values of  $Z$  this binomial distribution is attached to a Poisson one, with  $\lambda = 1$ . On the other hand, according to the ARF approach for streaming data, a Poisson distribution is used with  $\lambda = 6$ . This “feedback” has the practical effect of increasing the probability of assigning higher weights to instances during the training of the basic models [29].

ARF is an adaptation of the original Random Forest algorithm, which has been successfully applied to a multitude of machine learning tasks. In layman’s terms the original Random Forest algorithm is an ensemble of decision trees, which are trained using bagging and where the node splits are limited to a random subset of the original set of features. The “Adaptive” part of ARF comes from its mechanisms to adapt to different kinds of concept drifts, given the same hyper-parameters. Specifically, the 3 most important aspects of the ARF algorithm are it adds diversity through resampling (“bagging”); it adds diversity through randomly selecting subsets of features for node splits and it has one drift and warning detector per base tree, which cause selective resets in response to drifts. It also allows training background trees, which start training if a warning is detected and replace the active tree if the warning escalates to a drift. ARF was designed to be “embarrassingly” parallel, in other words, there are no dependencies between trees. The overall ARF pseudo-code is presented below in Algorithm 4 [29].

---

**Algorithm 4.** Adaptive Random Forests

---

```

function ARF (m, n, δw, δd)
    T ← CreateTrees(n)
    W ← InitWeits(n)
    B ← Ø
    while HasNext(S) do
        (x, y) ← next(S)
        for all t ∈ T do
            ŷ ← predict(t, x)
            W(t) ← P(W(t), ŷ, y)
            RFTreeTrain (m, t, x, y)
            if C (δw, t, x, y) then
                b ← CreateTrees()
                B(t) ← b
            end if
        end for
        for all b ∈ B do
            RFTreeTrain (m, b, x, y)
        end for
    end while
end function

```

---

Where  $m$ : maximum features evaluated per split;  $n$ : total number of trees ( $n = |T|$ );  $\delta_w$ : warning threshold;  $\delta_d$ : drift threshold;  $C(\cdot)$ : change detection method;  $S$ : Data stream;  $B$ : Set of background trees;  $W(t)$ : Tree  $t$  weight;  $P(\cdot)$ : Learning performance estimation function [29].

## 5 Datasets

The following four main categories of sounds have been determined in order to create highly complex scenarios that can potentially include the most likely cases that can be detected in an underwater space:

- Fishes: Several species of fish produce sounds with various mechanisms such as teeth, pharynx, fins, and shuttle bladder. 1076 sounds belonging to 10 fish species have been included in this category (e.g. Bidyanus Bidyanus, Epinephelus Adscensionis, Cynoscion Regalis, Carassius Auratus, Cyprinus Carpio, Rutilus Rutilus, Salmo Trutta, Oreochromis Mossambicus, Micropterus Salmoides, Oncorhynchus Mykiss).
- Mammals: Marine mammals produce and use sounds to orientate and communicate with each other. Totally 836 sounds belonging to 8 species of mammals are included in this category. (e.g. Delphinus Delphis, Erignathus Barbatus, Balaena Mysticetus, Phocoena Phocoena, Neophocaena Phocaenoides, Trichechus, Tursiops Truncates, Phoca Hispida).
- Anthropogenic Sounds: It comprises of 684 sounds belonging to 9 classes (Ship, Sonar, Zodiac, Torpedo, Wind Turbine, Scuba Noise, Bubble Curtain, Personal Water Craft, Airgun).
- Natural Sounds: Totally 477 sounds belong here classified in six clusters (Earthquake, Hydrothermal Vents, Ice Cracking, Rainfall, Lightning, Waves).

The Feature Extraction process [30] enables capturing of characteristics that precisely determine the uniqueness of each sound and helps distinguish between acoustic categories. The categories distinction is based on 34 characteristics related to statistical measurements obtained from the signal frequency information. In this research effort we have extracted the short-term feature sequences for an audio signal, using a frame size of 50 ms and a frame step of 25 ms (50% overlap). All sounds had a sampling rate of 44.1 kHz, 16-bit stereo resolution while their average duration was 10.3 s.

## 6 Results

In data batch cases using multiple classifiers, for estimating the real error during training, the full probability density of both categories should be known [31, 32]. The classification performance is estimated by the *Total Accuracy* (TA), *Root Mean Squared Error* (RMSE), *Precision* (PRE), *Recall* (REC), *F-Score* and *ROC Area* indices [33, 34]. The 10-fold cross validation is employed in this stage in order to obtain performance indices. Analytical values of the predictive capacity of the algorithm are presented in the following Tables 1, 2, 3, 4, 5 and 6.

In the case of stream data classification, we need to compare classification performance in terms of *Accuracy Kappa* statistic and *Kappa-Temporal* statistic. This is done by using the traditional immediate setting. The true label is presented right after the instance used for testing or the delayed setting (where there is a real delay between the moment an instance is presented and the moment its true label becomes available) [33].

**Table 1.** Performance metrics of Categories\_Dataset

Classifier	Classification accuracy & performance metrics					
	TA	RMSE	PRE	REC	F-Score	ROC Area
OSML-GRELMA	96.08%	0.1376	0.960	0.960	0.960	0.970

**Table 2.** Confusion matrix of Categories\_Dataset

Fishes	Mammals	Anthr_Sounds	Natural_Sounds	
<b>1042</b>	13	12	9	Fishes
14	<b>797</b>	17	8	Mammals
5	7	<b>659</b>	13	Anthr_Sounds
4	6	10	<b>457</b>	Natural_Sounds

**Table 3.** Performance metrics of Mammals\_Dataset

Classifier	Classification accuracy & performance metrics					
	TA	RMSE	PRE	REC	F-Score	ROC Area
OSML-GRELMA	92.18%	0.1571	0.922	0.922	0.922	0.955

**Table 4.** Confusion matrix of Mammals\_Dataset

a	b	c	d	e	f	g	h		
<b>142</b>	2	0	1	1	0	1	0	a = Delphinus Delphis	
1	<b>101</b>	3	0	0	5	0	4	b = Erignathus Barbatus	
1	2	<b>122</b>	1	0	0	0	2	c = Balaena Mysticetus	
1	1	2	<b>61</b>	1	1	0	3	d = Phocoena Phocoena	
2	2	3	1	<b>51</b>	0	2	2	e = Neophocaena Phocaenoides	
2	0	2	0	1	<b>82</b>	0	2	f = Trichechus	
2	0	0	0	0	1	<b>51</b>	1	g = Tursiops Truncates	
2	2	1	1	1	0	1	<b>162</b>	h = Phoca Hispida	

**Table 5.** Performance metrics of Fishes\_Dataset

Classifier	Classification accuracy & performance metrics					
	TA	RMSE	PRE	REC	F-Score	ROC Area
OSML-GRELMA	87.91%	0.1512	0.879	0.879	0.879	0.920

**Table 6.** Confusion matrix of Fishes\_Dataset

a	b	c	d	e	f	g	h	i	j
<b>139</b>	5	1	2	1	0	1	0	1	3
4	<b>91</b>	4	1	3	0	0	2	4	0
0	3	<b>116</b>	1	2	0	2	4	2	2
1	0	1	<b>88</b>	1	0	0	1	0	1
0	3	1	1	<b>100</b>	0	0	2	1	2
1	0	0	0	1	<b>58</b>	0	2	0	3
1	0	1	0	0	0	<b>94</b>	2	0	0
3	5	6	1	3	1	1	<b>103</b>	1	8
0	3	2	0	3	0	0	1	<b>80</b>	3
1	0	3	1	3	1	1	3	1	<b>78</b>

a = Bid/nus Bidyanus, b = Epin/lus

Adscen/nis, c = Cyn/cion Regalis, d = Cara/us  
Auratus

e = Cyp/nus Carpio, f = Rutilus Rutilus,

g = SalmoTru, h = Oreo/mis Mossambicus

i = Micr/rus Salmoides, j = Oncor/hus Mykiss

Table 7 below, presents the results of the scenarios applied on streaming data in this research. Validation of the results was done by employing the Prequential Evaluation method [34]. The training window used 1000 instances. It should be clarified that the following Table 7 uses average values for every evaluation measure.

**Table 7.** Validation metrics when streaming data are used

Classifier	Classification accuracy & performance metrics		
	Accuracy	Kappa statistic	Kappa temporal statistic
<b>Categories_Dataset</b>			
ARF	94.48%	73.91%	74.53%
<b>Mammals_Dataset</b>			
ARF	92.16%	71.37%	73.14%
<b>Fish_Dataset</b>			
ARF	88.11%	68.59%	70.36%

## 7 Discussion and Conclusions

This paper presents an innovative, reliable, low-demand and highly effective system of MAHE and sound analysis, based on sophisticated computational intelligence. The development of FLAME\_H is based on the optimal combination of two highly efficient and fast learning algorithms that create a comprehensive intelligent system of active environmental security using a Lambda Architecture approach. The sophisticated

application described herein, combined with the promising results that have emerged, constitutes a credible innovative proposal for the standardization and design of biosecurity and biodiversity protection. This implementation follows a Reactive strategy for dealing with invasive species as it combines training of two counter diametrically opposite classifiers to detect incoming contrasts and to discard them. Training is done by using datasets that respond to specialized, realistic scenarios. In addition, this framework implements a Big data analysis approach that attempts to balance latency, throughput, and fault tolerance using integrated and accurate views of historical data, while at the same time it is making optimum use of new entrant data flows. The operating scenarios proposed with the combination of batch and streaming data, create capabilities for a fully-defined configuration of model's parameters and for high-precision classification or correlation.

The basic innovation of the proposed FLAME\_H is the implementation of an intelligent ML system, based solely on fully automated methods of detecting sound events using COIN. This innovation provides important solutions and improves the way environmental problems and, in particular, biodiversity and bio-security mechanisms work and deal. Also, a significant innovation is the architecture of the proposed computational intelligence system, which combines and exploits Lambda architecture, that is, the combination of both batch and streaming data analysis, using fast and extremely accurate ML algorithms to solve a multidimensional and complex real-life problem. ML delivers intelligence and significantly boosts the environmental protection mechanisms as it is an important defense tool against asymmetric environmental threats. The FLAME\_H simplifies and automates the sound recognition and the invasive species detection procedures, while minimizing human intervention by combining the EL\_GROSEMARI and ARF algorithms for the first time in the literature. Finally, one more innovation is found in the way of collecting and selecting the data, (which emerged after extensive research) as well as the development of the final data set used, which is complex and has a high dimension, but it can be used effectively in training.

Future extensions-improvements should focus on further optimizing the parameters of the algorithm used by the Lambda architecture, so that an even more efficient, accurate, and faster classification process is achieved. Also, it would be important to study the expansion of this particular system by implementing Lambda architecture in a parallel and distributed data analysis system (Hadoop). Finally, an additional element that could be studied in the direction of future expansion concerns the operation of FLAME\_H with methods of self-improvement and meta-learning in order to fully automate the process of locating the species.

## References

1. Rahel, F., Olden, J.D.: Assessing the effects of climate change on aquatic invasive species. *Conserv. Biol.* **22**(3), 521–533 (2008). <https://doi.org/10.1111/j.1523-1739.2008.00950.x>
2. Abdulla, A., Linden, O.: Maritime Traffic Effects on Biodiversity in the Mediterranean Sea: Review of Impacts, Priority Areas and Mitigation Measures. IUCN, Centre for Mediterranean Cooperation, vol. 184, pp. 08 (2008)

3. Miller, W.: The structure of species, outcomes of speciation and the species problem: ideas for paleobiology. *Paleogeograohy Palaeoclimatol. Palaeoecol.* **176**(1–4), 1–10 (2001). [https://doi.org/10.1016/s0031-0182\(01\)00346-7](https://doi.org/10.1016/s0031-0182(01)00346-7)
4. Lyon, R.: Human and Machine Hearing: Extracting Meaning from Sound. Cambridge University Press, Cambridge (2017). <https://doi.org/10.1017/9781139051699>
5. Deng, L., Yu, D.: Deep learning: methods and applications. *Found. Trends Signal Process.* **7** (3–4), 197–387 (2014). <https://doi.org/10.1561/2000000039>
6. Zhang, J., Yin, J., Zhang, Q., Shi, J., Li, Y.: Robust sound event classification with bilinear multi-column ELM-AE and two-stage ensemble learning. *Eurasip J. Audio Speech Music Process.* **2017**(1), 11 (2017). <https://doi.org/10.1186/s13636-017-0109-1>
7. Dedić, N., Stanier, C.: Towards differentiating business intelligence, big data, data analytics and knowledge discovery. In: Piazolo, F., Geist, V., Brehm, L., Schmidt, R. (eds.) *ERP Future 2016*. LNBP, vol. 285, pp. 114–122. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-58801-8\\_10](https://doi.org/10.1007/978-3-319-58801-8_10)
8. Kiran, M., Murphy, P., Monga, I., Dugan, J., Baveja, S.S.: Lambda architecture for cost-effective batch and speed big data processing, pp. 2785–2792 (2015). <https://doi.org/10.1109/bigdata.2015.7364082>
9. <http://www.cabi.org/isc>
10. Yamato, Y., Kumazaki, H., Fukumoto, Y.: Proposal of lambda architecture adoption for real time predictive maintenance. In: 2016 CANDAR, pp. 713–715. IEEE (2016). <https://doi.org/10.1109/candar.2016.0130>
11. Demertzis, K., Iliadis, L.: Detecting invasive species with a bio-inspired semisupervised neurocomputing approach: the case of *Lagocephalus sceleratus*. *Neural Comput. Appl.* **28** (6), 1225–1234 (2017). <https://doi.org/10.1007/s00521-016-2591-2>
12. Demertzis, K., Iliadis, L.: Intelligent bio-inspired detection of food borne pathogen by DNA barcodes: the case of invasive fish species *Lagocephalus sceleratus*. In: Iliadis, L., Jayne, C. (eds.) *Engineering Applications of Neural Networks EANN 2015 Communications in Computer and Information Science*, vol. 517, pp. 89–99. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-23983-5\\_9](https://doi.org/10.1007/978-3-319-23983-5_9)
13. Demertzis, K., Iliadis, L., Anezakis, V.D.: A deep spiking machine-hearing system for the case of invasive fish species. In: Proceedings of 2017 IEEE International Conference on Innovations in Intelligent Systems and Applications, Gdynia, Poland, pp. 23–28 (2017). <https://doi.org/10.1109/inista.2017.8001126>
14. Demertzis, K., Iliadis, L.: Adaptive elitist differential evolution extreme learning machines on big data: intelligent recognition of invasive species. In: Angelov, P., Manolopoulos, Y., Iliadis, L., Roy, A., Vellasco, M. (eds.) INNS 2016. AISC, vol. 529, pp. 333–345. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-47898-2\\_34](https://doi.org/10.1007/978-3-319-47898-2_34)
15. Demertzis, K., Iliadis, L.: Evolving smart URL filter in a zone-based policy firewall for detecting algorithmically generated malicious domains. In: Gammerman, A., Vovk, V., Papadopoulos, H. (eds.) SLDS 2015. LNCS (LNAI), vol. 9047, pp. 223–233. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-17091-6\\_17](https://doi.org/10.1007/978-3-319-17091-6_17)
16. Demertzis, K., Iliadis, L.: SAME: an intelligent anti-malware extension for android art virtual machine. In: Núñez, M., Nguyen, N.T., Camacho, D., Trawiński, B. (eds.) ICCCI 2015. LNCS (LNAI), vol. 9330, pp. 235–245. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-24306-1\\_23](https://doi.org/10.1007/978-3-319-24306-1_23)
17. Hinton, G., Deng, L., Yu, D., et al.: Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Process. Mag.* **29**(6), 82–97 (2014). <https://doi.org/10.1109/MSP.2012.2205597>

18. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (eds.) Proceedings of the 25th International Conference on Neural Information Processing Systems, NIPS 2012, USA, pp. 1097–1105 (2012)
19. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: William, W., Mccallum, C., Mccallum, A., Sam, T.R. (eds.) Proceedings of the 25th International Conference on Machine Learning, ICML 2008, pp. 160–167. ACM, New York (2008). <https://doi.org/10.1145/1390156.1390177>, ISBN 978-1-60558-205-4
20. Deselaers, T., Hasan, S., Bender, O., Ney, H.: A deep learning approach to machine transliteration. In: Proceedings of the Fourth Workshop on Statistical Machine Translation StatMT 2009, pp. 233–241. Association for Computational Linguistics, Stroudsburg (2009)
21. Lanez, N.D., Georgiev, P., Qendro, L.: DeepEar: robust smartphone audio sensing in unconstrained acoustic environments using deep learning. In: Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, pp. 283–294 (2015). <http://dx.doi.org/10.1145/2750858.2804262>
22. Alom, M.Z., Alam, M., Taha, T.M., Iftekharuddin, K.M.: Object recognition using cellular simultaneous recurrent networks and convolutional neural network. In: Proceedings of the International Joint Conference on Neural Networks (IJCNN 2017), pp. 2873–2880, Anchorage (2017). <https://doi.org/10.1109/ijcnn.2017.7966211>
23. Zhang, D., Lee, D.J., Zhang, M., Tippets, B.J., Lillywhite, K.D.: Object recognition algorithm for the automatic identification and removal of invasive fish. Biosyst. Eng. **145**, 65–75 (2016). <https://doi.org/10.1016/j.biosystemseng>
24. Han, K., Yu, D., Tashev, I.: Speech emotion recognition using deep neural network and extreme learning machine. In: Chng, E.S., Li, H., Meng, H., Ma, B., Xie, L. (eds.) INTERSPEECH 2014, Proceedings of the Annual Conference of the International Speech Communication Association, pp. 223–227 (2014)
25. Zhao, Z., Zhang, S.H., Xu, Z.Y., et al.: Automated bird acoustic event detection and robust species classification. Ecol. Inf. **39**, 99–108 (2017). <https://doi.org/10.1016/j.ecoinf.2017.04.003>
26. Cambria, E., Guang-Bin, H.: Extreme learning machines. In: IEEE InTeLLIGenT SYSTeMS, 541-1672/13 (2013)
27. Huang, G.B.: An insight into extreme learning machines: random neurons, random features and kernels. Cogn. Comput. **6**(3), 376–390 (2014). <https://doi.org/10.1007/s12559-014-9255-2>
28. Sun, K., Zhang, J., Zhang, C., Hu, J.: Generalized extreme learning machine autoencoder and a new deep neural network. Neurocomputing **230**, 374–381 (2017)
29. Gomes, H.M., Bifet, A., Read, J., et al.: Adaptive random forests for evolving data stream classification. Mach. Learn. **106**(9–10), 1469–1495 (2017). <https://doi.org/10.1007/s10994-017-5642-8>
30. Giannakopoulos, T.: pyAudioAnalysis: an open-source Python library for audio signal analysis. PLoS ONE **10**(12) (2015). <https://doi.org/10.1371/journal.pone.0144610>
31. Žliobaitė, I., Bifet, A., Read, J., Pfahringer, B., Holmes, G.: Evaluation methods and decision theory for classification of streaming data with temporal dependence. Mach. Learn. **98**(3), 455–482 (2015). <https://doi.org/10.1007/s10994-014-5441-4>

32. Vinagre, J., Jorge, A.M., Gama, J.: Evaluation of recommender systems in streaming environments. In: Workshop on Recommender Systems Evaluation: Dimensions and Design (REDD 2014), Held in Conjunction with RecSys (2014). <https://doi.org/10.13140/2.1.4381.5367>
33. Mao, J., Jain, A.K., Duin, P.W.: Statistical pattern recognition: a review. *IEEE Trans. Pattern Anal. Mach. Intell.* **22**(1), 4–37 (2000). <https://doi.org/10.1109/34.824819>
34. Fawcett, T.: An introduction to ROC analysis. *Pattern Recogn. Lett.* **27**(8), 861–874 (2006). <https://doi.org/10.1016/j.patrec.2005.10.010>



# Deep Learning and Change Detection for Fall Recognition

Sotiris K. Tasoulis<sup>1</sup>(✉), Georgios I. Mallis<sup>1</sup>, Spiros V. Georgakopoulos<sup>1</sup>, Aristidis G. Vrahatis<sup>1</sup>, Vassilis P. Plagianakos<sup>1</sup>, and Ilias G. Maglogiannis<sup>2</sup>

<sup>1</sup> Department of Computer Science and Biomedical Informatics,  
University of Thessaly, Volos, Greece

{stasoulis,gmallis,spirosgeorg,arisvrahatis,vpp}@uth.gr

<sup>2</sup> Department of Digital Systems, University of Piraeus, Piraeus, Greece  
imaglo@unipi.gr

**Abstract.** Early fall detection is a crucial research challenge since the time delay from fall to first aid is a key factor that determines the consequences of a fall. Wearable sensors allow a reliable way for daily-life activities tracking, able to detect immediately a high-risk fall via a machine learning framework. Towards this direction, accelerometer devices are used widely for the assessment of fall risk. Although there is a plethora of studies under this perspective with promising results, several challenges still remain such as the extremely demanding data and power management as well as the discovery of false positive falls. In this work we propose a complete methodology based on the combination of the computationally demanding convolutional neural networks along with a lightweight change detection method. Our basic assumption is that it is possible to control computational resources for the operation of a classifier, suffice to be activated when a strong change in user's movements is identified. The proposed methodology was applied to real experimental data providing reliable results that justify the original hypothesis.

**Keywords:** Deep Learning · Change detection · Fall detection · Wearable devices

## 1 Introduction

It is widely known that falls in older people are among the main causes of fatal injury while nonfatal injuries usually requires hospitalization. An indicative example is the falls results according to the U.S. Centers for Disease Control and Prevention (falls and fall related), where around 2.8 million injuries treated in emergency departments recorded annually of which 800,000 needed hospitalization and more than 27,000 patients died.<sup>1</sup> Although the falls seem to hurt only the third age, there are also other cases of fall-related injuries such as among

<sup>1</sup> <https://www.cdc.gov/injury/wisqars/>.

adults with developmental disabilities [14]. Unfortunately, falls have additional implications apart from the most important which is the human health and life.

Timely fall detection consist of a major research challenge since the time delay from fall to first aid is a key factor that determines a fall severity [16]. An early detection allows the immediate assistance reducing the risk of a fatal or quite serious injury [3]. Hence, a fast and efficient fall detection can provide important assistance to older adults [35]. Although the large amount of approaches towards this direction, fall detection faces several challenges such as separation of a fall from other actions (e.g. lying) [20].

Furthermore, the recent progress of technology related to smart sensors and Internet of Thing (IoT) open new roads to fall detection research [35]. As a sequence, the research community has shifted to approaches by integrating IoT and Cloud Computing technologies for online tracking to detect possible falls of older adults [13, 21, 25]. Online measurement of proper body acceleration enables researchers to capture falls and distinguished them from normal daily activities [34]. Thus, accelerometer-based approaches have gained ground in this research field in the last years [9]. A major limitation of online tracking is the computational cost for such analysis. Given the fact that heavy computational approaches (Deep Learning) have been proposed recently towards this direction with promising outcomes, there is an urgent need for smart computational solutions, which will offer efficiency with low computational cost at the same time. Our proposed methodology operates under this perspective trying to uncover online fall detection in lower complexity by minimizing data transfer requirements and on going computations. Core of the proposed methodological framework lies on the fact that for an online fall detection analysis we don't need a non-stop classifier but a classifier that would operates when the circumstances required it.

## 2 Related Work

Fall detection methods can be categorized to methods based on (i) vision, (ii) ambient sensors in the environment and (iii) wearable devices. The methods based on wearables often rely on smart sensors with embedded processing capabilities which can be attached to the human body. Because of this fact they seem more attractive to elderly, as they are practical, of low cost, can be used easily all day and can detect falls which may take place in random locations.

Nowadays, accelerometer-based approaches have gained ground in fall detection since its technological evolution has improved both its usability as the data parsing process [35]. Accelerometers can built in smartphone now allowing for implementation and design of an accurate way to detect timely fall accidents in one device. Towards this direction, several studies have been proposed with promising results. Kau et al. [17] proposed a fall accident detection model using a smart phone and the third generation (3G) networks. Similarly, Aguiar et al. [2] proposed a fall detection system based on smart phone by applying decision tree classification algorithm for data analysis. Maglogiannis et al. paired a Pebble Smart Watch together with an Android device in an attempt to recognize activity type, calculate the energy consumption and detect falls [22]. Shen et al. [29]

proposed a fuzzy scheme for fall prediction system on smartphones by discriminating the human's actions such as normal action and sport with fall risk. Chen et al. [8] proposed a wearable wireless fall detector using accelerometers. Their threshold-based system consists of two modules, a fall detection terminal and a remote one which can communicate with each other wirelessly. Tong et al. [31] proposed a method based on the hidden Markov model (HMM), using tri-axial accelerations. They used the acceleration time series of falls before the collision to train their model. Bourke et al. [4] described a threshold-based algorithm for fall detection, using the bi-axial angular velocity, while in [10] Georgakopoulos et. al. employed online dimensionality reduction and control charts to detect changes using devices having limited CPU power and memory resources.

In the perspective of Machine Learning, the major challenge in fall detection is to build a classifier with an efficient training step in order to categorize accurately human's actions, especially to discriminate falls instead of other actions. Recent accelerometer-based approaches have utilized various classification approaches, such as Support Vector Machines (SVM) [19, 28], Decision Trees (DT) [2, 7], k-Nearest Neighbor (kNN) [18] and Recurrent Neural Network [24]. Although such well-established classification schemes offered accurate results, accelerometer-based data monitoring and parsing creates large files under process, thus an imperative need is created for classification schemes able to handle this information volume. Deep Learning approaches are appropriate for this challenge while their operation is improved proportionally to data volume. To the best of our knowledge, there is no a comprehensive study to address the crucial challenge of fall detection by monitoring human's actions with accelerometer using deep learning approaches for classification while minimizing computational requirements.

### 3 Dataset Description and Preprocessing

States of human activity can be recorded using different sensors (accelerometer, gyroscope, and magnetometer), signals (acceleration, velocity, and displacement), and direction components (vertical and non-vertical), combined. Smartphones can easily record all these kinds of information but when focusing on smartwatch/smartband devices, accelerometer data are more relevant. Thankfully, x, y and z axis accelerations can be easily recorded while often contain a fair enough amount of information to describe different human activities.

In this work we employ the dataset presented in [24], where experiments were made with seven different subjects (persons), wearing a Microsoft Band 2 device on his/her wrist and repeated different types of fall 10 times<sup>2</sup>.

The raw data were with a sampling rate frequency of 31.25hz and the total number of samples is 51192. In what follows, for consistency we employ the train-test separation provided by the authors in [24] (33984 samples for the train set and 17208 for the test set). Each sample is constituted by 3 columns, each one for every recorder acceleration in x, y and z axis. Also available is the label of every

---

<sup>2</sup> <https://userweb.cs.txstate.edu/~hn12/data/SmartFallDataSet/>.

**Table 1.** Constructed datasets along with their respective specifications

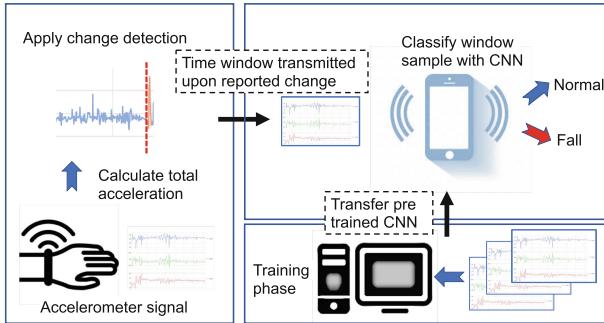
Dataset		Overlapping	Window (ms)	Label threshold	Observations	Falls
D1	Train	0%	250	0.5	4252	546
	Test	0%	250	0.5	2153	273
D2	Train	0%	750	0.5	1417	182
	Test	0%	750	0.5	717	91
D3	Train	50%	250	0.5	8504	1092
	Test	50%	250	0.5	4306	546
D4	Train	50%	750	0.5	2834	364
	Test	50%	750	0.5	1434	182

sample indicating whether it belongs to a regular activity or a fall. All samples are sorted according to their timestamp as such the dataset can be treated as a regular time series. As a result, every fall appears in the dataset as a consecutive batch of samples labeled accordingly.

In our analysis, we convert this stream of data by fusing a number of the consecutive raw samples, such that would result in a time window that would provide enough information to detect a fall. For this purpose, according to the literature [15] we endorse the hypothesis that the time window of an average fall usually lies within the range of 250–750 ms, from the beginning of loosing balance to the moment of collision. As such, for generality we consider two different time window sizes  $w$  in our analysis, one of 8 fused raw samples and one of 24 fused raw samples, so that the first time window size be  $8 \times 31.25 = 250$  ms and the second  $24 \times 31.25 = 750$  ms, respectively. Each sample of the resulting data set it is now characterized by a  $w \times 3$  matrix respectively. When needed the aforementioned matrix is transposed to a  $w \times 3$  dimensional vector.

To define the label of each window we introduce a parameter  $l$  called “label threshold” which is the ratio between the number of raw data samples labeled as falls within a window and the corresponding window size. Then we need to define a threshold  $l_h$  so that when  $l \geq l_h$  the corresponding window sample is being characterized as one that represents a fall. We experimented with a wide range of values for  $l_h$  and came to the conclusion that using a value close to 0.5 we get similar number of identified falls as the authors in [24]. The most representative cases are shown in Table 1 along with the other resulting fused data characteristics.

Finally, we consider the case of overlapping time windows in an attempt to eliminate any possibility of losing an actual fall by problematic window placement (raw samples from a fall could lie at the boundaries of two consecutive time windows). We experimented with overlapping degrees of 30%, 50% and 90%. Although it wouldn’t affect significantly the computational complexity of the proposed approach we consider the that last option appears to be an overkill due to extensive requirements in data management, as such we employ the 50%



**Fig. 1.** A complete overview of the proposed methodology.

degree of overlap to examine performance differences. The complete specifications of the generated data sets resulting from the aforementioned process are presented in Table 1 according to the total number of observations (number of fused time windows) and the corresponding number of observations belonging to the falls category.

## 4 Proposed Methodology and Prerequisites

In this Section we describe the complete methodology of an online scheme that can detect falls in real time while running continuously on a pair of connected devices. More precisely, we consider the combination of a wearable device (such as smartwatch) and a smartphone that communicate through a bluetooth connection. Given the current set up our purpose is twofold. Firstly, computations taking place on the wearable device need to be lightweight enough due to restrictions on both computational power and battery consumption while, communication through bluetooth needs to be as rare as possible to maximize battery saving on both devices.

For this purpose we propose an algorithmic scheme where fall identification takes place on the smartphone device through a sophisticated pre-trained classifier that classifies batches of sensor samples transmitted from the wearable device. We avoid continuously transmission by employing a lightweight change detection algorithm which detect significant changes upon the acceleration sensors directly onto the wearable device. When the algorithm detects a change, a batch of sensor readings is transmitted to the smartphone device for further processing. We argue and we experimentally show that is possible to define appropriate parameters such that the actual falls are detected while in the meantime, the number of false alarms is low enough to guarantee significant computational savings. To achieve this we take advantage of the total acceleration metric to transform the 3-dimension sensor signal to univariate, thus  $a_{total} = \sqrt{a_x^2 + a_y^2 + a_z^2}$  where  $a_x^2$ ,  $a_y^2$  and  $a_z^2$  each corresponding x, y and z axis accelerations. Then we

apply the univariate CUSUM methodology in an online fashion using estimated parameters based on the predefined training set.

Subsequently for the classification task, we specify the time series as a series of time windows (either overlapping or not) as described in Sect. 3 and we train the classifier in this fashion. When a change is triggered the time window that contains the change point is transmitted to the smartphone device to be classified.

The smartphone is equipped with a trained CNN with the minimum number of layers and feature maps in order to be computational efficient and achieving simultaneously, high classification accuracy. The CNN takes as input the  $w \times 3$  matrix of raw accelerometer signals that corresponds to the last  $w$  samples of the 3 dimensional accelerometer data and return the predicted class label. A complete overview of the proposed methodology is illustrated in Fig. 1.

#### 4.1 CUSUM for Change Detection

Monitoring a process over time using a change detection algorithm allows quick detection of unusual states. Usually, there are some historical process data used to construct the control limits, then the process is monitored for an ongoing basis where observations falling outside the control limits or unusual patterns of observations signal that the process has shifted from in-control process settings and as such some action need to take place.

The cumulative sum (CUSUM) algorithm was first proposed by Page in [26] for on-line and off-line change detection and it has been shown that generally performs better than Shewhart charts in detecting small shifts in the mean of a process. The CUSUM control chart have received a great deal of attention in modern industries while being an active research topic with various recent applications [11, 30, 32] and proposed variations or extensions [1, 27, 33].

For the online version of the CUSUM change detection algorithm we consider a sequence of independent random variables  $y_k$ , where  $y_k$  is a sensor signal at the current time instant  $k$  (discrete time), with a probability density  $p_\theta(y)$  depending only upon one scalar parameter  $\theta$ . Before the unknown change time  $t_0$ , the parameter  $\theta$  is equal to  $\theta_0$ , and after the change it is equal to  $\theta_1 \neq \theta_0$ . Then, the problem is to detect and estimate this parameter change.

In our case, the samples (sensor signals) are arriving at each time instant and the decision rule is computed. We will use the following notation. Let

$$S_k = \sum_{i=1}^k s_i, \quad \text{where } s_i = \ln \frac{p_{\theta_1}(y_i)}{p_{\theta_0}(y_i)}, \quad (1)$$

is the log-likelihood ratio for the observations from  $y_k$  to  $y_k$  and  $k$  be the current time instant. We refer to  $s_i$  as sufficient statistic. Let us now consider the particular case where the distribution is Gaussian with mean value  $\mu$  and constant variance  $\sigma$ . In this case, the changing parameter  $\theta$  is  $\mu$  and the sufficient statistic  $s_i$  is

$$s_i = \frac{\theta_1 - \theta_0}{\sigma^2} \left( y_i - \frac{\theta_0 + \theta_1}{2} \right). \quad (2)$$

The corresponding decision rule is then, at each time instant, to compare this difference to a threshold as follows:

$$g_k = S_k - m_k \geq h, \quad \text{where } m_k = \min_{1 \geq j \geq k} S_j. \quad (3)$$

The stopping time is

$$t_a = \min\{k : S_k \geq m_k + h\}. \quad (4)$$

This decision rule is a comparison between the cumulative sum  $S_k$  and an adaptive threshold  $m_k + h$ . Because of  $m_k$ , this threshold not only is modified on-line but also keeps the complete memory of the entire information contained in the past observations. Moreover, in the case of a change in the mean of a Gaussian sequence,  $S_k$  is a standard integration of the observations.

The detection threshold  $h$  is a user-defined tuning parameter in which the appropriate form for its determination is based on the average run length function, which is defined as the expected number of samples before an action is taken [26]. More precisely one has to set the mean time between false alarms  $ARL_0$  and the mean detection delay  $ARL_1$ . These two specific values of the ARL function depend on the detection threshold  $h$ , and can thus be used to set the performance of the CUSUM algorithm to the desired level for a particular application [12].

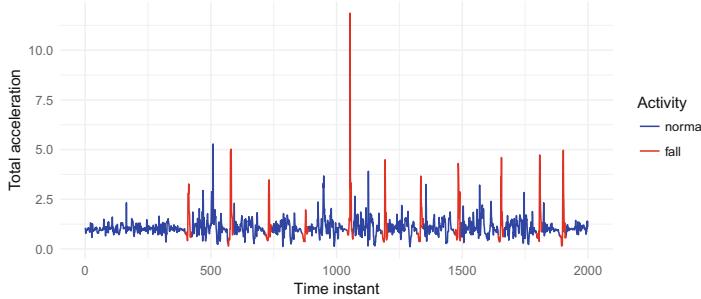
## 4.2 Convolutional Neural Networks

Convolutional Neural Networks are multi-stages model and are considering as an extensions of the well-known Multi-layer Feed-forward Neural Networks (MFNNs) characterized by a deep structure that enables feature extraction from raw input signal data through layers of adaptable filtering components. CNN have mainly applied on image data but are not limited to this type of data since there are many applications that use CNN on one-dimensional signals [6, 23].

In our case the dataset is composed by one dimensional accelerometer raw data signal of three channels. By training a CNN model with a sufficient dataset, the proper information from each temporal non-stationary signal channel are extracted through the successively convolution operations. The model is trained simultaneously with the three channels of one-dimensional signal combining the input channels in the fully connected convolutional layer. The basic CNN component is the convolutional layers. In the output of a convolutional layer a number of feature maps are produced. A feature map  $F_j$  is constructed by convolution (\*) of the output  $S_i$  of the previous layer with one dimensional kernel filters  $K_{i,j}$ , and a bias value  $b_j$ :

$$F_j = \sum_{i=1}^N S_i * K_{i,j} + b_j. \quad (5)$$

However the convolutional operations are very demanding in computations. For this reason, restricted by the computational capabilities of a mobile device, the number of convolutional layers, have to be minimum, while in the meantime,



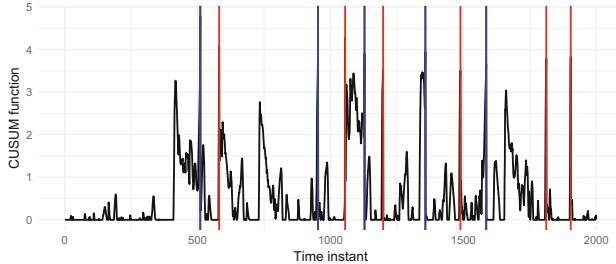
**Fig. 2.** An example of the total acceleration  $a_{total}$  annotated according to the raw sample class. (Color figure online)

maintaining the model efficiency as much as possible. The feature maps of the last convolutional layer flatten, feeding a fully-connected layers of neurons, producing feature maps with a dimension of  $1 \times 1$  elements. These types of layers belongs to the class of trainable layers (training is performed by finding suitable values for the connection weights).

## 5 Experimental Results

In the first part of our experimental analysis we investigate the performance of the CUSUM methodology onto the aforementioned dataset. The 3-dimensional times series is transformed to univariate using the methodology described in Sect. 4. We may visually investigate a sample of the resulting time series in Fig. 2 where the samples belonging to different classes are represented with different colors. As previously discussed we observe the time series nature of the dataset where subsequent falls occur after a number of normal activities. Finally, we also observe situations where although a fall is taking place, the total acceleration is not significantly high while on the other hand, there are normal activities with high values of total acceleration.

In what follows, we also employ the predefined separation into train and test sets and use the train set for parameter estimation so that  $\theta_0 = \mu_0$  and  $\theta_1 = \mu_1$ , where  $\mu_0$  and  $\mu_1$  are the mean values of the samples belonging to the normal activity and fall activity classes respectively. Then we investigate the  $h$  parameter according to the number of false alarms and missed falls. We consider a detected change as false alarm if the label of the current sample at the time instant of reported change belong to the normal activity class. In addition, if the algorithm do not detect a change during the time period of an actual fall this is considered as a missed fall. The results are reported in Table 2 where, considering that we need to minimize lost falls while also reducing computations as much as possible, we intuitively choose the value 0.5 as more appropriate for the  $h$  parameter since lower values imply significantly higher number of false



**Fig. 3.** An example of the cusum function along with the reported changes in vertical lines. Blue lines correspond to false alarms while red to correct reports. (Color figure online)

**Table 2.** The  $h$  parameter analysis with respect to reported changes, false alarms and lost falls.

$h$ value	0.1	0.3	0.5	0.7	0.9	1.1	1.3	1.5	1.7	1.9
Changes	884	668	551	476	423	382	352	332	318	295
False alarms	685	482	368	297	247	207	179	161	149	132
Lost falls	0	2	2	6	7	8	10	11	15	20

alarms. An example of the CUSUM function along with the reported true and false alarms in different colors is illustrated in Fig. 3 for the same time series example used in Fig. 2.

At this point we can estimate computational savings of the proposed approach by reporting the ratio between the number of total reported changes for  $h = 0.5$  (551) and the total number of time windows constituting the train set. For the non overlapping windows the computational savings are at least 61% and up to 87% while for the overlapping windows case is up to 94%, which actually means that there are 94% fewer transmissions of sample batches between the devices and similarly fewer classifications by the CNN on the smartwatch device. Finally, we may examine the performance of the method for  $h = 0.5$  on the test set according to the false alarms and missed falls. The algorithm reports 334 detected changes for which 232 are false alarm, while there are no missed falls, suggesting the coherent applicability of the method.

Having accomplished the task of minimizing computations with minimal cost we proceed to the next step of our analysis. To this end, we employ a series of classification methodologies to compare against the described CNN architecture in an attempt to justify the use of deep learning for the classification task at hand. These are Linear Discriminant Analysis (LDA), Support Vector Machines (SVM), k-Nearest Neighbor (KNN) and Artificial Neural Networks (ANN). A basic linear kernel have been chosen for SVM, the  $k$  parameter was set to 10 for KNN while 50 epochs and batch size equal to the number of samples have been used to train the ANN. All methods are applied to the datasets described

**Table 3.** Classification results with respect to Accuracy and Balanced Accuracy.

Dataset	LDA	SVM	KNN	ANN	CNN
	ACC/BACC	ACC/BACC	ACC/BACC	ACC/BACC	ACC/BACC
D1	0.8797/0.5366	0.8732/0.5000	0.9085/0.6987	0.9057/0.7253	0.9289/ <b>0.9233</b>
D2	0.8982/0.6599	0.8926/0.6380	0.894/0.5824	0.9512/0.8687	0.9679/ <b>0.9535</b>
D3	0.8818/0.5456	0.8732/0.5000	0.8978/0.6534	0.9008/0.6874	0.8899/ <b>0.8806</b>
D4	0.8849/0.5866	0.8731/0.5000	0.9205/0.7197	0.9177/0.6970	0.9198/ <b>0.9236</b>

in Sect. 3, where each  $w \times 3$  data sample is transformed into a  $w \times 3$  dimensional vector. In contrast, the CNN takes as input samples in the original  $w \times 3$  matrix form. Its architecture consist of one convolutional layer with 5 feature maps and one dimension kernel of size 5, one fully-connected layer with 10 neurons and an output layer with two neurons and the softmax logistic regression layer error function. All methods are practically applied to raw data instead of using any kind of feature extraction which would impose further computational requirements and would limit generalization.

For evaluation, along with the typical Accuracy score which will certainly lead to an optimistic estimate when a classifier is tested on an imbalanced dataset, we employ the Balanced Accuracy metric [5]. The results are reported at Table 3, where it is evident that the CNN method significantly outperforms others confirming our hypothesis. In general, we observe that the largest window size 750 ms (datasets D2 and D4) improves performance for all methods while for overlapping window datasets (D3 and D4) there is a slight decrease, which could be an effect of having additional windows characterized as fall in the dataset.

## 6 Concluding Remarks

Most of the classical approaches used for activity recognition and fall detection rely on heuristics or hand-crafted feature extraction methods, which limit generalization. In addition, existing solutions on device applications are rare and require extensive resource optimization. In this work, we make an attempt to tackle both of these challenges by proposing a method with wide generalization by operation upon raw sensor accelerometer signal while preserving relatively low resource demands. To achieve this we combine an efficient change detection algorithm along with Deep Learning that as shown in our experimental analysis significantly enhance classification accuracy. The propose scheme provide details about the on-line implementation of this methodology employing a smartwatch/smartband device and a smartphone. In our future work, we intend to construct and examine real world datasets generated specifically for this concept while providing further algorithmic developments.

**Acknowledgments.** This project has received funding from the Hellenic Foundation for Research and Innovation (HFRI) and the General Secretariat for Research and Technology (GSRT), under grant agreement No. 1901.

## References

1. Abujiya, M., Riaz, M., Lee, M.H.: Enhanced cumulative sum charts for monitoring process dispersion. *PloS One* **10**, e0124520 (2015)
2. Aguiar, B., Rocha, T., Silva, J., Sousa, I.: Accelerometer-based fall detection for smartphones. In: 2014 IEEE International Symposium on Medical Measurements and Applications (MeMeA), pp. 1–6. IEEE (2014)
3. Bagalà, F., et al.: Evaluation of accelerometer-based fall detection algorithms on real-world falls. *PLOS ONE* **7**(5), 1–9 (2012)
4. Bourke, A., ÓLaighin, G.: A threshold-based fall-detection algorithm using a bi-axial gyroscope sensor. *Med. Eng. Phys.* **30**, 84–90 (2008)
5. Brodersen, K.H., Ong, C.S., Stephan, K.E., Buhmann, J.M.: The balanced accuracy and its posterior distribution. In: 2010 20th International Conference on Pattern Recognition, pp. 3121–3124, August 2010
6. Brynolfsson, J., Sandsten, M.: Classification of one-dimensional non-stationary signals using the Wigner-Ville distribution in convolutional neural networks. In: 2017 25th European Signal Processing Conference (EUSIPCO), pp. 326–330, August 2017
7. Castillo, J.C., Carneiro, D., Serrano-Cuerda, J., Novais, P., Fernández-Caballero, A., Neves, J.: A multi-modal approach for activity classification and fall detection. *Int. J. Syst. Sci.* **45**(4), 810–824 (2014)
8. Chen, D., Feng, W., Zhang, Y., Li, X., Wang, T.: A wearable wireless fall detection system with accelerators. In: 2011 IEEE International Conference on Robotics and Biomimetics, pp. 2259–2263, December 2011
9. Chen, L., Hoey, J., Nugent, C.D., Cook, D.J., Yu, Z.: Sensor-based activity recognition. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **42**(6), 790–808 (2012)
10. Georgakopoulos, S.V., Tasoulis, S.K., Maglogiannis, I., Plagianakos, V.P.: On-line fall detection via mobile accelerometer data. In: Chbeir, Richard, Manolopoulos, Yannis, Maglogiannis, Ilias, Alhajj, Reda (eds.) AIAI 2015. IAICT, vol. 458, pp. 103–112. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-23868-5\\_8](https://doi.org/10.1007/978-3-319-23868-5_8)
11. Georgakopoulos, S.V., Tasoulis, S.K., Plagianakos, V.P.: Efficient change detection for high dimensional data streams. In: 2015 IEEE International Conference on Big Data (Big Data), pp. 2219–2222, October 2015
12. Granjon, P.: The CUSUM algorithm a small review (2014)
13. Greene, S., Thapliyal, H., Carpenter, D.: IoT-based fall detection for smart home environments. In: 2016 IEEE International Symposium on Nanoelectronic and Information Systems (iNIS), pp. 23–28, December 2016
14. Hsieh, K., Heller, T., Miller, A.B.: Risk factors for injuries and falls among adults with developmental disabilities. *J. Intellect. Disabil. Res.* **45**(1), 76–82 (2001)
15. Huang, C.L., Chung, C.Y.: A real-time model-based human motion tracking and analysis for human-computer interface systems. *EURASIP J. Adv. Signal Process.* **2004**(11), 616891 (2004)
16. Igual, R., Medrano, C.T., Plaza, I.: Challenges, issues and trends in fall detection systems. *Biomed. Eng. Online* **12**, 66 (2013)
17. Kau, L.J., Chen, C.S.: A smart phone-based pocket fall accident detection, positioning, and rescue system. *IEEE J. Biomed. Health Inform.* **19**(1), 44–56 (2015)
18. Kepski, M., Kwolek, B.: Fall detection using ceiling-mounted 3D depth camera. In: 2014 International Conference on Computer Vision Theory and Applications (VISAPP), vol. 2, pp. 640–647. IEEE (2014)

19. Kwolek, B., Kepski, M.: Human fall detection on embedded platform using depth maps and wireless accelerometer. *Comput. Methods Programs Biomed.* **117**(3), 489–501 (2014)
20. Ma, X., Wang, H., Xue, B., Zhou, M., Ji, B., Li, Y.: Depth-based human fall detection via shape features and improved extreme learning machine. *IEEE J. Biomed. Health Inform.* **18**(6), 1915–1922 (2014)
21. Maglogiannis, I., Doukas, C.: Intelligent health monitoring based on pervasive technologies and cloud computing. *Int. J. Artif. Intell. Tools* **23**(03), 1460001 (2014)
22. Maglogiannis, I., Ioannou, C., Tsanakas, P.: Fall detection and activity identification using wearable and hand-held devices. *Integr. Comput.-Aided Eng.* **23**, 161–172 (2016)
23. Manganaro, G., de Gyvez, J.P.: One-dimensional discrete-time CNN with multiplexed template-hardware. *IEEE Trans. Circuits Syst. I: Fundam. Theory Appl.* **47**(5), 764–769 (2000)
24. Mauldin, T.R., Canby, M.E., Metsis, V., Ngu, A.H.H., Rivera, C.C.: SmartFall: A smartwatch-based fall detection system using deep learning. *Sensors* **18**(10), 3363 (2018)
25. Gia, T.N., et al.: IoT-based fall detection system with energy efficient sensor nodes, November 2016
26. Page, E.S.: Continuous inspection schemes. *Biometrika* **41**(1/2), 100–115 (1954)
27. Perry, M., Pignatiello, J.J.: Estimating the time of step change with Poisson CUSUM and EWMA control charts. *Int. J. Prod. Res.* **49**, 2857–2871 (2011)
28. Pierleoni, P., Belli, A., Palma, L., Pellegrini, M., Pernini, L., Valenti, S.: A high reliability wearable device for elderly fall detection. *IEEE Sens. J.* **15**(8), 4544–4553 (2015)
29. Shen, R.K., Yang, C.Y., Shen, V.R., Chen, W.C.: A novel fall prediction system on smartphones. *IEEE Sens. J.* **17**(6), 1865–1871 (2017)
30. Tasoulis, S., Doukas, C., Plagianakos, V., Maglogiannis, I.: Statistical data mining of streaming motion data for activity and fall recognition in assistive environments. *Neurocomputing* **107**, 87–96 (2013)
31. Tong, L., Song, Q., Ge, Y., Liu, M.: Hmm-based human fall detection and prediction method using tri-axisial accelerometer. *IEEE Sens. J.* **13**(5), 1849–1856 (2013)
32. Tran, P.H., Tran, K.P.: The efficiency of CUSUM schemes for monitoring the coefficient of variation. *Appl. Stoch. Model. Bus. Ind.* **32**(6), 870–881 (2016)
33. Wang, D., Zhang, L., Xiong, Q.: A nonparametric CUSUM control chart based on the Mann-Whitney statistic. *Commun. Stat.-Theory Methods* **46**, 2017 (2017)
34. Wang, J., Zhang, Z., Bin, L., Lee, S., Sherratt, R.: An enhanced fall detection system for elderly person monitoring using consumer home networks. *IEEE Trans. Consum. Electron.* **60**, 23–29 (2014)
35. Xu, T., Zhou, Y., Zhu, J.: New advances and challenges of fall detection systems: a survey. *Appl. Sci.* **8**(3), 418 (2018)



# Image Classification Using Deep Neural Networks: Transfer Learning and the Handling of Unknown Images

Vedang Chauhan<sup>1</sup> , Keyur D. Joshi<sup>2</sup> , and Brian Surgenor<sup>3</sup>

<sup>1</sup> Western New England University, Springfield, MA, USA  
vedang.chauhan@wne.edu

<sup>2</sup> Ahmedabad University, Ahmedabad, Gujarat, India  
joshikeyurd@gmail.com

<sup>3</sup> Queen's University, Kingston, ON, Canada  
brian.surgenor@queensu.ca

**Abstract.** Deep learning is a subset of machine learning that is powerful at recognizing patterns and extensively used for image classification. However, it typically requires a large amount of data and it is computationally expensive for training an application from scratch. ImageNet database has millions of images pertaining to different categories that are acquired by years of hard work. Getting such a database for every application is tough and time consuming. Transfer learning is an alternative to conventional training. Transfer learning results in much faster and easier training of a network. This research set out to evaluate the effect of transfer learning on the performance of a Deep Neural Network (DNN). Pre-trained AlexNet was selected, modified and retrained for 3 image classification applications (gears, connectors and coins) with a modest database. This approach gave 99% classification accuracy using transfer learning. To test the robustness of the network, unknown images were added to one of the classes and the accuracy was reinforced using a probability threshold. This approach succeeded in compensating for the effect of unknowns in the accuracy.

**Keywords:** Image classification · Machine learning · Transfer learning · Deep neural networks · Classification accuracy · Handling unknown images

## 1 Introduction and Background

Pattern Recognition (PR) can be considered as a form of machine learning that deals with the recognition of patterns and regularities in data [1]. In supervised pattern recognition, a pattern is recognized to fit one of a number of pre-defined classes. This is known as classification. In general, however, it is difficult to have a manageable number of classes that can deal with every possible known image. This problem is made even more difficult with the introduction of unknown images (whose class is not known) to the system. A simple supervised system will classify the unknown images into one of the known classes depending on the nearest similar class. There is no mechanism to reject the unknown images by the system. In this work, this problem is

addressed. Three distinct applications (gears, connectors and coins) are examined for image classifications for test purposes.

ANN is most commonly applied as a supervised PR algorithm and is unable to deal with unknown classes. Reference [2] achieved 92% accuracy in extracting numerical information from partially degraded and aged Indian coins. They used a rotation invariant character recognition process which employed a multichannel Gabor filter together with a back-propagation ANN. Reference [3] studied the problem of coin recognition with ANNs and concluded that accuracy of the system depended on number of factors including: number of images per class, number of classes and training-testing strategies.

Study of Deep Neural Networks (DNN) is a very active area of research. Several research papers on deep learning are available for image classification [4–7]. DNN is a multilayer neural network that uses a parallel computing approach to reduce processing time. It is claimed that DNNs can approach the performance of the human brain [8]. Reference [9] demonstrated that discriminative DNN models can be easily fooled, that is, they classify many unrecognizable images with near certainty as a member of a recognizable class. To counter this criticism, Ref. [10] suggested that the likelihood of negative images (or fooling images) appearing in a training dataset was of low probability, and thus was rarely or never observed in the testing dataset.

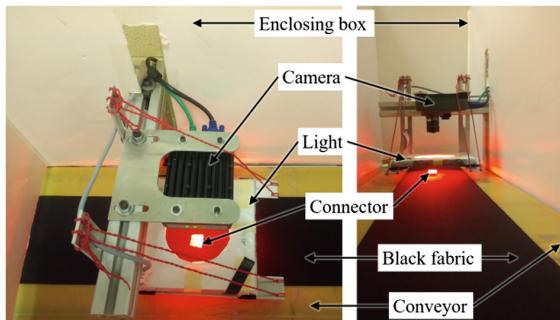
Transfer learning is a concept in machine learning where the model developed previously for another application would be used for a new but similar application with some modifications [11]. This is useful in DNN as the database tends to be very large. More information on transfer learning can be found from [12]. An interesting study of feature transferability was conducted by [13] concluding that initialization with transferred features can improve generalization performance even after fine tuning on a new problem. Our objective was to evaluate transfer learning as applied to AlexNet [4]. AlexNet works with millions of images and has over 1000 classes. It is considered to be better than LeNet [14] in terms of accuracy and less complex than GoogleNet [15]. The images in AlexNet's database were also considered to be similar to the images in our own database.

## 2 Experimental Setup

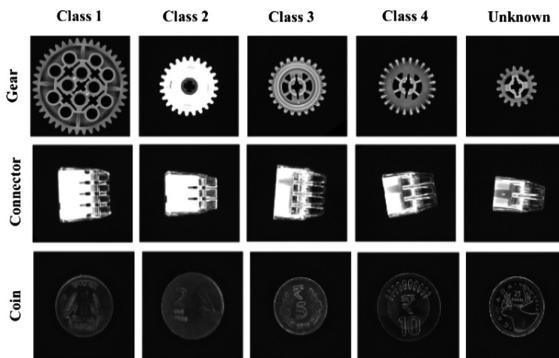
The experimental setup was developed for the three applications: (1) small plastic gears, (2) clear wire connectors and (3) metallic Indian coins. A total 4 classes were prepared for training and a 5<sup>th</sup> class was added for testing (assuming the system had the ability to deal with an unknown class).

Figure 1 shows the experimental setup and Fig. 2 gives sample images of the 5 classes for the 3 applications. The parts were fed to the conveyor using a plastic chute. The conveyor in the system was from Dorner, Model 2200 with a flush mounting package. The conveyor did not have sidewalls, a feature that provided for flexible lighting and camera arrangements. The speed of the conveyor was adjustable from 0.5 to 50 m/min. This provided a way to test the system at different speeds and to check to see whether system performance was sensitive to speed. A stretchable black fabric was used to provide a black background for the image. To detect the presence of a part, a

proximity sensor with a range of 4 mm was used. Once the part has been detected by the sensor, it signaled the camera through an Arduino Uno R3 microcontroller to acquire an image.



**Fig. 1.** Experimental setup for part recognition task (Color figure online)



**Fig. 2.** Sample images of five prepared classes for the three applications

A smart camera from National Instruments, NI 1732 was used for image acquisition. The aim of using the camera was to use the system in real time. It had an inbuilt processor which could run a program developed in Vision Builder software. For the selected applications, the largest part was 44 mm square with smallest feature around 1 mm. Therefore, a resolution of  $640 \times 480$  was found to be enough. A lens from Kowa, LM6JC, was used with the camera with 6 mm of focal length.

Previous work suggested to use dark-field lighting [16] as it can minimize the effect of shadows on the visible pattern of the parts in the image. Generally, white light provides more color detail. However, the red light was selected as the color was not a significant differentiator for the selected applications. Moreover, red light can be used for most industrial applications [17] involving a digital camera. A diffused industrial grade dark-field light from Advanced Illumination, AI 1660, was selected to illuminate the parts. The diffuser was prepared from the wax paper to smooth the effect of the sharp

lighting. The light intensity depended on the voltage supplied with the range of 16–24 V. The intensity with 19 V supply was found to be enough based on experiments.

A typical inspection speed is on the order of 100 to 300 parts/min [18]. In this work, a speed of 100 parts/min (equivalent to 3 m/min as the conveyor speed) was selected. In the setup, the light was mounted 35 mm above the surface of the part and the working distance between the part and the camera was set to 100 mm. A camera frame rate of 60 fps provided images without any visible blur. Rotational invariance was one of the requirements for our system. For each image acquired from a distinct part, twenty images were generated by digital rotation and by centering the part in the region of interest. These generated images were referred as conditioned images. A total of 2500 images per application were generated by conditioning. From the database, 2000 images were considered for the training of the 4 classes (500 images each) and 500 images were reserved for testing (100 images per class for 4 classes plus 100 images of unknown category). The images were organized in two folders: training and testing. The database has been made available online, can be downloaded from the link <http://my.me.queensu.ca/People/Surgenor/Laboratory/Database.html>. The database has two folders (1) unconditioned (w/o rotations) and (2) conditioned (w/ rotations).

### 3 Methodology

#### 3.1 DNN Background

DNN identifies discriminative features of an image directly from the training image dataset (instead of being identified manually by the user). A feature is an important property of an image that describes the specific structure of the image such as points, edges or objects. DNN takes an automatic feature extraction approach. By eliminating the step of manual feature extraction, DNN-based methods claim to be more accurate than conventional classification methods. They are popular in the field of image classification [19], speech recognition [20] and sentiment classification [21].

Depending on the number of layers and degree of complexity, different types of DNN architectures are possible. The most common architectures are (1) LeNet, (2) ALexNet and (3) GoogLeNet. LeNet is a DNNs approach originally developed to recognize handwritten digits by using backpropagation in a feedforward net with many hidden layers. In ILSVRC2012 (Large Scale Visual Recognition Challenge 2012) [22], AlexNet [4] was proposed. It was the winner of the ILSVRC2012 classification task. GoogLeNet is one of the newer DNN approaches and consists of 22 middle layers, with improved utilization of the computing resources inside the network. Initially, a DNN was built and trained from scratch using NVIDIA DIGITS. However, the accuracy obtained was disappointing [23]. It was hypothesized that a transfer learning approach would produce better results.

#### 3.2 Transfer Learning

Building and training a DNN from scratch requires expertise and considerable time to design the network. The network contains multiple layers and to determine the exact

required number of layers and the exact order of layers, various combinations are needed to be checked. Each designed network needs to be trained using the available data. Moreover, to learn the mapping between input and output data, many images are required if the network is being trained from scratch. To overcome these limitations, for this research work, a concept of transfer learning was adopted.

Transfer learning is the process of taking a pre-trained DNN and fine-tuning it to learn a new task. Using transfer learning is usually much faster and easier than training a network from scratch because it can quickly transfer learned features to a new task using a smaller number of training images. There are numerous ways to implement transfer learning. For this research, a pre-trained model approach was selected. The training images used for pre-trained networks were a subset of the ImageNet database used in ImageNet Large-Scale Visual Recognition Challenge (ILSVRC). The popular networks for transfer learning are AlexNet, LeNet, GoogleNet, VGG19, VGG16, SqueezeNet, resnet18, resnet50, resnet101 and inceptionresnetv2. These networks vary in terms of depth, size, operations per prediction, parameters and input image size. The networks with higher depth and more parameters require larger database and are computationally expensive.

Considering the above factors, AlexNet was selected as a pre-trained network. AlexNet is a convolutional neural network that is trained on more than a million images from the ImageNet database. The network is 8 layers deep and can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals. As a result, the network has learned rich feature representations for a wide range of images. The network has an image input size of  $227 \times 227$ . The images of the objects for the selected 3 applications have similarity with the images used for AlexNet.

### 3.3 Transfer Learning with AlexNet

MATLAB's Machine learning and Deep Learning Toolbox [24] was used for building and training the DNN. The toolbox has a tool developed specifically for DNNs. The tool, Deep Network Designer, provides intuitive GUI for building and training the network. The following paragraphs explain the procedure of transfer learning as applied to AlexNet.

Transfer learning was implemented by performing the following steps:

1. Select a pre-trained network and import it into the tool.
2. Replace the final layers with new layers adapted to the new data set. This includes,
  - a. Specifying the new number of classes based on the training images.
  - b. Setting learning rates to learn faster in new layers than in the transferred layers.
3. Export the network and develop the training and testing algorithm.

#### Select a Pre-trained Network and Import It into the Tool

The Deep Learning Toolbox provides a selection of pre-trained image classification networks that have learned rich feature representations suitable for a wide range of applications. Transfer learning works best when the similarity between the new images and the original images used to train the network is high. AlexNet was selected because it is one the fastest networks with a high accuracy. AlexNet has depth of 8 layers,

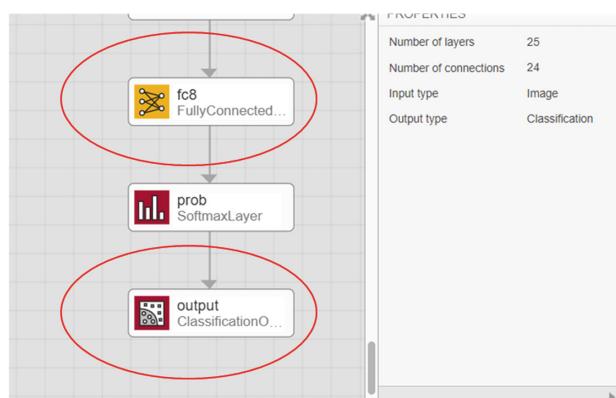
245 MB size, 0.72 Billion operations per prediction, 61 Million parameters and input image size of  $227 \times 227$  pixels. These numbers are small compared other computationally expensive network. AlexNet was subsequently downloaded and imported to the tool.

### Replace the Final Layers with New Layers Adapted to the New Data Set

AlexNet comprises of 25 layers. There are 8 layers with learnable weights: 5 convolutional layers, and 3 fully connected layers. Selecting a layer in the tool, its properties can be seen in the property pane. The network classifies input images using the last learnable layer and the final classification layer. To retrain a pre-trained network to classify new images, these final layers were replaced with the new layers adapted to the new data set. To use a pre-trained network for transfer learning, the number of classes updated to match the new data set. The default AlexNet has 1000 output classes. The number of output classes for the applications of this research work were 4. For AlexNet, the last learnable layer is a fully connected layer. This layer was deleted and a new fully connected layer with the output size of 4 was inserted to replace it. Similarly, the final output classification layer also needed to be replaced to reflect the number of output classes in the given database. The number of outputs adjusted to the correct size during training. Both these layers are highlighted in Fig. 3. The network parameters of the fully connected layer were adjusted to improve the learning procedure. Weight learn rate factor and Bias learn rate factors were set to 10. Once modified, the network was analyzed to assure that no errors and warnings were present.

### Export the Network and Developing the Training and Testing Algorithm

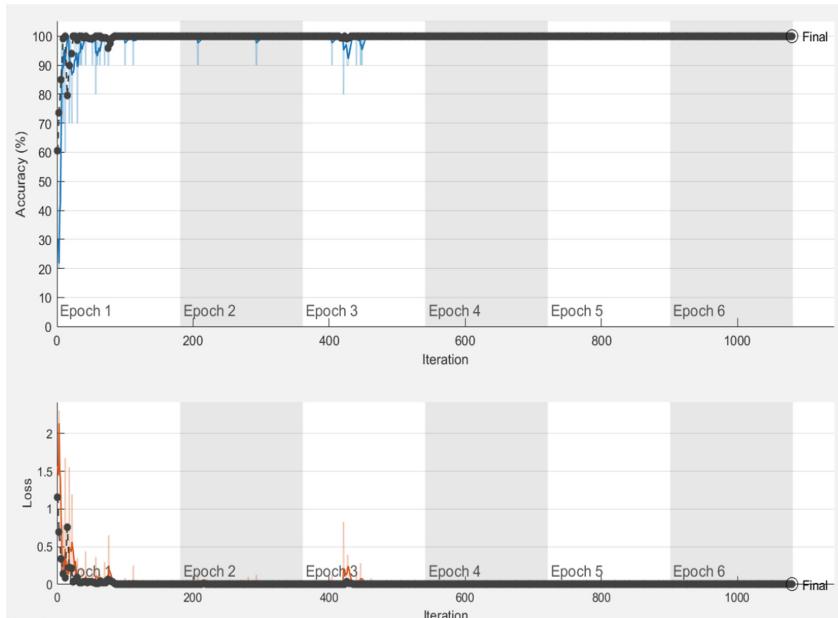
The modified network was renamed and exported to the workspace. An algorithm was written to feed training and test images to the network. Images in the database were  $340 \times 340$  pixels. The images were resized to  $227 \times 227$  for input to the network. The training database for each application contained 2000 images in 4 classes. The images were divided 90% for training (1800 images) and 10% for validation (200 images). A separate database of 500 images was used for testing the prediction accuracy of the network.



**Fig. 3.** Modifying AlexNet for transfer learning

Before training, the network parameters were adjusted to improve the training process. For transfer learning, Initial Learn Rate was set to a small value to slow down learning in the transferred layers. Also, the learning rate factors for the fully connected layer were increased to speed up learning in the new final layers. This combination of learning rate settings resulted in fast learning only in the new layers and slower learning in the other layers. The number of epochs was set to 6. An epoch is a full training cycle on the entire training data set. For transfer learning, training for many epochs is not required. The data were shuffled every epoch. The mini-batch size, that is the number of images to use in each iteration was set to 10. The validation frequency was set to 3 and the training plot to monitor progress was turned on. The network was trained for each application separately. The output for the training plot is shown in the Fig. 4. The plot shows how the accuracy improved over the epochs and the loss reduced. It took 50 min to train the network with a single CPU with spec i7, 1.8 GHz, 16 GB RAM.

The network outputs final class with a probability associated with it. The developed algorithm calculated the classification accuracy of the network with and without considering the probability. For accuracy with the probability, a probability threshold of 99% was used to filter the results. Having a high probability threshold resulted in a few false negative results (i.e. rejected some of the correct classifications). However, it resulted in correct output with a high confidence. To test the reliability of the network, it was also tested with unknown class images. These were images that did not fit to any of the trained classes and were not the part of training. The confusion matrix and accuracy were determined for testing with unknown classes with and without considering the probabilities. The analysis of the results is given in the next section.



**Fig. 4.** AlexNet training using transfer learning

## 4 Results and Discussion

Pre-trained AleNex was modified and trained using the transfer learning for all 3 applications (gears, connectors and coins). Due to a strong similarity between images of the pre-trained AlexNet and the new database, the transfer learning resulted in a faster learning (average training time 50 min) with a very high accuracy achieved with the minimum iterations of 1080. As a result, the training and validation accuracy was 100% for the new database. The separate database of test images was used for additional testing. The test images contained 100 images of each class for 4 classes (i.e. 400 images) and additional 100 images of unknown objects. Hence, a total 500 images were used for testing. The unknown object images were never seen during training and didn't fit any of the 4 training classes. The network was trained and tested for each application separately. For simplicity, the intermediate results are presented only for the connector application and the overall results are presented for all 3 applications.

For each test image, the network outputs a class and a probability associated with the classification. The classification accuracy was calculated based on the number of correct classifications by the network for the test database. The accuracy was determined under 4 distinct test conditions: (1) testing with known classes without considering the probabilities, (2) testing with added unknown class without considering the probabilities; (3) testing with known classes and considering the probabilities and (4) testing with added unknown class and considering the probabilities. The results were presented as confusion matrices and the accuracies were reported. Since, the network was originally trained with only 4 classes, for batch testing it could be tested with only 4 classes. Thus, unknown class images were added to Class 4 for the test conditions (2) and (4).

The confusion matrix for the classification accuracy for test condition (1) is given in Table 1. Since, only 4 known classes were used for testing, the network resulted in the perfect classification of 100%. For each test class, all 100 images were predicted as the correct class using the network. To prove the robustness of the network, 100 images of unknown objects were added to Class 4 for the test condition (2). The confusion matrix is presented in Table 2. For this test condition, the output of the network was considered without its probability. It can be seen that out of 200 images of Class 4, all images were predicted as Class 4 by the network. i.e. 100 images of the unknown class were also predicted as Class 4 images. The resulting classification accuracy was 80%.

**Table 1.** Connector test results with known classes and without probabilities

Connectors w/o prob	Known classes only	Predicted class				
		Class 1	Class 2	Class 3	Class 4	Unknown
Actual class	Class 1	100	0	0	0	0
	Class 2	0	100	0	0	0
	Class 3	0	0	100	0	0
	Class 4	0	0	0	100	0
	Unknown	0	0	0	0	0

Classification accuracy: 100%

**Table 2.** Connector test results with an added unknown class and without probabilities

Connectors w/o prob	Unknown added to class 4	Predicted class				
		Class 1	Class 2	Class 3	Class 4	Unknown
Actual class	Class 1	100	0	0	0	0
	Class 2	0	100	0	0	0
	Class 3	0	0	100	0	0
	Class 4	0	0	0	100	0
	Unknown	0	0	0	100	0

Classification accuracy: 80.00%

The confusion matrix for the classification accuracy for test condition (3) is given in Table 3. For this condition, the classification results were reinforced by considering the probability of each classification. For each test image, first the output class was predicted using the network. In the second stage, the probability associated with that classification was determined. If the probability score was higher than 99%, then only the output class was retained, otherwise the output was referred as an unknown class. The high probability threshold might result in some false negatives. i.e. the correct predicted class treated as the unknown class. For example, 5 images of Class 2; 2 images of Class 3; and 4 images of Class 4 were referred as unknown class due to their low probability scores. The classification accuracy was 97.60% for this test condition.

For test condition (4), the unknown class images were added to Class 4 and the results were strengthened by considering the probabilities. The confusion matrix is provided as the Table 4. Using the probability, out of 100 images of unknown class, 27 were predicted as Class 4 and 73 were predicted as the unknown class. Comparing the results with the condition (3) were all 100 unknown class images were misclassified as Class 4, the test condition (4) resulted in only 27 misclassifications. As a result, the accuracy improved from 80% to 93.20%.

For the overall classification accuracy, the results of all three applications are presented in Fig. 5 (left). The accuracy results are presented without considering the probabilities. The large drop in the accuracy can be seen between the results of testing with known classes only and the added unknown class.

**Table 3.** Connector test results with known classes and with probabilities

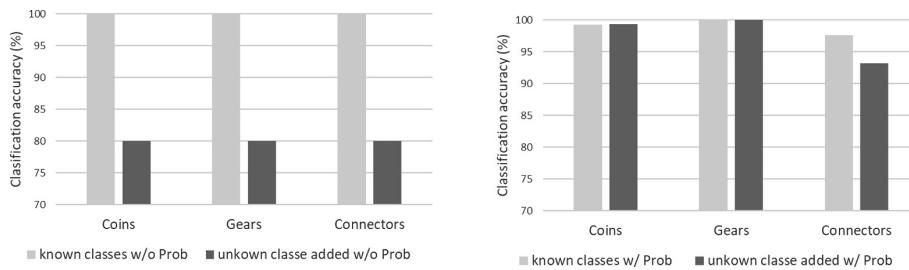
Connectors w/ prob	Known classes only	Predicted class				
		Class 1	Class 2	Class 3	Class 4	Unknown
Actual class	Class 1	100	0	0	0	0
	Class 2	0	95	0	0	5
	Class 3	0	0	98	0	2
	Class 4	0	0	0	95	5
	Unknown	0	0	0	0	0

Classification accuracy: 97.60%

**Table 4.** Connector test results with an added unknown class and with probabilities

Connectors w/ prob	Unknown added to class 4	Predicted class				
		Class 1	Class 2	Class 3	Class 4	Unknown
Actual class	Class 1	100	0	0	0	0
	Class 2	0	95	0	0	5
	Class 3	0	0	98	0	2
	Class 4	0	0	0	100	0
	Unknown	0	0	0	27	73

Classification accuracy: 93.20%

**Fig. 5.** Classification accuracy without probability (left) and with probability (right)

The results of the classification accuracy for all 3 applications with considering the probabilities are given in Fig. 5 (right). For these results, the output of AlexNet was corrected based on the probability of the classification and the probability threshold. As a result, some false negatives were reported for testing with only known classes. On the upside, the classification accuracy improved significantly for testing with the added unknown class. The accuracy difference between testing with known class and added unknown class reduced greatly compared to the results without considering the probabilities.

## 5 Conclusions and Future Work

Deep neural networks (DNNs) are excellent tool for image classification. Building and training a DNN from scratch requires time, copious amounts of data and is computationally expensive. Moreover, a DNN consists of many layers. The performance of the network varies based on the number and order of these layers. Moreover, during training, millions of internal parameters are required to be tuned. Transfer learning is an alternative solution. Researchers in the field of machine learning have contributed to the community by sharing their pre-trained networks. For a given application, a similar pre-trained network can be accessed, modified and retrained with a new database. Using transfer learning is usually much faster and easier than training a network from scratch as learned features can be quickly transferred to a new task using a smaller number of training images. In this research, AlexNet trained on ImageNet database was

selected, modified and retrained. The training with 4 classes took only 50 min and resulted in 100% validation accuracy. During testing, the network was tested with both known classes only and an added unknown class. The accuracy of the network dropped from 100% to 80% with the addition of an unknown class. However, reinforcing the classification results using the probabilities and setting the higher probability threshold, significantly improved the accuracy. The accuracy increased from 80% to 97%. The cause of errors were some false negative results where the output class was considered as unknown due to a low probability score. For the coin and gear applications, the accuracy improved by 20% for testing with added unknown images. For the connector application, the accuracy was improved by 17% for testing with unknown images.

In order to test the reliability and robustness of transfer learning, further tests should be conducted with different types of parts. Future work includes testing with new databases and more images in each database. There are more than a dozen pre-trained networks available for transfer learning. Examining the effect of transfer learning applied to these newest networks is the goal for future work. Each pre-trained network has many tuning options. Guidelines need to be developed regarding the modification and tuning of pre-trained networks using transfer learning. The effect of batch size, learning rate and other parameters can be the subject of future work. To improve the reliability and robustness of the network in handling unknown class images, the fusion of ANN or fuzzy based network with DNN should be studied. Also, the effect of image quality on the performance of the network will be examined. An image quality index will be defined like the one provided in the literature, for example [25].

## References

- Dougherty, G.: Introduction. In: Dougherty, G. (ed.) *Pattern Recognition and Classification*, pp. 1–7. Springer, New York (2013). [https://doi.org/10.1007/978-1-4614-5323-9\\_1](https://doi.org/10.1007/978-1-4614-5323-9_1)
- Bremananth, R., Balaji, B., Sarkari, M., Chitra, A.: A new approach to coin recognition using neural pattern analysis. In: IEEE Indicon Conference, Chennai, India, pp. 366–370 (2005)
- Chauhan, V., Joshi, K.D., Surgenor, B.: Machine vision for coin recognition with ANNs: effect of training and testing parameters. In: Boracchi, G., Iliadis, L., Jayne, C., Likas, A. (eds.) *EANN 2017. CCIS*, vol. 744, pp. 523–534. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-65172-9\\_44](https://doi.org/10.1007/978-3-319-65172-9_44)
- Krizhevsky, A., Sutskever, I., Hinton, G.: ImageNet classification with deep convolutional neural networks. In: *Advances in Neural Information Processing Systems*, pp. 1097–1105 (2012)
- LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. *Nature* **521**, 436–444 (2015)
- Rastegari, M., Ordonez, V., Redmon, J., Farhadi, A.: XNOR-Net: ImageNet classification using binary convolutional neural networks, pp. 1–17. [arXiv:1603.05279v4](https://arxiv.org/abs/1603.05279v4) (2016)
- Fadaeddini, A., Eshghi, M., Majidi, B.: A deep residual neural network for low altitude remote sensing image classification. In: 6th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS), Kerman, Iran, pp. 43–46 (2018)
- Ciresan, D., Meier, U., Schmidhuber, J.: Multi-column deep neural networks for image classification. In: *Computer Vision and Pattern Recognition*, Providence, USA, pp. 3642–3649 (2012)

9. Nguyen, A., Yosinski, J., Clune, J.: Deep neural networks are easily fooled: high confidence predictions for unrecognizable images. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, USA, pp. 427–436 (2015)
10. Szegedy, C., et al.: Intriguing properties of neural networks. arxiv preprint [arXiv:1312.6199](https://arxiv.org/abs/1312.6199) (2014)
11. Torrey, L., Shavlik, J.: Transfer learning. In: Handbook of Research on Machine Learning Applications (2009)
12. Pan, S.J., Yang, Q.: A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **22**(10), 1345–1359 (2010)
13. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: Advances in Neural Information Processing Systems 27, NIPS 2014, pp. 1–14 (2014)
14. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**(11), 2278–2324 (1998)
15. Szegedy, C., et al.: Going deeper with convolutions. In: Conference on Computer Vision and Pattern Recognition (CVPR), Boston, USA, pp. 1–9 (2015)
16. Joshi, K., Chauhan, V., Surgenor, B.: Real-time recognition and counting of Indian currency coins using machine vision: a preliminary analysis. In: Proceedings of the Canadian Society for Mechanical Engineering (CSME) International Congress, Kelowna, Canada (2016)
17. Cognex: Color Light Selection Guide. <https://www.cognex.com/products/machine-vision/2d-machine-vision-systems/in-sight-7000-series/colored-light-selection-guide>. Accessed 31 Mar 2018
18. Chauhan, V.: Fault detection and classification in automated assembly machines using machine vision. Doctoral thesis, Department of Mechanical and Materials Engineering, Queen's University, Canada, (2016)
19. Shah, S., Bennamoun, M., Boussaid, F.: Iterative deep learning for image set based face and object recognition. *Neurocomputing* **174**, 866–874 (2015)
20. Noda, K., Yamaguchi, Y., Nakadai, K., Okuno, H., Ogata, T.: Audio-visual speech recognition using deep learning. *Appl. Intell.* **42**, 722–737 (2015)
21. Zhou, S., Chen, Q., Wang, X.: Active deep learning method for semi-supervised sentiment classification. *Neurocomputing* **120**, 536–546 (2013)
22. Russakovsky, O., et al.: ImageNet large scale visual recognition challenge. *Int. J. Comput. Vis.* **115**(3), 211–252 (2015)
23. Joshi, K.D.: A flexible machine vision system for small part inspection based on a hybrid SVM/ANN approach. Doctoral thesis, Department of Mechanical and Materials Engineering, Queen's University, Canada (2018)
24. Machine Learning and Deep Learning Toolbox, MATLAB R2018b (2018)
25. Mittal, A., Soundararajan, R., Bovik, A.: Making a completely blind image quality analyzer. *IEEE Signal Process. Lett.* **20**(3), 209–212 (2013)



# LEARNAE: Distributed and Resilient Deep Neural Network Training for Heterogeneous Peer to Peer Topologies

Spyridon Nikolaidis<sup>(✉)</sup> and Ioannis Refanidis

University of Macedonia, 54636 Thessaloniki, Greece  
[{sp.nikola,yrefanid}@uom.edu.gr](mailto:{sp.nikola,yrefanid}@uom.edu.gr)

**Abstract.** LEARNAE is a framework proposal for decentralized training of Deep Neural Networks (DNN). The main priority of LEARNAE is to maintain a fully distributed architecture, where no participant has any kind of coordinating role. This solid peer-to-peer concept covers all aspects: Underlying network protocols, data acquiring/distribution and model training. The result is a resilient DNN training system with no single point of failure. LEARNAE focuses on use cases where infrastructure heterogeneity and network unreliability result to an always changing environment of commodity-hardware nodes. In order to achieve this level of decentralization, new technologies had to be utilized. The main pillars of this implementation are the ongoing projects of IPFS and IOTA. IPFS is a platform for a purely decentralized filesystem, where each node contributes local data storage. IOTA aims to be the networking infrastructure of the upcoming IoT reality. On top of these, we propose a management algorithm for training a DNN model collaboratively, by optimal exchange of data and model weights, always using distribution-friendly gossip protocols.

**Keywords:** Decentralized neural network training · Distributed asynchronous stochastic gradient decent · Model averaging · Peer-to-Peer topologies · Distributed Ledger Technology · IPFS · IOTA

## 1 Introduction

During the past years, scientific community has made significant advances regarding parallel DNN training. There are many aspects that can be dealt with in a decentralized way. Each implementation may also have a different level of decentralization, based on the intended use case. In most cases the proposed works rely on a centralized approach like a parameter server [1, 2]. These approaches require high performance infrastructure, since all nodes have to communicate with the server and exchange models after each optimization step. Decentralized attempts, such as [3], use local optimization and asynchronous model merging reducing the communication demand, but the parameter server is still a bandwidth bottleneck limiting scalability. Other distributed deep learning systems [4, 5] are able to overcome this bottleneck, but for doing so they need low-latency networking like InfiniBand, which results to very high setup cost and narrows down use cases. Proposals like [6] focus on medium performance hardware

but they still utilize synchronous frameworks like [7], which are not the optimal choice for loosely connected peers.

In this paper we propose a framework that stretches decentralization and tolerance to maximum. Based on purely distributed peer-to-peer technology, it has no need for servers or any kind of strict synchronization. Its intended use case are environments with commodity-hardware nodes and networking infrastructure with moderate latency and connectivity. Our approach supports versatile data acquiring from different types of sources, including lightweight IoT devices, and uses novel Distributed Ledger Technologies as the data diffusion mechanism.

The rest of the paper is structured as follows: Sect. 2 presents technological background knowledge, whereas Sect. 3 presents the LEARNAE architecture. Section 4 presents experimental results to evaluate the whole approach and, finally, Sect. 5 concludes the paper and identifies future research challenges.

## 2 Technological Background

This section presents the related technological background that forms the basis for the LEARNAE framework.

### 2.1 IPFS

IPFS is a distributed filesystem for peer-to-peer networks, where no node has special privileges compared to others. It is comprised of previously established and successful techs, plus a novel block exchange protocol. Its main goal is to achieve efficient storage and availability of big data, with no need of a centralized authority, supporting features like immutability, deduplication, versioning and content-based addressing [8].

IPFS may be seen as an intuitive combination of Git [9], the distributed source code version control system, and BitSwap, a novel data replication incentivizing algorithm inspired by BitTorrent [10], the decentralized block exchange protocol.

In IPFS every node is identified by a unique ID which is the cryptographic hash of a public key. Although a node owner has the freedom to change this ID, it is not advised, since by doing so they will lose all the benefits the node has gained by its participation to the network. Unlike most filesystems, IPFS is not using an addressing method based on the location the data are stored in, instead any file can be acquired by just using the cryptographic hash of its contents. Files larger than a specific size are sliced to blocks which are assigned with their corresponding hash. In order to route a request, each IPFS node utilizes a Distributed Hash Table (DHT), a ledger (based on [11–13]) that contains pairs of block/peer information. For a small block, DHT contains the actual block data, while for a larger one it contains the IDs of peers that can serve the specific block.

Regarding block exchange strategy, every IPFS node maintains a list of blocks it needs and one of blocks it already has. A major difference compared to BitTorrent's method [14] is that those lists are not limited to a specific torrent or a group of such, but

may include any block that has appeared to the network, no matter what file they are part of. Since there will be cases where two peers will not need a block stored in each other, or cases where a node needs nothing, some kind of credit has to be applied. Each node is incentivized to increase its credit with its peers, because by doing so it also increases the possibility others will help it acquire the blocks it will need in the future. For a pair of peers this credit is in fact the balance of verified bytes exchanged between the two.

