

Ajith Abraham · Niketa Gandhi ·
Thomas Hanne · Tzung-Pei Hong ·
Tatiane Nogueira Rios ·
Weiping Ding *Editors*

Intelligent Systems Design and Applications

21st International Conference
on Intelligent Systems Design and
Applications (ISDA 2021) Held During
December 13–15, 2021

Lecture Notes in Networks and Systems

Volume 418

Series Editor

Janusz Kacprzyk, Systems Research Institute, Polish Academy of Sciences,
Warsaw, Poland

Advisory Editors

Fernando Gomide, Department of Computer Engineering and Automation—DCA,
School of Electrical and Computer Engineering—FEEC, University of Campinas—
UNICAMP, São Paulo, Brazil

Okyay Kaynak, Department of Electrical and Electronic Engineering,
Bogazici University, Istanbul, Turkey

Derong Liu, Department of Electrical and Computer Engineering, University
of Illinois at Chicago, Chicago, USA

Institute of Automation, Chinese Academy of Sciences, Beijing, China

Witold Pedrycz, Department of Electrical and Computer Engineering, University of
Alberta, Alberta, Canada

Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland

Marios M. Polycarpou, Department of Electrical and Computer Engineering,
KIOS Research Center for Intelligent Systems and Networks, University of Cyprus,
Nicosia, Cyprus

Imre J. Rudas, Óbuda University, Budapest, Hungary

Jun Wang, Department of Computer Science, City University of Hong Kong,
Kowloon, Hong Kong

The series “Lecture Notes in Networks and Systems” publishes the latest developments in Networks and Systems—quickly, informally and with high quality. Original research reported in proceedings and post-proceedings represents the core of LNNS.

Volumes published in LNNS embrace all aspects and subfields of, as well as new challenges in, Networks and Systems.

The series contains proceedings and edited volumes in systems and networks, spanning the areas of Cyber-Physical Systems, Autonomous Systems, Sensor Networks, Control Systems, Energy Systems, Automotive Systems, Biological Systems, Vehicular Networking and Connected Vehicles, Aerospace Systems, Automation, Manufacturing, Smart Grids, Nonlinear Systems, Power Systems, Robotics, Social Systems, Economic Systems and other. Of particular value to both the contributors and the readership are the short publication timeframe and the world-wide distribution and exposure which enable both a wide and rapid dissemination of research output.

The series covers the theory, applications, and perspectives on the state of the art and future developments relevant to systems and networks, decision making, control, complex processes and related areas, as embedded in the fields of interdisciplinary and applied sciences, engineering, computer science, physics, economics, social, and life sciences, as well as the paradigms and methodologies behind them.

Indexed by SCOPUS, INSPEC, WTI Frankfurt eG, zbMATH, SCImago.

All books published in the series are submitted for consideration in Web of Science.

For proposals from Asia please contact Aninda Bose (aninda.bose@springer.com).

More information about this series at <https://link.springer.com/bookseries/15179>

Ajith Abraham · Niketa Gandhi ·
Thomas Hanne · Tzung-Pei Hong ·
Tatiane Nogueira Rios ·
Weiping Ding
Editors

Intelligent Systems Design and Applications

21st International Conference on Intelligent
Systems Design and Applications (ISDA
2021) Held During December 13–15, 2021

Editors

Ajith Abraham
Scientific Network for Innovation
and Research Excellence
Machine Intelligence Research
Labs (MIR Labs)
Auburn, WA, USA

Niketa Gandhi
Scientific Network for Innovation
and Research Excellence
Machine Intelligence Research
Labs (MIR Labs)
Auburn, WA, USA

Thomas Hanne
Institut für Wirtschaftsinformatik
Fachhochschule Nordwestschweiz
Olten, Switzerland

Tzung-Pei Hong
Department of Computer Science
and information Engineering
National University of Kaohsiung
Kaohsiung, Taiwan