Since the main concept of IPFS is decentralization, no data servers could be used to store the files. All nodes participating to the network must provide some local storage (similar to [15]). Every requested block is fetched from another peer and then saved to the local storage. If a peer owner wishes to permanently retain a file locally, they can “pin” it to avoid garbage collection. Considering all the above, IPFS should be able to be used as resilient big dataset distribution method, ensuring load balancing between the nodes and a configurable data replication, in order to anticipate node downtime on loosely connected commodity hardware.

**PubSub.** IPFS supports a Publish-Subscribe scheme which allows instant messaging. If a peer needs access to a specific “topic”, it subscribes to it and by doing so can listen and broadcast data to that channel. As expected, PubSub has no need for centralized authorities and all its messages propagate using gossip protocol. LEARNAE utilizes IPFS PubSub feature to exchange training metadata among the peers.

## 2.2 IOTA

The IOTA network [16] aims to create a decentralized infrastructure to support all forms of data exchange in the upcoming Internet of Things (IoT) era. Although it supports a cryptocurrency token, automated micropayments between devices is just a portion of its use cases. It is based on a structure known as Directed Acyclic Graph (DAG), which in the IOTA community is referred to as the “Tangle” [17]. In general IOTA attempts to achieve a consensus via permissionless procedures, using the Tangle as a Distributed Ledger Technology (DLT) and a gossip protocol to propagate transactions throughout the network. In order to attach a new transaction, a node has to confirm two previous transactions, so each addition contributes to the overall stability and performance of the network [18].

**Masked Authenticated Messages.** The IOTA feature-of-interest for this paper is known as “Masked Authenticated Messages” (MAM). This feature allows lightweight IoT devices to attach zero-valued transactions to the Tangle by just performing a small amount of “Proof of Work” (PoW), which is solving a cryptographic puzzle. PoW is necessary in order to avoid spamming from bad actors. Each of these transactions includes a data portion, making MAM an IoT-oriented way to broadcast data streams, which can support both encryption and authentication.

MAM messaging is based on a publish-subscribe logic. A data stream is constructed as a singly linked list, where each transaction points to the next one. If someone knows the address (called “root”) of a specific transaction, they can read all of

the following stream, but will have no access to previous data. Knowing the first root of a MAM chain grants access to all the messages it contains.

A MAM stream may have one of three different accessibility modes: Public, Restricted and Private. In Public mode, everyone can view the messages. In Restricted mode, only those who know a “sidekey” can read the data. In private mode, only the stream owner can access the data, since it requires knowing the seed that created the MAM root sequence.

## 3 Implementation

### 3.1 Design Decisions

**Parallelism Type.** The first decision that has to be made is between model [19] and data [20, 21] parallelism, although there are works that propose hybrid systems. LEARNAE adopts data parallelism. According to this approach, each worker keeps the entire model locally and processes it using a subset of the training data.

**Propagation.** After processing on workers, the produced models have to be combined. The two major methods for doing so are weight and update averaging, each of them having its own advantages and drawbacks. This proposal uses weight averaging, thus after training the actual value (not just the update) of each model parameter is - under specific conditions - averaged with the actual value of the correspondent parameter of a selected worker’s model [22].

**Coordination.** The method for merging the above models can also have different levels of decentralization, as seen in the following Table 1:

**Table 1.** Different approaches regarding training coordination.

Level of decentralization	Coordinator entity
None	Parameter server
Low	Cluster of parameter servers
Medium	Some peers have elevated role
High	None

A parameter server has the task of receiving, combining and redistributing the averaged data. The use of a server in most cases results in a training speed increase, but it also creates a single point of failure and - in large scale networks - a bandwidth bottleneck. This drawback can be reduced using more servers that cooperate with each other. For cases where the presence of a server is not feasible or desired, some of the participating peers are assigned special coordinating tasks, while also functioning as all other training workers. At the edge of this spectrum are the implementations where no

node has additional duties regarding coordination, creating a fully distributed environment, which is the design followed by the current proposal.

**Synchronicity.** The training procedure may be synchronous or asynchronous. In synchronous designs the coordinating entity ensures that all results are only combined with others produced at the same training phase. In asynchronous designs there is no such need and the results of a worker can be embedded into the global model under more loose time-based rules. Each approach has pros and cons. Synchronous training may converge faster, since it prevents merging of irrelevant data, but it may have locks from slow peers that can delay the whole process. Asynchronous training achieves high worker utilization, but suffers from high gradient staleness, meaning that by the time a worker submits its results they are already out of date compared to the global model. Many strategies have been proposed [23] to mitigate those downsides, resulting to a large number of variants, especially for Asynchronous Stochastic Gradient Decent. LEARNAE adopts an asynchronous design, although it contains features which, if used in future implementations, may inject a configurable amount of synchronicity.

### 3.2 Architecture

LEARNAE is based on a versatile working scheme and can adapt to different environments. Regarding data, it supports use cases where all training data are poured into the network during the initial phase, before any training. It also supports use cases where data feeding is a continuous task and the training of the DNN model is an always-progressing procedure. Thus, there is no limitation about the time training data may be injected into the network, which is a critical feature when it comes to censor data streaming from IoT devices.

For enhanced versatility there are 4 different types of node roles (Table 2). What role a node has depends on whether it has access to training data and on its computational capability.

**Table 2.** Supported node types and their features.

Node role	Platform	Model training	Data feeding
Full	IPFS + IOTA	Yes	Yes
Trainer	IPFS + IOTA	Yes	No
Feeder (fat)	IPFS	No	Yes
Feeder (thin)	IOTA	No	Yes

The first three roles in Table 2 require enough computational power to support participation to the IPFS swarm. The fourth role is indented for IoT domain, since data streaming through MAM messages may be broadcasted even by lightweight sensor devices. Figure 1 demonstrates data flow between nodes of different roles.

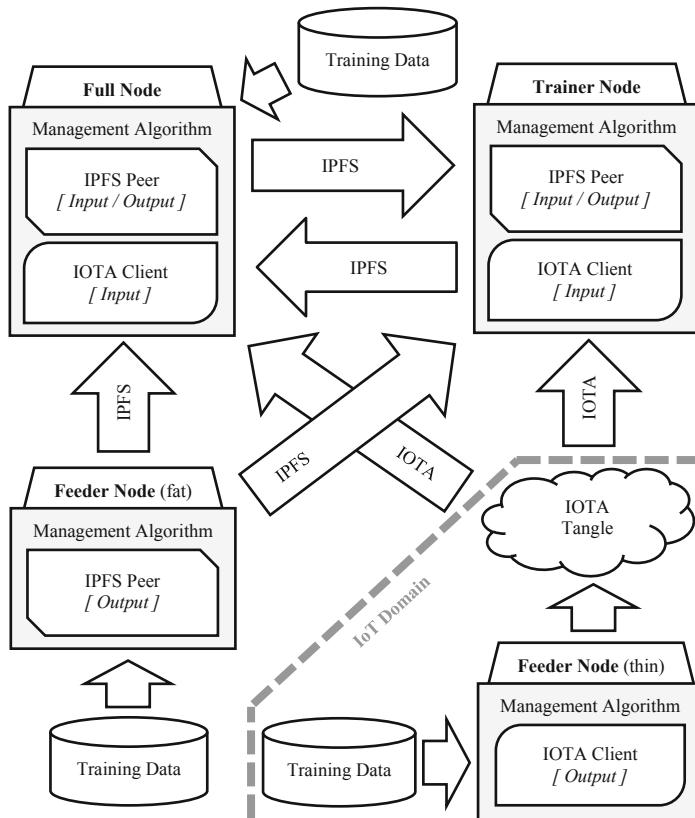


Fig. 1. Data flow between different node roles.

The ID of the used publish-subscribe channel (IPFS and/or IOTA) is the connecting link between peers. Knowing this ID allows a peer to participate to the network by listening and broadcasting messages. Figure 2 shows the workflow of a node's “Listening thread”. There are 3 different types of messages:

**Slice Hashlist.** This message contains the hash of a file that a node made available on IPFS. This file contains a list of hashes of training data slices that were also made available on IPFS by the same peer.

**Remote Model.** This message contains the hash of a file that a node made available on IPFS. This file contains the model of that peer in HD5 format. Other metadata of the model are also included, like the achieved accuracy and the maturity of the model (the number of the training cycles elapsed up to its creation).

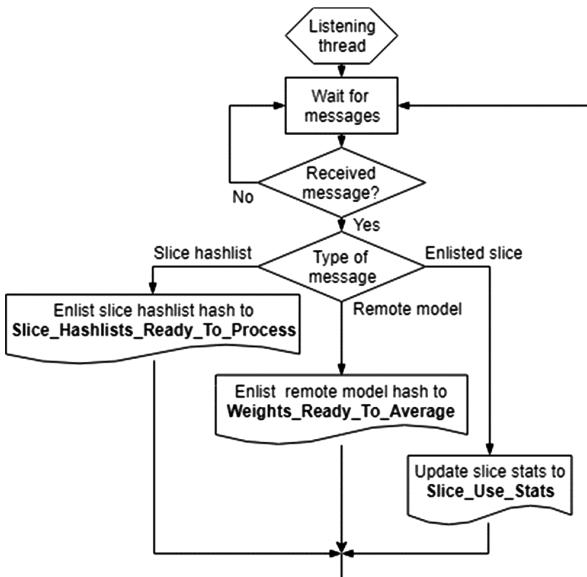
**Slice Use Stats.** This message informs all participating peers that a specific data slice has been used for training by a peer. This info allows the implementation of “overuse threshold” feature, that is optionally setting a limit on how many times a data slice may be used for training on different nodes.

A node may perform up to four main tasks, as described in Table 3:

**Table 3.** Node tasks

Task	Description
Adding	Add new data to IPFS
Pinning	Make remote IPFS data available locally
Training	Train local model
Averaging	Average local and a remote model

In order to maximize node utilization, Learnae uses a different execution thread for each task. All of them can work simultaneously, with the exception of the pair Training/Averaging, since those both need read-write access to local model. Figures 3 and 4 demonstrates the work cycle of a full node.

**Fig. 2.** Workflow of a node's “Listening thread”.

## 4 Evaluation

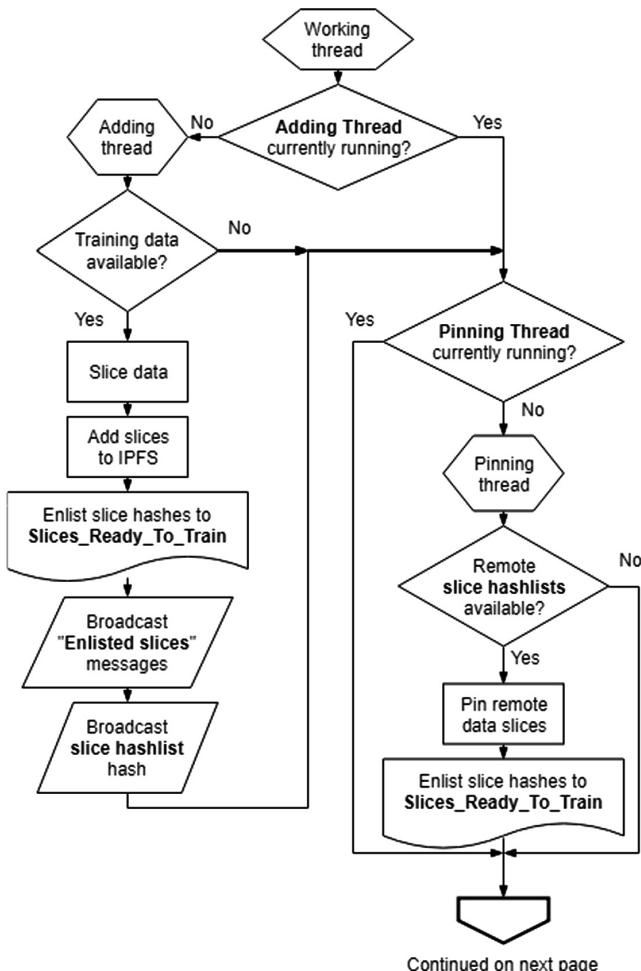
### 4.1 Current Scope

This first paper is a preliminary study on the viability of using modern DLTs as data diffusion mechanism of an asynchronous Stochastic Gradient Decent algorithm. Since this is the initial approach, the following scope limits are applied:

- The performance of IPFS under the described use case will be reviewed. All tests will be run on a single machine, simulating different peers using virtualization. This

paper focuses on proof-of-concept metrics and all tests will be short runs of one-hour timespan. Although the training dataset contains approximately 10 million instances, only a limited fraction of those were used.

- Full deployment on actual network of commodity hardware, comparison to traditional methods, extended time runs, quantity-based metrics and full training data usage will be studied in a subsequent work.



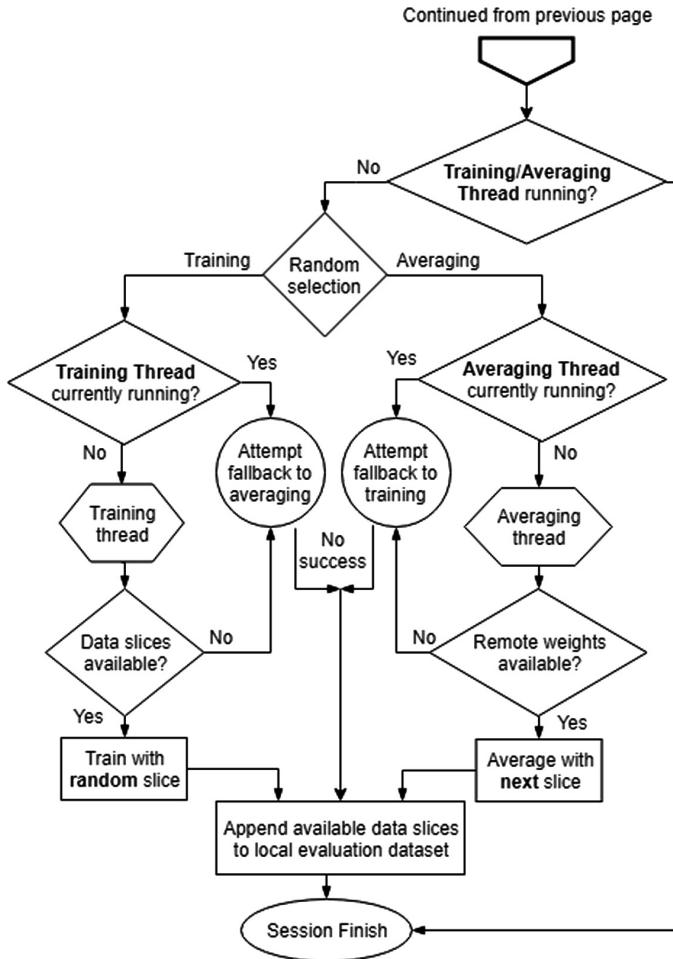
**Fig. 3.** Workflow of a node's “Working thread” (1/2).

## 4.2 Simulation

The simulation runs of this first approach were executed on a virtual network of 10 workstations. The network was implemented on a single commodity computer with

Docker containers running the IPFS daemons. The coordinating application was developed in C# and it contains both the distributed training management algorithm and the logging mechanism. The dataset used was HEPMASS [24] (exotic particle detection).

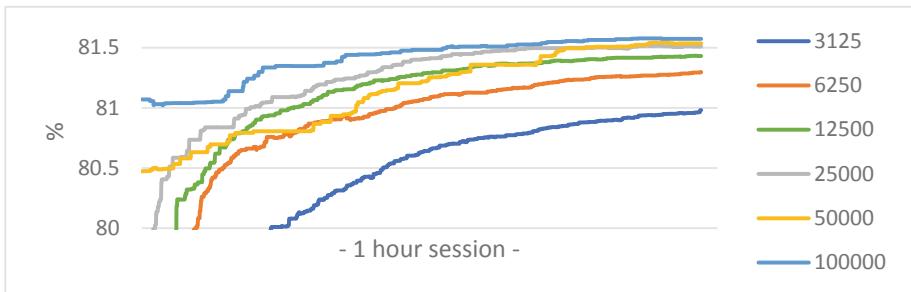
As seen in the following figures, simulation aims to study the effect of some key characteristics like data slice size and overuse threshold, among other resilience-oriented features like duplication level and overhead.



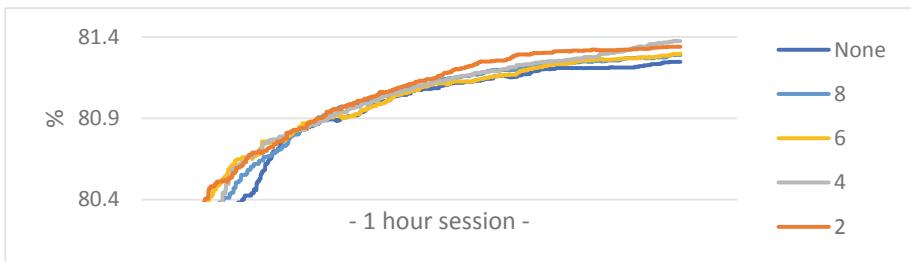
**Fig. 4.** Workflow of a node’s “Working thread” (2/2).

Simulation shows that increase in slice size has a positive impact on average accuracy of the produced models (Fig. 5). Lower overuse threshold results to slightly better accuracy, since it reduces repetitive training with same data (Fig. 6).

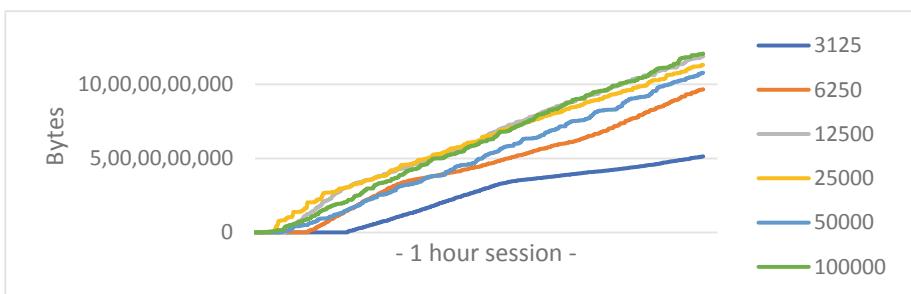
As expected, total bytes sent are proportional to selected slice size (Fig. 7). The same applies to average resilience (Fig. 8), which is defined as the number of nodes owning a requested slice/model. Figure 9 demonstrates the percent of duplicate data sent, that is an unavoidable overhead due to gossip-based protocol. This percentage, although enormously high at the beginning, is quickly declining by time and minimized for larger slice sizes.



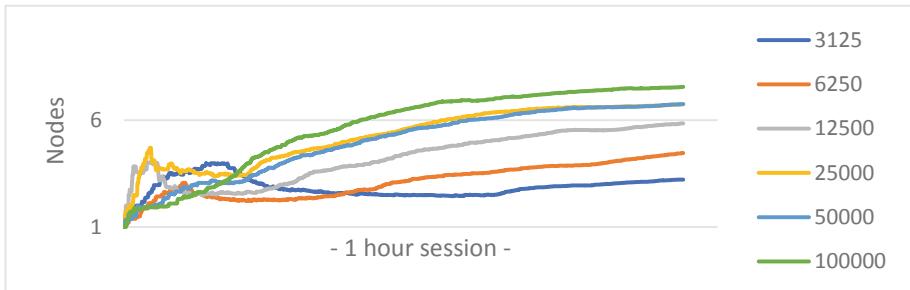
**Fig. 5.** Average accuracy per slice size (overuse threshold: 6)



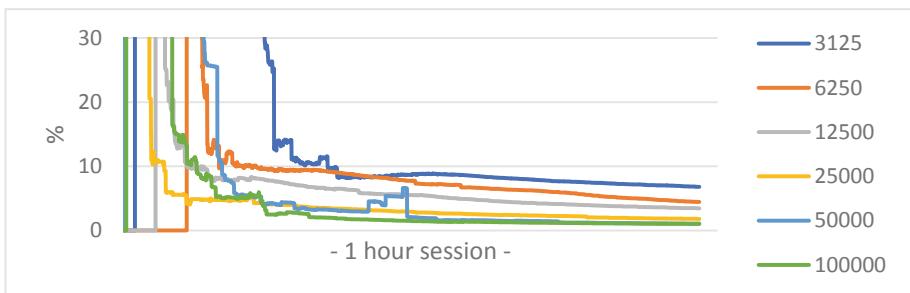
**Fig. 6.** Average accuracy per overuse threshold (slice size: 6250)



**Fig. 7.** Total bytes sent per slice size (overuse threshold: none)



**Fig. 8.** Average resilience per slice size (overuse threshold: 2)



**Fig. 9.** Duplicate data sent per slice size (overuse threshold: none)

## 5 Conclusions and Future Work

This paper is a first approach on using Distributed Ledger Technology as the data diffusion mechanism for decentralized DNN training, in order to achieve a fully distributed and peer-to-peer architecture. Thus, this study should be seen as a proof-of-concept for the specific recipe. Although the used dataset contains approximately 10 million instances, only a small fraction of those (100,000) were used in this phase.

Many interesting questions remain to be addressed, especially those concerning quantity-based metrics: What is the performance on real-life commodity hardware with realistic network latencies and downtimes? How well can such an ecosystem scale? What is the achieved accuracy when using more data? What is the maturity level of new concepts like IOTA Tangle? What are the specifics of the performance/resilience tradeoff? All these questions will be the subject of future work.

**Acknowledgements.** This research is funded by the University of Macedonia Research Committee as part of the “Principal Research 2019” funding program.

## References

1. Abadi, M., et al.: TensorFlow: a system for large-scale machine learning. In: Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, pp. 265–283 (2016)
2. Dean, J., et al.: Large scale distributed deep networks. In: Advances in Neural Information Processing Systems, pp. 1223–1231 (2012)
3. Zhang, S., Choromanska, A., LeCun, Y.: Deep learning with elastic averaging SGD. In: Advances in Neural Information Processing Systems, pp. 685–693 (2015)
4. Chen, T., et al.: MXNet: a flexible and efficient machine learning library for heterogeneous distributed systems. In: Proceedings of LearningSys (2015)
5. Iandola, F.N., Ashraf, K., Moskewicz, M.W., Keutzer, K.: FireCaffe: near-linear acceleration of deep neural network training on compute clusters (2015)
6. Langer, M., Hall, A., He, Z., Rahayu, W.: MPC SGD—a method for distributed training of deep learning models on spark. IEEE Trans. Parallel Distrib. Syst. (2018)
7. Zaharia, M., et al.: Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing. In: Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation, pp. 15–28 (2012)
8. Benet, J.: IPFS - Content Addressed, Versioned, P2P File System
9. Mashtizadeh, A.J., Bittau, A., Huang, Y.F., Mazieres, D.: Replication, history, and grafting in the Ori file system. In: Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles, pp. 151–166. ACM (2013)
10. Cohen, B.: Incentives build robustness in BitTorrent. In: Workshop on Economics of Peer-to-Peer Systems, vol. 6, pp. 68–72 (2003)
11. Baumgart, I., Mies, S.: S/Kademlia: a practicable approach towards secure key based routing. In: Parallel and Distributed Systems International Conference (2007)
12. Freedman, M.J., Freudenthal, E., Mazieres, D.: Democratizing content publication with coral. In: NSDI, vol. 4, p. 18 (2004)
13. Wang, L., Kangasharju, J.: Measuring large-scale distributed systems: case of BitTorrent mainline DHT. In: 2013 IEEE Thirteenth International Conference on Peer-to-Peer Computing (P2P), pp. 1–10. IEEE (2013)
14. Levin, D., LaCurts, K., Spring, N., Bhattacharjee, B.: BitTorrent is an auction: analyzing and improving BitTorrent’s incentives. In: ACM SIGCOMM Computer Communication Review, vol. 38, pp. 243–254. ACM (2008)
15. Dean, J., Ghemawat, S.: LevelDB—a fast and lightweight key/value database library by Google (2011)
16. IOTA Foundation. <https://www.iota.org>. Accessed 14 Feb 2019
17. Popov, S.: The Tangle, 30 April 2018
18. Popov, S., Saa, O., Finardi, P.: Equilibria in the Tangle, 3 March 2018
19. Coates, A., Huval, B., Wang, T., Wu, D., Catanzaro, B., Andrew, N.: Deep learning with COTS HPC systems. In: Proceedings of the 30th International Conference on Machine Learning, pp. 1337–1345 (2013)
20. Zhang, X., Trmal, J., Povey, D., Khudanpur, S.: Improving deep neural network acoustic models using generalized maxout networks. In: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE (2014)
21. Miao, Y., Zhang, H., Metze, F.: Distributed learning of multilingual DNN feature extractors using GPUs (2014)

22. Povey, D., Zhang, X., Khudanpur, S.: Parallel training of deep neural networks with natural gradient and parameter averaging (2014)
23. Seide, F., Fu, H., Droppo, J., Li, G., Yu, D.: 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech DNNs. In: Fifteenth Annual Conference of the International Speech Communication Association (2014)
24. HEPMASS Dataset. <http://archive.ics.uci.edu/ml/datasets/hepmass>. Accessed 14 Feb 2019



# Predicting Customer Churn Using Artificial Neural Network

Sanjay Kumar<sup>(✉)</sup> and Manish Kumar<sup>(✉)</sup>

Department of Computer Science and Engineering,  
Delhi Technological University, New Delhi, India

sanjay.kumar@dtu.ac.in, sanjaykr6090@gmail.com,  
manishkumar.xciv@gmail.com

**Abstract.** Switching of customers from one service provider to another service provider is known as customer churn. With the surge of the technologies and increased customer awareness, retaining customers has become vital for a company's growth. Several studies have been carried out to keep a check on the customer churn of companies and analyze churn prediction but the accuracy rate is not up to the mark. Recently with the extensive research in the field of Artificial Intelligence it has become possible to dig to the core of the factor responsible for customer churn. We present an effective solution to this challenging problem of customer churn prediction using the data set of telecommunication industry and Artificial Neural Networks to determine the factors influencing the customer churn and optimize the solutions by experimenting with different activation functions.

**Keywords:** Activation function · Artificial Neural Networks · Churn prediction · Deep learning · Machine learning

## 1 Introduction

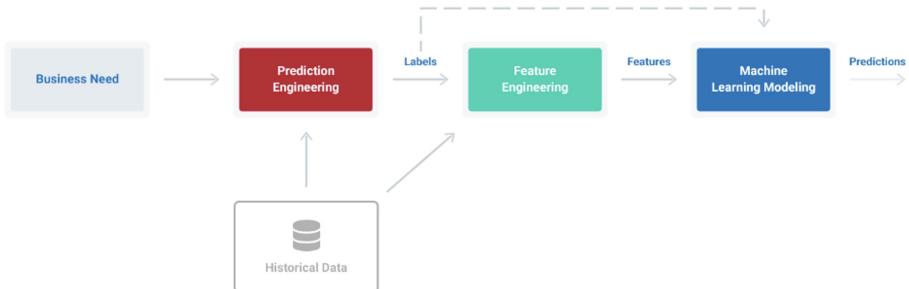
Customer switching from one service provider to other service provider particularly in the subscription based services is called customer churn [5]. It is one of the most influential factors for different service-based industries mainly in telecommunication industries and several other companies which provides subscription-based services such as online entertainment like Netflix, Amazon Prime etc. Customer churn analysis is one of the major factors in the overall growth of a company. Leaving customer churn unchecked can heavily degrade a company's business as customers might start preferring services of other companies and leave the current one.

Recent studies state that expense of making new customer is more than retaining the existing customer as retaining. Jahromi et al. [12] highlighted that the existing customer often leads to major increase in sales and dropped marketing cost [5, 6]. Over the years, several factors have been proposed to be the crucial factors in determining the reason behind the customer churn and also to find out the probability of a customer churn [5, 6]. Having seen the importance of this alarming issue for the service providers for their existence in the competitive marketplace has led to the intense deep dive in

development of the predictive tools and methods supporting vital task in modelling and classification process [8, 9, 13].

In recent years there has been huge increase in the amount of data that is collected and processed to extract meaningful and valuable information across wide areas of business. The meaningful information that is obtained is then used by the companies for customer relationship management (CRM) [1, 14]. Although there are many techniques that have been successfully applied in predicting customer churn like using SVM [1, 13, 15], induction [2], logistic regression, decision Trees [7], Naive Bayes and neural networks [4] in the domain of airlines, banking and many more sectors, deep learning approaches for the scenario still have lots to be explored. Older techniques [2, 10, 11] like logistic regression are less accurate as illustrated by Vafeiadis et al. [3] than the newer evolved techniques like deep learning. Therefore, this study puts light on findings and results of data accumulated from telecommunication [16] by using deep learning techniques with Keras library.

In this paper a Multi-Layered Perceptron is modelled using Keras package in R language to predict customer churn and the factors that dominates such situations. The overall process is described in Fig. 1.



**Fig. 1.** Steps involved in the analysis

## 2 Dataset and Data Preprocessing

We provide a brief overview of the data set used in this paper for analysis and findings. The data set used is from telecommunication industry, consisting of a set of 21 variables and 7043 observations. Data contents are service based information (Internet services, Multiple lines, Phone services, Online backup, Online security, Device protection) customer account information (Customer ID, Paperless bill, Payment method, Monthly charge, Total charge) and demographic information variables (Gender, partners, dependents, senior citizen). For each record we also have its corresponding churn as true or false.

Firstly, we split the dataset in the ratio of 4:1 where the former is used for training and latter is for testing. Irrelevant and incorrect records are removed from the training set by dropping the undesired variables (candidate ID is dropped in the following

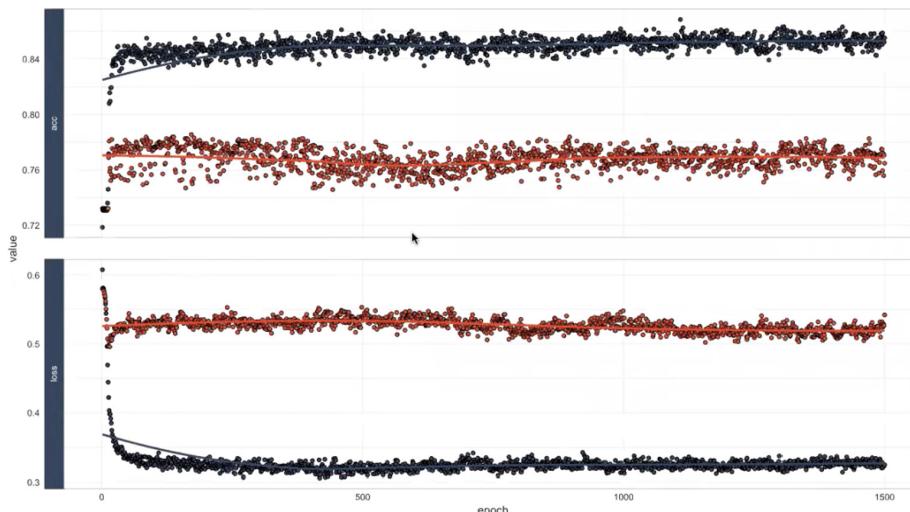
experiment of this paper) and we are finally left with 20 variables and 7034 observations. Then the data is pruned and transformed by one hot encoding.

### 3 Implementation with Results

Proceeding, we build an Artificial Neural Network consisting of a multi-layered perceptron with 3 dense layers each having 16 with initializer function that is uniform and RELU activation. For regularization, a dropout layer is used in the input layer, to prevent overfitting. We set Adam as our optimizer and loss function as binary cross-entropy and train the keras model for 35 and 150 on varying batch sizes. Initially a training accuracy of 84.6% was obtained. Changing the units of neurons to 35, setting kernel initializer to ‘uniform’ and activation function to sigmoid, we achieved 76.4% validation and 85.5% training accuracy. The training results are depicted in Table 1 and the learning curve is shown in Fig. 2.

**Table 1.** Results obtained in various iterations of the model

Batch size	Hidden units	Activation	Epochs	Validation accuracy	Validation loss	Train accuracy
100	16	RELU	150	0.7879	0.4516	0.8332
60	16	RELU	150	0.7850	0.4768	0.8469
50	16	RELU	35	<b>0.8027</b>	0.4264	0.8179
50	35	Sigmoid	1500	0.7648	0.5421	0.8553
50	75 and 35 resp.	Sigmoid	1500	0.75	1.015	<b>0.9314</b>



**Fig. 2.** Learning curve and Loss curve over 1500 epochs

In Fig. 2 we illustrate how the accuracy and loss is changing over different. Final accuracy obtained on training data and validation is 85.53% and 76.5% respectively.

Changing the units of neurons in first hidden layer to 75 s hidden layer neurons to 35, setting kernel initializer to ‘uniform’ and activation function to sigmoid, dropout rate to 0.1 and the optimizer as Adam optimizer we get 93.14% accuracy on training data but validation data scores 75% accuracy.

## 4 Evaluation Measures

To evaluate the classifier performance for different schemes with appropriate parameter the measures used are precision, recall, accuracy and F-measure and these are calculated from content of confusion matrix. True positive and false negative cases are denoted as TP and FN and false positive and true negative cases are denoted by FP and TN respectively.

- Precision =  $TP/TP + FP$
- Recall =  $TP/TP + FN$
- Accuracy =  $TP + TN/TP + FP + TN + FN$

Since we are dealing with binary classification output is converted to vector as the prediction of class generates class value that are usually in matrix of 0’s and 1’s. Estimate is made of keras Table 2.

**Table 2.** Estimated values by our model

	Truth	Estimate	Class probability
	<fact>	<fact>	<dbl>
1	Yes	Yes	0.631
2	Yes	Yes	0.599
3	No	No	0.00753
4	No	No	0.00753
5	No	No	0.00756
6	No	No	0.0968
7	No	No	0.0982
8	No	Yes	0.620
9	No	No	0.00753
10	No	No	0.00756

Creating the confusion matrix Table 3 from model results that indicates actual versus predicted classes and the confusion table obtained is depicted in below table.

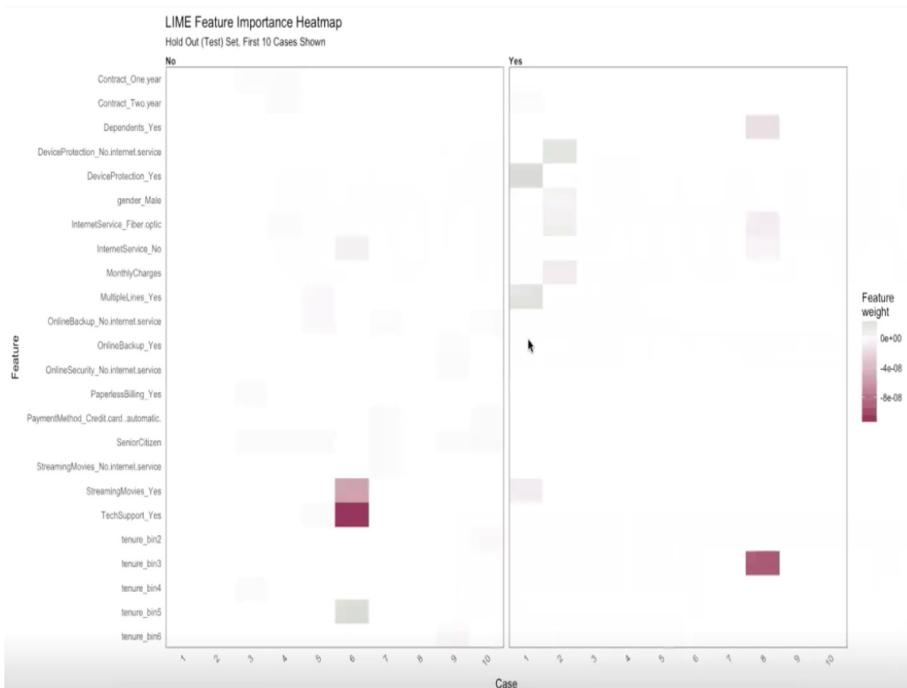
**Table 3.** Confusion table of model

Prediction	Truth	
	No	Yes
No	866	130
Yes	183	227

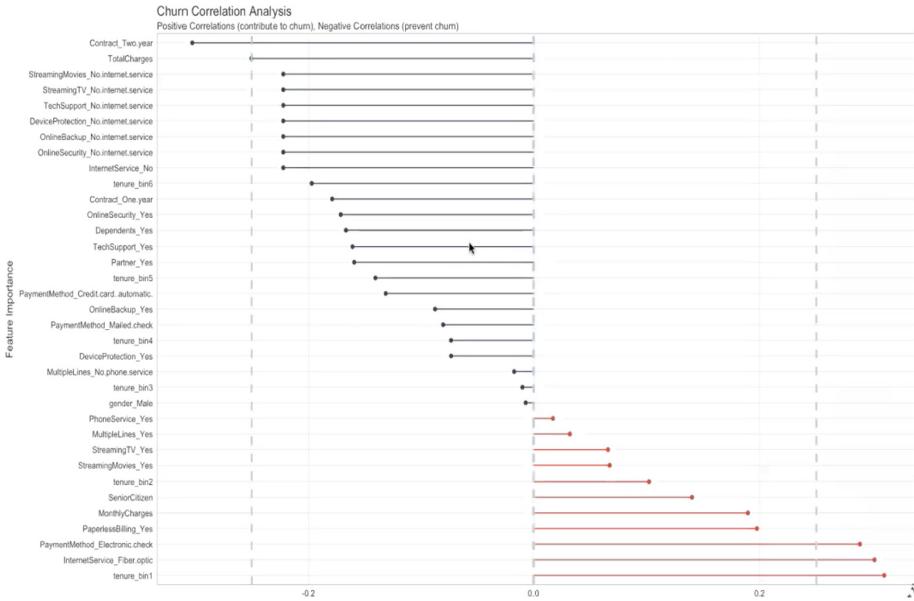
From the confusion matrix precision and recall value is calculated to achieve 0.869 and 0.826 respectively. Higher values indicate better performance by the model while lower values indicate inaccuracy. We can conclude that our model performs well on par with industry standards and requirements.

## 5 Feature Importance Visualisation

To highlight which features are important from the set of features we use a heat map where more important features are highlighted with the darker shade.

**Fig. 3.** Feature importance heatmap

The Heat Map in Fig. 3, demonstrates that the features such as Tech support and Streaming movie has more weighted factors that are responsible for the customer churn. The last step in churn analysis is to study churn correlation analysis result of all the features.



**Fig. 4.** Churn correlation analysis result (Color figure online)

In the above shown Fig. 4, it is clearly illustrated that positive contribution contributes to churn and negative contribution prevents customer churn. In the above figure the features in red increase the likelihood of churn in which the feature “tenure” dominates the churn the most then it is affected by features like “internet service fibre optics” and so on while the features such as “Contract”, “Total charge” that are in blue decreases the likelihood of customer churn.

## 6 Conclusion

We have designed a multi-layer perceptron which can predict customer churn with a high accuracy rate of over 80%. Simulations were performed using the state-of-the-art method of classification for customer churn prediction on the publicly available dataset. Initially evaluation was carried out using different batch size, epoch, number of neurons in the particular layers, initializer, activation function and optimizers and at the end the best possible result was found with good accuracy of prediction and the factor that were responsible for churn are highlighted in the analysis clearly. The work presented in this paper has put some light on the accuracy and performance using different techniques

ranging from changing epoch, batch size, number of neurons in the layers, different activation function and optimizer. Our model can be used in business organisations to decide an optimal balance between cost of targets and customer retention. In the future work some additional techniques and parameters will be explored and also use more detailed and larger data set from industry so that the statistical importance of the result can be maximized and can serve useful to industries for significant growth in the competitive market.

## References

1. Zhao, J., Dang, X.H.: Bank customer churn prediction based on support vector machine: taking a commercial bank's VIP customer churn as the example. In: 2008 4th International Conference on Wireless Communications, Networking and Mobile Computing, pp. 1–4. IEEE (2008)
2. Verbeke, W., Martens, D., Mues, C., Baesens, B.: Building comprehensible customer churn prediction models with advanced rule induction techniques. *Expert Syst. Appl.* **38**(3), 2354–2364 (2011)
3. Vafeiadis, T., Diamantaras, K.I., Sarigiannidis, G., Chatzisavvas, K.C.: A comparison of machine learning techniques for customer churn prediction. *Simul. Model. Pract. Theory* **55**, 1–9 (2015)
4. Sharma, A., Panigrahi, D., Kumar, P.: A neural network based approach for predicting customer churn in cellular network services. arXiv preprint [arXiv:1309.3945](https://arxiv.org/abs/1309.3945) (2013)
5. Mozer, M.C., Wolniewicz, R.H., Grimes, D.B., Johnson, E., Kaushansky, H.: Churn reduction in the wireless industry. In: Advances in Neural Information Processing Systems, pp. 935–941 (2000)
6. Ngai, E.W., Xiu, L., Chau, D.C.: Application of data mining techniques in customer relationship management: a literature review and classification. *Expert Syst. Appl.* **36**(2), 2592–2602 (2009)
7. Lemmens, A., Croux, C.: Bagging and boosting classification trees to predict churn. *J. Mark. Res.* **43**(2), 276–286 (2006)
8. Farquad, M.A.H., Ravi, V., Raju, S.B.: Churn prediction using comprehensible support vector machine: an analytical CRM application. *Appl. Soft Comput.* **19**, 31–40 (2014)
9. Kianmehr, K., Alhajj, R.: Calling communities analysis and identification using machine learning techniques. *Expert Syst. Appl.* **36**(3), 6218–6226 (2009)
10. Neslin, S.A., Gupta, S., Kamakura, W., Lu, J., Mason, C.H.: Defection detection: measuring and understanding the predictive accuracy of customer churn models. *J. Mark. Res.* **43**(2), 204–211 (2006)
11. De Bock, K.W., Van den Poel, D.: Reconciling performance and interpretability in customer churn prediction using ensemble learning based on generalized additive models. *Expert Syst. Appl.* **39**(8), 6816–6826 (2012)
12. Jahromi, A.T., Stakhovych, S., Ewing, M.: Managing B2B customer churn, retention and profitability. *Ind. Mark. Manag.* **43**(7), 1258–1268 (2014)
13. Xia, G.E., Jin, W.D.: Model of customer churn prediction on support vector machine. *Syst. Eng. Theory Pract.* **28**(1), 71–77 (2008)
14. Zhao, Yu., Li, B., Li, X., Liu, W., Ren, S.: Customer churn prediction using improved one-class support vector machine. In: Li, X., Wang, S., Dong, Z.Y. (eds.) ADMA 2005. LNCS (LNAI), vol. 3584, pp. 300–306. Springer, Heidelberg (2005). [https://doi.org/10.1007/11527503\\_36](https://doi.org/10.1007/11527503_36)

15. Chen, Z.Y., Fan, Z.P., Sun, M.: A hierarchical multiple kernel support vector machine for customer churn prediction using longitudinal behavioral data. *Eur. J. Oper. Res.* **223**(2), 461–472 (2012)
16. Chouiekh, A.: Machine learning techniques applied to prepaid subscribers: case study on the telecom industry of Morocco. In: 2017 Intelligent Systems and Computer Vision (ISCV), pp. 1–8. IEEE (2017)



# Virtual Sensor Based on a Deep Learning Approach for Estimating Efficiency in Chillers

Serafín Alonso<sup>1</sup>(✉) , Antonio Morán<sup>1</sup> , Daniel Pérez<sup>1</sup> , Perfecto Reguera<sup>1</sup> , Ignacio Díaz<sup>2</sup> , and Manuel Domínguez<sup>1</sup>

<sup>1</sup> Grupo de investigación en Supervisión, Control y Automatización de Procesos Industriales (SUPPRESS), Esc. de Ing. Industrial e Informática, Universidad de León, Campus de Vegazana s/n, 24007 León, Spain

{saloc,a.moran,dperl,prega,manuel.dominguez}@unileon.es

<sup>2</sup> Electrical Engineering Department, University of Oviedo, Edif. Departamental 2, Campus de Viesques s/n, 33204 Gijón, Spain  
idiaz@uniovi.es  
<http://suppress.unileon.es>

**Abstract.** Intensive use of heating, ventilation and air conditioning (HVAC) systems in buildings entails an analysis and monitoring of their efficiency. Cooling systems are key facilities in large buildings, and particularly critical in hospitals, where chilled water production is needed as an auxiliary resource for a large number of devices. A chiller plant is often composed of several HVAC units running at the same time, being impossible to assess the individual cooling production and efficiency, since a sensor is seldom installed due to the high cost. We propose a *virtual sensor* that provides an estimation of the cooling production, based on a deep learning architecture that features a 2D CNN (*Convolutional Neural Network*) to capture relevant features on two-way matrix arrangements of chiller data involving thermodynamic variables and the refrigeration circuits of the chiller unit. Our approach has been tested on an air-cooled chiller in the chiller plant at a hospital, and compared to other state-of-the-art methods using 10-fold cross-validation. Our results report the lowest errors among the tested methods and include a comparison of the true and estimated cooling production and efficiency for a period of several days.

**Keywords:** HVAC systems · Efficiency · Cooling power · Virtual sensor · Deep learning · Convolutional Neural Network

---

This work was supported in part by the Spanish Ministerio de Ciencia e Innovacion (MICINN) and the European FEDER funds under project CICYT DPI2015-69891-C2-1-R/2-R.

## 1 Introduction

Energy consumption in large buildings, represents more than 20% of the global energy consumption in developed countries. The proliferation of heating, ventilation and air conditioning (HVAC) systems is one of the main reasons behind such a high consumption [16]. In central air conditioning systems, chillers are the main energy consumers, consuming more than 40% of the total energy in commercial and industrial buildings [17]. Thus, their efficiencies have a significant effect on the overall energy performance of these buildings.

Several Energy Efficiency Indicators (EEI) can be used to determine the chiller efficiency [1]. Most of the EEI require measuring the cooling power delivered to chilled water (the chiller output). However, manufacturers do not usually include energy meters in their chiller designs due to high installation, maintenance and recalibration costs [14]. A physical cooling meter can be replaced by a virtual sensor [13].

*Virtual sensors* refer to software, usually including mathematical models that allow measuring process variables or quantities using indirect measurements of related variables, useful in cases where physical sensors are not available, expensive, slow or imprecise. They are a low-cost and non-invasive choice to obtain observations from a real system [15] and have been widely applied for flow and efficiency metering in cooling plants [18, 19]. Data based models for virtual sensors using radial basis functions, multilayer perceptrons and other machine learning methods, can provide accurate estimations for cooling power [3, 11] provided input-output training data are available, which in cooling systems can be acquired with portable energy meters.

Recent deep learning methods have been used for time ahead cooling prediction [4, 6]. However, those methods have been barely applied for estimating the current output of a virtual sensor. Therefore, we suggest that deep learning methods can achieve more accurate models for virtual sensors of cooling power. We propose here a virtual sensor based on a deep learning approach for estimating the cooling production and efficiency in chillers. The proposed model, based on 2D Convolutional Neural Network (2D CNN) [12], is compared to other state-of-the-art methods and tested on a real air-cooled chiller of the plant at the Hospital of León achieving better results.

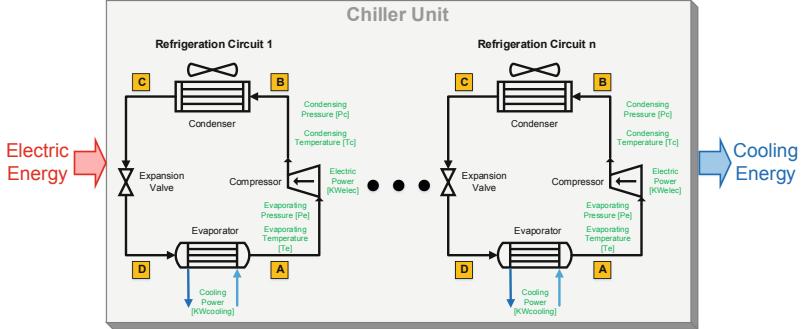
This paper is organized as follows: Sect. 2 states the problem. In Sect. 3, the adopted methodology is exposed. Here, the proposed deep approach is explained in detail. In Sect. 4, the experiment is presented and results are discussed. Finally, conclusions and future work are drawn in Sect. 5.

## 2 Problem Statement

### 2.1 Chiller Efficiency

A chiller is an HVAC system in charge of providing cooling energy to building facilities. Normally, a chiller is formed by a set of refrigeration circuits with similar or even different capacity in order to achieve a better adaptation to

variable cooling loads (see Fig. 1). Each individual circuit provides cooling power according to central chiller control. The total cooling production of the chiller is the sum of the cooling power from each circuit.



**Fig. 1.** A chiller unit with several refrigeration circuits.

Measuring the total cooling production of a chiller is crucial to know the chiller efficiency over time. Typically, chiller manufacturers use the COP (*coefficient of performance*) indicator as an efficiency ratio, providing theoretical values for given surrounding conditions during the manufacturing process. However, chiller efficiency varies over time and depends on external conditions, so it is required to monitor this performance indicator. COP values can be easily computed using the total cooling power and electric power (see Eq. 1).

$$COP = \frac{CoolingPower[KW]}{ElectricPower[KW]} = \frac{KWcooling}{KWelec} = \frac{\sum_{i=1}^n KWcooling_i}{\sum_{i=1}^n KWelec_i} \quad (1)$$

Manufacturers incorporate electric power meters in the chillers since they use the compressor current for controlling cooling capacity. However, they seldom include cooling power meters, because it is not essential for capacity control and increases the cost of the chiller. Moreover, cooling power meters are based on a flow meter, whose measuring principle is usually invasive, thereby becoming a new drawback for their installation. Furthermore, chiller units used to be placed outside so, low temperatures could damage the flow meter. Therefore, virtual sensor implementation becomes crucial in order to measure the cooling power and to compute COP value in a chiller. The virtual sensor should be able to estimate the cooling production and the COP value based on internal variables of the refrigeration circuits.

Applying the energy conservation equation  $Q - W = \Delta H$  to the theoretical refrigeration cycle [10], we have  $Q_{evaporator} = \Delta H = H_D - H_A$ . So, cooling production can be obtained using Eq. 2:

$$KWcooling = Q_{evaporator}\varepsilon = (H_D - H_A)\varepsilon \quad (2)$$

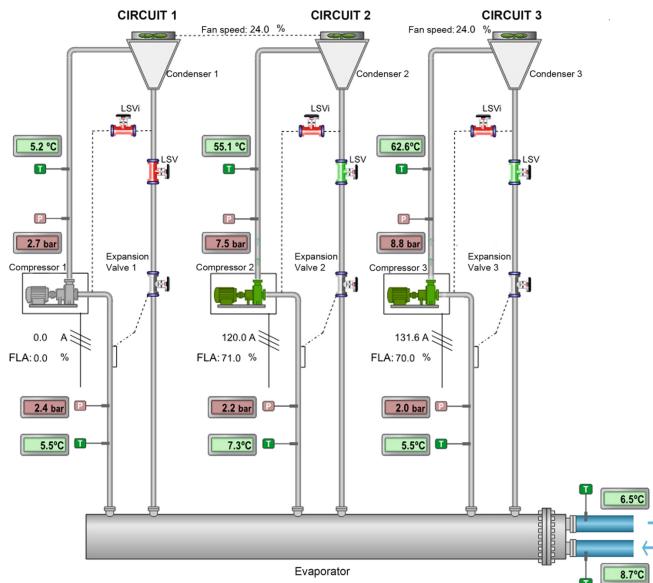
$\varepsilon$  is the efficiency of the heat exchanger. It demonstrates that cooling production depends mainly on enthalpies  $H$  in the evaporator input and output (see Fig. 1). On the other hand, COP value of the theoretical refrigeration cycle can be defined as follows (see Eq. 3):

$$COP = \frac{Q_{evaporator}}{W_{compressor}} = \frac{H_D - H_A}{H_A - H_B} \quad (3)$$

It proves that chiller performance (COP) depends mainly on enthalpies  $H$ . The enthalpy of an ideal gas provides information about internal energy variations and depends mainly on kinetic and vibration energies of molecules, i.e., on temperature and pressure. On the other hand, the enthalpy is a magnitude which characterizes the state of a system in equilibrium, but it does not consider how the system reaches that state. Thus, it can be stated that cooling production and COP depend on enthalpies (given the type of refrigeration gas), i.e. gas pressure and temperature in suction and discharge lines and compressor work. Our hypothesis is that these variables can define the state of the system.

## 2.2 Air-Cooled Chiller at the Hospital of León

The chiller plant at the Hospital of León has been used in the experimental setup. Basically, that plant consists of 5 air-cooled and 2 water-cooled chillers. An air-cooled chiller is used for the experiments.



**Fig. 2.** The air-cooled chiller with 3 refrigeration circuits.

**Table 1.** Main internal variables for each refrigeration circuit of the chiller.

Symbol	Name	Unit
Te	Evaporating temperature	°C
Pe	Evaporating pressure	KPa
Tc	Condensing temperature	°C
Pc	Condensing pressure	KPa
KWelec	Compressor electric power	KW
KWcooling	Cooling power	KW

The air-cooled chiller (model Petra APSa 400-3) has a maximum cooling capacity of 400 tons (approximately 1407 kW) and includes 3 refrigeration circuits (see Fig. 2). Each one is composed of a screw compressor, an electronic expansion valve (EEV) and a condenser in V form. A common evaporator is used for the 3 circuits. The compressor, driven by a three-phase induction motor (400 V; 109 kW), has a maximum displacement of 791 m<sup>3</sup>/h of R134a refrigeration gas. Its capacity can be regulated between 50–100% of maximum value by means of two auxiliary load and unload valves. The condensers have 16 fans of 1.5 kW, driven by variable speed drives. Note that the condensing control signal is common to the 3 circuits.

The control board regulates the operation of 3 refrigeration circuits. It communicates with a BMS (*Building Management System*) which collects and stores data from main internal variables (listed in Table 1) using Modbus protocol.

It should be remarked that it is impossible to measure the individual cooling production of each refrigeration circuit, since the evaporator is common to all the circuits and only the total cooling production is accessible for measuring. Moreover, the condensing control signal is common for the 3 refrigeration circuits, so overpressures in one circuit can affect to the other two circuits (assuming all of them are running). Thus, some interactions among the refrigeration circuits are expected in this chiller. On the other hand, dependencies among variables are expected since the refrigeration cycle is closed, i.e., suction and compressor variables will determine the evolution of discharge variables.

### 3 Methodology

Based on the considerations exposed in Sect. 2, the proposed approach should take into account the following aspects in order to address the regression problem:

- A chiller unit can comprise several refrigeration circuits, which provide cooling energy.
- Cooling energy and efficiency depend on the state of the refrigeration circuits, which is defined by internal variables (pressures, temperatures and compressor work).

- Interactions among refrigeration circuits could appear, depending on the chiller manufacturing structure.
- Dependencies among variables are expected since the refrigeration cycle is closed.

The cooling production of a refrigeration circuit can be defined as a function  $f$  of its main internal variables (see Eq. 4).

$$KWcooling_i = f(Te_i, Pe_i, Tc_i, Pc_i, KWelec_i) \quad \forall i \in \{1, \dots, n\} \quad (4)$$

According to this, a virtual sensor for the overall chiller production ( $KWcooling$ ) can be designed using the next regressor

$$\begin{aligned} Te_1, Pe_1, Tc_1, Pc_1, KWelec_1, Te_2, Pe_2, Tc_2, Pc_2, KWelec_2, \dots \\ \dots, \dots, \dots, \dots, \dots, \dots, \dots, Te_n, Pe_n, Tc_n, Pc_n, KWelec_n \end{aligned} \quad (5)$$

and obtaining a model function  $f$  that relates it to  $KWcooling$ . To model  $f$ , we propose treating the instances of regressor (5) as *images* of size (circuits  $\times$  variables), which define the state of the chiller. Then, we propose to use a 2D convolutional layer to allow us to extract relationships among refrigeration circuits and variables. As proved in Sect. 2, the chiller state can be characterized by internal variables of each refrigeration circuit. Let us suppose several “photos” of the chiller are taken over time. These images contain information about different chiller states that can be used as input data. This suggests the use of an image processing method which allows to detect patterns in 2D images. Our approach is based on a 2D Convolutional Neural Network (2D CNN) which takes advantage of coherence among refrigeration circuits and variables of the chiller.

The adopted methodology is depicted in Fig. 3. First, real data are collected from chiller. Data from internal variables (see Table 1) and from cooling power are required. These data are merged and preprocessed. Next, the deep approach is trained and validated in order to choose the best model function  $f$  for the virtual sensor and to delimit its error range. Finally, the model is deployed as a virtual sensor, enabling monitoring and providing the estimation of cooling power and COP computation. Note that COP is computed from the *estimated* cooling power and the *measured* electric power.

### 3.1 Deep Learning Model Architecture

Our approach is based on 2D Convolutional Neural Network (2D CNN). It consists of several layers: an input layer whose dimension is (circuits  $\times$  variables), a 2D CNN layer to detect relationships among circuits and variables, a fully connected layer and an output layer with dimension 1. The choice of the filter kernel is crucial since it should consider all possible pair-wise combinations among circuits and also among variables, so it should be (2, 2). If the number of circuits is too high, data augmentation can be required to obtain new *images* by shuffling circuits. Note that, the number of pair-wise relationships among circuits is given by  $\frac{c^2 - c}{2}$ , assuming a filter kernel of (2, 2).

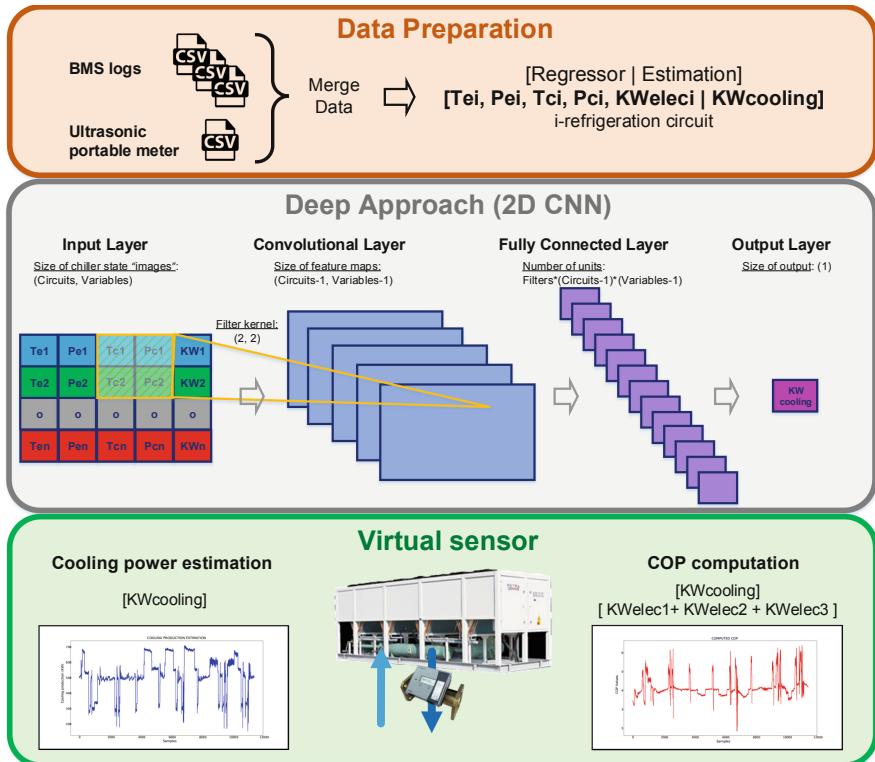


Fig. 3. Methodology based on a deep learning approach.

A downsampling layer is not required since the size of the images is expected to be small. A nonlinear activation function is used for all units. Then, a fully connected layer is applied with resulting units. Finally, an output layer with dimension 1 provides cooling estimation as a virtual sensor measurement.

The proposed deep approach is compared with other linear and non-linear methods used widely in the literature: (a) two **linear methods**, including *Multiple Linear Regression* (MLR) [8] and *Random Sample Consensus* (RANSAC), based on selecting uniformly at random a subset of data samples to estimate model parameters [5]; and (b) several **nonlinear methods**, including a kernel method *Support Vector Regression* (SVR) [2], a *shallow Multilayer Perceptron* (Shallow MLP), with just one hidden layer [9] and trained using backpropagation algorithm, a *deep Multilayer Perceptron* (Deep MLP) with many hidden layers using a special initialization strategy to avoid the vanishing/exploding gradient problem [7].

## 4 Results

### 4.1 Collecting Data

Data have been collected from two sources. First, we have gathered data from BMS (*Building Management System*) logs (plant manager subsystem). BMS stores these data when changing in order to optimize storage capacity. The second data source is an ultrasonic portable meter (Fluxus F601 by Flexim). It supplies the lack of cooling power meter in the chiller. Cooling power is acquired and stored each minute from flow and leaving and return chilled water temperatures.

Both data sources (CSV format) were preprocessed. For that, data from BMS logs were resampled with 1 min and then synchronized and merged with data from ultrasonic portable meter.

### 4.2 Experiments

An experiment has been performed to test our approach. Data from air-cooled chiller no. 5 were selected from 2 months (2018 December and 2019 January), with a sampling time of 1 min, so the number of samples was 38169. Note that, that chiller was not running consecutive time since its operation was alternated with other chillers in the plant. Five regressor variables are used,  $T_e$ ,  $P_e$ ,  $T_c$ ,  $P_c$ ,  $KWelec$  (see Table 1) for each of the 3 refrigeration circuits, resulting in 15 regressor variables. The estimated variable was the cooling power  $KWcooling$ . Total data were split into 2 datasets: The **training and test model dataset** (70% of total data) is used to train and test all models. The proposed approach, 2D CNN, and the remaining linear and non-linear models are trained using this dataset. A 10-fold cross validation has been applied to test models and select the best one. The **virtual sensing test dataset** (30% of total data) is used to test virtual sensor estimation of cooling power and COP. In this case, we suppose that the F601 portable meter is disconnected and the virtual sensor based on the proposed 2D CNN model is used to measure cooling production and efficiency.

The hyperparameters for each model were tuned after several preliminary experiments, choosing the best ones in each scenario.

The proposed 2D CNN model consists of a 4 layers, an input layer (3, 5, 1), a 2D CNN layer (2, 4, 32), a flatten layer (256) and a dense output layer (1). A dropout regularization (0.001) was applied to avoid overfitting. Activation relu function was selected. The number of filters was 32 with a filter kernel of (2, 2) in order to detect pair-wise patterns among circuits and variables. The padding was defined as valid, achieving feature maps with a lower dimension. No downsampling is required due to small size of input images.

The SVR model uses a radial basics function as kernel with 0.01 gamma coefficient. The penalty parameter C of the error was established in 0.001 and the epsilon-tube within which no penalty is associated in the training loss function was 0.01.

The Shallow MLP model consists of 3 layers, a input layer with a dimension of 15, a hidden layer (64 units) and an output layer (1 unit). It is trained with backpropagation algorithm.

The Deep MLP model consists of 12 layers, a input layer with a dimension of 15, 10 hidden identical layers (64 units) and an output layer (1 unit). The dropout regularization was 0.001 to avoid overfitting and relu was selected as activation function (also for Shallow MLP). For all methods, the training epochs were 1000.

### 4.3 Model Validation

A 10-fold cross validation has been carried out to test and validate our approach. Using training and test dataset (70% of total samples), 2D CNN and the remaining methods have been trained and tested. MAE (Mean Absolute Error), MAPE (Mean Absolute Percentage Error) errors have been selected as evaluation metrics since that values are easily understood by any engineer. Average errors and standard deviations of each 10-fold iteration have been computed. Two additional evaluation metrics are also included in order to compare the performance of our approach (and that of the other methods) against the linear (MLR) approximation, which provides a closed-form and fast solution and so, it is taken as a baseline reference:

$$MAE_R = 1 - \frac{MAE_{(method_m)}}{MAE_{(MLR)}}, \quad MAPE_R = 1 - \frac{MAPE_{(method_m)}}{MAPE_{(MLR)}}$$

They allow us to check how much the scores of a certain method improve (+ errors) or worsen (- errors) with respect to MLR scores (reference method).

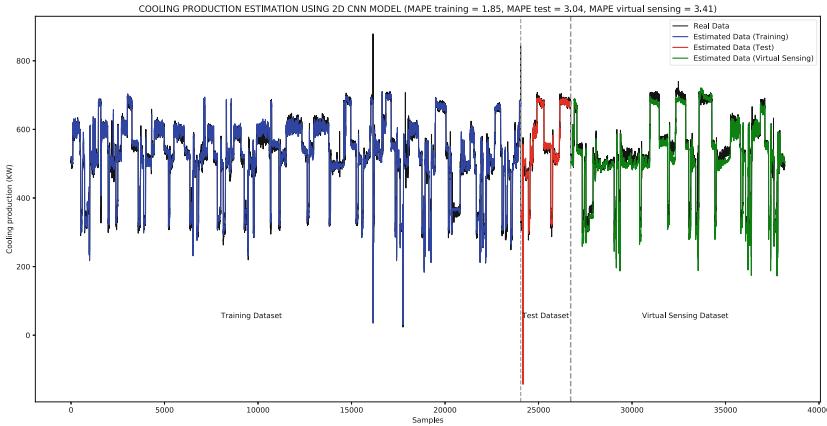
According to train errors (see Table 2), Deep MLP is the best method, improving around 67% a MLR and nearly 13% the second one (Shallow MLP). Our approach is the third method providing also very low errors (MAPE is 1.85% and MAE is 8.96 KW).

According to test errors (see Table 2) and focusing on MAPE errors, our approach is the best method, improving around 29% a MLR and nearly 13% the second one (SVR). The difference with the following methods (Deep MLP and

**Table 2.** Training and test errors.

Method	Training dataset				Test dataset			
	MAE (mean $\pm$ std)	$MAE_R$	MAPE (mean $\pm$ std)	$MAPE_R$	MAE (mean $\pm$ std)	$MAE_R$	MAPE (mean $\pm$ std)	$MAPE_R$
MLR	$16.64 \pm 0.65$	0	$3.55 \pm 0.15$	0	$19.80 \pm 5.94$	0	$4.27 \pm 1.62$	0
RANSAC	$18.25 \pm 2.70$	-0.10	$4.31 \pm 0.89$	-0.21	$20.43 \pm 5.80$	-0.03	$4.76 \pm 1.75$	-0.11
SVR	$10.23 \pm 0.34$	0.38	$2.14 \pm 0.09$	0.40	$15.82 \pm 6.89$	0.20	$3.58 \pm 1.88$	0.16
Shallow MLP	$7.94 \pm 0.31$	0.52	$1.65 \pm 0.08$	0.54	$15.95 \pm 6.58$	0.19	$3.79 \pm 2.19$	0.11
Deep MLP	<b><math>5.53 \pm 0.40</math></b>	<b>0.68</b>	<b><math>1.13 \pm 0.09</math></b>	<b>0.67</b>	$16.66 \pm 6.85$	0.16	$3.64 \pm 1.58$	0.15
2D CNN	$8.96 \pm 0.46$	0.46	$1.85 \pm 0.09$	0.48	<b><math>13.24 \pm 3.71</math></b>	<b>0.33</b>	<b><math>3.04 \pm 1.23</math></b>	<b>0.29</b>

Shallow MLP) is 14% and 18%, respectively. Checking MAE errors, our approach is also the winner. It can be stated we could estimate cooling production and efficiency either with a relative error of 3.04% or with an absolute error of 13.24 KW (see Fig. 4). Considering the standard deviation ( $\pm 1.23$ ), relative errors can range between 1.81% and 4.27% (in the best and the worst scenario). Therefore, we can conclude that 2D CNN is the best method to build the virtual sensor.



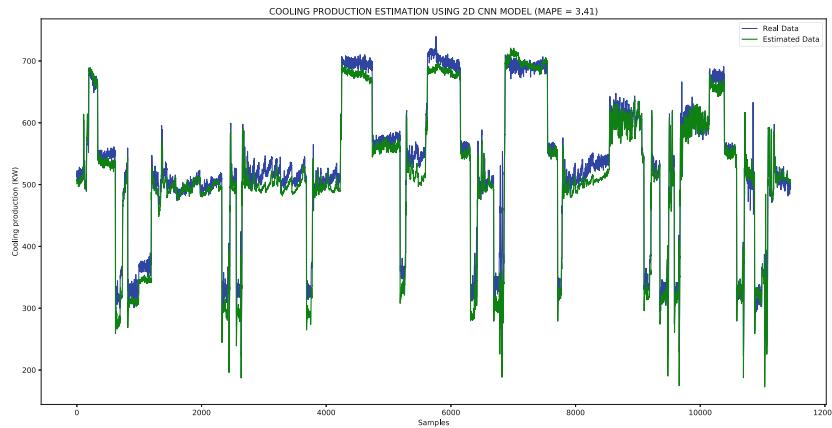
**Fig. 4.** Cooling production estimation using the proposed deep approach (2D CNN) for training, test and virtual sensing datasets.

#### 4.4 Virtual Sensor Test and COP Computation

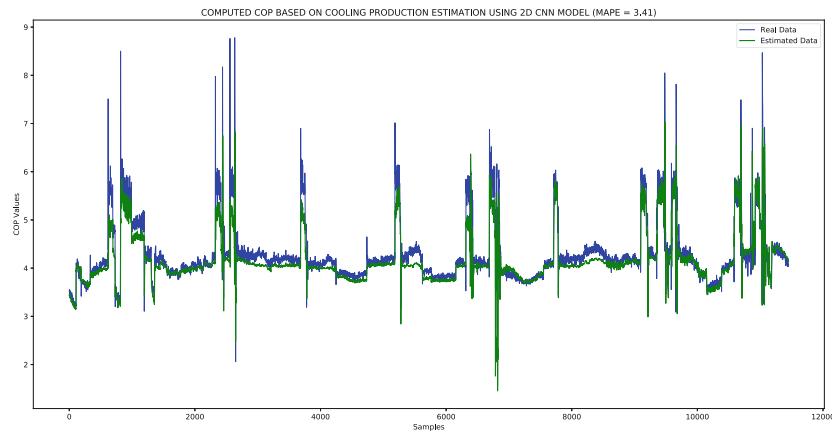
Once the 2D CNN model was validated, we performed a new test using virtual sensing dataset (30% of the total data). Now, we suppose the portable F601 meter is disconnected from the chiller and we try to verify the virtual sensor measurement, estimating the cooling production and efficiency. In this case, we have chosen MAPE error to test estimations.

The results can be observed in Fig. 5. The virtual sensor provides measurements of cooling power with an error of 3.41% and with an absolute error of 16.43 KW. It is very accurate, except when starting or stopping a compressor.

The final aim of virtual sensor is to monitor chiller efficiency. For that, COP value is computed using estimated cooling power and measured electric power (sum of 3 electric compressor powers) for all virtual sensing dataset. The result can be seen in Fig. 6. The chiller efficiency can be estimated with a relative error of 3.41% and with an absolute error of 0.16. Note that the maximum errors occur when a compressor starts or stops.



**Fig. 5.** Cooling production estimation using the proposed deep approach (2D CNN) for virtual sensing dataset.



**Fig. 6.** Computed COP values using cooling production estimation.

## 5 Conclusions

In this paper, we have proposed a virtual sensor for cooling power estimation based on available internal chiller variables (temperatures, pressures and compressor power) and using a deep convolutional neural network (2D CNN). The proposed architecture uses a convolutional layer that takes advantage of coherence between the three refrigeration circuits of the chiller, and was systematically compared to a set of state-of-the-art methods, using several performance metrics with a 10-fold validation methodology, where our proposed method achieved the best results.

On the application side, the developed virtual sensor is very valuable for several reasons. First, the estimations of the virtual sensor can replace the current

measurements from the expensive portable measuring system used for training, that can be only available provisionally. This methodology can be extensible (“copy-pasted”) to any chiller, specially to the 5 identical chillers in this plant, resulting in a highly cost-effective way to track and monitor the overall cooling power of the plant. Second, the availability of electric power consumption and an accurate enough estimation of cooling power allows to have also an estimation of the chiller efficiency, highly valuable for energy optimization of the overall plant.

## References

1. Alves, O., Monteiro, E., Brito, P., Romano, P.: Measurement and classification of energy efficiency in HVAC systems. *Energy Build.* **130**, 408–419 (2016). <https://doi.org/10.1016/j.enbuild.2016.08.070>
2. Cortes, C., Vapnik, V.: Support-vector networks. *Mach. Learn.* **20**(3), 273–297 (1995). [www.scopus.com](http://www.scopus.com), cited By (since 1996): 4606
3. Escobedo-Trujillo, B., Colorado, D., Rivera, W., Alaffita-Hernández, F.: Neural network and polynomial model to improve the coefficient of performance prediction for solar intermittent refrigeration system. *Sol. Energy* **129**, 28–37 (2016). <https://doi.org/10.1016/j.solener.2016.01.041>
4. Fan, C., Xiao, F., Zhao, Y.: A short-term building cooling load prediction method using deep learning algorithms. *Appl. Energy* **195**, 222–233 (2017). <https://doi.org/10.1016/j.apenergy.2017.03.064>
5. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* **24**(6), 381–395 (1981)
6. Fu, G.: Deep belief network based ensemble approach for cooling load forecasting of air-conditioning system. *Energy* **148**, 269–282 (2018). <https://doi.org/10.1016/j.energy.2018.01.180>
7. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016). <http://www.deeplearningbook.org>
8. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning, 2nd edn. Springer, New York (2009). <https://doi.org/10.1007/978-0-387-84858-7>
9. Haykin, S.S.: Neural Networks and Learning Machines, 3rd edn. Pearson Education, London (2009)
10. Klein, S.A.: Design considerations for refrigeration cycles. In: International Refrigeration and Air Conditioning Conference, vol. 190, pp. 511–519. Purdue e-Pubs (1992). <http://docs.lib.psu.edu/iracc/190>
11. Kusiak, A., Li, M., Zheng, H.: Virtual models of indoor-air-quality sensors. *Appl. Energy* **87**(6), 2087–2094 (2010). <https://doi.org/10.1016/j.apenergy.2009.12.008>
12. LeCun, Y., Bengio, Y.: Convolutional networks for images, speech, and time series. In: The Handbook of Brain Theory and Neural Networks, pp. 255–258. MIT Press, Cambridge, MA, USA (1998)
13. Li, H., Yu, D., Braun, J.E.: A review of virtual sensing technology and application in building systems. *HVAC&R Res.* **17**(5), 619–645 (2011). <https://doi.org/10.1080/10789669.2011.573051>
14. McDonald, E., Zmeureanu, R.: Virtual flow meter to estimate the water flow rates in chillers. *ASHRAE Trans.* **120**, 200–208 (2014)

15. McDonald, E., Zmeureanu, R.: Development and testing of a virtual flow meter tool to monitor the performance of cooling plants. *Energy Procedia* **78**, 1129–1134 (2015). <https://doi.org/10.1016/j.egypro.2015.11.071>
16. Pérez-Lombard, L., Ortiz, J., Pout, C.: A review on buildings energy consumption information. *Energy Build.* **40**(3), 394–398 (2008). <https://doi.org/10.1016/j.enbuild.2007.03.007>
17. Saidur, R., Hasanuzzaman, M., Mahlia, T., Rahim, N., Mohammed, H.: Chillers energy consumption, energy savings and emission analysis in an institutional buildings. *Energy* **36**(8), 5233–5238 (2011). <https://doi.org/10.1016/j.energy.2011.06.027>. PRES 2010
18. Wang, H.: Water flow rate models based on the pipe resistance and pressure difference in multiple parallel chiller systems. *Energy Build.* **75**, 181–188 (2014). <https://doi.org/10.1016/j.enbuild.2014.02.017>
19. Zhao, X., Yang, M., Li, H.: Development, evaluation and validation of a robust virtual sensing method for determining water flow rate in chillers. *HVAC&R Res.* **18**(5), 874–889 (2012). <https://doi.org/10.1080/10789669.2012.667036>

# **Deep Learning - Convolutional ANN**



# Canonical Correlation Analysis Framework for the Reduction of Test Time in Industrial Manufacturing Quality Tests

Paul Alexandru Bucur<sup>(✉)</sup> and Philipp Hungerländer

Department of Mathematics, Alpen-Adria-Universität Klagenfurt,  
Universitätsstraße 65-67, 9020 Klagenfurt, Austria  
[pabucur@edu.aau.at](mailto:pabucur@edu.aau.at), [phhunger@aau.at](mailto:phhunger@aau.at)

**Abstract.** In industrial manufacturing processes, quality control tests are performed in order to measure product characteristics which help assess and classify the product's quality. In this work, we focus on quality tests during which a signal is recorded for each product. We propose the usage of data-driven methods for a potential reduction of the test duration, without inducing loss in the quality classification performance. While in industrial practice most features extracted from the signals are still often hand crafted by domain experts and used as input to shallow classifiers, more advanced classification methods such as Convolutional Neural Networks (CNNs) are able to combine the feature extraction, selection and classification into a single process.

In this paper we first use CNNs to determine whether an excerpt of the recorded signal exists which, starting at time 0, contains already enough information so as to match the classification performance reached with the usage of the complete signal. Second, we apply the Canonical Correlation Analysis (CCA) framework to investigate how the features extracted and selected from multiple, successively increasing excerpts relate to the features the quality test was originally designed to measure. Third, we analyze the presence of noise among the classification-relevant features extracted from the increasing excerpts. The suitability of the proposed framework is validated using a real-world dataset from the automotive industry, showing that the test time of the corresponding vibroacoustical quality test can be reduced by 77.78%, thus ensuring a high practical relevance of the findings.

**Keywords:** Canonical Correlation Analysis · Convolutional Neural Networks · Quality test time reduction

---

Supported by ThyssenKrupp Presta AG by means of experimental data and domain expert feedback.

## 1 Introduction

The industrial manufacturing paradigm shift known as Industry 4.0 and the associated modernization in terms of intelligent asset monitoring and data and sensor fusion are significantly increasing quality control capabilities [1]. As a result, the prediction of abnormal behaviour in products or manufacturing machines, a central pillar of quality control, is profiting from a previously unknown amount of available data. Combined with vast computational resources, this data enables the usage of intelligent, data-driven methods such as machine learning for diverse descriptive, predictive or prescriptive tasks [2, 3]. The modernization induced by the implementation of Industry 4.0 standards has also affected product performance tests, which are conducted in order to measure product characteristics that facilitate quality control.

In a contribution closely related to our work, Fuzzy-Bayesian networks were used to predict the performance of refrigeration compressors [4]. Due to their superior predictive power, the information needed for the performance prediction could be inferred from a short excerpt of the original measurement, thus reducing the necessary test time. Although the used approach is applicable in a broader context than the one resulting from the experimental dataset considered by the authors, Bayesian networks feature a series of disadvantages: on the one hand, learning their structure from data is known to be NP-hard under specific conditions [5], which for large datasets may be computationally prohibitive. On the other hand, the specification of a prior can be challenging in domains which are still heavily relying on expert knowledge, such as engineering. Sequential probability ratio tests [6, 7] represent another technique which can be employed for time reduction in quality tests. Starting with a null and an alternative hypothesis, after each additional observation the decision must be made whether to accept one of them and thus stop the test, or to proceed with further monitoring. In its original form, this technique only offers a stopping rule from which a new necessary, potentially shorter, test time can be derived, without further investigating the behaviour of the information and noise parts in the features extracted and selected from the quality test signals.

In this paper, we propose the combination of Convolutional Neural Networks (CNNs) and the Canonical Correlation Analysis (CCA) technique [8] as an alternative for the potential reduction of the time required for quality tests. The CCA framework has been used in a wide spectrum of scenarios, such as unsupervised learning with multiple available views [9], feature learning from a view when a second view is available during training, but not at test time [10, 11] or multi-view regression [12]. Other recent contributions [13, 14] have also applied CCA in order to increase the interpretability of the representations learned by the hidden layers of neural networks. In [14], the authors use the CCA framework to compute new, maximally correlated representations between the encodings of a hidden layer at a specific time during the training process and the encodings of the same layer at the final training timestep. Comparing the sorted CCA coefficients for different timesteps of the training, they observe that by the timestep the neural network has already reached the final performance, a significant num-

ber of CCA coefficients have not yet converged, concluding that they must belong to noise. In our work, after using a CNN to identify the earliest possible stopping time for the quality test, we apply the same approach as the authors in [14] to investigate the presence of noise in the features extracted from excerpts of the quality test recording.

The main contributions of this paper help tackle multiple engineering challenges and can be summarized as follows:

- (a) Combination of a CNN and the CCA framework for a potential reduction of the time needed for industrial manufacturing quality tests.
- (b) Application of the technique proposed in [14] to investigate the relationship between classification-relevant features learned via a CNN from an excerpt of the quality test recording and the features resulting from the full recording.
- (c) Practical contribution by means of a test time reduction of 77.78% for the vibroacoustical quality test represented by a real-world engineering dataset.

The remainder of this paper is organized as follows: Sect. 2 formalizes the problem, with the methodology proposed for its solution introduced in Sect. 3. In Sect. 4 we describe the engineering dataset used for the confirmation of the suitability of our approach, together with the architecture of the employed CNN and the performed computational experiments. In Sect. 5 we present the results and discuss their interpretability. Finally, Sect. 6 concludes the work presenting potential further research directions.

## 2 Problem Description

Let  $\mathcal{D}$  denote a dataset consisting of a total of  $l$  sensor recordings of common integer length  $T$  and corresponding binary labels  $y$  representing the quality of the tested products. The goal is twofold: first, we analyze whether an excerpt of the full recording exists which, starting at time 0, contains already enough information in order to reach the same classification performance as with the usage of the full recording. Second, we relate the classification-relevant features from the increasing excerpts and the features from the full recording, which the test was originally designed to measure. At the same time, we make two important assumptions: on the one hand, we assume that the dataset  $\mathcal{D}$  contains a sufficient number of samples from all known quality issues. On the other hand, we require the suitability of the hand crafted features defined by domain experts for the full recordings. For cases where this assumption can not be guaranteed, we propose the usage of a CNN for the feature extraction and selection in Sect. 3.2.

## 3 Methodology

The CCA framework can be employed for the identification of mutual information in two observation matrices originating from an underlying process by inferring information from the cross-covariance matrices. In the scenario described

in Sect. 2, the underlying process consists of the recording of measurements by a specific sensor in the context of an industrial quality test.

If domain experts would predefine features for each excerpt length and associate them with the same physical quantities, a direct comparison of single features between excerpts would be possible. However, when employing more complex methods for quality assessment purposes such as CNNs, the difficulty of comparing and interpreting the encodings of their hidden layers is significantly higher, hence the importance of the CCA similarity measure. In this way, even if the nature of the features learned from the excerpts may not possess a direct physical meaning, they can be known to have a high representational similarity in terms of CCA correlation with the physical features the quality test was designed to measure.

### 3.1 Canonical Correlation Analysis

Following [15], the mathematical formulation of the CCA assumes two random vectors,  $(P_1, P_2) \in \mathbb{R}^{a_1} \times \mathbb{R}^{a_2}$ . Denoting by  $(\Sigma_{P_1, P_1}, \Sigma_{P_2, P_2})$  their covariances and by  $\Sigma_{P_1, P_2}$  their cross-covariance, the CCA identifies pairs of maximally correlated linear projections of the two random vectors, using two sets of weights  $(w_1, w_2)$ . The optimal values of both weight sets maximize the correlation coefficient  $\rho$  between  $(w'_1 P_1, w'_2 P_2)$ :

$$(w_1^*, w_2^*) = \arg \max_{w_1, w_2} \frac{w'_1 \Sigma_{P_1, P_2} w_2}{\sqrt{w'_1 \Sigma_{P_1, P_1} w_1 w'_2 \Sigma_{P_2, P_2} w_2}}.$$

Further requiring the linear projections to feature unit variance, we obtain:

$$(w_1^*, w_2^*) = \arg \max_{w'_1 \Sigma_{P_1, P_1} w_1 = w'_2 \Sigma_{P_2, P_2} w_2 = 1} w'_1 \Sigma_{P_1, P_2} w_2.$$

If the optimization problem is extended to the identification of  $k \leq \min(a_1, a_2)$  projections, subsequent projections are required to be uncorrelated with the previous ones. By constructing two matrices  $Z_j \in \mathbb{R}^{a_j \times k}$  whose columns, for  $j \in \{1, 2\}$ , represent the first  $k$  weight vectors  $w_j^i$ , the problem can be formalized as:

$$\begin{aligned} & \text{maximize} && \text{Tr}(Z'_1 \Sigma_{P_1, P_2} Z_2) \\ & \text{subject to} && Z'_1 \Sigma_{P_1, P_1} Z_1 = Z'_2 \Sigma_{P_2, P_2} Z_2 = I. \end{aligned}$$

One of the multiple representations of the solution was offered in [16], who by defining:

$$T := \Sigma_{P_1, P_1}^{-\frac{1}{2}} \Sigma_{P_1, P_2} \Sigma_{P_2, P_2}^{-\frac{1}{2}}, \quad (1)$$

identify the optimal objective value as the sum of the top  $k$  singular values of  $T$ , encountered at  $(Z_1^*, Z_2^*) = (\Sigma_{P_1, P_1}^{-\frac{1}{2}} U_k, \Sigma_{P_2, P_2}^{-\frac{1}{2}} V_k)$ , where  $U_k, V_k$  are the matrices of the first  $k$  left-, respectively right- singular vectors of  $T$ .

### 3.2 Test Time Reduction

In our approach, we use the CCA framework in order to gain insights about the representational similarity of the features deemed suitable for quality classification purposes resulting from an analysis of the full quality test recording, on the one hand, and from an excerpt of the full recording, starting at time 0, on the other hand. To this effect, we introduce a set of ordered indexes  $\mathcal{T} = \{\tau_1, \tau_2, \dots, \tau_{n-1}, \tau_n = \mathbb{T}\}$  which split the time axis of the recording into  $n$  equidistant intervals. We further denote each excerpt of the recording starting at time 0 and ending at time  $\tau_i \in \mathcal{T}$  as  $\psi_i$ , whereas the set of excerpts increasing in length shall be denoted by  $\Psi = \{\psi_1, \dots, \psi_{n-1}, \psi_n\}$ . From this definition, it quickly results that  $\psi_n$  denotes the full recording.

In a first step, for each excerpt  $\psi_i \in \Psi$  we compute a set of features which help explain the quality labels  $y$  of the full recording  $\psi_n$ . Depending on the nature of the engineering scenario for which the quality test was designed, diverse feature extraction and selection mechanisms may be known to domain experts, especially for  $\psi_n$ . In their absence, we propose the usage of a CNN as a tool capable of integrating feature extraction, selection and classification and reaching impressive results in terms of signal classification performance [17–19]. For each  $\psi_i \in \Psi$ , we denote the corresponding feature set by  $\phi_i$  and introduce  $\Phi = \{\phi_1, \dots, \phi_{n-1}, \phi_n\}$ . In Sect. 4.3 we describe the feature extraction and selection scheme employed for the experimental dataset.

In a second step, we perform the CCA between each of the distinct feature sets  $\phi_i$ , where  $i \in [1, n - 1]$ , and  $\phi_n$ , computing multiple CCA components. In a similar manner to [14], we then analyze the convergence of the CCA coefficients as the excerpt length increases. We proceed to identify an index  $\iota \leq n$  (the new length of the quality test) for which the classification performance, under usage of the features  $\phi_\iota$ , already matches the classification performance attainable by using the features  $\phi_n$ .

Observing how many of the CCA coefficients have already converged to their final values by  $\iota$  allows a deeper understanding of the encoding  $\phi_\iota$ . Analogous to [14], we take an interest in whether the early-stabilizing CCA components remain stable for increasing excerpt length and in the development of the unstable parts of the  $\phi_i$  encodings, which may represent noise. To this effect, we use the CCA to maximize the correlation between the  $\phi_i$  features for a low  $i \in \mathcal{T}$  index and for one slightly after  $\frac{n}{2}$ , computing a total of  $\xi$  CCA components. We group the first  $\eta$  components featuring the highest correlation coefficients in a vector  $C$  and the  $\eta$  last components with the lowest correlation coefficients in a vector  $Q$ . Finally, we again compute  $\eta$  CCA coefficients between the distinct  $\phi_i \in \Phi$  and  $C$  and  $Q$ , respectively. Following [14], for two feature sets  $\phi_q, \phi_g \in \Phi$  we then define a distance measure based on the average correlation coefficient:

$$d(\phi_q, \phi_g) := 1 - \frac{1}{\eta} \sum_{i=1}^{\eta} \rho^i(\phi_q, \phi_g), \quad (2)$$

where  $\rho^i$  denotes the correlation coefficient corresponding to the  $i$ -th CCA component between  $\phi_q$  and  $\phi_g$ . The development of the distance values between

the distinct  $\phi_i \in \Phi$  encodings and the vectors  $C$  and  $Q$  offers a very intuitive visualization of the behaviour of the stable and unstable parts of the feature encodings. From an engineering viewpoint, it is important to know that the features selected for classification from each excerpt contain a common part, which can be expected to generalize well to previously unseen data samples.

## 4 Computational Experiments

### 4.1 Experimental Dataset

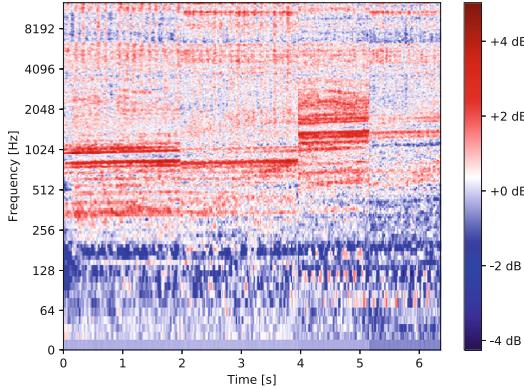
The experimental dataset employed in this work originates from the daily operations of *ThyssenKrupp Presta AG*, a supplier of steering gears. Undesired vibrations in a steering gear and the resulting noise can often be traced back to one of its subcomponents, the ball nut assembly (BNA). Due to this reason, the produced BNA are subject to a vibroacoustical test which consists of steering the product first left, then right, at two different mean rotational speeds:  $300^\circ/s$  and  $500^\circ/s$ . During each test, the rotational velocity is linearly increased from an initial 97.5% of the mean speed to a final value of 102.5%. Two accelerometers, positioned perpendicularly on the product, record the vibrations emanating from the steering movement with a sampling rate of 25.6 kHz. For the quality assessment, the BNA vibrational signals are transformed to obtain their encoding as order spectra, a frequency domain representation.

Acoustic domain experts then define a set of non-overlapping zones for the spectra, each consisting of a left and right border and an upper threshold. Between the zone borders, the order spectra may not violate the upper thresholds; if the threshold of any zone is violated, the corresponding BNA fails the quality test. The BNA which receive a positive quality assessment are assigned a label of 0, while the faulty ones are labeled as 1. The dataset consists of a total of 8424 BNA and is split into a training set consisting of 5616 BNA (172 of which failed the quality test) and a test set featuring 2808 BNA (86 faulty BNA). In our setting, the significance of the production order of the BNA hinders a cross-validation approach: the feature extraction and selection mechanisms shall also account for possible trends or changes over time in the BNA production. We thus construct the validation dataset with a stratified selection of the last 20% of the samples belonging to the training set.

### 4.2 Data Preprocessing

Due to the vibroacoustic nature of the data, in a first preprocessing step we compute for each signal excerpt  $\psi_i \in \Psi$  the Short Time Fourier Transform (STFT) using a FFT window size of 2048, a Hanning window of length 2048 and a hop length of 512. In a second step we average over the spectrograms of both accelerometers, then concatenate these averaged spectrograms resulting from both mean rotational speeds and both turning directions along their time axis. In order to offer an overview of the differences between both quality classes,

we separately average over the spectrograms with label 1 and those with label 0. Figure 1 depicts the result of the subtraction of the mean spectrogram with label 0 from the mean spectrogram with label 1.



**Fig. 1.** Overview of the averaged spectrograms with label 0 (resulting from the full quality test recording  $\psi_n$ ) subtracted from the averaged spectrograms with label 1. Interestingly, frequencies can be identified for which the qualitatively superior BNA feature higher amplitudes than their faulty counterparts. These frequencies are however either irrelevant for the steering gear vibroacoustic behaviour or diverse mechanisms ensure their damping.

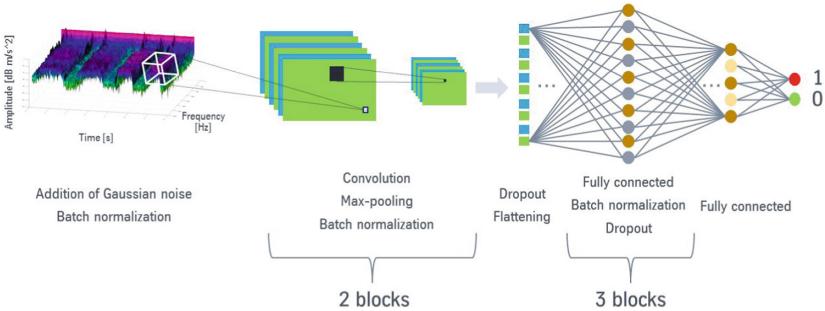
Furthermore, due to the high data imbalance in which Class 0 has a prevalence of 96.94%, we additionally use the Synthetic Minority Oversampling Technique (SMOTE) [20] to create new, synthetic samples with label 1 for the training data. We avoid rebalancing the validation dataset, making sure that the classifier learn and select features belonging to the true data distribution. In a last preprocessing step, we standardize the spectrograms to feature zero mean and unit variance.

### 4.3 Architecture of the Convolutional Neural Network

The encoding of the vibrational signals as spectrograms paves the way for the usage of CNNs as a feature extraction, selection and classification mechanism [21–23], a technique which we also employ. In a first step, the network adds Gaussian noise with a standard deviation of the noise distribution of 0.2 to the inputs in each batch, aiming to ensure regularization and avoid overfitting. After a batch normalization step, two identical convolutional blocks follow. Each block begins with a convolutional layer featuring 50 neurons, a filter spanning 5 units on the frequency- and 3 units on the time axis and ReLU activation. The convolution is followed by a max-pooling operation, where the max pooling window spans 2 units on the frequency- and 2 units on the time axis, and by

batch normalization. After a dropout layer which sets 50% of the input units to 0 at each training update, the data is flattened and run through 3 identical fully connected blocks. Each such block consists of a fully connected layer with 500 neurons and ReLU activation, followed by batch normalization and dropout with a fraction of 70% of the units set to 0 at each training iteration. In a penultimate step another fully connected layer is applied, featuring 200 neurons and ReLU activation. The encoding of this layer is the result of the feature extraction and selection step and thus represents the features  $\phi_i \in \Phi$ , the random vector used as input to the linear CCA as described in Sect. 3.2. In this work, we chose not to consider the features extracted and selected by domain experts as  $\phi_n$ , but use the same approach as for the previous feature sets  $\phi_i \in \Phi$ . Choosing  $n = 18$ , we compute a total of 150 CCA components between each  $\phi_i$ , where  $i \in [1, 17]$ , and the final  $\phi_{18}$ . A final fully connected layer featuring softmax activation maps the encoding of the previous layer to 2 units which represent the two quality classes.

The training process uses the Adam optimizer with a learning rate of 0.0001 and the binary crossentropy as loss, a batch size of 50 and 10 training epochs. Apart from direct architectural choices such as the number of convolutional and fully connected blocks and their layer structure or the choice of the optimizer and the loss, the rest of the described quantities were treated as hyperparameter and optimized using random search. A schematic overview of the employed CNN is offered in Fig. 2.

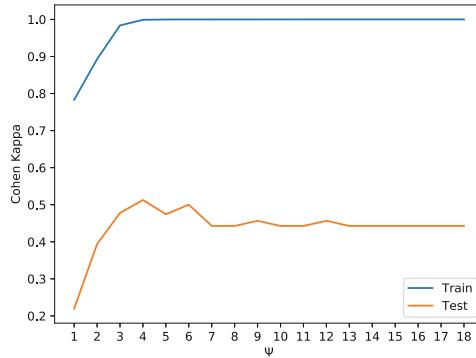


**Fig. 2.** Schematic overview of the employed Convolutional Neural Network architecture, featuring the convolutional- and fully-connected blocks.

## 5 Results and Interpretability

Due to the high data imbalance and the resulting no-information rate, we chose together with acoustic domain experts Cohen's Kappa [24] as a single value metric to assess the classification performance. The classification results of the  $\phi_i \in \Phi$  features corresponding to the different  $\psi_i \in \Psi$  excerpts are depicted for the training and test data in Fig. 3. The number of true positives, false positives,

true negatives and false negatives associated with the Cohen Kappa value of 0.44 obtained for  $\phi_{18}$  on the test data is 2722, 61, 25 and 0, respectively. Despite multiple regularization techniques such as the usage of dropout or early-stopping the training process as soon as the loss on the validation data begins to increase again, we still observe a significant gap between the performance on the training data, on the one hand, and validation and test data, on the other hand. Since the performance on the validation data is almost identical to the test data, we trace the performance gap back to three origins. First, the noisiness of the quality labels is a known problem, since the binary classification is a simplification of the reality, where different quality nuances exist. Second, the industrial production processes change with time due to a plethora of reasons, leading to shifts in the data distribution. Third, it is also possible that the created synthetic data samples do not bear enough resemblance to the validation and test data.

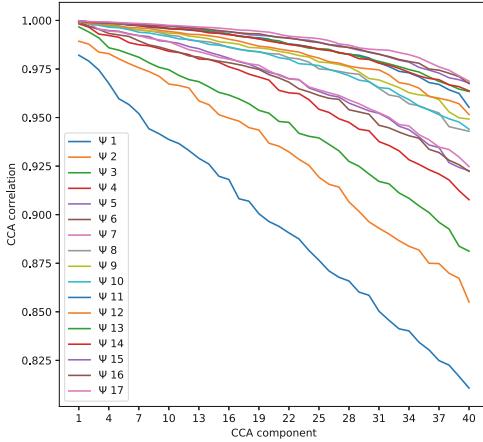


**Fig. 3.** The classification performance of the CNN with respect to the Cohen Kappa metric for the training and test data. By  $\psi_4$ , the CNN achieves the same classification performance as with  $\psi_{18}$ , the full recording.

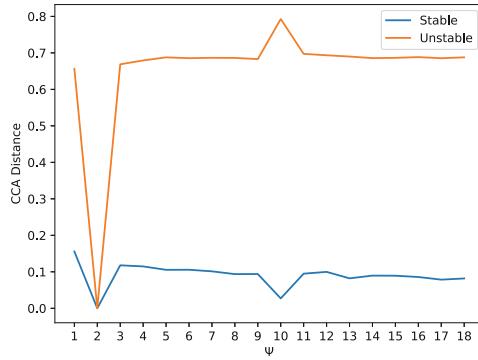
Despite the performance gap, we observe that for  $\psi_4$ , the classification performance of the features  $\phi_4$  extracted and selected by the CNN already matches the final performance obtained with the usage of  $\phi_{18}$ . We thus identify an early stopping time of the test of  $\iota = 4$ , which implies that the test duration could be reduced by 77.78%. In Fig. 4, we observe that by this time, many of the CCA coefficients have not yet converged to their final values, indicating that the corresponding components potentially represent noise in the encodings  $\phi_i \in \Phi$ .

The development of the distance measure introduced in Eq. 2 for the increasing excerpts  $\psi_i \in \Psi$  is depicted in Fig. 5 for  $\xi = 150$  CCA components computed between  $\phi_2$  and  $\phi_{10}$  and a length of  $\eta = 10$  of the  $C$  and  $Q$  vectors.

We observe that, as expected, the CCA components which stabilize early remain stable, with the unstable ones rapidly losing correlation. This implies that the features  $\phi_i \in \Phi$  contain a common part which is fully learned from each excerpt: most probably, this part of the encoding will also generalize well. Since



**Fig. 4.** Development of each CCA component’s correlation coefficient between the features extracted and selected from the increasing excerpts  $\psi_i \in \Psi$  and from the full recording,  $\psi_n$ . By  $\psi_4$ , the classification performance of the CNN has already converged; yet, many CCA coefficients have not converged and still continue to change.



**Fig. 5.** Development of the CCA distance between the stable (respectively unstable) parts of the  $\phi_i \in \Phi$  features and the complete representation  $\phi_i \in \Phi$ , computed using the mean CCA coefficient as described in Eq. 2. While the stable part maintains a high correlation to the encoding  $\phi_i \in \Phi$  in larger excerpts, the unstable part rapidly loses correlation and remains that way.

the rest of the encoding is not stable across the increasing excerpts, it likely does not generalize to unseen data samples.

## 6 Conclusion

In this paper we proposed the combination of Convolutional Neural Networks and Canonical Correlation Analysis as a framework to be used for the potential

reduction of test time in industrial manufacturing quality tests. Considering a set of increasing excerpts of the full test recording, all starting at 0, we also showed that some classification-relevant features remain stable over all excerpts, while others stay uncorrelated and may only represent statistical noise. The practical relevance of our findings is guaranteed, as our computational experiments performed on a real-world dataset from the automotive industry showed that the test time necessary for the vibroacoustic testing of ball nut assemblies can be reduced by 77.78% using our approach.

Since most datasets representing industrial manufacturing quality issues are heavily imbalanced, a potential direction for further work consists of the analysis of the robustness of the proposed framework against different dataset rebalancing techniques. Designed for discrete features, the consequences of their application in scenarios such as ours are non-trivial. Specifically, even if the synthetically created samples help improve the classification performance, it is not automatically ensured that they also possess a physical meaning. Additionally, it would be highly interesting to compare our results obtained via the usage of the linear CCA variant to the performance of its non-linear counterparts, the kernel CCA or the deep CCA.

## References

1. Dalenogare, L.S., Benitez, G.B., Ayala, N.F., Frank, A.G.: The expected contribution of Industry 4.0 technologies for industrial performance. *Int. J. Prod. Econ.* **204**, 383–394 (2018)
2. Diez-Olivan, A., Del Ser, J., Galar, D., Sierra, B.: Data fusion and machine learning for industrial prognosis: trends and perspectives towards Industry 4.0. *Inf. Fusion* **50**, 92–111 (2019)
3. Tao, F., Qi, Q., Liu, A., Kusiak, A.: Data-driven smart manufacturing. *J. Manuf. Syst.* **48**, 157–169 (2018)
4. Penz, C.A., Flesch, C.A., Nassar, S.M., Flesch, R.C., De Oliveira, M.A.: Fuzzy Bayesian network for refrigeration compressor performance prediction and test time reduction. *Expert. Syst. Appl.* **39**(4), 4268–4273 (2012)
5. Chickering, D.M., Heckerman, D., Meek, C.: Large-sample learning of Bayesian networks is NP-hard. *J. Mach. Learn. Res.* **5**, 1287–1330 (2004)
6. Wald, A.: Sequential tests of statistical hypotheses. *Ann. Math. Stat.* **16**(2), 117–186 (1945)
7. Wald, A., Wolfowitz, J.: Optimum character of the sequential probability ratio test. *Ann. Math. Stat.* **19**(3), 326–339 (1948)
8. Hotelling, H.: Relations between two sets of variates. *Biometrika* **28**(3/4), 321–377 (1936)
9. Hardoon, D.R., Szedmak, S., Shawe-Taylor, J.: Canonical correlation analysis: an overview with application to learning methods. *Neural Comput.* **16**(12), 2639–2664 (2004)
10. Arora, R., Livescu, K.: Kernel CCA for multi-view learning of acoustic features using articulatory measurements. In: *Symposium on Machine Learning in Speech and Language Processing* (2012)

11. Bucur, P.A., Frick, K., Hungerländer, P.: Quality classification methods for ball nut assemblies in a multi-view setting. Optimization online e-prints, eprint 2018-09-6796 (2018)
12. Kakade, S.M., Foster, D.P.: Multi-view regression via canonical correlation analysis. In: Bshouty, N.H., Gentile, C. (eds.) COLT 2007. LNCS (LNAI), vol. 4539, pp. 82–96. Springer, Heidelberg (2007). [https://doi.org/10.1007/978-3-540-72927-3\\_8](https://doi.org/10.1007/978-3-540-72927-3_8)
13. Raghu, M., Gilmer, J., Yosinski, J., Sohl-Dickstein, J.: SVCCA: singular vector canonical correlation analysis for deep learning dynamics and interpretability. In: Advances in Neural Information Processing Systems, pp. 6076–6085 (2017)
14. Morcos, A.S., Raghu, M., Bengio, S.: Insights on representational similarity in neural networks with canonical correlation. arXiv e-prints, [arXiv:1806.05759](https://arxiv.org/abs/1806.05759) (2018)
15. Galen, A., Arora, R., Bilmes, J., Livescu, K.: Deep canonical correlation analysis. In: International Conference on Machine Learning, pp. 1247–1255 (2013)
16. Bibby, J.M., Kent, J.T., Mardia, K.V.: Multivariate Analysis. Academic Press, London (1979)
17. Janssens, O., et al.: Convolutional neural network based fault detection for rotating machinery. *J. Sound Vib.* **377**(Suppl. C), 331–345 (2016)
18. Yildirim, Ö., Plawiak, P., Tan, R.-S., Acharya, U.R.: Arrhythmia detection using deep convolutional neural network with long duration ECG signals. *Comput. Biol. Med.* **102**, 411–420 (2018)
19. Jing, L., Zhao, M., Li, P., Xu, X.: A convolutional neural network based feature learning and fault diagnosis method for the condition monitoring of gearbox. *Measurement* **111**, 1–10 (2017)
20. 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)
21. Costa, Y.M., Oliveira, L.S., Silla Jr., C.N.: An evaluation of convolutional neural networks for music classification using spectrograms. *Appl. Soft Comput.* **52**, 28–38 (2017)
22. Choi, K., Fazekas, G., Sandler, M.: Explaining deep convolutional neural networks on music classification. arXiv preprint [arXiv:1607.02444](https://arxiv.org/abs/1607.02444) (2016)
23. Wu, Y., Mao, H., Yi, Z.: Audio classification using attention-augmented convolutional neural network. *Knowl.-Based Syst.* **161**, 90–100 (2018)
24. Cohen, J.: A coefficient of agreement for nominal scales. *Educ. Psychol. Meas.* **20**(1), 37–46 (1960)



# Convolutional Neural Network for Detection of Building Contours Using Multisource Spatial Data

George Papadopoulos<sup>1</sup>(✉), Nikolaos Vassilas<sup>1</sup>,  
and Anastasios Kesidis<sup>2</sup>

<sup>1</sup> Department of Computer Engineering,  
University of Western Attica, Attica, Greece  
[{ms13038, nvass}@uniwa.gr](mailto:{ms13038, nvass}@uniwa.gr)

<sup>2</sup> Department of Surveying and Geoinformatics Engineering,  
University of Western Attica, Attica, Greece  
[akesidis@uniwa.gr](mailto:akesidis@uniwa.gr)

**Abstract.** Building reconstruction from aerial photographs and other multi-source urban spatial data is a task endeavored using a plethora of automated and semi-automated methods ranging from point processes, classic image processing and laser scanning. Here, we describe a convolutional neural network (CNN) method for the detection of building borders. In particular, the network is based on the state of the art super-resolution model SRCNN and accepts aerial photographs depicting densely populated urban area data as well as their corresponding digital elevation maps (DEM). Training is performed using three variations of this urban data set and aims at detecting building contours through a novel super-resolved heteroassociative mapping. Another novelty of our approach is the design of a modified custom loss layer, named Top-N, whereby the mean square error (MSE) between the reconstructed output image and the provided ground truth (GT) image of building contours is computed on the  $2N$  image pixels with highest values, where  $N$  is the number of contour pixels in GT. Assuming that most of the  $N$  contour pixels of the GT image are also in the top  $2N$  pixels of the reconstruction, this modification balances the two pixel categories and improves the generalization behavior of the CNN model. It is shown, in our experiments, that the Top-N cost function offers performance gains in comparison to standard MSE. Further improvement in generalization ability of the network is achieved by using dropout.

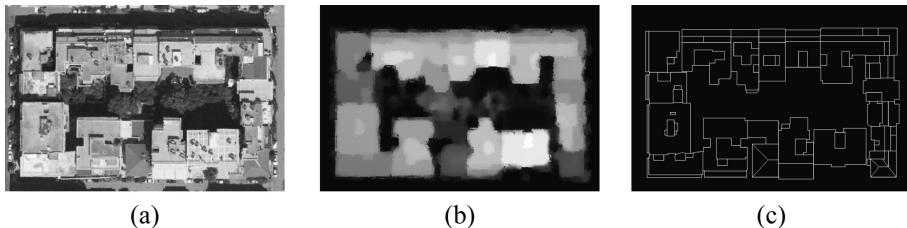
**Keywords:** Building contours · Convolutional neural networks · Digital elevation maps

## 1 Introduction

Building contour detection in an urban setting is a complex problem for automated computer methods to tackle due to high object density and scene complexity. It does have diverse applications which range from virtual tourism, transportation navigation and landscape planning. For this reason, significant research has been conducted and

progress has been gradually achieved over the last decades. Early methods were severely constrained due to their reliance on a generic model which assumed that buildings follow a certain pattern and thus failed to provide reliable results when applied to varied urban environments [1]. Other early attempts used shadow data combined with 2D building blobs derived from digital elevation data [2]. Unfortunately, such models were hampered due to low-resolution ground sampling data, occlusions and shadows. Some researchers have used photogrammetric techniques which availed of stereoscopic images with several of these methods using optical images while others elevation data. An example of the former category is Lang and Forstner [3] and Fraser et al. [4], who reconstructed 3D buildings from high-resolution IKONOS stereo imagery. Airborne laser scanning equipment became more reliable and refined during the late 1990s and early 2000s, thus becoming an important source of obtaining digital surface maps (DSM). Mass and Moleman developed various approaches to detecting building contours using DSMs [5]. Fusing optical and elevation data was the next logical step thus Haala [6] combined DSMs with optical images in order to extract buildings and trees in an urban environment. There still exist probabilistic methods that accept DEMs and utilize an object approach. Lafarge et al. [7] used marked point processes to roughly approximate building contours via rectangular structures. These rectangular footprints were then regularized by taking into account the local context of each rectangle and detecting roof height discontinuities. Descombes and Zerubia [8] altered the previous method by introducing an energy function which takes into account the height of the building as well as prior knowledge about the general layout of buildings in urban settings. Simulated annealing was then employed in order to minimize the energy function. Still other researchers used observed point clouds from LIDAR data. Rottensteiner et al. [9] separated points being on the ground from those belonging to buildings and other objects. This was accomplished by an analysis of the height differences of a digital surface model passing through the original LIDAR points and a digital terrain model. During the last years, there has been an increasing number of contributions that apply deep convolutional neural networks to applications that return a whole image instead of the category of the presented data. For instance, Dong et al. [10] used a three-layer convolutional network, named SRCNN, to learn a direct mapping between low and high resolution images. This mapping was represented by a deep convolutional neural network that took the low-resolution image as input and returned a high-resolution version of the image. The results were comparable or even better than well-established sparse coding dictionary methods. In our application we propose a deep convolutional neural network that can directly detect building contours. Due to the nature of the work in [10] which exhibits several features that were considered akin to our application, we applied a modified version of this network as the basis of our own network. In our case the modified SRCNN, which we have named BCDCNN (Building Contour Detector Convolutional Neural Network) accepts a tuple of available data in the form <[optical, DEM], GT> which is comprised of an optical and a DEM input pair along with the

corresponding ground truth output. The network is expected to approximate the GT data given the [optical, DEM] pair. A typical example of data<sup>1</sup> used in our experiments is shown in Fig. 1.



**Fig. 1.** (a) Optical (grayscale) channel input, (b) DEM channel input, and (c) Ground truth data.

The rest of the paper is organized as follows. Section 2 briefly discusses the architecture of BCDCNN. Section 3 describes the experiments that were conducted by tweaking the various parameters of the proposed network architecture, in order to investigate the system's performance under various setups. Section 4 presents the results of our experiments and discusses the efficiency of the proposed method while Sect. 5 draws the conclusions.

## 2 Methodology

### 2.1 Formulation of BCDCNN

The proposed BCDCNN model is based on the Super-Resolution Convolutional Neural Network (SRCNN) presented by Dong et al. [10]. However, if one can say that SRCNN implements a super-resolved autoassociative mapping in the sense that a low resolution image is mapped onto the high resolution version of itself, BCDCNN implements a super-resolved heteroassociative mapping since low resolution elevation data are mapped onto their associated high resolution building contours available during training from the ground truth data. In particular, similar to the first convolutional layer of SRCNN, BCDCNN accepts a low resolution elevation map at the input which is upsampled to the desired higher resolution (the upsampling scale is determined by the corresponding high resolution optical image) using the joint (optical + DEM) mean-shift based upsampling algorithm described in [11]. Following Dong et al. [10], we will assume that the high resolution optical image combined with the preprocessed low resolution elevation map constitute the mixed resolution input  $X$  to the network. The goal of the convolutional network is then to reconstruct an image  $F(X)$  that is similar to the corresponding ground truth high-resolution building contours

---

<sup>1</sup> These data constitute the first variation of the training set, as explained in the Methodology section.

image  $Y$ . In order to accomplish this, BCDCNN uses a mapping  $F$  from input to reconstruction (output) which consists of the following three operations:

- *Patch extraction and representation.* Patches from the mixed resolution image  $X$  are extracted, then processed by the filter bank of the first convolutional layer and, finally, represented as a set of feature maps. This can be mathematically expressed as the operation

$$F_1(X) = \max(0, W_1 * X + B_1) \quad (1)$$

where  $W_1 = \{W_1^k \mid 1 \leq k \leq N_1\}$  and  $B_1 = \{B_1^k \mid 1 \leq k \leq N_1\}$  with  $W_1^k$  being the  $k$ -th 3-D filter of the first layer's filter bank  $W_1$ ,  $B_1^k$  the corresponding bias term and  $F_1(X)$  the set of  $N_1$  feature maps. As induced by Eq. (1), this layer includes a ReLU non-linearity. Each of the  $N_1$  filters is of size  $s_1 \times s_1 \times N_0$ , with  $N_0$  denoting the number of channels in the input image ( $N_0 = 2$  for the first layer). Finally, operator '\*' signifies convolution.

- *Non-linear feature map transformation.* In the second convolutional layer, the  $N_1$  feature maps generated by the previous operation are non-linearly transformed into another set of  $N_2$  feature maps by applying  $N_2$  filters of size  $s_2 \times s_2 \times N_1$  and then, as before, passing the results from a ReLU. This operation can be described mathematically as

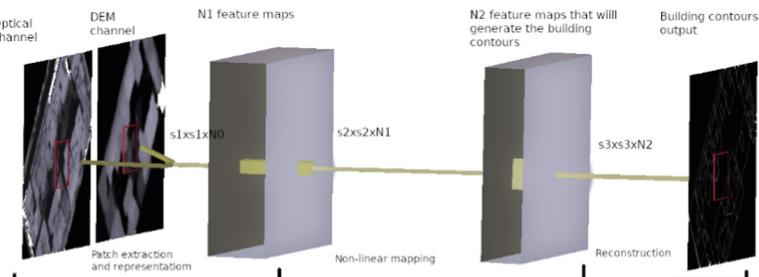
$$F_2(X) = \max(0, W_2 * F_1(X) + B_2) \quad (2)$$

where  $W_2$  contains  $N_2$  filters of size  $s_2 \times s_2 \times N_1$  and  $B_2$  is  $N_2$ -dimensional.

- *Building contour reconstruction:* Finally, the feature maps of the previous stage are aggregated to generate the high-resolution building contour image. The reconstruction operation is implemented as a linear convolution layer,

$$F_3(X) = W_3 * F_2(X) + B_3 \quad (3)$$

where  $W_3$  corresponds to a single filter of size  $s_3 \times s_3 \times N_2$  and  $B_3$  is the final layer's bias term.



**Fig. 2.** The proposed 3-layer convolutional system architecture.

Figure 2 illustrates the proposed network architecture in which the input to the network, the optimal output and the size of the convolution kernels applied at each layer are shown.

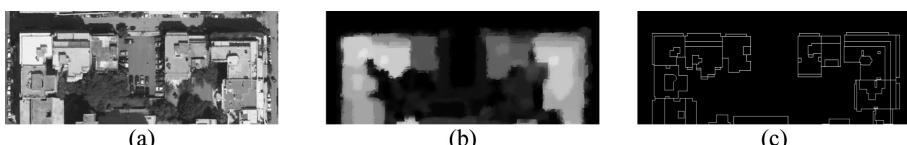
The goal is to get a building contour map  $F(Y)$  which is as close as possible to the ground truth. However, unlike classification type of applications in which the training procedure associates input images to, usually, a few class labels, the proposed system is presented with a far more difficult and challenging problem, that is, learning a heteroassociative mapping from a quite limited training set of  $\langle \text{input}, \text{output} \rangle$  pairs and then expecting to generalize on new pairs of building top-view images. Further elaborating on the complexity of our data sources there are four different types of edges that the network must learn to differentiate.

- Elevation edges that are simultaneously optical edges, which is mostly the case.
- Optical edges that are not elevation edges: For instance, rooftops of neighboring buildings of different colors but same heights.
- Elevation edges that are not optical: For example, a rooftop of the same color as an adjacent street and at different heights.
- Implied edges: For instance, rooftops with the same color and same height. This is the most difficult case.

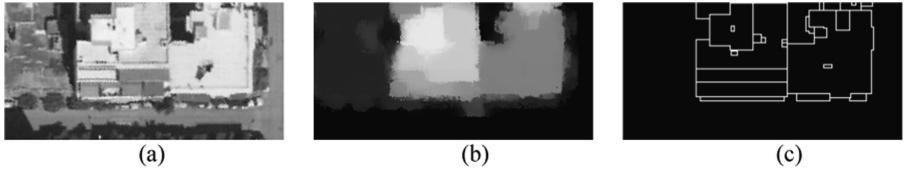
Just to make the problem even more difficult, the available elevation data – carrying most of the building contours information – are at a five times lower spatial resolution than the optical images and the associated building contours. Hence, the proposed CNN architecture is actually performing a combination of elevation data super-resolution assisted by available high-resolution optical images and a heteroassociative mapping to building contours.

## 2.2 Data Preprocessing

The proposed network is trained using high resolution aerial orthophotographs of Kallithea, a densely populated area in Attica, Greece, as well as the corresponding low resolution digital elevation model and the high resolution ground truth building contours. Figures 1a and b depict the optical and elevation data of a building block (named BLOCK1), respectively. In particular, to arrange the two data sources as two channels of a multimodal image, the depicted DEM has been upsampled with a scale of 5 using the joint mean shift algorithm [11]. A second block of buildings from the same area has also being selected and sliced to produce a complementary dataset for training (Fig. 3) and testing (Fig. 4). As before, the DEM channels have been 5x upsampled using joint mean shift.

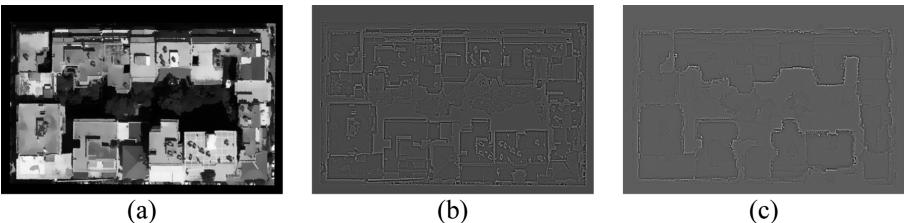


**Fig. 3.** (a) Optical, (b) DEM, and (c) Ground truth training data from BLOCK2.



**Fig. 4.** (a) Optical, (b) DEM, and (c) Ground truth test data.

Furthermore, we used three variations of the training data. More specifically, the first variation is comprised of the original optical data and the mean shift upsampled DEM data (Figs. 1, 3 and 4). Moving on to the second variation, the optical channel has also been filtered with the mean shift edge preserving smoothing algorithm [12] with the so filtered BLOCK1 been shown in Fig. 5a. Finally, in the third variation the mean shift optical & DEM data have been filtered by a Laplacian of Gaussian (LoG) operator (see Figs. 5b and c). The last two variations have been considered as an attempt to reduce the effective dimensionality of the input data and improve the generalization ability of the proposed system.



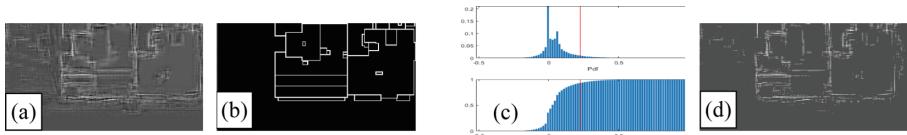
**Fig. 5.** Filtered BLOCK1 data: (a) Mean shift filtered optical channel, (b) LoG-filtered optical image, and (c) LoG-filtered DEM.

Although test data for the last two variations are not shown, it will be implied that the same mean shift and LoG processing has also been applied in the test dataset in all experiments performed on variations 2 or 3.

### 2.3 Cost Function Modification

A typical cost function to be minimized during network training is the root mean square error (RMSE) between the actual reconstruction and the ground truth which in this particular application is a binary image with 1 for pixels belonging to the building contour and 0 for all other pixels. However, only a very small proportion of pixels in the GT image (and its subimages and patches thereafter) will have a value of 1 and will pull the corresponding neuron outputs of the reconstruction layer. All the remaining neuron outputs (or pixel values), no matter how close to zero they are, will be pushed towards zero. Although we have not included any postprocessing stage to clean up the reconstructed binary contours, it is intuitively evident that the neuron outputs of the

reconstructed image that correspond to background and have close to zero or negative values, could be set aside from the derivative computations of the back-propagation phase, e.g. by setting them to zero. On the other hand, neuron outputs wrongly close to 1 should play a role in the back-propagation phase in order to be pushed down to lower values. A second point we can make regarding weight adaptation in this application is that all output neurons share the same weights and that these weights should be given a chance to adapt in such a way as to satisfy confronting demands: to push some output neurons to 1 and other neurons to 0. Since the proportion of 1-pixels is much smaller than that of 0-pixels, it is expected that the shared weights will prioritize minimizing the error of the “many” background pixels instead of the “few” contour pixels. This comment highlights network training difficulties in heteroassociative mappings that arise due to unequal pixel-class probabilities and resembles the necessity for class-balanced datasets in classification problems. In order to balance the weight adaptation process to serve equally well the contour and non-contour pixels, we propose to substitute the typical RMSE cost criterion that involves all neuron outputs of the reconstruction layer by a novel custom cost layer which we have named Top-N. Under this scheme the RMSE between the reconstructed image and the corresponding GT is calculated only for those pixels that belong to the  $2N$  pixels with highest values, where  $N$  is the number of contour pixels in GT. Assuming that most of the  $N$  contour pixels of the ground truth image are also in the top  $2N$  pixels of the reconstruction, this scheme satisfies the imposed balancing criterion.



**Fig. 6.** (a) A low-level reconstruction of the test image, (b) the corresponding GT, (c) pdf and cdf of intensity levels and Top-N threshold, and (d) Top-N version of the reconstruction.

In practical terms, the threshold used to specify the top  $2N$  pixel values is calculated as follows: We calculate the probability distribution function and cumulative distribution function of the intensity levels for each image used during training and retain only the pixels that have an intensity above that threshold<sup>2</sup>. This is depicted in Figs. 6 (a) – (d), which show a low quality reconstruction of the test data, the corresponding ground truth, the Top-N threshold calculated as the percentage of pixels above the Top-N intensity and the Top-N version of the reconstruction, respectively.

<sup>2</sup> Actually, we use the average value of the intensity level for a whole batch in order to accelerate the computation procedure.

## 3 Experiments

### 3.1 Training Set Preparation

To satisfy the requirement of large numbers of training data to properly train deep neural networks we performed data augmentation [13]. Firstly, we extracted  $33 \times 33$  patches of the input data (optical + DEM) along with the corresponding  $21 \times 21$  patches of the GT data (GT patches are smaller due to “valid” convolutions with  $9 \times 9$ ,  $1 \times 1$  and  $5 \times 5$  kernels). The data were then augmented with rotations at multiples of  $90^\circ$  and with their vertical flips. In this manner we constructed tuples of input data and GT in the form <[optical\_section, DEM\_section], GT\_section>. The procedure described in the following sections was followed for each of the three variations of our training data set (original, Mean-Shift processed, LoG processed).

### 3.2 BCDCNN Configurations and Applied Metrics

We ran experiments with two convolution kernel sizes for the second layer using  $1 \times 1$  and  $3 \times 3$  mapping kernels. Across all three layers of our model the sizes of the convolution kernels we tested for were 9-1-5<sup>3</sup> and 9-3-5<sup>4</sup>. In addition, our network used 64 feature maps for the first layer, 32 for the second and 1 for the last, which is henceforth denoted as 64-32-1. Furthermore, for each training data set and each convolution kernel size we assessed performance for three cases: (a) No dropout; (b) Dropout 50%, i.e. dropout rate of 50% after the RELU activation function of the first layer; (c) Dropout 50%-50%, i.e. dropout rate of 50% after the RELU activations of the first and second layers. We used a learning rate of  $10^{-4}$  for layers 1 and 2 while the learning rate was  $10^{-5}$  for layer 3. Also the weight decay was  $5 \times 10^{-3}$  for all layers and the batch size was set to 128. In total, we ran 18 experiments for the 64-32-1 configuration. Finally, we utilized the RMSE and PSNR metrics to assess performance of our network.

## 4 Experimental Results

All presented results pertain to the 9-1-5 or 9-3-5 convolution kernel choices and to the 64-32-1 feature maps configuration, i.e. number of feature maps at the output of each convolutional layer. Actually, we conducted several tests to assess how the number of feature maps affect the performance of the network. Specifically, we experimented with networks of 128-64-1 and 256-128-1 feature map configurations. However, even though performance is increased (the RMSE for the Original data sets at epoch 60 decreases from 3,4048 to 3,3384 and then to 3,2077 for the larger configurations), the heavy computational costs prohibited their use in the sequel.

---

<sup>3</sup>  $9 \times 9$ ,  $1 \times 1$ ,  $5 \times 5$  for the first, second and third layer respectively.

<sup>4</sup>  $9 \times 9$ ,  $3 \times 3$ ,  $5 \times 5$  for the first, second and third layer respectively.

#### 4.1 Comparison Between MSE and Top-N Custom Loss Layers

Our Top-N custom cost layer leads to lower RMSE and higher PSNR values as shown in Tables 1, 2 and 3. Table 1 depicts the RMSE and PSNR of our test data for the case of training on the Original, the Mean Shift and LoG data sets, respectively. In all cases our custom Top-N layer exhibits lower RMSE and higher PSNR values than the typical MSE cost layer. For instance, in Table 1 regarding the Dropout 50% 9-1-5 case the proposed Top-N cost layer produced an RMSE 3.37% lower than the corresponding MSE cost layer. Comparing corresponding entries for the PSNR for the Original data (Table 1), 5 out of 6 entries have a higher value for the 9-3-5 network. Likewise, for the Mean Shift and LoG processed data (Tables 2 and 3) most PSNR entries are higher for the 9-3-5 network. Nonetheless, since the 9-3-5 configuration was by 16.5% slower than the 9-1-5 configuration and since – as shown in Tables 1 through 3 – there was only a slight improvement either in RMSE or in PSNR compared to 9-1-5, we decided to consider the more practical 9-1-5 configuration as was also argued in Dong et al. [10].

**Table 1.** RMSE and PSNR for test data trained on the Original dataset.

Loss layer	Dropout 50%		Dropout 50%-50%		NoDropout	
	RMSE	PSNR	RMSE	PSNR	RMSE	PSNR
	Min	Max	Min	Max	Min	Max
Top-N (9-1-5)	0,10442	15,205	0,10537	15,087	0,10565	15,268
Top-N (9-3-5)	0,10620	15,217	0,10591	15,269	0,11408	15,257
MSE (9-1-5)	0,10806	14,293	0,10961	14,780	0,10802	14,919
MSE (9-3-5)	0,10886	14,866	0,10816	14,888	0,10793	14,930

**Table 2.** RMSE and PSNR for test data trained on the Mean Shift processed dataset.

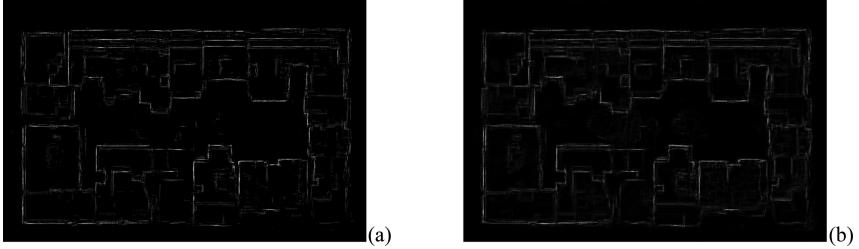
Loss layer	Dropout 50%		Dropout 50%-50%		NoDropout	
	RMSE	PSNR	RMSE	PSNR	RMSE	PSNR
	Min	Max	Min	Max	Min	Max
Top-N (9-1-5)	0,10423	15,257	0,10549	15,079	0,10486	15,067
Top-N (9-3-5)	0,10263	15,263	0,10343	15,263	0,10865	15,283
MSE (9-1-5)	0,10833	14,903	0,10977	14,807	0,10926	14,832
MSE (9-3-5)	0,10817	14,912	0,10912	14,841	0,10923	14,842

**Table 3.** RMSE and PSNR for test data trained on the LoG processed dataset.

Loss layer	Dropout 50%		Dropout 50%-50%		NoDropout	
	RMSE	PSNR	RMSE	PSNR	RMSE	PSNR
	Min	Max	Min	Max	Min	Max
Top-N (9-1-5)	0,10948	14,831	0,10811	14,929	0,10779	15,127
Top-N (9-3-5)	0,10608	15,031	0,10637	15,106	0,10763	15,148
MSE (9-1-5)	0,11005	14,647	0,11314	14,489	0,11093	14,803
MSE (9-3-5)	0,10955	14,005	0,10988	14,746	0,10839	14,824

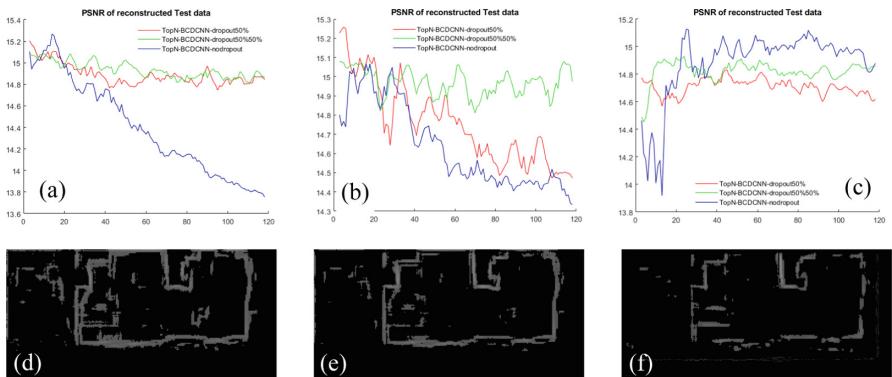
## 4.2 Detection of Building Borders

In Fig. 7 two typical reconstructions for the training data Block1 are shown for two configurations of the network. BCDCNN was able to learn the association of building contours to the input data sources.



**Fig. 7.** (a) Reconstruction of train data for Original data set and Top-N Cost Layer, (b) Reconstruction of train data for Original data set and MSE Cost Layer

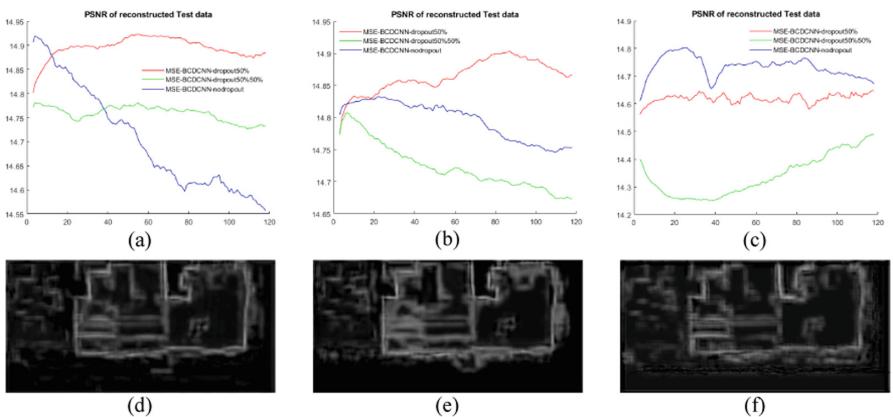
Our network can also generalize as the reconstructions of the test data for networks trained on the three variations for the proposed Top-N and MSE loss layer show.



**Fig. 8.** Top row: PSNR for test data on network trained for (a) Original data and Top-N cost layer, (b) Mean Shift data and Top-N cost layer, (c) LoG data and Top-N cost layer. Bottom row: Reconstruction of test data at PSNR peak for (d) Original data and Top-N cost layer, (e) Mean Shift data and Top-N cost layer, (f) LoG data and Top-N cost layer.

According to Fig. 8a the highest PSNR for the test data set was at epoch 55 for the dropout 50% case. We thus found the peak of the PSNR curves and then reconstructed at that specific instant. This process was repeated for all our training data variations and the resulting reconstructions are shown in Figs. 8d through f. The corresponding experiments for the MSE cost layer are shown in Fig. 9. It has to be noted that we did not employ any post-processing stage to improve the obtained building contours as this

will be the case of a relaxation system currently under development. From Figs. 8 and 9 we readily observe that deciding about how to improve the generalization ability of the network is not straightforward. Perhaps, one can say that when the effective input dimensionality is high (i.e. when the variance of the input pixel values is large) as is the case for the Original data sets, the network exhibits poor generalization behaviour (see the blue curves of Figs. 8a and 9a). As the effective dimensionality is progressively reduced through the imposed smoothing from the Mean Shift and LoG data preprocessing the generalization ability of the network is improved and, in the case of LoG, even surpasses the cases that use dropout in one or two layers.



**Fig. 9.** Top row: PSNR for test data on network trained for (a) Original data and MSE cost layer, (b) Mean Shift data and MSE cost layer, (c) LoG data and MSE cost layer. Bottom row: Reconstruction of test data at PSNR peak for (d) Original data and MSE cost layer, (e) Mean Shift data and MSE cost layer, (f) LoG data and MSE cost layer.

A second remark that we can make is that by using 50% dropout on one or two layers the network resists better to overfitting. Specifically, in the case of training data sets with relatively high effective dimensionality as is the case with the Original and Mean Shift processed data sets, dropout (either in one or in two layers) proves to be the necessary choice for network generalization. Finally, in accordance to the comparative results of Tables 1, 2 and 3, a comparison of Figs. 8 and 9 shows that there is a slight improvement in PSNR under any training data set variation when using the Top-N cost layer.

## 5 Conclusions – Future Work

We have demonstrated a deep neural network configuration that given low resolution elevation data of an urban area and corresponding high resolution optical data of the same area can perform a hetero associative mapping to a new image, which contains the building contours. Our proposed Top-N custom layer seems to offer performance benefits, wherein the RMSE and PSNR exhibit better performance for the Top-N layer as opposed to the MSE cost layer. We also examined the effect of adding more feature

maps and we have shown that dropout is mostly necessary in order for the model to generalize. It is very interesting to notice that training with the LoG data set was the only case in which the network managed to generalize without using dropout (Figs. 8c and 9c), presumably a result of the reduced dimensionality. The problem we tried to solve using deep neural networks is extremely complex due to the varying context around true building contours in an urban environment. We conjecture that given more training data the performance of the network will increase but hand-crafting such ground truth data is a very tedious and time costly procedure. In the near future we intend to build a pixel-based contour detector and a super-resolution system for digital elevation data (DEM) capable of increasing the resolution of such data by at least 5 times.

## References

1. Mason, S., Baltsavias, E.: Image-based reconstruction of informal settlements. In: Gruen, A., Baltsavias, E.P., Henricsson, O. (eds) Automatic Extraction of Man-Made Objects from Aerial and Space Images (II). Monte Verità (Proceedings of the Centro Stefano Franscini Ascona), pp. 97–108. Birkhäuser, Basel (1997). [https://doi.org/10.1007/978-3-0348-8906-3\\_10](https://doi.org/10.1007/978-3-0348-8906-3_10)
2. Li, J., Ruther, S.H.: IS-Modeller: a low-cost image-based tool for informal settlements planning. In: Geoinformatics 1999 Conference, Ann Arbor (1999)
3. Lang, F., Forstner, W.: 3D-city modeling with a digital one-eye stereo system. In: Proceedings of the XVIII ISPRS-Congress (1996)
4. Fraser, C.S., Baltsavias, E., Gruen, A.: Processing of Ikonos imagery for submetre 3D positioning and building extraction. *ISPRS J. Photogramm. Remote Sens.* **56**(3), 177–194 (2002)
5. Maas, H.-G., Vosselman, G.: Two algorithms for extracting building models from raw laser altimetry data. *ISPRS J. Photogramm. Remote Sens.* **54**(2-3), 153–163 (1999)
6. Haala, N., Brenner, C.: Extraction of buildings and trees in urban environments. *ISPRS J. Photogramm. Remote Sens.* **54**(2-3), 130–137 (1999)
7. Lafarge, F., et al.: Automatic building extraction from DEMs using an object approach and application to the 3D-city modeling. *ISPRS J. Photogramm. Remote Sens.* **63**(3), 365–381 (2008)
8. Ortner, M., Descombes, X., Zerubia, J.: Building outline extraction from digital elevation models using marked point processes. *Int. J. Comput. Vis.* **72**(2), 107–132 (2007)
9. Rottensteiner, F., Briese, C.: A new method for building extraction in urban areas from high-resolution LIDAR data. In: International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences, vol. 34, No. 3/A, pp. 295–301 (2002)
10. Dong, C., Loy, C.C., He, K., Tang, X.: Image super-resolution using deep convolutional networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **38**, 295–307 (2016)
11. Vassilas, N., Tsenoglou, T., Ghazanfarpour, D.: Mean shift-based preprocessing methodology for improved 3D buildings reconstruction. *WASET Int. J. Civ. Environ. Struct. Constr. Architectural Eng.* **9**(5), 575–580 (2015). (also in Proc. ICCVISP 2015, Berlin, Germany)
12. Comanicu, D., Meer, P.: Mean shift: a robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(5), 603–619 (2002)
13. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint [arXiv:1409.1556](https://arxiv.org/abs/1409.1556) (2014)

# **Fuzzy - Vulnerability - Navigation Modeling**



# A Meta-multicriteria Approach to Estimate Drought Vulnerability Based on Fuzzy Pattern Recognition

M. Spiliotis<sup>1</sup> , A. Iglesias<sup>2</sup> , and L. Garrote<sup>3</sup>

<sup>1</sup> Department of Civil Engineering, Democritus University of Thrace,  
Kimmeria Campus, 67100 Xanthi, Greece

m.spiliotis@gmail.com

<sup>2</sup> Department of Agricultural Economics,  
Technical University of Madrid, Madrid, Spain  
ana.iglesias@upm.es

<sup>3</sup> Department of Hydraulics, Energy and Environment,  
Technical University of Madrid, Madrid, Spain  
l.garrote@upm.es

**Abstract.** The objective of this paper is to explore a new integrated approach to estimate drought vulnerability taking into account the characteristics of a system that make it likely to be affected by an external risk. A meta-multicriteria approach is adopted since the problem itself modulates the multiple criteria method. Firstly, relevant information is grouped into drought sensitivity and adaptive capacity criteria. Instead of the estimation of a unique score for the vulnerability, a classification of the vulnerability to drought into several categories is proposed. Based on the maximum and minimum values of the above criteria initially, four non-ordered categories are established initially to characterize the vulnerability to drought. In order to classify water-scarce countries into the four or more categories the fuzzy pattern recognition is exploited. The proposed approach is applied to estimate drought vulnerability in selected Mediterranean countries. A choice that strengthens the meta-multicriteria character of the proposed approaches is that the categories are not ordered, but they are modulated from all the combination of the extreme points.

**Keywords:** Vulnerability · Fuzzy pattern recognition · Meta-multicriteria methods · Drought

## 1 Introduction

Drought is a natural temporary condition of reduction in precipitation and water availability with respect to normal values (e.g. mean values), spanning a long a significant period of time and covering a wide region (Iglesias et al. 2009). Risk assessment of droughts is increasingly required, especially because their impacts do not only depend on temporary precipitation values below the normal conditions, but to a great degree on social factors and management (Iglesias et al. 2015; Brack et al. 2009).

Drought vulnerability refers to the characteristics of a group in terms of its capacity to anticipate, cope with, resist and recover from the impact of drought (Iglesias et al. 2009). Assessment of vulnerability involves identifying the characteristics of the systems that modify the level of drought risk. Understanding the vulnerability to drought may increase the preparedness of a region and hence limit the greatest and most devastating effects of drought. Here the authors aim to understand the underlying causes of vulnerability derived from inadequate structures, management and technology or by economic, environmental and social factors, in order to provide information for the potential design of drought management. Vulnerability to drought in Mediterranean countries is determined by exposure to lower precipitation and also by the diverse social response (Iglesias et al. 2007).

The usual indicator-based approaches (e.g., Naumann et al. 2014) or the multi-criteria approaches (e.g., Nardo et al. 2005) are useful to understand the role of policy relevant variables and to propose responses to drought. However, a limitation of both methods is the attribution of weights to the variables analyzed. Although weights may be assigned by participatory methods allowing stakeholders to establish priorities, this approach is unrealistic at the regional or continental scale.

Here a meta-monicriteria method is proposed combined with a fuzzy analysis in order to assess the drought vulnerability at the country level. First, multi-levels of aggregation to combine several indicators are established; and second the final aggregation is based on a logical structure which is based on the nature of the examined problem.

Tsakiris et al. (2015) proposed a similar method in order to evaluate measures for responding to water shortage based on the definition of two sets of criteria: the beneficial and the constraining criteria. The evaluation is directed by the sense of compatibility of the constraining criteria with the beneficial criteria. This compatibility between the constraining criteria with the beneficial criteria is expressed by the use of suitable fuzzy implications.

The concept of non-ordered categories provides a meta-monicriteria property since the non-ordered categories are modulated based on the problem itself and they can more cover the decision space (Fig. 1). In (Kazakis et al. 2018; Spiliotis et al. 2015) the use of non-ordered categories to deal with the roadmap for water scarcity is proposed. Belacel et al. (2001) used the non-ordered categories in medicine diagnostics in cases where the ordered categories were unsuitable.

Here the concepts of sensitivity and adaptive capacity are used to explore water vulnerability to drought in a novel approach. First four major categories are defined based on the level of adaptive capacity and the sensitivity of each geographical unit. The categories are defined and hence, the categorization is achieved based on the fuzzy pattern recognition. A corresponding roadmap in order to reduce vulnerability to drought for each category may be proposed. Here non-ordered categories are used. The categories consist of all possible combinations between the highest and lowest values of the adaptive capacity and the sensitivity criteria. Finally, the authors apply the proposed methodology to the case of the Mediterranean region. This is a good case study to test the methodology due to the fact that a variety of countries with different meteorological, hydrological, ecological, economic and social regimes exist in the region. The information used in this application is extracted from publicly available

global datasets; however the method is designed to be easily applicable on a smaller scale, where more detailed data could be obtained directly from stakeholders or local datasets. The differences between other applications of fuzzy pattern recognition in water resources management problems (e.g. Zhou et al. 1999; Xuesen et al. 2009), will be presented more analytically below. In brief, in this work, non-ordered categories instead of ordered categories are used.

## 2 The Approach to Explore Drought Vulnerability

### 2.1 Criteria to Define Drought Vulnerability

The proposed categorization of drought vulnerability is based on two groups of criteria. In the first group, the sub-criteria or indices are aggregated to evaluate the sensitivity of the examined water system. In order to evaluate the sensitivity the authors put emphasis on the physical parameters of the system. Sensitivity variables mainly reflect the sensitivity of regional water resource conditions, agricultural systems, and population density to drought (Wu et al. 2013). In fact the values of the examined first group of criteria, which comprise the sensitivity criterion, are difficult to change through policy.

The second group of sub-criteria indices, which are aggregated separately, describes the adaptive capacity to drought of the examined water system. Adaptation capacity variables mainly reflect the factors, which reduce exposure and vulnerability. The value of these variables can be changed following a set of policy measures.

Initially, a system is vulnerable to drought if it is sensitive, namely if the system can be potentially affected or impacted by drought. Secondly, the magnitude of the real impacts depends on the capacity of the system to adapt to drought. Sequentially, the challenge is to define a way to aggregate the evaluation of the sensitivity and the adaptive capacity criteria.

An interesting point of the proposed methodology is that instead of the estimation of a unique score for the vulnerability, a classification of the vulnerability to drought into several categories is proposed. As mentioned before, the examined categories are produced by the possible combinations between the extreme values of the adaptive capacity criterion and the sensitivity criterion even if the two concepts are in conflict.

### 2.2 Sensitivity Criteria

First the first set of variables are combined in order to assess the degree of sensitivity to drought. As mentioned before, the sensitivity criterion mainly reflects the sensitivity of regional water resource conditions, agricultural systems, and population density to drought (Wu et al. 2013). Several variables may be selected to represent the multiple dimensions of the system, including hydrological, economical and geographical factors. The variables selected in this work are population density ( $\text{inhab}/\text{km}^2$ ), long-term average precipitation in depth ( $\text{mm}/\text{yr}$ ), total water withdrawal per capita ( $\text{m}^3/\text{inhab}$ ) and dam capacity per capita ( $\text{m}^3/\text{inhab}$ ). These values represent physical factors and are difficult to change in the long term.

A simple additive aggregator is used to combine the normalized values of the variables which are linked to sensitivity. Thus, the degree of sensitivity for the examined area,  $S$ , is determined as follows:

$$S(x) = \sum_{p=1,\dots,P} w_p \cdot \mu_p(x) \quad (1)$$

In which  $P$  is the number of sensitivity criteria and  $\mu_p$  and  $w_p$  the membership function and the weight of each criterion  $p$  respectively. The evaluation of the sensitivity criteria is related to the question “has the system the ability to address a soft (or mild) drought episode”? The selected indices are shown in Table 1. Similar indices for obtaining the sensitivity criteria can be found in (Wu et al. 2013).

The weighted aggregation of the sub-criteria or indexes is less subjective compared with the final weighted aggregation of the criteria, since many times the sub-criteria lead to the same direction (e.g. sub-criteria to assess the water availability for the sensitivity main criterion).

### 2.3 Adaptive Capacity Criteria

Adaptive capacity variables are selected to reflect the factors that reduce exposure and vulnerability. The value of these variables can be changed by implementing a set of policy measures. The adaptive capacity criterion for a water system describes to what degree the water system has the ability to reorganize its components (e.g. to achieve the reduction of the demand, to implement an emergency plan etc.) to cope with and recover from a drought. The adaptive capacity variables describe the human adaptive capacity, the water adaptive capacity and furthermore, the adaptive capacity based on the nature of water demands. In the same way as in the sensitivity criterion, a simple additive aggregator is used to combine the normalized values of the variables which are linked to adaptive capacity. Thus, the degree of the adaptive capacity for the examined area,  $AC$  is determined as follows:

$$AC(x) = \sum_{j=1,\dots,J} w_j \cdot \mu_j(x) \quad (2)$$

In which  $J$  is the number of adaptive capacity criteria and  $\mu_j$  and  $w_j$  are the membership function and the weight of each adaptive capacity criterion  $j$  correspondingly.

### 2.4 Combining Sensitivity and Adaptive Capacity

At the final stage the combination of the sensitivity and the adaptive capacity is modulated based on the categorization of these criteria. All combinations of sensitivity and adaptive capacity are possible (Fig. 1a).

Based on the sensitivity and adaptive capacity criteria four categories are initially considered: (1) the ideal point (high adaptive capacity and low sensitivity); (2) anti-ideal point (low adaptive capacity and high sensitivity); (3) no problem-no action (low

**Table 1.** Criteria, variables and sources of data used in this study

Criteria		Indices	Units	Source of data
Sensitivity criteria	S1	Population density	inhab/km <sup>2</sup>	AQUASTAT (FAO) <a href="http://www.fao.org/nr/water/aquastat/data/query/index.html?lang=en">http://www.fao.org/nr/water/aquastat/data/query/index.html?lang=en</a>
	S2	Long-term average precipitation in depth	mm/yr	AQUASTAT (FAO)
	S3	Total water withdrawal per capita	m <sup>3</sup> /inhab/yr	AQUASTAT (FAO)
	S4	Dam capacity per capita	m <sup>3</sup> /inhab	AQUASTAT (FAO)
Adaptive capacity criteria	AC1	Fertilizer consumption	kilograms per hectare of arable land	World bank <a href="http://data.worldbank.org/indicator">http://data.worldbank.org/indicator</a>
	AC 2	Improved water source	% of population with access	AQUASTAT (FAO)
	AC 3	Energy use	kg of oil equivalent per capita	World bank
	AC 4	GDP per capita	2010 US\$	World bank
	AC 5	Human Development Index	dimensionless	UNDP
	AC 6	Freshwater withdrawal as % of total actual Renewable water resources	%	AQUASTAT (FAO)
	AC 7	Share of the cultivated area equipped for irrigation	%	AQUASTAT (FAO)

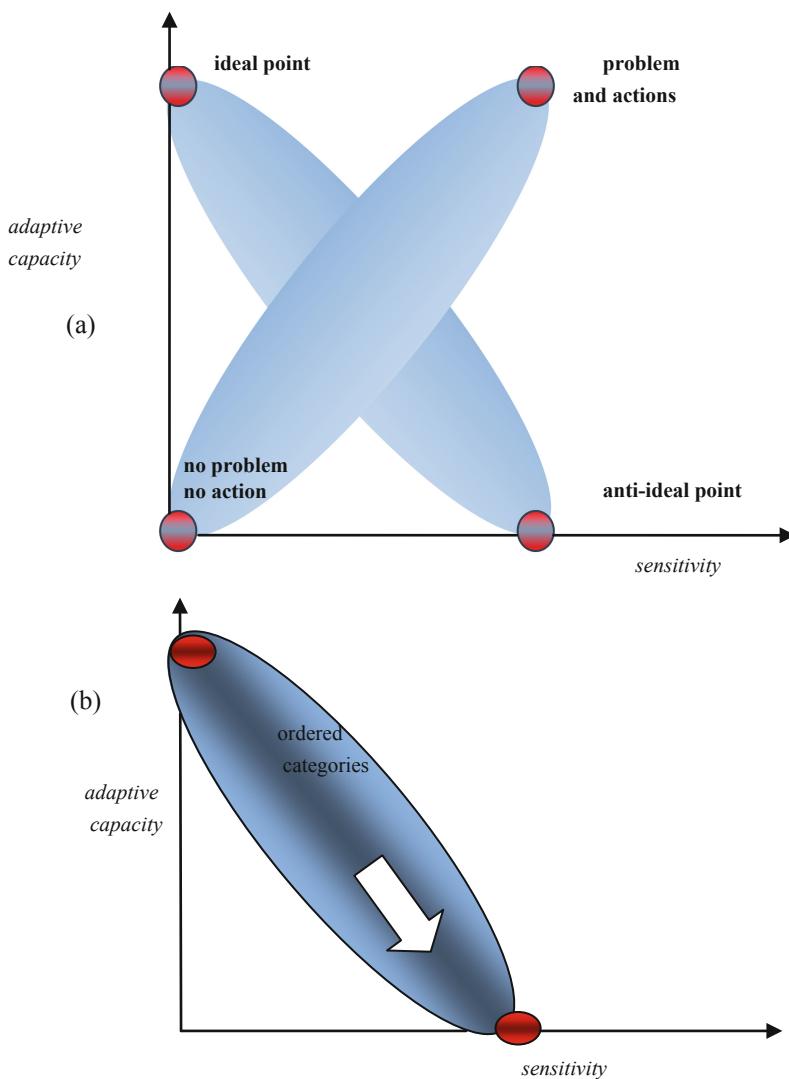
sensitivity and low adaptive capacity); and (4) high sensitivity and high adaptive capacity simultaneously (problem and action).

By applying the fuzzy pattern recognition based method, the evaluation of each alternative (country) works together for all the four categories. By following the fuzzy pattern recognition process, for each country the sum of the four evaluations (one for each category) is equal to one. Another interesting point of view is that the categories are non-ordered since the combination between a positive (adaptive capacity) and a negative (sensitivity) criterion is used.

In general let a classification problem with  $N$  alternatives (here, the number of the examined countries) and  $M$  criteria (in the examined application, only two criteria are used,  $M = 2$ ). The classification problem may be concisely expressed as follows:

$$D = \begin{bmatrix} x_{11} & \dots & x_{1M} \\ \dots & x_{im} & \dots \\ x_{N1} & \dots & x_{NM} \end{bmatrix} \quad (3)$$

where  $D$  is the matrix which contains the score of the criteria with respect to each alternative (here the countries). Hence, where  $x_{im}$  is the score of alternative  $i$  for criterion  $m$ .



**Fig. 1.** Synthesis (a) Synthesis between the sensitivity and the adaptive capacity and definition of the corresponding four (non ordered) categories (b) The usual approximation of ordered categories.

As the elements of the matrix of Eq. 3 normally contain entries of different orders, scales and importance, it is common to convert them to a common base and express in the form of a relative membership degree matrix R by adopting either of the following standardisation procedures depending on the benefit objectives (Xuesen et al. 2009):

$$r_{im} = x_{im} / \max x_{im}, \quad i = 1, \dots, N; \quad m = 1, \dots, M \quad (4)$$

Therefore the initial matrix D is transformed as follows:

$$R = \begin{bmatrix} r_{11} & \dots & r_{1M} \\ \dots & r_{im} & \dots \\ r_{N1} & \dots & r_{NM} \end{bmatrix} \quad (5)$$

Very often the ideal and anti-ideal patterns (here, fictitious countries with ideal and anti-ideal scores) are selected to be (e.g. Zhou et al. 1999):  $G = (1, \dots, 1)$ ,  $B = (0, \dots, 0)$  respectively. In our case, instead of only two points (ideal and anti-ideal points) four categories, and hence four points, are used.

The proposed method has a significant difference compared with the other applied fuzzy multicriteria methods based on pattern recognition (e.g. Zhou et al. 1999; Xuesen et al. 2009), since in the proposed method the synthesis of criteria with different monotony takes place and hence, the evaluation lead to non-ordered categories.

Let us consider the  $k^{th}$  pattern. The evaluation of the criterion  $m$  for the pattern  $k$  is written as  $v_{k,m}$ . The most widely used measure of distance is the following:

$$d_{k,i}(P^k, r_i) = \left[ \sum_{m=1}^M [w_m(r_{i,m} - v_{k,m})]^2 \right]^{1/2} \quad (6)$$

The next critical concept is the membership degree  $\mu_{k,i}$  which indicates the relative membership degree of alternative  $i$  belonging to pattern  $k$ . The membership degree  $\mu_{k,i}$  takes into account the score of each country (alternative  $i$ ) compared with all categories and not only with the examined category. Assuming the membership degree matrix of each country belonging to each category is as follows:  $U = (\mu_{k,i})_{K \times N}$ , where  $\mu_{k,i}$  is the membership degree of country  $i$  belonging to pattern  $k$ . A basic property that must satisfy the matrix U is that the sum of the membership values for each alternative under all patterns (here  $k = 1, \dots, 4$ ) is equal to one (e.g. Shouyu and Guangtao 2003):

$$\sum_{k=1}^K \mu_{k,i} = 1 \quad (7)$$

In general, the methodology of fuzzy sets comprises a mapping from a general set X to the closed interval [0,1] which is described by its membership function (Chrysafis and Papadopoulos 2009; Bardossy et al. 1990). Therefore the above constraint can be easily achieved since the membership function takes values between zero and one.

Finally the membership degree is selected aiming to minimize the following objective function which expresses the sum of the relative distances of the alternatives from the patterns:

$$\min \left( F = \sum_{i=1}^N f(r_i) \right), \quad f(r_i) = \sum_{k=1}^K (\mu_{k,i} \cdot d_{k,i}(P^k, r_i))^2 \quad (8)$$

By using the Lagrange theory of optimization (because of Eq. 7) it is easy to see that the membership degree of belonging of the alternative  $i$  at the pattern  $k$  is equal to:

$$\mu_{k,i} = \frac{1}{\sum_{l=1}^K \left( \frac{d_{k,i}(P^k, r_i)}{d_{l,i}(P^l, r_i)} \right)^2} \quad (9)$$

Only two criteria are initially established in this article, the adaptive capacity and the sensitivity criterion. Furthermore apart from the ideal and anti-ideal patterns, as the TOPSIS method, the use of another two additional categories to describe better the vulnerability to water drought is proposed. From this point, the moderate states are not essential in multicriteria evaluation since they cannot lead to a roadmap. Each category is described by the corresponding pattern as follows (Fig. 1a):

- (a) Pattern of first category: ideal point (0, 1)
- (b) Pattern of second category: anti-ideal point (1, 0)
- (c) Pattern of third category: no problem-no action (0,0)
- (d) Pattern of fourth category: high sensitivity and high adaptive capacity simultaneously (1,1).

In which the first score expresses the degree of sensitivity and the second score the degree of adaptive capacity. Therefore, according to the proposed methodology  $M = 2$  (number of criteria) and  $K = 4$  (number of categories).

An important property of the proposed method is that the sum of the membership values for each country under all the four patterns of the corresponding categories is equal to one.

In this point it should be justified that the characterization of the proposed method as meta-multicriteria method even if weights are used to evaluate the distance between the patterns and the alternatives (countries).

As the other multicriteria methods do, the proposed method, deals with the extreme points but apart from the TOPSIS method (Fig. 1b), the proposed method covers a larger space of the decision space. As the multicriteria methods do, the reference points are given a priori, whilst the fuzzy pattern recognition is used only to classify the countries into the pre-defined categories.

Next, instead of four categories, based on the same key idea, a larger amount of categories can be established. This point is discussed in the case-study section.

### 3 Drought Vulnerability of Countries in the Mediterranean Region

The analysis focuses on the Mediterranean region. The countries examined are: Albania, Algeria, Cyprus, Egypt, Greece, France, Israel, Italy, Morocco, Portugal, Spain, Tunisia, Turkey and Malta. The data of each variable considered is normalised based on the minimum and the maximum evaluation score of each index.

The methodology of fuzzy pattern recognition is implemented to distinguish to which degree each country belongs to the four selected categories. As mentioned before, each category is described by a multicriteria score of the corresponding pattern (it can be seen as a fictitious country). Initially, four selected patterns describe all the extreme combinations between the evaluation of the adaptive capacity and the sensitivity criteria. A basic property that must be verified is that the sum of the membership values for each country under all patterns is equal to one (Eq. 7). The results are shown in Fig. 2.

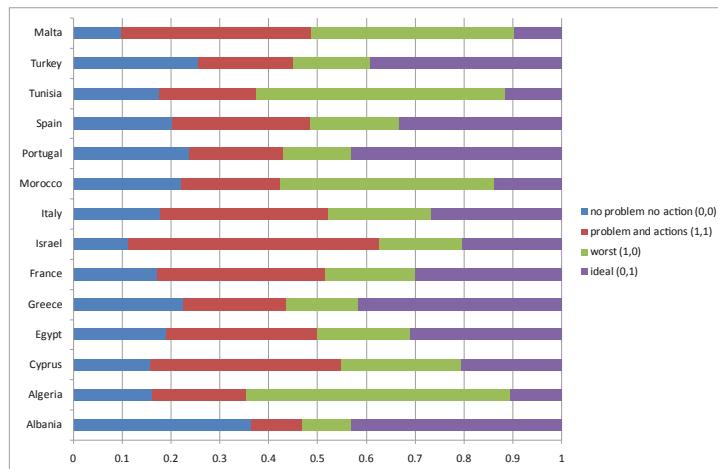


Fig. 2. Categorization based on the fuzzy pattern recognition approach

In this real case study, all Mediterranean countries have a certain degree of vulnerability in each category.

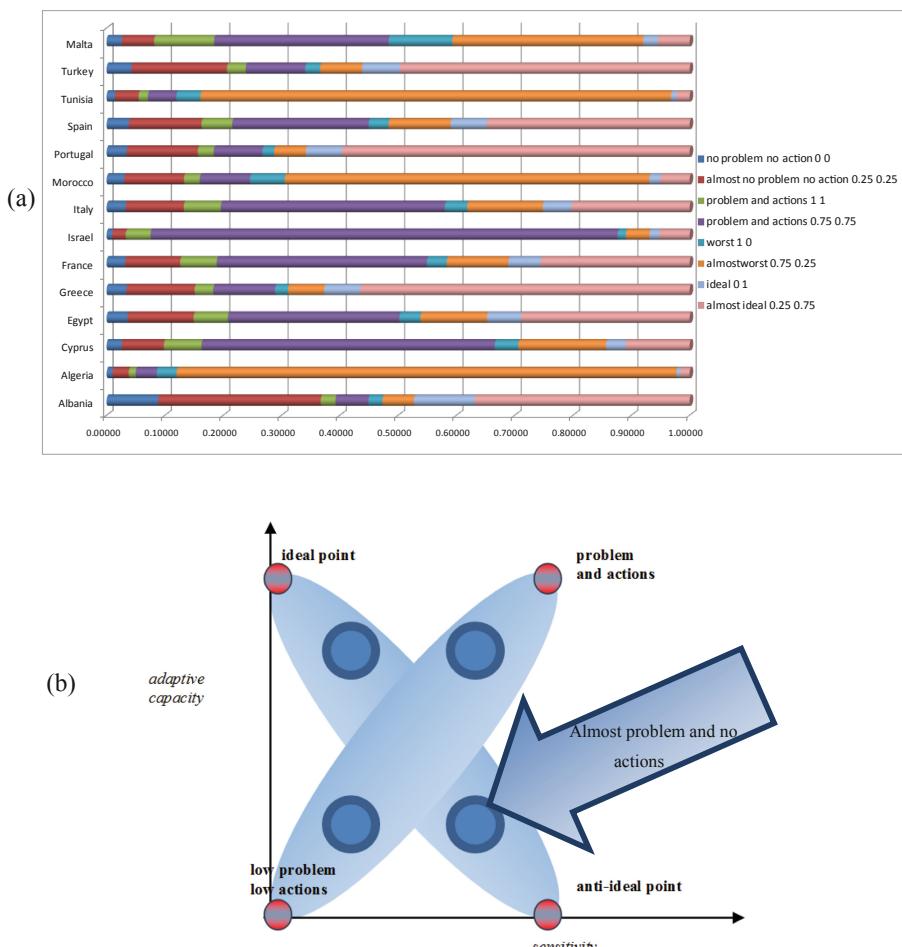
For this reason, the methodology of fuzzy pattern recognition is expanded by adopting a high partitioning of the categories and intermediate categories (Fig. 3). Hence, even the methodology insists on extreme points it recognizes some more absolute states. Consequently, by following a higher partitioning, now in most cases the model produced membership functions with values significantly greater than 0.5 for at least one category for each country and hence the categorization of each country becomes clearer.

This new categorization seems more compatible with the reality. For instance, indeed there are countries as Israel and Cyprus where their vulnerability can be

characterized within the category of “*almost problem and (adaptive) actions* (0.75, 0.75)” whilst there are countries as in the south of the Mediterranean Sea where their vulnerability to drought belongs to the category of “*almost worst (almost problem and no (adaptive) actions)* (0.75, 0.25)”.

A disadvantage of the analysis is that this type of analysis cannot describe the differences between the regions inside each country. For instance, Greece is a typical example that presents both significant differences of the hydrological regime and the concentration of the population. In fact, apart from others, Greece has plenty of surface water potential in most cases far from the water demand centers. These inequalities cannot be described from indicators which cover all the country.

An object for further investigation would be the normalization of the indices, since usually, the greatest value of the samples is used. The influence of this practice can be



**Fig. 3.** (a) Categorization based on the fuzzy pattern recognition approach with high partitioning  
(b) Interpretation of the fuzzy pattern recognition classes with high partitioning.

reduced by using outranking methods, which are based on binary comparisons (e.g. Spiliotis et al. 2015), to evaluate the adaptive capacity and the sensitivity criteria under many indices.

## 4 Conclusions

The main idea of this work is the assessment of the vulnerability to drought based on the synthesis of the sensitivity and the adaptive capacity criteria. Taking into account the extreme values of the above two criteria, initially four non - ordered categories are established to characterize the vulnerability to drought. However a high partitioning leads to a more distinguished classification. Finally the countries are classified based on the synthesis between a positive (adaptive capacity) and a negative (sensitivity) criterion. Since both negative and positive criterion are used, non-ordered categories are produced.

The categorization into the pre selected non-ordered categories is based on fuzzy pattern recognition with multiple patterns (not only the ideal and the anti-ideal pattern as the usual multicriteria methods). The patterns are selected based on the physical problem itself. The method of fuzzy pattern recognition enables us, without significant complexities, to establish more categories to make the picture clearer.

The proposed method can be characterized as meta-monicriteria method since they use not only one multicriteria method but a sophisticated schedule whilst the final evaluation is between non-ordered categories based on the nature of the problem itself. Further work could include means to modulate individually the adaptive capacity and the sensitivity criteria into the multicriteria approach.

## References

- Iglesias, A., Garrote, L., Cancelliere, A., Cubillo, F., Wilhite, D.: Coping with Drought Risk in Agriculture and Water Supply Systems. Drought Management and Policy Development in the Mediterranean, vol. 26, p. 320. Springer, Netherlands (2009). <https://doi.org/10.1007/978-1-4020-9045-5>
- Iglesias, A., Garrote, L., Martín-Carrasco, F.: Drought risk management in Mediterranean river basins. *Integr. Environ. Assess. Manag.* **5**(1), 11–16 (2015)
- Brack, W., Posthuma, L., Hein, M., von der Ohe, P.: European river basins at risk. *Integr. Environ. Assess. Manag.* **5**(1), 2–4 (2009)
- Iglesias, A., Garrote, L., Flores, F., Moneo, M.: Challenges to manage the risk of water scarcity and climate change in the mediterranean. *Water Resour. Manage* **21**(5), 775–788 (2007)
- Naumann, G., Barbosa, P., Garrote, L., Iglesias, A., Vogt, J.: Exploring drought vulnerability in Africa: an indicator based analysis to inform early warning systems. *Hydrol. Earth Syst. Sci. Discuss.* **10**(10), 12217–12254 (2014)
- Nardo, M., Saisana, M., Saltelli, A., Tarantola, S., Hoffman, A., Giovannini, E.: Handbook on Constructing Composite Indicators: Methodology and User Guide, No. 2005/3. OECD publishing (2005)

- Tsakiris, G., Spiliotis, M., Vangelis, H., Tsakiris, P.: Evaluation of measures for combating water shortage based on beneficial and constraining criteria. *Water Resour. Manage* **29**(2), 505–520 (2015)
- Kazakis, N., Spiliotis, M., Voudouris, K., Pliakas, F.K., Papadopoulos, B.: A fuzzy multicriteria categorization of the GALDIT method to assess seawater intrusion vulnerability of coastal aquifers. *Sci. Total Environ.* **593–594**, 552–566 (2018)
- Spiliotis, M., Martín-Carrasco, F., Garrote, L.: A fuzzy multicriteria categorization of water scarcity in complex water resources systems. *Water Resour. Manage* **29**(2), 521–539 (2015)
- Belacel, N., Vincke, P., Scheiff, J.M., Boulassel, M.R.: Acute leukemia diagnosis aid using multicriteria fuzzy assignment methodology. *Comput. Methods Prog. Biomed.* **64**, 145–151 (2001)
- Wu, D., Yan, D.-H., Yang, G.-Y., Wang, X.-G., Xiao, W.-H., Zhang, H.-T.: Assessment on agricultural drought vulnerability in the Yellow River basin based on a fuzzy clustering iterative model. *Nat. Hazards* **67**(2), 919–936 (2013)
- Xuesen, L., Bende, W., Mehrotra, R., Sharma, A., Guoli, W.: Consideration of trends in evaluating inter-basin water transfer alternatives within a fuzzy decision making framework. *Water Resour. Manage* **23**(15), 3207–3220 (2009)
- Zhou, H.C., Wang, G.L., Yang, Q.: A multi-objective fuzzy recognition model for assessing groundwater vulnerability based on the DRASTIC system. *Hydrol. Sci. J.* **44**, 611–618 (1999)
- Shouyu, C., Guangtao, F.: A DRASTIC-based fuzzy pattern recognition methodology for groundwater vulnerability evaluation. *Hydrol. Sci. J.* **48**(2), 211–220 (2003)
- Chrysafis, K.A., Papadopoulos, B.K.: Cost-volume-profit analysis under uncertainty: a model with fuzzy estimators based on confidence intervals. *Int. J. Prod. Res.* **47**(21), 5977–5999 (2009)
- Bardossy, A., Bogardi, I., Duckstein, L.: Fuzzy regression in hydrology. *Water Resour. Res.* **26**(7), 1497–1508 (1990)



# Bioinspired Early Prediction of Earthquakes Inferred by an Evolving Fuzzy Neural Network Paradigm

Mario Malcangi<sup>1</sup>(✉) and Marco Malcangi<sup>2</sup>

<sup>1</sup> Computer Science Department, Università degli Studi di Milano,  
via Celoria 18, 20133 Milan, Italy  
mario.malcangi@unimi.it

<sup>2</sup> Department of Heart Science, Università degli Studi di Milano,  
via Mangiagalli 34, 20133 Milan, Italy  
marco.malcangi@studenti.unimi.it

**Abstract.** Earthquakes could be early predicted as demonstrated by animal's behavior that are able to detect the leading wave part of the seismic wave (the P-wave). P-waves travel faster than S-wave wave (the shaking wave), so they reach the seismic sensors early (tens of seconds to minutes in advance) compared to the P-wave.

A bioinspired framework could be implemented mimicking the animal's behaviour related to the event of an incoming earthquake.

Training a Fuzzy Neural Network to recognize the P-waves, early prediction of earthquakes is feasible and an adequate recovery strategy could be implemented. A technological motivation is the availability of OTS (off-the-shelf) vibration sensors and the fast development of IoT (Internet of Things) toward the new paradigm IoE (Internet of Everything).