Tatiane Nogueira Rios
Federal University of Bahia
Ondina, Brazil

Weiping Ding
Nantong University
Nantong Shi, Jiangsu, China

ISSN 2367-3370

ISSN 2367-3389 (electronic)

Lecture Notes in Networks and Systems

ISBN 978-3-030-96307-1

ISBN 978-3-030-96308-8 (eBook)

<https://doi.org/10.1007/978-3-030-96308-8>

© The Editor(s) (if applicable) and The Author(s), under exclusive license
to Springer Nature Switzerland AG 2022

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

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

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

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

Preface

Welcome to the 21st International Conference on Intelligent Systems Design and Applications (ISDA'21) held in the World Wide Web. ISDA'21 is hosted and sponsored by the Machine Intelligence Research Labs (MIR Labs), USA. ISDA'21 brings together researchers, engineers, developers and practitioners from academia and industry working in all interdisciplinary areas of computational intelligence and system engineering to share their experience, and to exchange and cross-fertilize their ideas. The aim of ISDA'21 is to serve as a forum for the dissemination of state-of-the-art research, development and implementations of intelligent systems, intelligent technologies and useful applications in these two fields.

ISDA'21 received submissions from 34 countries, and each paper was reviewed by at least five or more reviewers, and based on the outcome of the review process, 132 papers were accepted for inclusion in the conference proceedings (36% acceptance rate). First, we would like to thank all the authors for submitting their papers to the conference and for their presentations and discussions during the conference. Our thanks go to program committee members and reviewers, who carried out the most difficult work by carefully evaluating the submitted papers. Our special thanks to the following plenary speakers, for their exciting plenary talks:

- Yukio Ohsawa, The University of Tokyo, Japan
- Juergen Branke, University of Warwick, UK
- Cengiz Toklu, Beykent University, Turkey
- Günther Raidl, Technische Universität Wien, Austria
- Kalyanmoy Deb, Michigan State University, USA
- Oscar Cordon, University of Granada, Spain
- Andries Engelbrecht, University of Stellenbosch, South Africa
- Antônio de Padua Braga, Federal University of Minas Gerais, Brazil
- Frédéric Guinand, Le Havre Normandy University, France
- Marco Dorigo, Université Libre de Bruxelles, Belgium

We express our sincere thanks to the organizing committee chairs for helping us to formulate a rich technical program. Enjoy reading the articles!

Ajith Abraham

Thomas Hanne

Weiping Ding

General Chairs

Tzung-Pei Hong

Tatiane Nogueira Rios

Program Chairs

ISDA 2021—Organization

General Chairs

| | |
|---------------|--|
| Ajith Abraham | Machine Intelligence Research Labs (MIR Labs), USA |
| Thomas Hanne | University of Applied Sciences and Arts Northwestern Switzerland, Switzerland |
| Weiping Ding | Nantong University, China |

Program Chairs

| | |
|-----------------------|--|
| Tzung-Pei Hong | National University of Kaohsiung, Taiwan |
| Tatiane Nogueira Rios | Universidade Federal da Bahia, Brazil |

Publication Chairs

| | |
|---------------|---|
| Niketa Gandhi | Machine Intelligence Research Labs, USA |
| Kun Ma | University of Jinan, China |

Special Session Chair

| | |
|--------------------|-------------------------------------|
| Gabriella Casalino | University of Bari Aldo Moro, Italy |
|--------------------|-------------------------------------|

Publicity Chairs

| | |
|----------------------------|---|
| Aswathy S. U. | Jyothi Engineering College, Kerala, India |
| Pooja Manghirmalani Mishra | University of Mumbai, Maharashtra, India |
| Mahendra Kanojia | MVLU College, Maharashtra, India |
| Anu Bajaj | Machine Intelligence Research Labs (MIR Labs), Washington, USA |

International Publicity Team

| | |
|--------------------|---|
| Mabrouka Salmi | National School of Statistics and Applied Economics (ENSSEA), Kolea, Tipaza, Algeria |
| Phoebe E. Knight | UTeM, Malaysia |
| Marco A. C. Simões | Bahia State University, Brazil |
| Hsiu-Wei Chiu | National University of Kaohsiung, Taiwan |
| Serena Gandhi | Media Specialist, CA, USA |

International Program Committee

| | |
|--|--|
| Abdul Syukor Mohamad Jaya | Universiti Teknikal Malaysia Melaka, Malaysia |
| Alfonso Guarino | University of Foggia, Italy |
| Alzira Mota | Polytechnic Institute of Porto, School of Engineering, Portugal |
| André Borges Guimarães Serra e Santos | Polytechnic Institute of Porto, Portugal |
| Angelo Genovese | Università degli Studi di Milano, Italy |
| Anoop Sreekumar R. S. | Manonmaniam Sundaranar University, India |
| Antonella Falini | University of Bari Aldo Moro, Italy |
| Antonella Guzzo | University of Calabria, Italy |
| Anu Bajaj | Machine Intelligence Research Labs, USA |
| Arun B Mathews | MTHSS Pathanamthitta, India |
| Aswathy S. U. | Jyothi Engineering College, India |
| Bay Vo | Ho Chi Minh City University of Technology (HUTECH), Vietnam |
| Biju C. V. | Jyothi Engineering College, India |
| Blerina Spahiu | Università degli Studi di Milano Bicocca, Italy |
| Carlos Pereira | ISEC, Portugal |
| Christian Veenhuis | Technische Universität Berlin, Germany |
| Claudio Savaglio | Institute for High-Performance Computing and Networking (ICAR-CNR), Italy |
| Daniele Schicchi | Institute for Educational Technology, National Research Council of Italy, Italy |
| Devi Priya Rangasamy | Kongu Engineering College, India |
| Dipanwita Thakur | Banasthali University, India |
| E. M. Roopa Devi | Kongu Engineering College, India |
| Ela Pustulka | FHNW Olten, Switzerland |
| Elizabeth Goldberg | Federal University of Rio Grande do Norte, Brazil |
| Fabio Scotti | Università degli Studi di Milano, Italy |
| Fariba Goodarzian | Machine Intelligence Research Labs, USA |
| Federico Divina | Pablo de Olavide University, Spain |
| Gabriella Casalino | University of Bari Aldo Moro, Italy |
| Gautami Tripathi | Jamia Hamdard, India |

| | |
|-------------------------------------|---|
| Gianluca Zaza | University of Bari Aldo Moro, Italy |
| Gonglin Yuan | Victoria University of Wellington, New Zealand |
| Hudson Geovane de Medeiros | Federal University of Rio Grande do Norte, Brazil |
| Isabel S. Jesus | Institute of Engineering of Porto, Portugal |
| Islame Felipe da Costa Fernandes | Federal University of Rio Grande do Norte (UFRN), Brazil |
| Ivo André Soares Pereira | University Fernando Pessoa, Portugal |
| János Botzheim | Eötvös Loránd University, Hungary |
| João Ferreira | Instituto Universitário de Lisboa, Portugal |
| José Everardo Bessa Maia | State University of Ceará, Brazil |
| José Raúl Romero | University of Cordoba, Spain |
| K. Ramesh | Hindustan Institute of Technology and Science, India |
| Kaushik Das Sharma | University of Calcutta, India |
| Kavita Jain | University of Mumbai, Maharashtra, India |
| Kingsley Okoye | Tecnologico de Monterrey, Mexico |
| Kosisochukwu Judith Madukwe | Victoria University of Wellington, New Zealand |
| Leandro Coelho | Pontificia Universidade Católica do Parana, Brazil |
| M. Rubaiyat Hossain Mondal | Bangladesh University of Engineering and Technology, Bangladesh |
| Mahendra Kanojia | Sheth L.U.J. and Sir M.V. College, India |
| Manisha Satish Divate | Usha Pravin Gandhi College of Arts, Science and Commerce, Maharashtra, India |
| Mariella Farella | University of Palermo, Italy |
| Mohd Abdul Ahad | Jamia Hamdard, New Delhi, India |
| Murilo Oliveira Machado | Universidade Federal do Mato Grosso do Sul, Brazil |
| Nidhi Sindhwani | Amity University, Noida, India |
| Niketa Gandhi | Machine Intelligence Research Labs, USA |
| Nuno Miguel Gomes Bettencourt | Polytechnic Institute of Porto (ISEP/IPP), Portugal |
| Nurul Azma Zakaria | Universiti Teknikal Malaysia Melaka, Malaysia |
| Oscar Castillo | Tijuana Institute of Technology, Mexico |
| Pasquale Ardimento | Università degli studi di Bari Aldo Moro, Italy |
| Patrick Hung | Ontario Tech University, Canada |
| Paulo Moura Oliveira | University of Trás-os-Montes and Alto Douro, Portugal |
| Pietro Picerno | Università Telematica “e-Campus”, Italy |
| Pooja Manghirmalani Mishra | University of Mumbai, India |
| Priya P. Sajan | C-DAC, Kerala, India |
| Rafael Barbudo Lunar | University of Córdoba, Spain |
| Reeta Devi | Kurukshetra University, India |