**Keywords:** Earthquake · Seismic waves · EFuNN · Biomimetic · Early prediction · Vibration sensor · Internet of Everything

## 1 Introduction

The belief that animals can predict earthquakes has been around for centuries [1]. The most emblematic historical case was recorded on 373 B.C. It was reported by the historians that animals (rats, snakes and weasels) abandoned the Greek town of Helice just days before an earthquake devastated the town. The knowledge of this capability of the animals is well known and related to their fine senses that are able to detect and match very small vibrations, infrasounds and ultrasounds, gases, and other physical signs that anticipate an earthquake event.

Geologists [2] disagree that correlation between earthquakes and animals behavior exists, mainly because physical signs that anticipate the earthquake are not known. Anyway dogs demonstrated to be highly sensible to some earthquake events anticipating signs, maybe vibrations or electrical. An effective application of the use of dogs as early earthquake detector (like illegal drugs sniffer dogs & explosives sniffer dogs) was successfully tested in China. On 1975 Chinese authorities days before a 7.3

magnitude earthquake, evacuated the people from Haicheng town (saving at least 150,000 people (estimated) from injuries and fatalities). The evacuation was successfully based on widespread accounts of unusual animal (dogs) behavior.

The investigation on the animals behavior related to the earthquake events can lead to identify the physical signs that can be detected by an electronic sensor and matched by an Artificial Neural Network (ANN). After motion triggered cameras were been located in the Yanachaga National Park (Perù), researchers observed significant behavior in animals weeks before (three weeks) an earthquake struck the whole region.

Animals shown what is known as “serotonin syndrome” due to unbalanced positive ions in the environment. Serotonin regulates the mood in the animals and the humans. Positive ions concentration increases in the environment when rocks in the ground crust are stressed during the build-up of an earthquake. So, more positive ions concentration produces more serotonin in the animals and more excitation that induce them to move to positive ions low concentration area (down to the valleys from the hills). If ionization sensors could be available as a commodity, then the increasing of environment’s ionization could be monitored and used to trigger an earthquake early warning system.

An environment sign that confirms the ionization-theory is the observation of lighting preceding the earthquake event. Such lighting were been observed since ancient when strong earthquakes happened. The most recent was been reported on 2009 when a strong earthquake devasted L’Aquila (a town at the center of Italy). Just few seconds before the earthquake’s hit, people saw ten-centimeter flames of light flickering above a stone street.

On November 12th, 1988, was been reported a bright purple-pink light along the St. Lawrence River in Quebec, 11 days before the powerful quake happened.

People reported similar observations of lights before the great 1906 quake in San Francisco.

Only 0.5% of earthquakes create the conditions for lightning in a limited geographical areas, but Freund [3] and other scientists are working on an earthquake forecasting system that includes earthquake lights among the indicators.

Lightning can be observed and ionization can be sensed, but unfortunately such sensors don’t exist as electronic commodities, so no application of the ionization theory exists, apart the use of the animals like a sensor, looking to their mood. No lightning prediction was also been deployed.

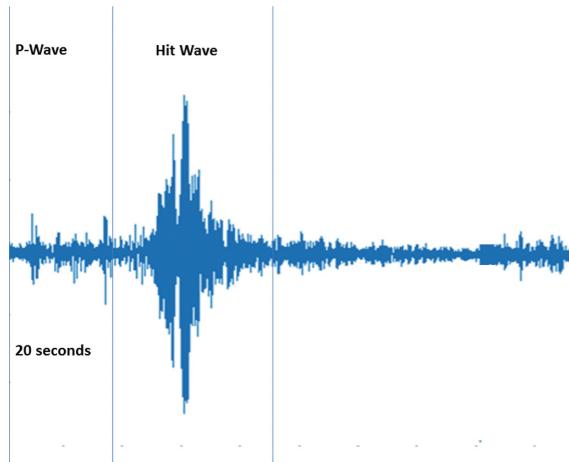
An alternative hypothesis could be that most of animals are sensible mainly to small vibrations and that they are able to detect and to interpret such signs as an incoming danger.

There are two types of seismic waves: body waves (Fig. 1) and surface waves. Body waves travel deep in the ground and surface waves travel at the surface level of the ground. Body waves travel faster than surface waves and reach us early. Body waves are composed of two different waves the p-wave and the s-wave. The p-wave (primary wave) is compressional because it causes vibrations parallel to its direction. The s-wave is shacking because causes vibrations orthogonal to its direction. P-wave is faster than other weaves (2–5 km/s). The P-wave do not includes the hit (maximum intensity) vibration and it arrives seconds to minutes in advance. Animals sense small

vibration by their ears, feet and whiskers, then match and learn about the associated risk evolving their learning along the time.

A reasonable bio-inspired solution to early prediction of earthquakes could be based on electronic vibration sensors as sensing and detecting devices, and on applying the Evolving Fuzzy Neural Network (EFuNN) paradigm [4–6, 12] to learn on line and to match the small vibrations that lead the disruptive wave.

There are several technological motivations that validate the bio-inspired approach to the development of a system for early detection of earthquake. One is that vibrational sensors are devices Commercially Of-The-Shelf (COTS) available in volumes (MEMS accelerometers) and are smart (embeds a MicroController Unit (MCU), capable to run advanced pattern matching paradigms). Because Internet is evolving toward the IoE (Internet of Everything) networking paradigm, massive displacement and networking of smart sensors will be available to deploy an effective early warning service.



**Fig. 1.** Body waves are seismic waves composed of a P-wave and S-wave: P-wave is compressional, S-wave is shaking and includes the hit.

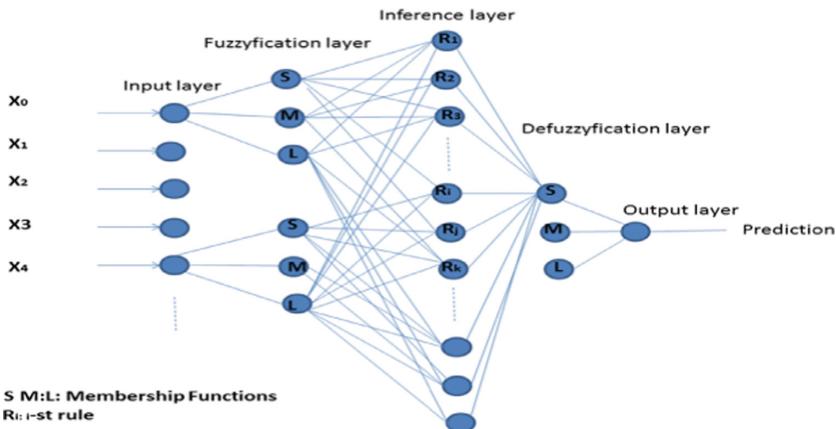
## 2 What Is EFuNN?

EFuNN is an Evolving Fuzzy Neural Network paradigm, an implementation of the ECOS (Evolving COnnectionist System) paradigm [10]. It enables on-line adaptation and evolves incrementally and adaptively in real-time. EFuNN is a connectionist paradigm based on fuzzy rules and a fuzzy inference engine.

EFuNN [11] consists of a 5 layer architecture (Fig. 2). The layers are those required by a fuzzy logic engine. The first layer inputs the crisp values. The second layer fuzzyfies the crisp values inputted at the second layer. The third layer infers by rules generating fuzzy data to be defuzzified by the forth layer. The fourth layer executes the defuzzification of the output data applying a weighted function and a saturated linear activation function outputting crisp control values.

The peculiarity of the EFuNN is that the five layers fuzzy architecture corresponds to a five layers artificial neural network (ANN) architecture. The ANN's learning capabilities can be applied to set up the fuzzy logic engine's knowledge as nodes of the ANN. Nodes evolve by learning. Rules are nodes of the ANN.

Two important capabilities are embedded in EFuNN paradigm: the fuzzy logic's capability to infer by rules and the ANN's capability to learn by data. This paradigm is the best strategy to challenge the key task of the fuzzy logic (the knowledge set up) according to a bio-inspired approach.



**Fig. 2.** EfuNN is a five layers artificial neural network where each layer corresponds to a layer of a fuzzy logic engine.

### 3 Data Set and Training

A set of p-waves patterns were been extracted from earthquake recordings done with a seismographer lacated at the Centro Geofisico Prealpino – Varese – Italy [7].

To build up the dataset to train and test the EFuNN, sampled data of the p-waves patterns were been extracted and labeled using the SeisGram2 k visualization and analysis software [8] and a Matlab script to assemble the data according to the formats requested by the Neucom simulation and modeling environment [9] running the EFuNN paradigm.

The Matlab script receive as input the full seismogram sampled data stream and window three types patterns:

- Background noise
- P-wave
- Hit-wave

After patterns windowing, the script executes the labeling and formatting as it follow:

Pattern 1:  $S_1, S_2, S_3, S_4, \dots, S_j$ , Label1

Pattern 2:  $S_1, S_2, S_3, S_4, \dots, S_j$ , Label2

Pattern 3:  $S_1, S_2, S_3, S_4, \dots, S_j$ , Label3

j: j-th sample

## 4 Training and Test

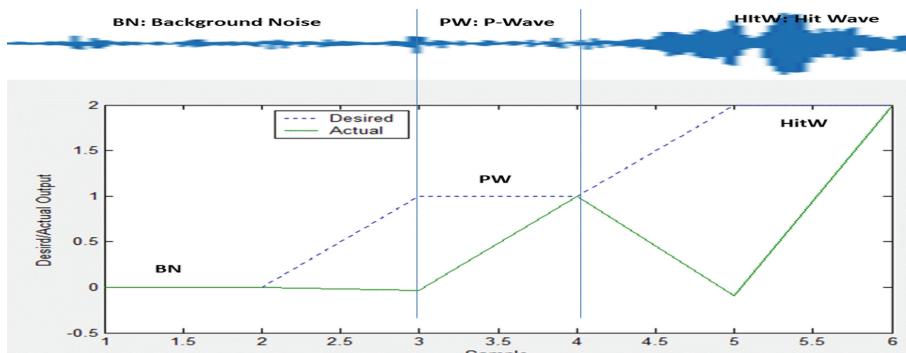
To train the EFuNN the p-waves-based dataset has been applied to the EFuNN. After training, the test was executed to validate the EFuNN capability to match the p-wave pattern preceding an earthquake event.

The data set was split randomly in two parts (80% and 20%). 80% of the data set was been applied to train the EFuNN, 20% of the data set has been applied to test the EFuNN.

Train and test setup was as follows:

- Sensitivity threshold: 0.9
- Error Threshold: 0.1
- Number of membership functions: 3
- Learning rate for W1: 0.1
- Learning rate for W2: 0.1
- Node age: 60
- Maximum field: 0.5

Then a new test, a sequence data stream (background noise - p-wave, hit-wave) was been built so the EFuNN was been tested on a full earthquake wave sequence. This test (Fig. 3) demonstrated that the EFuNN can infer run-time on an incoming earthquake event triggering the alert system early before of the hit-wave (the disruptive one) arrival.



**Fig. 3.** After the training, the EFuNN at test-time demonstrated to be able to recognize the incoming P-wave before the Hit-wave arrival.

According to the test results, after a background noise sequence (BN), the P-wave pattern (PW) was detected and matched before the Hit-Wave (HitW) arrived (also the Hit-Wave is detected and matched).

## 5 Results Evaluation and Future Developments

This research demonstrated that a bio-inspired approach to the development of an early prediction system is feasible applying bio-inspired Artificial Neural Networks (ANNs) paradigm to predict the earthquake events seconds to minutes in advance.

**Better performance could be achieved if the evolving and on-line learning of the EFuNN paradigm is applied more extensively.**

A future development will concern the study of the seismologic waves captured by the MEMS (Micro-Electro Mechanical Systems) vibration sensors to finalize a personal early warning earthquake system and a IOT/IOE (Internet of Thing/Internet of Every Thing) networking to apply swarm computing methods and brain inspired ANN paradigms [9, 13].

We consider that pattern matching strategy only is no fully adequate to successfully accomplish the earthquake early detection task, because each earthquake event differs from others due to different geographical pathways, intensity and geophysical formation. So, more investigation is required to setup a successful framework for early prediction of earthquake, considering that bio inspiration is necessary to accelerate the knowledge discovery, but it is not sufficient to copy the nature, because in the nature evolution strategy the failure is not a problem (it is part of the process), so the same paradigm could not be applied to human being.

**Acknowledgements.** Thanks are due to Dr. Paolo Valisa (Centro Geofisico Prealpino – Varese –Italy) that enabled us to collect seismographic data at the Centro Geofisico Prealpino – Varese - Italy.

A special acknowledgment is due to Prof. Nikola Kasabov, Auckland University of Technology, Director KEDRI – Knowledge Engineering and Discovery Research Institute, for his invaluable suggestions on how to get the most from the EFuNN’s evolving capabilities.

## References

1. Mott, M.: Can Animals Sense Earthquakes. National Geographics, 11 November 2003
2. Gupta, R.P.: Remote Sensing Geology. Springer, Heidelberg (2018). <https://doi.org/10.1007/978-3-662-55876-8>
3. Freund, F.T., Derr, J.S.: Prevalence of earthquake lights associated with rift environments. *Seismol. Res. Lett.* **85**(1), 159–178 (2014)
4. Alves, E.I.: Earthquake forecasting using neural networks: results and future work. *Nonlinear Dyn.* **44**, 341–349 (2006)
5. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Kluwer Academic Publishers, New York (1981)
6. Kasabov, N., Kim, J.S., Watts, M., Gray, A.: FuNN/2-a fuzzy neural network architecture for adaptive learning and knowledge acquisition. *Inf. Sci. Appl.* **101**(3–4), 155–175 (1997)

7. <https://www.astrogeo.va.it/sismologia/sismi.php>
8. <http://alomax.free.fr/seisgram/SeisGram2K.html>
9. <http://www.kedri.aut.ac.nz/areas-of-expertise/data-mining-and-decision-support-systems/neucom>
10. Kasabov, N.: Evolving Connectionist Systems: The Knowledge Engineering Approach. Springer, Heidelberg (2007). <https://doi.org/10.1007/978-1-84628-347-5>
11. Kasabov, N.: EFuNN. IEEE Tr SMC (2001)
12. Kasabov, N.: Evolving fuzzy neural networks – algorithms, applications and biological motivation. In: Yamakawa, T., Matsumoto, G. (eds.) Methodologies for the Conception, Design and Application of the Soft Computing, World Computing, pp. 271–274 (1998)
13. Kasabov, N.K.: Time-Space, Spiking Neural Networks and Brain-Inspired Artificial Intelligence. SSBN, vol. 7. Springer, Heidelberg (2019). <https://doi.org/10.1007/978-3-662-57715-8>



# Enhancing Disaster Response for Hazardous Materials Using Emerging Technologies: The Role of AI and a Research Agenda

Jaziar Radianti<sup>1</sup>, Ioannis Dokas<sup>2(✉)</sup>, Kees Boersma<sup>3</sup>,  
Nadia Saad Noori<sup>4</sup>, Nabil Belbachir<sup>4</sup>, and Stefan Stieglitz<sup>5</sup>

<sup>1</sup> University of Agder, Kristiansand, Norway  
jaziar.radianti@uia.no

<sup>2</sup> Demokritus University of Thrace, Xanthi, Greece  
idokas@civil.duth.gr

<sup>3</sup> Vrije University, Amsterdam, The Netherlands  
f. k. boersma@vu.nl

<sup>4</sup> NORCE Research, Oslo, Norway  
{nano, nabe}@norceresearch.no

<sup>5</sup> University of Duisburg Essen, Duisburg, Germany  
stefan.stieglitz@uni-due.de

**Abstract.** Despite all efforts like the introduction of new training methods and personal protective equipment, the need to reduce the number of First Responders (FRs) fatalities and injuries remains. Reports show that advances in technology have not yet resulted in protecting FRs from injuries, health impacts, and odorless toxic gases effectively. Currently, there are emerging technologies that can be exploited and applied in emergency management settings to improve FRs protection. The aim of this paper is threefold: First, to conduct scenario analysis and situations that currently threaten the first responders. Second, to conduct gap analysis concerning the new technology needs in relation to the proposed scenarios. Third, to propose a research agenda and to discuss the role of Artificial Intelligence within it.

**Keywords:** Hazardous materials · Emerging technologies · Emergency management · First responders

## 1 Introduction

First Responders (FRs) have one of the deadliest jobs in the world since they operate very close to unpredicted dangers, such as a sudden gas explosion, release of deadly chemicals, building collapse and heart attacks. As a result, several FRs are injured or lose their lives in action due to the lack of awareness of unpredicted risks, of hazardous materials and material exposure. FR fatalities compose more than half of the total fatalities of the incident in some cases. For example, the 2007 forest fires in Artemida, Greece resulted in 26 fatalities 3 of which were firefighters (11%), the 2013 West Fertilizer Company incident in Texas, USA resulted in 15 victims where the FR victims accounted for 53% fatalities. In the 2015 chemical blast in Tianjin-China, there were

173 fatalities, where 104 of them or 60% were FRs [1]. Likewise, in the event of a high-rise building fire in Tehran 2017, 16 out of 22 fatalities or 72% were FRs. This reveals how far FRs are exposed to deadly risks in response operations.

Situational Awareness (SA) also has a significant impact on the incident management and coordination, where it is critical for “all knowledge that is accessible and can be integrated into a coherent picture, when required, to assess and cope with a situation” [2]. Today, the integration of Information and Communication Technology (ICT) and mobile technologies in emergency management is reshaping communications and information exchange between the command and control centres (C2C) and FRs on the incident site. Also, ICT has provided several opportunities for advanced-sensing, computing and communicating through smartphones, wearable-portable devices, robotics and unmanned aerial vehicles (UAVs).

However, reports show that advances in technology have not yet resulted in protecting FRs from injuries, health impacts, and odorless toxic gases effectively [3]. For example, a recent study reveals that the mean number of firefighters’ fatalities in Sweden has increased from 2000 to 2016 [4]. Indeed, the International Forum to Advance First Responder Innovation’s [5] has published a list of four FR capability gaps, namely the ability to: (a) Know the location of responders and their proximity to risks and hazards in real time. (b) Detect, monitor and analyze passive and active threats and hazards at incident scenes in real time. (c) Rapidly identify hazardous agents and contaminants. (d) Incorporate information from multiple and nontraditional sources (e.g., crowdsourcing and social media) into incident command.

Briefly, we see some capability gaps concerning the identification of hazardous agents and detecting, monitoring and analyzing passive and active hazard. For example, FR personnel often need to be close enough to the sources before realizing the presence of a passive and active hazard. In short, there is a strong need to innovate with new technologies for first responders to address their capability gaps. Sometimes, the issues not only about emerging technologies but also methods how to use existing technologies efficiently so that FRs are protected. Here, often a common operational picture and situational awareness play a role, and technologies can enhance these situations.

The aim of this paper is threefold: First, to conduct scenario analysis and situations that may put FRs in different types of threats. Second, to conduct gap analysis concerning the new technology needs and specifications in relation to the proposed scenarios. Third, to propose a research agenda and to discuss the role of Artificial Intelligence (AI).

This paper is organized into five sections. Section 2 describes the theoretical background on the importance of common operational picture and situational awareness. Section 3 describes three scenarios where the hazardous materials can come into the picture and be unexpected extra disasters. Section 4 elaborate examples of potentially relevant, emerging technologies that can be scrutinized further for protecting the FRs. Section 5 is a proposed Research Agenda. Section 6 is the concluding remark.

## 2 Theoretical Background

To understand the risks and threats exposed to the first responders, literature in emergency management has emphasized the importance of the three following perspectives. First, Common Operational Picture (COP). Second, Shared Situation Awareness (SA), and third, Collective Sensemaking and Advanced Decision Making.

The COP is a way to ‘achieve a sufficient level of shared information among the different organizations and jurisdictions participating in disaster operations at different locations, so all actors readily understand the constraints on each and the possible combinations of collaboration and support among them under a given set of conditions’ [6]. SA is a precondition of any COP. It is the perception of environmental elements and events concerning time or space, the comprehension of their meaning, and the projection of their status after some variable has changed, such as time, or some other variable, such as a predetermined event. SA involves being aware of what is happening at times of uncertainty to understand how information, events, and one’s actions will impact goals and objectives, both immediately and in the near future [7]. One with an adept sense of situation awareness generally has a high degree of knowledge with respect to inputs and outputs of a system, an innate “feel” for situations, people, and events that play out because of variables the subject can control. Lacking or inadequate situation awareness among responders with different backgrounds (i.e., shared situation awareness) has been identified as one of the primary factors in accidents attributed to human error. SA is related to Advanced Decision Making, which is the process followed in processing information to assess situations for making collective decisions and taking collaborative actions. Decision making is part of all management tasks and that it is particularly important for emergency managers as they often need to take decisions quickly on very inadequate information [8, 9]. Collaborative group decision making plays an important part in first response operations. We see collaborative interactions through the construct of heedful interrelating, which refers to interacting with sensitivity to the task at hand while at the same time paying attention to how one’s actions affect overall group functioning [10–12]. Advanced decision making is related to the process of collaborative sensemaking.

Collective sensemaking is about building a sufficient level of shared understanding in a sensemaking process in which members of different professional organizations (de/re)construct information influenced by their institutional background to find out what is going on in times of uncertainty [13]. This sensemaking process is based upon the knowledge responders have gained through (1) education, including training/exercises; (2) storytelling; and (3) past experiences [14, 15]. FRs have to constantly make sense of the situation and of the actions of other FRs (collectively) because of the rapidly changing environment [16]. Sensemaking can be understood as a steady process [17] of gaining knowledge through the transformation and integration of new information into cognitive schemata [18], which is particularly essential in crises to understand such unstable situations and to make adequate decisions. Therefore, it aims to reduce and bridge knowledge gaps [19–21] and can be impacted by several social factors like opinions [19, 20] as well as interactions, discussions or information from other individuals [13] to find common ground for decision making.

For the first responder to have/get an adequate overview of the crisis situation and the actions of (other) responding organizations, they need to create a common operational picture (COP). Information/data is needed to come to a coherent but dynamic COP. In practice (in line with the literature) this happens only if they have sufficient situational awareness (SA). SA is based on the collecting and sharing of information. SA is not static (as the crisis situation continuously evolves and more responding organizations become active), nor univocal or unambiguous. On the contrary, SA is ambiguous and multivocal, because responding organizations will give (different) meanings to the crisis situation, to the information and to what is needed to respond to the crisis (i.e., decision making). A concept of collective sensemaking approach to understand and unravel the multivocality and ambiguity, to understand how various responders with their specific professional norms and routines interpret the situation, and finally how they enact the responding practices in such a way (that is, collectively) that the joint effort and operations become more efficient.

In the next section, we provide examples of the most frequent man-made and natural disasters involving hazardous substance. The aim is to illustrate that people can easily lose their situational awareness in such events and different understanding of the situations. Moreover, people often don't know that hazardous material leaked into, e.g., burning building, or even into water systems, where the danger is not only involving the first responders alone but also society in general.

### 3 Scenario Analysis

Disastrous events concerning hazardous materials can occur in different settings. Three scenarios presented below can serve as an illustration, why new technologies are required. We select three examples of scenarios, i.e., industrial accidents, natural hazard, and terrorist attack.

#### 3.1 Industrial Accidents

Industrial accidents involving the release of dangerous substances, explosions or fire frequently occur in Europe, causing severe adverse effects to people, properties and the environment. Major toxic spills from mining activities in Europe occurred such as Baia Mare, Romania (2000), Aude/Malvesi, France (2004), Kolontár, Hungary (2010) – all due to dam failures, and in Borsa, Romania (2005) – due to the accidental release of 300 m<sup>3</sup> of cyanide solution into a river. Likewise, leakage of pipelines in a process plant, followed by fire and explosion happened such as in Priolo Gargallo, Italy (2006), explosion and fire rupture of pipeline in Dormagen Germany (2008).

One of the threats to the FRs when implementing their duties in industrial accidents are these unknown hazardous materials (hazmat). Hazmat is chemical substances that if released or misused can pose a threat to property, the environment or health. Such chemicals are prevalent in many industries and products which often use materials that may be explosive, flammable, corrosive, poisonous, radioactive, or otherwise toxic or dangerous. Depending on the nature of the hazardous material, the result of a release or spill can include death, serious injury, long-lasting health effects, or physical damages

of buildings and environment [22]. It can cause fatalities and injuries of workers, damage to property and infrastructure on site and in the surrounding area, critical service disruptions, contamination.

### 3.2 Natural Hazard Event

Flooding is the primary risk faced by many countries. Flood events frequently occur across the multiple countries in the form of a river, flash and water surface floods, and coastal flooding. Flooding understood in broad terms to include floods from rivers, mountain torrents, and floods from the sea in coastal areas, but exclude floods from sewage systems, as defined under the European Floods Directive. Several major flood events have occurred.

For instance, in 2017 in Greece, according to the official data from the Hellenic Civil Protection Command Center 70 floods events were being reported that needed the deployment of First Responders. Other major floods events occurred in the Former Yugoslav Republic of Macedonia and Albania (2015, 2016), Serbia and Croatia (2014), Slovenia (2014), Sardinia and Slovakia (2013) and Austria, Germany and Czech Republic (2013), Norway (2017). Floods often lead into cascading effects such as disruption of critical services and infrastructure, the outbreak of epidemic or epizootic events and damage to industrial facilities causing the release of chemical, biological, radiological and nuclear substance [23].

### 3.3 Terrorist Attack Event

The fear of terrorism is one of the risks in Europe that deeply rooted in people's mind. Over the past few years, Europe has been struck by several major terrorist attacks such as in Madrid, Spain 2004, in Oslo, Norway, 2011 and in Paris, France, 2015 that created mass panics. According to Europol [24], in 2016 a total of 142 failed, foiled and completed attacks were reported and resulted in 142 victims died, and 379 people were injured in Europe. Among the techniques that are often used in these terrorist attacks are Home-Made Explosives, improvised explosive device attacks on soft targets and the use of suicide person-borne. The use of Chemical, biological, radiological and nuclear, food contaminants and radioactive materials are examples of techniques that are feared.

A serious terrorist attack can have severe impacts including mortality, injury, psychological distress, economic losses, and critical infrastructure damages. From the medical emergency services perspective, there is a risk of a high number of casualties such as wounds, burns or even mass casualties in such an event. One of the usages of social media technology is to help the emergency medical services for faster response by spotting victim location and severity of their injuries. A terrorist attack can also be related to the other types of risks such as epidemics, pandemics, CBRN threat and bio-terrorism against facilities with a hazardous substance which again will expose local citizens and environment with substance release that can be poisonous, in addition to the critical infrastructure collapse, damage of property and infrastructure on site and in the surrounding area, critical service disruptions and contamination.

In these three examples of scenarios, the first responders should react quickly, minimizing the loss of lives of the affected people and themselves. If hazardous materials are involved in these examples of disasters, they may be severely exposed to e.g., poisonous gasses, explosive and other dangerous substance. We have high hope that existing and emerging technologies can help the first responders detecting early all potential hazards and have better preparedness in a disaster. The next section describes the example of emerging technologies that potentially are useful for emergency management.

## 4 Emerging Technologies

Currently, there are emerging technologies that can be exploited and applied to improve the way how FRs can be protected, such as:

- Unmanned Aerial Vehicles (UAV) with various mounted sensors and cameras: currently researchers examined different levels of the autonomy UAV from fully controlled by operators, computer action alternatives, computer narrow down the choice, the computer executes an action upon the operator's approval to fully controlled by computer and ignoring the operators [25].
- Wearable devices and wearable sensors (Wrist-worn, head-mounted and others) that come as existing products and research prototype. For example:
  - Wrist-worn: smartwatches and wrist bands with or without touchscreen display such as Apple iWatch, Samsung Gear, Pebble Time, Fitbit flex existing products and Smart-watch Life-Saver (prototype).
  - Head-mounted devices: smart glasses such as Funiki Ambient glasses, Recon Jet, Microsoft HoloLens (existing products) and Google glass (prototype),
  - Smart jewelry designed a for health-monitoring such as a smart ring (existing product) or other jewelry such as typing ring and gesture detection ring (prototype)
  - Electronic garments, i.e., clothing items that also serve as wearables such as Athos, Spinovo (existing products) or Dooplesleep (prototype)
  - E-Patch, i.e., sensor patches that can adhere to the skin for fitness tracking or haptic applications, sensing and data transmission. For example, health patch, Motorola e-tattoo or stamp platform (existing products) or duoskin, smart tooth patch (prototype) [26].
- *AR/VR technologies*: Currently, different technologies have been available to support the immersive experience with virtual reality and augmented reality such as head-mounted display Oculus rift and HTC Vive. Also, some AR/VR experience can be obtained by using smartphones such as Google Cardboard and the Galaxy Gear VR headset. In addition, immersive video and 360-degree video add additional VR/AR experience possibilities [27].
- *Robotics*: Nowadays, the technology and application areas have been developed rapidly for industrial purposes, rehabilitation and surgery, search and rescue, self-driving vehicles, assistive technology, home care, manufacturing and so on. Examples of care robots: Lifting, exoskeletons, assistive, companion, talking, emotional,

service [28] Research on swarm robotics have focused on several topics such as Aerial manipulation, counter-swarm pursuit, target search, and tracking, surveillance monitoring and mapping [29]. There are more examples indicating that the robotics is increasingly becoming an attractive research area with useful new applications.

- *Real-time systems*: Today's robust networks allowed the researcher to put "real-time" as a feature or a selling point of any newly created systems in any area. Countless technologies are offering real-time systems as a part of the delivery.
- *Smartphone* with advanced computing power, connectivity, battery, and storage have changed this device into a handy multi-purpose device that can be used for collecting images, videos, audio, location and other sensor data [30, 31].
- *Surveillance camera* with automatic detections (behavioral and facial recognition, object detections, object tracking and so on). Surveillance camera itself is not new, but how people use and process the videos and images have improved significantly, especially after the advancement of image processing techniques developed in artificial intelligence domain.
- *Super-computers* i.e., high level of performance computers that allow processing very large databases and conduct a big amount of computations, such as iDataPlex, Shaheen II, Hazel Hen and Trinity. They have 301,056 cores, 185,088 cores, 196,608 cores and 65,320 cores respectively [32]. Advances such as multi-core processors and GPGPUs (General Purpose Graphics Processing Units) have enabled a powerful machine for personal use. Supercomputers will continuously support the advancements of activities dealing with UAV, AR/VR, robotics and so on as more data have been collected and need huge computing power to deliver results in nearly real time.

These technologies generate new types of massive data. Also, some technologies have been exploited by the public to generate citizen information, especially social media, and the use of various mobile and web app that allows researchers to collect data through crowd-sourcing technique. Likewise, various research benefited from current fast development of data analysis techniques, especially artificial intelligence especially machine learning, deep learning, and computer vision.

## 5 Discussion: Research Agenda and the Role of AI

Despite all efforts like the introduction of new training methods and personal protective equipment, the need to reduce the number of FRs fatalities and injuries, to as much as reasonably possible level, remains. New emerging technologies can be used to minimize this problem. However, these technologies must be analyzed (re)designed, studied, tested in emergency management settings and evaluated, considering FRs protection against multiple and unexpected dangers and to what extent they can truly enhance SA and COP.

Therefore, we plan to conduct research in the following areas:

- (1) On-Site Threat Detection (real-time portable platform for hazmat detection and monitoring)
- (2) Risk Monitoring and Safe Management of Threats

- (3) Threat Prevention (real-time monitoring of personnel's psychological status, big data analytics and dispersion-prediction projection of hazardous substance).

AI technologies have a very important role to play in this research agenda. AI can be utilized to retrieve useful types of data from different sources and to generate useful information. To enhance social media data analytics services to be utilized by FRs to enhance their SA dynamically. Can enable FRs to manage and respond, with useful answers drawn from messy, real-world datasets, to a large number of emergency calls by people that ask specific, targeted questions. AI can also be used to enhance predictive analytics programs and through that to enhance the readiness level of FR teams. Lastly AI can be used for the development of advanced chemical sensors that can identify in real time an unlimited number of chemicals in real-world environments. Such a sensor in a wearable edition can provide real time threat detection to the FRs teams that operate in the field.

## 6 Concluding Remarks and Future Works

There are some gaps when it comes to the technologies that can protect the FRs from hazardous materials, especially the identification of hazardous agents and detecting, monitoring and analyzing passive and active hazard. We found that there are various new, promising research directions, exploiting new technologies to improve the safety of the FRs. In addition AI technologies can be used in conjunction with other emerging technologies to provide enhanced COP, SA and enhanced threat detection and protect FRs by minimizing their casualties. Our future directions are to pursue and operationalize our research agenda into a set of concrete studies that can contribute to the area of emergency management, safety and risk management.

## References

1. McGarry, S.L., et al.: Preventing the preventable: the 2015 Tianjin explosions, in Harvard T. H Chan School of Public Health, Hongkong jockey Club Disaster Preparedness and Response Institute, Hongkong (2017)
2. Sarter, N.B., Woods, D.D.: Situation awareness: a critical but ill-defined phenomenon. *Int. J. Aviat. Psychol.* **1**(1), 45–57 (1991)
3. Hall, A.H.: A Survey of Firefighter Cancers and Other Chronic Diseases: Preliminary Results (2016)
4. Svensson, S.: Firefighter fatalities in Sweden, 1937–2016. *Brandteknik Lunds tekniska högskola*, Lund (2017)
5. IFAFRI: Capability Gap 2 “Deep Dive” Analysis Synopsis (2017)
6. Comfort, L.K.: Crisis management in hindsight: cognition, communication, coordination, and control. *Public Adm. Rev.* **67**, 189–197 (2007)
7. Endsley, M.R.: Designing for Situation Awareness: An Approach to User-Centered Design. CRC Press (2016)
8. Klein, G.: The recognition-primed decision (RPD) model: looking back, looking forward. In: *Naturalistic Decision Making*, pp. 285–292 (1997)
9. Zsambok, C.E., Klein, G.: *Naturalistic Decision Making*. Psychology Press (2014)

10. Weick, K.E., Roberts, K.H.: Collective mind in organizations: heedful interrelating on flight decks. In: *Administrative Science Quarterly*, pp. 357–381 (1993)
11. Druskat, V.U., Pescosolido, A.T.: The content of effective teamwork mental models in self-managing teams: ownership, learning and heedful interrelating. *Hum. Relat.* **55**(3), 283–314 (2002)
12. Weber, K., Glynn, M.A.: Making sense with institutions: context, thought and action in Karl Weick's theory. *Organ. Stud.* **27**(11), 1639–1660 (2006)
13. Weick, K.E., Sutcliffe, K.M., Obstfeld, D.: Organizing and the process of sensemaking. *Organ. Sci.* **16**(4), 409–421 (2005)
14. Endsley, M.R.: Toward a theory of situation awareness in dynamic systems. *Hum. Factors* **37**(1), 32–64 (1995)
15. Taber, N., Plumb, D., Jolemore, S.: “Grey” areas and “organized chaos” in emergency response. *J. Workplace Learn.* **20**(4), 272–285 (2008)
16. Wolbers, J., Boersma, K.: The common operational picture as collective sensemaking. *J. Conting. Crisis Manag.* **21**(4), 186–199 (2013)
17. Mirbabaei, M., Zapatka, E.: Sensemaking in Social Media Crisis Communication—A Case Study on the Brussels Bombings in 2016 (2017)
18. Pentina, I., Tarafdar, M.: From “information” to “knowing”: exploring the role of social media in contemporary news consumption. *Comput. Hum. Behav.* **35**, 211–223 (2014)
19. Dervin, B.: Sense-making theory and practice: an overview of user interests in knowledge seeking and use. *J. Knowl. Manag.* **2**(2), 36–46 (1998)
20. Savolainen, R.: The sense-making theory: reviewing the interests of a user-centered approach to information seeking and use. *Inf. Process. Manag.* **29**(1), 13–28 (1993)
21. Stieglitz, S., et al.: Sensemaking and communication roles in social media crisis communication (2017)
22. Rabjohn, A.: The human cost of being a ‘first responder’. *J. Bus. Cont. Emerg. Plann.* **6**(3), 268–271 (2013)
23. SWD: Overview of Natural and Man-made Disaster Risks the European Union may face, in 176 Commission Staff Working Document, Brussels (2017)
24. Europol: EU Terrorism Situation and Trend Report (TE-SAT) (2017)
25. Atyabi, A., MahmoudZadeh, S., Nefti-Meziani, S.: Current advancements on autonomous mission planning and management systems: an AUV and UAV perspective. *Ann. Rev. Control* **46**, 196–215 (2018)
26. Seneviratne, S., et al.: A survey of wearable devices and challenges. *IEEE Commun. Surv. Tutor.* **19**(4), 2573–2620 (2017)
27. Garcia-Luna-Aceves, J.J.H., Westphal, C.D.: Network support for AR/VR and immersive video application: a survey. In: Proceedings of the 15th International Joint Conference on e-Business and Telecommunications, ICETE 2018, vol. 1 (2018)
28. Rantanen, T., et al.: The adoption of care robots in home care—a survey on the attitudes of Finnish home care personnel. *J. Clin. Nurs.* **27**(9–10), 1846–1859 (2018)
29. Chung, S., et al.: A survey on aerial swarm robotics. *IEEE Trans. Rob.* **34**(4), 837–855 (2018)
30. Radiani, J., Gonzalez, J.J., Granmo, O.-C.: Publish-subscribe smartphone sensing platform for the acute phase of a disaster: a framework for emergency management support. In: 2014 IEEE International Conference on Pervasive Computing and Communication Workshops (PERCOM WORKSHOPS). IEEE (2014)
31. Radiani, J., Lazreg, M.B., Granmo, O.-C.: Fire simulation-based adaptation of SmartRescue App for serious game: design, setup and user experience. *Eng. Appl. Artif. Intell.* **46**, 312–325 (2015)
32. Ryabko, B., Rakitskiy, A.: Theoretical approach to performance evaluation of supercomputers. *J. Circ. Syst. Comput.* **27**(04), 1850062 (2018)

# **Machine Learning Modeling - Optimization**



# Evolutionary Optimization on Artificial Neural Networks for Predicting the User's Future Semantic Location

Antonios Karatzoglou<sup>(✉)</sup>

Chassis System Control, Advance Engineering, Robert Bosch, Abstatt, Germany  
[antonios.karatzoglou@de.bosch.com](mailto:antonios.karatzoglou@de.bosch.com)

**Abstract.** Location prediction has gained enormously in importance in the recent years. For this reason, there exists a great variety of research work carried out at both the academia and the industry. At the same time, there is an increasing trend towards utilizing additional semantic information aiming at building more accurate algorithms. Existing location prediction approaches rely mostly on data-driven models, such as Hidden Markov Chains, Bayes Networks and Artificial Neural Networks (ANN), with the latter achieving usually the best results. Most ANN-based solutions apply Grid Parameter Search and Stochastic Gradient Descent for training their models, that is, for identifying the optimal structure and weights of the network. In this work, motivated by the promising results of genetic algorithms in optimizing neural networks in temporal sequence learning areas, such as the gene and the stock price index prediction, we propose and evaluate their use in optimizing our ANN-based semantic location prediction model. It can be shown that evolutionary algorithms can lead to a significant improvement with respect to its predictive performance, as well as to the time needed for the model's optimization.

**Keywords:** Evolutionary algorithms · Artificial Neural Networks · Semantic trajectories · Semantic location prediction

## 1 Introduction

The growing use of GPS technology in mainstream devices, such as mobile phones, smartwatches and cars, to name but a few, lay the basis of a number of services that utilize the location information to provide their users with more accurate and timely solutions. These services are referred to as *Location-based Services (LBS)*. Recently, LBS providers invest increasingly on location prediction techniques to further improve the user experience of their services.

So far, a big variety of different models has been investigated in both the academic and the private sector. These include probabilistic approaches like Hidden Markov Models (HMM) and Bayes Networks, as well as other methods like Support Vector Machines (SVM) and Artificial Neural Networks (ANN). In general,

ANN-based prediction algorithms have shown to be very effective in learning temporal sequences. This behaviour could be also confirmed in several location prediction scenarios in the existing literature and motivated us to use them in our work as well. However, neural networks come with a certain handicap that concerns mostly their training and their adjustment to the respective prediction task. In order to be able to perform well, they need to have the appropriate topology and an optimal set of hyperparameters. Finding the right values can be very tricky and very time consuming, since the typical search process relies on an exhaustive grid search. Genetic algorithms represent a promising alternative.

Most of the aforementioned models build upon plain GPS data. However, there exists a group of researchers that apply their models on semantically enriched GPS trajectories, so called *semantic trajectories*. Semantic Trajectories provide the models with an additional conceptual view upon the movement patterns of the users (see Sect. 3). Our work is based on exact this type of trajectories. To the best of our knowledge, there is no study evaluating the usage of genetic algorithms for optimizing a semantic trajectory prediction model.

The rest of this paper is structured as follows. First, Sect. 2 gives a brief overview of some of the most related work. Then, in Sect. 3, we describe the concept of *semantic trajectories*. Section 4 provides the theory behind the evolutionary algorithms and describes how these can be used for the optimization of neural networks. Finally, Sects. 5 and 6 list our evaluation results and some concluding notes respectively.

## 2 Related Work

Modelling human movement patterns and predicting upon them represents a well researched and evaluated topic in the scientific community. As already mentioned in the introductory section, most work relies on numerical data, like GPS or cell tower data. In the last decade though, a new group of researchers utilize semantic information for leveraging their models and the respective predictive performance. The first part of this section discusses the most relevant research in the field of semantic-enhanced location prediction. The second part describes a group of works, in which genetic algorithms are used in a location prediction scenario. In contrast to our work, none of them concerns semantic outdoor trajectories.

Ying et al. deployed in [32] their own Geographic Semantic Information Database (GSID) to semantically enrich GPS and cell tower ID trajectories. The GSID refers to a POI<sup>1</sup> database containing semantic (geo-)information of landmarks and associated location types. Ying et al. use the returned semantic trajectories to propagate a prefix tree model that serves as basis for their improved next location prediction algorithm. In [31], they extend their model by adding temporal information explicitly to its input patterns. Taking these temporal patterns additionally into account helps improve their former model. Karatzoglou et al. explore in their research a large variety of semantic trajectory modelling

---

<sup>1</sup> Point of Interest.

and prediction methods. In [7–9], they explore the use of context-specific multi-dimensional Markov Chains as well as the degree of semantic enrichment, that is the type and the amount of the additional semantic information that can be fed into the model, to achieve better accuracy scores. Furthermore, they propose a context-driven semantic similarity based approach for dynamic clustering of locations, which provides their model with the necessary flexibility to overcome, among others and to a certain degree, sparse and inconsistent training datasets. In [10,11], they evaluate the performance of various neural network architecture types including the Feed-Forward (FFNN), the Recurrent (RNN), the Long Short-term Memory (LSTM) and the Convolutional Neural Network (CNN) with the LSTM and the CNN performing overall best. Finally, their work in [6] extends the LSTM-based model and investigates the application of Sequence to Sequence Learning (Seq2Seq) and its impact on both the short- and the long-term prediction. Samaan et al. model human trajectories using conceptual maps, which are very close to the notion of semantic trajectories [19–21]. In addition, a set of user profile information encoded in XML, such as the user's preferences and her schedule, are used to support their probabilistic inference process. Ridhawi et al. present in their work a similar approach for indoor tracking of users [17,18]. In contrast to Samaan et al., Rhidawi et al. use ontologies for modelling and storing the users' information. In [13], Long et al. parse the (textual) input of Location-Based Social Network (LBSN) users to analyze their spatial and temporal movement patterns at a higher semantic level. Krishnamurthy et al. exploit the same type of data to predict the next semantic location of mobile LBSN users [12]. For this purpose, they analyze the semantic similarity between locations and the users' input (checkins) based on several semantic similarity measures, like the Jaccard [5] and the Tversky [24] Index. Ye et al. in [30] use LBSN data as well and use the within included semantic labels of locations as input for their Mixed Hidden Markov Model (MHMM). Finally, Wannous et al. and Malki et al. propose a multi-ontology based approach in combination with a set of rules created by a group of experts to model and reason about semantically annotated movement and activity patterns of marine mammals [15,26–29].

There exists a rather limited number of papers that use evolutionary algorithms, either in a direct or in an indirect manner, in a location prediction scenario. Mantoro et al. propose a direct approach, in which a genetic algorithm is used to represent the visits of a certain user at a certain location at a certain time [16]. The evolution process returns simply the most likely places to be occupied by each user. Mala et al. apply a genetic algorithm for capturing and evolving the moving speed and the current traffic situation over time [14]. This information is then used to support the location transition probability function and consequently the overall prediction performance. In [23], Tiwari et al. make use of the genetic algorithm to encode and evolve the sparse User-Location matrix that contains the preference scores between each location and user. Their model is able to outperform the classic Matrix Factorization approach. Finally, Vlahogianni et al. present in [25] a genetic optimization approach for optimizing neural networks with respect to a short-term traffic flow prediction use case. In

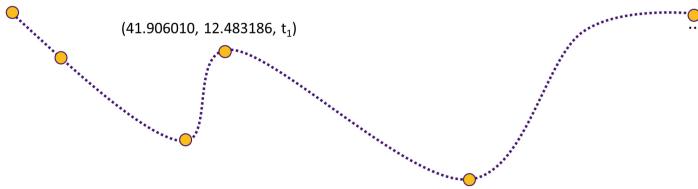
contrast to the aforementioned models, this work concentrates on the optimization of a neural network model for predicting upon *semantic trajectories*.

### 3 Semantic Trajectories

A *trajectory* is a spatio-temporal sequence that describes the movement of objects in space within a certain temporal interval. The most common trajectories nowadays are GPS trajectories. These are usually defined as a sequence of GPS points, that is, triples containing the objects' latitude ( $lat_i$ ), longitude ( $long_i$ ) and a corresponding point of time ( $t_i$ ) as displayed in the following equation:

$$Traj_{GPS} = (lat_1, long_1, t_1), (lat_2, long_2, t_2), \dots \quad (1)$$

Figure 1 shows a daily GPS trajectory as a sequence of GPS points.



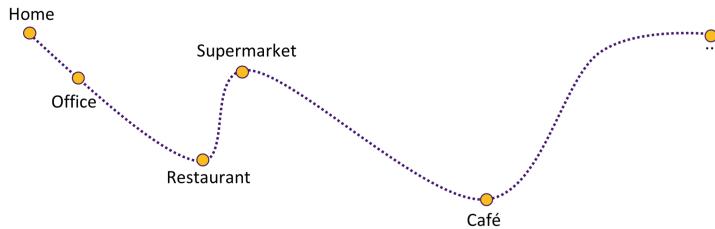
**Fig. 1.** Example of a GPS trajectory as a sequence of GPS points.

In 2007, Alvares et al. and Spaccapietra et al. highlighted in their work the benefits of adding a conceptual, semantic layer upon the numeric trajectories when analyzing moving objects [1, 22]. Taking semantic information additionally into account provides the analysts with a deeper insight into their movement patterns and thus with a greater understanding of their moving behaviour. A *semantic trajectory* refers to such a semantically enriched trajectory and with respect to human trajectories, it consists of a sequence of a few significant locations. *Significant locations* represent locations at which users stay long enough to perform a certain activity (e.g., *mall*, *restaurant*, *office* or *gym*) [2]. Semantic trajectories conform in a certain way with Hägerstrand's notion of a *space-time prism* that describes the trade-off between time and space depending on the respective activity [4]. Figure 2 illustrates the reduction of thousands of recorded GPS points to a sequence of a few significant semantic locations.

This work concentrates on modeling and predicting upon exact this type of semantic trajectories.

### 4 Artificial Neural Network Optimization Based on Evolutionary Algorithms (EA)

*Evolutionary Algorithms (EA)* represent a group of stochastic optimization algorithms that emulate the evolution processes of nature. The underlying idea relies



**Fig. 2.** Example of a 1-day long *semantic* Trajectory.

on searching at each step over a whole population of possible solutions to find the optimal one. The respective possible solutions are encoded as *chromosomes*, each containing a set of *genes*.

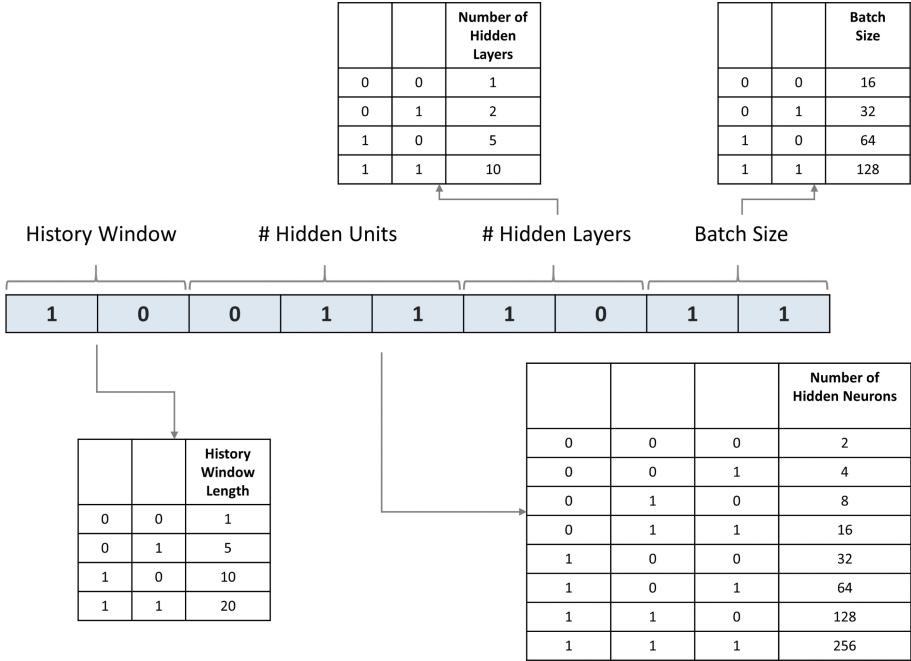
The core process of evolution algorithms consists of the following steps:

- Random generation of an initial population of chromosomes
- Evaluation of the fitness of each individual chromosome
- Selection of a group of chromosomes for the next generation based on their fitness (*Selection*)
- Use of selected chromosomes as parents to generate new chromosomes either by combining them (*Crossover*) or altering them (*Mutation*) based on some predefined probability distribution.
- Repetition of above last 3 steps until a certain criterion is fulfilled.

As already mentioned before in this work, the above process has been successfully used for optimizing the hyperparameters of neural networks, like its architecture (number of hidden layers, neurons and their interconnections), the learning rate and the training batch size, to name but a few. In the matter of fact, they represent one of the most promising approaches when it comes to a derivative-free, non-backpropagation optimization. In this work, we evaluate the use of a genetic algorithm for optimizing the structure of the neural network with respect to the optimization process' temporal expenditure during the training and the overall test accuracy in the semantic location prediction scenario. In particular, we concentrate on finding the best possible values for the following hyperparameters:

- Number of hidden layers
- Number of hidden neurons (LSTM units in our case)
- Batch size (during training)
- Number of past locations taken into consideration at each time (History).

Furthermore, in contrast to the *direct binary encoding*, that is used to represent the complete network architecture by explicitly encoding each neuron and their position in the network as well as the existing interconnections between them, in our work, we make use of the *indirect binary encoding*. That is, we encode solely the parameters in which we are interested in based on the value

**Fig. 3.** Hyperparameter chromosome.

range that we want to investigate. Figure 3 illustrates the composed hyperparameter chromosome that is going to be evolved.

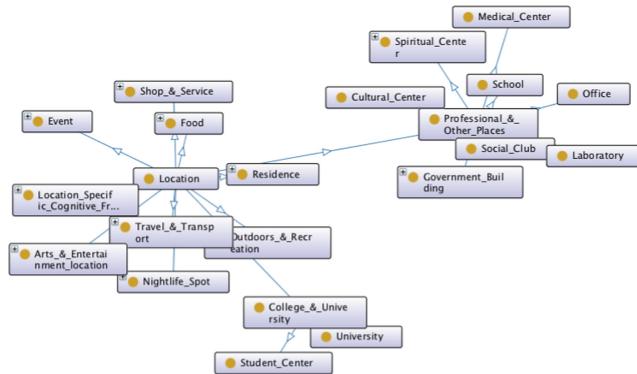
There exist many different types of evolutionary algorithms, such as the *Evolutionary Strategy*, the *Evolutionary Programming* and the *Genetic Algorithms (GA)*. In this work, we concentrate on the latter one. Moreover, our genetic algorithm applies the *Roulette selection* scheme for generating new populations out from the older ones. The particular scheme favours the survival of the fittest genes. Similar to other optimization techniques, evolutionary and thus genetic algorithms are defined by a set of hyperparameters. In our case, these are listed below:

- Size of the population (number of chromosomes)
- Number of generations (number of iterations of the evolution process)
- Initial Gene distribution
- Crossover probability
- Mutation probability.

## 5 Evaluation

Semantic trajectories can be represented at various levels, depending on the applied semantic granularity, e.g., *food location* vs. *restaurant* vs. *fast food restaurant* vs. *burger joint*. We tested our approach on modeling human trajectories

at two different semantic representation levels, e.g. *restaurant* vs. *burger joint*, *Chinese restaurant*, *pizzeria*,... or *night life location* vs. *bar*, *night club*, *disco*, *cinema*,... . Our semantic location taxonomy is based on the Foursquare venue categorization<sup>2</sup>. Figure 4 shows a small part of our Reality Mining semantic location taxonomy.

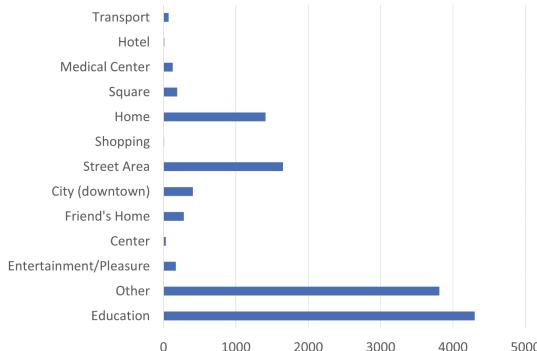


**Fig. 4.** Part of our Reality Mining location taxonomy.

We evaluated our approach using the MIT Reality Mining dataset [3]. The Reality Mining dataset contains semantically annotated trajectory data from 100 mobile users over a period of 9 months. Before using it for evaluation, we cleansed and preprocessed the data, among others, by removing nonsensical data (e.g., huge location jumps in extremely short time intervals) and by filtering out extremely sparse annotators. Figure 5 shows the distribution of the resulted locations for the high level case. It is apparent that the Reality Mining dataset is extremely unbalanced. In order to compensate this unbalance in our evaluation outcome, we used a weighted macro accuracy metric that takes this outbalance explicitly into account. A Markov Chain model, as well as the Convolutional (CNN) of [10] and a LSTM-based Neural Network optimized through a typical grid search served as our baseline. The parameters can be found in Table 1.

To optimize our LSTM-based model, we applied a simple genetic algorithm. Table 2 contains the respective hyperparameter values. These were obtained through a short testing process. We ran the evolution process twice. Once for the low and once for the high representation level trajectories. Thus, at the end, we received two optimal hyperparameter configuration sets for our predictive model (Table 3). It can be seen that the values for both the low and the high level case are pretty similar. The number of past locations to be considered (history window) as well as the number of hidden layers are identical. Interestingly, the largest difference occurs with respect to the number of LSTM units per hidden layer. It seems that our LSTM model needs a wider topology when it comes

<sup>2</sup> <https://developer.foursquare.com/docs/resources/categories>.



**Fig. 5.** Distribution of the semantic locations in the (filtered) MIT dataset at the higher semantic representation level.

**Table 1.** Hyperparameter values of our baseline models.

Hyperparameter	Markov chains values	CNN values	LSTM values
Length of history window	3	10	2
Number of LSTM units	-	-	128
Number of hidden layers	-	1	1
Batch size	-	100	16
Number of Kernels	-	50	-
Kernel size	-	2	-

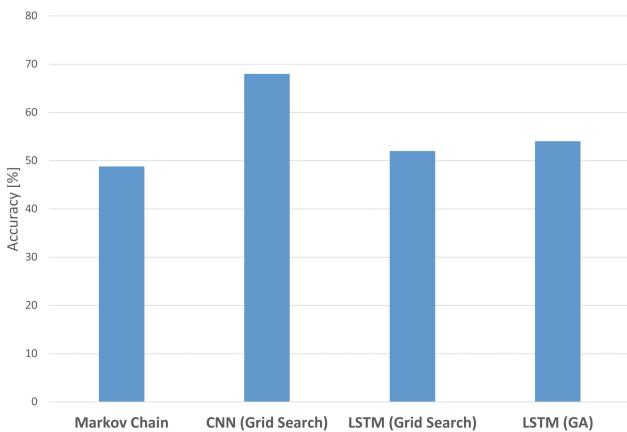
**Table 2.** Hyperparameters of the genetic algorithm.

Hyperparameter	Value
Population size	4
Number of generations	4
Gene distribution	Bernoulli distribution
Crossover probability	0.4
Mutation probability	0.1

**Table 3.** Optimal hyperparameter s values determined through the genetic algorithm.

Hyperparameter	Value (low level)	Value (high level)
Length of history window	5	5
Number of LSTM units	2	64
Number of hidden layers	1	1
Batch size	16	32

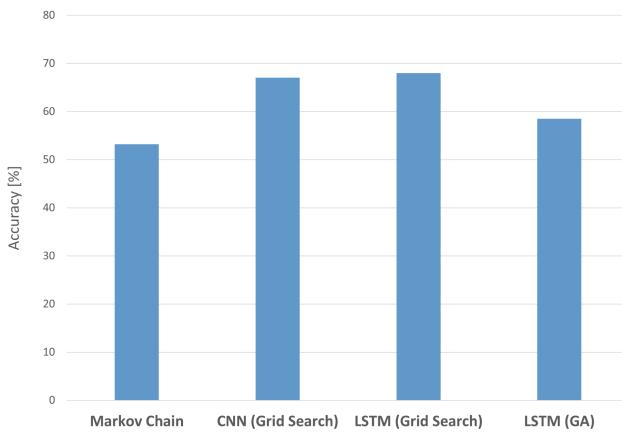
to modeling high level human movement patterns. And this, despite the fact that the overall number of high level location types and thus the corresponding input vector is smaller compared to the number of unique locations at the lower level. Figure 6 shows the results for the low representation level case. We can see that the GA-optimized network is able to outperform both the LSTM model that was optimized through an exhaustive grid search as well as the Markov Chain model of 3. Order. Solely, and as expected based on former investigations in [10], the CNN approach is better. Thus, the genetic algorithm based optimization techniques seems to enhance the standard LSTM, however, it couldn't lead to giant leaps.



**Fig. 6.** Accuracy scores at the lower semantic representation level.

At the higher level, all models perform generally better. The GA-based LSTM is again able to outperform the Markov Chain model, but this time, in contrast to the low level case, it can't reach the scores of both the standard LSTM and the CNN. Once again, the results here confirm the overall best performance of the LSTM and the CNN in [10]. The reduced performance of the GA-based LSTM could be mainly attributed to a certain overfitting effect during the training process due to the smaller number of location classes in combination with the large number of hidden units while keeping the population number the same.

In terms of time needed to reach to an optimal hyperparameter set, the optimal parameter search lasted much less compared to the exhaustive grid search, by a factor of 0.8. A random grid search should probably be equally fast, but the overall final results are usually not as optimal as with the population based genetic algorithmic approach, due to its “naive” high variance when selecting points randomly in the hyperparameter space.



**Fig. 7.** Accuracy scores at the higher semantic representation level.

## 6 Conclusion

Semantically enriched Location Based Services have recently gained increasingly in importance. For this reason, there exists a growing variety of semantic location prediction approaches in the current literature. One of the most common and promising methods are the ones that rely on Artificial Neural Networks. However, finding the appropriate network topology and the respective hyperparameters is a very time consuming process. Evolutionary algorithms help shortening this process while keeping the models' performance high. This work explores the use of Genetic Algorithms in optimizing a Long Short-Term Memory neural network in a semantic trajectory modeling and prediction scenario. It can be shown that genetic algorithms are capable of optimizing the LSTM and help them reach high accuracy scores outperforming baseline systems, such as a Markov Chains and a Grid Search based optimized LSTM model.

## References

1. Alvares, L.O., Bogorny, V., Kuijpers, B., de Macedo, J.A.F., Moelans, B., Vaisman, A.: A model for enriching trajectories with semantic geographical information. In: Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems, GIS 2007, pp. 22:1–22:8. ACM, New York (2007). <https://doi.org/10.1145/1341012.1341041>
2. Ashbrook, D., Starner, T.: Using GPS to learn significant locations and predict movement across multiple users. Pers. Ubiquitous Comput. **7**(5), 275–286 (2003)
3. Eagle, N., Pentland, A.S.: Reality mining: sensing complex social systems. Pers. Ubiquitous Comput. **10**(4), 255–268 (2006)
4. Hägerstrand, T.: What about people in regional science? Pap. Reg. Sci. **24**(1), 7–24 (1970)
5. Jaccard, P.: The distribution of the flora in the alpine zone. New Phytol. **11**(2), 37–50 (1912)

6. Karatzoglou, A., Jablonski, A., Beigl, M.: A Seq2Seq learning approach for modeling semantic trajectories and predicting the next location. In: Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems. ACM (2018)
7. Karatzoglou, A., Khler, D., Beigl, M.: Semantic-enhanced multi-dimensional Markov chains on semantic trajectories for predicting future locations. *Sensors* **18**(10) (2018). <https://doi.org/10.3390/s18103582>. <http://www.mdpi.com/1424-8220/18/10/3582>
8. Karatzoglou, A., Köhler, D., Beigl, M.: Purpose-of-visit-driven semantic similarity analysis on semantic trajectories for enhancing the future location prediction. In: International Conference on Pervasive Computing and Communications Workshops (PerCom). IEEE (2018)
9. Karatzoglou, A., Lamp, S.C., Beigl, M.: Matrix factorization on semantic trajectories for predicting future semantic locations. In: Wireless and Mobile Computing, Networking and Communications (WiMob), pp. 1–7. IEEE (2017)
10. Karatzoglou, A., Schnell, N., Beigl, M.: A convolutional neural network approach for modeling semantic trajectories and predicting future locations. In: Kůrková, V., Manolopoulos, Y., Hammer, B., Iliadis, L., Maglogiannis, I. (eds.) ICANN 2018. LNCS, vol. 11139, pp. 61–72. Springer, Cham (2018). [https://doi.org/10.1007/978-3-030-01418-6\\_7](https://doi.org/10.1007/978-3-030-01418-6_7)
11. Karatzoglou, A., Sentürk, H., Jablonski, A., Beigl, M.: Applying artificial neural networks on two-layer semantic trajectories for predicting the next semantic location. In: Lintas, A., Rovetta, S., Verschure, P.F.M.J., Villa, A.E.P. (eds.) ICANN 2017. LNCS, vol. 10614, pp. 233–241. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-68612-7\\_27](https://doi.org/10.1007/978-3-319-68612-7_27)
12. Krishnamurthy, R., Kapanipathi, P., Sheth, A.P., Thirunarayanan, K.: Knowledge enabled approach to predict the location of Twitter users. In: Gandon, F., Sabou, M., Sack, H., d'Amato, C., Cudré-Mauroux, P., Zimmermann, A. (eds.) ESWC 2015. LNCS, vol. 9088, pp. 187–201. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-18818-8\\_12](https://doi.org/10.1007/978-3-319-18818-8_12)
13. Long, X., Jin, L., Joshi, J.: Exploring trajectory-driven local geographic topics in foursquare. In: Proceedings of the 2012 ACM Conference on Ubiquitous Computing, UbiComp 2012, pp. 927–934. ACM, New York (2012)
14. Mala, C., Loganathan, M., Gopalan, N.P., SivaSelvan, B.: A novel genetic algorithm approach to mobility prediction in wireless networks. In: Ranka, S., et al. (eds.) IC3 2009. CCIS, vol. 40, pp. 49–57. Springer, Heidelberg (2009). [https://doi.org/10.1007/978-3-642-03547-0\\_6](https://doi.org/10.1007/978-3-642-03547-0_6)
15. Malki, J., Wannous, R., Bouju, A., Vincent, C.: Temporal reasoning in trajectories using an ontological modelling approach. *Control Cybern.* **41** (2012)
16. Mantoro, T., Muataz, Z., Ayu, M.A.: Mobile user location prediction: genetic algorithm-based approach. In: 2010 IEEE Symposium on Industrial Electronics and Applications (ISIEA), pp. 345–349. IEEE (2010)
17. Ridhawi, I.A., Aloqaily, M., Karmouch, A., Agoulmine, N.: A location-aware user tracking and prediction system. In: 2009 Global Information Infrastructure Symposium, pp. 1–8, June 2009
18. Ridhawi, Y.A., Ridhawi, I.A., Karmouch, A., Nayak, A.: A context-aware and location prediction framework for dynamic environments. In: 2011 IEEE 7th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), pp. 172–179, October 2011

19. Samaan, N., Benmammar, B., Krief, F., Karmouch, A.: Prediction-based advanced resource reservation in mobile environments. In: Canadian Conference on Electrical and Computer Engineering 2005, pp. 1411–1414 (2005)
20. Samaan, N., Karmouch, A., Kheddouci, H.: Mobility prediction based service location and delivery. In: Canadian Conference on Electrical and Computer Engineering 2004 (IEEE Cat. No. 04CH37513), vol. 4, pp. 2307–2310, May 2004
21. Samaan, N., Karmouch, A.: A mobility prediction architecture based on contextual knowledge and spatial conceptual maps. *IEEE Trans. Mob. Comput.* **4**(6), 537–551 (2005)
22. Spaccapietra, S., Parent, C., Damiani, M.L., de Macedo, J.A., Porto, F., Vangenot, C.: A conceptual view on trajectories. *Data Knowl. Eng.* **65**(1), 126–146 (2008). <https://doi.org/10.1016/j.datak.2007.10.008>
23. Tiwari, S., Kaushik, S.: Modeling personalized recommendations of unvisited tourist places using genetic algorithms. In: Chu, W., Kikuchi, S., Bhalla, S. (eds.) DNIS 2015. LNCS, vol. 8999, pp. 264–276. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-16313-0\\_20](https://doi.org/10.1007/978-3-319-16313-0_20)
24. Tversky, A.: Features of similarity. *Psychol. Rev.* **84**(4), 327 (1977)
25. Vlahogianni, E.I., Karlaftis, M.G., Golias, J.C.: Optimized and meta-optimized neural networks for short-term traffic flow prediction: a genetic approach. *Transp. Res. Part C: Emerg. Technol.* **13**(3), 211–234 (2005)
26. Wannous, R., Malki, J., Bouju, A., Vincent, C.: Modelling mobile object activities based on trajectory ontology rules considering spatial relationship rules. In: Amine, A., Otmane, A., Bellatreche, L. (eds.) Modeling Approaches and Algorithms for Advanced Computer Applications, pp. 249–258. Springer, Cham (2013). [https://doi.org/10.1007/978-3-319-00560-7\\_29](https://doi.org/10.1007/978-3-319-00560-7_29)
27. Wannous, R., Malki, J., Bouju, A., Vincent, C.: Time integration in semantic trajectories using an ontological modelling approach. In: Pechenizkiy, M., Wojciechowski, M. (eds.) New Trends in Databases and Information Systems, pp. 187–198. Springer, Heidelberg (2013). [https://doi.org/10.1007/978-3-642-32518-2\\_18](https://doi.org/10.1007/978-3-642-32518-2_18)
28. Wannous, R., Malki, J., Bouju, A., Vincent, C.: Trajectory ontology inference considering domain and temporal dimensions—application to marine mammals. *Future Gener. Comput. Syst.* **68**, 491–499 (2016)
29. Wannous, R., Vincent, C., Malki, J., Bouju, A.: An ontology-based approach for handling explicit and implicit knowledge over trajectories. In: Morzy, T., Valdoriez, P., Bellatreche, L. (eds.) ADBIS 2015. CCIS, vol. 539, pp. 403–413. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-23201-0\\_41](https://doi.org/10.1007/978-3-319-23201-0_41)
30. Ye, J., Zhu, Z., Cheng, H.: What's your next move: user activity prediction in location-based social networks. In: Proceedings of the 2013 SIAM International Conference on Data Mining, pp. 171–179. SIAM (2013)
31. Ying, J.J.C., Lee, W.C., Tseng, V.S.: Mining geographic-temporal-semantic patterns in trajectories for location prediction. *ACM Trans. Intell. Syst. Technol.* **5**(1), 2:1–2:33 (2014)
32. Ying, J.J.C., Lee, W.C., Weng, T.C., Tseng, V.S.: Semantic trajectory mining for location prediction. In: Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS 2011, pp. 34–43. ACM, New York (2011)



# Global Minimum Depth in Edwards-Anderson Model

Iakov Karandashev<sup>1,2</sup> and Boris Kryzhanovsky<sup>1</sup>

<sup>1</sup> Center of Optical Neural Technologies,  
Scientific Research Institute for System Analysis RAS,  
Nakhimovskiy prosp., 36, b.1., Moscow 117218, Russia  
`{karandashev, kryzhanov}@niisi.ras.ru`

<sup>2</sup> Peoples Friendship University of Russia (RUDN University),  
6 Miklukho-Maklaya Street, Moscow 117198, Russian Federation

**Abstract.** In the literature the most frequently cited data are quite contradictory, and there is no consensus on the global minimum value of 2D Edwards-Anderson (2D EA) Ising model. By means of computer simulations, with the help of exact polynomial Schraudolph-Kamenetsky algorithm, we examined the global minimum depth in 2D EA-type models. We found a dependence of the global minimum depth on the dimension of the problem  $N$  and obtained its asymptotic value in the limit  $N \rightarrow \infty$ . We believe these evaluations can be further used for examining the behavior of 2D Bayesian models often used in machine learning and image processing.

**Keywords:** Spectrum · Local minimum · Global minimum · Spin system · Spin glass system · Minimization · Exact polynomial algorithm · Edwards-Anderson model · Planar Ising model

## 1 Introduction

In many fields of science, it is necessary to know the global energy minimum for different systems. Namely, in informatics we use it when solving problems of quadratic optimization [1, 2], developing search algorithms for the global minimum and solving max-cut problems [3–7]. In neuroinformatics, we have to know the global minimum when developing associative memory systems and constructing neural networks and neural network minimization algorithms [7]. In physics, the knowledge of the global energy minimum is most frequently necessary when studying the behavior of spin glass systems and even when describing four-photon mixing in nonlinear media [8–13].

The question of calculation of the global minimum depth has been discussed over the years. However, since it has no decisive answer it remains a highly topical problem up to now. Indeed, in the literature the most frequently cited data are quite contradictory, and there is no consensus on the global minimum value (see references in [10]). To illustrate this statement, we present the values of the global minimum depth obtained by different methods:

$E_0 = 0$	TAP (Thouless et al. [8])
$E_0 = 1/\sqrt{2\pi}$	mean random field (Klein [9])
$E_0 = 0.5$	partition function (Tanaka and Edwards [10])
$E_0 = 2/\pi$	replica (Sherrington and Kirkpatrick [11])
$E_0 = 0.76 \sim 0.77$	Monte Carlo (Sherrington and Kirkpatrick [12])

Such a spread of values exists because until recently there were no exact calculation algorithms for the determination of  $E_0$ . This was the reason why different authors used different minimization methods, and consequently the obtained estimates were sufficiently far from the true value of  $E_0$ . New algorithms appeared recently. They allow us to calculate  $E_0$  exactly when examining spin systems on planar graphs with arbitrary boundary conditions [14]. Implementing these algorithms, we were able to refine our results [15] for the Edwards–Anderson model (the EA model).

In the present paper, we present an experimental analysis of the global minimum depth in the EA model, which is a spin system on an  $N = L \times L$  square lattice where only interactions with four nearest neighbors do not equal to zero. Formally, we have in mind a system whose behavior is described by a Hamiltonian

$$H = -\frac{1}{2} \sum_{i,j=1}^N J_{ij} s_i s_j \quad (2)$$

defined in the configuration space of states  $\mathbf{S} = (s_1, s_2, \dots, s_N)$  with binary variables  $s_i = \pm 1$ ,  $i = \overline{1, N}$ . Here  $N$  is the number of spins, and  $J_{ij}$  is a symmetric, zero-diagonal matrix ( $J_{ij} = J_{ji}$  and  $J_{ii} = 0$ ).

To describe the spectrum of the system, it is convenient to introduce the depth of the minimum that is defined by equation

$$E = \frac{1}{2N\sigma_J} \sum_{i=1}^N \sum_{j=1}^N J_{ij} s_i s_j \quad (3)$$

As we show in what follows, the normalization coefficient in Eq. (3) is quite universal since the value of  $E$  is almost independent of the dimension of the problem  $N$  as well as of the normalization of the matrix elements  $J_{ij}$ . In these notations, the Hamiltonian of the system has the form  $H = -N\sigma_J E$ , and its dependence on the dimension reduces to  $H \sim N$ .

As we see from Eq. (1), the results obtained by different authors are so very different that it is hardly possible to use them in the course of calculations. This was the reason why we performed a huge experiment having in mind to determine the basic spectral characteristics such as the mean value of the local minimum depth, the spectrum width, and the depth of the global minimum. Based on the obtained experimental data, we plotted the dependences of these characteristics on the dimension of the problem  $N$  and determined their asymptotic values in the limit  $N \rightarrow \infty$ .

The structure of the paper is as follows. In Sect. 2, we describe our experiment and analyze the obtained data. In Sect. 3, we discuss the results and the tables showing our experimental data.

## 2 Experiment

To define the value of  $E_0$ , we used an algorithm described in [14]. In the course of our experiment, we examined the classical EA-model (with the normal distribution of  $J_{ij}$ ) and the EA\*-model (with the uniform distribution of  $J_{ij}$ ). For the chosen model of the given size  $N = L \times L$ , we generated  $M$  matrices  $J_{ij}$  and determined  $M$  values  $E_{m0}$ ,  $m = \overline{1, M}$ . We used these data to calculate the mean value and the variance of the obtained values:

$$E_0 = M^{-1} \sum_{m=1}^M E_{m0}, \sigma_0^2 = M^{-1} \sum_{m=1}^M E_{m0}^2 - E_0^2 \quad (4)$$

The results of our experiments are collected in Table 1.

Based on the obtained data, we derived formulas that described the dependences of  $E_0$  and  $\sigma_0$  on  $N$ . We optimized these formulas by means of the least squares method. We minimized the value of the summary relative error and estimated the quality of the approximation formulas by the value of validity defined as

$$R^2 = 1 - \frac{\sum (x_{\text{exp}} - x_{\text{approx}})^2}{\sum (x_{\text{exp}} - \bar{x}_{\text{exp}})^2} \quad (5)$$

where  $x_{\text{exp}}$  are the experimental values,  $\bar{x}_{\text{exp}}$  are the means of the experimental values, and  $x_{\text{app}}$  are the values obtained using the approximation formulas.

### 2.1 EA-model

This is the Edwards–Anderson model for a two-dimensional lattice where spins interact with their four nearest neighbors only and nonzero matrix elements are normally distributed.

We found that the function of  $E_0$  is almost constant for large  $L$ . Analysis of our experimental data based on a standard maximization of value of  $R^2$  (5) shows that the more careful approximation functions have the form:

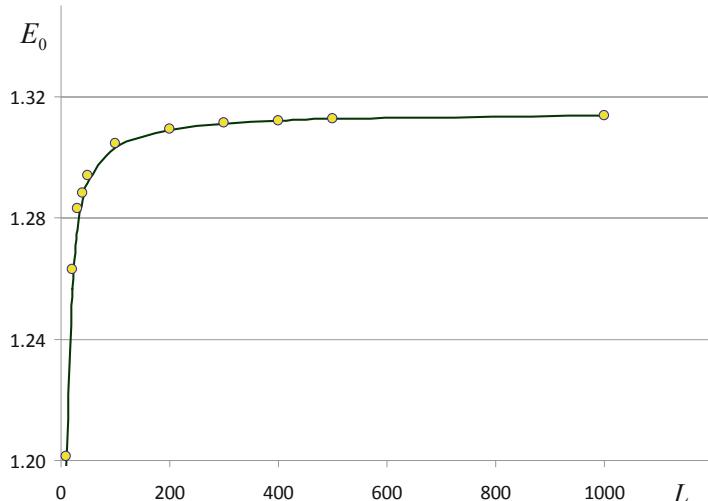
$$E_0 = 1.3151 - \frac{1.17}{L}, \sigma_0 = \frac{0.74}{L} + \frac{0.11}{L^2}. \quad (6)$$

The validities of these expressions are  $R^2 = 0.994$  and  $R^2 = 0.993$ , respectively.

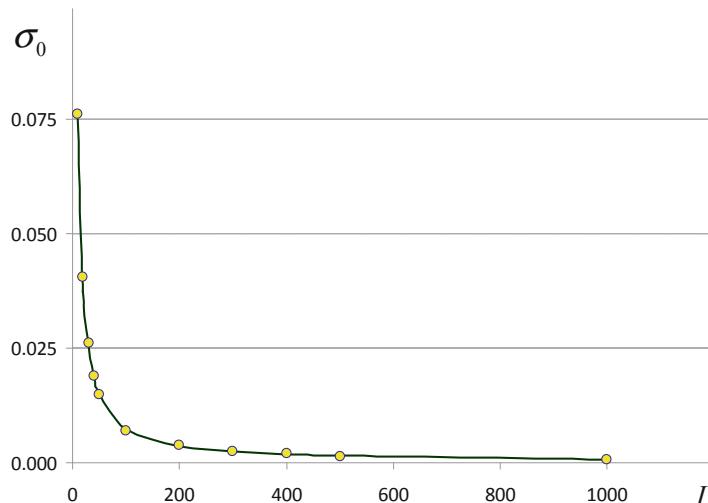
When comparing the expressions of Eq. (6) with the experiment, we see that these formulas describe them very well. In Fig. 1, we show that the function  $E_0 = E_0(N)$

matches perfectly with data of Table 1. The value of the relative error  $Err = 1 - E_0^{(\text{exp})}/E_0^{(\text{approx})}$  is less than  $2 \cdot 10^{-3}$ .

The function  $\sigma_0 = \sigma_0(N)$  (the second expression of Eq. (6)) in Fig. 2 also describes the data of Table 1 very well. The value of the relative error is less than 0.4%.



**Fig. 1.**  $E_0$  vs  $L$ . EA-model: line – Eq. (6), circles – experiment.



**Fig. 2.**  $\sigma_0$  vs  $L$ . EA-model: line – Eq. (6), circles – experiment.

**Table 1.** Characteristics of global minima

$L$	$M$	EA-model		EA*-model	
		$E_0$	$\sigma_0$	$E_0$	$\sigma_0$
10	200	1.20245	0.07611	1.27347	0.05579
20	200	1.26305	0.04049	1.32793	0.02848
30	200	1.28416	0.02608	1.34307	0.01934
40	200	1.28838	0.01882	1.34864	0.01514
50	200	1.29569	0.01493	1.35738	0.01066
100	200	1.30448	0.00695	1.36681	0.00517
200	200	1.30957	0.00392	1.37173	0.00294
300	100	1.31136	0.00254	1.37337	0.00187
400	100	1.31229	0.00193	1.37467	0.00144
500	30	1.31279	0.00135	1.37498	0.00113
1000	30	1.31390	0.00074	1.37564	0.00054

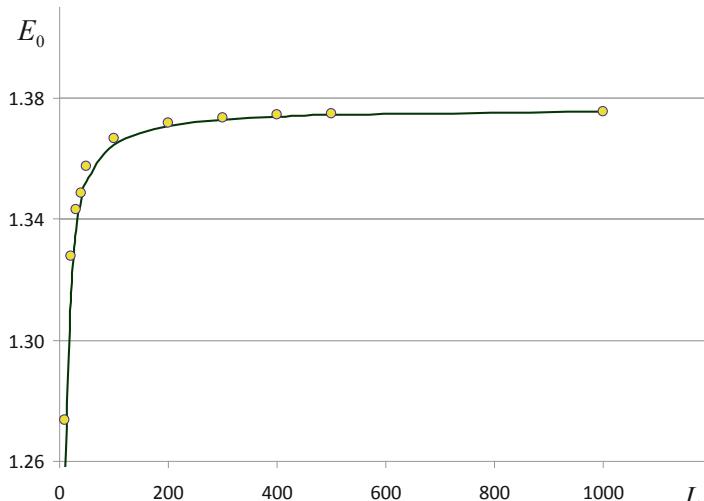
## 2.2 EA\*-model

This is the same Edwards–Anderson model for a two-dimensional lattice, but here the uniform distribution is used in place of the normal distribution.

In this case, approximation functions obtained after our analysis of the experimental data have the form

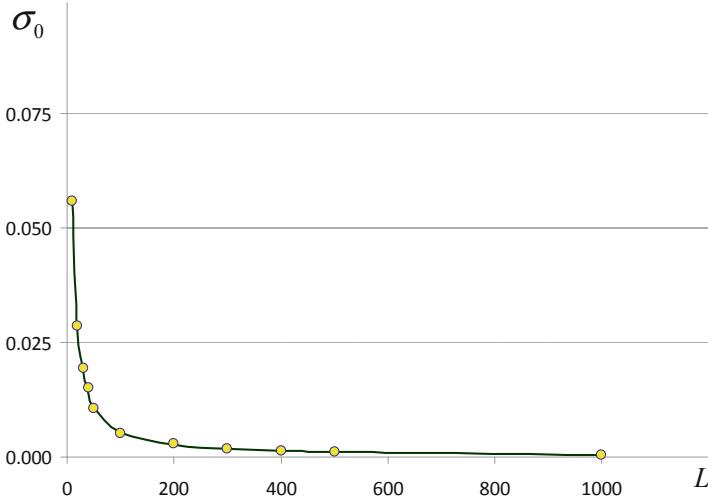
$$E_0 = 1.3769 - \frac{1.23}{L}, \quad \sigma_0 = \frac{0.54}{L} + \frac{0.2}{L^2} \quad (7)$$

The validities of these expressions are  $R^2 = 0.996$  and  $R^2 = 0.992$ , respectively.

**Fig. 3.**  $E_0$  vs  $L$ . EA-model: line – Eq. (6), circles – experiment.

Comparing the expressions of Eq. (7) with the experiment, we see that they describe it very well. In Fig. 3, we present the dependence  $E_0 = E_0(N)$  (the first expression of Eq. (7)) that matches perfectly with the data from Table 1. When  $L > 50$ , the relative error is less than  $2 \cdot 10^{-4}$ .

The dependence  $\sigma_0 = \sigma_0(N)$  (the second expression of Eq. (7)) shown in Fig. 4 also describes the data from Table 1 very well. Here the relative error is less than 0.5%.



**Fig. 4.**  $\sigma_0$  vs  $L$ . EA\*-model: line – Eq. (7), circles – experiment.

### 3 Discussion

Our analysis of the two models allowed us to derive empirical relations in Eqs. (6) and (7) for the most important characteristics of the global minima (see Eqs. (6) and (7)). Our goal was to obtain expressions which with a high certainty described the dependences of these characteristics on  $N$  in the whole range of the dimensions of the problem that we were able to examine. Based on these results, we had to determine the asymptotic behavior of these characteristics when  $N \rightarrow \infty$ . Evidently there are different approaches to approximation of the experimental data in Table 1. Consequently, it is possible to obtain a list of different expressions, and some of them can be even more accurate than the expressions of Eqs. (6) and (7). However, this fact does not change the goal of our study: independent of the form of the obtained approximation functions, they have to describe correctly the behavior of the characteristics inside the test range of  $N$  and provide trustworthy asymptotic values when  $N \rightarrow \infty$  (see Table 2).

**Table 2.** Asymptotic values of  $E_0$  and  $\sigma_0$  ( $N \rightarrow \infty$ ).

	$E_0$	$\sigma_0$
EA-model	$1.3151 \pm 0.002$	$0.74/L$
EA*-model	$1.3769 \pm 0.002$	$0.54/L$

As we see, the data of Table 2 differ significantly from the values presented in Eq. (1). The point is that when minimizing the functional of Eq. (2) with a view to calculating  $E_0$  different authors used different minimization algorithms. To do that, they defined  $E_0$  as the energy corresponding to the deepest minimum, which under a reasonable number of tests frequently was far from  $\bar{E}_0$ . As an example, let us discuss the results of numerical experiments [15] in which they defined the energy of the deepest minimum  $E^*$ . Then, for the relative distance

$$\delta E = 100\% \cdot \frac{E_0 - E^*}{\bar{E}_0} \quad (8)$$

between  $E_0$  and  $E^*$  when  $L \geq 100$  we obtain:

$$\delta E = 16.45 \% \pm 0.5 \%, \quad \text{for EA-model,} \quad (9)$$

$$\delta E = 16.61 \% \pm 0.4 \%, \quad \text{for EA*-model.} \quad (10)$$

From our point of view, this is a possible reason why the estimates of  $E_0$  obtained by different authors differ so significantly. Namely, when the size of the system is sufficiently large ( $L \geq 30$ ) such an approach is not applicable since the probability of finding the global minimum in the course of a random search is exponentially small: it is  $\sim \exp(-0.04N)$ .