| | |
|----------------------------------|---|
| Rohit Anand | DSEU, G.B. Pant Okhla-1 Campus, New Delhi, India |
| Ruggero Donida Labati | Università degli Studi di Milano, Italy |
| Sabri Pllana | Center for Smart Computing Continuum, Forschung Burgenland, Austria |
| Sidemar Fideles Cezario | Federal University of Rio Grande do Norte, Brazil |
| Sílvia Maria Diniz Monteiro Maia | Federal University of Rio Grande do Norte, Brazil |
| Sindhu P. M. | Nagindas Khandwala College, India |
| Subodh Deolekar | Welingkar Institute of Management Development and Research, India |
| Sulaima Lebbe Abdul Haleem | South Eastern University of Sri Lanka, Sri Lanka |
| Susana Cláudia Nicola de Araújo | Instituto Superior de Engenharia do Porto (ISEP), Portugal |
| Syariffanor Hisham | Universiti Teknikal Malaysia Melaka, Malaysia |
| Thatiana C. Navarro Diniz | Federal Rural University of the Semi-arid Region, Brazil |
| Thiago Soares Marques | Federal University of Rio Grande do Norte, Brazil |
| Thomas Hanne | University of Applied Sciences and Arts Northwestern Switzerland, Switzerland |
| Tzung-Pei Hong | National University of Kaohsiung, Taiwan |
| Wen-Yang Lin | National University of Kaohsiung, Taiwan |
| Wenbin Pei | Lanzhou University, China |
| Yibao Zhang | Lanzhou University, China |
| Youssef Ghanou | Moulay Ismail University of Meknes, Morocco |
| Zurina Saaya | Universiti Teknikal Malaysia Melaka, Malaysia |

Contents

| | |
|---|----|
| Open-Ended Automatic Programming Through Combinatorial Evolution | 1 |
| Sebastian Fix, Thomas Probst, Oliver Ruggli, Thomas Hanne, and Patrik Christen | |
| Deep Face Mask Detection: Prevention and Mitigation of COVID-19 | 13 |
| Sahar Dammak, Hazar Mliki, and Emna Fendri | |
| Extracting Emotion and Sentiment Quotient of Viral Information Over Twitter | 23 |
| Pawan Kumar, Reiben Eappen Reji, and Vikram Singh | |
| Maintaining Scalability in Blockchain | 34 |
| Anova Ajay Pandey, Terrance Frederick Fernandez, Rohit Bansal, and Amit Kumar Tyagi | |
| Thoracic Disease Chest Radiographic Image Dataset: A Comprehensive Review | 46 |
| Priyanka Malhotra, Sheifali Gupta, Atef Zaguia, and Deepika Koundal | |
| Batch Normalization and Dropout Regularization in Training Deep Neural Networks with Label Noise | 57 |
| Andrzej Rusiecki | |
| Intelligent Software Engineering: The Significance of Artificial Intelligence Techniques in Enhancing Software Development Lifecycle Processes | 67 |
| Vaishnavi Kulkarni, Anurag Kolhe, and Jay Kulkarni | |
| Honey Bee Queen Presence Detection from Audio Field Recordings Using Summarized Spectrogram and Convolutional Neural Networks . . . | 83 |
| Agnieszka Orłowska, Dominique Fourer, Jean-Paul Gavini, and Dominique Cassou-Ribehart | |

Formal Verification Techniques: A Comparative Analysis for Critical System Design 93
Rahul Karmakar

Investigating Drug Peddling in Nigeria Using a Machine Learning Approach 103
Oluwafemi Samson Balogun, Sunday Adewale Olaleye, Mazhar Moshin, Keijo Haataja, Xiao-Zhi Gao, and Pekka Toivanen

Selective Information Control and Layer-Wise Partial Collective Compression for Multi-Layered Neural Networks 121
Ryotaro Kamimura

Semantic Representation Driven by a Musculoskeletal Ontology for Bone Tumors Diagnosis 132
Mayssa Bensalah, Atef Boujelben, Yosr Hentati, Mouna Baklouti, and Mohamed Abid

Centrifugal Pump Fault Diagnosis Using Discriminative Factor-Based Features Selection and K-Nearest Neighbors 145
Zahoor Ahmad, Md. Junayed Hasan, and Jong-Myon Kim

Transfer Learning with 2D Vibration Images for Fault Diagnosis of Bearings Under Variable Speed 154
Zahoor Ahmad, Md Junayed Hasan, and Jong-Myon Kim

Performance Evaluation of Microservices Featuring Different Implementation Patterns 165
Leandro Costa and António Nestor Ribeiro

Lifetime Optimization of Sensor Networks with Mobile Sink and Solar Energy Supply 177
Mehdi Achour and Amine Boufaied

Counting Vehicle by Axes with High-Precision in Brazilian Roads with Deep Learning Methods 188
Adson M. Santos, Carmelo J. A. Bastos-Filho, and Alexandre M. A. Maciel

Imbalanced Learning for Robust Moving Object Classification in Video Surveillance Applications 199
Rania Rebai Boukhriss, Ikram Chaabane, Radhouane Guermazi, Emna Fendri, and Mohamed Hammami

Mining Frequently Traveled Routes During COVID-19 210
George Obaido, Kehinde Aruleba, Oluwaseun Alexander Dada, and Ibomoiye Domor Mienye

| | |
|--|------------|
| Analysis of Performance Improvement for Speaker Verification by Combining Feature Vectors of LPC Spectral Envelope, MFCC and pLPC Pole Distribution | 220 |
| Haruki Shigeta, Kodai Komatsu, Shun Oyabu, Kazuya Matsuo, and Shuichi Kurogi | |
| CASTA: Clinical Assessment System for Tuberculosis Analysis | 231 |
| Ramisetty Kavya, Jonathan Samuel, Gunjan Parihar, Y. Suba Joyce, Y. Bakthasingh Lazarus, Subhrakanta Panda, and Jabez Christopher | |
| Bearing Fault Classification of Induction Motor Using Statistical Features and Machine Learning Algorithms | 243 |
| Rafia Nishat Toma and Jong-myon Kim | |
| Evidential Spammers and Group Spammers Detection | 255 |
| Malika Ben Khalifa, Zied Elouedi, and Eric Lefèvre | |
| NLP for Product Safety Risk Assessment | 266 |
| Michael Hellwig, Steffen Finck, Thomas Mootz, Andreas Ehe, and Florian Rein | |
| Augmented Reality for Fire Evacuation Research: An A'WOT Analysis | 277 |
| El Mostafa Bourhim | |
| Optimization of Artificial Neural Network: A Bat Algorithm-Based Approach | 286 |
| Tarun Kumar Gupta and Khalid Raza | |
| ResD Hybrid Model Based on Resnet18 and Densenet121 for Early Alzheimer Disease Classification | 296 |
| Modupe Odusami, Rytis Maskeliūnas, Robertas Damaševičius, and Sanjay Misra | |
| Quantum Ordering Points to Identify the Clustering Structure and Application to Emergency Transportation | 306 |
| Habiba Drias, Yassine Drias, Lydia Sonia Bendimerad, Naila Aziza Houacine, Djaafar Zouache, and Ilyes Khennak | |
| Patterns for Improving Business Processes: Defined Pattern Categorization | 316 |
| Nesrine Missaoui and Sonia Ayachi Ghannouchi | |
| SAX-Preprocessing Technique for Characters Recognition Using Gyroscope Data | 326 |
| Mariem Taktak and Slim Triki | |
| Lower Limb Movement Recognition Using EMG Signals | 336 |
| Sali Issa and Abdel Rohman Khaled | |