**Acknowledgements.** The work was supported by Russian Foundation for Basic Research (RFBR Project 18-07-00750).

## References

1. Hartmann, A.: Calculation of ground states of four-dimensional  $\pm J$  Ising spin glasses. *Phys. Rev. B* **60**, 5135–5138 (1999)
2. Kryzhanovsky, B., Kryzhanovsky, V.: Binary optimization: on the probability of a local minimum detection in random search. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2008. LNCS (LNAI), vol. 5097, pp. 89–100. Springer, Heidelberg (2008). [https://doi.org/10.1007/978-3-540-69731-2\\_10](https://doi.org/10.1007/978-3-540-69731-2_10)
3. Houdayer, J., Martin, O.C.: Hierarchical approach for computing spin glass ground states. *Phys. Rev. E* **64**, 056704 (2001)
4. Litinskii, L.B., Magomedov, B.M.: Global minimization of a quadratic functional: Neural networks approach. *Pattern Recog. Image Anal.* **15**(1), 80–82 (2005)
5. Karandashev, Y.M., Kryzhanovsky, B.V.: Transformation of energy landscape in the problem of binary minimization. *Dokl. Math.* **80**(3), 927–931 (2009)
6. Liers, F., Junger, M., Reinelt, G., Rinaldi, G.: Computing exact ground states of hard Ising spin glass problems by branch-and-cut. In: New Optimization Algorithms in Physics, pp. 47–68. Wiley (2004)
7. Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. *Proc. Nat. Acad. Sci. USA* **79**, 2554–2558 (1982)
8. Thouless, D.J., Anderson, P.W., Palmer, R.G.: Solution of solvable model of a spin glass. *Philos. Mag.* **35**, 593–601 (1977)

9. Klein, M.W.: Comparison of the self-consistent mean-random-field approximation with the  $n \rightarrow 0$  expansion of Sherrington and Kirkpatrick for spin glasses and with experiment. *Phys. Rev. B* **14**, 5008–5017 (1976)
10. Tanaka, F., Edwards, S.F.: Analytic theory of the ground state properties of a spin glass. I. Ising spin glass. *J. Phys. F: Metal Phys.* **10**, 2769–2778 (1980)
11. Sherrington, D., Kirkpatrick, S.: Solvable model of a spin-glass. *Phys. Rev. Lett.* **35**, 1792–1796 (1975)
12. Kirkpatrick, S., Sherrington, D.: Infinite-ranged models of spin-glasses. *Phys. Rev. B* **17**, 4384–4403 (1978)
13. Kryzhanovsky, B.V., Karapetyan, A.R., Glushko, B.A.: Theory of energy exchange and conversion via four-wave mixing in a nondissipative chi (3) material. *Phys. Rev. A* **44**(9), 6036–6042 (1991)
14. Schraudolph, N., Kamenetsky, D.: Efficient exact inference in planar Ising models. In: *Advances in Neural Information Processing Systems 21 (NIPS 2008)* (2008)
15. Kryzhanovsky, B., Malsagov, M.: The spectra of local minima in spin-glass models. *Opt. Memory Neural Netw. (Inf. Opt.)* **25**(1), 1–15 (2016)



# Imbalanced Datasets Resampling Through Self Organizing Maps and Genetic Algorithms

Marco Vannucci<sup>(✉)</sup> and Valentina Colla

TeCIP Institute, Scuola Superiore Sant'Anna, Pisa, Italy

{m.vannucci,v.colla}@santannapisa.it

<https://www.santannapisa.it/it/istituto/tecip-tecip-institute>

**Abstract.** The paper presents a novel approach for the resampling of imbalanced datasets aiming at the improvement of classifiers performance. The method exploits two self-organizing-maps for the determinations of the clusters of majority and minority data. Clusters centroids are used to select the samples whose under-sampling or over-sampling is more convenient while the optimal resampling rates are determined through a genetic algorithm that maximizes the classifier performance. The algorithm is tested on several datasets coming from both the UCI repository and real industrial applications and compared to other widely used resampling methods.

**Keywords:** Imbalanced datasets · Resampling · Self organizing maps · Genetic Algorithms

## 1 Introduction

The problem of the detection of rare patterns through data-driven techniques is common to different fields. Many practical tasks envisage the identification of uncommon samples by means of a classifier trained by using a dataset that, due to the nature of the observed phenomenon, is imbalanced [22]. Examples of such situation can be found in the industrial field in machine fault or defect identification [2, 18], in medical applications for the diagnosis of particular pathologies, in finance for the recognition of fraudulent transactions. All these applications focus on the correct identification of the rare situation that, in the specific contexts, is the one of interest: defects have to be avoided in order to preserve product quality, machine faults have to be spotted to limit their consequences, diseases must be correctly diagnosed. These problems also share the fact that the misclassification of frequent patterns (the so-called *false alarms*) is strongly preferable than the missed detection of rare ones.

Multiple interacting factors make hard the satisfactory classification of imbalanced datasets by employing standard classifiers. First of all the basic assumption of even distribution among classes which is implicitly done by classifiers

such as Artificial Neural Networks (ANN) [3], Decision Trees (DT) and Support Vector Machines (SVM). The goal of most machine learning algorithms is the achievement of an optimal *overall* performance, that is satisfied in the case of balanced classes but, in the case of imbalanced datasets, standard classifiers result to be biased toward the majority class and, as a consequence, the minority class is neglected [10]. An additional element that contributes to the difficulty of the task is the complexity of the classification problem: in facts, in presence of complex decision boundaries and highly overlapping classes the mission of the learner becomes harder and standard classifier tend to solve conflicts in favor of majority class [8]. The complexity of the classification of imbalanced datasets was demonstrated to be proportional to the imbalance degree [8] and the low quality (i.e. presence of noise and outliers) of data that prevent the classifier to correctly represent the different classes either due to the low number of available data or their reliability.

In this paper a method based on the combined use of Self–Organizing–Maps (SOM) and Genetic Algorithms (GA) for the pre–processing of unbalanced datasets is presented. This method tries to efficiently merge the characteristics of main approaches into an optimization context for the maximization of classifier performance. The organization of the paper is the following one: Sect. 2 introduces the main families of methods designed for dealing with class imbalance. In Sect. 3 the main characteristics of the proposed algorithm are described while the performance it achieves on a set of tests involving imbalanced dataset are reported in Sect. 4. Finally, in Sect. 5 conclusions are drawn and future perspectives of the proposed approach are outlined.

## 2 Facing Class Imbalance in Classification Tasks

The relevance of the problems described in previous section led to the development of several algorithms designed to its solution. These methods are traditionally categorized according to the way they try to counterbalance the class imbalance: on one hand, the *internal* ones are based on the creation or modification of algorithms for that specific purpose; on the other hand, the *external* methods modify the training dataset by reducing the imbalance ratio before it is used for the tuning of an arbitrary classifier.

Internal methods include those algorithm that *directly* promote the detection of rare samples. They are often designed to solve specific problems thus they often lack of generality and portability. The *Cost–Sensitive Learning* (CSL) [7] technique belongs to this family. In these approaches the training of the classifier is pursued according to an asymmetric misclassification matrix that attributes a higher weight to the missed detection of rare patterns with respect to false alarms (whose number generally rises when using this technique). CSL can be employed for the training of many standard classifiers. In [17] for instance the LASCUS method exploits a CSL approach within a fuzzy inference system to assign the clusters determined through a SOM to the frequent and rare classes, favoring the latter one. In [15] an asymmetric cost matrix is used during the training of a DT for the choice of the split attribute of the nodes of the tree.

Many literature works are based on the adaptation of existing algorithms to cope with datasets unbalance. An ad-hoc Radial Basis Function network with rectangular activation function in the hidden layer and thus named *RecBF* was proposed in [16]: the RecBF achieved interesting results in terms of detection of rare patterns due to precision in the determination of the boundary of the input regions associated to rare samples. Similarly, in [23] a customized SVM that employs a fuzzy inference system expressly designed for the identification of rare patterns is presented. Several works also use ensemble-methods (EM) such as [9] where a boosting technique is implemented for the growing of the ensemble: at each step a new learner is added and trained by using only samples belonging to the rare class. Another approach based on the use of different aggregation weights encouraging the detection of infrequent patterns is proposed in [24].

## 2.1 Focus on External Methods

The method proposed in this work belongs to external approaches. This category of approaches tries to improve classifiers performance by reducing the unbalance rate of the training dataset through the so-called *resampling*. Resampling increases the rate  $\alpha$  of rare samples with respect to the total number of observations. One of the main advantages of this category of methods is that they are extremely portable. Since they do not require modifications on the algorithm side, they can be used for the pre-processing of a training dataset to be fed to any kind of classifier. An issue is the determination of the *best* value of  $\alpha$  that mostly depends on problem characteristics and no rationale for its determination exists and thus must be often empirically set. Resampling can take place either by removing frequent samples – the so-called *under-sampling* – or adding rare ones – the so-called *over-sampling*. None of these two approaches, that can be combined, is prevalent for all the problems [1] and both of them have some criticality.

It is possible to apply a wide variety of strategies for *over* and *under* sampling. The most basic one consists in the *random* selection of the samples to remove or replicate. Although this simple method can lead to interesting results, it assumes some risks. On one hand the removal can interest samples with high informative content, resulting harmful for the classifier performance. On the other hand random oversampling can lead to the formation of compact *clusters* of minority samples that reduces, instead of expanding, the area of the input space associated to the rare class, giving rise to over-fitting problems [1,8].

In order to avoid these detrimental effects, more sophisticated resampling strategies have been developed aiming to the selection of the samples whose removal or addition maximizes the benefits for the classifier. An example of *focused* oversampling is found in [11] where only the infrequent samples in proximity of boundary regions between the minority and majority classes are replicated: this selection broadens the input domain that the classifier assigns to the minority class and reduces eventual conflicts that standard classifiers would solve in favor of majority class. Another oversampling approach that goes beyond the

pure replication of existing samples is proposed in [5] with the SMOTE (Synthetic Minority Oversampling TEchnique) algorithm. This method creates synthetic unfrequent samples locating them where they likely could be (i.e. along the lines connecting two existing minority samples). The main advantage of this widely used method is the exploitation of original (although synthetic) information, which avoids the overfitting that can be caused by the pure samples replications. On the other hand, the risk that comes with SMOTE, is the possible introduction of misleading information due to the wrong positioning of the created samples, for instance, in the neighborhood of clusters of frequent samples. An evolution of the SMOTE algorithm is the SUNDO method [4] that tries to overcome this issue by calculating the placing of the newly created samples according to the samples of the other class. Focused undersampling is investigated as well in literature works. In [12] and [13] frequent samples from the regions where inter-class conflicts are more frequent are removed. This selection aims at reducing majority class redundancy without decreasing the dataset informative content. In [21] undersampling is performed by using a two SOMs that determine the centroids of the clusters of frequent and rare samples. This information is used to guide the selection of the frequent samples to be removed in order to maximize the impact of this operation selecting, for instance, samples belonging to dense majority clusters or samples in conflicting regions. Several attempts of combining *Smart–Undersampling* and *Smart–Oversampling* have been done. In particular the OGAR algorithm (Optimal GA-based Resampling) [20] combines these two approaches and optimally determines the resampling rates by using a GA. The resampling method presented in this work extends the investigation pursued in [21] by exploiting the SOM-based clusterization for both over- and under-sampling, moreover it goes beyond it by using a GA for optimizing the two resampling rates.

### 3 The Proposed Approach

In this work an approach based on the optimal combination of *over* and *under* sampling is proposed in order to simultaneously take the advantages and avoid the drawbacks of both the approaches. The two rebalancing methods are applied on the basis of the spatial distribution of original samples according to optimal intervention rates determined through a GA. In brief, this approach exploits two different SOMs that determine two distinct clusterizations of frequent and infrequent samples respectively. The outcomes of the SOMs are used in order to determine the regions of the domain where the two classes of samples are more dense. Such information is used to calculate a ranking among the frequent samples that determines their suitability for the removal and another ranking applied on a set of synthetically created infrequent samples that estimates the impact of their inclusion within a training dataset.

Given an original dataset  $D$ , characterized by an  $\alpha$  unbalance rate, the whole process can be summarized through the following points that will be discussed more in detail in the subsequent parts of this section:

1. the frequent  $D^-$  and unfrequent  $D^+$  samples of  $D$  are used to train two distinct SOMs. The two SOMs determine two sets of centroids (one for each neuron/cluster whose number is automatically determined):  $C_F$  and  $C_U$  respectively
2. a set of  $M$  synthetic unfrequent samples  $S^+$  is created by means of the SMOTE algorithm. The quantity  $M$  of these samples is determined so as to completely rebalance the training dataset  $D$
3. four ranking, two for  $D^-$  and the others for  $S^+$  are calculated for  $D^-$  and  $S^+$  samples. At the top of the rankings applied to  $D^-$  samples, the frequent samples whose elimination from the training dataset is more useful will be located whilst, at the top of the ranking of synthetic data, those samples whose inclusion in the dataset is mostly beneficial will be found
4. a GA-based optimization simultaneously determines the rates of samples
  - to remove according to the undersampling rankings
  - to addto the datasets according to the oversampling rankings
5. the so-determined rates are applied on the  $D^-$  and  $S^+$  datasets to form the final resampled training dataset  $D^*$ . Undersampling is performed by eliminating from  $D^-$  the frequent samples at the top of the undersampling ranking according to the two optimal undersampling rates. Oversampling takes place by adding to this dataset the synthetic samples at the top of the  $S^+$  samples rankings according to the two optimal oversampling rates.

For the clustering of the frequent and rare samples SOMs are used in order to exploit their capability of preserving the distribution and topology of the original data. At the end of the SOMs training, more clusters will be located in the regions of the space where more original data are placed so as to represent with more details such regions. Further, the natural spatial relationship among samples will be kept at the level of clusters: samples that are close each other will be associated to the same cluster or to a nearby one. The number of neurons of the two SOMs, that determines the number of clusters, is chosen by means of the Silhouette criterion [6] which evaluates the goodness of different clusterings on the basis of the similarity among each cluster and the data samples associated to it. In this case different hexagonal topology SOMs are evaluated, varying the number of neurons and the shape of the map through this criterion in order select the best one. The 4 criteria adopted for the resampling are based on the distances of data samples with respect to the centroids  $C_F$  and  $C_U$ . The number of these centroids is set by means of the previously described *silhouettemethod*. The criteria designed for the ranking of frequent samples  $D^-$  aim at putting to the top of the ranking those samples whose presence in the training dataset would lead to a degradation of classifier performance. The two criteria for the selection of undersampled data, calculated for each frequent sample are:

1. Average Distance from Frequent-Class Centroids (ADFC), shown in Eq. 1 and calculated for each sample  $d \in D^-$ . The higher this metric based on Euclidean distance, the more isolated a frequent sample with respect to the others is thus its elimination would free a region of the domain where the

classifier decision boundaries associated to the infrequent class would expand

$$ADFC(d) = \text{average}_{c \in C_F} (\|d - c\|) \quad (1)$$

2. Minimum Distance from Rare Centroid (MDRC), shown in Eq. 2 and calculated for each sample  $d \in D^-$ . This metric is higher for frequent samples closer to infrequent ones. The elimination of samples for which this measure is high reduces the conflicts among frequent and rare samples favoring the expansion of rare class in the domain region from which the sample is removed

$$MDRC(d) = - \min_{c \in C_R} (\|d - c\|) \quad (2)$$

The criteria that drive the ranking of the synthetically created infrequent samples grant higher evaluations for the samples in  $S^+$  whose presence within the training dataset would broaden the domain region associated to this class, possibly avoiding any conflict with the frequent class. The two criteria are implemented as follows:

1. Minimum Distance from Frequent Centroid (MDFC), depicted in Eq. 3, applies to all samples  $d \in S^+$  and is higher for those ones farther from the regions where frequent samples are more dense. The use of this criterion avoids one of the main limitations of SMOTE that, when creating an arbitrary synthetic sample, does not check for the eventual proximity of frequent samples increasing the risk of conflicts

$$MDFC(d) = \min_{c \in C_F} (\|d - c\|) \quad (3)$$

2. Location in Contended Region (LCR), calculated by Eq. 4, measures the membership of each sample in  $S^+$  to regions of the domain halfway between regions characterized by the presence of rare and frequent samples. These boundary regions, if populated by synthetic rare samples, are likely assigned to this latter class by the classifier, broadening the rare class domain without clear class conflicts

$$LCR(d) = - \left| \min_{a \in C_F} (\|d - a\|) - \min_{b \in C_R} (\|d - b\|) \right| \quad (4)$$

These four interacting criteria are at the basis of the simultaneous under- and over-sampling process of the original data. More in detail, if the training dataset  $D^*$  is initially set as  $D^* = D$ , two rates  $R_{ADCF}$  and  $R_{MDCR}$  of samples will be selected according to their respective ranking (from the tops) and removed from  $D^*$ . Contextually, two rates  $R_{MDCF}$  and  $R_{LCR}$  of synthetically created samples will be taken from their associated ranking and added to the training dataset  $D^*$ .

The optimal resampling rates associated to each ranking are determined through a GAs-based optimization in order to maximize the benefits in terms of classification performance. The GA candidate solutions are coded as 4 elements

vectors ( $[R_{ADCF}, R_{MDCR}, R_{MDCF}, R_{LCR}]$ ) where each element is associated to the resampling percentage in the range [0%, 100%]. GA candidate solutions are randomly initialized and evolved through the generations by means of a *single point*-type crossover and a mutation operator that varies in a range [-5%, +5%] one random element within selected chromosomes. Selection operator is based on the roulette wheel technique, while the terminal condition is satisfied when a given number of generations of GAs is completed or the stall condition is reached. Given a candidate solution  $s$ , GAs engine operates as follows, exploiting a fitness function which is based on a decision tree (DT) although the procedure is completely independent on the classifier type:

1.  $s$  is used to build the corresponding resampled training dataset  $\widehat{D}^*$ ;
2. a DT is trained by using  $\widehat{D}^*$  via the C4.5 algorithm;
3. the performance of DT is evaluated on both the not-resampled training dataset  $D$  and an additional validation dataset  $D_{VD}$  (characterized by the same unbalance ration of  $D$ ) on the basis of Eq. 5 [19]:

$$E = 1 - \frac{\gamma TPR - FPR}{ACC + \mu FPR} \quad (5)$$

where  $TPR$  represents the True Positive Rate,  $FPR$  the False Positive Rate,  $ACC$  the overall accuracy while  $\gamma$  and  $\mu$  are two empirical parameters whose values are 0.7 and 0.3 respectively. The calculated value  $E$  varies in the range [0, 1] and is close to 0 for well performing classifiers. Equation (5) is used for the calculation of the performance  $E_{TR}$  on  $D_{TR}$  and  $E_{VD}$  on  $D_{VD}$

4. The final fitness (the lower the better) for  $s$  is calculated as

$$Fitness = E_{TR} + |E_{TR} - E_{VD}| \quad (6)$$

where the  $|E_{TR} - E_{VD}|$  term is added to take into account the effect of overfitting.

The final outcome of the GA corresponds to the optimal resampling rates that are applied for the creation of the classifier training dataset  $D^*$ .

## 4 Experimental Tests

The SOM-based resampling method proposed in this paper has been tested on several tasks that involve imbalanced datasets. In all these datasets, that come both from the UCI repository [14] and real industrial applications, the number of samples belonging to the class whose identification is more important is far low with respect to the others. The main features of the test datasets are summarized in Table 1 where the origin of each dataset is shown together with its number of samples and variables and the original unbalance rate.

A description and contextualization of the data coming from the UCI repository can be found on-line, the others, all deriving from the steel industry, are briefly described below. The *Nozzle Clogging* (NC) dataset was formed to get

**Table 1.** Main characteristics of the original datasets used in the tests.

Source	Dataset	Unb.Rate	Samples	Vars
UCI rep.	CARDATA	3.8%	1728	6
	NURSERY	2.5%	1296	8
	SATELLITE	9.7%	6435	36
Industrial	NOZZLE CLOGGING	1.2%	3756	6
	MSQ-1	24%	1915	11
	MSQ-2	0.3%	21784	9

better understanding of the occlusion phenomenon that affects ladle nozzles during the continuous casting of steel. This dataset collects for each cast various measurements from sensors and machine operating parameters to be associated to the eventual clogging occurrence. The correct detection of such patterns can avoid machine faults and improve steel quality. On the other hand the generation of some false alarms is tolerable. The two *Metal Sheet Quality* (MSQ1, MSQ2) collect data for the automatic grading of steel coils. They include the information obtained by the coils surface inspection system which are used for assessing the compliance of each product with quality standards. Since it is fundamental not to put into market defective products, the correct identification of non-complying sheets (the rare patterns) is of utmost importance.

For the sake of validity of the performed test campaign, all the original datasets used in this work have been divided, prior to any processing step, in a training and test set (70% and 30% of samples respectively), each one with the same unbalance ratio as the original dataset. The results obtained by the SOM-Based Resampling (SBR) on these classification tasks are compared to those achieved by other approaches used for imbalanced datasets resampling: random oversampling and undersampling, SMOTE oversampling, SOM-Undersampling and OGAR, further, the results obtained by using the original dataset are reported for measuring the impact of the each resampling approach.

The resampled datasets obtained by the use of each of the listed techniques is exploited for the training of a DT tuned by means of the C4.5 algorithm. The GA engine settings for the methods (OGAR and SBR) that exploit it for reaching the optimal unbalance ratio, set the population cardinality to 100 and a terminal condition that stops the search when 50 generations are completed. The other operators (i.e. crossover, selection) are the same for these approaches (already described in Sect. 3) in order to grant the comparability of the results. For the other methods that do not determine in an automatic manner the optimal under- and/or over-sampling rates, several runs of the algorithms were performed, varying the resampling rates from 5% up to 50% and the best rates have been considered.

The results achieved by all the tested approaches are summarized in the following tables. For each method the following information are provided (Tables 2, 3, 4, 5, 6 and 7):

- number of clusters (neurons of the SOM) (when applied)
- optimal under-sampling and over-sampling ratios (OptUnd and OptOve in the tables, when applied). In the case of the proposed approach these features sums up the outcomes of the associated rankings ( $R_{ADCF} + R_{MDCR}$  for the under-sampling,  $R_{MDCF} + R_{LCR}$  for over-sampling)
- unbalance ratio (optimal for OGAR and SBR, the one for which the best results were achieved on the other methods)
- ACC, TPR, FPR

**Table 2.** Results achieved by tested methods on the CARDATA dataset.

Method	Clus.	OptUnd	OptOve	Unb.	ACC	TPR	FPR
Non resampling	-	-	-	-	97	82	3
Rand. Unders.	-	-	-	8	98	92	2
Rand. Overs.	-	-	-	10	99	91	1
SMOTE	-	-	-	25	99	81	1
SOM–Und.	(16, 36)	-	-	5	99	100	1
OGAR	-	5	15	24	99	100	1
SBR	(9,15)	16	21	22	99	100	1

**Table 3.** Results achieved by tested methods on the NURSERY dataset.

Method	Clus.	OptUnd	OptOve	Unb.	ACC	TPR	FPR
Non resampling	-	-	-	-	99	83	1
Rand. Unders.	-	-	-	10	97	97	3
Rand. Overs.	-	-	-	25	99	93	1
SMOTE	-	-	-	10	99	84	1
SOM–Und.	(16, 36)	-	-	10	96	100	4
OGAR	-	47	47	48	97	99	3
SBR	(6,20)	15	22	22	97	98	3

The results achieved on the datasets coming from the UCI repository put into evidence the good performance of the proposed method if compared to standard and advanced resampling techniques. On the CARDATA and NURSERY datasets all the approaches are able to achieve a satisfactory accuracy, even the no-resampling strategy. Nevertheless the methods that perform data resampling are able to sensibly increase the TPR and keep low the FPR. On the CARDATA dataset the performance of the advanced methods are equivalent

**Table 4.** Results achieved by tested methods on the SATELLITE dataset.

Method	Clus.	OptUnd	OptOve	Unb.	ACC	TPR	FPR
Non resampling	-	-	-	-	91	55	5
Rand. Unders.	-	-	-	50	82	80	18
Rand. Overs.	-	-	-	25	91	54	5
SMOTE	-	-	-	45	90	62	7
SOM–Und.	(16, 16)	-	-	15	85	79	13
OGAR	-	33	38	40	85	88	15
SBR	(9, 25)	11	8	17	92	88	8

while on the NURSERY dataset the performance of SBR is slightly worse, in terms of TPR, with respect to the SOM–Undersampling (that does not optimize automatically the clustering it is based on) and OGAR. On the SATELLITE dataset the SBR results the best performing method: it significantly raises the TPR with respect to standard algorithms and it overcomes in overall accuracy the advanced approaches due to a noticeable reduction of FPR. It is worth to note that such results are obtained through a moderate resampling of the original dataset.

**Table 5.** Results achieved by tested methods on the NOZZLE CLOGGING dataset.

Method	Clus.	OptUnd	OptOve	Unb.	ACC	TPR	FPR
Non resampling	-	-	-	-	98	5	2
Rand. Unders.	-	-	-	20	90	52	10
Rand. Overs.	-	-	-	10	99	11	1
SMOTE	-	-	-	25	96	0	4
SOM–Und.	(16, 16)	-	-	25	84	79	16
OGAR	-	32	37	36	89	73	11
SBR	(12, 16)	26	15	18	93	76	7

The good performance of the SBR method are confirmed also on the industrial datasets where its is able to grant a high TPR, always much higher with respect to standard resampling methods and comparable to the advanced ones but with a markedly lower value of FPR: on the NOZZLE CLOGGING method it is 9% lower with respect to SOM–Undersampling (whose TPR is just 3% higher); on the MSQ–2 dataset FPR is 3% lower than OGAR (same TPR) and 23% lower of the SOM–Undersampling (whose TPR is 10% higher).

In general the resampling operated by SBR on the original dataset is less strong with respect to the other approaches that achieve similar TPR: it results in an unbalance ratio more similar to the *natural* one and which is likely due

**Table 6.** Results achieved by tested methods on the METAL SHEETS QUALITY 1 dataset.

Method	Clus.	OptUnd	OptOve	Unb.	ACC	TPR	FPR
Non resampling	-	-	-	-	86	75	7
Rand. Unders.	-	-	-	30	83	68	12
Rand. Overs.	-	-	-	25	85	67	9
SMOTE	-	-	-	50	84	67	11
SOM–Und.	(9, 25)	-	-	40	89	69	5
OGAR	-	21	19	36	86	76	10
SBR	(4, 18)	5	9	28	88	79	7

**Table 7.** Results achieved by tested methods on the METAL SHEETS QUALITY 2 dataset.

Method	Clus.	OptUnd	OptOve	Unb.	ACC	TPR	FPR
Non resampling	-	-	-	-	99	0	0
Rand. Unders.	-	-	-	2	99	4	1
Rand. Overs.	-	-	-	20	99	13	1
SMOTE	-	-	-	5	99	10	1
SOM–Und.	(9, 25)	-	-	5	71	75	29
OGAR	-	48	49	53	90	65	9
SBR	(8, 36)	45	22	29	94	65	6

to an accurate selection of the frequent samples to remove and of the synthetic samples to add (in all cases but the CARDATA this percentage is at least 10% lower). The reason of this success is that probably the adopted selection strategy reaches its goal of reducing the classification conflicts (a well known problem for SMOTE, for instance).

## 5 Conclusions and Future Works

In this paper the problem of the classification of imbalanced datasets was dealt by means of a novel resampling approach based on the use of SOMs. This kind of neural networks is used in order to characterize the clusters of both the frequent and infrequent samples. The associated centroids are then used for creating a set of rankings among frequent and synthetically created rare samples that measure the positive impact of their exclusion or inclusion respectively in a training dataset. The optimal rates of removal and addition according to each adopted criteria are determined by means of a GA that, through the fitness function, takes into account the performance of an arbitrary classifier given a candidate training dataset in the context of imbalanced classification.

The proposed approach is tested on datasets coming from literature and industrial problems that mainly focus on the identification of rare patterns. The achievements of SBR are satisfactory as it is able to improve the rate of detection of infrequent patterns generating a lower rate of false alarms with respect to other resampling methods. These results and the analysis of the under- and over-sampling behavior put into evidence the goodness of the samples selection procedures implemented by the method that is able to limit the classification conflicts encountered by other approaches.

In the future the method will be improved by including new criteria for the under- and over-sampling and increasing the efficiency of the existing ones. In addition, a control strategy for the aggregation of the criteria based on the use of a fuzzy inference system is ongoing.

## References

1. Batista, G.E.A.P.A., Prati, R.C., Monard, M.C.: A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explor. Newsl.* **6**(1), 20–29 (2004)
2. Borselli, A., Colla, V., Vannucci, M., Veroli, M.: A fuzzy inference system applied to defect detection in flat steel production. In: 2010 IEEE World Congress on Computational Intelligence, WCCI 2010 (2010)
3. Cateni, S., Colla, V., Vannucci, M.: A genetic algorithm-based approach for selecting input variables and setting relevant network parameters of a som-based classifier. *Int. J. Simul.: Syst. Sci. Technol.* **12**(2), 30–37 (2011)
4. Cateni, S., Colla, V., Vannucci, M.: A method for resampling imbalanced datasets in binary classification tasks for real-world problems. *Neurocomputing* **135**, 32–41 (2014)
5. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. *J. Artif. Int. Res.* **16**(1), 321–357 (2002)
6. De Amorim, R.C., Hennig, C.: Recovering the number of clusters in data sets with noise features using feature rescaling factors. *Inf. Sci.* **324**, 126–145 (2015)
7. Elkan, C.: The foundations of cost-sensitive learning. In: International Joint Conference on Artificial Intelligence, vol. 17, pp. 973–978. Lawrence Erlbaum Associates Ltd (2001)
8. Estabrooks, A., Jo, T., Japkowicz, N.: A multiple resampling method for learning from imbalanced data sets. *Comput. Intell.* **20**(1), 18–36 (2004)
9. Fan, W., Stolfo, S.J., Zhang, J., Chan, P.K.: AdaCost: misclassification cost-sensitive boosting. In: Proceedings of the Sixteenth International Conference on Machine Learning, ICML 1999, pp. 97–105. Morgan Kaufmann Publishers Inc., San Francisco (1999)
10. He, H., Garcia, E.A.: Learning from imbalanced data. *IEEE Trans. Knowl. Data Eng.* **21**(9), 1263–1284 (2009)
11. Japkowicz, N.: The class imbalance problem: significance and strategies. In: Proceedings of the 2000 International Conference on Artificial Intelligence (ICAI), pp. 111–117 (2000)
12. Kubat, M., Matwin, S.: Addressing the curse of imbalanced training sets: One-sided selection. In: Proceedings of the Fourteenth International Conference on Machine Learning, pp. 179–186. Morgan Kaufmann (1997)

13. Laurikkala, J.: Improving identification of difficult small classes by balancing class distribution. In: Quaglini, S., Barahona, P., Andreassen, S. (eds.) AIME 2001. LNCS (LNAI), vol. 2101, pp. 63–66. Springer, Heidelberg (2001). [https://doi.org/10.1007/3-540-48229-6\\_9](https://doi.org/10.1007/3-540-48229-6_9)
14. Lichman, M.: UCI machine learning repository (2013). <http://archive.ics.uci.edu/ml>
15. Ling, C.X., Yang, Q., Wang, J., Zhang, S.: Decision trees with minimal costs. In: Proceedings of the Twenty-first International Conference on Machine Learning, ICML 2004, p. 69. ACM, New York (2004)
16. Soler, V., Prim, M.: Rectangular basis functions applied to imbalanced datasets. In: de Sá, J.M., Alexandre, L.A., Duch, W., Mandic, D. (eds.) ICANN 2007. LNCS, vol. 4668, pp. 511–519. Springer, Heidelberg (2007). [https://doi.org/10.1007/978-3-540-74690-4\\_52](https://doi.org/10.1007/978-3-540-74690-4_52)
17. Vannucci, M., Colla, V.: Novel classification method for sensitive problems and uneven datasets based on neural networks and fuzzy logic. *Appl. Soft Comput. J.* **11**(2), 2383–2390 (2011)
18. Vannucci, M., Colla, V., Nastasi, G., Matarese, N.: Detection of rare events within industrial datasets by means of data resampling and specific algorithms. *Int. J. Simul.: Syst. Sci. Technol.* **11**(3), 1–11 (2010)
19. Vannucci, M., Colla, V., Sgarbi, M., Toscanelli, O.: Thresholded neural networks for sensitive industrial classification tasks. In: Cabestany, J., Sandoval, F., Prieto, A., Corchado, J.M. (eds.) IWANN 2009. LNCS, vol. 5517, pp. 1320–1327. Springer, Heidelberg (2009). [https://doi.org/10.1007/978-3-642-02478-8\\_165](https://doi.org/10.1007/978-3-642-02478-8_165)
20. Vannucci, M., Colla, V.: Genetic algorithms based resampling for the classification of unbalanced datasets. *Smart Innov. Syst. Technol.* **73**, 23–32 (2018)
21. Vannucci, M., Colla, V.: Self organizing maps based undersampling for the classification of unbalanced datasets. In: 2018 International Joint Conference on Neural Networks (IJCNN), pp. 1–6, July 2018
22. Vannucci, M., Colla, V.: Classification of unbalanced datasets and detection of rare events in industry: issues and solutions. In: Jayne, C., Iliadis, L. (eds.) EANN 2016. CCIS, vol. 629, pp. 337–351. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-44188-7\\_26](https://doi.org/10.1007/978-3-319-44188-7_26)
23. Wu, Y., Shen, L., Zhang, S.: Fuzzy multiclass support vector machines for unbalanced data. In: 2017 29th Chinese Control And Decision Conference (CCDC), pp. 2227–2231, May 2017
24. Yuan, Z., Bao, D., Chen, Z., Liu, M.: Integrated transfer learning algorithm using multi-source tradaboost for unbalanced samples classification. In: 2017 International Conference on Computing Intelligence and Information System (CIIS), pp. 188–195, April 2017



# Improvement of Routing in Opportunistic Communication Networks of Vehicles by Unsupervised Machine Learning

Ladislava Smítková Jankú<sup>(✉)</sup> and Kateřina Hyniová

Faculty of Information Technology, Department of Digital Design,  
Czech Technical University in Prague, Prague, Czech Republic  
[janku@ikt.cz](mailto:janku@ikt.cz)

**Abstract.** This paper deals with the problem of an application of machine learning in order to improve routing in a special class of opportunistic networks of vehicles called cluster opportunistic networks. We have proposed the hierarchical routing algorithm which combines three strategies in order to improve routing in OPNs: metric based on the node affiliation with detected OPN geographic sector, metrics based on the node affiliation with the communication community constructed in the spatio-temporal domain with time constraints and metric based on the node local encounter measure. The proposed routing scheme combines these four metrics to make decisions on message forwarding. The proposed routing method performance has been evaluated on simulation scenario and compared to Epidemic routing.

**Keywords:** Opportunistic network · Network of vehicles · Traffic simulator · Routing improvement · Unsupervised machine learning · Cluster analysis

## 1 Introduction

In our research, the communication structure of vehicles is considered to be a cluster opportunistic communication network. The cluster opportunistic communication networks are the opportunistic networks with special properties, particularly the distinct localities. The opportunistic communication networks (OPN) or delay tolerant networks (DTN) are ad-hoc networks where no assumption is made on the existence of a complete physical path between two nodes wishing to communicate [18]; the source and destination nodes needn't be connected to the same network at the same time. Nodes in OPNs disseminate messages with the “store-carry-forward” routing principle. The messages are transmitted when the node opportunistically meets the another node. The characteristics of node movement can influence message transmission. The network nodes can be mobile robots, wireless equipment carried by people, vehicles, wild animals, unmanned aerial vehicles. The opportunistic communication network does not

contain any fixed base stations or fixed routers. The routing is provided by the nodes of the network. Each node provides several functions simultaneously: it serves as (i) source node, (ii) destination node, (iii) transmission node, (iv) transportation node. As the source node, the node generates a message for the destination node. As the transmission node, the node transmits messages received from other nodes. As the transportation node, the node moves and transports messages received from other nodes.

## 2 Previous Work

In recent years, many different algorithms for routing in OPN/DTN have been proposed. Unfortunately, there is no unique taxonomy of OPN routing protocols, instead of that, several taxonomies have been proposed. Pelusi et al. [18] have adopted a hierarchical taxonomy of OPN routing protocols proposed by Zhang [17]. At the highest level of this taxonomy, the OPN routing algorithms are classified into two classes: *routing without infrastructure* and *routing with infrastructure*. The class first class contains methods designed for completely flat ad hoc networks, while the second class contains algorithms in which some form of infrastructure is used in order to opportunistically forward messages. Research presented in this paper is considered to be applicable only on OPNs which support the routing without infrastructure.

Hong et al. [28] have proposed a hierarchical taxonomy of OPN routing protocols based on routing protocol reactivity or proactivity. At the highest level, the OPN routing protocols are classified into two categories: proactive routing and reactive routing protocols. The proactive routing class contains methods which use the centralized or offline knowledge about the mobile network to make the routing decision. The reactive routing class contains methods, in which the nodes compute forwarding strategies through the contact history, without a global or predetermined knowledge. The examples of proactive routing protocols are knowledge-based routing schemes [8], RAPID [3], Routing in cyclic mobile space [13], Capacity-aware routing using throw-boxes [5], and Mobyspace [10] or ML-SOL [22]. The examples of reactive routing protocols are First Contact, Epidemic [27], PROPHET [12], Spray and wait [24], Seek and focus [26], Spray and focus [25], Bubble Rap [7], Social network-based multi-casting [4], or Island Hopping [20]. Context-based algorithms do not use flood techniques but they try to select the nodes to which the message should be forwarded.

Another approach to OPN routing protocols classification has been adopted by Moreira et al. [14–16], who have proposed a hierarchical taxonomy which is taking into account both way of message transmission and OPN social and topological features, such as contact frequency and age, resource utilization, community formation, common interests or node popularity. At the highest level, the OPN routing protocols are classified into three categories: (i) forwarding-based routing protocols, (ii) flooding-based routing protocols, and (iii) replication-based routing protocols.

Xia et al. [29] proposed a hierarchical taxonomy of OPN routing protocols primary in the context of social aware routing. At the highest level, the OPN

routing protocols are classified into two categories: (i) unicast routing and (ii) multicast routing. The unicast routing protocols are further divided into two groups: (i) community-based routing and (ii) community-independent routing. *Community-based routing* class of OPN routing protocols contains OPN routing protocols, which uses knowledge obtained from community detection and formation in order to improve routing performance. As examples of community-based routing protocols, Xia et al. [29] have discussed BUBBLE RAP [7], LocalCom, Gently and Diverse Routing. Our approach presented in this paper is community-based routing.

Ahmad et al. [1] have proposed a taxonomy, which divides OPN routing protocols into six main classes: Geographic, Link State-aware, Context-aware, Probabilistic, Optimization Based and Cross Layer routing protocols.

The simulations are conducted with the Opportunistic Network Environment (ONE) simulator [9], which has been reported previously as a simulation environment in scientific literature on OPN routing protocols. Using the ONE simulator, Li et al. [11] have studied how the selfish behaviors of nodes affect the performance of DTN multicast. They used standard mobility model available in the ONE simulator. Socievole et al. compared six different routing protocols using simulation scenario with random way-point mobility model in the simulator ONE [21]. Spaho et al. [23] conducted simulations with the ONE simulator in order to evaluate and compare the performance of four different routing protocols in a many-to-one communication opportunistic network. In [6] the simulator ONE has been used to evaluate the performance of SRAMSW routing algorithm.

### 3 Cluster Opportunistic Communication Networks

Our research is focused only on one specific class of the opportunistic networks. We call this class cluster opportunistic communication networks. The networks belonging to this class has the following properties: (1) the set of node movement destinations are mostly not selected randomly, rather they correspond to some node activity, and the node visit them repeatedly. (2) Each node moves predominantly, but not necessarily fully, in an area that is smaller than that the area of the entire opportunistic network. This area is called node territory. Node territories of different nodes can overlap. (3) There are node territories of groups of nodes, called node clusters, which do not overlap. (4) There are typical traces which the node clusters are connected through. (5) The communication distance of the nodes is short. Vehicles do not move randomly and do not visit random locations in a real world.

Each opportunistic network can be described by the temporal contact graph of nodes and by the multidimensional matrix of node coordinates in time, which represent node positions in the real-world physical area.

The OPN contact graph  $G(N, E, F(t))$  is the temporal graph, where  $N = n_1, n_2, \dots, n_j$  is the set of  $j$  opportunistic network graph vertices, while  $E = e_1, e_2, \dots, e_k$  is the set of  $k$  graph edges and  $F(t)$  is a set of edge functions ( $f1(t), f2(t), \dots$ ) which define the existence of the edge at time  $t$ . The vertices

correspond to the nodes of the OPN. If the finite opportunistic network graph exists, the existence of the OPN communication paths for all pairs of vertices can be theoretically computed analytically, however this task is consider to be a NP-hard problem, so it is computationally intractable.

The main idea of community-based routing is that the community has a strong impact on human mobility pattern. At first, the mobile nodes are grouped into communities by certain community detection algorithm. Secondly, the routing scheme is proposed and the messages are forwarded in accordance to this routing schema. In our research, we try to find communities by application of unsupervised machine learning rather than by contact graph partitioning using clique computation algorithm or modularity.

The centrality is a network property which characterizes the node importance in the network. The most recognized centralities are closeness centrality, degree centrality and betweenness centrality.

The *closeness centrality* of a vertex  $v$ , for a given graph  $G := (V, E)$  with  $N$  vertices and  $|E|$  edges, is defined as

$$C(x) = \frac{1}{\sum_y d(y, x)} \quad (1)$$

where  $d(y, x)$  is the distance between vertices  $x$  and  $y$ .

It is impossible to compute *closeness centrality* for disconnected graphs, because the distance between two nodes, which belong to two distinct components of graph, has not a finite value.

$$H(x) = \sum_{y \neq x} \frac{1}{d(y, x)} \quad (2)$$

where  $1/d(y, x) = 0$  if there is no path from  $y$  to  $x$ .

*Betweenness* is a centrality measure of a vertex within a graph, which quantifies the number of how many times the node acts as a bridge along the shortest path between two other nodes.

$$C_B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}} \quad (3)$$

where  $\sigma_{st}$  is total number of shortest paths from node  $s$  to node  $t$  and  $\sigma_{st}(v)$  is the number of those paths that pass through  $v$ .

Both betweenness and closeness centralities of all vertices in a graph involve calculating the shortest paths between all pairs of vertices on a graph. The nodes which have high values of betweenness centrality become to the most important nodes for routing in OPNs. The equations above are valid for static graphs; for temporal graphs it is necessary to compute centralities through the time development of the graph. It is computational expensive. Using centralities as routing metrics give very good results and it is involved in several routing schemes as ML-SOL, for example. The disadvantage of application particularly betweenness

centrality is increasing computational time for large temporal networks. In our work, we try to supply the centralities by learning criteria from geographic data and by constructing local communication communities.

### 3.1 Detection of OPN Geographical Sectors

This section describes the partitioning of the OPN geographical area in accordance to its geographical structure. The proposed method is based on application of unsupervised machine learning to node positions data. We used well-known k-means algorithm with Euclidean distance. At first, the geographical sectors of the OPN are computed over the spatio-temporal data. Because of the spatio-temporal nature of the data, each node can appear in more than in one cluster (but with a different time coordinate). Secondly, we compute a node affiliation into detected geographic sectors. In order to eliminate random or rare node presence in sectors, we propose a metric of node affiliation into cluster: the number of node positions inside geographical sector is divided by all the node positions in time. If this value is higher than  $\delta$ , the node is affiliate with the sector  $i$ . It is necessary set the value of  $\delta$  manually, we use the 0.05, e.g. at least five percent of all node positions must be inside the geographical sector. This value is dependent both on OPN node mobility and geographical high level structure and must be set with respect to these characteristics. For each node  $X$  we compute GSEC( $X$ ), a set of labels of geographical sectors which the node is affiliate to. Finally, we constructed a graph of the connection of clusters. The graph nodes represent clusters. The graph edges represent connection of clusters by the set of vehicles moving in an opportunistic network.

### 3.2 Construction of Communication Community in Spatio-Temporal Domain with Time Constraints

This section describes how the local communication communities are constructed from the spatio-temporal data. In opposition to method based on k-clique communities, where the communities are constructed from the k-cliques (fully connected sub-graphs) or to method based on modularity, we explore another approach based on communication contact graph construction in pre-defined time slots. In order to find communication communities, we investigated clustering method which uses the neighborhood node distance metrics, iterative node labeling and time constraints. The proposed method is based on building of community using community labels. We divided training data into time slots. The duration of each time slot is 15 min and it is divided into time windows of duration 2 s. The local communication communities are constructed for each time slot. At the beginning, each single node represents a community. For each node, the nodes in communication distance in time window are found and added to the same community (relabeling). The computation is repeated for the next time window. As the process proceeds, the number of communities in node population decreases. The iterative aspect of our approach is similar to the iterative graph partitioning method called Label Propagation Algorithm (LPA) proposed

by Raghavan et al. [19]. In opposite to Raghavan's approach, our method does not assign the vertex the label that is most prevalent in the vertex' neighborhood, but rather is looking for connected node communities and operates in spatio-temporal domain.

### 3.3 Clustering Node Positions into Locations

The measurement of node position co-ordinates with sampling frequency  $f$  generates a huge amount of data. Moreover, if the node visits the same physical location again, the measured node position co-ordinates can gently vary from the node co-ordinates obtained from the previous measurements in the same location. In order to process data from more measurements to learn a mobility pattern, it's better to recognize the presence of the node in location rather than work directly with exact node co-ordinates. For each node, we apply a variant of k-means clustering algorithm to find clusters of node positions and we call these clusters *locations*. We use the variant of k-means clustering method proposed by Ashbrook et al. [2] who have been interested in the task of finding locations from GPS data, which is some kind of locally applied k-means clustering. The k-means clustering is not applied to all node position data simultaneously: only the node positions in defined local neighborhood are clustered at each step. In image processing this variant of k-means clustering is sometimes called mean shift clustering. As result, we obtain a matrix  $T \times N$ , where  $T$  is a number of where each row represents node locations in time. These vectors are used to construct communication communities instead of original node positions vectors.

## 4 Routing Method

On the basis of data analysis described above, we proposed the hierarchical routing algorithm which combines three strategies in order to improve routing in OPNs: (i) the node affiliation with detected OPN geographic sector, (ii) the node affiliation with the communication community constructed in spatio-temporal domain with time constraints (iii) the node local encounter measure.

The proposed routing scheme has two phases: training and testing. During the testing phase, knowledge is extracted from the data as it is described above. The labels and routing tables are computed centrally during the training phase. During the testing phase, the simulation is conducted. When two nodes meet in this simulation, they exchange messages in accordance to rules defined by the proposed routing algorithm. FORWARD denotes message transmission (one message copy in a system exists), KEEP denotes that the node carrying a message takes it. COPY\_TOUT denotes epidemic routing with timeout (messages are spread by epidemic routing in local communities). GSEC( $X$ ) denotes a set of labels describing an affiliation of node  $X$  to the particular geographical sectors.  $O(A)^{\alpha_A}$  denotes the local communication community of node  $X$  for the time slot  $\alpha$ . We use time slots 15 min. Let's assume nodes A and B encounter. They start communication and in accordance to the message exchange policy, each of them

selects a set of messages to be exchanged. These messages are maintained in PLAN A list in node A and PLAN B in node B list. Then for each message the routing rules are applied. At first, the nodes compare their labels of communication communities  $C(A)$ ,  $C(B)$ ,  $C(DEST)$ , if all three nodes A, B, DEST are from the same communication community, the Local Encounter Metrics is applied. The DEST denotes the destination node of the message. The message is forwarded if the node popularity of the source node is smaller than the popularity of the node receiving message. Similar approach is used by LABEL or BUBBLE RAP for routing in local communities. If the nodes A, B and DEST are not from the same local communication community, the rule 2 is applied. Node A compares the affiliation  $GSEC(DEST)$  of DEST node to geosectors to the current meeting position. If the  $GSEC(DEST) = GSEC(\text{current position})$ , the routing scheme for searching the local communication community is applied. Otherwise, the rule 3 is applied. If B and DEST are affiliated to more common geographical sectors, the message is forwarded, otherwise node A keeps the message.

#### ROUTING ALGORITHM PSEUDOCODE

- startCommunication
- selectMessagesToExchange(PLAN A, PLAN B)
- makeDecision(node U, node V, node DEST)
- makeDecision(node V, node U, node DEST)
- messageExchange
- endCommunication

#### MAKE DECISION PROCEDURE PSEUDOCODE

##### **makeDecision (A, B, DEST)**

1. **if**  $C(A) == C(B)$  **and**  $C(A) == C(DEST)$  **then**
  - if**  $E(A) < E(B)$ 
    - then** return COPY\\_TOUT;
    - else** return KEEP;
  - end if**
  - end if**

2. if  $GSEC(DEST)$  contains the label of the geographical sector of the meeting point then Search the first time interval  $\alpha_A$  where  $O(A)^{\alpha_A} == O(DEST)^{\alpha_A}$ .

Search the first time interval  $\alpha_B$  where  $O(B)^{\alpha_B} == O(DEST)^{\alpha_B}$ .

- if  $\alpha_A == \infty$  **and**  $\alpha_B == \infty$

**then**

- if**  $O(A)^\alpha \cap O(DEST)^{\alpha+1}$  **or**  
 $O(A)^\alpha \cap O(O(DEST)^{\alpha+2})^{\alpha+1}$  **or**  
 $O(A)^\alpha \cap O(O(O(DEST)^{\alpha+3})^{\alpha+2})^{\alpha+1}$
- then** return FORWARD;
- else** return KEEP;

**end if**

**else**

- if**  $\alpha_B == \alpha$  **or**  $\alpha_B < \alpha_A$

```

        then return FORWARD;
        else return KEEP;
    end if
end if
end if

3. if  $GSEC(A) \cap GSEC(DEST) \neq \emptyset$  then
    if  $|GSEC(A) \cap GSEC(DEST)| < |GSEC(B) \cap GSEC(DEST)|$ 
        then return FORWARD;
        else return KEEP;
    end if
end if

```

#### 4.1 Message Exchange Policy

We proposed a message exchange policy, which is applied every time two nodes encounter. This policy enables to exchange a set of messages even if the message buffers of both encountering nodes are full using planning. If two node encounter, the lists of messages planned to be evaluated by the routing metrics are computed at first. The message exchange procedure is called as a part of routing algorithm.

### 5 Performance Evaluation

The proposed routing algorithm was experimentally evaluated on the data in the simulator ONE [9]. Xia et al. [29] summarize research on socially aware routing and forwarding protocols and have demonstrated that proposed methods are mainly compared with Epidemic and PROPHET routing protocols. We decided to compare our method to well-known epidemic routing protocol. Epidemic routing protocol is a flooding protocol and it is based on general broadcasting of messages. It determines an upper bound for message delivery ratio and a lower bound for message delivery delay.

#### 5.1 Performance Metrics

We adopted the following metrics to evaluate the performance of the proposed algorithm.

*Message Delivery Ratio.* It is computed as the ratio of the delivered messages to the number of all of transmitted messages. The larger values of message delivery ratio imply the better routing protocol performance.

*Overhead Cost Ratio.* It is computed as the number of transmitted messages in the network divided by the number of all created unique messages. This performance metrics reflects the efficiency of the evaluated routing protocol.

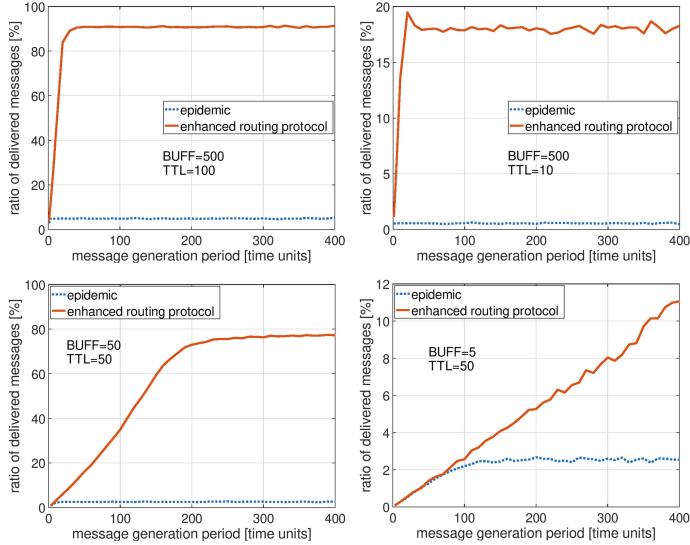
*Average Message Delivery Delay.* The lower values of average message delivery delay imply better routing protocol performance.

## 5.2 Simulation Scenario

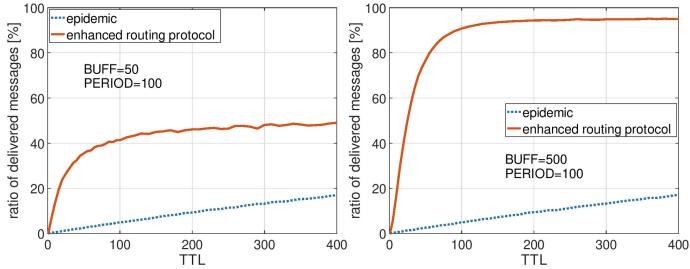
The area of the opportunistic network was generated from the road map of the city of Venice. The size of the area of the opportunistic network was  $2224\text{ m} \times 2225\text{ m}$ . We generated scenario with 129 vehicles (nodes). Each vehicle has assigned a set of destinations  $S$ . Each node moves from one destination from the set of destinations to another one, and than to another one. The traces the nodes move between destinations are generated automatically by the simulator using Dijkstra algorithm. The sets of destinations was selected in order to achieve distribution of traces typical for cluster opportunistic networks. The number of clusters was set to 6. 119 nodes have several local destinations in a part of the city. 10 vehicles have trace destination all over the area of simulation environment. We tested communication for different sizes of message buffer (the “buffer” parameter): 5, 50, 500 messages. If the buffer overflows, the messages that cannot be stored in buffer are discarded (lost). The node positions were collected each 0.1 simulation unit. The training data were collected for the time of 43200 simulation units. Messages are generated by the nodes during whole simulation. We tested the method performance for the different periods of message generation by nodes.

## 5.3 Simulation Results

The messages are injected into system as follows: each node generates a new message periodically. The number of geographical sectors was estimated by the method to 14. The number of local communication communities estimated in each time slot oscillates between 20 and 34. We conducted experiments on simulation scenario for different time periods of message generation in order to observe the influence of number of messages injected into system to the message delivery ratio. The message delivery ratio as a function of the message generation period is shown on Fig. 1. Both the proposed method and Epidemic doesn’t work well when the message generation period is to high (high number of generated messages in the system). As the message generation period decreases, we can observe a rapid improvement in the number of delivered messages for the proposed method in comparison to the epidemic routing. Figure 1 shows the relation between message delivery ratio and the parameter TTL. The proposed method outperformed Epidemic routing in overhead cost ratio: the overhead cost ration of the proposed method reaches approximately only 20% of overhead cost ratio of Epidemic routing, but this result was expected due to the flooding character of Epidemic routing. The proposed method outperform Epidemic routing also in message delivery delay performance metrics (Fig. 2).



**Fig. 1.** Ratio of delivered messages depending on the message generation period.



**Fig. 2.** Ratio of delivered messages depending on the TTL value.

## 6 Conclusion

This paper deals with the issue of an application of the unsupervised machine learning in order to improve routing for the communication in a special class of opportunistic networks of vehicles called cluster opportunistic networks. With respect to existing work in this area, we have proposed the hierarchical routing algorithm which combines three routing strategies in order to improve routing in OPNs: metric based on the node affiliation with detected OPN geographic sector, metric based on the node affiliation with the communication community constructed in spatio-temporal domain with time constraints, metric based on the node local encounter measure and the metric for a special types of nodes called geoconnect. The proposed routing scheme combines these three metrics to make decisions on message forwarding. In comparison to existing approaches, the communities are constructed only locally for predefined time slots. The advan-

tage of the proposed method of community construction is its linear computational complexity, which enables compute communication communities of large temporal networks. We experimentally verified the proposed method on a simulation scenario. An improvement in the number of delivered messages can be observed in comparison to the Epidemic routing. Future research will be focused on comparison of this method to existing social-aware methods as Bubble Rap or ML-SOL.

## References

1. Ahmad, K., Udzir, N.I., Deka, G.C.: Opportunistic Networks: Mobility Models, Protocols, Security, and Privacy. CRC Press, Boca Raton (2018)
2. Ashbrook, D., Starner, T.: Using GPS to learn significant locations and predict movement across multiple users. *Pers. Ubiquit. Comput.* **7**(5), 275–286 (2003)
3. Balasubramanian, A., Levine, B., Venkataramani, A.: DTN routing as a resource allocation problem. In: ACM SIGCOMM Computer Communication Review, vol. 37, pp. 373–384. ACM (2007)
4. Gao, W., Li, Q., Zhao, B., Cao, G.: Multicasting in delay tolerant networks: a social network perspective. In: Proceedings of the Tenth ACM International Symposium on Mobile Ad Hoc Networking and Computing, pp. 299–308. ACM (2009)
5. Gu, B., Hong, X.: Capacity-aware routing using throw-boxes. In: 2011 IEEE Global Telecommunications Conference, GLOBECOM 2011, pp. 1–5. IEEE (2011)
6. Guan, J., Chu, Q., You, I.: The social relationship based adaptive multi-spray-and-wait routing algorithm for disruption tolerant network. *Mob. Inf. Syst.* **2017** (2017)
7. Hui, P., Crowcroft, J., Yoneki, E.: BUBBLE RAP: social-based forwarding in delay-tolerant networks. *IEEE Trans. Mob. Comput.* **10**(11), 1576–1589 (2011)
8. Jain, S., Fall, K., Patra, R.: Routing in a delay tolerant network, vol. 34. ACM (2004)
9. Keranen, A., Ott, J., Kerkkäinen, T.: The one simulator for DTN protocol evaluation (2009). <http://www.scipress.org/e-library/sof/pdf/0441.PDF>. Accessed 22 Jan 2018
10. Leguay, J., Friedman, T., Conan, V.: DTN routing in a mobility pattern space. In: Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-Tolerant Networking, pp. 276–283. ACM (2005)
11. Li, Y., Su, G., Wu, D.O., Jin, D., Li, S., Zeng, L.: The impact of node selfishness on multicasting in delay tolerant networks. *IEEE Trans. Veh. Technol.* **60**(5), 2224–2238 (2011)
12. Lindgren, A., Doria, A., Schelén, O.: Probabilistic routing in intermittently connected networks. In: ACM International Symposium on MobiHoc Ad Hoc Networking and Computing, MobiHoc 2003, 01–03 June 2003 (2003)
13. Liu, C., Wu, J.: Routing in a cyclic mobispace. In: Proceedings of the 9th ACM International Symposium on Mobile ad Hoc Networking and Computing, pp. 351–360. ACM (2008)
14. Moreira, W., Mendes, P.: Social-aware opportunistic routing: the new trend. In: Woongang, I., Dhurandher, S., Anpalagan, A., Vasilakos, A. (eds.) Routing in Opportunistic Networks, pp. 27–68. Springer, New York (2013). [https://doi.org/10.1007/978-1-4614-3514-3\\_2](https://doi.org/10.1007/978-1-4614-3514-3_2)

15. Moreira, W., Mendes, P., Sargent, S.: Assessment model for opportunistic routing. In: 2011 IEEE Third Latin-American Conference on Communications, pp. 1–6. IEEE (2011)
16. Moreira, W., Mendes, P., Sargent, S.: Opportunistic routing based on daily routines. In: 2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), pp. 1–6. IEEE (2012)
17. Pelusi, L., Passarella, A., Conti, M.: Beyond MANETs: dissertation on opportunistic networking. IITCNR Technical report (2006)
18. Pelusi, L., Passarella, A., Conti, M.: Opportunistic networking: data forwarding in disconnected mobile ad hoc networks. *IEEE Commun. Mag.* **44**(11), 134–141 (2006)
19. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. *Phys. Rev. E* **76**, 036106 (2007)
20. Sarafijanovic-Djukic, N., Pidrkowski, M., Grossglauser, M.: Island hopping: efficient mobility-assisted forwarding in partitioned networks. In: 2006 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks, vol. 1, pp. 226–235. IEEE (2006)
21. Socievole, A., De Rango, F., Coscarella, C.: Routing approaches and performance evaluation in delay tolerant networks. In: 2011 Wireless Telecommunications Symposium (WTS), pp. 1–6. IEEE (2011)
22. Socievole, A., Yoneki, E., De Rango, F., Crowcroft, J.: ML-SOR: message routing using multi-layer social networks in opportunistic communications. *Comput. Netw.* **81**, 201–219 (2015)
23. Spaho, E., Bylykbashi, K., Barolli, L., Kolici, V., Lala, A.: Evaluation of different DTN routing protocols in an opportunistic network considering many-to-one communication scenario. In: 2016 19th International Conference on Network-Based Information Systems (NBiS), pp. 64–69. IEEE (2016)
24. Spyropoulos, T., Psounis, K., Raghavendra, C.S.: Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In: Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-Tolerant Networking, pp. 252–259. ACM (2005)
25. Spyropoulos, T., Psounis, K., Raghavendra, C.S.: Spray and focus: efficient mobility-assisted routing for heterogeneous and correlated mobility. In: Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PerComW 2007), pp. 79–85. IEEE (2007)
26. Spyropoulos, T., Psounis, K., Raghavendra, C.S.: Efficient routing in intermittently connected mobile networks: the multiple-copy case. *IEEE/ACM Trans. Netw. (ToN)* **16**(1), 77–90 (2008)
27. Vahdat, A., Becker, D., et al.: Epidemic routing for partially connected ad hoc networks (2000)
28. Woungang, I., Dhurandher, S.K., Anpalagan, A., Vasilakos, A.V.: Routing in Opportunistic Networks. Springer, New York (2013). <https://doi.org/10.1007/978-1-4614-3514-3>
29. Xia, F., Liu, L., Li, J., Ma, J., Vasilakos, A.V.: Socially aware networking: a survey. *IEEE Syst. J.* **9**(3), 904–921 (2015)



# Machine Learning Approach for Drone Perception and Control

Yograj S. Mandloi<sup>(✉)</sup> and Yoshinobu Inada

Tokai University, Hiratsuka-shi, Kanagawa 259-1292, Japan  
8bemmo077@mail.u-tokai.ac.jp, inada@tokai-u.jp,  
<http://www.ea.u-tokai.ac.jp/inada/>

**Abstract.** This study focuses on the application of machine learning and neural networks for the action selection and better understanding of the environment for controlling unmanned aerial vehicles, instead of explicit models to achieve the same task. Implementation of machine learning and deep learning algorithms such as non-linear regression were combined with neural networks to learn the system dynamics of a drone for the prediction of future states. Behavior cloning method is applied to mimic the actions of autopilot and comparative study of the decisions of autopilot and learned model were conducted in a simulated environment. The deep convolutional neural network was utilized for the visual perception task in the forest environment by detecting trees as obstacles. The prediction of future states and mimicking the autopilot actions were realized with relatively small error to the data from explicit model and the tree detection was successful even in the low sunlight condition.

**Keywords:** Drone perception · Supervised learning · Deep learning · Object detection

## 1 Introduction

Drones have significant potential with many practical applications like aerial delivery systems, search and rescue operation, monitoring etc. Utilizing machine learning methods for the design and development of drones can make various drone operations better and more efficient. The recent development in the computational devices and the availability of data are enabling advancement in the field of machine learning especially, deep learning. Application of deep learning for visual perception can be seen in previous works related to self-driving car [8] and drone navigating through forest trail [1] while using the classical approach to model dynamics and low-level commands of these systems. Both studies were focused on the vision and need a path to navigate. This study attempts to consider the dynamics, low level control and perception in a forest environment without any path.

As artificial neural network (ANN) works as a universal function approximator [2–4], the mapping between the current state and next states can be achieved

which is the dynamic model of the drone. Similarly, for autopilot behaviors, the sensor data of different state is mapped to appropriate action to achieve a particular goal. In the presented work, the take-off flight of the drone in a simulated environment is considered. To add visual perception for obstacle detection task such as tree detection in a forest environment, the same approach was used while utilizing a special type of artificial neural network called deep convolutional neural network (CNN). All models are an example of supervised learning, where the correct output of each input was given to the model while training.

## 2 Method

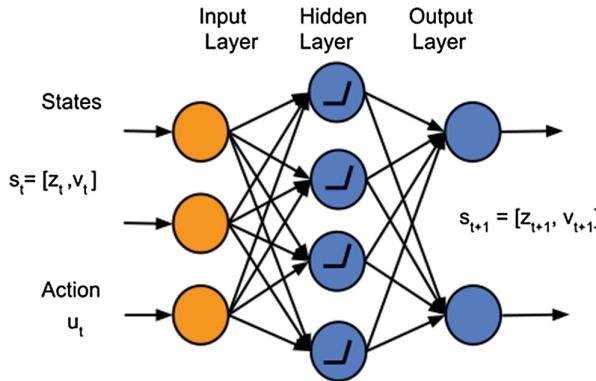
Before the learning process, the correct or actual data were given for each task and all different tasks such as learning dynamics or detection tree were optimized independently.

### 2.1 Learning the Drone Model

Instead of writing the equation of drone motion, we let the NN model to determine the dynamic behavior of drone. A 2D nonlinear regression model to predict the next state was developed based on the given current state and the action taken in that state. The deterministic time-invariant dynamic system can be defined as follows

$$\dot{x} = f(x, u) \quad (1)$$

where  $x$  is a state vector,  $u$  is the input to the system and  $f$  is a function of states and inputs.



**Fig. 1.** Neural network model for takeoff dynamics

The NN model Fig. 1, consists linear unit at input  $\mathbf{l}_{in}$  and output  $\mathbf{l}_{out}$  layers with the non linearity in the hidden layer  $\mathbf{l}_{hid}$  and parameter  $\theta = [\theta_0, \theta_1]$  depicted

by Eqs. (2)–(4) where  $\mathbf{X} = [z_t, v_t, u_t]$  is input to the NN and  $z_t, v_t, u_t$  are height, vertical velocity and thrust command at time  $t$  respectively. The output of this network gives next state as  $\mathbf{s}_{t+1} = [z_{t+1}, v_{t+1}]$ .

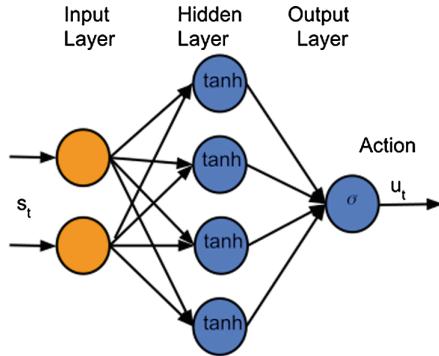
$$\mathbf{l}_{in} = \theta_0^\top \mathbf{X} \quad (2)$$

$$\mathbf{l}_{hid} = \max(\mathbf{0}, \mathbf{l}_{in}) \quad (3)$$

$$\mathbf{l}_{out} = \theta_1^\top \mathbf{l}_{hid} \quad (4)$$

## 2.2 Behavioral Cloning

To model the autopilot, a parameterized policy  $\pi_\phi(a_t | \mathbf{s}_t)$  with parameter  $\phi = [\phi_0, \phi_1]$  is defined which is a function of states  $\mathbf{s}_t = [z_t, v_t]$  and output the action  $a_t$  to be taken in that state. Two layer NN model was defined with linear operation at the input layer  $\mathbf{h}_\phi(\mathbf{s}_t)$ , hyperbolic tangent function in hidden units, and sigmoid  $\sigma$  function [5] in the output layer, Eqs. (5)–(7). Output of sigmoid function ranges from 0 to 1 which can be easily interpreted as the percentage of maximum thrust value (Fig. 2).



**Fig. 2.** Neural network model for pilot behavior

$$\mathbf{h}_\phi(\mathbf{s}_t) = \phi_0^\top \mathbf{s}_t \quad (5)$$

$$\mathbf{l}_{hid} = \tanh(\mathbf{h}_\phi(\mathbf{s}_t)) \quad (6)$$

$$\pi_\phi(a_t | \mathbf{s}_t) = \sigma(\phi_1^\top \mathbf{l}_{hid}) \quad (7)$$

where

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (8)$$

### 2.3 Visual Perception

For detection and localization of trees in an image, existing object detection models are not especially well trained on tree image data, so a new dataset was prepared, and CNN model was trained on that dataset. The first tree image data was collected by a camera and then converted into suitable data format with the correct label and the bounding box showing the correct location of the trees in the image (see Fig. 3). These images were fed into the CNN single-shot detection mobilenet architecture [7] with many units of depthwise convolution, which is operation along the third dimension of image data followed by batch normalization, and activation function which is the same operation as applied in Eq. (3).



**Fig. 3.** Image for training the CNN

### 2.4 Learning

Here learning is basically a parameter optimization process. The learning process is to find the weights of NN which minimize the objective function. The objective function for model predictive network  $J_{mp}$  is defined as the sum of squared error between predicted and actual state over training dataset with total  $T$  time steps. Equations (9)–(10) define the optimization problem for the dynamic model. For parameter update, the gradient descent algorithm Eq. (11) with backpropagation [3] and batch normalization [6] used and the optimal parameter  $\theta^*$  was found.

$$J_{mp} = \frac{1}{T} \sum_{t=1}^T \| \mathbf{s}_{t+1} - \mathbf{l}_{out} \|^2 \quad (9)$$

$$\theta^* = \operatorname{argmin}_\theta J_{mp} \quad (10)$$

$$\theta \leftarrow \theta - \alpha \nabla_{\theta} J_{mp} \quad (11)$$

Similarly, the objective function  $J_p$  is defined for learning autopilot behavior while considering the error between the autopilot command and the NN pilot command and parameter update done.

$$J_p = \frac{1}{T} \sum_{t=1}^T \|\pi_{\phi}(a_t | s_t) - u_t\|^2 \quad (12)$$

$$\phi^* = \operatorname{argmin}_{\phi} J_p \quad (13)$$

$$\phi \leftarrow \phi - \beta \nabla_{\phi} J_p \quad (14)$$

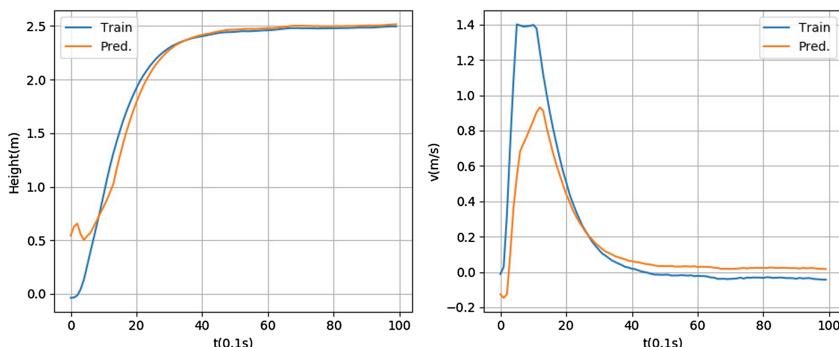
$\alpha$  and  $\beta$  are hyperparameters with value less than one.

### 3 Results

Experiments were conducted for takeoff and hover flight of drone in a simulated environment. NN models described in previous sections were built using Google open source deep learning framework Tensorflow [13]. All calculations were done offline on the ground computer however the trained model can be used on board. Take off and hover flight was simulated in ROS-gazebo environment [9,10]. An open source autopilot firmware PX4 [11] was used for generation of thrust commands for taking off and hovering. State and control command data were collected from the ground control station application Qgroundcontrol [12].

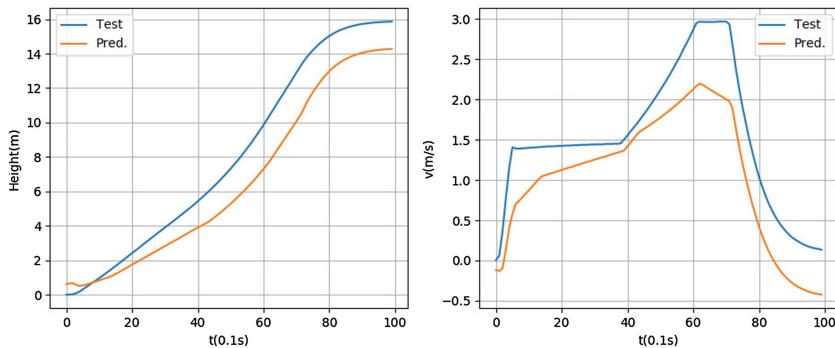
#### 3.1 Learning Drone Model

The collected height, velocity, and thrust command data from a simulated flight for takeoff used to train and test the NN model for ten seconds while each time step size was 0.1 s. For training, one step state prediction was obtained for height and velocity. Figure 4 shows the NN model prediction during the training period,



**Fig. 4.** NN model prediction while training

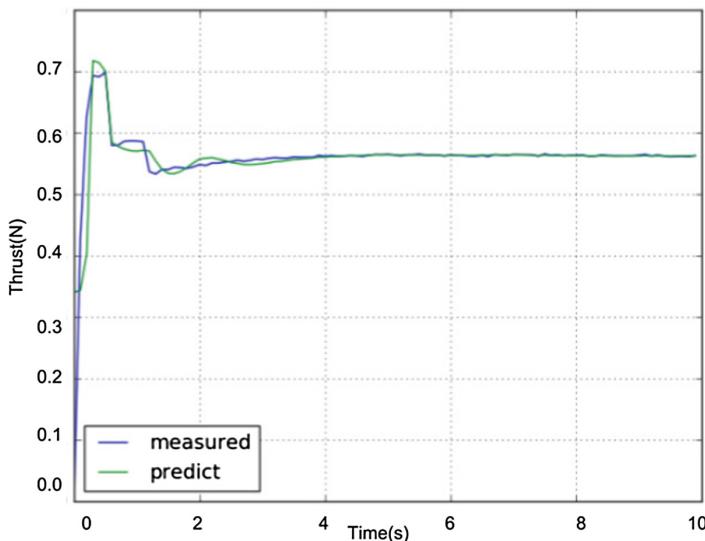
the prediction for height is very close to actual height except for the initial time of flight. Prediction in the case of testing on new states and action input data gave similar behavior with some errors as shown in Fig. 5.



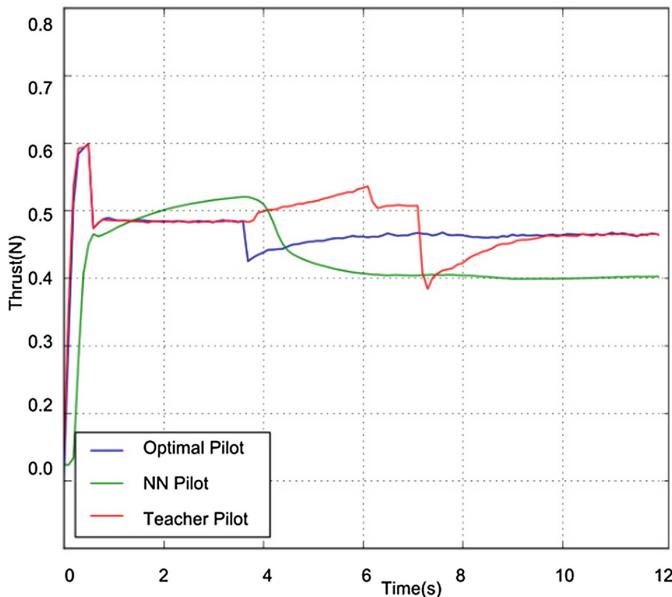
**Fig. 5.** Comparison of NN prediction with actual states

### 3.2 Mimicking Autopilot

Figure 6 shows the measured and predicted autopilot thrusts after the end of training where the input states were same with the training dataset, while Fig. 7 shows the predicted action for the new state input data.



**Fig. 6.** Neural network and autopilot thrust commands during training

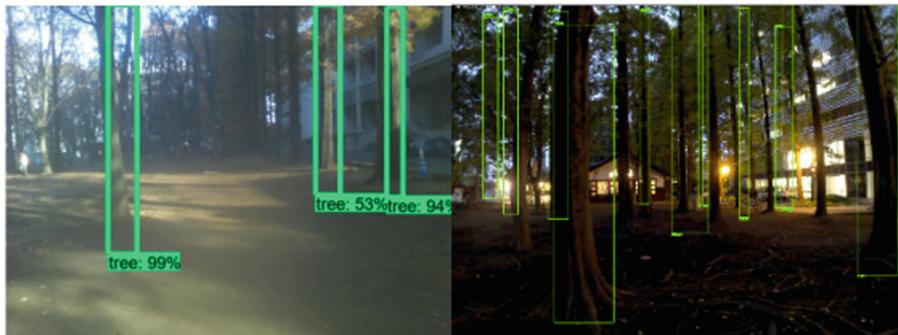


**Fig. 7.** Neural network and autopilot thrust compared with desired thrust while testing

Although the pilot NN produced smoother thrust commands than the teacher autopilot in the test case, the built NN as performed better on training data compared to the test data which needs more improvement to generate a more generalized model.

### 3.3 Tree Detection

Testing in bright and dim sunlight conditions was conducted as shown in Fig. 8. As the training data set only had images with moderate brightness, it performed better for similar distribution. In results shown below the bounding boxes predicted by CNN represents the detection of trees with the percentage confidence.



**Fig. 8.** Detection of trees in the bright and dim sunlight conditions

## 4 Future Work

In the presented study, numerical simulation results were discussed for one step prediction of states and action scoping for multiple step prediction with the application of NN. After the training process, all parameters of NN model will be fixed hence only the inference of learned model can be run on actual drone and the control decision can be made on board. Experiments with actual drone flights need to be done. Further research would be to implement the build vision model on the drone for the experimentation while making improvements in current models for better generalization. Extending the NN perception model to larger state space and machine learning approach for motion planning will be considered for further study.

## References

1. Zhilenkov, A.A., Epifantsev, I.R.: System of autonomous navigation of the drone in difficult conditions of the forest trails. In: IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (2018). <https://doi.org/10.1109/EIConRus.2018.8317266>
2. Gallant, S.I.: Perceptron-based learning algorithms. *IEEE Trans. Neural Netw.* **1**(2), 179–191 (1990)
3. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. *Nature* **323**(6088), 533–536 (1986)
4. Hornik, K.: Approximation capabilities of multilayer feedforward networks. *Neural Netw.* **4**(2), 251–257 (1991). [https://doi.org/10.1016/0893-6080\(91\)90009-T](https://doi.org/10.1016/0893-6080(91)90009-T)
5. Cybenko, G.: Approximations by superpositions of sigmoidal functions. *Math. Control Sig. Syst.* **2**(4), 303–314 (1989)
6. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
7. Howard, A.G., et al.: MobileNets: efficient convolutional neural networks for mobile vision applications. arXiv preprint [arXiv:1704.04861](https://arxiv.org/abs/1704.04861) (2017)
8. Bojarski, M., et al.: End to end learning for self-driving cars. arXiv preprint [arXiv:1604.07316v1](https://arxiv.org/abs/1604.07316v1) (2016)
9. Gazebo. <http://gazebosim.org/>
10. ROS Documentation. <http://wiki.ros.org/Documentation>
11. PX4 Development. <https://dev.px4.io/en>
12. QGroundControl. <http://qgroundcontrol.com/>
13. Google Tensorflow API. <https://www.tensorflow.org/>

# **ML - DL Financial Modeling**



# A Deep Dense Neural Network for Bankruptcy Prediction

Stamatiосs-Aggelos N. Alexandropoulos<sup>(✉)</sup>, Christos K. Aridas,  
Sotiris B. Kotsiantis, and Michael N. Vrahatis

Computational Intelligence Laboratory (CILab), Department of Mathematics,  
University of Patras, 26110 Patras, Greece  
 [{alekst,sotos,vrahatis}@math.upatras.gr](mailto:{alekst,sotos,vrahatis}@math.upatras.gr), [char@upatras.gr](mailto:char@upatras.gr),  
<http://cilab.math.upatras.gr>

**Abstract.** Bankruptcy prediction is a problem that is becoming more and more interesting. This problem concerns in particular financial and accounting researchers. Nevertheless, it is a field that gathers the focus of companies, creditors, investors and in general firms which are interested in investments or transactions. Because of a variety of parameters, such as multiple accounting ratios or many potential explanatory variables, the complexity of this problem is very high. For this reason, the probability for a company to go bankrupt or not is very difficult to be calculated. Moreover, the precise determination of the bankruptcy is a very important issue. All the above details constitute a complex problem and by taking into account the data that need to be processed, we conclude that machine learning techniques and reliable predictive models are necessary. In this paper, the effectiveness of a dense deep neural network in bankruptcy prediction relating to solvent Greek firms is tested. The experimental results showed that the provided scheme gives promising outcomes.

**Keywords:** Bankruptcy prediction · Artificial Neural Networks · Prediction models

## 1 Introduction

Bankruptcy is a vitally important problem with a variety of aspects that are particularly relevant to an economic system. There are different types of bankruptcies and their consequences in the field of business and society are many and varied [2]. Banks are very interested in this problem. Firstly, a creditor in order to approve a loan must take into account a number of parameters such as the age, the consistency in payments, proximity to any other loans, and therefore predict the probability of bankruptcy. Such decisions are very important for the development of a company and consequently for the economy of a country. If

---

Supported by Hellenic State Scholarships Foundation (IKY).

such decisions are taken very carefully, the economic system can be strengthened and business growth blossom.

Considering the financial crisis of recent years that has hit many economies in the world, it is easy to understand the importance of timely and credible bankruptcy forecasting. In addition, there is an urgent need for well-structured risk management models as well as correction associated with economic inconsistencies of a bank's customers. The economic and financial stability, as well as the healthy development of enterprises, seems directly related to the prevention of credit risk and bankruptcies in a market, as it is noted in [11].

The basic two approaches in order to predict loan default or bankruptcy are the following: (a) Structural approaches. In these approaches, the interest rates and firm attributes are examined for an outcome of the default probability and (b) Statistical approaches. These methods outcome the desirable probability based on mining the data. This paper aims through a dense deep neural network to accurately predict the possibility of bankruptcy.

The reader may reach informative reviews about the methods used to predict bankruptcy in [5] and [17]. In the first work, various standard statistical methodologies implemented on business failure are studied, while in the second one statistical and intelligent methods are presented. Furthermore, in [4] and [24] the reader can be informed for more recent survey works. Despite the fact that a variety of methods have been developed over the last ten years, there are aspects that need further study. For this reason, there is enough space for new approaches which can handle bankruptcy prediction in a more accurate way. The aim of the researchers is to provide schemes that improve existing models and ensure security and stability in business markets. The settlement of bankruptcy prediction is studying intensely [26] and new techniques have been developed in order to tackle this problem [22].

This study provides a deep dense artificial neural network for bankruptcy prediction. Based on the inherent ability of artificial neural networks in handling difficult problems such as image recognition, speech recognition etc. we test the performance of a Deep Dense Multilayer Perceptron in bankruptcy prediction task.

The rest of the paper is organized as follows. In the next section, a brief presentation of related works is provided. In Sect. 3, we describe the datasets which are used in our work. In Sect. 4 the proposed method is presented, and experimental and comparisons with our method to well-known algorithms are exhibited. Finally, the paper ends with a short discussion and some future research remarks in Sect. 5.

## 2 Related Work

The problem of timely and valid bankruptcy prediction has attracted interest not only from financial analysts or researchers in the field of economic science but also from researchers in the scientific area of Machine Learning. As it is already mentioned, the methods developed to solve this problem over the last decade are

many and varied. Indicatively, we refer the reader to *Artificial Neural Networks* [9], *instance-based learners* [1], *Decision Trees* [8], *Support Vector Machines* [28] among others.

An ensemble classifier scheme that combines well-known learners such as *Decision Trees*, *Back Propagation Neural Networks* and *Support Vector Machines* has proposed in [13] to predict bankruptcy by exploiting only the advantages of individual classifiers. This approach adopts the decision-making strategy of financial institutions where many experts are asked before the final decision is taken. Thus, everyone's opinion counts and a more complete decision is formed about whether will be given a credit, a loan or if there is a risk of a bankrupt company. In particular, the provided approach selectively combine the expected probabilities given by each classifier and the experimental results showed better performance than stacking ensemble using the weighting or voting strategy.

A comparison between several prediction models such as *Artificial Neural Networks*, *Decision Trees*, *Support Vector Machines* and *Logistic Regression* tackling the bankruptcy prediction was made in [21]. The authors taking into account the obtained experimental results and the simplicity of Decision Trees have recommended these models with the minimum support required for a rule in order to tackle the bankruptcy prediction problem.

In [25] a *meta-learning* scheme that is inspired by the stacking methodology has been proposed. This approach combines two-level classifiers to make a bankruptcy prediction. In the first level, data preprocessing takes place in order to filter noisy or unrepresentative training data. Thus, the classifiers in the second level, receive better representative training data and the prediction is more accurate. The experiments conducted by the authors showed that the proposed method exceeds stacked generalization method and also, it obtains a better prediction accuracy than neural networks, decision trees, and logistic regression.