| | |
|--|------------|
| A Model of Compactness-Homogeneity for Territorial Design | 346 |
| María Beatriz Bernábe-Loranca, Rogelio González-Velázquez, Carlos Guillen Galván, and Erika Granillo-Martínez | |
| Automated Cattle Classification and Counting Using Hybridized Mask R-CNN and YOLOv3 Algorithms | 358 |
| R. Devi Priya, V. Devisurya, N. Anitha, N. Kalaivaani, P. Keerthana, and E. Adarsh Kumar | |
| UTextNet: A UNet Based Arbitrary Shaped Scene Text Detector | 368 |
| Veronica Naosekham, Sushant Aggarwal, and Nilkanta Sahu | |
| VSIM-AV: A Virtual Simulation Platform for Autonomous Vehicles . . . | 379 |
| Leila Haj Meftah and Rafik Braham | |
| Image Segmentation Using Matrix-Variate Lindley Distributions | 389 |
| Zitouni Mouna and Tounsi Mariem | |
| Improving Speech Emotion Recognition System Using Spectral and Prosodic Features | 399 |
| Adil Chakhtouna, Sara Sekkate, and Abdellah Adib | |
| Spare Parts Sales Forecasting for Mining Equipment: Methods Analysis and Evaluation | 410 |
| Egor Nikitin, Alexey Kashevnik, and Nikolay Shilov | |
| Data-Centric Approach to Hepatitis C Virus Severity Prediction | 421 |
| Aniket Sharma, Ashok Arora, Anuj Gupta, and Pramod Kumar Singh | |
| Automatic Crack Detection with Calculus of Variations | 432 |
| Erika Pellegrino and Tania Stathaki | |
| Deep Squeeze and Excitation-Densely Connected Convolutional Network with cGAN for Alzheimer’s Disease Early Detection | 441 |
| Rahma Kadri, Mohamed Tmar, Bassem Bouaziz, and Faiez Gargouri | |
| Recognition of Person Using ECG Signals Based on Single Heartbeat | 452 |
| Sihem Hamza and Yassine Ben Ayed | |
| Semantic Segmentation of Dog’s Femur and Acetabulum Bones with Deep Transfer Learning in X-Ray Images | 461 |
| D. E. Moreira da Silva, Vitor Filipe, Pedro Franco-Gonçalo, Bruno Colaço, Sofia Alves-Pimenta, Mário Ginja, and Lio Gonçalves | |
| Automatic Microservices Identification from Association Rules of Business Process | 476 |
| Malak Saidi, Mohamed Daoud, Anis Tissaoui, Abdelouahed Sabri, Djamal Benslimane, and Sami Faiz | |

Toward a Configurable Thing Composition Language for the SIoT . . . 488
Soura Boulaares, Salma Sassi, Djamel Benslimane, Zakaria Maamar,
and Sami Faiz

**Comparison of Different Processing Methods of Joint Coordinates
Features for Gesture Recognition with a RNN in the MSRC-12 498**
Júlia Schubert Peixoto, Anselmo Rafael Cukla, Daniel Welfer,
and Daniel Fernando Tello Gamarra

**Summary Generation Using Natural Language Processing Techniques
and Cosine Similarity 508**
Sayantan Pal, Maiga Chang, and Maria Fernandez Iriarte

**An Approach for Constructing a Simulation Model for Dynamic
Analysis of Information Security System 518**
Ivan Gaidarski and Pavlin Kutinchev

**Enhanced Prediction of Chronic Kidney Disease Using Feature
Selection and Boosted Classifiers 527**
Ibomoiye Domor Mienye, George Obaido, Kehinde Aruleba,
and Oluwaseun Alexander Dada

**An Adaptive-Backstepping Digital Twin-Based Approach for Bearing
Crack Size Identification Using Acoustic Emission Signals 538**
Farzin Piltan and Jong-Myon Kim

**Implementation-Oriented Feature Selection in UNSW-NB15 Intrusion
Detection Dataset 548**
Mohammed M. Alani

Augmented Reality SDK's: A Comparative Study 559
El Mostafa Bourhim and Aziz Akhiate

**Hybrid Neural Network for Hyperspectral Satellite Image
Classification (HNN) 567**
Maissa Hamouda and Med Salim Bouhlel

**Implementation of the Business Process Model and Notation in the
Modelling of Patient's Clinical Workflow in Oncology 576**
Nassim Bout, Rachid Khazaz, Ali Azougaghe, Mohamed El-Hfid,
Mounia Abik, and Hicham Belhadaoui

**Mobile Cloud Computing: Issues, Applications and Scope
in COVID-19 587**
Hariket Suresh Kumar Sheth and Amit Kumar Tyagi

Designing a Humanitarian Supply Chain for Pre and Post Disaster Planning with Transshipment and Considering Perishability of Products 601
Faeze Haghgoo, Ali Navaei, Amir Aghsami, Fariborz Jolai, and Ajith Abraham

Innovative Learning Technologies as Support to Clinical Reasoning in Medical Sciences: The Case of the “FEDERICO II” University 613
Oscar Tamburis, Fabrizio L. Ricci, Fabrizio Consorti, Fabrizio Pecoraro, and Daniela Luzi

Convolutional Neural Networks (CNN) Model for Mobile Brand Sentiment Analysis 624
Hamidah Jantan and Puteri Ika Shazereen Ibrahim

How Knowledge-Driven Class Generalization Affects Classical Machine Learning Algorithms for Mono-label Supervised Classification 637
Houcemeddine Turki, Mohamed Ali Hadj Taieb, and Mohamed Ben Aouicha

Deep Residual Network for Autonomous Vehicles Obstacle Avoidance 647
Leila Haj Meftah and Rafik Braham

Modeling Travelers Behavior Using FSQCA 657
Oumayma Labti and Ez-zohra Belkadi

AHP Approach for Selecting Adequate Big Data Analytics Platform . . . 667
Naima EL Haoud and Oumaima Hali

Combining Bert Representation and POS Tagger for Arabic Word Sense Disambiguation 676
Rakia Saidi and Fethi Jarray

Detection of Lung Cancer from CT Images Using Image Processing . . . 686
S. Lilly Sheeba and L. Gethsia Judin

An Overview of IoT-Based Architecture Model for Smart Home Systems 696
Odamboy Djumanazarov, Antti Väänänen, Keijo Haataja, and Pekka Toivanen