Another study based on ensemble methods as reliable predictive models for solving the bankruptcy prediction problem has been carried out in [27]. Particularly, the authors combined *IG* based feature selection with the standard *Boosting* procedure in order to reinforce the performance of base learners. The proposed *FS-Boosting* approach compared with well-known *Bagging* and *Boosting* approaches achieved promising results and performed the best average accuracy on two of the considering bankruptcy datasets in any condition.

In [14] a recent research study about the performance of semi-supervised methods for addressing bankruptcy prediction task has been conducted. The authors include in their study well-known semi-supervised algorithms such as C4.5, *k*-Nearest Neighbors and Sequential Minimal Optimization algorithm. The experimental results showed that the semi-supervised algorithms are really competitive with the corresponding supervised algorithms.

Although the problem of accurate bankruptcy prediction is particularly important for various financial institutions, studies that were based on probabilistic models are not so many. Such a study [3] was conducted to address this issue. Specifically, in this work, Gaussian processes classifier was applied in comparison with Support Vector Machines and the Logistic Regression approach. Furthermore, an informative visualization of the conducted experiments

was presented, so that the reader can easily understand the content of their study. The experiments conducted showed that Gaussian processes can improve the classification performance and successfully deal with bankruptcy prediction.

Extensive research based on real-world datasets from American firms was conducted in [6]. The authors tested well-known Machine Learning models such as Support Vector Machines, Bagging, Boosting, and Random Forest against Logistic Regression and Artificial Neural Networks. The fundamental point of their study was the usage, of six additional complementary financial indicators, including original Altman's Z-score. This leads to superior performance by Bagging, Boosting, and Random Forest models. Moreover, the last models achieved the highest accuracy relating to all the other methods.

An algorithm, named TACD, based on the ant colony strategy proposed for predicting bankrupt and non-bankrupt in [18]. The provided model is simple and easy to use. Moreover, this method handles continuous data and thus, data discretization can be avoided. The experimental tests over three real-world datasets and in comparison with several strategies showed that the presented method provides effective results.

Recently, the inherent difficulty of automated decision systems for accurate outcomes using natural language seems to be treated through deep learning techniques [16]. Specifically, the authors tested the effectiveness of deep neural networks over a very difficult problem, the financial decision support. In this research, traditional Machine Learning approaches take part in such as Ridge regression, Random Forest, AdaBoost, Gradient Boosting. In addition, Transfer Learning Techniques were tested, such as RNN with pre-training and LSTM with both pre-training and word embeddings. The results obtained showed that deep models give reliable and accurate outcomes and in many cases are better than traditional bag-of-words models.

In [19] the importance of feature selection process in building strong prediction models is presented. Particularly, the authors studied the appropriate combination between the feature selection technique and the classification method. Thus, both filter and wrapper-based methods were studied regarding the feature selection methods. On the other hand, statistical and machine learning models were studied concerning the classification process. Furthermore, two well-known ensemble techniques, the Bagging and Boosting methods were used in order to make comparisons. This work concluded that the genetic algorithm as the wrapper-based feature selection method performs better than the filter-based one. Moreover, the combination of a genetic algorithm with naive Bayes and Support Vector Machine classifiers without bagging and boosting achieves the best prediction error rates.

In the Greek context, recently, Active Learning approaches for bankruptcy prediction problem was studied in [15]. For a more informative study about bankruptcy prediction problem as well as bankruptcy prediction models the reader is referred to [7].

### 3 Data Description

The source of the data we use comes from the National Bank of Greece and the business database containing the financial information of the companies, named ICAP. In particular, the bankruptcy deposits that we have included in our study are related to the years 2003 and 2004. In addition, the collection of financial statements for the years before the bankruptcy was taken by the ICAP database. The financial data are related to a period of three years. We denote these years as follows: (a) The bankrupt year is marked as *year 0*, (b) The year before the failure is noted as *year -1* while (c) Three years before is considered as *year -3*.

In order to build a good bankruptcy sample, we include 50 bankruptcies in the final dataset. For each bankrupt firm, we sampled two healthy firm with about the same characteristics. Thus, our sample consists of 150 individual firms and 450 firm-year observations. Due to missing financial values and ratio overlaps the

**Table 1.** The used dependent variables.

Class	Variables	Short description
Profitability	OPIMAR	Operating income divided by net sales
	NIMAR	Net income divided by sales
	GIMAR	Gross income divided by sales
	ROCE	Net income pre tax divided by capital employed
	ROE	Net income pre tax divided by shareholder's equity capital
Liquidity-Leverage	EQ/CE	Shareholder's equity to capital employed
	CE/NFA	Capital employed to net fixed assets
	TD/EQ	Total debt to shareholder's equity capital
	CA/CL	Current assets to current liabilities
	QA/CL	Quick assets to current liabilities
	WC/TA	Working capital divided by total assets
Efficiency	COLPER	Average collection period for receivables
	INVTURN	Average turnover period for inventories
	PAYPER	Average payment period to creditors
	S/EQ	Sales divided by Shareholder's equity capital
	S/CE	Sales divided by capital employed
	S/TA	Sales divided by total assets
Growth	GRTA	Growth rate of total assets $(TAt - TAt - 1)/(ABS(TAt) + ABS(TAt - 1))$
	GRNI	Growth rate of net income
	GRNS	Growth rate of net sales
Size	SIZE	Size of firm is the $\ln(\text{Total assets}/\text{GDP price index})$

**Table 2.** Distribution of bankrupted firms across 24 industries and calendar years.

Industry	Year 2003	Year 2004	Total
Advertisement	1	2	3
Agriculture and Farming	1	0	1
Clothing	2	2	4
Constructions	2	0	2
Electronics Equipment	0	1	1
Food	0	2	2
Freight Forwarding	1	0	1
Health Services	0	1	1
Industrial Minerals	0	1	1
Information Technology	0	1	1
Logistics	0	1	1
Machinery	0	2	2
Metal Products	1	0	1
Motor Vehicle Trade & Maintenance	1	0	1
Other Services	0	1	1
Plastic and Rubber	0	1	1
Private Education	1	0	1
Publishing & Printing	1	0	1
Restaurants	0	1	1
Retail Trade	3	7	10
Supermarkets	0	1	1
Telecommunications	0	2	2
Textiles	3	1	4
Wholesale Trade	2	4	6
<b>Total</b>	<b>19</b>	<b>31</b>	<b>50</b>

final input variables were measured on 21. In Table 1, there is a brief description of the financial variables included in the present research. The characteristics of bankrupt firms are exhibited in Table 2.

## 4 Proposed Method and Experimental Results

Deep Neural Networks have been successfully applied in many difficult tasks such as image processing, speech and image recognition, blueprints identification etc. In the recent years, Deep Learning attracts the interest widely and thus, Deep Networks have been used for tackling the bankruptcy problem.

In work [20] the authors studied the application of deep learning methods in bankruptcy forecasting. Specifically, two deep learning architectures were tested

and the predictions were based on textual disclosures. The experimental results showed that deep learning models give a promising framework for predicting financial outcomes.

Another deep learning technique was tested in bankruptcy prediction task in [12]. In particular, convolutional neural networks were applied to the prediction of stock price movements. In detail, a set of financial ratios are represented as a grayscale image. Thus, the network was trained and tested based on that image. The experimental results showed that the convolutional neural network has higher performance compared to other traditional methods such as Decision Trees or AdaBoost.

In our work a *Deep Dense Multilayer Perceptron (DDMP)* is applied to address bankruptcy prediction task. Neural networks with two hidden layers can represent functions with any kind of shape. In general, there is not theoretical reason to use neural networks with any more than two hidden layers with simple data sets.

Specifically, we use an artificial neural network with two hidden layers. The decision of the number of neurons in the hidden layers is a very important issue of the neural network architecture. Despite the fact that these layers do not directly interact with the external environment, they have a tremendous influence on the final output. The number of neurons in each of these hidden layers must be carefully considered. The usage of many neurons in the hidden layers can result in various problems. Firstly, too many neurons in the hidden layers may result in overfitting. Overfitting occurs when the neural network has so much information processing capacity that the limited amount of information contained in the training set is not enough to train all the neurons in the hidden layers. A second problem may occur even when the training data is sufficient. An inordinately large number of neurons in the hidden layers can increase the training time of the network. In order to secure the ability of the network to generalize, the number of neurons must be kept as low as possible. If one has a large excess of neurons, the network becomes a memory bank that can recall the training set to perfection, but it does not perform well on samples that were not in the training set.

In the first hidden layer, we used as number of neurons the  $[2/3]$  of the number of input attributes and as activation function, the *ReLU* were used. In the second hidden layer, the  $[1/3]$  of the number of input attributes were used as the number of neurons and the *ReLU* activation function was used again. Moreover, the Drop-out technique (10%) was considered and as loss function the *LOSSBinaryXENT* function was used.

Dropout is an approach to regularization in neural networks which helps reducing interdependent learning amongst the neurons. In the training phase, for each hidden layer, for each training sample and for each iteration, the dropout procedure ignores a random fraction of nodes (and the corresponding activations). Dropout forces a neural network to learn more robust features that are useful in conjunction with many different random subsets of the other neurons.

**Table 3.** AUC scores of the algorithms in our bankruptcy dataset.

	Cart	NB	LR	MP	DDMP
2 years before	0.532	0.579	0.586	0.584	<b>0.627</b>
1 year before	0.539	0.588	0.643	0.605	<b>0.664</b>
Last year	0.671	0.647	0.646	0.648	<b>0.732</b>

Dropout roughly doubles the number of iterations required to converge. However, the training time for each epoch is less.

We have compared the *DDMP* method using Keras library [10] with other well-known methods such as the *Logistic Regression*, the simple *Multilayer Perceptron* model with one hidden layer, the *Naive Bayes* approach and the *Cart* method. For, the experiments have been also used the available implementations from Scikit-learn [23]. In Table 3 the obtained results for our comparison are exhibited. We compute Area Under the Receiver Operating Characteristic Curve (AUC) because the examined dataset is imbalanced.

The experimental results showed that the proposed architecture achieves the best results.

## 5 Discussion and Concluding Remarks

Bankruptcy prediction is a difficult problem. The need of accurate intelligent predictive models is of high importance. In the last decade financial researchers and companies are in position to predict which firms will bankrupt or not. This happens due to several predicting methods which have been developed. However, more accurate models are still required.

According to Table 3, our deep dense network method performs better than other examined algorithms. Nevertheless, it should not be omitted the fact that in our study only financial ratio attributes have been used. Thus, the performance of our approach could be improved if other essential quantitative attributes would be added in the dataset.

**Acknowledgements.** S.-A. N. Alexandropoulos is co-financed by Greece and the European Union (European Social Fund-ESF) through the Operational Programme «Human Resources Development, Education and Lifelong Learning» in the context of the project “Strengthening Human Resources Research Potential via Doctorate Research” (MIS-5000432), implemented by the State Scholarships Foundation (IKY).

## References

1. Ahn, H., Kim, K.: Bankruptcy prediction modeling with hybrid case-based reasoning and genetic algorithms approach. *Appl. Soft Comput.* **9**(2), 599–607 (2009)
2. Altman, E.I., Hotchkiss, E.: *Corporate Financial Distress and Bankruptcy: Predict and Avoid Bankruptcy, Analyze and Invest in Distressed Debt*, vol. 289. Wiley, Hoboken (2010)

3. Antunes, F., Ribeiro, B., Pereira, F.: Probabilistic modeling and visualization for bankruptcy prediction. *Appl. Soft Comput.* **60**, 831–843 (2017)
4. Appiah, K.O., Chizema, A., Arthur, J.: Predicting corporate failure: a systematic literature review of methodological issues. *Int. J. Law Manag.* **57**(5), 461–485 (2015)
5. Balcaen, S., Ooghe, H.: 35 years of studies on business failure: an overview of the classic statistical methodologies and their related problems. *Br. Account. Rev.* **38**(1), 63–93 (2006)
6. Barboza, F., Kimura, H., Altman, E.: Machine learning models and bankruptcy prediction. *Expert Syst. Appl.* **83**, 405–417 (2017)
7. Chaudhuri, A., Ghosh, S.K.: Bankruptcy Prediction Through Soft Computing Based Deep Learning Technique. Springer, Heidelberg (2017). <https://doi.org/10.1007/978-981-10-6683-2>
8. Cho, S., Hong, H., Ha, B.C.: A hybrid approach based on the combination of variable selection using decision trees and case-based reasoning using the mahalanobis distance: For bankruptcy prediction. *Expert Syst. Appl.* **37**(4), 3482–3488 (2010)
9. Cho, S., Kim, J., Bae, J.K.: An integrative model with subject weight based on neural network learning for bankruptcy prediction. *Expert Syst. Appl.* **36**(1), 403–410 (2009)
10. Chollet, F., et al.: Keras (2015). <https://keras.io>
11. Erdogan, B.E.: Long-term examination of bank crashes using panel logistic regression: Turkish banks failure case. *Int. J. Stat. Probab.* **5**(3), 42 (2016)
12. Hosaka, T.: Bankruptcy prediction using imaged financial ratios and convolutional neural networks. *Expert Syst. Appl.* **117**, 287–299 (2019)
13. Hung, C., Chen, J.H.: A selective ensemble based on expected probabilities for bankruptcy prediction. *Expert Syst. Appl.* **36**(3), 5297–5303 (2009)
14. Karlos, S., Kotsiantis, S., Fazakis, N., Sgarbas, K.: Effectiveness of semi-supervised learning in bankruptcy prediction. In: 2016 7th International Conference on Information, Intelligence, Systems and Applications (IISA), pp. 1–6. IEEE (2016)
15. Kostopoulos, G., Karlos, S., Kotsiantis, S., Tampakas, V.: Evaluating active learning methods for bankruptcy prediction. In: Frasson, C., Kostopoulos, G. (eds.) Brain Function Assessment in Learning. LNCS (LNAI), vol. 10512, pp. 57–66. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-67615-9\\_5](https://doi.org/10.1007/978-3-319-67615-9_5)
16. Kraus, M., Feuerriegel, S.: Decision support from financial disclosures with deep neural networks and transfer learning. *Decis. Support Syst.* **104**, 38–48 (2017)
17. Kumar, P.R., Ravi, V.: Bankruptcy prediction in banks and firms via statistical and intelligent techniques—a review. *Eur. J. Oper. Res.* **180**(1), 1–28 (2007)
18. Lalbakhsh, P., Chen, Y.P.P.: TACD: a transportable ant colony discrimination model for corporate bankruptcy prediction. *Enterp. Inf. Syst.* **11**(5), 758–785 (2017)
19. Lin, W.C., Lu, Y.H., Tsai, C.F.: Feature selection in single and ensemble learning-based bankruptcy prediction models. *Expert Syst. Appl.* **36**, e12335 (2018)
20. Mai, F., Tian, S., Lee, C., Ma, L.: Deep learning models for bankruptcy prediction using textual disclosures. *Eur. J. Oper. Res.* **274**(2), 743–758 (2019)
21. Olson, D.L., Delen, D., Meng, Y.: Comparative analysis of data mining methods for bankruptcy prediction. *Decis. Support Syst.* **52**(2), 464–473 (2012)
22. Onan, A., et al.: A clustering based classifier ensemble approach to corporate bankruptcy prediction. *Alphanumeric J.* **6**(2), 365–376 (2018)
23. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011)

24. Pereira, V.S., Martins, V.F.: Estudos de previsão de falências-uma revisão das publicações internacionais e brasileiras de 1930 a 2015. *Revista Contemporânea de Contabilidade* **12**(26), 163–196 (2015)
25. Tsai, C.F., Hsu, Y.F.: A meta-learning framework for bankruptcy prediction. *J. Forecast.* **32**(2), 167–179 (2013)
26. Tseng, F.M., Hu, Y.C.: Comparing four bankruptcy prediction models: logit, quadratic interval logit, neural and fuzzy neural networks. *Expert Syst. Appl.* **37**(3), 1846–1853 (2010)
27. Wang, G., Ma, J., Yang, S.: An improved boosting based on feature selection for corporate bankruptcy prediction. *Expert Syst. Appl.* **41**(5), 2353–2361 (2014)
28. Yang, Z., You, W., Ji, G.: Using partial least squares and support vector machines for bankruptcy prediction. *Expert Syst. Appl.* **38**(7), 8336–8342 (2011)



# Stock Price Movements Classification Using Machine and Deep Learning Techniques-The Case Study of Indian Stock Market

Nagaraj Naik<sup>(✉)</sup> and Biju R. Mohan

Department of Information Technology,  
National Institute of Technology, Karnataka, Surathkal, India  
[it16fv04.nagaraj@nitk.edu.in](mailto:it16fv04.nagaraj@nitk.edu.in), [bijurmohan@gmail.com](mailto:bijurmohan@gmail.com)  
<http://www.nitk.ac.in>

**Abstract.** Stock price movements forecasting is an important topic for traders and stock analyst. Timely prediction in stock yields can get more profits and returns. The predicting stock price movement on a daily basis is a difficult task due to more ups and down in the financial market. Therefore, there is a need for a more powerful predictive model to predict the stock prices. Most of the existing work is based on machine learning techniques and considered very few technical indicators to predict the stock prices. In this paper, we have extracted 33 technical indicators based on daily stock price such as open, high, low and close price. This paper addresses the two problems, first is the technical indicator feature selection and identification of the relevant technical indicators by using Boruta feature selection technique. The second is an accurate prediction model for stock price movements. To predict stock price movements we have proposed machine learning techniques and deep learning based model. The performance of the deep learning model is better than the machine learning techniques. The experimental results are significant improves the classification accuracy rate by 5% to 6%. National Stock Exchange, India (NSE) stocks are considered for the experiment.

**Keywords:** ANN · Boruta feature selection · Deep learning · SVM

## 1 Introduction

Generally, the financial time series movements predictions is a difficult task due to unstable stock data which is noisy and nonlinear. The variation in policies such as economic policy, macroeconomic data, political uncertainty, and government policy are affected in the direction of the stock market. This can be reflected in stock prices and stock market fluctuated and volatile due to this reason. Classification, regression and pattern recognition problems have been solved using Artificial Neural Networks (ANN) over the years. Stock market data is the

time series data which is more volatile during day trade and it has tremendous noise. The structure of data is complex due to high dimensionality. Therefore, to make accurate decisions in stock markets, fundamental analysis, technical analysis, and artificial intelligence methods have been used by professional traders. Artificial intelligence techniques are widely used for predicting nonlinear, noisy and chaotic kind of data. In the past, most of the studies were considered data mining methods and Neural Networks (NN). Most of the existing NN work had a limitation in learning the larger amount of nonlinear, complex stock data and extracting features of larger amount data is a difficult task.

The contribution of this study can be summarized as follows. First is the technical indicator feature selection and identification of the relevant technical indicators by using Boruta feature selection techniques. The second is an accurate prediction model for stock prices.

## 2 Related Work

Zhong et al. [18] studied data mining method for forecasting stock prices on a daily basis. The study considered various financial and economic features and dimension of the feature has been reduced by techniques, namely fuzzy robust principal component analysis and KPCA. Stock data which is noisy and nonlinear, however reducing the noise could be effective while constructing the forecasting model. To accomplish this task, the integration of PCA and SVR have been proposed. In this first step, a set of technical indicators is calculated from the daily transaction data of the target stock and then PCA applied to these values aiming to extract the principal components. After filtering the principal components, a model is finally constructed to forecast the future price of the target stocks [7]. The three feature selection techniques have been discussed, namely PCA, genetic algorithms and decision trees for forecasting the stock prices [15]. Most of the literature PCA is applied for data representation and transformation. However, PCA is considered for linearly transforms the high dimension data into new low dimensional data. Therefore the KPCA method has been proposed to handle the nonlinear data by using appropriate kernel parameters [5]. Nahil et al. [12] introduced Kernel Principal Component Analysis (KPCA) to reduce the dimensions of the technical indicator feature.

Moving average convergence, divergence, and exponential moving average are stock technical indicators have been studied to identify the short term of stock prices [1]. Chourmouziadis et al. [6] addressed the problem of bull and bear market trends using fuzzy logic. Lin et al. [10] proposed PCA to reduce and filter the noise in the data. However, most of the study, PCA improves the prediction accuracy is very small. Deep learning extensively used in medical image classification, big data analysis, electronic health record analysis, Parkinson's disease diagnosis and so on [14].

### 3 Data Specification

In this paper, stock data are collected from <http://www.nseindia.com>. The data contain information about stock such as stock day open price, day low price, day high price and day close price. We have considered banking sector stock, namely ICICI Bank, Yes Bank, Kotak Bank and SBI Bank. The dataset range is obtained from the year 2009 to 2018. Each stock on the closing basis, we have assigned a stock class tag up and down by comparing the stock current price and its previous price.

### 4 Proposed Work

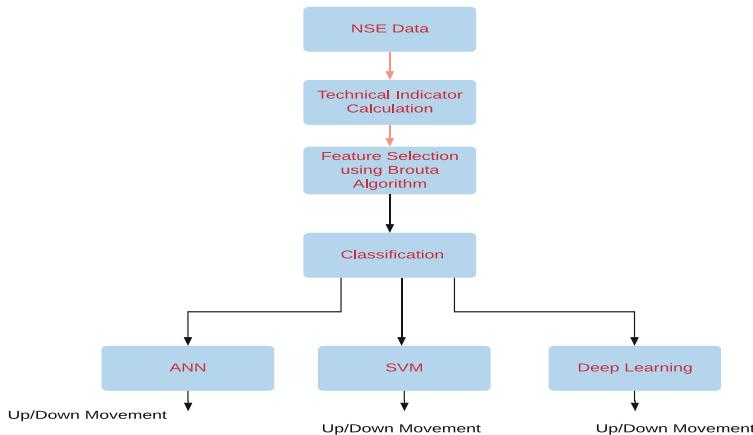
The flow of the proposed model is described in Fig. 1. The data are retrieved from NSE. The study considered 33 different combinations of technical indicators and computed based on formulas [9] which are described in Table 1.

**Table 1.** Technical indicators and its formulas Kara et al. [9]

Technical indicator name	Calculation	Number of days
Simple moving average (SMA)	$(C_t + C_{t-1} + \dots + C_{t-n+1})/n$	5, 10, 14, 30, 50, 100, 200
Exponential moving average	$(C_t - SMA(n)_{t-1}) * (2/n + 1) + SMA(n)_{t-1}$	5, 10, 14, 30, 50, 100, 200
Momentum indicator	$C_t - C_n - 9$	5, 10, 14
Stochastic oscillator	$100 * ((C_t - L_t(n))/(H_t(n) - L_t(n)))$	14
Stochastic oscillator	$(100 * ((C_t - L_t(n))/(H_t(n) - L_t(n))))/3$	14
Moving average convergence divergence	$SMA(n) - SMA(n)$	26, 13, 19, 45, 25, 15
Relative strength index	$100 - (100/(1 + Avg(Gain)/Avg(Loss)))$	14, 28
Williams R	$((H_t - C_t)/(H_n - L_n)) \times 100$	14, 28, 50, 100
Accumulation distribution index	$H_t - C_{t-1}/H_t - L_t((C_t/L)/(H/C))/(H/L)$	14
Commodity channel index	$((H + L + C/3) - SMA)/(0.015 * Mean deviation)$	14, 50, 100

#### 4.1 Technical Indicator Selection

The proposed task is carried out by using two approaches. First is Boruta feature selection method which is used to select the important feature of technical indicators. In this method it create duplicate/shadow copies of input feature to make



**Fig. 1.** Overall proposed work.

dataset as random. Random shuffling of data removes their correlations with the outcome variable. Random forest algorithm has been applied to find important technical indicator feature based on higher mean values( $Z$ ). In this algorithm, we have considered the  $Z$  score threshold value as 0.80. If any technical indicator feature has a threshold value is greater than 0.80 then it is considered for classification. The step by step proposed Boruta feature selection algorithm is stated in Algorithm 1. We have carried out this task by using Boruta package in R programming. Second task is accurate prediction model. Feature selection performed on technical indicator using Boruta algorithm and selected technical indicator feature is given as input to the prediction model.

---

#### Algorithm 1

---

- 1: Input 33 technical indicators feature F.
  - 2: Create duplicate/shadow copies of technical indicators feature D.
  - 3: Do the random Shuffle original technical indicators F and duplicate copies of technical indicator D to remove their correlations with the outcome variable.
  - 4: Apply random forest algorithm to find important technical indicator feature based on higher mean values .
  - 5: Calculate Z score by using Mean/Std deviation.
  - 6: Find the maximum Z score on duplicates technical indicator feature.
  - 7: Remove technical indicator feature if Z is less than Technical indicator feature.
- 

## 4.2 Prediction Model

**Deep Learning.** Deep learning is extensively used in medical image classification, big data analysis and electronic health record analysis. The literature

suggested that these methods gain the highest accuracy compares to the other machine learning algorithm. Kara et al. [9] has been proposed a framework for stock prediction and it used a three-layer artificial neural network. In our proposed work deep learning in H2O is implemented. Feature selection performed on technical indicator using Boruta algorithm and selected technical indicator feature is given as input to the deep learning model. The deep learning model is used to classify stock price up and down movement and it is described in Fig. 2. It has five layers of interconnected neuron units through which data is transformed. The input layer neurons represent technical indicators feature which is denoted by  $t_i$  and  $W_i$  denotes the weights of the neurons. Stochastic gradient descent with back-propagation has been used to adjust the weight. Bias input is given to each layer except the output layer of the model. The objective function  $L(W, Bias|j)$  aims is to reduce the classification error in the data.

The weighted combination of input summation is denoted in Eq. 1.

$$\alpha = \left( \sum_{i=1}^n W_i t_i + Bias \right) \quad (1)$$

The activation function Tanh and rectified linear units are used. The model supports the regularization function to avoid overfitting as shown in Eq. 2.

$$L(W, Bias|j) = L(W, Bias|j) + \lambda 1 R1(W, Bias|j) + \lambda 2 R2(W, Bias|j) \quad (2)$$

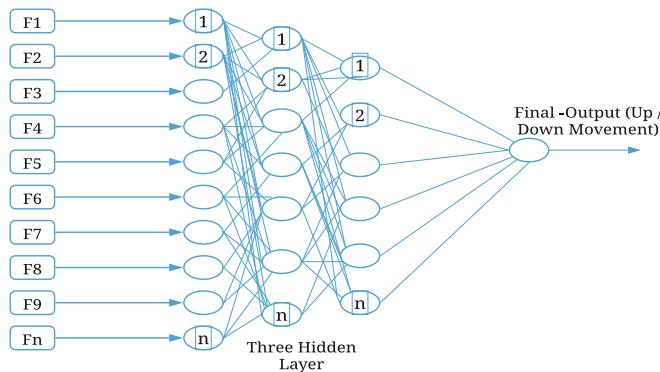
**ANN Model.** Feature selection performed on technical indicator using Boruta algorithm and selected technical indicator feature is given as input to the ANN. In this work, the ANN is used to classify the stock price. ANN has three layers, each layer is connected to the other. The neurons represent the technical indicators. The sigmoid function activation function is used in the ANN model. The threshold value 0.5 has been set. A gradient descent momentum parameters are considered to determine the weights and to reduce the global minimum.

**Support Vector Machines (SVM).** In this prediction model, we have studied SVM for two classes, namely up and down problems. SVM is based on the VC learning theory and one of its major components were developed by Vapnik [4, 16]. SVMs are also showing strong performances in real-world applications. SVM hyperplane is constructed based on input vectors. To separate the input vector two hyperplane is constructed. Stock market data are non-linear separable datasets and SVM can be more effective when datasets are non-linear.

Polynomial and radial basis kernel functions are shown in Eqs. 3 and 4.

$$PolynomialFunction : K(f_i, f_j) = (f_i \cdot f_j + 1)^d \quad (3)$$

$$RadialBasisFunction : K(f_i, f_j) = \exp(\gamma \|f_i - f_j\|^2) \quad (4)$$



**Fig. 2.** Proposed five layer deep learning model

The degree of polynomial function is represented by  $d$  and  $\gamma$  represents the constant of radial basis function. We have varied SVM parameter's degree value from 1 to 4 and gamma is 0.1 to 5 to get the best accuracy.

## 5 Experimental Results and Discussion

Each stock on the closing basis, we have assigned a stock class tag up and down by comparing the stock current price and its previous price. We have used Accuracy and F-Measure to evaluate the performance of deep and machine learning model. The Accuracy and F-measure are given in Eqs. 5 and 6. NSE datasets

**Table 2.** Result comparision

Stock	Ten technical indicators Patel et al. [13]					
	ANN		SVM		RF	
	Accuracy	F-Measure	Accuracy	F-Measure	Accuracy	F-Measure
ICICI Bank	73.12%	0.7470	68.55%	0.6935	77.12%	0.7877
SBI Bank	74.12%	0.7248	70.35%	0.7080	78.85%	0.7987
Yes Bank	72.12%	0.7414	71.35%	0.7130	77.15%	0.7638
Kotak Bank	73.12%	0.7532	72.35%	0.7210	76.35%	0.7637
Stock	Proposed model					
	ANN		SVM		Deep learning	
	Accuracy	F-Measure	Accuracy	F-Measure	Accuracy	F-Measure
ICICI Bank	79.42%	0.796	76.61%	0.769	<b>83.10%</b>	<b>0.815</b>
SBI Bank	79.60%	0.793	77.01%	0.776	<b>84.50%</b>	<b>0.824</b>
Yes Bank	78.32%	0.781	76.63%	0.753	<b>83.67%</b>	<b>0.833</b>
Kotak Bank	78.60%	0.733	77.51%	0.786	<b>83.90%</b>	<b>0.844</b>

consist of 2400 rows. We have used tenfold cross-validation in the experiment. The Experiment is carried out in the R Studio platform.

$$\text{Accuracy} = \frac{\text{TruePositive} + \text{TrueNegative}}{\text{TruePositive} + \text{TrueNegative} + \text{FalsePositive} + \text{FalseNegative}} \quad (5)$$

$$F\text{-Measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \quad (6)$$

The performance of the proposed prediction model is better than existing work and it is described in Table 2. We have fine-tuned the parameter settings to different levels to maximize the model accuracy.

## 6 Conclusion

Stock market predictions is a difficult task for stock fund managers and financial analysts due to unstable stock data which is noisy and nonlinear. The paper focused on stock price movements classification on a daily basis. We conclude that boruta feature selection is a useful method for identification of relevant technical indicators. The study also demonstrated that deep learning model performance is better than machine learning techniques. The contribution of this study can be summarized as follows. First is the technical indicator feature selection and identification of the relevant technical indicators by using Boruta feature selection techniques. The second is an accurate prediction model for stocks. The stock data is collected from the National Stock Exchange (NSE), India.

**Acknowledgment.** This work is supported by the Visvesvaraya Ph.D Scheme for Electronics and IT the departments of MeitY, Government of India. The Task carried out at the Department of IT, NITK Surathkal, Mangalore, India.

## References

1. Anbalagan, T., Maheswari, S.U.: Classification and prediction of stock market index based on fuzzy metagraph. *Procedia Comput. Sci.* **47**, 214–221 (2015)
2. Anish, C.M., Majhi, B.: Hybrid nonlinear adaptive scheme for stock market prediction using feedback flann and factor analysis. *J. Korean Stat. Soc.* **45**(1), 64–76 (2016)
3. Barak, S., Modarres, M.: Developing an approach to evaluate stocks by forecasting effective features with data mining methods. *Expert Syst. Appl.* **42**(3), 1325–1339 (2015)
4. Boser, B.E., Guyon, I.M., Vapnik, V.N.: A training algorithm for optimal margin classifiers. In: Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pp. 144–152. ACM (1992)
5. Cao, L.J., Chua, K.S., Chong, W.K., Lee, H.P., Gu, Q.M.: A comparison of PCA, KPCA and ICA for dimensionality reduction in support vector machine. *Neurocomputing* **55**(1–2), 321–336 (2003)

6. Chourmouziadis, K., Chatzoglou, P.D.: An intelligent short term stock trading fuzzy system for assisting investors in portfolio management. *Expert Syst. Appl.* **43**, 298–311 (2016)
7. Chowdhury, U.N., Rayhan, M.A., Chakravarty, S.K., Hossain, M.T.: Integration of principal component analysis and support vector regression for financial time series forecasting. *Int. J. Comput. Sci. Inf. Secur. (IJCSIS)*, **15** (2017)
8. Cortes, C., Vapnik, V.: Support-vector networks. *Mach. Learn.* **20**(3), 273–297 (1995)
9. Kara, Y., Boyacioglu, M.A., Baykan, Ö.K.: Predicting direction of stock price index movement using artificial neural networks and support vector machines: the sample of the Istanbul stock exchange. *Expert Syst. Appl.* **38**(5), 5311–5319 (2011)
10. Lin, X., Yang, Z., Song, Y.: Short-term stock price prediction based on echo state networks. *Expert Syst. Appl.* **36**(3), 7313–7317 (2009)
11. Long, W., Lu, Z., Cui, L.: Deep learning-based feature engineering for stock price movement prediction. *Knowl.-Based Syst.* **164**, 163–173 (2019)
12. Nahil, A., Lyhyaoui, A.: Short-term stock price forecasting using kernel principal component analysis and support vector machines: the case of Casablanca stock exchange. *Procedia Comput. Sci.* **127**, 161–169 (2018)
13. Patel, J., Shah, S., Thakkar, P., Kotecha, K.: Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert Syst. Appl.* **42**(1), 259–268 (2015)
14. Tahmassebi, A., Gandomi, A., McCann, I., Schulte, M., Goudriaan, A., Meyer-Baese, A.: Deep learning in medical imaging: fMRI big data analysis via convolutional neural networks. In: Proceedings of the Practice and Experience on Advanced Research Computing. ACM (2018)
15. Tsai, C.-F., Hsiao, Y.-C.: Combining multiple feature selection methods for stock prediction: union, intersection, and multi-intersection approaches. *Decis. Support Syst.* **50**(1), 258–269 (2010)
16. Vapnik, V.N., Chervonenkis, A.J.: Theory of pattern recognition (1974)
17. Weng, B., Ahmed, M.A., Megahed, F.M.: Stock market one-day ahead movement prediction using disparate data sources. *Expert Syst. Appl.* **79**, 153–163 (2017)
18. Zhong, X., Enke, D.: Forecasting daily stock market return using dimensionality reduction. *Expert Syst. Appl.* **67**, 126–139 (2017)



# Study of Stock Return Predictions Using Recurrent Neural Networks with LSTM

Nagaraj Naik<sup>(✉)</sup> and Biju R. Mohan

Department of Information Technology, National Institute of Technology,  
Surathkal, Karnataka, India  
[it16fv04.nagaraj@nitk.edu.in](mailto:it16fv04.nagaraj@nitk.edu.in), [bijurmohan@gmail.com](mailto:bijurmohan@gmail.com)  
<http://www.nitk.ac.in>

**Abstract.** Stock price returns forecasting is challenging task for day traders to yield more returns. In the past, most of the literature was focused on machine learning algorithm to predict the stock returns. In this work, the recurrent neural network (RNN) with long short term memory (LSTM) is studied to forecast future stock returns. It has the ability to keep the memory of historical stock returns in order to forecast future stock return output. RNN with LSTM is used to store recent stock information than old related stock information. We have considered a recurrent dropout in RNN layers to avoid overfitting in the model. To accomplish the task we have calculated stock return based on stock closing prices. These stock returns are given as input to the recurrent neural network. The objective function of the prediction model is to minimize the error in the model. To conduct the experiment, data is collected from the National Stock Exchange, India (NSE). The proposed RNN with LSTM model outperforms compared to an feed forward artificial neural network.

**Keywords:** Long term short memory · Recurrent neural network · Stock return

## 1 Introduction

Stock market predictions is a difficult task for stock fund managers and financial analysts due to unstable stock data which is noisy and nonlinear. The variation in policies such as economic, macroeconomic data, political uncertainty, and government policy are affected in the direction of the stock market. This impact may be reflected in stock prices and the market may be volatile. For a day trader to gain more profits, it is significant to know how to identify the quality of stocks for intraday trading. Most of the traders are not able to gain profits because they fail to select appropriate stocks to trade during the day. Hence there is a need for the short-term daily trading framework is to predict stock price. This will help investors and traders to gain the profit from the day trade. In this paper, we have proposed the recurrent neural network (RNN) with long short term memory (LSTM) to forecast future stock returns.

The related works are described in Table 1. Extensive literature suggested that most of the stock returns work is based on artificial neural network, fuzzy, simulation-based and genetic algorithm. In this paper, the recurrent neural network with LSTM is studied to forecast future stock returns. It has the ability to keep the memory of historical stock returns in order to forecast future stock return output.

**Table 1.** Related work

Author	Method	Target output	Performance metric
Enke and Mehdiyev [3]	Feature selection + fuzzy	Stock price	RMSE
Chourmouziadis and Chatzoglou [2]	Fuzzy system	Portfolio	Trading simulation composition
Qiu, Song, and Akagi [10]	ANN + genetic algorithm, simulated annealing	Stock return	MSE
Zhong and Enke [17]	Dimension reduction + ANN	Market direction (up or down)	Trading simulation, statistical tests
<b>Our study</b>	<b>RNN with LSTM</b>	<b>Stock return</b>	<b>MAE, RMSE</b>

## 2 Related Work

Existing trading rules were not gainful for future periods when the market condition changes dynamically. Chourmouziadis and Chatzoglou [2] proposed short-term technical trading strategy by considering the daily price of the stock using fuzzy systems.

An automatic way of buying and selling financial securities without the help of portfolio managers has been discussed. The combination of technical trading indicators like moving average, alpha, beta and volatility of the stock over a period of time has been proposed [1]. Nakano et al. [7] proposed a method in which non-linear financial time-series data are considered and machine learning techniques were used for predicting stock prices. Mousavi et al. [6] proposed generalized Exponential Moving Average technical indicator model to predict the stock prices. The future performance of stock indices has been studied using fuzzy time series modeling [11]. Return and risk are important objectives for managing a portfolio. Macedo et al. [5] proposed a model to enhance technical trading rule indicator based on Moving average convergence/divergence, Relative Strength Index, Bollinger Bands and Contrarian Bollinger Bands. Artificial Neural Network(ANN) has been widely used in predicting stock for financial markets. Zhang and Wu [16] proposed back propagation ANN to predict stock prices and indices.

Technical indicators like moving average, moving average convergence and divergence, relative strength index and commodity channel index have been

used to predict the stock price [14]. Performing feature extraction could help to reduce the redundant features, which can reduce the measurements, storage requirements and the running time of classifiers. It also avoids the curse of dimensionality and improves prediction performance as well as facilitate data visualization and understanding [13]. Ticknor [12] proposed artificial neural network approach to improve the prediction performance. Preis et al. [9] hypothesized that investors may use a Google hits ratio of pages are used to take the decision to predict stock price. Macroeconomic factors are believed to influence stock market movements. Machine learning methods, which are data-driven and assumption-free have become more popular in stock market prediction [15].

### 3 Data Specification

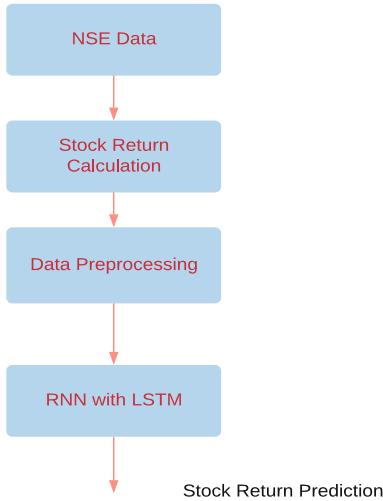
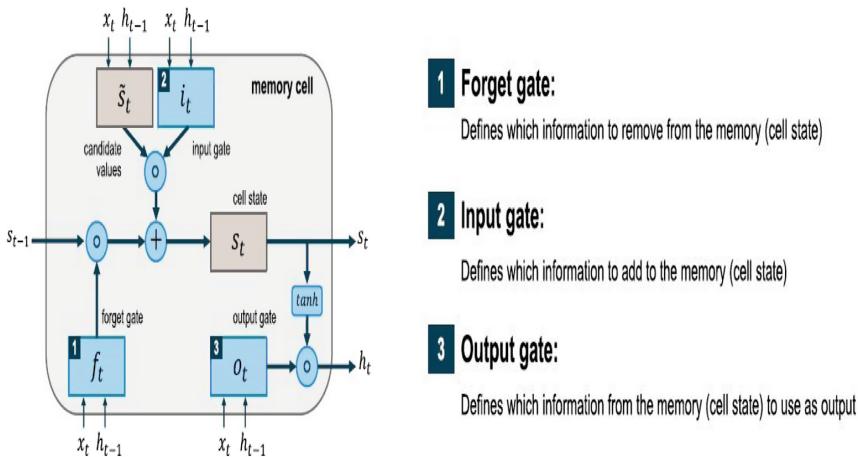
In this paper, stock data are collected from <http://www.nseindia.com>. The data contain information about stock such as stock day open price, day low price, day high price and day close price. We have considered CIPLA stock, ITC stock, TCS stock, ONGC stock and Nifty index for the experiment. The dataset range is obtained from the year 2009 to 2018.

### 4 Proposed Work

The proposed workflow of stock return forecasting is described in Fig. 1. The stock data are collected from Indian stock exchange, i.e., national stock exchange (NSE). The stock returns are calculated based on stock closing prices. Let  $C^s = C_t^s$  be defined as the closing price of stock  $s$  at time  $t$  and simple returns of stock over  $n$  period are given in Eq. 1. We standardize the stock returns by subtracting the mean and dividing them by the standard deviation. These stock returns are given as input to the recurrent neural network.

$$Return_t^{sn} = \frac{C_t^s}{C_{t-n}^s} - 1 \quad (1)$$

**RNN with LSTM.** The proposed model that takes stock return as input data from the recent past and predicts the stock returns for the next 24 h in the future. Existing literature suggested that RNN is not able to hold long-term dependencies in stock returns [4, 8]. Therefore LSTM has been proposing to capture long-term dependencies in stock returns. The LSTM organized as a cell, each cell has an internal state variable that passes information from one cell to another cell. A sigmoid layer of forget gate takes the previous output at  $t - 1$  and the present input at time  $t$  and performed concatenation operation. The output of this layer lies between 0 and 1 and it is shown in Fig. 2. If  $ft = 0$  then the internal state variable is completely forgotten, and  $ft = 1$  it will be passed through one cell to another.

**Fig. 1.** Overall proposed work.**Fig. 2.** RNN with LSTM framework for stock return forecasting [4,8].

The forget gate and input gate are given Eqs. 2, 3, 4, 5, 6 and 7.

$$f_t = \sigma(W_f \cdot [(h_{t-1} - x_t)] + b_f) \quad (2)$$

$$i_t = \sigma(W_i \cdot [h_{t-1} - x_t] + b_i) \quad (3)$$

$$C_t = \tanh(W_c \cdot [h_{t-1} - x_t] + b_c) \quad (4)$$

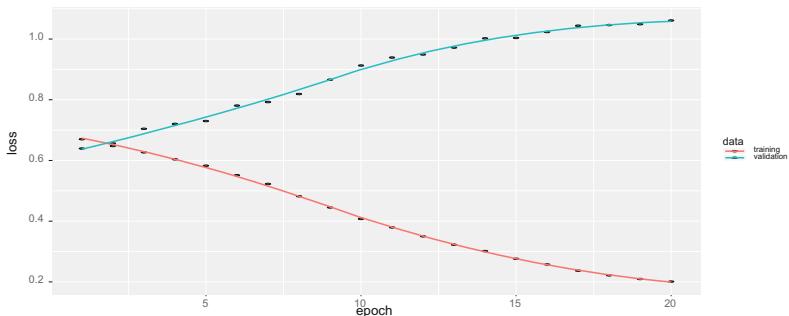
$$C_t = f_t \cdot C_{t-1} + I_t \cdot C_t \quad (5)$$

$$O_t = \sigma(W_o \cdot [h_{t-1} - x_t] + b_o) \quad (6)$$

$$h_t = O_t \cdot \tanh C_t \quad (7)$$

## 5 Experimental Results and Discussion

NSE datasets consist of 2400 rows, we have split 70% data for training and 30% for validation. The RNN model built in three layers, and we have considered the rectifier unit as the activation function. The experiment is carried out in R Studio platform. Decreasing in training rate and increasing validation rate suggests that the model is overfitting and it is described in Fig. 3. Therefore, we have added dropout function to RNN layers that avoid the overfitting problem and it is described in Fig. 4.

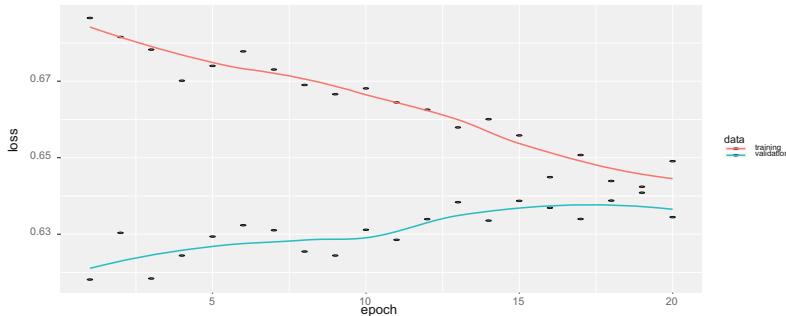


**Fig. 3.** Overfitting.

MAE and RMSE is used to evaluate the performance of the prediction model and it is described in Eqs. 8, 9. The proposed model outperforms compared to with feed forward artificial neural network(ANN) and it is shown in Table 2.

$$MAE = \frac{1}{n} \sum_{t=1}^n |e_t| \quad (8)$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{t=1}^n e_t^2} \quad (9)$$

**Fig. 4.** Adding dropout function to RNN.**Table 2.** Result comparison.

Stock name	Prediction model	MAE	RMSE
CIPLA	Feed Forward ANN	28.8583	31.52
<b>CIPLA</b>	<b>Proposed Model</b>	<b>0.1956</b>	<b>24.34</b>
ITC	Feed Forward ANN	29.7392	33.55
<b>ITC</b>	<b>Proposed Model</b>	<b>0.1741</b>	<b>23.54</b>
TCS	Feed Forward ANN	29.7392	25.89
<b>TCS</b>	<b>Proposed Model</b>	<b>0.1859</b>	<b>25.47</b>
ONGC	Feed Forward ANN	29.7392	24.56
<b>ONGC</b>	<b>Proposed Model</b>	<b>0.1645</b>	<b>23.78</b>
Nifty	Feed Forward ANN	27.35	24.87
<b>Nifty</b>	<b>Proposed Model</b>	<b>0.1895</b>	<b>25.90</b>

## 6 Conclusion

Stock price movements forecasting is challenging task for day traders to yield more returns. Recurrent neural network with LSTM is a state-of-the-art method for sequence learning. They are less commonly applied to stock return predictions. The first the recurrent neural network with LSTM is studied to forecast the future stock returns. Second, considered a recurrent dropout in RNN layers to avoid overfitting in the model. The future work can be time series forecasting of stock prices by combining technical and fundamental analysis of stocks.

**Acknowledgment.** This work is supported by the Visvesvaraya Ph.D Scheme for Electronics and IT the departments of MeitY, Government of India. The Task carried out at the Department of IT, NITK Surathkal, Mangalore, India.

## References

1. Berutich, J.M., López, F., Luna, F., Quintana, D.: Robust technical trading strategies using GP for algorithmic portfolio selection. *Expert Syst. Appl.* **46**, 307–315 (2016)
2. Chourmouziadis, K., Chatzoglou, P.D.: An intelligent short term stock trading fuzzy system for assisting investors in portfolio management. *Expert Syst. Appl.* **43**, 298–311 (2016)
3. Enke, D., Mehdiyev, N.: Stock market prediction using a combination of stepwise regression analysis, differential evolution-based fuzzy clustering, and a fuzzy inference neural network. *Intell. Autom. Soft Comput.* **19**(4), 636–648 (2013)
4. Graves, A.: Generating sequences with recurrent neural networks. arxiv preprint [arxiv: 1308.0850](https://arxiv.org/abs/1308.0850) (2013)
5. Macedo, L.L., Godinho, P., Alves, M.J.: Mean-semivariance portfolio optimization with multiobjective evolutionary algorithms and technical analysis rules. *Expert Syst. Appl.* **79**, 33–43 (2017)
6. Mousavi, S., Esfahanipour, A., Zarandi, M.H.F.: A novel approach to dynamic portfolio trading system using multitree genetic programming. *Knowl.-Based Syst.* **66**, 68–81 (2014)
7. Nakano, M., Takahashi, A., Takahashi, S.: Generalized exponential moving average (EMA) model with particle filtering and anomaly detection. *Expert Syst. Appl.* **73**, 187–200 (2017)
8. Olah, C.: Understanding LSTM networks (2015)
9. Preis, T., Moat, H.S., Stanley, H.E.: Quantifying trading behavior in financial markets using google trends. *Sci. Rep.* **3**, 01684 (2013)
10. Qiu, M., Song, Y., Akagi, F.: Application of artificial neural network for the prediction of stock market returns: the case of the Japanese stock market. *Chaos, Solitons Fractals* **85**, 1–7 (2016)
11. Rubio, A., Bermúdez, J.D., Vercher, E.: Improving stock index forecasts by using a new weighted fuzzy-trend time series method. *Expert Syst. Appl.* **76**, 12–20 (2017)
12. Ticknor, J.L.: A bayesian regularized artificial neural network for stock market forecasting. *Expert Syst. Appl.* **40**(14), 5501–5506 (2013)
13. Tsai, C.-F., Hsiao, Y.-C.: Combining multiple feature selection methods for stock prediction: union, intersection, and multi-intersection approaches. *Decis. Support Syst.* **50**(1), 258–269 (2010)
14. Tsai, C.-F., Lin, Y.-C., Yen, D.C., Chen, Y.-M.: Predicting stock returns by classifier ensembles. *Appl. Soft Comput.* **11**(2), 2452–2459 (2011)
15. Vaisla, K.S., Bhatt, A.K.: An analysis of the performance of artificial neural network technique for stock market forecasting. *Int. J. Comput. Sci. Eng.* **2**(6), 2104–2109 (2010)
16. Zhang, Y., Lenan, W.: Stock market prediction of s&p 500 via combination of improved bco approach and bp neural network. *Expert Syst. Appl.* **36**(5), 8849–8854 (2009)
17. Zhong, X., Enke, D.: Forecasting daily stock market return using dimensionality reduction. *Expert Syst. Appl.* **67**, 126–139 (2017)

## **Security - Anomaly Detection**



# Comparison of Network Intrusion Detection Performance Using Feature Representation

Daniel Pérez<sup>(✉)</sup>, Serafín Alonso, Antonio Morán, Miguel A. Prada,  
Juan José Fuentes, and Manuel Domínguez

University of León, Campus de Vegazana s/n, 24007 León, Spain  
[{dperl,saloc,a.moran,ma.prada,jj.fuentes,manuel.dominguez}@unileon.es](mailto:{dperl,saloc,a.moran,ma.prada,jj.fuentes,manuel.dominguez}@unileon.es)

**Abstract.** Intrusion detection is essential for the security of the components of any network. For that reason, several strategies can be used in Intrusion Detection Systems (IDS) to identify the increasing attempts to gain unauthorized access with malicious purposes including those base on machine learning. Anomaly detection has been applied successfully to numerous domains and might help to identify unknown attacks. However, there are existing issues such as high error rates or large dimensionality of data that make its deployment difficult in real-life scenarios. Representation learning allows to estimate new latent features of data in a low-dimensionality space. In this work, anomaly detection is performed using a previous feature learning stage in order to compare these methods for the detection of intrusions in network traffic. For that purpose, four different anomaly detection algorithms are applied to recent network datasets using two different feature learning methods such as principal component analysis and autoencoders. Several evaluation metrics such as accuracy, F1 score or ROC curves are used for comparing their performance. The experimental results show an improvement for two of the anomaly detection methods using autoencoder and no significant variations for the linear feature transformation.

**Keywords:** Anomaly detection · Feature representation · Network intrusion detection

## 1 Introduction

The great advances in network technologies entail a rise of the complexity of network attacks. For this reason, network intrusion detection plays a key role in the security of information systems and, therefore, it has become an active research area [1, 5]. In this context, intrusion can be considered as an attempt to compromise the security of a computer or network elements. It can be of

---

This research was supported by the Regional Government of Castilla y León and the European Regional Development Fund under project LE045P17.

two types: external, where unauthorized users try to gain access to the system, and internal, which are more frequent and users with different permission roles could have access to resources of the system. Moreover, different situations can be labelled as intrusions, ranging from worms that try to propagate through the network without authorization to denial of service (DoS) which focus on disrupting the resources of a system on a network. Intrusion detection systems (IDS) are devices that monitor a network in order to find any malicious activity. They are commonly classified in different types: Host-based IDS (HIDS) that analyses the internals of an individual system and Network-based IDS (NIDS) that monitors traffic between the devices of a network trying to find suspicious patterns [3]. The techniques for intrusion detection include misuse-based approaches that look for known malicious activity mostly using signatures and anomaly-based approaches which consider as an anomaly any intrusive action and would potentially detect unknown intrusions. Although both techniques have been extensively studied [5], misuse detectors are much commonly deployed in real systems.

Anomaly detection methods attempt to estimate a model of the normal behaviour of data according to a specific criteria and find patterns deviated from the resulting model [6]. These methods have been extensively applied to network intrusion detection [1,3,5]. However, their application in real scenarios has traditionally been unusual because it implies to deal with some issues [25]. For instance, network traffic presents large variability so anomalous behaviour can sometimes be related to performance, or high false positives also involve the evaluation of potential alarms which are actually normal situations.

Besides, there are other different aspects such as labelling or scaling the data that improve the success of these techniques. In addition, feature selection helps to reduce complexity and understand data interpretation. Although there are different existing strategies, representation learning and deep learning have provided enormous advances in several areas [2] such as computer vision or natural language processing.

In this work, a comparison of anomaly detection tasks is made using a feature representation of data for network intrusion detection. For that purpose, different methods of anomaly detection are compared in order to evaluate how feature transformation affects them, using four recent network datasets that provide real situations. The organization of the paper is structured as follows: in Sect. 2, different anomaly detection approaches are described and some examples for their application in intrusion detection are mentioned; in Sect. 3, the method used is illustrated; in Sect. 4, the datasets, the configuration of the experiments and the results are discussed and, finally, the conclusions are summarized in Sect. 5.

## 2 Related Work

There have been numerous efforts to survey available techniques for the implementation of anomaly detection tasks [6], specifically applied to network intrusion detection [1,3,5]. Common approaches can be grouped into categories

depending on how the method detects outliers. Next, they are briefly reviewed and some related works about network intrusion detection are mentioned.

Generally statistical-based approaches assume normal data points are generated from a Gaussian distribution. The estimation of their parameters can be sensitive to outliers so that robust estimators were proposed, like minimum covariance determinant [22]. There are examples that use statistical approaches for intrusion detection systems such as HIDE [29] which uses statistical modelling along with neural network classifiers or PAYL [28] that computes statistical parameters of the application payload, estimated from normal behaviour using a 1-gram model and then evaluated in terms of Mahalanobis distances. Other strategy to detect anomalies is based on distance-based approaches considering data instances with  $N$  features as a  $N$ -dimensional vector. For instance, One-Class Support Vector Machines (OC-SVM) constructs a hyperplane that aims to separate with a maximum margin the normal instances from the anomalous ones. Moreover, clustering methods [10] like K-means can also be used where the anomaly score is evaluated using the distance between new data points and computed centroids. Related examples for intrusion detection include Khan et al. [12], who proposed a combination of a hierarchical clustering with a SVM classification or Muda et al. [20] that computes initially K-means cluster centroids and then applies Naive Bayes classification in the final stage to distinguish between five different classes. Other proposals use ensemble-based methods such as bootstrap aggregation (bagging) or boosting that combine individual results of multiple classifiers to achieve a final decision. Similarly, Isolation Forest [15] creates an ensemble of decision trees isolating anomalies instances.

Since the performance of machine learning methods is generally affected by the number of the data dimensions, there are algorithms to select and transform data features providing another representation of the data. On one hand, irrelevant features can be eliminated in terms of information redundancy removal and accuracy improvement. Several feature selection methods have been proposed in the intrusion detection domain [7]. Some algorithms use an optimization criteria (wrapper), others compute independent features (filter) and hybrid methods try to combine both approaches for a better performance. On the other hand, feature transformation algorithms estimates a latent space that provides a new representation of the data. Dimensionality reduction techniques can be used for that purpose, like Principal Component Analysis (PCA) which computes linearly the principal components with largest variance. The use of autoencoders as dimensionality reduction tool was proposed in [9] whose low-dimensional representation can improve the performance of different tasks. Although there are other dimensionality reduction techniques, for instance those based on neighbour embeddings or spectral methods [14], the active recent research in deep learning has provided an increasing interest in approaches related to representation learning [2].

There are similar works that propose approaches related to neural networks, deep learning and anomaly detection. Previous examples include the combination of deep belief network with linear one-class SVM [8] for unsupervised

anomaly detection of high-dimensional data, discussing the performance of the hybrid model. On the other hand, a model composed by a deep autoencoder and a variant of one-class SVM using random Fourier features is introduced in [21]. An ensemble of autoencoders, called Kitsune [18], is proposed for network intrusion detection to differentiate between normal and abnormal traffic patterns. Finally, a NIDS is proposed in [11] that uses two-stage process with a sparse auto-encoder for learning features and soft-max regression, using labelled data for classification. Although these methods use networks not only for anomaly detection but also for dimensionality reduction of data features, in this work the performance of auto-encoder is studied with respect to different approaches for anomaly detection to evaluate which one is more suitable and it also uses recent datasets to test more realistic scenarios.

### 3 Proposed Method

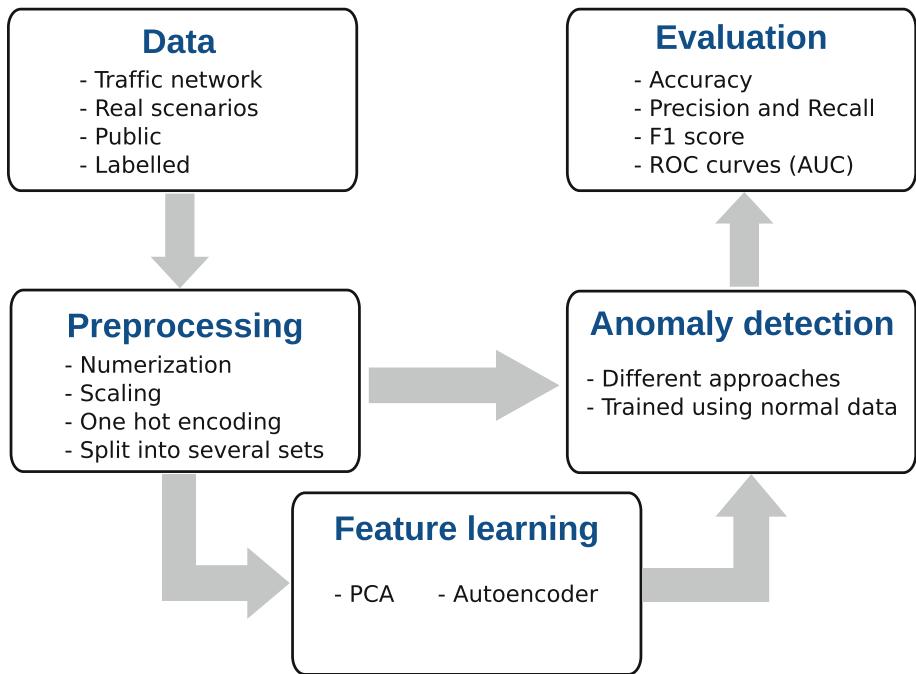
In this work, a feature learning stage is combined with well-known anomaly detection methods to detect network intrusions. Taking into account the complexity of the application area, real traffic data should be considered in order to provide more realistic scenarios for the analysis. The variables included in data are usually essential features such as protocol, service, flags, bytes between source and destination or their IP addresses and, in some cases, additional ones including statistical or aggregation measures like sum of connections or mean values. In this case, an intrusion is considered as an individual point labelled in data which is a simplification of the consequences provoked by a network attack.

A preparation stage for preprocessing data should be done. In that stage, the transformation of categorical attributes like protocol type into numeric values is performed and also normalization of data provides scaling of the features so that they are between similar ranges of values. Besides, the variables with a few unique values can be transformed into binary values using one hot encoding. Finally, data are split into train, validation and test sets.

The feature representation estimates a reduced latent space of the data by means of unsupervised learning, that is, without using data labels of the status of the network. As a baseline, the widely-used method PCA is used for reducing the dimensionality of the input data by computing a feature representation. Also, a deep auto-encoder is used with training data to compute the latent representation in the bottleneck so that the encoder provides the representation of new data. The number of the low-dimensional space is considered taking into account a trade-off between a significant reduction of the dimensionality of data without an excessive loss of information. The same number is used in the transformation of the reduced features for comparison purposes of the methods.

Once the feature transformation is done, the anomaly detection methods are trained using normal instances from the labels of the data for train and validation sets. Then the prediction of test data is performed after a data transformation into the resulting latent space. A flowchart diagram for the architecture of the method is represented in Fig. 1.

The comparison of the resulting performance is evaluated with several measures commonly used for binary classification. The well-known evaluation metrics are accuracy, precision, recall, F1 score, and Receiver Operating Characteristic (ROC) curves. The accuracy measures the fraction of the instances correctly categorized; precision denotes the proportion of true positives between all the positive predicted ones, and recall refers to the ratio of true positive between the real positive instances. Furthermore, F1 score helps to consider both precision and recall to evaluate a model computing their harmonic average. ROC curve shows false positive rates between true positive rates for several thresholds and area under curve (AUC) measures the capacity of distinguish between the two classes.



**Fig. 1.** Description of the architecture for the proposed method.

## 4 Experimental Methodology

Several experiments were performed using publicly available datasets based on real traffic for network intrusion detection. A previous feature learning stage was applied and various evaluation metrics were used in order to compare the performance.

## 4.1 Datasets

In the majority of the previous works reviewed (see Sect. 2) about network intrusion detection, only a couple of datasets are widely used for the assessment of the detection systems [1,3], i.e. DARPA 98 and 99 from MIT’s Lincoln Laboratory and KDDCup’99. However, these datasets have several shortcomings which have already been identified in the literature [16,17,27]. This leads to consider that other datasets might be more suited to evaluate the detection of contemporary network attacks. For that reason, the following recent datasets have been used in the experiments in order to consider more realistic situations:

- **UNSW-NB15** [19] possess a hybrid of the real modern normal traffic and synthesized attack activities. It was generated using an attack automatic generation tool called IXIA PerfectStorm.
- **NSL-KDD** [27] was created in order to improve the KDDCup’99 dataset. Although the dataset still suffers some problems to be considered a complete representative of modern networks, it can be used as a reference for comparison purposes because of its wide use.
- **CIC-IDS-2017** [24] covers updated attacks with more than 80 features and labelled for benign and intrusive flows. Concretely, the data used here correspond exactly to working hours of Wednesday.
- **Kyoto** [26], built on 3 years of real traffic data (Nov. 2006–Aug. 2009) which were obtained from different kinds of honeypots.

All datasets include labels about normal and different types of attacks occurred in the network which are used for training and evaluation of anomaly detection tasks. A description of the datasets such as number of instances, the attributes or dimensions obtained after one hot encoding of some of the features and the percentage of anomalies is detailed in Table 1.

**Table 1.** Description of the network datasets.

Data	Instances	Attributes	Dimensions	% of anomalies
UNSW-NB15	257673	45	230	63.9
NSL-KDD	148517	43	143	48.1
CIC-IDS-2017	691695	83	83	36.4
Kyoto	364725	24	46	82.9

**Preprocessing.** The categorical features included in data were transformed to numerical values, in some cases using a one hot encoding to obtain a set of binary variables for those features with few categories. For that reason, the number of dimensions can be augmented with respect to original attributes of data. The transformed features depend on the dataset, but there are some in common for the majority of datasets, for example service, protocol type or flag. Additionally,

a min-max scaling were made so that each feature is scaled to a range of values between [0, 1] on the training set and then also transform the validation and test data. Despite the variety of the intrusions labelled in data, they are all grouped only into one category, that is, are considered only two classes (normal and anomaly) in the analysis.

## 4.2 Experiments

First, four methods are applied for anomaly detection where only normal instances are used for training the different methods. These methods that were used are:

- **Local Outlier Factor (LOF):** [4] assigns to each object a degree about how it is isolated with respect to a specific neighbourhood. The number of the k-nearest neighbours selected after several tests is set to 60 for all datasets used in the experiments.
- **One-Class Support Vector Machine (OC-SVM):** only uses one class for estimating a model and detects new data different from that class as outliers [23]. The kernel used in this work is a radial basis function (RBF) with  $\gamma = 0.1$ , fixed experimentally.
- **Isolation Forest (IF):** creates an ensemble of trees that isolate anomalies instead of fitting normal instances, which is a different approach for outlier detection [15].
- **Robust Covariance (RC):** implements a minimum covariance determinant which is a highly robust algorithm for estimating covariance matrix in multivariate data [22].

For computing the latent representation, the dimensionality of data is reduced using either Principal Component Analysis (PCA) as the linear baseline method and the encoder obtained from an autoencoder. The design of the neural network can be essentially considered as a common deep autoencoder. The input layer has a size equal to the dimensionality of input data which is reduced using several hidden layers using rectifier linear unit (ReLU) as activation functions except for the last layer where a sigmoid function is used. The optimization stage is performed using the Adam algorithm [13]. The selected batch size is 256 and epochs for training have been set to 700, they are experimentally fixed according to the datasets used. The details for the representation learning stage are described in Table 2. The latent dimension computed using PCA has the same dimension of the bottleneck of the autoencoder for comparison purposes. The layers of the encoding were selected in order to obtain a significant reduction of the dimensionality of data.

## 4.3 Results and Discussion

The results of the experiments are presented in this section. The evaluation measures of the anomaly detection methods applied to the network datasets are

**Table 2.** Details of the feature transformation.

Data	Encoding layers	PCA dim
UNSW-NB15	{230, 120, 60, 20}	20
NSL-KDD	{143, 100, 80, 20}	20
CIC-IDS-2017	{83, 80, 40, 20}	20
Kyoto	{46, 40, 20, 5}	5

detailed in Table 3. In this table, the accuracy, precision, recall and F1 score indicate the performance of the corresponding method for each dataset, also including the previous feature learning stage using PCA and encoder network. The best resulting F1 score for each dataset is highlighted between all the methods used. Furthermore, area under curve (AUC) and ROC curves are shown in Fig. 2 in a matrix form where the rows correspond to each dataset and the columns the method applied. In case of equal F1 scores, the AUC value is considered for selecting the best one.

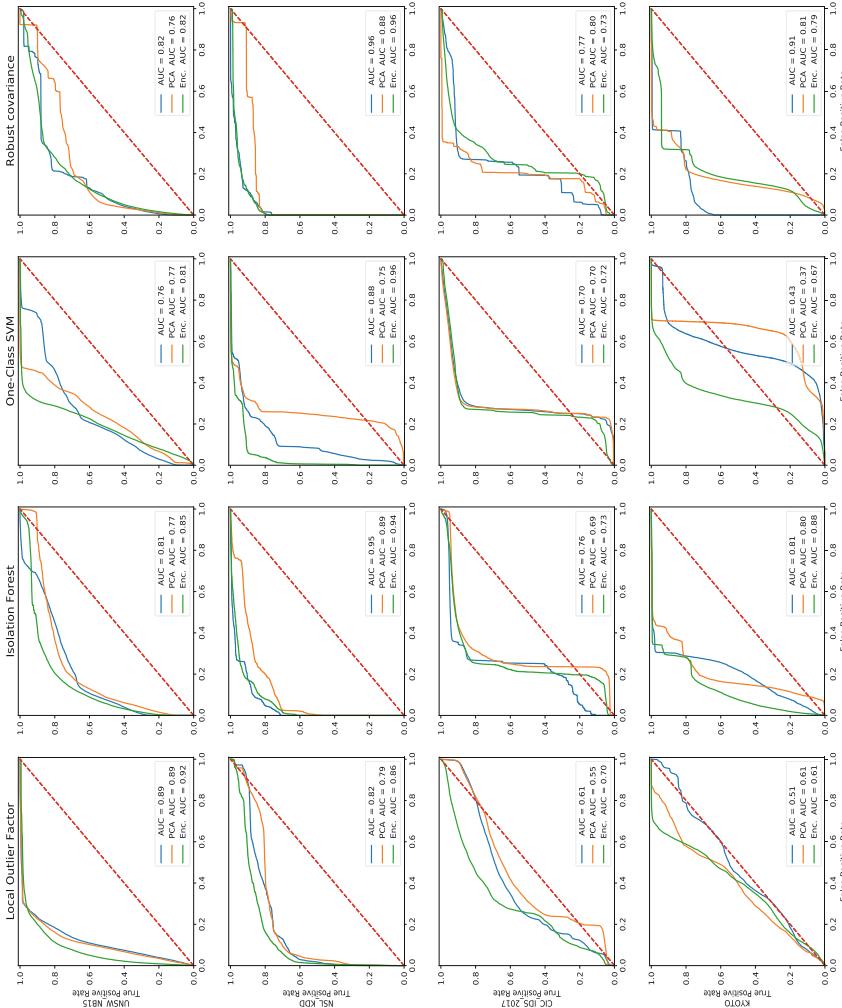
Several changes can be observed in the performance of anomaly detection tasks as a result of feature learning stage. The most significant improvement is produced using One-Class SVM method, where the use of the auto-encoder computing a feature representation shows better evaluation metrics for all datasets used in the experiments. In addition, the auto-encoder representation also produces small enhancements in the results using Local Outlier Factor, as it is shown in the Fig. 2.

However, the feature representation barely affects the effectiveness for anomaly detection using the Isolation Forest and Robust Covariance methods. There are only improvements for both methods using CIC-IDS-2017 dataset, shown by the values of F1 scores (see Table 3). Moreover, in some cases it is preferable the application of these two methods using the original data without any feature learning.

On the other hand, PCA transformation produces generally similar results to original data and, in some cases even worse than original features. There are only a few cases where the representation computed by PCA overcomes the rest. In these cases, the method used is Robust Covariance which seems to be the most suitable one to a previous PCA feature learning. This can reflect that linear techniques could only work in specific scenarios and they might be insufficient for a general type of analysis. Finally, it is remarkable that results from the experiments show in some cases a poor performance, for example Kyoto data using Local Outlier Factor.

**Table 3.** Performance of the proposed methods for intrusion detection.

		LOF			OC SVM			Isolation Forest			Robust Cov.						
		Acc.	Precision	Recall	F1	Acc.	Precision	Recall	F1	Acc.	Precision	Recall	F1				
UNSW-NB15	-	0.83	0.96	0.72	0.82	0.57	0.97	0.22	0.36	0.55	0.96	0.19	0.31	0.45	0.41	0.01	0.01
	PCA	0.83	0.95	0.72	0.82	0.72	0.99	0.51	0.67	0.44	0.1	2e-3	4e-3	0.48	0.77	0.08	0.14
	Encoder	0.82	0.96	0.7	<b>0.82</b>	0.79	0.96	0.64	0.77	0.46	0.72	0.03	0.06	0.46	0.81	0.03	0.05
NSL-KDD	-	0.43	0.1	1e-3	1e-3	0.76	0.93	0.63	0.75	0.75	0.97	0.58	0.73	0.63	0.99	0.35	0.52
	PCA	0.42	1e-5	1e-5	1e-5	0.75	0.93	0.61	0.73	0.5	0.96	0.13	0.23	0.43	0.73	5e-3	0.01
	Encoder	0.43	0.04	3e-4	6e-4	0.92	0.92	0.95	<b>0.93</b>	0.62	0.98	0.35	0.51	0.43	0.25	2e-3	4e-3
CIC-IDS-2017	-	0.68	0.81	0.74	0.77	0.64	0.76	0.74	0.75	0.67	0.96	0.56	0.71	0.36	0.93	0.13	0.22
	PCA	0.63	0.79	0.68	0.73	0.62	0.73	0.75	0.74	0.59	0.94	0.46	0.62	0.45	0.99	0.24	0.39
	Encoder	0.72	0.76	0.89	<b>0.82</b>	0.6	0.7	0.8	0.75	0.74	0.87	0.75	0.81	0.72	0.87	0.72	0.79
Kyoto	-	0.46	0.62	0.31	0.41	0.54	0.8	0.33	0.47	0.5	0.99	0.19	0.32	0.49	0.99	0.16	0.28
	PCA	0.49	0.82	0.22	0.34	0.53	0.81	0.3	0.44	0.59	0.99	0.33	0.5	0.6	0.99	0.36	0.53
	Encoder	0.57	0.93	0.33	0.49	0.68	0.81	0.61	<b>0.7</b>	0.47	0.99	0.13	0.23	0.4	0.96	0.03	0.06



**Fig. 2.** ROC curves obtained from each method and datasets using the direct method and feature transformation using PCA and encoder network.

## 5 Conclusions

Network intrusion detection is an active research area in a continuous development. Although there have been numerous efforts to address several challenges, anomaly-based approaches are sometimes difficult to be applied in real systems for intrusion detection.

In this work, feature learning is used for network intrusion detection through its application as a previous stage to four different anomaly detection techniques applied to recent datasets. The methods used for computing the latent representation of data are PCA and the encoder part of an auto-encoder that introduces non-linearity. The main improvement for the datasets is shown for One-Class SVM method using the latent space computed by the auto-encoder. In contrast, PCA transformation does not show relevant enhancement in order to be applied as a previous feature learning stage.

Future work includes the study of other types of auto-encoders and techniques including different feature selection methods combined with more algorithms for anomaly detection that can help to improve the identification of intrusions.

## References

1. Ahmed, M., Mahmood, A.N., Hu, J.: A survey of network anomaly detection techniques. *J. Netw. Comput. Appl.* **60**, 19–31 (2016)
2. Bengio, Y., Courville, A., Vincent, P.: Representation learning: a review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(8), 1798–1828 (2013)
3. Bhuyan, M.H., Bhattacharyya, D.K., Kalita, J.K.: Network anomaly detection: methods, systems and tools. *IEEE Commun. Surv. Tutor.* **16**(1), 303–336 (2014)
4. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. *ACM SIGMOD Rec.* **29**, 93–104 (2000)
5. Buczak, A.L., Guven, E.: A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Commun. Surv. Tutor.* **18**(2), 1153–1176 (2015)
6. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection - a survey. *ACM Comput. Surv.* **41**(3), 15:1–15:44 (2009). <https://doi.org/10.1145/1541880.1541882>
7. Chen, Y., Li, Y., Cheng, X.-Q., Guo, L.: Survey and taxonomy of feature selection algorithms in intrusion detection system. In: Lipmaa, H., Yung, M., Lin, D. (eds.) *Inscrypt 2006. LNCS*, vol. 4318, pp. 153–167. Springer, Heidelberg (2006). [https://doi.org/10.1007/11937807\\_13](https://doi.org/10.1007/11937807_13)
8. Erfani, S.M., Rajasegarar, S., Karunasekera, S., Leckie, C.: High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning. *Pattern Recognit.* **58**, 121–134 (2016)
9. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. *Science* **313**(5786), 504–507 (2006). <https://doi.org/10.1126/science.1127647>
10. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. *ACM Comput. Surv. (CSUR)* **31**(3), 264–323 (1999)

11. Javaid, A., Niyaz, Q., Sun, W., Alam, M.: A deep learning approach for network intrusion detection system. In: Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS), pp. 21–26. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering) (2016)
12. Khan, L., Awad, M., Thuraisingham, B.: A new intrusion detection system using support vector machines and hierarchical clustering. VLDB J. **16**(4), 507–521 (2007)
13. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. CoRR abs/1412.6980 (2014). <http://arxiv.org/abs/1412.6980>
14. Lee, J.A., Verleysen, M.: Nonlinear Dimensionality Reduction. Springer, New York (2007). <https://doi.org/10.1007/978-0-387-39351-3>
15. Liu, F.T., Ting, K.M., Zhou, Z.-H.: Isolation forest. In: Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, ICDM 2008, pp. 413–422. IEEE Computer Society (2008)
16. Mahoney, M.V., Chan, P.K.: An analysis of the 1999 DARPA/Lincoln Laboratory evaluation data for network anomaly detection. In: Vigna, G., Kruegel, C., Jonsson, E. (eds.) RAID 2003. LNCS, vol. 2820, pp. 220–237. Springer, Heidelberg (2003). [https://doi.org/10.1007/978-3-540-45248-5\\_13](https://doi.org/10.1007/978-3-540-45248-5_13)
17. McHugh, J.: Testing intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln laboratory. ACM Trans. Inf. Syst. Secur. (TISSEC) **3**(4), 262–294 (2000)
18. Mirsky, Y., Doitshman, T., Elovici, Y., Shabtai, A.: Kitsune: an ensemble of autoencoders for online network intrusion detection. arXiv preprint [arXiv:1802.09089](https://arxiv.org/abs/1802.09089) (2018)
19. Moustafa, N., Slay, J.: UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: Military Communications and Information Systems Conference (MilCIS), pp. 1–6. IEEE (2015)
20. Muda, Z., Yassin, W., Sulaiman, M., Udzir, N.I., et al.: A k-means and Naive Bayes learning approach for better intrusion detection. Inf. Technol. J. **10**(3), 648–655 (2011)
21. Nguyen, M.N., Vien, N.A.: Scalable and interpretable one-class SVMs with deep learning and random fourier features. arXiv preprint [arXiv:1804.04888](https://arxiv.org/abs/1804.04888) (2018)
22. Rousseeuw, P.J., Driessen, K.V.: A fast algorithm for the minimum covariance determinant estimator. Technometrics **41**(3), 212–223 (1999)
23. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural Comput. **13**(7), 1443–1471 (2001)
24. Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: ICISSP, pp. 108–116 (2018)
25. Sommer, R., Paxson, V.: Outside the closed world: on using machine learning for network intrusion detection. In: 2010 IEEE Symposium on Security and Privacy, pp. 305–316. IEEE (2010)
26. Song, J., Takakura, H., Okabe, Y., Eto, M., Inoue, D., Nakao, K.: Statistical analysis of honeypot data and building of Kyoto 2006+ dataset for NIDS evaluation. In: Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security, pp. 29–36. ACM (2011)
27. Tavallaei, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the KDD CUP 99 data set. In: Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defence Applications (2009)

28. Wang, K., Stolfo, S.J.: Anomalous payload-based network intrusion detection. In: Jonsson, E., Valdes, A., Almgren, M. (eds.) RAID 2004. LNCS, vol. 3224, pp. 203–222. Springer, Heidelberg (2004). [https://doi.org/10.1007/978-3-540-30143-1\\_11](https://doi.org/10.1007/978-3-540-30143-1_11)
29. Zhang, Z., Li, J., Manikopoulos, C., Jorgenson, J., Ucles, J.: HIDE: a hierarchical network intrusion detection system using statistical preprocessing and neural network classification. In: Proceedings of the IEEE Workshop on Information Assurance and Security, pp. 85–90 (2001)



# Cyber Security Incident Handling, Warning and Response System for the European Critical Information Infrastructures (CyberSANE)

Spyridon Papastergiou<sup>1</sup>✉, Haralambos Mouratidis<sup>2</sup>,  
and Eleni-Maria Kalogeraki<sup>1</sup>

<sup>1</sup> Department of Informatics, University of Piraeus,  
80 Karaoli and Dimitriou Str., 18534 Piraeus, Greece  
paps@unipi.gr, elmaklg@unipi.gr

<sup>2</sup> School of Computing, Engineering and Mathematics, University of Brighton,  
Brighton BN2 4GJ, UK  
H.Mouratidis@brighton.ac.uk

**Abstract.** This paper aims to enhance the security and resilience of Critical Information Infrastructures (CIIs) by providing a dynamic collaborative, warning and response system (CyberSANE system) supporting and guiding security officers and operators (e.g. Incident Response professionals) to recognize, identify, dynamically analyse, forecast, treat and respond to their threats and risks and handle their daily cyber incidents. The proposed solution provides a first of a kind approach for handling cyber security incidents in the digital environments with highly interconnected, complex and diverse nature.

**Keywords:** Incident handling · Web mining · Data fusion and risk assessment

## 1 Introduction

In the digital era, Critical Infrastructures (CIs) are operating under the premise of robust and reliable ICT components, complex ICT infrastructures and emerging technologies (e.g. IoT, Cloud Computing) and are transforming into Critical Information Infrastructures (CIIs) that can offer a high degree of flexibility and efficiency in the communication and coordination of advanced services. The increased usage of information technology in modern CIIs means that they are becoming more vulnerable to the activities of hackers and other perpetrators of cyber-related crime.

Over the last few years, it is a common phenomenon to see daily headlines describing major cyber-attacks or some new strain of malware or insidious social engineering technique being used to attack ICT infrastructures. In particular, CIIs have become lately targets for cyberattacks attracting the attention of security researchers, cyber-criminals, hacktivists (e.g. Anonymous, LulzSec) and other such role-players (e.g. cyber-spies). These cyber actors have significantly evolved their tactics, techniques and procedures to include next-generation malware toolkits available in various locations on the internet (e.g. deep web, dark web) and new data exfiltration methods that give them an asymmetric quantum leap in capability. In the past years, there have

been a number of cybersecurity meltdowns and high-profile breaches affecting critical infrastructures, such as the recent ransomware attacks, WannaCry and WanaCrypt0r 2.0, which affected more than 230,000 computers in over 150 countries. In most cases, the adversaries targeted the organizations' interconnected infrastructures as a means of spreading their harmful malware to a broader audience. Obviously, the impact of a compromised CII can extend far beyond the corporate boundaries, putting not just individual organizations but also their dependent entities at risk.

In 2016, the Commission introduced the E.U. Directive NIS 2016 that enforces all CIIs to report to an appropriate Computer Security Incident Response Team (CSIRT) any incident having a substantial impact on the provision of their services. Unfortunately, these efforts mostly focused on providing just the legal basis and creating an assurance framework for boosting the cyber security culture across sectors which are vital for the EU economy and society and moreover rely heavily on ICTs. Nevertheless, there has been a lack of innovation to capture and correlate events and information associated with cyber-attacks in CIIs as well as lack of appropriate approaches that support and facilitate effective cooperation among the CIIs entities in terms of exchanging specific cybersecurity risk and threats information.

The paper proposes a state of the art solution, the CyberSANE system, which aims to improve the detection and analysis of cyber-attacks and threats on CIIs and increases the knowledge of the current cyber threat landscape. In particular, the CyberSANE system helps the organizations to raise their preparedness, improve their cooperation with each other, and adopt appropriate steps to manage security risks, report and handle security incidents. The rest of the paper is structured as follows. Sections 2 and 3 present the related work and the main aspects of the proposed incident handling approach respectively. Section 4 describes the CyberSANE system and its key components. Section 5 illustrates the components data flows and the overall system operation; and finally Sect. 6 draws the conclusions.

## 2 Current Efforts of Incident Handling in Critical Information Infrastructures

The main goal of the security incident handling and response process is to define the main aspects and principles for coordinating the effort that should be applied in managing a security breach/incident/event [1, 2]. In principle, choosing the right approach for incident handling proves to be complicated. In recent years, a number of security incident response approaches and frameworks [3–14] have been introduced by the research and industrial communities as well various standardization bodies. Although, many of these approaches provide specific technical guidelines, aiming to enhance the security incident response capabilities of the organizations, they present significant limitations. In particular, Grimes (2007) argues that most of the existing incident response approaches follow a linear process that is outdated and does not support the highly efficient capability that is required to handle and manage today's incidents. Therefore, a progression flaw exists in these processes, since if one phase in the linear process is not completed, the entire process cycle may stop midstream. [15] notes that current incident response processes are too focused on the containment,

eradication, and recovery-related activities and usually ignore, skip or do not emphasize on other important steps of incident management, such as investigations actions. [16] proposal gives emphasis on proactive preparation and reactive learning to encourage security incident learning. [17–19] argue that the existing incident handling approaches do not provide adequate guidance on how to conduct effective forensic investigations. Hence, current methods' limitation to assist and guide the investigators in forensic evidence analysis, undermines the value of the evidence and fails to promote incident resolution.

In addition, the available security information and event management solutions lack significant reactive and post-incident capabilities for managing incidents and events in the scope of the ICT-based CIIs providing inadequate technical guidance to the incident response professionals on how to detect, investigate and reproduce attacks. As such, and despite the socioeconomic importance of tools and techniques for handling incidents there is still no easy, structured, standardized and trusted way to manage and forecast interrelated, cybersecurity incidents in a way that takes into account the heterogeneity and complexity of the CIIs and the increasingly sophisticated types of attacks. Therefore, there is a pressing need for devising novel systems for efficient CIIs incident handling and support thorough and common understanding of cyber-attack situations in a timely manner.

In a nutshell, the main limitations [20–22] of the existing approaches are the following: (i) the traditional linear incident response models are too slow, ineffective and do not support the highly efficient capability that is required to handle and manage today's incidents; (ii) focus mostly on the proactive element (i.e. provide assistance and information to help prepare, protect, and secure) of the incident management; (iii) current approaches do not provide enough insight into the underlying causes of the incident; (iv) poor provisions for incident planning; (v) undermine the value of forensic evidence possibly required for subsequent legal action; (vi) do not take into account the risk-related results produced by existing risk assessment methodologies.

### 3 CyberSANE Incident Handling Approach

The proposed incident handling approach aims to address the aforementioned limitations of these existing methodologies and tools, providing a step-by-step guidance to manage incidents and breaches on CIIs occurred due to cyber attacks. On this account, CyberSANE pursues to combine active approaches that are used to detect and analyse anomaly activities and attacks in real-time with reactive approaches that deals with the analysis of the underlying infrastructure to assess an incident in order to provide a more holistic and integrated approach to incident handling. In this vein, CyberSANE aims to enhance the incident detection capabilities of the existing methods described in the previous section with a more efficient, elastic and scalable reasoning approach. The main characteristics of the proposed approach are the following: (i) learning from unstructured data without the need to understand the content; (ii) identification of unusual activities that match the structural patterns of possible intrusions (instead of predefined rules); and (iii) automatic identification and adaption to a change of the underlying infrastructure.

CyberSANE treats the handling of a cyber incident as a dynamic experimental environment that can be optimized involving all relevant CIIs' operators and security experts. CyberSANE's approach is based on simulations to facilitate the evaluation and analysis of an identified incident and support the investigation decision making process in a rigorous manner. The pursuit of CyberSANE is to support incident handling process with advanced correlation capabilities in terms of accuracy and efficiency strengthening the rational analysis. In this context, CyberSANE relies on pioneering mathematical models (e.g. machine learning, deep learning and Global Artificial Intelligence (AI) techniques) for analysing, compiling, combining and correlating all incident-related information and data from different levels and contexts (e.g. taking into consideration information, data and opinions collected and analysed from existing risk assessment frameworks (including the CYSM, MEDUSA, MITIGATE and SAURON approaches [23–25], knowledge and information acquired from previous incident investigations as well as evidential data extracted from the compromised cyber systems). Thus, these techniques will be able to identify, extract and analyse the most relevant parts of the information related to the initial incident in order to find the relationships between the compromised devices/systems and these evidences.

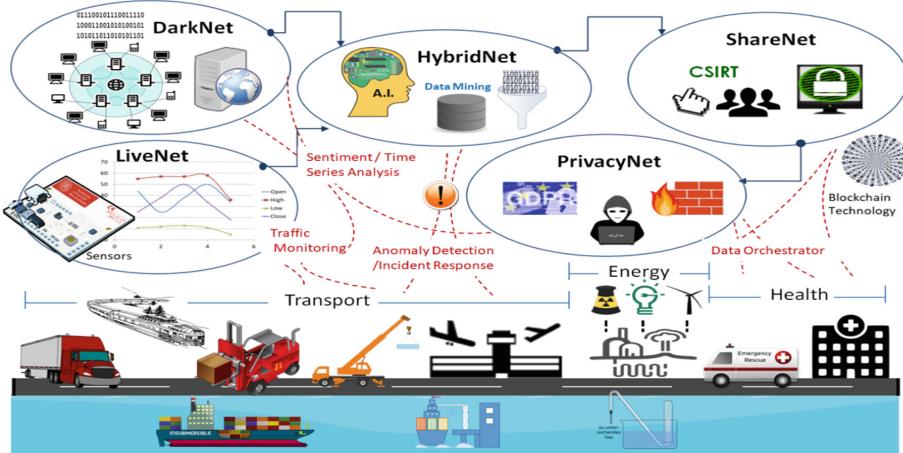
Moreover, efficient simulation experiments for generating multi-order evidence dependencies have been used to generate and construct secure, reliable and valid chains of evidence anticipating how the attack is progressing. Additionally, taking into account the effects of a security incident, real-time insights, alerts and warnings will be produced to increase situation awareness, inform the CIIs' stakeholders about the effects of the events and guide them how to react.

The main contribution of the current approach that differentiates it from the aforementioned related incident handling proposals is that it combines existing machine learning techniques, such as clustering and hidden Markov models, with deep learning and Global Artificial Intelligence (AI) to develop an innovative way that optimizes the automatic analysis of huge amounts of events, information and evidence. To identify malicious actions in the cyber assets, such as abnormal behaviours, it combines both structured data (e.g. logs and network traffic) and unstructured data (e.g. data coming from social networks and dark web) in a privacy-aware manner. Furthermore, it adopts deep semantic analysis techniques together with Natural Language Processing (NLP) methods (e.g. Named Entity Recognition and Word Sense Disambiguation) to extract important information from multilingual security-related contexts, facilitating multilingual data generation and exploitation within the networked ecosystem.

## 4 CyberSANE Incident Handling system

CyberSANE is an advanced, configurable and adaptable, Security and Privacy Incident Handling system (CyberSANE system), towards effective security incident detection and handling. The main goal of the proposed system is to improve, intensify and coordinate the overall security efforts for the effective and efficient identification; investigation, mitigation and reporting of realistic multi-dimensional attacks within the interconnected web of cyber assets in the CIIs and security events. The proposed

approach takes into consideration and addresses both technical and cognitive challenges (Fig. 1).



**Fig. 1.** CyberSANE incident handling system

From technical perspective, the system aims at collecting, compiling, processing and fusing all individual incident-related information ensuring their integrity and validity following the generic phases of ISO/IEC 27035:2016. In contrast, from cognitive point of view, the decision makers should be able to understand the technical aspects of an attack and draw conclusions on how to respond. In order to realize this vision, the CyberSANE system will be composed of five main components:

- The *Live Security Monitoring and Analysis (LiveNet)* component which is able to monitor, analyze, and visualize organizations' internal live network traffic in real time. This environment aggregates and visualizes traffic flowing through live networks as well as alerts given out by security appliances installed at critical points in the network with an overemphasis on threat prevention solutions.
- The *Deep and Dark Web mining and intelligence (DarkNet)* component which monitors the Dark and Deep Web in order to grasp and analyse the big picture of global malware/cybersecurity activities.
- The *Data Fusion, Risk Evaluation and Event Management (HybridNet)* component that receives security related information on potential cyber threats from both LiveNet and Darknet respectively in order to analyze and evaluate the security situation inside an organization.
- The *Intelligence and Information Sharing and Dissemination (ShareNet)* component disseminates and shares information of useful incident-related information with relevant parties (e.g. industry cooperation groups, Computer Security Incident Response Teams - CSIRTs) about the effects and danger of incidents characterized diffusing threats.

- The *Privacy & Data Protection (PrivacyNet)* Orchestrator which provides a set of privacy (anonymization, pseudonymization, obfuscation), data protection orchestration and consistency capabilities.

It should be noted that the proposed solution and the incorporating techniques is able to operate in heterogeneous, large-scale, cross-border CIIs that are characterized by the following features: (i) complex, highly distributed, and large-scale cyber systems (including IoT and cyber-physical) with respect to the number of entities involved; (ii) heterogeneity of the underlying networks interconnecting the physical-cyber systems; and (iii) different levels of exposure to attacks. The following Sections provide a detailed description of the each component.

#### 4.1 Live Security Monitoring and Analysis (LiveNet) Component

The LiveNet is an advanced and scalable Live Security Monitoring and Analysis component capable of preventing and detecting threats and, in case of a declared attack, capable of mitigating the effects of an infection/intrusion. The main objective of this component is to implement the Identification, Extraction, Transformation, and Load process for collecting and preparing all the relevant information, serving as the interface between the underlying CIIs and the CyberSANE system. It includes proper cyber security monitoring sensors with network-based Intrusion Detection Systems (IDS), innovative Anomaly detection modules and endpoint protection solutions for accessing and extracting information, on a real-time basis, in order to detect complex and large-scale attacks (e.g. Advanced Persistent Threats). The incident-related information that reside in different and heterogeneous cyber systems may include various types of data, such as: active (unpatched) vulnerabilities in the technological infrastructure; misuse detection in the network or in the systems, including both host-based and network-based IDS deployment and integration; anomaly detection in the network or in the systems; system availability signals; network usage and bandwidth monitoring; industry proprietary protocol anomalies; SCADA vulnerabilities, etc.

LiveNet incorporates appropriate data management and reasoning capabilities for: near real-time identification of anomalies, threats, risks and faults and the appropriate reactions; (ii) proactive reaction to threats and attacks; and (iii) dynamic decision making in micro, macro and global level according to the end user's needs and the identified incidents/threats. These capabilities are empowered with more innovative algorithms based on techniques such as machine learning, deep learning and AI that identify previously unknown attacks. This component provides an abstraction of the collected information to the Data Fusion, Risk Evaluation and Event Management (HybridNet) component of the CyberSANE system. Moreover, all incidents-related information captured from LiveNet will be parsed, filtered, harmonized and enriched to ensure that only the data necessary for the multivariate and multidimensional analysis are available to the other components (e.g. HybridNet). Thus, LiveNet contributes as follows: (i) preventing a flood of irrelevant or repeated information from cluttering the HybridNet processing component; and (ii) consolidating the different data contents and formats towards a uniform perspective in order to provide the upper components a unified and convenient way to handle the information.

## 4.2 Deep and Dark Web Mining and Intelligence (DarkNet) Component

The Deep and Dark Web mining and intelligence (DarkNet) component provides the appropriate Social Information Mining capabilities that will allow the exploitation and analysis of security, risks and threats related information embedded in user-generated content (UGC). This is achieved via the analysis of both the textual and meta-data content available from such streams. Textual information is processed to extract data from otherwise disparate and distributed sources that may offer unique insights on possible cyber threats. Examples include the identification of situations that can become a threat for the CIIs with significant legal, regulatory and technical considerations. Such situations are: organization of hacktivist activities in underground forums or IRC channels; external situations that can become a potential threat to the CIIs (e.g. relevant geopolitical changes); disclosure of zero day vulnerabilities; sockpuppets impersonating real profiles in social networks etc. Entities (e.g., events, places) and security-related information will be uniquely extracted from textual content using advanced Natural Language Processing (NLP) techniques, such as sentiment analysis.

## 4.3 Data Fusion, Risk Evaluation and Event Management (HybridNet) Component

The Data Fusion, Risk Evaluation and Event Management (HybridNet) component provides the intelligence needed to perform effective and efficient analysis of a security event based on: (i) information derived and acquired by the LiveNet and DarkNet components; and (ii) information and data produced and extracted from this component. In particular, HybridNet component retrieves incidents-related data via the LiveNet component from the underlying CIIs and data from unstructured and structured sources (e.g. from Deep and Dark Web) consolidated in a unified longitudinal view which are linked, analyzed and correlated, in order to achieve semantic meaning and provide a more comprehensive and detailed view of the incident. In CyberSANE, a formal and uniform representation of digital evidence along with their relationships has been used to encapsulate all concepts of the forensic field and provide a common understanding of the structure of all information linking to evidence among the CIIs' operators and the forensics investigators. The main goal of the analysis process is to continuously carry out the assessment (e.g. identification of on-going attacks and related information, such as what is the stage of the attack and where is the attacker) and prediction (i.e. identification of possible scenarios of future attacks through forecasting models). HybridNet incorporates fusion models based on existing mathematical models (e.g. data mining, AI, deep learning, machine learning and visualization techniques). These models will support and provide reasoning capabilities for the near real-time identification of anomalies, threats and attacks, assessing any possible malicious actions in the cyber assets such as abnormal behaviors or malicious connections to identify unusual activities that match the structural patterns of possible intrusions. Once an attack is detected or predicted a simulation will be performed to form the full representation of the attack. In particular, a Security Incident/Attack Simulation Environment undertakes to generate and construct all secure, reliable and valid chains of evidence allowing: (i) the identification of the attacker's behavior so far;

(ii) the identification of the attacker's goals and strategies and prediction of their next actions; and (iii) the accurate assessment of the impact of an incident on the CII and the damage caused so far.

The Security Incident/Attack Simulation Environment of the CyberSANE system comprises a set of novel mathematical instruments, including mathematical models for simulating, analyzing, optimizing, validating, monitoring simulation data and optimizing security incident handling process. Specifically, these instruments include: (i) a bundle of novel process/attack analysis and simulation techniques for designing, executing, analyzing and optimizing threat and attack simulation experiments that will produce appropriate evidence and information that facilitate the identification, assessment and mitigation of the CII-related risks; (ii) graph theory to implement attack graph generation, to perform security incident analysis and to strengthen the prognosis of future malefactor steps; (iii) pioneering mathematical techniques for analyzing, compiling and combining information and evidences about security incidents and attacks/threats patterns and paths in order to find relationships between the recovered forensic artefacts and piecing the evidential data together to develop a set of useful chain of evidence (linked evidence) associated with a specific incident; (iv) innovative simulation techniques which will optimize the automatic analysis of diverse data; (v) innovative techniques in order to link optimization and simulation. In this context, this simulation environment is fed with information about an incident and proceeds to calculate and generate a number of possible attack graphs (routes of possible attacks) and graphs of linked evidence (chains of evidence) and also compute probabilities for a sequence of events on top of these graphs. The resulting probabilistic estimate for the compromised CIIs' assets will be used to identify, model and represent the course of an attack as it propagates across the CIIs. It should be noted the HybridNet component continuously updates the simulation engine with information collected and piece of information, thereby enabling both understanding which assets might have been compromised, as well as gain more accurate estimates on the likelihood that other assets might be compromised in the future.

#### **4.4 Intelligence and Information Sharing and Dissemination (ShareNet) Component**

The ShareNet component provides the necessary threat intelligence and information sharing capabilities within the CIIs and with relevant parties (e.g. industry cooperation groups, CSIRTS). It is responsible for the instantiation of the adopted intelligence model; in particular, ShareNet undertakes the identification and dissemination of, the right and sanitized information that have to be shared in a usable format and in a timely manner. This environment produces and circulates notifications containing critical information, enhancing the perception of the current situation and improving the projection into the future. It should be noted that all potential evidence from the systems that are suspected to be part of the infrastructure being investigated are forensically captured, stored and exchanged in a way that their integrity is maintained using the security and data protection methods of the PrivacyNet Orchestrator.

To this end, ShareNet follows a trusted and distributed intelligence and incident sharing approach to facilitate and promote the collaboration and secure and

privacy-aware information sharing of the CIIs' operators with relevant parties (e.g. industry cooperation groups, CSIRTs), in order to exchange risk incident-related information, through specific standards and/or formats (STIX), improving overall cyber risk understanding and reduction. Privacy preserving is another important issue considered at every phase of sharing, applying methods such as anonymization or pseudo anonymization and encryption techniques incorporated in and made available from PrivacyNet Orchestrator. This brings forward a mixture of several cryptographic techniques that holds certain security guarantees.

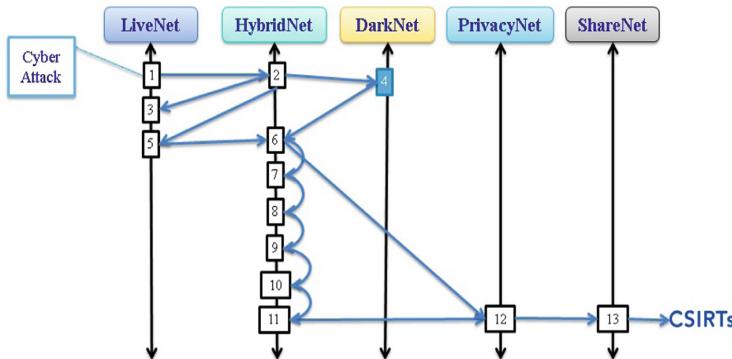
#### 4.5 Privacy & Data Protection (PrivacyNet) Orchestrator

Through the specific “Privacy & Data Protection Orchestrator” (PrivacyNet), it is possible to coordinate the abovementioned components of the CyberSANE system in order to ensure desired-levels of data protection for sensitive incident-related information, enabling the possibility to apply such protection in all phases of cyber security incident handling flow. The main purpose the PrivacyNet is to manage and orchestrate the application of the innovative privacy mechanisms and maximize achievable levels of confidentiality and data protection towards compliance with the highly-demanding provisions in the GDPR in the context of protecting sensitive incident-related information within and outside CIIs. To this end, PrivacyNet sets up the security and data “protection configurations” allowing security experts and members of the incident response team to specify all the protection steps that have to be performed and the required conditions to execute them, which can be referred to GDPR-based rules (and to other guidance for its application by the European Data Protection Board, formerly Art. 29 Data Protection Working Party).

In addition, the orchestration approach of the CyberSANE allows applying the most appropriate security and data protection methods depending on the user's privacy requirements, which cover a wide range of techniques including anonymization, location privacy, obfuscation, pseudonymization, searchable encryption, multi-party computation and verifiable computation, in order to meet the highly demanding regulatory compliance obligations, for example in relation to accountability towards data protection supervisory authorities, for adequate management of informed consent etc. For this reason, novel techniques and processes for enhancing the secure distribution and storage of all forensic artifacts in order to protect them from unauthorized deletion, tampering revision and sharing (e.g. Attribute-based Encryption (ABE) and block-chain technologies) have been combined.

### 5 CyberSANE Components Data Flow and System Operation

The abovementioned CyberSANE components communicate to carry out 13 operational data flows as illustrated in Fig. 2 and outlined below. Through, those data flows and the overall operation, the CyberSANE system assists all the phases of security assessment and reduction of cyber risks on relevant operational scenarios.



**Fig. 2.** CyberSANE system's & components' operation

The operational data flows supported by the CyberSANE system are the following: (1) Detected cyber attack visualized in the LiveNet Component; (2) Determine whether their organization has fallen victim to a cyber-attack; (3) Discovery, extraction and collection of raw data from various sources (e.g. servers, logs) and in different format (4) Extraction, harmonization and processing data from distributed sources (Dark Web, Social Media) that offer unique insights on the cyber threats and provide information about latest mechanisms of cyber-attacks; (5) Collected data is normalized, cleansed to remove redundant information and transformed into a common representation format; (6) All relevant information extracted are analyzed and correlated to provide a more comprehensive and detailed view of the incident; (7) Dependency evidence chains are generated; (8) Identification of on-going attacks (Identification of the attacker's behavior, prediction of next actions); (9) Evaluate the risks; Assess the impact and the assessment and cascading effects; and Formulate mitigation plan; (10) Prediction of possible scenarios of future attacks; (11) Visualization of incident related information enabling deep understanding of the situation and decision making (Notifications); (12) Secure and privacy aware managing and storing of incident-related information; and (13) Information sharing and dissemination of useful incident-related information.

## 6 Conclusions

The paper aims to leverage collected security information to find new ways of protection for technology assets, enabling the entity at risk to evaluate the risk and invest to limit that risk in an optimal way. Providing a way to securely collect both structured data (e.g. logs and network traffic) and unstructured data (e.g. data coming from social networks and dark web) and making them available for analysis fosters new innovations that will only unravel after having access to such data, harnessing its full potential. CyberSANE's has a twofold aim; to minimize the exposure to security risks/threats and help CIIs' operators to respond successfully to relevant incidents.

The ground-breaking nature of the proposed incident handling approach is based on: (i) the identification of attacks and incidents using innovative approaches and

algorithms of unobserved components techniques and linear state-space models producing meaningful information from cyber systems, (ii) the combination of active incident handling approaches with reactive approaches producing real-time insights, alerts and warnings about cyber events, (iii) innovative normalization process that unifies all relevant incident-related information gathered from heterogeneous CIIs, (iv) novel attacks' scenarios and evidence representation with simulation techniques and visualization tools that increase the efficiency of investigation results, (v) hybridization forms of mathematical models and combinations of data mining, Global artificial intelligence, machine learning that optimize evidential data from different sources.

The proposed CyberSANE System meets its objectives embedding core security features allowing faster and better operation of advanced cyber security functionalities. These aspects comprise an innovative, knowledge based, collaborative security and response dynamic system which increase the agility of the investigators and encourage continuous learning throughout the incident life cycle.

**Acknowledgements.** The authors would like to thank the University of Piraeus Research Centre for its continuous support.

## References

1. West-Brown, M.J., Stikvoort, D., Kossakowski, K.P., Killcrece, G., Ruefle, R.: Handbook for computer security incident response teams (CSIRTs). (No. CMU/SEI-2003-HB-002). Carnegie-Mellon Univ Pittsburgh PA Software Engineering Inst. (2003a)
2. Wiik, J., Kossakowski, K.P.: Dynamics of incident response. In: 17th Annual FIRST Conference on Computer Security Incident Handling, Singapore (2005)
3. British Standards Institution. BS ISO/IEC 27035:2011 - Information Technology. Security Techniques. Information Security Incident Management (2011)
4. Cichonski, P., Scarfone, K.: Computer Security Incident Handling Guide Recommendations. NIST, Gaithersburg (2012). National Institute of Standards and Technology (NIST)
5. ENISA CSIRTs by Country-Interactive Map. <https://www.enisa.europa.eu/topics/csirts-in-europe/csirt-inventory/certs-by-country-interactive-map>
6. Northcutt, S.: Computer Security Incident Handling Version 2.3.1 (2003)
7. Vangelos, M.: Incident response: managing. In: Encyclopedia of Information Assurance, pp. 1442–1449. Taylor & Francis (2011)
8. Werlinger, R., Muldner, K., Hawkey, K., Beznosov, K.: Preparation, detection, and analysis: the diagnostic work of it security incident response. Inf. Manag. Comput. Secur. **18**(1), 26–42 (2010)
9. Khurana, H., Basney, J., Bakht, M., Freemon, M., Welch, V., Butler, R.: Palantir: a framework for collaborative incident response and investigation. In: Proceedings of the 8th Symposium on Identity and Trust on the Internet, p. 38e51 (2009)
10. Grobauer, B., Schreck, T.: Towards incident handling in the cloud. In: Proceedings of the 2010 ACM Workshop on Cloud Computing Security Workshop (CCSW 10), pp. 77–85 (2010)
11. Monfared, A., Jaatun, M.G.: Handling compromised components in an IaaS cloud installation. J. Cloud Comput. Adv. Syst. Appl. **1**, 16 (2012)

12. Line, M.B.: A case study: preparing for the smart grids-identifying current practice for information security incident management in the power industry. In: 2013 7 International Conference on IT Security Incident Management and IT Forensics, IT Security Incident Management and IT Forensics (IMF), pp. 26–32. IEEE (2013)
13. Cusick, J.J., Ma, G.: Creating an ITIL inspired incident management approach: roots, response, and results. In: Network Operations and Management Symposium Workshops (NOMS Wksp) 2010 IEEE/IFIP, pp. 142–148. IEEE (2010)
14. Connell, A., Palko, T., Yasar, H.: Cerebro: a platform for collaborative incident response and investigation. In: 2013 IEEE International Conference on Technologies for Homeland Security (HST) (2013)
15. Ahmad, A., Hadgkiss, J., Ruighaver, A.B.: Incident response teams-challenges in supporting the organisational security function. *Comput. Secur.* **31**(5), 643–652 (2012)
16. Shadden, P., Ahmad, A., Ruighaver, A.B.: Informal learning in security incident response teams. In: 2011 Australasian Conference on Information Systems (2011)
17. Casey, E.: Investigating sophisticated security breaches. *Commun. ACM* **49**(2), 48–55 (2006)
18. Nnoli, H., Lindskog, D., Zavarsky, P., Aghili, S., Ruhl, R.: The governance of corporate forensics using COBIT, NIST and increased automated forensic approaches. In: 2012 International Conference on Privacy, Security, Risk and Trust. IEEE (2012)
19. Tan, T., Ruighaver, T., Ahmad, A.: Incident handling: where the need for planning is often not recognised. In: 1st Australian Computer, Network & Information Forensics Conference (2003)
20. FireEye. The Need for Speed: 2013 Incident Response Survey (2013)
21. Grispos, G., Glisson, W.B., Storer, T.: Rethinking security incident response: the integration of agile principles. arXiv preprint [arXiv:1408.2431](https://arxiv.org/abs/1408.2431) (2014)
22. Ab Rahman, N.H., Choo, K.K.R.: A survey of information security incident handling in the cloud. *Comput. Secur.* **49**, 45–69 (2015)
23. Papastergiou, S., Polemi, D.: Securing maritime logistics and supply chain: the Medusa and MITIGATE approaches. *Maritime Interdiction Operations Journal* **14**(1), 42–48 (2017). Proceedings of 2nd NMIOTC Conference on Cyber Security. ISSN 2242-441X
24. Papastergiou, S., Polemi, N.: MITIGATE: a dynamic supply chain cyber risk assessment methodology. In: Yang, X.S., Nagar, A., Joshi, A. (eds.) Smart Trends in Systems, Security and Sustainability. LNNS, vol. 18, pp. 1–9. Springer, Heidelberg (2018). [https://doi.org/10.1007/978-981-10-6916-1\\_1](https://doi.org/10.1007/978-981-10-6916-1_1)
25. Kalogeraki, E.-M., Papastergiou, S., Polemi N.: SAURON real-life use cases: terrorists attack a cruise ship berthed at a port facility. In: The 9th NMIOTC Annual Conference “Fostering Projection of Stability through Maritime Security: Achieving Enhanced Capabilities and Operational Effectiveness” 5–7 June 2018 (2018)



# Fault Diagnosis in Direct Current Electric Motors via an Artificial Neural Network

Theofanis I. Aravanis<sup>1(✉)</sup>, Tryfon-Chrysovalantis I. Aravanis<sup>2</sup>,  
and Polydoros N. Papadopoulos<sup>3</sup>

<sup>1</sup> Department of Business Administration, University of Patras, Patras, Greece  
[taravanis@upatras.gr](mailto:taravanis@upatras.gr)

<sup>2</sup> Stochastic Mechanical Systems and Automation (SMSA) Laboratory,  
Department of Mechanical Engineering and Aeronautics, University of Patras,  
Patras, Greece

[aravanis@mech.upatras.gr](mailto:aravanis@mech.upatras.gr)

<sup>3</sup> Department of Mechanical Engineering T.E., TEI of Western Greece,  
Patras, Greece  
[polidoros.18@gmail.com](mailto:polidoros.18@gmail.com)

**Abstract.** The combined problem of fault detection and classification (referred to as fault diagnosis) of Direct Current (DC) electric motors via a simple, yet powerful, technique based on an Artificial Neural Network (ANN) is proposed. The ability of an ANN in identifying patterns with high fidelity—without the need of any rigorous mathematical model of the system under investigation—leads to an excellent diagnosis performance, even for faults that result in almost indistinguishable output system responses (both in time and in frequency domain). The flexibility and speed of the presented method indicate that it can easily be applied to on-line fault diagnosis as well.

**Keywords:** Direct current electric motors · Artificial neural networks · Artificial intelligence · Fault detection and classification

## 1 Introduction

*Electric motors* are ubiquitous components of the infrastructure, consuming approximately 60% of all the electric power produced. As a consequence, appropriate and reliable operation is deemed necessary.

Design criteria, such as the margins of the nominal parameter values of a system, can ensure acceptable performance when there are slight operating disturbances. Nevertheless, if the system dynamics change significantly due to a component failure, the result may be suboptimal or, in the worst case, catastrophic. In order to ensure safety and proper maintenance, *fault detection* and, if possible, *fault classification* (also referred to as *fault diagnosis*) in the operating systems are imperative. The fault diagnosis should be ideally employed using as

few sensors as possible. In cases where the faults are “small” and incipient, the problem turns out to be highly challenging.

Multiple approaches have been proposed for the problem of fault detection and classification in electric motors—the interested reader may refer to [7] for a survey confined to Alternating Current (AC) induction (asynchronous) motors.

In this article, a simple, yet powerful, technique for fault detection and classification of *Direct Current* (DC) electric motors is proposed, in the core of which lies an *Artificial Neural Network* (ANN). Loosely speaking, an ANN is a computing system, vaguely inspired by the biological neural networks, which constitute living organisms’ brains [6]. Such a system “learns” to perform tasks by considering examples, generally without being explicitly programmed with any task-specific rules and without need of a rigorous mathematical model for the specific system.<sup>1</sup> It is worth-noting that the work, although limited to DC motors, has a variety of extensions in a plethora of types of electric motors.

Other studies may be, in their majority, characterized as *threshold-oriented*, in the sense that the techniques involved are based on the intuition that, when a (signal or parameter) value exceeds a present threshold, a fault is detected [4]. However, this is often problematic, since the problem of classification of the fault is not easily addressed, as well as such practices may lead to many false alarms. In fact, under varying operating conditions, the effects of the considered faults on the dynamics are almost fully “masked” by those of the operating conditions. In addition, for these methods to be applied, an accurate physics-based motor model is necessary.

Another great portion of fault detection and classification approaches requires *pre-processing* of the measured signals—such as Fourier transform analysis, statistical analysis of a residual [5,8]—demanding significant resources, thus making them unsuitable for quick on-line tests.

The goal of the present study is to take advantage of the powerful ability of an ANN to identify *patterns* with high fidelity—which in turn addresses the majority of the aforementioned disadvantages of other techniques—and test its performance in the problem of fault diagnosis in DC motors. As far as the proposed method is concerned, the ANN consists of an *input layer*, two *hidden layers*, and an *output layer*. The input layer of the ANN receives (without any pre-processing) the measured signals of the *armature current* and *angular velocity* of a DC motor under load condition, and the output layer *identifies* and *classifies* the (potential) fault. The ANN, essentially, recognizes the patterns of the corresponding signals; hence, it recognizes the *unique* “profile” of *each* state condition of the system. This, in turn, deviates from the rigid threshold-oriented approach; therefore, the false alarms are significantly reduced or even disappear. Furthermore, the proposed technique requires *no pre-processing* of the measured signals, making it fast and flexible for *on-line* testing.

The rest of this paper is organized as follows: Sect. 2 presents the model of the system, Sect. 3 analyses the fault detection and classification problem, whereas

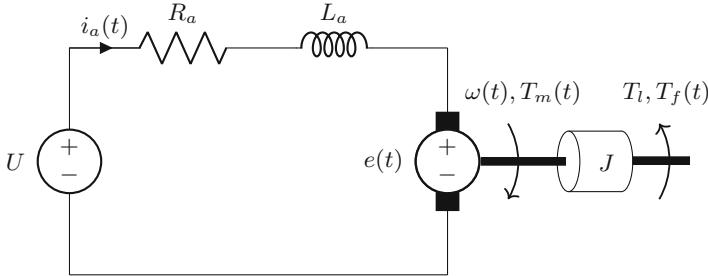
---

<sup>1</sup> Artificial Neural Networks have been successively used on a variety of applications, including computer vision, speech recognition, machine translation and many more.

in Sect. 4 the Artificial Neural Network training and testing results are discussed. The last section is devoted to some concluding remarks.

## 2 System Modelling

The system layout of a DC motor is shown in Fig. 1, where it is assumed that the DC voltage,  $U$ , and load torque,  $T_l$ , are the inputs, and armature current,  $i_a(t)$ , and angular velocity of the shaft,  $\omega(t)$ , are the outputs of the system.



**Fig. 1.** Model of the DC motor.

The torque generated by a DC motor,  $T_m(t)$ , is proportional to the armature current and the magnetic field strength. Herein, the magnetic field is assumed to be constant, and, therefore, the motor torque is proportional only to the armature current by a constant factor  $K_t$ ; i.e.,  $T_m(t) = K_t \cdot i_a(t)$ . The load torque,  $T_l$ , is considered to be constant, with small load disturbances. The friction torque,  $T_f(t)$ , is considered to be proportional to the angular velocity of the shaft by a constant factor  $b$ ; i.e.,  $T_f(t) = b \cdot \omega(t)$ . The back electromotive force (EMF),  $e(t)$ , is also proportional to the angular velocity by a constant factor  $K_e$ ; i.e.,  $e(t) = K_e \cdot \omega(t)$ .

Applying Kirchhoff's Law in the electrical part, Newton's Second Law in the mechanical part, and choosing as state variables  $z_1 = i_a(t)$  and  $z_2 = \omega(t)$ , we have the following equations describing the system:

$$L_a \cdot \frac{di_a(t)}{dt} + R_a \cdot i_a(t) + e(t) = U \quad (1)$$

$$J \cdot \frac{d\omega(t)}{dt} = T_m(t) - T_l - T_f(t) \quad (2)$$

$$\dot{z}_1 = \frac{1}{L_a} \cdot (U - R_a \cdot z_1 - K_e \cdot z_2) \quad (3)$$

$$\dot{z}_2 = \frac{1}{J} \cdot \left( -T_l + K_t \cdot z_1 - b \cdot z_2 \right) \quad (4)$$

The state-space representation of the system is given by the following two equations:

$$\dot{\underline{z}} = \underline{A} \cdot \underline{z} + \underline{B} \cdot \underline{u} \quad (5)$$

$$\underline{y} = \underline{C} \cdot \underline{z} + \underline{D} \cdot \underline{u} \quad (6)$$

Table 1, subsequently, presents the nominal parameters of the modelled system.

**Table 1.** Nominal parameters of the modelled system.

Motor parameter	Symbol	Nominal value
Electric resistance	$R_a$	$5.73 (\Omega)$
Electric inductance	$L_a$	$0.003 (\text{H})$
Moment of inertia	$J$	$80.45 \cdot 10^{-6} (\text{kg m}^2)$
Electromotive force constant	$K_e$	$0.06 (\text{V}/(\text{rad/s}))$
Motor torque constant	$K_t$	$0.06 (\text{N m/A})$
Motor viscous friction constant	$b$	$5 \cdot 10^{-6} (\text{N m}/(\text{rad/s}))$

Then, the following matrices are derived for the examined model (note that all states are outputs, i.e.,  $\underline{z} = \underline{y}$ ):

$$\underline{A} = \begin{bmatrix} -\frac{R_a}{L_a} & -\frac{K_e}{L_a} \\ \frac{K_t}{J} & -\frac{b}{J} \end{bmatrix} = \begin{bmatrix} -1910 & -20 \\ 746 & -1243 \end{bmatrix}, \underline{B} = \begin{bmatrix} \frac{1}{L_a} & 0 \\ 0 & -\frac{1}{J} \end{bmatrix} = \begin{bmatrix} 333 & 0 \\ 0 & -12500 \end{bmatrix}$$

$$\underline{C} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \underline{D} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \underline{u} = \begin{bmatrix} U \\ T_l \end{bmatrix}, \underline{z} = \begin{bmatrix} i_a(t) \\ \omega(t) \end{bmatrix}, \underline{y} = \begin{bmatrix} i_a(t) \\ \omega(t) \end{bmatrix}$$

### 3 The Fault Diagnosis Problem

This section is devoted to the definition of the fault diagnosis problem. In particular, the fault scenarios, along with the simulations of the system in healthy and faulty conditions, are presented.

### 3.1 The Fault Scenarios

Three *fault scenarios* are considered and simulated by modifying the nominal values of the physical parameters of the system, as described in the following [1–4]:<sup>2</sup>

- The first fault scenario corresponds to an increase of the armature resistance  $R_a$  by 20% over the nominal value.
- The second fault scenario corresponds to an increase of the armature inductance  $L_a$  by 200% over the nominal value.
- The last fault scenario corresponds to a decrease of both the electromotive force constant  $K_e$  and the motor torque constant  $K_t$  by 25% under the nominal values.

The aforementioned scenarios may physically correspond to brush faults, connection faults, short circuits or broken coils in armature, short circuits in field winding, to name a few.<sup>3</sup> Note that, in all three examined fault scenarios, a fluctuation of  $\pm 3\%$  of the corresponding modified parameter (i.e.,  $R_a$ ,  $L_a$ ,  $K_e$  and  $K_t$ ) is considered, in order to capture a range of operating disturbances, testing the robustness of the proposed ANN.

Eventually, a total of 12 samples for the training of the ANN were utilized; 3 samples for each one of the four operating states of the motor (one healthy and three faulty).

### 3.2 Simulations of the Healthy and Faulty System

The output (time) response of the system is derived using the MATLAB's `lsim` function, at a sample rate  $f_s = 1\text{ kHz}$ . The time vector contains 1001 points (from  $t = 0\text{ s}$  to  $t = 1\text{ s}$ ). Each output signal is *normalized* in the interval  $[0, 1]$ , in order to “feed” the input layer of the ANN, with all inputs being at an unbiased comparable range.

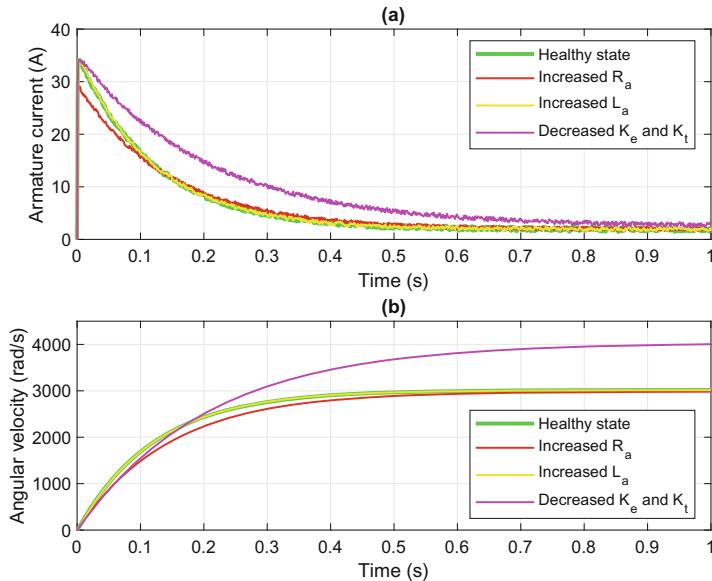
The DC input voltage  $U$  and the load torque  $T_l$  are set equal to 200 V and 0.1 N m, respectively. Their fluctuations,  $\pm 2\%$  for the voltage and  $\pm 10\%$  for the load torque, are approximated by zero-mean uniform white noise, and represent the varying operating conditions.

Indicative armature current and angular velocity responses for all four operating states of the system (one healthy and three faulty) are depicted in Fig. 2. A *Fast Fourier Transform* (FFT) is employed in order to study the frequency content of the two output signals, for all four states of the system. The results are illustrated in Fig. 3.

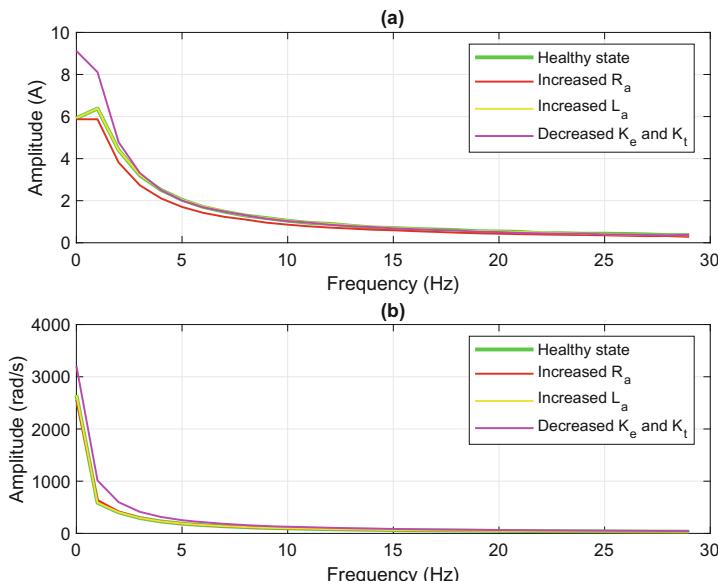
---

<sup>2</sup> The model of the DC motor is exclusively used for the data generation of the healthy and faulty system.

<sup>3</sup> Brush faults are quite common failures in DC motors, and they are typically a consequence of unsMOOTH contact of the brushes, dirty collector or brushes, unadjusted brush pressure springs, oval shaped collector and consumed brush life [1].



**Fig. 2.** Indicative waveforms of the armature current (a) and angular velocity (b), for all four operating states of the system.



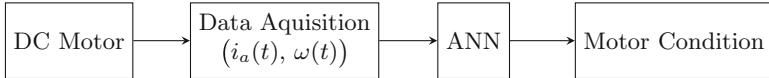
**Fig. 3.** Spectrum of the armature current (a) and angular velocity (b), for all four operating states of the system.

As is evident from Figs. 2 and 3, for all four scenarios, the waveforms of the healthy and faulty system, as well as their frequency content, are *almost identical*, indicating a highly challenging fault detection problem.<sup>4</sup>

## 4 The Artificial Neural Network

The adopted technique for fault diagnosis in DC motors is depicted in Fig. 4. The suggested method is composed of the following steps:

- Firstly, the data (armature current and angular velocity signals) is obtained from the DC motor and normalized in the interval  $[0, 1]$ , in order to appropriately drive the ANN.
- Subsequently, the normalized signals are imported in the (input layer of the) ANN, without any analysis or pre-processing.
- Finally, the motor condition is decided as an output of the ANN.



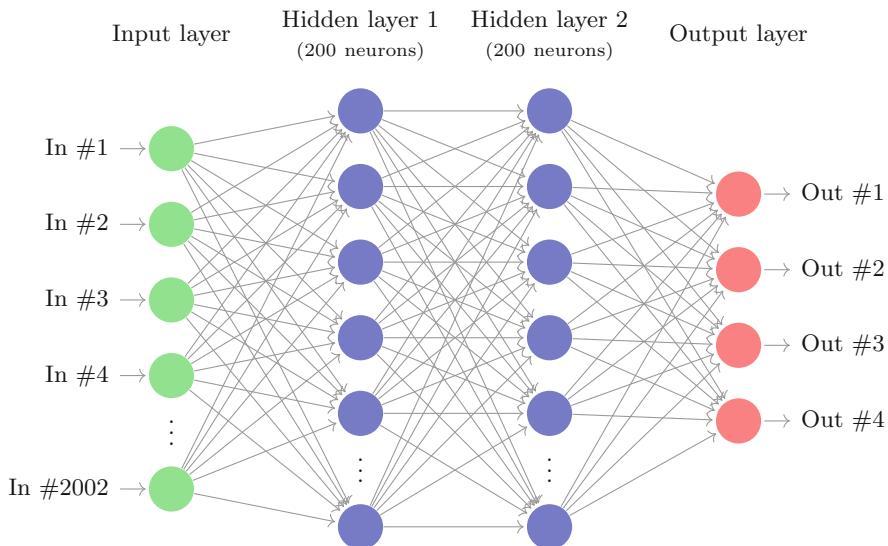
**Fig. 4.** Block diagram of the proposed technique for fault diagnosis in DC motors.

For training and evaluating the ANN, the contemporary and powerful machine learning framework *TensorFlow* is utilized, along with the high-level neural networks API, *Keras*.<sup>5</sup>

The ANN is a *four-layer* (one *input layer*, two *hidden layers* and one *output layer*), *feed-forward, fully-connected* neural network, as depicted in Fig. 5. The input layer of the network is a **Flatten** layer, which merely transforms the format of the waveforms from a 2D-array (of 1001 by 2 values) to a 1D-array of  $1001 \cdot 2 = 2002$  values. Recall that the signals of the input layer are the armature current and the angular velocity of the DC motor, and the length of each signal is 1001 points. The **Flatten** layer has no parameters to learn; it only reformats the data. After this layer, the network consists of a sequence of three **Dense** layers. The first two **Dense** (hidden) layers have 200 nodes (or neurons) each, with a *rectifier* activation function (**relu**). The third (output) layer is a 4-node *softmax* layer; each node contains a score that indicates the probability that the current waveform belongs to one of the four fault classes (see Table 2 below). The network weights are initialized to a small random number generated from a uniform distribution, in this case between 0 and 0.05.

<sup>4</sup> More evidently for the healthy case and the faulty scenario where  $L_a$  is increased by 200%.

<sup>5</sup> TensorFlow was developed by the Google Brain team, and can be found at <https://www.tensorflow.org>.



**Fig. 5.** Topology of the ANN.

After the construction of the model, it is compiled using *logarithmic loss* as the classification loss function, which for a categorical classification problem is defined in Keras as `sparse_categorical_crossentropy`. The optimization algorithm used is the `adam` optimization algorithm, which is an extension of *stochastic gradient descent*.

#### 4.1 Training Phase

As mentioned above, the input data (armature current and angular velocity) for training the ANN is constructed of successive ranges of 12 samples, representing the four states of the operating conditions of the DC motor. That is, healthy state (3 samples), increased armature resistance (3 samples), increased armature inductance (3 samples), and decreased electromotive force and motor torque constants (3 samples).

Each motor state is mapped to a single target of the output layer of the ANN, as it is shown in Table 2. Each target is represented by a set of a binary state of each output neuron of the network.

The accuracy of the model on the training data reached 100%, with a total training time of about 5 s (500 epochs of about 10 ms each).

#### 4.2 Test Results and Predictions

The test dataset contains 1000 samples (250 samples for each of the four operating conditions of the system) that are *not contained* in the training dataset.

**Table 2.** Targets of the output layer of the ANN and fault classes.

Target	Class No.	Motor condition
$\{1, 0, 0, 0\}$	1	Healthy state
$\{0, 1, 0, 0\}$	2	Increased $R_a$ by 20%
$\{0, 0, 1, 0\}$	3	Increased $L_a$ by 200%
$\{0, 0, 0, 1\}$	4	Decreased $K_e$ and $K_t$ by 25%

The accuracy of the model on test phase reached 99.7%, which indicates the successful training (without over-fitting).<sup>6</sup>

The confusion matrix (error matrix), presented in Table 3, allows visualization of the model's performance. A confusion matrix  $\mathcal{C}$  is such that  $\mathcal{C}_{i,j}$  is equal to the number of instances *known* to be in class  $i$ , but *predicted* to be in class  $j$ . The accuracy of the model on test phase can, also, be calculated/verified from Table 3 as follows:

$$\text{Accuracy} = \frac{\sum \text{Correct}}{\text{Total Samples}} = \frac{247 + 250 + 250 + 250}{1000} = 0.997 \text{ or } 99.7\%.$$

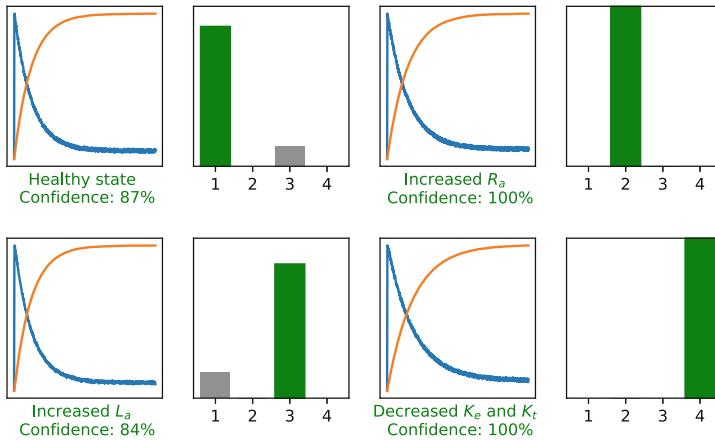
**Table 3.** Confusion matrix (error matrix) of the model.

	Class No. 1 (Predicted)	Class No. 2 (Predicted)	Class No. 3 (Predicted)	Class No. 4 (Predicted)
Class No. 1 (Actual)	247	0	3	0
Class No. 2 (Actual)	0	250	0	0
Class No. 3 (Actual)	0	0	250	0
Class No. 4 (Actual)	0	0	0	250

Figure 6 depicts the accuracy and confidence of predictions on four indicative samples (armature current and angular velocity) of the test dataset, where each sample belongs to a different fault class. Notice that, in these four cases, the model made correct predictions.

It is evident that the confidence level of the classification is 100% in the scenarios of Classes No. 2 and 4. For the scenarios of Classes No. 1 and 3, the

<sup>6</sup> Over-fitting is the production of an analysis that corresponds too closely or exactly to a particular set of data, and may, therefore, fail to fit additional data or predict future observations reliably.



**Fig. 6.** Accuracy and confidence of predictions on four indicative samples of the test dataset (armature current in blue, angular velocity in orange). Green label indicates correct prediction. (Color figure online)

confidence level is 87% and 84%, respectively. In this case, the model classifies the input signals of Class No. 1, with a confidence level of 13% (gray label in Fig. 6), as they were extracted from the Class No. 3, and that of Class No. 3, with a confidence level of 16%, as they were extracted from the Class No. 1. This is not surprising, since the scenarios of the healthy state and the faulty scenario of the increased armature inductance result in *almost identical* signals (both in time and frequency domain), as it was highlighted in Figs. 2 and 3. Table 3 also indicates the challenge of the discrimination between the input signals belonging to Class No. 1 and Class No. 3, as 3 instances were predicted to be in the latter, although known to be in the former.

The trained model can straightforwardly be used for instant evaluations of several motor states.

Clearly, the obtained results demonstrate that the proposed method achieves high accuracy in the combined problem of fault detection and classification, pinpointing a promising way to diagnose DC motors faults, in real-world applications.

## 5 Conclusions

In this paper, the combined problem of detection and classification of faults in a DC motor was investigated via an ANN-based method. Utilizing such a novel connectionist system, a high and accurate performance was achieved.

The main results of this study can be summarized as follows:

- Faults corresponding to 20% increase in the armature resistance, 200% increase in the armature inductance, and 25% decrease in both the

- electromotive force constant and the motor torque constant, with respect to the nominal values, are detectable at perfect (above 99.7%) detection rates.
- The achieved results were obtained with challenging output responses (waveforms) of the DC motor, in the sense that are both in time and frequency domain almost identical, and with no pre-process of the measured signals.
  - The ANN was trained with a small training dataset, and tested with a sufficiently large test dataset.
  - Disturbances and noise (for certain parameters of the system) were considered in simulations, in order to take into account the varying operating conditions.
  - Key to the high performance of the proposed method is the powerful ability of an ANN to identify patterns with high fidelity.
  - The ANN does not need a rigorous mathematical model of the underlying system for fault diagnosis.
  - The flexibility and speed of the presented method indicate that it can easily be applied to on-line fault diagnosis.
  - The training and evaluating of the ANN were employed in the contemporary and powerful tool TensorFlow, constituting a representative implementation of the state-of-the-art ANN modelling methodologies.

Future work is to be devoted to the extension of the fault scenarios, as well as to the application of the proposed technique to other types of electric motors.

## References

1. Bay, O.F., Bayir, R.: A fault diagnosis of engine starting system via starter motors using fuzzy logic algorithm. *Gazi Univ. J. Sci.* **24**, 437–449 (2011)
2. Filbert, D., Schneider, C., Spannhake, S.: Model equation and fault detection of electric motors. Technical report, IFAC Fault Detection, Supervision and Safety for Technical Processes, Baden-Baden, Germany (1991)
3. Isermann, R.: Fault diagnosis of electrical drives. In: Isermann, R. (ed.) *Fault-Diagnosis Applications*, pp. 49–80. Springer, Heidelberg (2011). [https://doi.org/10.1007/978-3-642-12767-0\\_3](https://doi.org/10.1007/978-3-642-12767-0_3)
4. Liu, X.Q., Zhang, H.Y., Liu, J., Yang, J.: Fault detection and diagnosis of permanent-magnet DC motor based on parameter estimation and neural network. *IEEE Trans. Ind. Electron.* **89**, 1021–1030 (2000)
5. Patan, K., Korbicz, J., Głowacki, G.: DC motor fault diagnosis by means of artificial neural network. In: Proceedings of the International Conference on Informatics in Control, Automation and Robotics, ICINCO 2007, pp. 11–18 (2007)
6. Samarasinghe, S.: *Neural Networks for Applied Sciences and Engineering: From Fundamentals to Complex Pattern Recognition*. Auerbach Publications, Boca Raton (2006)
7. Trigeassou, J.C.: *Electrical Machines Diagnosis*. Wiley, Hoboken (2013)
8. Yu, K., Yang, F., Guo, H., Xu, J.: Fault diagnosis and location of brushless DC motor system based on wavelet transform and artificial neural network. In: Proceedings of the 2010 International Conference on Electrical Machines and Systems. IEEE (2010)

# **1st PEINT Workshop**



# On Predicting Bottlenecks in Wavefront Parallel Video Coding Using Deep Neural Networks

Natalia Panagou<sup>1</sup> , Panagiotis Oikonomou<sup>1</sup> ,  
Panos K. Papadopoulos<sup>2</sup> , Maria Koziri<sup>1</sup> ,  
Thanasis Loukopoulos<sup>2</sup> , and Dimitris Iakovidis<sup>2</sup>

<sup>1</sup> Department of Computer Science and Telecommunications,  
University of Thessaly, Papasiopoulou 2-4, 35131 Lamia, Greece  
`{napanagou, paikonom, mkoziri}@uth.gr`

<sup>2</sup> Department of Computer Science and Biomedical Informatics,  
University of Thessaly, Papasiopoulou 2-4, 35131 Lamia, Greece  
`{ppapadopoulos, luke, diakovidis}@uth.gr`

**Abstract.** Video coding incurs high computational complexity particularly at the encoder side. For this reason, parallelism is used at the various encoding steps. One of the popular coarse grained parallelization tools offered by many standards is wavefront parallelism. Under the scheme, each row of blocks is assigned to a separate thread for processing. A thread might commence encoding a particular block once certain precedence constraints are met, namely, it is required that the left block of the same row and the top and top-right block of the previous row have finished compression. Clearly, the imposed constraints result in processing delays. Therefore, in order to optimize performance, it is of paramount importance to properly identify potential bottlenecks before the compression of a frame starts, in order to alleviate them through better resource allocation. In this paper we present a simulation model that predicts bottlenecks based on the estimated block compression times produced from a regression neural network. Experiments with datasets obtained using the reference encoder of HEVC (High Efficiency Video Coding) illustrate the merits of the proposed model.

**Keywords:** Wavefront · Video coding · HEVC · Deep neural networks

## 1 Introduction

The ever increasing demands for higher video resolution led to the proliferation of 4K cameras and TV sets. Unfortunately, the departure from the UHD era resulted in higher bandwidth demands, which the popular H.264/AVC standard [1] proved insufficient to handle. As an example, YouTube suggests a roughly 50MBps transmission rate for 4K videos coded with H.264/AVC [2]. Recognizing the necessity for a more efficient video coding standard the MPEG group launched the High Efficiency Video Coding standard [3] (also referred to as H.265). Similarly, AOMedia (a consortium of blue chip companies in the hi-tech industry), recently launched AV1 codec [4] as a royalty free

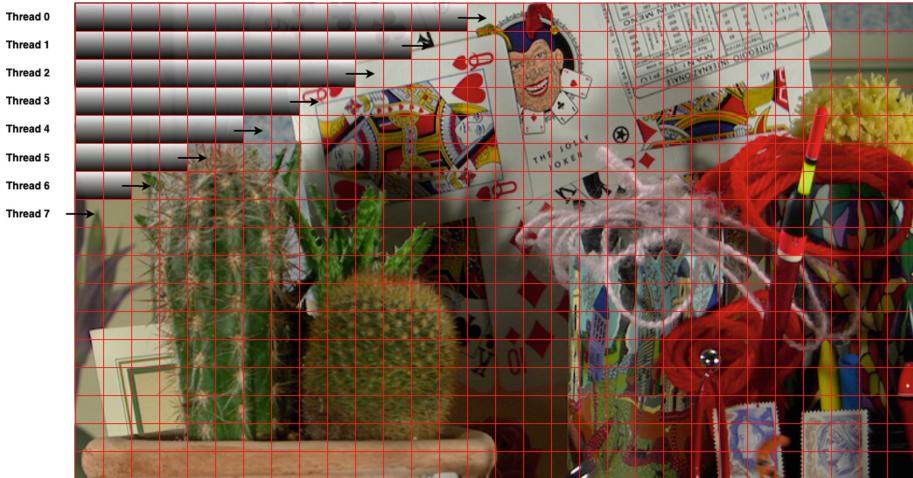
standard. The aforementioned standards have demonstrated a capacity to increase compression efficiency by up to 50% (for the same video quality) compared to H.264/AVC [5]. Furthermore, work on the successor of HEVC is ongoing under the term Versatile Video Coding (VVC) [6] which targets an additional 50% compression improvement, compared to HEVC and AV1.

While the latest developments in video standards have resulted in impressive performance as far as coding efficiency is concerned, these merits come at a particularly high computational overhead which can only be harnessed through parallelism. Such parallelism comes at various granularities, that can span all the way from the coarser level where whole frames or GOPs (Groups of Pictures) are considered as separate coding tasks (and assigned to different threads) [7], down to the finest level involving instruction based SIMD (Single Instruction Multiple Data) parallelism, e.g., at transform or SAD calculations [8].

Of particular interest are block based parallel methods. Under these methods, the compression of one or more blocks of pixels a frame is partitioned into, e.g.,  $16 \times 16$  Macroblocks in H.264/AVC or  $64 \times 64$  CTUs (Coding Tree Units) in HEVC, form independent tasks that can be processed in parallel. In HEVC, three such methods are provided and signaled in order for the decoder to take advantage thereof, namely slices [9], tiles [10] and wavefront [11]. Among the three, wavefront leads to better coding efficiency and has been included to practical codec implementations, e.g., x265 [12].

In wavefront, the codec takes advantage of the potential parallelization that is allowable based on prediction dependencies. For HEVC, this entails that in order to commence the compression of  $CTU_{ij}$  (the CTU positioned at the  $i^{\text{th}}$  row and  $j^{\text{th}}$  column), the processing of  $CTU_{i(j-1)}$  (left CTU),  $CTU_{(i-1)j}$  (top CTU) and  $CTU_{(i-1)(j+1)}$  (top-right CTU) must have finished. As can be shown in Fig. 1, the precedence constraints result in a row processing delay of at least two CTUs, which is unavoidable. However, Fig. 1 shows the rather ideal case where all threads proceed at the same speed. In fact, differences in compression times of particular CTUs can lead to further latencies. Depending on the available computational resources, such additional bottlenecks in wavefront processing can be alleviated provided they are identified in advance. For instance, in a heterogeneous environment whereby processing cores have different characteristics, a judicious resource allocation scheme could assign bottleneck CTUs to the fastest cores, leaving the rest to be processed by the slower ones.

In this paper we propose a model for identifying bottlenecks in wavefront processing. The model is based on predicting CTU compression times based on past history, using a regression neural network [13]. Predictions are then used to calculate the bottlenecks that would be perceived in the processing of each row, assuming each row is processed by a separate thread. Through experiments with dataset values obtained from the encoding of common test video sequences using the HM reference software for HEVC [14], it is illustrated that the model can achieve good prediction accuracy (correlation is more than 0.86). Furthermore, the bottleneck calculations incur negligible time overhead (in the order of few msec per frame) and can thus, be seamlessly incorporated in codecs. To the best of our knowledge this is the first work tackling wavefront parallelism from this standpoint.



**Fig. 1.** Example of wavefront parallel video compression (Cactus sequence).

The rest of the paper is organized as follows. Section 2 presents related work on block based parallelism. The proposed model is discussed in Sect. 3 and experimentally evaluated in Sect. 4. Finally, Sect. 5 concludes the paper.

## 2 Related Work

Block based parallelism in video coding has been exploited in the context of slices, tiles and wavefront. Slice partitioning was used in H.264/AVC and was inherited in HEVC. Slices essentially form sub-frames that can be coded and transmitted independently (prediction and entropy coding dependencies are broken on slice boundaries). In H.264/AVC a slice could be defined by taking Macroblocks in consecutive raster order, while in HEVC slices can also be defined as groups of tiles taken again in raster order. Since slices aim at facilitating network transmission they carry header information that is unnecessary in the other two block parallel methods. Therefore, from a bitrate perspective slices are expensive compared to tiles and wavefront thus, their use as parallelization granule although important in H.264/AVC [15] is rather limited in HEVC.

Tiles are defined by introducing an  $M \times N$  grid that splits a frame into  $M$  vertical and  $N$  horizontal zones each containing a number of CTU rows and columns. At the boundaries of a tile's rectangle, dependencies are broken enabling independent processing. A number of studies demonstrated the high parallelization potential that is achievable by tile parallelization. In [16] the authors proposed to use a large number of tiles in order to smooth load imbalance experienced at CPU cores. Since using large number of tiles hinders coding efficiency many works advocated the use of a number of tiles equaling the available processing cores. In [17] a tile resizing scheme was proposed with the aim of reducing load imbalances. The scheme was based on estimating

CTU compression time from the past average. Prediction methods and partitioning algorithms specifically tuned for LowDelay (LD) video coding were proposed in [18] and [19]. In [20] tile resizing was considered with the aim of reducing coding losses. Finally, in [21] a tile resizing and CPU scheduling algorithm is illustrated that aims at improving speedup when the number of processors is fewer than the number of tiles.

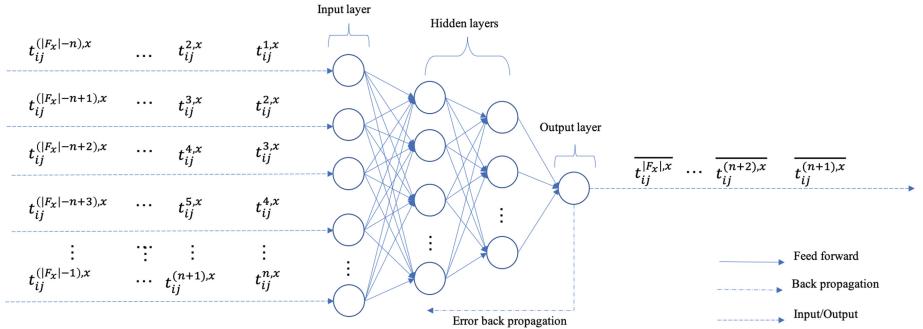
Wavefront Parallel Processing (WPP) was proposed in [22] for H.264. In WPP threads are assigned complete rows thus, whenever precedence constraints are not fulfilled a thread pauses execution which can be wasteful particularly under limited processing resources. Overcoming this deficiency was the aim of [23] which studied WPP in HEVC. The authors advocated the use of thread pooling and thread assignment consisting of one CTU each time. In case precedence constraints do not allow a thread to proceed, the thread returns to the pool. In this way, waiting overheads are avoided but other overheads are introduced due to the fact that a finishing thread must return to the pool before being assigned another CTU. In [11] Overlapping Wavefront (OWF) was proposed and evaluated in the decoder side. In WPP a thread finishing its assigned row returns in case no other row can be assigned. On the other hand, in OWF it proceeds by processing the first row of the next frame. Expanding the parallelization potential of wavefront was the aim of [24] and [25] where precedence constraints were modeled not only within a single frame (as per WPP), but also between a frame and its reference frames. Thus, the scope of wavefront was expanded to account for parallelism over multiple frames.

We view the work of this paper as complimentary to the aforementioned papers. Wavefront can in principle be not only applied on a per frame basis, but also on a per tile or slice basis. Regardless of the application basis or the parallelization scope (one or more frames), bottlenecks can occur from the enforcement of precedence constraints. Thus, predicting them in advance can contribute in their alleviation through adequate resource allocation and/or task assignment strategies. In this paper we focus on WPP as the baseline wavefront mechanism, nevertheless our approach is applicable to the other schemes of the literature with only small changes.

### 3 Wavefront Bottleneck Prediction

#### 3.1 Prediction Model

Since deep learning techniques [26] have been used successfully to tackle problems related to regression [27] and forecasting [13], in this paper a deep neural network (DNN) is presented to predict the coding times of CTUs. The DNN consists of two hidden layers with the same number of nodes. For each CTU we are interested on predicting one value (time). Therefore, the output layer consists of one node (regression problem). The input layer has a tunable number of nodes (let  $n$ ), each representing a previously observed value at a preceding frame (actual coding time of the CTU that is to be predicted). In the experiments we built our model based on the previous 8 coding times recorded, i.e.,  $n = 8$ . Our model consists of  $|S|$  input video sequences. Let  $S_x$  be the  $x^{\text{th}}$  video sequence assuming a total ordering of them  $1 \leq x \leq |S|$ . Each  $S_x$  consists of a set of  $|F_x|$  frames. Let  $F_{kx}$  be the  $k^{\text{th}}$  frame of  $S_x$  assuming a total ordering



**Fig. 2.** CTU prediction DNN.

of them  $1 \leq k \leq |F_x|$ . Each frame is divided into  $W \times H$  CTUs arranged in  $W$  columns and  $H$  rows. Let  $CTU_{ij}^{kx}$  denote the CTU positioned in the  $i^{\text{th}}$  row and the  $j^{\text{th}}$  column of  $F_{kx}$ , with  $0 \leq i \leq H - 1$  and  $0 \leq j \leq W - 1$ . Furthermore, let  $t_{ij}^{kx}$  denote the actual compression time of  $CTU_{ij}^{kx}$  and  $\bar{t}_{ij}^{kx}$  the predicted time.

When training multilayer networks, the general practice is to first divide the data into two subsets namely, a training set used for the network weights and a testing set. The training set consist of the first  $u$  input video sequences while the testing set by the following  $|S| - u$ . In each iteration, the DNN is fed with  $n$  input values and information moves forward from the input nodes, through the two hidden layers, to the output node. Prediction failure rate in the form of mean square error (MSE), is computed in the output node and distributed backwards throughout the network's layers (back propagation). All neurons in the hidden layers share the same activation function (sigmoid). Figure 2 depicts graphically the feed-forward deep neural network with back propagation that is used for prediction.

### 3.2 Bottleneck Delay Calculation

Having obtained the estimations for the CTU compression times, we can calculate the potential bottlenecks incurred in a wavefront parallel execution. Without loss of generality, we demonstrate the process (and notation) for a generic frame. The same process can be applied to any frame  $F_{kx}$ .

Let  $EST_{ij}$  denote the earliest start time the compression of  $CTU_{ij}$  can commence and  $t_{ij}$  the required time to process  $CTU_{ij}$ . The following holds:

$$EST_{ij} = \begin{cases} 0, & i = 0, j = 0 \\ EST_{i(j-1)} + t_{i(j-1)}, & i = 0 \\ \max((EST_{i(j-1)} + t_{i(j-1)}), (EST_{(i-1)(j+1)} + t_{(i-1)(j+1)})), & i > 0, j < W - 1 \\ \max((EST_{i(j-1)} + t_{i(j-1)}), (EST_{(i-1)j} + t_{(i-1)j})), & i > 0, j = W - 1 \end{cases} \quad (1)$$

Assuming each row is assigned to a separate thread, we are interested in quantifying the induced delays due to precedence constraints. Since row processing happens in a left to right manner, only delays incurred by the top-right constraint matter, in the sense that their overhead (if any) will be added to the processing times of the CTUs and will thus, affect the processing time of the row.

Let  $D_{ij}$  denote the aforementioned delay captured as the difference between the time  $CTU_{i(j-1)}$  finishes processing (left constraint is satisfied) and  $EST_{ij}$ . For the first CTU of a row there exists no left constraint (no CTU precedes it). We use as left constraint the time when the processing of the above row commences, which results in the following definition:

$$D_{ij} = \begin{cases} 0, i = 0 \\ EST_{(i-1)(j+1)} + t_{(i-1)(j+1)} - (EST_{i(j-1)} + t_{i(j-1)}), i > 0, 0 < j < W - 1 \\ EST_{(i-1)j} + t_{(i-1)j} - (EST_{i(j-1)} + t_{i(j-1)}), i > 0, j = W - 1 \\ EST_{(i-1)(j+1)} + t_{(i-1)(j+1)} - EST_{(i-1)j}, i > 0, j = 0 \end{cases} \quad (2)$$

We can calculate the delay values using (1) and (2) by iterating on the CTUs of a frame in raster order, for a time complexity of  $O(WH)$ .

**Table 1.** Video sequences.

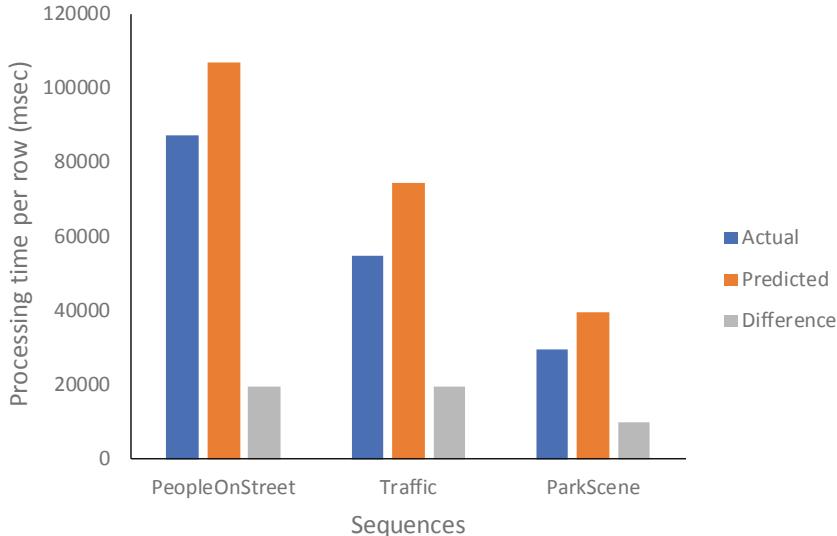
Sequence	Resolution	FPS	Frames	CtuCols	CtuRows
PeopleOnStreet	2560 × 1600	30	150	40	25
Traffic	2560 × 1600	30	150	40	25
BasketballDrive	1920 × 1080	50	500	30	17
BQTerrace	1920 × 1080	60	600	30	17
Cactus	1920 × 1080	50	500	30	17
Kimono	1920 × 1080	24	240	30	17
ParkScene	1920 × 1080	24	240	30	17

## 4 Experiments

### 4.1 Setup

We performed simulation based experiments using dataset obtained from real video encodings. The encodings were run on a Linux Server with two 12-core Intel Xeon E5-2650 running at 2.20 GHz. Class A and B test sequences were used for our evaluation with characteristics summarized on Table 1. HM 16.15 reference software for HEVC was used with the following settings: LowDelay (LD) setup [28] with one slice, first

frame I followed by P frames, GOP size was 4, bit depth was 8, CTU size was  $64 \times 64$ , max partitioning depth was 4, search mode was TZ and QP was set to 32.



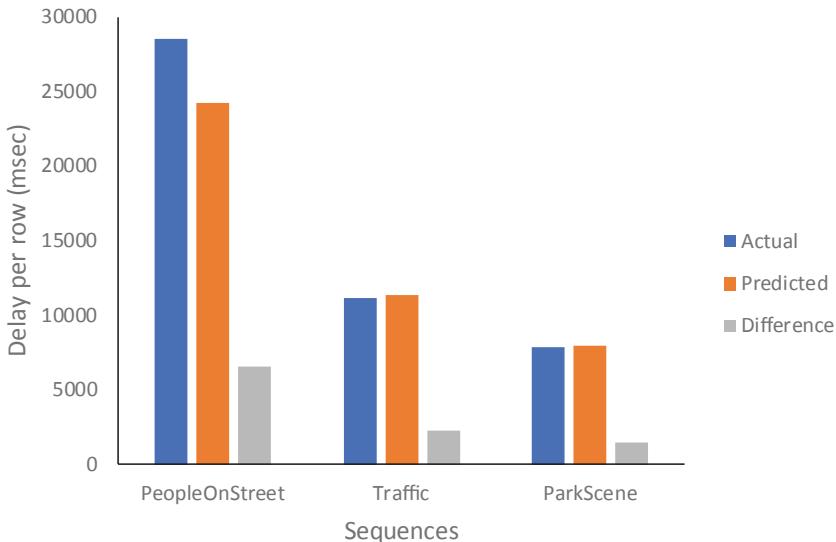
**Fig. 3.** Average row processing times.

## 4.2 Training Results

The neural network was implemented at Weka [29]. In every frame of a sequence the compression time of all CTUs is recorded and used subsequently for train or testing purposes. The following 4 Class B sequences were used for training: BasketballDrive, BQTerrace, Cactus and Kimono. In total 920,000 instances were used for training. The remaining Class B sequence (ParkScene) together with Class A sequences were used for testing. Table 2 summarizes the testing results.

**Table 2.** Testing results.

Sequences	Correlation coefficient	Mean absolute error (MAE)	Root mean squared error (RMSE)	Total number of instances
PeopleOnStreet	0.8628	22.3398	25.4731	142000
Traffic	0.8951	20.1156	21.3562	142000
ParkScene	0.8969	20.3566	22.1009	118320



**Fig. 4.** Average row delay.

### 4.3 Delay Results

Having obtained both the dataset with the actual compression times and the dataset with the values predicted by the DNN, next, we evaluated for the validation sequences the efficiency of the proposed methodology in predicting row delays. First, in Fig. 3 we plot the average processing time per row, calculated as the summation of the compression times of the corresponding CTUs. Both the actual and the predicted values are plotted together with the average per row absolute difference between the observed processing times and the predicted ones. As it can be observed, the predicted value at every row is higher (roughly 20 s (on average) for Class A sequences) compared to the recorded values. Regardless of the fact that predicted values tend to be overestimated, the introduced margin of differences is acceptable for encoding purposes. Furthermore, for the particular problem tackled in this paper, i.e., delay calculations, it is important to identify the relative differences in CTU processing times at each row rather than the actual values. Figure 4 plots the average delay per row. By comparing Figs. 3 and 4 it can be stated that the calculated delays account for a significant portion of the net processing time from CTU compression (roughly one third for PeopleOnStreet). The aforementioned fact outlines the importance of the problem tackled in this paper. Notice, that in ParkScene and Traffic the average delay per row of actual and predicted values almost coincide. Furthermore, the per row absolute differences between actual and predicted values is particularly small. This is a very encouraging result that clearly illustrates the merits of the proposed model.

## 5 Conclusions

In this paper we tackled the problem of identifying the bottlenecks in wavefront parallel video encoding. The proposed method involves the use of a deep neural network for predicting the compression time of the blocks a frame is split into and the subsequent calculation of expected delays due to precedence constraints. Experiments using real datasets obtained from HEVC coding, demonstrated that the resulting model can predict wavefront delays with relative accuracy.

**Acknowledgments.** This research has been co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH-CREATE-INNOVATE (project code: T1EDK-02070).

## References

1. Wiegand, T., Sullivan, G.J., Bjontegaard, G., Luthra, A.: Overview of the H. 264/AVC video coding standard. *IEEE Trans. Circ. Syst. Video Technol.* **13**, 560–576 (2003)
2. YouTube, Recommended upload encoding settings. <https://support.google.com/youtube/answer/1722171>
3. Sullivan, G.J., Ohm, J.R., Han, W.J., Wiegand, T.: Overview of the high efficiency video coding (HEVC) standard. *IEEE Trans. Circ. Syst. Video Technol.* **22**, 1649–1668 (2012)
4. AV1: Bitstream & Decoding Process Specification (2018). <https://aomedia.org/av1-bitstream-and-decoding-process-specification/>
5. Topiwala, P., Krishnan, M., Dai, W.: Performance comparison of VVC, AV1 and HEVC on 8-bit and 10-bit content. In: Applications of Digital Image Processing XLI International Society for Optics and Photonics, vol. 10752, p. 107520 (2018)
6. VVC: Versatile Video Coding (2018). <https://jvet.hhi.fraunhofer.de/>
7. Franche, J.F., Coulombe, S.: A multi-frame and multi-slice H. 264 parallel video encoding approach with simultaneous encoding of prediction frames. In: 2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet), pp. 3034–3038. IEEE (2012)
8. Lemmetti, A., Koivula, A., Viitanen, M., Vanne, J., Hämäläinen, T.D.: AVX2-optimized Kvazaar HEVC intra encoder. In: IEEE International Conference on Image Processing (ICIP), pp. 549–553 (2016)
9. Koziri, M.G., Papadopoulos, P., Tziritas, N., Dadaliaris, A.N., Loukopoulos, T., Khan, S.U.: Slice-based parallelization in HEVC encoding: realizing the potential through efficient load balancing. In: 18th IEEE International Workshop on Multimedia Signal Processing (MMSP), pp. 1–6 (2016)
10. Misra, K.M., Segall, C.A., Horowitz, M., Xu, S., Fuldsseth, A., Zhou, M.: An overview of tiles in HEVC. *IEEE J. Sel. Top. Signal Process.* **7**(6), 969–977 (2013)
11. Chi, C.C., et al.: Parallel scalability and efficiency of HEVC parallelization approaches. *IEEE Trans. Circ. Syst. Video Technol.* **22**, 1827–1838 (2012)
12. x265 HEVC encoder (2018). <http://x265.org>
13. Qiu, X., Zhang, L., Ren, Y., Suganthan, P.N., Amaralunga, G.: Ensemble deep learning for regression and time series forecasting. In: 2014 IEEE Symposium on Computational Intelligence in Ensemble Learning, pp. 1–6 (2014)
14. HM reference software. <http://hevc.hhi.fraunhofer.de>

15. Zhao, L., Xu, J., Zhou, Y., Ai, M.: A dynamic slice control scheme for slice-parallel video encoding. In: ICIP 2012, pp. 713–716 (2012)
16. Shafique, M., Khan, M.U.K., Henkel, J.: Power efficient and workload balanced tiling for parallelized high efficiency video coding. In: ICIP 2014, pp. 1253–1257 (2014)
17. Storch, I., Palomino, D., Zatt, B., Agostini, L.: Speedup-aware history-based tiling algorithm for the HEVC standard. In: ICIP 2016, pp. 824–828 (2016)
18. Koziri, M., et al.: Adaptive tile parallelization for fast video encoding in HEVC. In: Proceedings of the 12th International Conference on Green Computing and Communications (GreenCom), pp. 738–743 (2016)
19. Koziri, M., et al.: Heuristics for tile parallelism in HEVC. In: 25th European Signal Processing Conference (EUSIPCO), pp. 1514–1518 (2017)
20. Blumenberg, C., Palomino, D., Bampi, S., Zatt, B.: Adaptive content-based tile partitioning algorithm for the HEVC standard. In: PCS 2013, pp. 185–188 (2013)
21. Papadopoulos, P.K., Koziri, M.G., Loukopoulos, T.: A fast heuristic for tile partitioning and processor assignment in HEVC. In: Proceedings of the IEEE International Conference on Image Processing (2018)
22. Zhao, Z., Liang, P.: Data partition for waveform parallelization of H. 264 video encoder. In: Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS), p. 4 (2006)
23. Zhao, Y., Song, L., Wang, X., Chen, M., Wang, J.: Efficient realization of parallel HEVC intra encoding. In: Proceedings of the IEEE International Conference Multimedia and Expo Workshops (ICMEW), pp. 1–6 (2013)
24. Wang, Z.Y., Dong, S.F., Wang, R.G., Wang, W.M., Gao, W.: Dynamic macroblock waveform parallelism for parallel video coding. *J. Vis. Commun. Image Represent.* **28**, 36–43 (2015)
25. Wen, Z., Guo, B., Liu, J., Li, J., Lu, Y., Wen, J.: Novel 3D-WPP algorithms for parallel HEVC encoding. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1471–1475 (2016)
26. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. *Nature* **521**(7553), 436 (2015)
27. Lv, Y., Duan, Y., Kang, W., Li, Z., Wang, F.Y.: Traffic flow prediction with big data: a deep learning approach. *IEEE Trans. Intell. Transp. Syst.* **16**(2), 865–873 (2015)
28. Bossen, F.: Common test conditions and software reference configurations. In: Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, 5th meeting (2011)
29. Frank, E., Hall, M.A., Witten, I.H.: The WEKA Workbench. Online Appendix for ‘Data Mining: Practical Machine Learning Tools and Techniques, 4th edn. Morgan Kaufmann, Burlington (2016)



# Recognizing Human Actions Using 3D Skeletal Information and CNNs

Antonios Papadakis<sup>1</sup>, Eirini Mathe<sup>2,5</sup>, Ioannis Vernikos<sup>3</sup>, Apostolos Maniatis<sup>4</sup>,  
Evaggelos Spyrou<sup>2,4(✉)</sup>, and Phivos Mylonas<sup>5</sup>

<sup>1</sup> Department of Informatics and Telecommunications,  
University of Athens, Athens, Greece  
[sdi1400141@di.uoa.gr](mailto:sdi1400141@di.uoa.gr)

<sup>2</sup> Institute of Informatics and Telecommunications,  
National Center for Scientific Research - “Demokritos”, Athens, Greece  
[{emathe,espyrou}@iit.demokritos.gr](mailto:{emathe,espyrou}@iit.demokritos.gr)

<sup>3</sup> Department of Computer Science and Telecommunications,  
University of Thessaly, Lamia, Greece  
[ivernikos@uth.gr](mailto:ivernikos@uth.gr)

<sup>4</sup> General Department, University of Thessaly, Lamia, Greece  
[amaniatis@teiste.gr](mailto:amaniatis@teiste.gr)

<sup>5</sup> Department of Informatics, Ionian University, Corfu, Greece  
[fmylonas@ionio.gr](mailto:fmylonas@ionio.gr)

**Abstract.** In this paper we present an approach for the recognition of human actions targeting at activities of daily living (ADLs). Skeletal information is used to create images capturing the motion of joints in the 3D space. These images are then transformed to the spectral domain using 4 well-known image transforms. A deep Convolutional Neural Network is trained on those images. Our approach is thoroughly evaluated using a well-known, publicly available challenging dataset and for a set of actions that resembles to common ADLs, covering both cross-view and cross-subject cases.

**Keywords:** Human action recognition · Convolutional neural networks · Skeletal data

## 1 Introduction

Understanding of human actions from video has attracted an increasing interest during the last few years. This research area lies in the broader field of human-centered activity recognition and combines ideas and techniques mainly from the fields of computer vision and pattern recognition. There exist several human action understanding tasks. Wang et al. [23] proposed a categorization into the following sub-problems: gesture, action, interaction and group activity recognition. The performance of a gesture requires a relatively small amount of time, while the performance of an action requires a significant amount of time and contrary to a gesture, it typically involves more than one body parts. Moreover,

an interaction involves either a person and an object or two persons. Finally, a group activity may be defined as any combination of the aforementioned. Open challenges in the area of human action understanding include the representation, the analysis and the recognition of the actions [2], while a broad field of applications such as surveillance, assisted living, human-machine interaction, affective computing etc. have benefited.

Earlier recognition approaches such as the one of Schuldt et al. [19] were based on hand-crafted features, used to train traditional machine learning algorithms such as Support Vector Machines (SVMs). However, as it has been demonstrated, the accuracy of such approaches is drastically reduced, when the number of the actions significantly increases. Moreover, they may be unable to provide robustness to viewpoint changes, a case typically encountered in real-life scenarios. Recent advances in hardware and more particularly in graphics processing units (GPUs) have enabled fast training of deep neural network architectures [9]. Such approaches do not strictly require a feature extraction step; instead, features are “learnt” within the network. Also, their accuracy significantly increases with the increase of the available training examples.

In this paper, we build on previous works [17, 18] and propose the use of visual representations of human actions, based on well-known 2D image transformations. More specifically, we use the Discrete Fourier Transform (DFT), the Fast Fourier Transform (FFT), the Discrete Cosine Transform (DCT) and the Discrete Sine Transform (DST). First, we create raw signal images which capture the 3D motion of human skeletal joints over space and time. Then, one of the aforementioned transforms is applied into each of the signal images, resulting to an “activity” image, which captures the spectral properties of signal images. For classification, we propose a deep CNN architecture. We evaluate the proposed approach using the challenging PKU-MMD dataset [15] and present results for cross-subject and cross-view cases. We demonstrate that the proposed approach may be used in real-like environments for the recognition of activities of daily living (ADLs) [12].

The rest of this paper is organized as follows: Sect. 2 presents related research works in the field of human action recognition, limited to those that are based on visual representations of 3D skeletal information and CNN. Next, Sect. 3 presents the proposed methodology, i.e., the 2D representation of a skeleton based on a set of 3D joints, the construction of image representations capturing skeletal motion in space and time. Also, the CNN architecture is presented therein. Experiments are presented in Sect. 4 and are discussed in Sect. 5, where plans for future work are also included.

## 2 Related Work

As it has already been mentioned in Sect. 1, when working with deep learning approaches, a large-scale multi-class dataset may be the key to effectiveness and robustness. The first publicly available datasets such as the KTH [19], were limited to a small number of simple actions e.g., *walking*, *running*, *hand clapping*

etc.. Later, a next series of datasets such as the Hollywood dataset [11] targeted more realistic human actions e.g., *answer phone*, *get out of car*, *hand shake* etc., still being limited to a small number of classes. In less than a decade, more challenging datasets such as the UCF101 [21] and the HMDB [10] emerged, containing large numbers of more complex actions, including interactions with objects such as *playing cello*, *horse riding*, *swing baseball bat*, *fencing* etc. Recent large-scale datasets such as PKU-MMD [15] or the NTU [20], are comprised of large numbers of training video and depth sequences.

According to Wang et al. [23] human action recognition tasks may be divided into two major categories:

- **segmented recognition:** the given input video sequence contains *only* the action to be recognized. This means that any frame before/after the action, i.e., not depicting a part of the action, has been removed. In this case, Recurrent Neural Networks (RNNs) [5] or CNNs [13] are typically used.
- **continuous recognition:** the goal is to recognize actions within a given video; the video may or may not depict a single action. In that case, also known as “online” recognition, RNNs are typically used.

Note that when a CNN is used and the only available motion features are skeletal data, an intermediate visual representation of skeletal sequences is required. This representation should capture both spatial and temporal information regarding the motion of joints, i.e., in the 3D space over time. This information should be reflected to its color and/or texture properties. In this section our goal is to present research works that are based on visual representations of 3D skeletal data of human actions and training deep networks, i.e., an intermediate hand-crafted feature extraction step is not included in the process. Skeletal data typically consist of a set of skeletal joints moving in 3D space over time, i.e., for each joint 3 1D signals are generated per action. The extraction of joints from video requires depth information.

In the work of Du et al. [4], in order to preserve the spatial information the set of joints is split into five subsets corresponding to arms, legs and the trunk. Pseudo-colored images are generated by corresponding  $x$ ,  $y$  and  $z$  spatial coordinates to R, G and B components, respectively. To preserve temporal information, spatial representations are chronologically arranged. Wang et al. [24] proposed a representation of “joint trajectory maps,” wherein the motion direction is encoded by hue. Maps were constructed by appropriately setting saturation and brightness so as texture would correspond to motion magnitude; each was based on the projected trajectory of the skeleton to a Cartesian plane. Similarly, Hou et al. [6] transformed the extracted skeleton joints into a representation called “skeleton optical spectra,” so that hue changes would reflect to the temporal variation of skeletal motion. Li et al. [14] proposed the representation of “joint distance maps,” and opted for encoding the pair-wise joint distances in the 3 orthogonal 2D planes and also used a fourth one to encode distances in the 3D space, while hue was used to encode distance variations. Each map was separately classified and a late fusion scheme was adopted. In an effort for invariance to the initial position and orientation of the skeleton, Liu et al. [16] applied

transforms to skeletal sequences. The representation of each joint consisted of its 3D space coordinates, also adding time and joint label to create a 5D space. Upon projection to a 2D image using two of the aforementioned dimensions, the remaining three were used as R, G, B values to form pseudo-colored images. Finally, Ke et al. [8] did not extract 3D coordinates. Instead, they extracted translation, rotation and scale invariant features by subsets of joints as in [4]. From each, they extracted cosine distances and normalized magnitudes from vector representations generated from pairwise relative positions between joints. These representations were concatenated to form a 2D representation.

### 3 Human Action Recognition

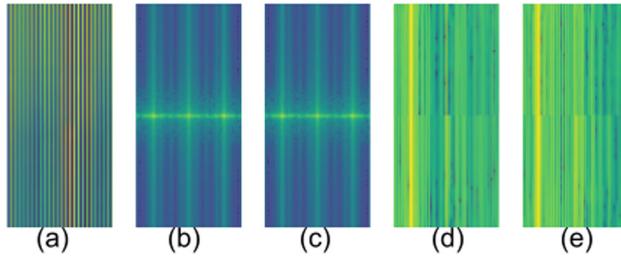
#### 3.1 Skeletal Information

The proposed approach requires as input 3D trajectories of skeletal joints (i.e.,  $x$ ,  $y$  and  $z$  coordinates for each) during an action. We work using data that have been captured with the Microsoft Kinect v2 sensor. More specifically, data consist of 25 human joints; up to 6 skeletons are simultaneously extracted in real time by using the Kinect SDK. A human skeleton corresponds to a graph; nodes correspond to body parts such as arms, legs, head, neck etc., while edges follow the body structure. Moreover, a parent-child relationship is implied. For example, the joint “HEAD” is parent of “NECK,” while the “NECK” is the parent of “SPINE\_SHOULDER,” etc. Each joint consists a 3D signal capturing its 3D position over time. Equivalently, this signal may be seen as 3 1D signals; each corresponding to a coordinate. Therefore, 75 such 1D signals result from the set of 25 joints and for any given video sequence. Note that their duration may vary, since different actions may require different amounts of time. Also different persons or even the same one may perform the same action with similar, yet not equal duration.

#### 3.2 Convolutional Neural Networks

Deep learning is a recent trend in the broader field of machine learning. It is based on the idea to use multiple intermediate interconnected layers within a network, so as to non-linearly process its input. In a sense, these layers are used to “learn” how to extract features in multiple levels of abstraction. During the last few years they have started playing a dominant role in several applications in areas such as computer vision, audio analysis, speech recognition etc., having successfully replaced traditional machine learning approaches. The latter often exhibit a drop of performance when used in real-life applications. During the last few years, research in practical computer vision problems has shifted to deep architectures.

When dealing with computer vision problems, the most popular approach is to train CNNs. A CNN resembles to traditional feed-forward networks; training takes place by forward and backward propagation of input data and error,



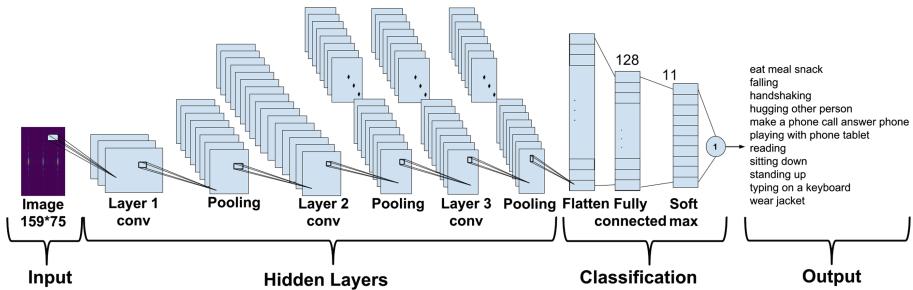
**Fig. 1.** (a) A signal image; activity image resulting upon (b) DFT; (c) FFT; (d) DCT; (e); DST. Action is *playing with phone/tablet*. DFT and FFT images have been processed with log transformation for visualization purposes. Figure best viewed in color. (Color figure online)

respectively. Their discrimination power lies to the fact that their *convolutional* layers are designed to learn a set of convolutional filters; during training, their parameters are learnt. Neurons are grouped into rectangular grids; each grid performs a convolution in a part of the input image. A *pooling* layer typically succeeds a single or a set of convolutional layers and sub-samples its input, to produce a single value from a small rectangular block. Finally, *dense* layers are those that are ultimately responsible for classification, based on the features that have been extracted by the convolutional layers and sub-sampled by the pooling ones.

### 3.3 Proposed Methodology

Our work has been partially inspired by the one of Jiang and Yin [7]. Similarly, we first create an image by concatenating the aforementioned 75 1D signals, which form the “signal” image. From each signal image we create an “activity” image, by applying one of the following transforms: (a) the 2D Discrete Fourier Transform (DFT); (b) the 2D Fast Fourier Transform (FFT); (c) the 2D Discrete Cosine Transform (DCT); and (d) the 2D Discrete Sine Transform (DST). From each transform we preserve only the magnitude, while we discard the phase. Note that DST and DCT are further processed by normalizing using the orthonorm. Obviously, in all cases the result is a 2D signal. Note that FFT is a fast implementation of DFT. Several implementations of FFT are approximations. Also even exact implementations are prone to floating point errors. Therefore, our goal was to assess whether the implementation of FFT we used showed a drop of performance compared to DFT. In Fig. 1 we illustrate an example signal image and the 4 corresponding activity images.

We herein remind that the goal of this work is limited to action classification. Therefore, it belongs to the category of segmented recognition (see Sect. 2), as it does not perform a temporal segmentation step. As we shall see in Sect. 4, we work using pre-segmented video sequences, aiming to only recognize the performed actions within each segment. We also assume that each segment contains exactly one action. To address the problem of temporal variability between



**Fig. 2.** The proposed CNN architecture.

actions and between users, an interpolation step is necessary. Upon experimentation, we set the duration of all videos equal to 159 frames, upon imposing a linear interpolation step. This way, the size of signal and activity images was fixed and equal to  $159 \times 75$ .

The architecture of the proposed CNN is presented in detail in Fig. 2. The first convolutional layer filters the  $159 \times 75$  input activity image with 32 kernels of size  $3 \times 3$ . The first pooling layer uses “max-pooling” to perform  $2 \times 2$  sub-sampling. The second convolutional layer filters the  $76 \times 34$  resulting image with 64 kernels of size  $3 \times 3$ . A second pooling layer uses “max-pooling” to perform  $2 \times 2$  sub-sampling. A third convolutional layer filters the  $36 \times 15$  resulting image with 128 kernels of size  $3 \times 3$ . A third pooling layer uses “max-pooling” to perform  $2 \times 2$  sub-sampling. Then, a flatten layer transforms the output image of the last pooling to a vector, which is then used as input to a dense layer using dropout. Finally, a second dense layer produces the output of the network. To avoid overfitting, the most popular approach, which is also adopted in this work is the use of the dropout regularization technique [22]: at each training stage several nodes are “dropped out” of the network. This way overfitting is reduced or even prevented, since complex co-adaptations on training data are prevented.

### 3.4 Implementation Details

For the implementation of the CNN we have used Keras [3] running on top of Tensorflow [1]. All data pre-processing and processing steps have been implemented in Python 3.6 using NumPy<sup>1</sup>, SciPy<sup>2</sup> and OpenCV.<sup>3</sup>

## 4 Experiments

### 4.1 Dataset

For the experimental evaluation of our approach we used the PKU-MMD dataset [15]. As it has already been mentioned, PKU-MMD is a large-scale benchmark

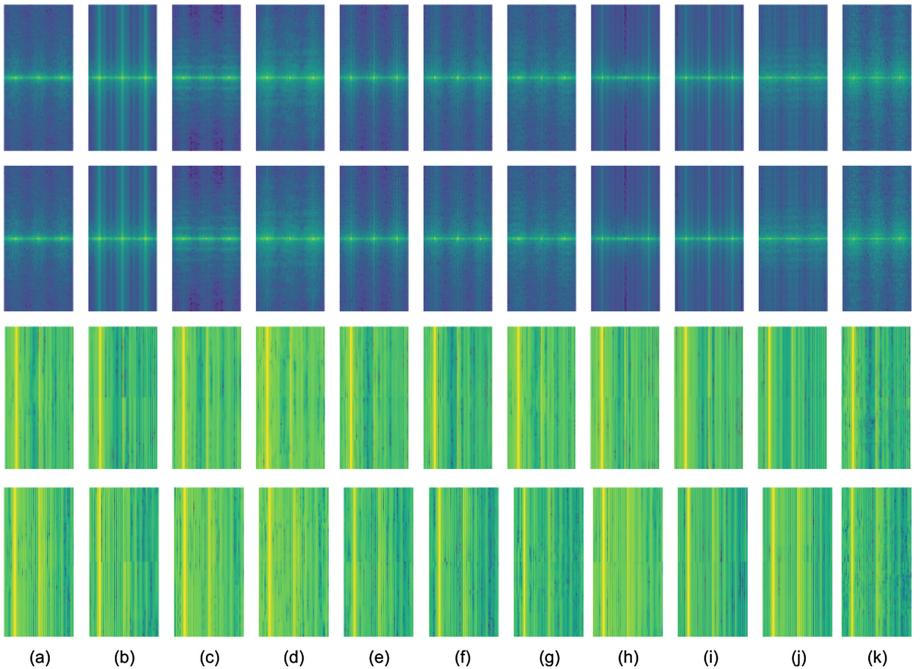
<sup>1</sup> <http://www.numpy.org/>.

<sup>2</sup> <https://www.scipy.org/>.

<sup>3</sup> <https://opencv.org/>.

focusing on human action understanding and containing approx. 20K action instances from 51 categories, spanning into 5.4M video frames. Note that 66 human subjects have participated in the data collection process, while each action has been recorded by 3 camera angles, using the Microsoft Kinect v2 camera. For each action example, raw RGB video sequences, depth sequences, infrared radiation sequences and extracted 3D positions of skeletons are provided.

Our experiments are divided into two parts. In the first part our goal was to assess whether the proposed approach may be used for ambient assistive living scenarios and more specifically for the recognition of ADLs. Therefore, we selected 11 out of the 51 classes of PKU-MMD, which we believe are the most close to ADLs or events in such a scenario. The selected classes are: *eat meal snack, falling, handshaking, hugging other person, make a phone call answer phone, playing with phone tablet, reading, sitting down, standing up, typing on a keyboard and wear jacket*. In Fig. 3 we illustrate sample signal and activity



**Fig. 3.** Examples of activity images from 11 classes and for the 4 transforms used. 1st row: DFT; 2nd row: FFT; 3rd row: DCT; 4th row: DST. (a) eat meal/snack; (b) falling; (c) handshaking; (d) hugging other person; (e) make a phone call/answer phone; (f) playing with phone/tablet; (g) reading; (h) sitting down; (i) standing up; (j) typing on a keyboard; (k) wear jacket. DFT and FFT images have been processed with log transformation for visualization purposes. Figure best viewed in color. (Color figure online)

images from these 11 classes and for all types of transforms. In the second part, we performed experiments with the whole dataset, i.e., with all 51 classes. In both parts, we worked only based on the skeletal data, discarding RGB, depth and infrared information.

## 4.2 Results

The evaluation protocol we followed is as follows: we first performed experiments per camera position; in this case both training and testing sets derived from the same viewpoint. Then, we performed cross-view experiments, where different viewpoints were used for training and for testing. The goal of these experiments was to test the robustness of the proposed approach in terms of transformation (e.g., a translation and a rotation), which could correspond to abrupt viewpoint changes which typically occur in real-life situations. Finally, we performed cross-subject experiments, where subjects were split in training and testing groups, i.e., any actor “participated” only into one of the groups. The goal of this subject was to test the robustness of our approach into intra-class variations. In real-life situations this is expected to happen when a system is trained e.g., within a laboratory environment and is deployed into a real ambient-assistive living environment. Note that in all cases we measured classification accuracy. Detailed results are depicted in Tables 1 and 2 for 11 and 51 classes, respectively. As it may be observed, in the first case, DST showed best accuracy for the majority of single- and cross-view experiments, followed by DCT. In cross-view experiments, while in the cross-subject case, DFT showed best accuracy, once again followed

**Table 1.** Experimental results denoting accuracy of the proposed approach in the 11 selected classes of the PKU-MMD dataset. M, L and R denote the middle, left and right camera angles, respectively.

Experiment	Train	Test	DFT	FFT	DCT	DST
Single-view	M	M	<b>0.89</b>	0.84	0.83	0.86
	L	L	0.76	0.82	0.78	<b>0.84</b>
	R	R	0.84	0.85	<b>0.89</b>	0.87
Cross-view	M	L	0.62	0.61	0.63	<b>0.64</b>
	M	R	0.58	0.61	<b>0.65</b>	0.63
	L	M	0.65	0.66	0.64	<b>0.72</b>
	L	R	0.41	0.38	0.32	<b>0.43</b>
	R	M	0.56	0.59	<b>0.64</b>	0.63
	R	L	0.32	0.37	0.33	<b>0.39</b>
	M, L	R	0.60	0.60	<b>0.62</b>	<b>0.62</b>
	M, R	L	<b>0.60</b>	0.57	0.59	<b>0.60</b>
	L, R	M	0.77	0.77	0.81	<b>0.82</b>
Cross-subject	M, L, R	M, L, R	<b>0.85</b>	0.79	0.83	0.81

**Table 2.** Experimental results denoting accuracy of the proposed approach in the 51 classes of the PKU-MMD dataset. M, L and R denote the middle, left and right camera angles, respectively.

Experiment	Train	Test	DFT	FFT	DCT	DST
Single-view	M	M	0.46	0.49	<b>0.66</b>	0.65
	L	L	0.46	0.52	0.60	<b>0.65</b>
	R	R	0.55	0.51	<b>0.69</b>	0.67
Cross-view	M	L	0.33	0.34	0.33	<b>0.36</b>
	M	R	0.32	0.31	<b>0.36</b>	0.32
	L	M	0.33	0.30	0.35	<b>0.39</b>
	L	R	<b>0.22</b>	<b>0.22</b>	0.13	0.14
	R	M	0.32	0.34	<b>0.36</b>	0.34
	R	L	<b>0.20</b>	0.19	0.13	0.14
	M, L	R	0.33	<b>0.34</b>	0.33	<b>0.34</b>
	M, R	L	0.32	0.33	0.32	<b>0.34</b>
	L, R	M	0.44	0.44	<b>0.55</b>	0.52
Cross-subject	M, L, R	M, L, R	0.50	0.52	<b>0.64</b>	0.63

by DCT. In the second case, DCT and DST showed best accuracy in the majority of cases, apart from the extreme cross-view cases where left angle was used for training and right for testing or vice versa, where DFT showed best accuracy, followed by FFT.

## 5 Conclusions and Future Work

In this paper we presented a novel approach which aims to recognize human actions in videos. Our approach is based on a novel representation of skeletal 3D motion which uses spectral image transformations and also on a novel CNN architecture. More specifically, a CNN was trained on images which resulted upon (a) concatenation of raw 1D signals corresponding to 3D motion of skeletal joints' coefficients and (b) application of a transform to the created image.

We evaluated the proposed approach using a state-of-the-art and challenging dataset, which consisted of sequences corresponding to 51 human actions. These sequences had been captured with 3 Kinect v2 cameras, under different camera angles and the skeletal joints of the human actors involved had been extracted. We performed experiments involving either a single camera (single-view) or more than one (cross-view). We also performed cross-subject experiments to evaluate the robustness of the approach. We mainly focused on a subset of 11 actions which in our opinion are the most close to real-life ADLs. However, we also experimented with the whole dataset. Our initial results indicate that the proposed approach may be successfully applied to human action recognition in

real-like conditions, yet a drop of performance is expected when significant changes of viewpoint occur.

Among our plans for future are the following: (a) investigation on methods for creating the signal image, possibly with the use of other types of sensor measurements such as wearable accelerometers, gyroscopes etc.; (b) investigation on image processing methods for transforming the signal image to the activity image; (c) use of simplified activity images by considering symmetries, e.g., in DFT and FFT (d) exploitation of other types of visual modalities in the process, such as RGB and depth data; (e) evaluation of the proposed approach on several other public datasets; and (f) application into a real-like or even real-life assistive living environment.

**Acknowledgment.** We acknowledge support of this work by the project SYNTELESIS “Innovative Technologies and Applications based on the Internet of Things (IoT) and the Cloud Computing” (MIS 5002521) which is implemented under the “Action for the Strategic Development on the Research and Technological Sector”, funded by the Operational Programme “Competitiveness, Entrepreneurship and Innovation” (NSRF 2014–2020) and co-financed by Greece and the European Union (European Regional Development Fund).

## References

1. Abadi, M., et al.: TensorFlow: a system for large-scale machine learning. In: Proceedings of the USENIX Symposium on Operating Systems Design and Implementation (OSDI) (2016)
2. Berretti, S., Daoudi, M., Turaga, P., Basu, A.: Representation, analysis, and recognition of 3D humans: a survey. ACM Trans. Multimed. Comput. Commun. Appl. (TOMM) **14**(1S), 16 (2018)
3. Chollet, F.: Keras (2015). <https://github.com/fchollet/keras>
4. Du, Y., Fu, Y., Wang, L.: Skeleton based action recognition with convolutional neural network. In: 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), pp. 579–583. IEEE (2015)
5. Graves, A., Mohamed, A.R., Hinton, G.: Speech recognition with deep recurrent neural networks. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 6645–6649. IEEE (2013)
6. Hou, Y., Li, Z., Wang, P., Li, W.: Skeleton optical spectra-based action recognition using convolutional neural networks. IEEE Trans. Circuits Syst. Video Technol. **28**(3), 807–811 (2018)
7. Jiang, W., Yin, Z.: Human activity recognition using wearable sensors by deep convolutional neural networks. In: Proceedings of the 23rd ACM International Conference on Multimedia, pp. 1307–1310 (2015)
8. Ke, Q., An, S., Bennamoun, M., Sohel, F., Boussaid, F.: SkeletonNet: mining deep part features for 3-D action recognition. IEEE Signal Process. Lett. **24**(6), 731–735 (2017)
9. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)

10. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: HMDB: a large video database for human motion recognition. In: 2011 International Conference on Computer Vision, pp. 2556–2563. IEEE (2011)
11. Laptev, I., Marszalek, M., Schmid, C., Rozenfeld, B.: Learning realistic human actions from movies. In: 2008 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8. IEEE (2008)
12. Lawton, M.P., Brody, E.M.: Assessment of older people: self-maintaining and instrumental activities of daily living. *Gerontol.* **9**(3 Part 1), 179–186 (1969)
13. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**(11), 2278–2324 (1998)
14. Li, C., Hou, Y., Wang, P., Li, W.: Joint distance maps based action recognition with convolutional neural networks. *IEEE Signal Process. Lett.* **24**(5), 624–628 (2017)
15. Liu, C., Hu, Y., Li, Y., Song, S., Liu, J.: PKU-MMD: a large scale benchmark for continuous multi-modal human action understanding. arXiv preprint [arXiv:1703.07475](https://arxiv.org/abs/1703.07475) (2017)
16. Liu, M., Liu, H., Chen, C.: Enhanced skeleton visualization for view invariant human action recognition. *Pattern Recognit.* **68**, 346–362 (2017)
17. Mathe, E., Mitsou, A., Spyrou, E., Mylonas, Ph.: Arm gesture recognition using a convolutional neural network. In: Proceedings of International Workshop Semantic and Social Media Adaptation and Personalization (SMAP) (2018)
18. Mathe, E., Maniatis, A., Spyrou, E., Mylonas, Ph.: A deep learning approach for human action recognition using skeletal information. In: Proceedings of World Congress “Genetics, Geriatrics and Neurodegenerative Diseases Research” (GeNeDiS) (2018)
19. Schuldt, C., Laptev, I., Caputo, B.: Recognizing human actions: a local SVM approach. In: Proceedings of the 17th International Conference on Pattern Recognition (ICPR 2004), vol. 03, pp. 32–36. IEEE Computer Society (2004)
20. Shahroud, A., Liu, J., Ng, T.T., Wang, G.: NTU RGB+D: a large scale dataset for 3D human activity analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1010–1019 (2016)
21. Soomro, K., Zamir, A.R., Shah, M.: UCF101: a dataset of 101 human actions classes from videos in the wild. arXiv preprint [arXiv:1212.0402](https://arxiv.org/abs/1212.0402) (2012)
22. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**(1), 1929–1958 (2014)
23. Wang, P., Li, W., Ogunbona, P., Wan, J., Escalera, S.: RGB-D-based human motion recognition with deep learning: a survey. *Comput. Vis. Image Underst.* **171**, 118–139 (2018)
24. Wang, P., Li, W., Li, C., Hou, Y.: Action recognition based on joint trajectory maps with convolutional neural networks. *Knowl.-Based Syst.* **158**, 43–53 (2018)



# Staircase Detection Using a Lightweight Look-Behind Fully Convolutional Neural Network

Dimitrios E. Diamantis<sup>(✉)</sup>, Dimitra-Christina C. Koutsiou,  
and Dimitris K. Iakovidis

Department of Computer Science and Biomedical Informatics,  
University of Thessaly, Lamia, Greece  
`{didiamantis, dkoutsiou, diakovidis}@uth.gr`

**Abstract.** Staircase detection in natural images has several applications in the context of robotics and visually impaired navigation. Previous works are mainly based on handcrafted feature extraction and supervised learning using fully annotated images. In this work we address the problem of staircase detection in weakly labeled natural images, using a novel Fully Convolutional neural Network (FCN), named LB-FCN *light*. The proposed network is an enhanced version of our recent Look-Behind FCN (LB-FCN), suitable for deployment on mobile and embedded devices. Its architecture features multi-scale feature extraction, depthwise separable convolutions and residual learning. To evaluate its computational and classification performance, we have created a weakly-labeled benchmark dataset from publicly available images. The results from the experimental evaluation of LB-FCN *light* indicate its advantageous performance over the relevant state-of-the-art architectures.

## 1 Introduction

Staircases can be found almost everywhere in different colors, shapes and sizes in both indoor and outdoor environments. Staircases are useful in everyday life; however, they can be seen also as an obstacle for the navigation of humans with disabilities, as well as the navigation of artificial, robotic, agents. The detection of a staircase can be even more difficult in unknown environments, especially for the visually impaired, where there is no previous knowledge about the surroundings, and they can become hazardous. Therefore, staircase detection can be considered as an important component of any system aiming to provide navigational assistance in either indoor or outdoor environments. In controlled, indoor environments, markers, such as augmented reality markers can be used to provide high success rate of staircase detection [1]. The detection problem usually becomes much harder in outdoor, uncontrolled environments, where different types of staircases of various sizes can be found under various illumination conditions, and can be observed from different viewpoints.

In this paper we address image-based staircase detection as a pattern recognition problem in the context of embedded and mobile devices. The main challenge is to be able to provide sufficient detection accuracy by utilizing the limited computational

resources of such devices, especially in outdoor environments with low latency and limited network accessibility. To address this challenge, we propose a novel lightweight Fully Convolutional neural Network (FCN) architecture as a modification of our recent Look-Behind FCN (LB-FCN) architecture [2]. This novel architecture, named LB-FCN *light*, has significantly fewer free parameters and requires fewer Floating Point Operations (FLOPs) compared to the previous LB-FCN and state-of-the-art architectures for mobile devices. This was achieved by implementing depthwise separable convolutions throughout the convolutional layers of the network. Also, it enables multi-scale feature extraction and residual learning, making it suitable for multi-scale staircase detection in both indoor and outdoor environments. To evaluate the performance of LB-FCN *light* we created a weakly labeled image dataset, with staircases found in natural images collected from publicly available datasets, i.e., a dataset with semantically labeled images as containing or not containing staircases.

The rest of the paper consists of four sections. In Sect. 2 the related work focusing on staircase detection is presented. In Sect. 3 we describe the proposed architecture and its advantages. In Sect. 4 we describe our weakly annotated staircase dataset, and the results of the experiments performed. The last section summarizes the conclusions that can be derived from this study along with our plans for future work.

## 2 Related Work

Staircase detection has been an active research topic in computer vision and robotics, with an increasing interest nowadays as we are going through the era of ubiquitous computing and pervasive intelligence. One of the first relevant works [3] was based on Gabor filters and concurrent line grouping for distant and close staircase detection respectively. In the context of autonomous vehicle navigation, an outdoor descending staircase detection algorithm was presented by [4], based on texture energy, optical flow, and scene geometry features. In the context of computer aided navigation of visually impaired in outdoor environments using a wearable stereo camera, [5] utilized Haar features and Adaboost learning providing real-time detection performance. A similar approach that utilizes Haar-like features and an improved staircase specific Viola-Jones detector was proposed in [6].

Frequency domain features obtained by ultrasonic sensors were investigated in [7], to detect and recognize floor and staircases in electronic white cane. A wearable RGB-D camera mounted on the chest of a visually impaired individual, was used in [8], where an indoor environment for staircase detection and modeling was proposed. Their approach is capable of providing information for the presence and location along with the number of steps of staircases. Recently an indoor staircase detection framework was proposed in [9], utilizing depth images, capable of running on mobile devices. The approach is based on the detection and clustering of image patches that have the surface vectors pointing to the top direction. In addition, information from the Inertial Measurement Unit (IMU) sensor of the device is used to calibrate the surface vectors with the camera orientation. Most of the current staircase detection approaches are supervised, requiring fully annotated training images from controlled environments, i.e., images indicating the location of the staircases within the images. Furthermore, to the

best of our knowledge the staircase detection has not been previously investigated to a sufficiently generic extent.

Although deep learning and more specifically Convolutional Neural Networks (CNNs) [10] have demonstrated impressive performance in computer vision applications, especially in natural image classification [11], staircase detection approaches have not been previously reported. While they are effective, conventional deep CNNs such as [12], suffer from high computational complexity mainly due to their large number of free parameters. As a result, high-end computational equipment such as Graphical Processing Units (GPUs) is needed for both training and testing time, limiting their use in indoor workstations. Recent studies such as [13–15] focus their interest in computational complexity reduction of CNN architectures, aiming to enable their usage in mobile and embedded devices. In this context, the tradeoff between computational efficiency and detection performance has been investigated, resulting in a state-of-the-art architecture called MobileNet-v2 [16], extending the original MobileNet-v1 proposed in [14]. More specifically this architecture keeps the basic principles of depthwise convolutions for the original design enhances it by adding linear bottleneck layers and shortcut connections between each bottleneck. Linear bottleneck layers were utilized as experimental evidence that the non-linear ones were damaging the extracted features between the bottlenecks. As a result of these changes the architecture contains 30% less parameters than MobileNet-v1 while providing a higher accuracy. Recently, we presented LB-FCN [2] architecture in the context of abnormality detection in medical images. The architecture featured multi-scale feature extraction modules composed of conventional convolutional layers, to better represent the different scales of abnormalities. In addition look-behind connections were used, which connect the input features to the output of each multi-scale feature extraction module. This was required, so that the high-level features will propagate throughout the network, allowing the network to converge faster and increasing the overall detection accuracy.

The core of LB-FCN *light* architecture is inspired by LB-FCN [2] and includes modification to enable efficient computations on mobile and embedded devices, while providing a sufficient staircase detection accuracy. More specifically, LB-FCN *light* extends the original LB-FCN design by replacing the multi-scale conventional convolutional layers with depthwise convolutional layers [17]. Key features of this architecture include the utilization of multi-scale depthwise separable convolution layers [17] and residual learning [18] connections which help to maintain relatively low number of free parameters, without sacrificing the detection accuracy.

### 3 Architecture

The design of the LB-FCN *light* architecture follows the FCN [19] network design, where only convolutional layers are utilized throughout the network. By replacing the fully connected layers, usually found in the classification layer of conventional CNN architectures such as [11, 12], a significant reduction of the number free parameters of the architecture can be achieved. Inspired by the MobileNet architecture, proposed in [14], depthwise separable convolutions [17] are implemented throughout the network

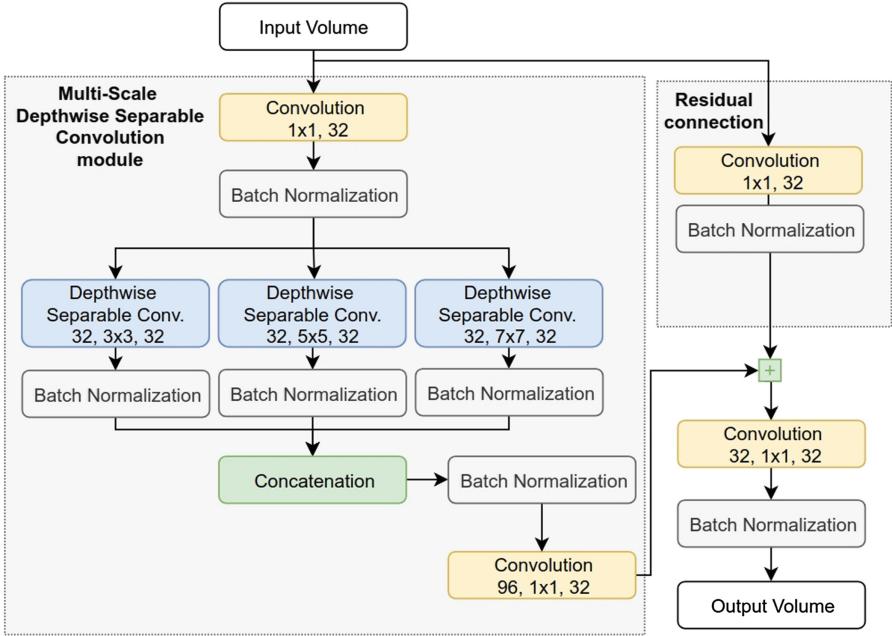
to further reduce the complexity of the overall architecture. While in conventional convolution the filters are connected on the entire depth of the input channels, in depthwise separable convolution the filter is applied separately on each channel. To connect the separate filters, the layers are followed by a  $1 \times 1$  conventional convolution.

The main component of LB-FCN *light* is the Multi-Scale Depthwise Convolution module (Fig. 1) which follows the principles established in [2]. This module is capable of extracting features from parallel depthwise separable convolution layers, each one with a different filter size. More specifically the layers extract features at three different scales:  $3 \times 3$ ,  $5 \times 5$  and  $7 \times 7$  respectively. The feature maps from each layer are then concatenated forming a multi-scale feature representation of the input which is then followed by  $1 \times 1$  convolution layer. The architecture features residual connections, which connect the input volume of the multi-scale module using adding operator aggregation with the output of it. This is done in order to preserve the higher level features extracted from the previous multi-scale blocks throughout the network.

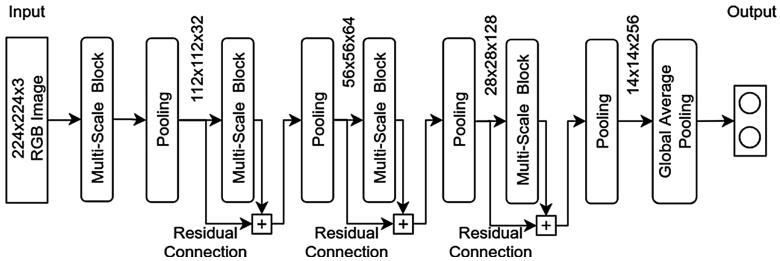
Following the FCN [19] approach which shows that conventional max pooling operation can be replaced with a convolutional based, we utilized convolutional pooling with filter size  $3 \times 3$  and stride 2. This introduces another level of non-linearity to the network while keeping the overall architecture logically unified. After each pooling operation the number of extracted filters of each convolutional layer is doubled. In total four multi-scale depthwise convolution modules are utilized in the network with three residual connections as illustrated in Fig. 2. For the staircase detection, a softmax layer of two neurons is used as the output of the network.

Throughout the architecture all convolution layers use ReLU activations followed by output batch normalization. The normalization is used so that the output of the convolution layers are centered on zero mean with the unit standard deviation. It has been empirically confirmed that output normalization can contribute in a faster network converge while reducing overfitting phenomenon. As a result of the above no Dropout layer [20] was used.

While we maintained the multi-scale feature extraction characteristics established in the original LB-FCN [2] architecture, the change in original filter size selection block increased the overall accuracy of the network. Furthermore we utilized conventional ReLU activation functions throughout the network instead of Parametric ReLU that were used in original LB- FCN architecture, which resulted in lower computational complexity without any significant detection performance overhead. The overall improvements made in original LB-FCN architecture, resulted in a significant increase in computational efficiency. As a result, LB-FCN light architecture is capable to efficiently run on mobile and embedded devices.



**Fig. 1.** The main building block of LB-FCN *light* architecture.



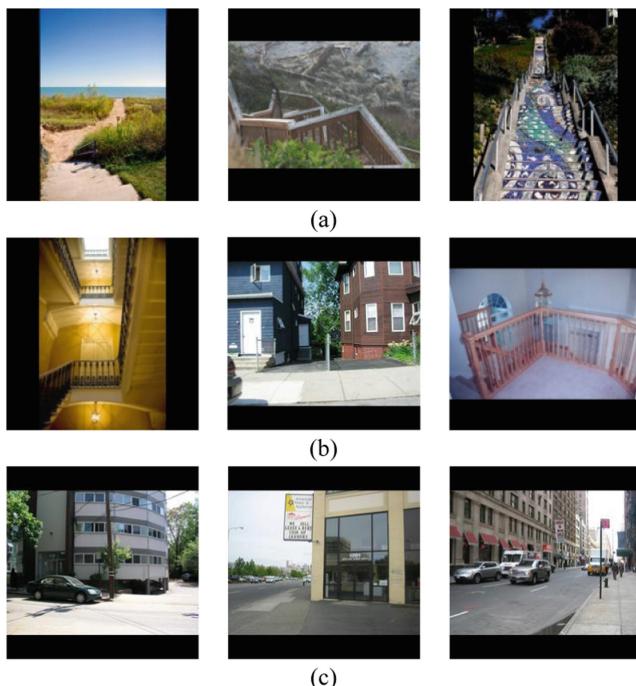
**Fig. 2.** The complete LB-FCN *light* architecture composed of four multi-scale blocks and three residual connections.

## 4 Experiments and Evaluation

### 4.1 Dataset

To evaluate the performance of the proposed architecture in the context of natural image staircase detection we have considered two publicly available datasets. The first dataset, named LM+Sun [21], is a fully annotated natural image dataset obtained from the combination of LabelMe Database [22] and SUN dataset [23]. The dataset consists of 45,676 images from 232 categories, found in indoor and outdoor environment under various conditions and sizes. For the purpose of our experiment we utilized a subset of

LM+Sun dataset which includes natural images found in urban and street areas. While the full LM+Sun dataset contains 314 staircase labeled images, most of them are found in indoor environments. Images containing staircases were also found in the urban and street subsets of this dataset, e.g., staircases of buildings that can be directly recognized by a human observer, considering: (a) staircases that have at least two steps, and (b) staircases covering >15% of the image (in staircases of smaller coverage the steps are not distinguishable; therefore, they cannot be perceived directly as such, without contextual information). To minimize the possibility of a human error in the annotation process, two reviewers separately reviewed and annotated the dataset, and found in total 245 images that include outdoor staircases. To further increase the number of outdoor staircase images, we have created a second dataset named “StairFlickr” which extends LM+Sun staircases with a total of 524 outdoor staircase images. StairFlickr dataset images were obtained from the popular photo management and sharing web application Flickr [24].



**Fig. 3.** Top: staircases found in StairFlickr dataset. Middle: staircases found in LM+Sun dataset. Bottom: non-staircases images from LM+Sun dataset.

For the purposes of our research, we omitted the fully annotated metadata provided about the staircases in the original LM+Sun dataset. This was performed as our architecture aims for staircase detection on solely weakly-labeled natural images. In total the described dataset includes 5,539 images from which 1,083 images contain

staircases<sup>1</sup>. Indicative images from this dataset are illustrated in Fig. 3. As it can be observed, the dataset includes various types of staircases found in various positions, sizes, capture from different viewpoints.

## 4.2 Evaluation Methodology

To evaluate the detection performance of the proposed architecture we followed the stratified 10-fold cross-validation (CV) procedure. The dataset was partitioned into 10 stratified subsets from which 9 were used for training and 1 for testing. This was repeated 10 times, each time selecting a different subset, until all folds have been tested. For each evaluation we calculated the accuracy (ACC), specificity (SPC), and sensitivity (TPR) of the trained model following the Eqs. (1–3), where true positives are denoted as TP, true negatives as TN, false positives as FP and false negatives as FN.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

$$SPC = \frac{TN}{TP + FP} \quad (2)$$

$$TPR = \frac{TP}{TP + FN} \quad (3)$$

$$FPR = 1 - SPC \quad (4)$$

To better evaluate the classification performance of the trained network, we utilized the Area Under ROC (AUC) measure. AUC measure is a reliable classification performance measure that is insensitive to imbalanced class distributions [25]. This was chosen as the total number of images containing staircases was significantly fewer than the rest of the rest natural images in the dataset.

## 4.3 Results

We trained the LB-FCN *light* architecture using the images from both Flickr and LM+Sun datasets. As the images differ from each other in both size and aspect ratio we rescaled the dataset to the standardized input size of the network which is  $224 \times 224$  pixels. To maintain the original aspect ratio of the images, they were padded with zeros to match the network’s input dimensions. It is worth mentioning that no further pre-processing step was applied to the images. As the proposed architecture focuses on weakly labeled images, the detailed annotations for the staircases provided by LM+Sun [21] dataset were ignored. We utilized only the semantic annotations of the images which indicate the presence or absence of staircases.

For the training of the network we utilized the Adam [26] optimizer with initial learning rate alpha = 0.001 and first and second moment estimates exponential decay

---

<sup>1</sup> A link to the dataset will be provided in the final manuscript.

rate  $\text{beta1} = 0.9$  and  $\text{beta2} = 0.999$  respectively. For the implementation of the architecture we utilized the Python Keras [27] library and the Tensorflow [28] tensor graph framework. The network was trained with mini-batch size of 32 samples on NVIDIA TITAN X GPU, equipped with 3584 CUDA [29] cores, 12 GB of RAM and base clock speed of 1417 MHz. On each fold we utilized the early-stopping technique where a small subset of the training fold was utilized as a validation dataset.

To evaluate the effectiveness in both detection accuracy and computational complexity reduction of LB-FCN *light* architecture we used the MobileNet-v2 [16] as a state-of-the-art architecture for comparison. The results obtained by the two architectures are illustrated in Table 1. The confusion matrix of LB-FCN *light* classification performance is illustrated in Table 3.

While the detection performance is slightly higher in case on LB-FCN *light*, the noticeable difference between the two architectures is the computational complexity requirements. Table 2 includes a comparison between the architectures in terms of both the number of trainable free parameters and the total number of required FLOPs. The improvements made on the original LB-FCN design, resulted in a significant reduction of the overall number of FLOPs, from  $1.3 \times 10^7$  down to  $0.6 \times 10^6$ , and reduction of the free parameters of the network, from  $8.2 \times 10^6$  down to  $0.3 \times 10^6$  respectively.

**Table 1.** Detection performance comparison, using 10-fold cross-validation, between state-of-the-art MobileNet-v2 [16] and our LB-FCN light architecture

Architecture	AUC (%)	Accuracy (%)	Specificity (%)	Sensitivity (%)
LB-FCN <i>light</i>	<b><math>88.93 \pm 1.86</math></b>	<b><math>91.89 \pm 2.12</math></b>	<b><math>93.80 \pm 2.61</math></b>	<b><math>84.05 \pm 3.51</math></b>
MobileNet-v2 [16]	$87.86 \pm 2.11$	$89.99 \pm 2.37$	$93.58 \pm 2.45$	$83.78 \pm 3.22$

**Table 2.** Computation complexity comparison between state-of-the-art MobileNet-v2 [16] and our LB-FCN *light* architecture

Architecture	FLOPs ( $\times 10^6$ )	Trainable free parameters ( $\times 10^6$ )
LB-FCN <i>light</i>	<b>0.6</b>	<b>0.3</b>
MobileNet-v2 [16]	4.7	2.2

**Table 3.** Confusion matrix of LB-FCN *light* classification performance.

	Staircases <i>actual</i>	Non-Staircases <i>actual</i>
Staircases <i>predicted</i>	910	276
Non-Staircases <i>predicted</i>	173	4180

## 5 Conclusions

We proposed a novel lightweight multi-scale FCN architecture that copes with the problem of staircase detection in natural images. To evaluate the performance of the architecture we extended the LM+Sun [21] natural image dataset with staircase images

obtained from Flickr [24] social network. To the best of our knowledge there has been no existing work in this field that utilize solely weakly-labeled images to detect staircases in the natural images. The key features of the proposed LB-FCN *light* architecture can be summarized as follows:

- It has a relatively low number of free parameters requiring an also low number of FLOPs, which makes it suitable to be used on mobile and embedded devices;
- It features multi-scale feature extraction design allowing the architecture to detect staircases of various sizes and under difficult conditions, such as natural images;
- Following the FCN [12] architecture approach it offers a lightweight and logically unified design;
- Compared to MobileNet-v2 [16] network, the proposed architecture offers a relatively lower number of FLOPs and free parameters and a slightly higher detection performance. This makes it attractive for lower-end mobile and embedded devices.

In our future work we are planning to evaluate the performance of the proposed architecture in larger weakly-labeled staircase natural image datasets, to further explore the potential of the architecture. Furthermore we plan to extend the purpose of LB-FCN *light* architecture to include the localization of the staircases within the images, by following a weakly supervised approach.

**Acknowledgments.** This research has been co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE – INNOVATE (project code:T1EDK-02070). It was also supported by the Onassis Foundation - Scholarship ID: G ZO 004-1/2018-2019. The Titan X used for this research was donated by the NVIDIA Corporation.

## References

1. Yu, X., Yang, G., Jones, S., Saniie, J.: AR marker aided obstacle localization system for assisting visually impairedn. In: IEEE International Conference on Electro/Information Technology (EIT), pp. 271–276 (2018)
2. Diamantis, D.E., Iakovidis, D.K., Koulaouzidis, A.: Look-behind fully convolutional neural network for computer-aided endoscopy. Biomed. Signal Process. Control **49**, 192–201 (2019)
3. Se, S., Brady, M.: Vision-based detection of staircases. In: Fourth Asian Conference on Computer Vision ACCV, vol. 1, pp. 535–540 (2000)
4. Hesch, J.A., Mariottini, G.L., Roumeliotis, S.I.: Descending-stair detection, approach, and traversal with an autonomous tracked vehicle. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 5525–5531 (2010)
5. Lee, Y.H., Leung, T.-S., Medioni, G.: Real-time staircase detection from a wearable stereo system. In: 2012 21st International Conference on Pattern Recognition (ICPR), pp. 3770–3773 (2012)
6. Maohai, L., Han, W., Lining, S., Zesu, C.: A robust vision-based method for staircase detection and localization. Cogn. Process. **15**(2), 173–194 (2014)

7. Bouhamed, S.A., Kallel, I.K., Masmoudi, D.S.: Stair case detection and recognition using ultrasonic signal. In: 2013 36th International Conference on Telecommunications and Signal Processing (TSP), pp. 672–676 (2013)
8. Pérez-Yus, A., López-Nicolás, G., Guerrero, J.J.: Detection and modelling of staircases using a wearable depth sensor. In: Agapito, L., Bronstein, Michael M., Rother, C. (eds.) ECCV 2014. LNCS, vol. 8927, pp. 449–463. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-16199-0\\_32](https://doi.org/10.1007/978-3-319-16199-0_32)
9. Ciobanu, A., Morar, A., Moldoveanu, F., Petrescu, L., Ferche, O., Moldoveanu, A.: Real-time indoor staircase detection on mobile devices. In: 2017 21st International Conference on Control Systems and Computer Science (CSCS), pp. 287–293 (2017)
10. LeCun, Y., et al.: LeNet-5, convolutional neural networks, p. 20 (2015). <http://yann.lecun.com/exdb/lenet>
11. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
12. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition, arXiv preprint [arXiv:1409.1556](https://arxiv.org/abs/1409.1556) (2014)
13. Iandola, F.N., Han, S., Moskewicz, M.W., Ashraf, K., Dally, W.J., Keutzer, K.: SqueezeNet: Alexnet-level accuracy with 50x fewer parameters and < 0.5 mb model size, arXiv preprint [arXiv:1602.07360](https://arxiv.org/abs/1602.07360) (2016)
14. Howard, A.G., et al.: Mobilenets: efficient convolutional neural networks for mobile vision applications, arXiv preprint [arXiv:1704.04861](https://arxiv.org/abs/1704.04861) (2017)
15. Zhang, X., Zhou, X., Lin, M., Sun, J.: ShuffleNet: an extremely efficient convolutional neural network for mobile devices. ArXiv e-prints, July 2017
16. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.-C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4510–4520 (2018)
17. Chollet, F.: Xception: Deep learning with depthwise separable convolutions, arXiv preprint, pp. 1610–2357 (2017)
18. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
19. Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.: Striving for simplicity: The all convolutional net. arXiv preprint [arXiv:1412.6806](https://arxiv.org/abs/1412.6806) (2014)
20. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
21. Tighe, J., Lazebnik, S.: SuperParsing: scalable nonparametric image parsing with superpixels. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6315, pp. 352–365. Springer, Heidelberg (2010). [https://doi.org/10.1007/978-3-642-15555-0\\_26](https://doi.org/10.1007/978-3-642-15555-0_26)
22. Russell, B.C., Torralba, A., Murphy, K.P., Freeman, W.T.: LabelMe: a database and web-based tool for image annotation. Int. J. Comput. Vision **77**(1–3), 157–173 (2008)
23. Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., Torralba, A.: Sun database: Large-scale scene recognition from abbey to zoo. In: IEEE Conference on 2010 Computer Vision and Pattern Recognition (CVPR), pp. 3485–3492 (2010)
24. Flickr Inc., “Find your inspiration. | Flickr.” 2019
25. Fawcett, T.: An introduction to ROC analysis. Pattern Recogn. Lett. **27**(8), 861–874 (2006)
26. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)

27. Chollet, F.: Keras. (2015)
28. Abadi, M., et al.: Tensorflow: a system for large-scale machine learning. In: OSDI, vol. 16, pp. 265–283 (2016)
29. Sanders, J., Kandrot, E.: CUDA by example: an introduction to general-purpose GPU programming. Addison-Wesley Professional, (2010)



# Obstacle Detection Based on Generative Adversarial Networks and Fuzzy Sets for Computer-Assisted Navigation

George Dimas<sup>✉</sup>, Charis Ntakolia, and Dimitris K. Iakovidis

Department of Computer Science and Biomedical Informatics,  
University of Thessaly, Lamia, Greece  
`{gdimas, cntakolia, diakovidis}@uth.gr`

**Abstract.** Obstacle detection addresses the detection of an object, of any kind, that interferes with the canonical trajectory of a subject, such as a human or an autonomous robotic agent. Prompt obstacle detection can become critical for the safety of visually impaired individuals (VII). In this context, we propose a novel methodology for obstacle detection, which is based on a Generative Adversarial Network (GAN) model, trained with human eye fixations to predict saliency, and the depth information provided by an RGB-D sensor. A method based on fuzzy sets are used to translate the 3D spatial information into linguistic values easily comprehensible by VII. Fuzzy operators are applied to fuse the spatial information with the saliency information for the purpose of detecting and determining if an object may interfere with the safe navigation of the VII. For the evaluation of our method we captured outdoor video sequences of 10,170 frames in total, with obstacles including rocks, trees and pedestrians. The results showed that the use of fuzzy representations results in enhanced obstacle detection accuracy, reaching 88.1%.

**Keywords:** Visually impaired · Generative Adversarial Networks · Fuzzy sets

## 1 Introduction

When visually impaired individuals (VII) visit unfamiliar environments, it is of major importance to feel safe and confident in moving in that environment. Obstacle detection and warning can play a crucial role in addressing the above challenge. An assistive technology for obstacle detection, that could interpret visual information into audible messages, could offer VII an additional, and more effective navigational aid, compared to the traditional ones, such as white cane or human guidance. This will provide them with the opportunity to have their hands free, so they can feel independent walking alone and even joining activities such as visiting museums and other indoor or outdoor spaces.

Today, the commercially available assistive technologies for navigation are mainly based on Global Positioning System (GPS). GPS assistive guidance approaches are unsuitable for VII since they lack of high accuracy in urban environments (the error can be estimated even to several meters); the signal can be easily lost during operation for

various reasons such as line-of-sight restrictions or multi-path effect. In addition, GPS needs sufficient number of directly visible satellites to work properly [1].

A factor that would make an assistive system for navigation suitable for VII is its ability to detect various types of objects in images by computer vision. The analysis of user requirements of the project ENORASI [2], showed that the users would like the system to detect in real-time mainly vertical objects such as trees, humans, stairs and terrain anomalies, while in parallel the system should be able to accurately and reliably detect the surrounding area for potential cultural sights. Computer vision methods for obstacle detection are also important in the robotic navigation domain [3], so any methodology being developed for a form of assistive/independent navigation can be applied for the navigation of VII to the robotic navigation domain and vice-versa.

Several studies investigating deep learning approaches in the literature review have addressed the issue of object detection in images. A Faster Region-Based Convolutional Neural Network (R-CNN) [4] for real-time object detection with region proposal networks was used to detect and track objects in [5]. Motion, sharpening and blurring filters were used to enhance feature representation. An approach enabling joint object detection, tracking and recognition was developed in the context of the DEEP-SEE framework, presented in [6]. An intelligent smart glass using deep learning machine vision techniques and Robot Operating System (ROS) was presented in [7], where three CNN architectures were used, namely Faster R-CNNs [4], You Only Look Once (YOLO) CNN [8] and Single Shot multibox Detectors (SSDs) [9]. However, the main purpose of these methods was to solely detect objects and not classify them as obstacles.

An obstacle detection module of a wearable mobility aid based on LeNet was proposed in [10], and a unified real-time object detection method based on a YOLO CNN was proposed in [8]. These machine learning-based methods consider obstacle detection as a 2D problem in the image plane. A few studies have considered obstacle detection for VII as a 3D problem, exploiting also depth information. Such information is usually derived from stereoscopic cameras or RGB-D sensors. In [11] a robust depth-aware obstacle detection system was presented for providing distance information for safety using a two-stage methodology: object segmentation and obstacle extraction. A multi-stage depth-aware random forest model using discriminative saliency fusion was developed for salient region detection [12]. For micro air vehicle flight applications, a multi-task deep architecture that jointly estimates depth and obstacles without computing a global map was proposed [13]. For traffic situations, a stereo and motion fusion approach using flow/depth constraint for obstacle detection was developed in the context of the Daimler-Chrysler urban traffic assistance project [14].

In this paper, we present a novel method for obstacle detection. It is based on a Generative Adversarial Network (GAN) for the detection of salient regions within images, and on fuzzy sets for combining the saliency information with the 3D spatial information acquired by an RGB-D sensor from the environment. Unlike previous approaches the proposed method enables the assessment of the degree to which an obstacle represents a threat to the user, and it provides an inherent way of naturally describing the current situation to the user using linguistic values. The use of fuzzy sets enables soft assessment of the bounds of a threat (e.g., there are overlapping distance intervals within which an obstacle can be considered both as a high or medium risk threats) which is proved to be more effective than conventional approaches, usually

employing hard bounds [15], e.g., an obstacle is considered as a threat of high risk after a specified distance. The human eye fixation map produced by the GAN, combined with the fuzzy interpretation of the depth values, produce a perceptually meaningful and interpretable information for the user. Also, compared with the machine learning-based obstacle detection methods that were mentioned above, our method does not need any training regarding the obstacle detection part. The only training that it takes place is that of the saliency prediction based on human eye fixation data.

The rest of this paper is consisting of 3 sections. Section 2 describes the proposed methodology and Sect. 3 describes the experiments and the obtained results. The last section summarizes the conclusions of our study.

## 2 Methodology

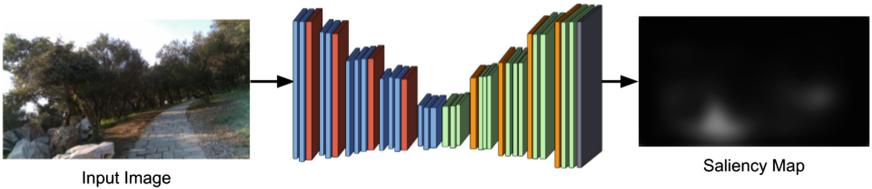
The proposed method consists of two components; the saliency map generation using a GAN trained on human eye fixations, called SalGAN [16] and a fuzzy set-based approach combining the 3D spatial information acquired by an RGB-D sensor to risk values for a possible obstacle threat.

### 2.1 Human Eye Fixation Saliency Maps

The saliency map generation is based on a GAN [17], which is a deep CNN (Fig. 1). The main motivation utilizing it in the saliency prediction, is that it produces saliency maps based on human eye fixations. Thus, an eye fixation-based salient map carries the information of what regions in an image would be interesting for a human. So forth, the obstacle detection process can become more intuitive, in the sense that the algorithm will be able to detect regions that humans consider as more salient. Furthermore, it can be trained with eye fixation data captured from individuals navigating through paths with obstacles, and hence extend its potentials.

The GAN architecture consists of two CNNs, a discriminator and a generator network. The combination of the two CNNs aims to predict visual saliency maps from an image. The generator produces the saliency maps in its output, and the discriminator optimizes its resemblance with ground truth saliency maps obtained from humans using eye tracking data [18]. The generator CNN has an encoder-decoder architecture. The encoder part has the same architecture as VGG-16 [19], including both convolutional and pooling layers (illustrated in grey and purple colour in Fig. 1). However, it does not have the final pooling and fully connected layers (FC) layers of VGG-16. The max-pooling layers of the encoder are used for downscaling of the feature maps. The encoder network weights are initialized with those of a VGG-16 model trained on the ImageNet dataset [20]. For the purpose of estimating saliency maps, only the last two groups of convolutional layers have been modified during training. The decoder part of the CNN has up-sampling instead of pooling layers followed by convolutional filters in order to increase the size of the output to match the size of the input image. The decoder architecture is identical to the encoder architecture but with the layers placed in reverse order. As an activation function, Rectified Linear Unit (ReLU) was used in all convolutional layers. To produce the saliency map, a final  $1 \times 1$  convolutional layer

was places with a sigmoid activation function. The weights of the decoder were randomly initialized. The discriminator CNN architecture consists of six  $3 \times 3$  convolution filters with 3 pooling layers and followed by 3 FC layers. The activation functions for the convolutional and FC layers are the ReLU and hyperbolic tangent function (tanh), respectively. Exception is the final layer, which uses the sigmoid activation function. Figure 1 illustrates the architecture of the saliency map generator. The convolutional layers of the encoder are depicted with a blue colour and the max-pooling layers with red; the convolutional layers of the decoder are depicted with green colour, the up-sampling layers with orange and the final convolutional layer with the sigmoid activation with grey colour. An example of a saliency map produced by the generative model, is illustrated in detail in Fig. 2.



**Fig. 1.** The SalGAN architecture for saliency map generation. (Color figure online)



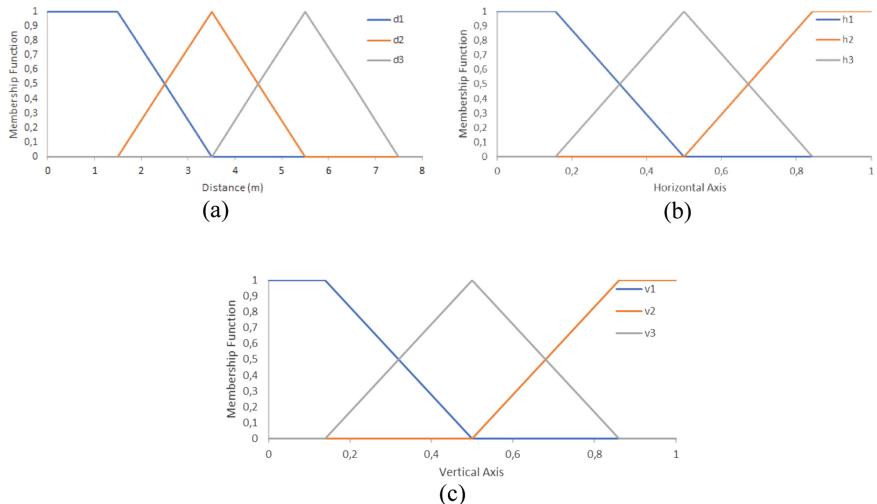
**Fig. 2.** Illustration of the generated saliency map from an input image. (a) Input image. (b) The generated saliency map.

## 2.2 Depth-Aware Obstacle Detection

The generated saliency map can be seen as a broad weighted region of interest in which an obstacle may reside, i.e., higher intensities of the saliency map correspond to higher likelihoods for possible objects of interest. In order to limit the search region for the detection of an obstacle, we exploit the values of a depth map produced by an RGB-D sensor. To assess the risk of a threatening obstacle for the VII, we employ a methodology based on fuzzy sets [21]. We consider a set of 3 fuzzy sets  $\Delta_1, \Delta_2, \Delta_3$ , which correspond to three different risk levels, expressible by linguistic values indicating high, medium and low risk respectively. Each fuzzy set represents a degree of risk that an object in a given depth value  $z$ , may impose a threat to the VII. The

universe of discourse for these fuzzy sets is the range of values of the depth maps produced by the RGB-D sensor. The fuzzy sets  $\Delta_1$  and  $\Delta_2$ , and the fuzzy sets  $\Delta_2$  and  $\Delta_3$ , are overlapping between each other, considering the uncertainty in the assessment of an obstacle threat as high or medium, and as medium or low respectively, upon its distance from the user. The respective membership functions  $d_i(z)$ ,  $i = 1, 2, 3$  are illustrated in Fig. 3(a), where  $z$  is a value of the estimated depth map.

For the spatial localization of an obstacle in the image plane, we constructed 6 additional fuzzy sets, namely  $H_1$ ,  $H_2$  and  $H_3$  for the horizontal axis, corresponding to the left, central and right part of the image, namely,  $V_1$ ,  $V_2$  and  $V_3$  for the vertical axis, corresponding to the up, central and bottom part of the image. The respective membership functions,  $h_i(x)$ , and  $v_i(y)$  are illustrated in Fig. 3(b), (c), where  $x \in [0, 1]$  and  $y \in [0, 1]$  are the horizontal and vertical coordinates within an image, normalized by image width and height, respectively.

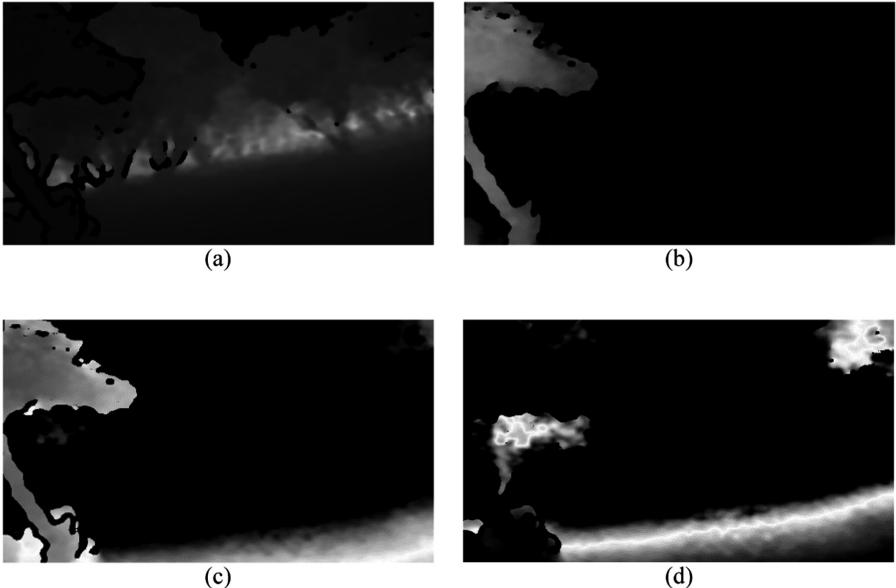


**Fig. 3.** Membership functions of fuzzy sets used for the localization of objects in the 3D space using linguistic variables. (a) Membership functions for low ( $d_1$ ), medium ( $d_2$ ), and high risk ( $d_3$ ) upon the distance of the user from an obstacle. (b) Membership functions for left ( $h_1$ ), central ( $h_2$ ) and right ( $h_3$ ) positions on the horizontal axis. (c) Membership functions for up ( $v_1$ ), central ( $v_2$ ) and bottom ( $v_3$ ) positions on the vertical axis.

Once we establish the membership functions for each fuzzy set, we create three different risk maps,  $\Delta_M^i$  based on the depth values and the membership functions of each fuzzy set  $\Delta_i$ . Each risk map consists of the responses of a membership function given a depth value  $z$  and it can be formally expressed as follows:

$$\Delta_M^i(x, y) = d_i(M_z(x, y)) \quad (1)$$

where  $M_z$  is the depth map of an RGB image,  $I_{RGB}$ . From Eq. (1) a total of 3 risk maps is derived, where each, represents regions of a degree of risk that an object may impose a threat to a person navigating within its range. A visual representation of the risk maps can be seen in Fig. 4. Figure 4(a) illustrates the depth map corresponding to Fig. 2(a), where the dark pixel values represent lower depth values (nearest distances) and the brighter pixel values represent higher depth values (further distances). Figures 4(b)-(d) are illustrations of the different risk maps produced by Eq. (1) using membership functions  $d_i(z)$ ,  $i = 1, 2, 3$ . The brighter pixel values of the risk maps indicate higher participation in  $\Delta_1$ ,  $\Delta_2$  and  $\Delta_3$  fuzzy sets respectively.



**Fig. 4.** A graphical representation of the risk maps. (a) Depth map  $M_z$  of the image in Fig. 2(a). (b) High-risk map  $\Delta_M^1$ . (c) Medium-risk map  $\Delta_M^2$ . (d) Low-risk map  $\Delta_M^3$ .

To further localize an obstacle imposing threat to a navigating VII, we combine the information from the saliency map  $S_M$  with a risk map  $\Delta_M^i$ , using the fuzzy AND operator ( $\wedge$ ) [21]:

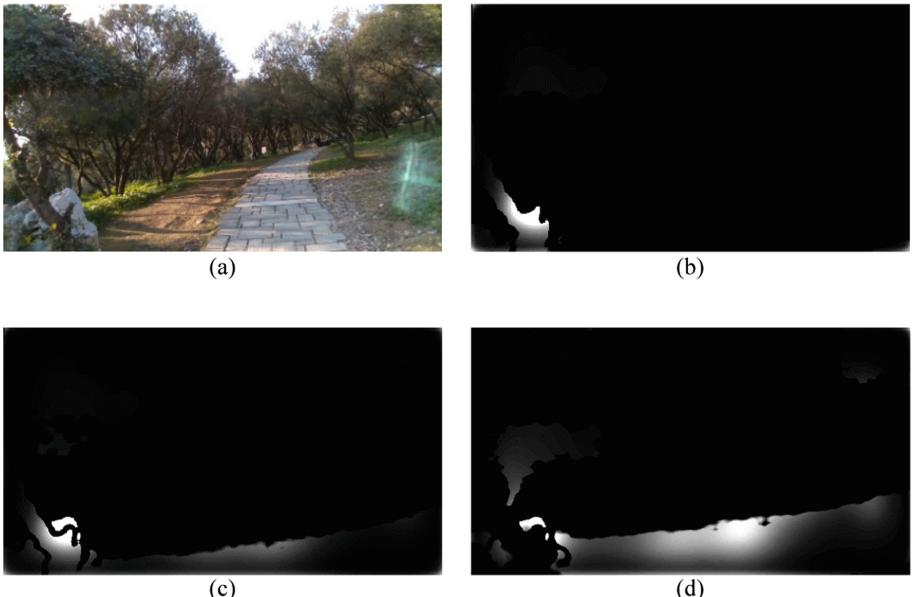
$$F_1 \wedge F_2 := \min(F_1(x, y), F_2(x, y)) \quad (2)$$

where  $F_1$ ,  $F_2$  are two 2D fuzzy maps, with values within  $[0, 1]$ , and  $x$ ,  $y$  represent the coordinates of each value within the 2D map. A risk map  $\Delta_M^i$  and a (normalized) saliency map  $S_M$ , can be considered as such fuzzy maps, since the risk maps are generated by the responses of fuzzy membership functions, and saliency maps indicate

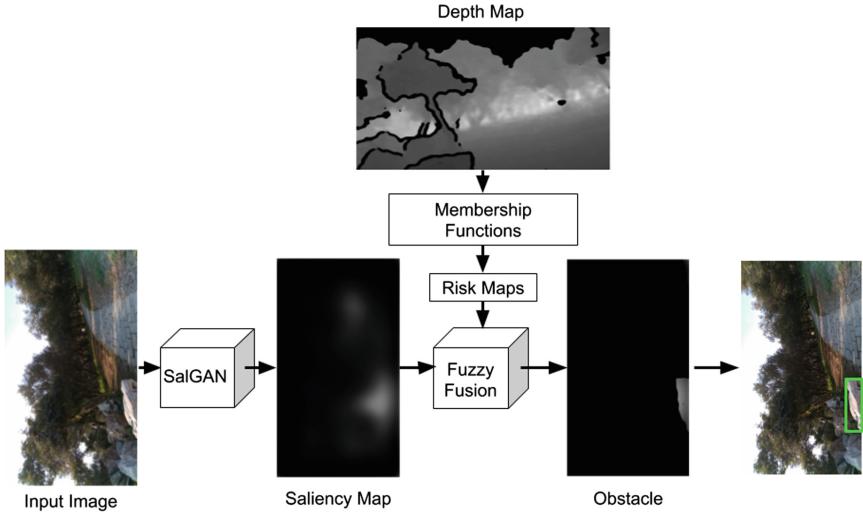
the degree to which a pixel belongs to a salient region. Thus, Eq. (3) gives a new image:

$$O_{obstacle}^i = S_M \wedge A_M^i \quad (3)$$

where non-zero pixel-values of  $O_{obstacle}^i$  represent the location of an obstacle and a degree of participation in a risk-level in the respective region. An example of the application of Eq. (3) for the combination of the saliency map of Fig. 2 with each of the different risk maps of Fig. 3 is illustrated in Fig. 5. In Fig. 5(b) the whitish area indicates that the respective object in image Fig. 5(a) (the tree on the left) is of high-risk so it is labeled as an obstacle to be avoided. The whitish area in Fig. 5(c) indicates the objects of medium-risk and the whitish area in Fig. 5(d) indicates that the respective objects are of low-risk. The fact that the lower part of the tree is highlighted in both Fig. 5(b) and (c) is due to the overlap of the respective membership functions. The overall pipeline of the proposed methodology is summarized in Fig. 6. The RGB image is used as input to SalGAN for the saliency map generation. The depth values  $z$  of the depth map are used as inputs to the membership functions for the risk maps generation as described by Eq. (1). Then, the risk maps are fused with the saliency map according to Eq. (3). In this example we use the  $A_M^1$  risk map to identify the obstacles that impose the highest threat to an individual that is within their range.



**Fig. 5.** A visual representation of the  $O_{obstacle}^i$ . (a) The original  $I_{RGB}$  image. (b) Image  $O_{obstacle}^1$ . (b) Image  $O_{obstacle}^2$ . (c) Image  $O_{obstacle}^3$ .



**Fig. 6.** A visual representation of the pipeline of the proposed methodology.

### 3 Experiments and Results

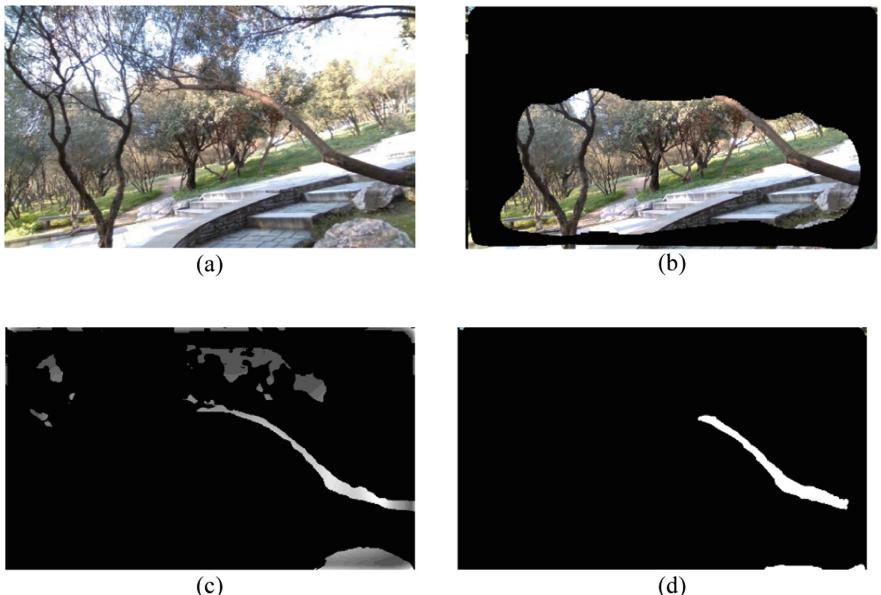
For the validation of our method, we captured 10,170 frames in total. The videos were captured using an RGB-D sensor, namely, the state-of-the-art Intel® RealSense™ D435. The size of this sensor is conveniently small ( $90 \times 25 \times 25$  mm) to be attached on a wearable system for the assistive navigation of the VII. It enables 3D depth sensing, with a maximum range of 10 m. The D435 sensor has two infrared (IR) cameras that enable stereoscopic vision, an IR projector and a high resolution RGB camera. The IR projector is used to improve the depth estimation. This is done by the stereoscopic system while projecting a static IR pattern on the scene. The IR pattern projection enables the texture enrichment of low texture scenes.



**Fig. 7.** Example detection of high risk obstacles (nearest crowd and tree branches). (a) Original image RGB image. (b) The corresponding image  $O_{obstacle}^1$  obtained by using membership function  $d_1$ .

The originally captured videos were uniformly subsampled, by acquiring a sample every 5 frames. After this process, the resulting dataset was composed of a total of 2034 frames, including 6 obstacle categories, namely trees or tree branches, ground anomalies, crowds and stones. The obstacles in the dataset were annotated by a human observer. For the obstacle detection, we used the high-risk map  $\Delta_M^1$  (Fig. 7).

The distance intervals characterizing the risk of an obstacle have been determined by the user requirements [2], and the form of the respective membership functions have been empirically determined. The interval of the high-risk membership function was set to be at  $0 < z \leq 3.5$  m. Unlike previous methodologies, we do not consider this distance as a hard threshold for the identification of an obstacle as threatening. As it can be seen in Fig. 3(a) our high-risk membership function is within the interval  $0 < z \leq 1.5$  m, and it starts degrading up to the 3.5 m where it becomes zero. In that manner, we are able to capture the desired distance up to 2 m and mark it as high risk but also consider the uncertainty around it, i.e., within the interval  $1.5 \text{ m} < z \leq 3.5$  m. This way the risk of an approaching obstacle can be progressively assessed with respect to its distance.

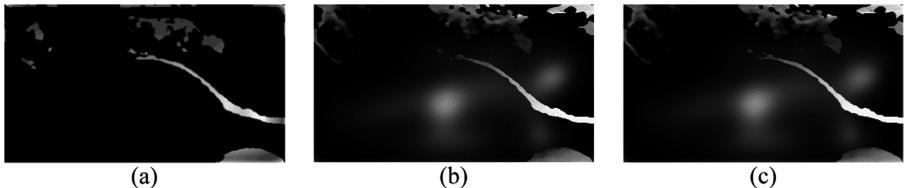


**Fig. 8.** A qualitative comparison of the fuzzy approach (a)  $I_{RGB}$  input image. (b) The hard-thresholded saliency map overlaid to the input image (c) Image  $O^1_{obstacle}$  obtained by the proposed methodology. (d) Obstacle mask after hard thresholding of the depth map corresponding to the region of interest defined in (b).

**Table 1.** Confusion matrix

Predicted values	True values	
	Obstacle	Other
Obstacle	47.50%	6.66%
Other	5.20%	40.64%

The application of the proposed methodology for obstacle detection on the available dataset resulted in an accuracy of 88.1%, with the respective specificity to be 85.9% and the sensitivity 90.1%. The classification results are summarized in detail in Table 1. We compared our approach with a state-of-the-art methodology, where hard thresholding is employed. For a threshold at 3 m the hard thresholding method produced an accuracy of 81.1%, with a specificity of 79.1% and a sensitivity of 82.9%, whereas by further reducing the threshold, the detection performance was degraded. A qualitative comparison of the two approaches is illustrated in Fig. 8. Figure 8(b) illustrates the hard-thresholded saliency map overlaid to the input image. It can be noticed that the rock at the bottom right is not included in the region of interest defined by the saliency map. This region of interest is used to isolate a respective region of the depth map, which is subsequently hard-thresholded to obtain possibly threatening obstacle regions. The result of this process is illustrated in Fig. 8(d), where the rock is falsely not considered as a threat. Comparatively in Fig. 8(c) the image  $O_{obstacle}^1$  obtained by the proposed methodology includes the rock. This is because of the fuzzy fusion applied between the saliency map and the risk-map. The fuzzy fusion operation, defines a region that may not be considered sufficiently salient by SalGAN, but due to the uncertainty-aware evaluation of the depth map using fuzzy sets, a certain degree of risk is assigned. Figure 9 illustrates a qualitative comparison of  $O_{obstacle}^1$  obtained using various T-norms and S-norms, including fuzzy AND Fig. 9(a), OR Fig. 9(b) and SUM Fig. 9(c) operators. It can be observed that the fuzzy AND operation produces the most reliable results.



**Fig. 9.** An illustration of  $O_{obstacle}^1$  using different fuzzy operation between the saliency map  $S_M$  and the risk map  $A_M^1$ . (a) Fuzzy AND. (b) Fuzzy OR. (c) Fuzzy SUM.

## 4 Conclusions

In this work we presented a novel methodology for obstacle detection in the context of safe navigation of VII. It is based on a GAN which produces saliency maps, based on human eye fixations. These maps are co-evaluated with the depth information obtained by an RGB-D camera to assess the risk of an obstacle. Fuzzy sets were used to translate the values of the depth maps to risk levels, and a fuzzy fusion of depth and saliency information was applied to enable enhanced detection of obstacles.

The proposed approach, which is based on fuzzy sets, demonstrated a more robust performance in comparison to the current approach, which is based on hard thresholding. In addition, the proposed approach enables the translation of the 3D spatial information into linguistic values easily comprehensible by VII.

As a future work, we intend to extend this methodology by creating multidimensional membership functions for the risk-level interpretation based on the speed and the location of an obstacle. Also, we intend to investigate various fuzzy set-based approaches to image fusion, and to create an extended dataset that will be made publicly available, to foster research in this domain.

**Acknowledgments.** This research has been co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE – INNOVATE (project code:T1EDK-02070). The Titan X used for this research was donated by the NVIDIA Corporation.

## References

1. Rodríguez, A., Bergasa, L.M., Alcantarilla, P.F., Yebes, J., Cela, A.: Obstacle avoidance system for assisting visually impaired people. In: Proceedings of the IEEE Intelligent Vehicles Symposium Workshops, Madrid, Spain, p. 16 (2012)
2. Iakovidis, D.K., Diamantis, D., Dimas, G., Ntakolia, C., Spyrou, E.: Digital enhancement of cultural experience and accessibility for the visually impaired. In: Paiva, S. (ed.) Improved Mobility for the Visually Impaired. Springer (2019, to appear)
3. Brassai, S.T., Iantovics, B., Enachescu, C.: Optimization of robotic mobile agent navigation. Stud. Inform. Control **21**, 403–412 (2012)
4. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99 (2015)
5. Kaur, B., Bhattacharya, J.: A scene perception system for visually impaired based on object detection and classification using multi-modal DCNN. arXiv preprint [arXiv:1805.08798](https://arxiv.org/abs/1805.08798) (2018)
6. Tapu, R., Mocanu, B., Zaharia, T.: DEEP-SEE: joint object detection, tracking and recognition with application to visually impaired navigational assistance. Sensors **17**, 2473 (2017)
7. Suresh, A., Arora, C., Laha, D., Gaba, D., Bhambri, S.: Intelligent smart glass for visually impaired using deep learning machine vision techniques and robot operating system (ROS). In: Kim, J.-H., Myung, H., Kim, J., Xu, W., Matson, E.T., Jung, J.-W., Choi, H.-L. (eds.) RiTA 2017. AISC, vol. 751, pp. 99–112. Springer, Cham (2019). [https://doi.org/10.1007/978-3-319-78452-6\\_10](https://doi.org/10.1007/978-3-319-78452-6_10)

8. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
9. Liu, W., et al.: SSD: single shot multibox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 21–37. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-46448-0\\_2](https://doi.org/10.1007/978-3-319-46448-0_2)
10. Poggi, M., Mattoccia, S.: A wearable mobility aid for the visually impaired based on embedded 3D vision and deep learning. In: 2016 IEEE Symposium on Computers and Communication (ISCC), pp. 208–213 (2016)
11. Lee, C.-H., Su, Y.-C., Chen, L.-G.: An intelligent depth-based obstacle detection system for visually-impaired aid applications. In: 2012 13th International Workshop on Image Analysis for Multimedia Interactive Services, pp. 1–4. IEEE (2012)
12. Song, H., Liu, Z., Du, H., Sun, G.: Depth-aware saliency detection using discriminative saliency fusion. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1626–1630. IEEE (2016)
13. Mancini, M., Costante, G., Valigi, P., Ciarfuglia, T.A.: J-MOD 2: joint monocular obstacle detection and depth estimation. IEEE Robot. Autom. Lett. **3**, 1490–1497 (2018)
14. Heinrich, S.: Fast obstacle detection using flow/depth constraint. In: 2002 Intelligent Vehicle Symposium, pp. 658–665. IEEE (2002)
15. Chen, L., Guo, B., Sun, W.: Obstacle detection system for visually impaired people based on stereo vision. In: 2010 Fourth International Conference on Genetic and Evolutionary Computing, pp. 723–726. IEEE (2010)
16. Pan, J., et al.: Salgan: visual saliency prediction with generative adversarial networks. arXiv preprint [arXiv:1701.01081](https://arxiv.org/abs/1701.01081) (2017)
17. Goodfellow, I., et al.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp. 2672–2680 (2014)
18. Bylinskii, Z., Recasens, A., Borji, A., Oliva, A., Torralba, A., Durand, F.: Where should saliency models look next? In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9909, pp. 809–824. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-46454-1\\_49](https://doi.org/10.1007/978-3-319-46454-1_49)
19. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint [arXiv:1409.1556](https://arxiv.org/abs/1409.1556) (2014)
20. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database (2009)
21. Nguyen, H.T., Walker, C.L., Walker, E.A.: A First Course in Fuzzy Logic. CRC Press, Boca Raton (2018)

# Author Index

- Ahmad, Nor Bahiah 221  
Al Khafaf, Nameer 31  
Alexandropoulos, Stamatios-Aggelos N. 435  
Alonso, Serafín 307, 463  
Anezakis, Vardis-Dimitris 246  
Anuar, Syahid 221  
Araujo, Vanessa S. 208  
Araujo, Vinicius J. Silva 208  
Aravanis, Theofanis I. 488  
Aravanis, Tryfon-Chrysovalantis I. 488  
Aridas, Christos K. 435
- Badică, Costin 139  
Batista, Lucas O. 208  
Beez, Ulrich 55  
Belbachir, Nabil 368  
Ben Hamadou, Abdelmajid 197  
Bhat, Neeraj 104  
Bhowmik, Arka 104  
Boersma, Kees 368  
Bouraoui, Amal 197  
Bucur, Paul Alexandru 323
- Cateni, Silvia 66  
Chauhan, Vedang 274  
Chengeta, Kennedy 176  
Colla, Valentina 66, 399
- de Campos Souza, Paulo V. 208  
Demertzis, Konstantinos 246  
Dettori, Stefano 66  
Diamantidis, Dimitrios E. 522  
Díaz, Ignacio 307  
Dimas, George 533  
Dokas, Ioannis 368  
Domínguez, Manuel 307, 463
- Fuertes, Juan José 463
- Garro, Beatriz A. 125  
Garrote, L. 349  
Georgakopoulos, Spiros V. 262  
Goodwin, Morten 43  
Guimaraes, Augusto J. 208
- Hassan, Hasniza 221  
Hülsmann, Jens 55  
Humm, Bernhard G. 55  
Hungerländer, Philipp 323  
Hyniová, Kateřina 412
- Iakovidis, Dimitris K. 522, 533  
Iakovidis, Dimitris 501  
Iglesias, A. 349  
Iliadis, Lazaros 246  
Inada, Yoshinobu 424
- Jalili, Mahdi 31, 151  
Jamoussi, Salma 197  
Jiao, Lei 43  
Joorabloo, Nima 151  
Joshi, Keyur D. 274
- Kalogeraki, Eleni-Maria 476  
Karandashev, Iakov 391  
Karatzoglou, Antonios 379  
Kaupp, Lukas 55  
Kesidis, Anastasios 335  
Kolhe, Mohan Lal 43  
Kotsiantis, Sotiris B. 435  
Koutsiou, Dimitra-Christina C. 522  
Koziri, Maria 501  
Kryzhanovsky, Boris 391  
Kumar, Manish 299  
Kumar, Sanjay 104, 299  
Kyrkou, Lamprini 188
- Leon, Florin 235  
Logofătu, Doina 139, 235  
Loukopoulos, Thanasis 501
- Maglogiannis, Ilias G. 262  
Malcangi, Marco 361  
Malcangi, Mario 361  
Mallis, Georgios I. 262  
Mandloi, Yograj S. 424  
Maniatidis, Apostolos 511  
Mathe, Eirini 511  
Matino, Ismael 66

- Matino, Ruben 66  
Mendoza, Leonardo 115  
Mohan, Biju R. 445, 453  
Morán, Antonio 307, 463  
Mouratidis, Haralambos 476  
Muñoz, Cristian 115  
Mylonas, Phivos 511
- Naik, Nagaraj 445, 453  
Nalmpantis, Christoforos 19, 80, 188  
Nielsen, Henrik Kofoed 43  
Nikolaidis, Spyridon 286  
Noori, Nadia Saad 368  
Ntakolia, Charis 533  
Ntalampiras, Stavros 93
- Oikonomou, Panagiotis 501  
Oommen, B. John 3  
Opalic, Sven Myrdahl 43
- Panagou, Natalia 501  
Papadakis, Antonios 511  
Papadopoulos, George 335  
Papadopoulos, Panos K. 501  
Papadopoulos, Polydoros N. 488  
Papastergiou, Spyridon 476  
Patachi, Andreea-Iulia 235  
Pérez, Daniel 307, 463  
Plagianakos, Vassilis P. 262  
Potamias, Rolando-Alexandros 164
- Potamitis, Ilyas 93  
Prada, Miguel A. 463
- Radianti, Jaziar 368  
Refanidis, Ioannis 286  
Reguera, Perfecto 307  
Ren, Yongli 151
- Shirvani, Abdolreza 3  
Silveira, Arthur 115  
Siolas, Georgios 164  
Smítková Janků, Ladislava 412  
Sokolowski, Peter 31  
Spiliotis, M. 349  
Spyrou, Evangelos 511  
Stafylopatis, Andreas 164  
Stieglitz, Stefan 368  
Surgenor, Brian 274  
Symeonidis, Nikolaos 19
- Tasoulis, Sotiris K. 262  
Thuan, Lam Gia 139
- Vannucci, Marco 399  
Vassilas, Nikolaos 335  
Vazquez, Roberto A. 125  
Vernikos, Ioannis 511  
Vrahatis, Aristidis G. 262  
Vrahatis, Michael N. 435  
Vrakas, Dimitris 19, 80, 188