Real Time Tracking of Traffic Signs for Autonomous Driving Using Monocular Camera Images 707
Sneha Hegde, K and Srividhya Kannan

| | |
|---|-----|
| Metaheuristic Methods for Water Distribution Network Considering Routing Decision | 723 |
| Ahmad Hakimi, Reza Mahdizadeh, Hossein Shokri Garjan, Amir Khiabani, and Ajith Abraham | |
| Prediction of Moroccan Stock Price Based on Machine Learning Algorithms | 735 |
| Abdelhadi Ifleh and Mounime El Kabbouri | |
| R-DCNN Based Automatic Recognition of Indian Sign Language | 742 |
| S. Subhashini, S. Revathi, and S. Shanthini | |
| VReason Grasp: An Ordered Grasp Based on Physical Intuition in Stacking Objects | 754 |
| Xiang Ji, Qishu Chen, Tao Xiong, Tianyu Xiong, and Huiliang Shang | |
| Comparative Evaluation of Genetic Operators in Cartesian Genetic Programming | 765 |
| Abdul Manazir and Khalid Raza | |
| Prediction of Credibility of Football Player Rating Using Data Analytics | 775 |
| Manaswita Datta and Bhawana Rudra | |
| DDoS Attack Detection on IoT Devices Using Machine Learning Techniques | 787 |
| Sunil Kumar, Rohit Kumar Sahu, and Bhawana Rudra | |
| Functionality and Architecture for a Platform for Independent Learners: KEPLAIR | 795 |
| Stefano Ferilli, Domenico Redavid, Davide Di Pierro, and Liza Loop | |
| Aircraft Conflict Resolution Using Convolutional Neural Network on Trajectory Image | 806 |
| Md Siddiqur Rahman, Laurent Lapasset, and Josiane Mothe | |
| Evaluation of Techniques for Predicting a Build Up of a Seizure | 816 |
| Abir Hadriche, Ichrak ElBehy, Amira Hajjej, and Nawel Jmail | |
| A Real-Time Stereoscopic Images Rectification and Matching Algorithm Based on Python | 828 |
| Elmehdi Adil, Mohammed Mikou, and Ahmed Mouhsen | |
| Named Entities as a Metadata Resource for Indexing and Searching Information | 838 |
| Flavio Izo, Elias Oliveira, and Claudine Badue | |
| Brazilian Mercosur License Plate Detection and Recognition Using Haar Cascade and Tesseract OCR on Synthetic Imagery | 849 |
| Cyro M. G. Sabóia and Pedro Pedrosa Rebouças Filho | |

Designing Scalable Intrusion Detection Systems with Stacking Based Ensemble Learning 859
A. Sujan Reddy, S. Akashdeep, S. Sowmya Kamath, and Bhawana Rudra

Retrofitting Stormwater Harvest System in Dispersing Reliable Water Supply in a Climate-Smart City 870
Bwija Mukome, Muhammed Seyam, and Oseni Amoo

Predicting and Analysis the Bitcoin Price Using Various Forecasting Model 879
E. M. Roopa Devi, R. Shanthakumari, R. Rajdevi, S. Dineshkumar, A. Dinesh, and M. Keerthana

Improved Sentence Similarity Measurement in the Medical Field Based on Syntactico-Semantic Knowledge 890
Wafa Wali and Bilel Gargouri

Analysis of the Brazilian Artisanal Cheese Market from the Perspective of Social Networks 900
Thallys da Silva Nogueira, Vitor Agostinho Mouro, Kenya Beatriz Siqueira, and Priscila V. Z. C. Goliatt

PONY: Predicting an Object’s Next_Location Using YOLO 910
Aris E. Ignacio and Julian Antonio S. Laspoña

Role of Machine Learning in Authorship Attribution with Select Stylometric Features 920
Sumit Gupta, Tapas Kumar Patra, and Chitrita Chaudhuri

COVID Detection Using Chest X-Ray and Transfer Learning 933
Saksham Jain, Nidhi Sindhwani, Rohit Anand, and Ramani Kannan

ECFAR: A Rule-Based Collaborative Filtering System Dealing with Evidential Data 944
Nassim Bahri, Mohamed Anis Bach Tobji, and Boutheina Ben Yaghlane

Enhancing Photography Management Through Automatically Extracted Metadata 956
Pedro Carvalho, Diogo Freitas, Tiago Machado, and Paula Viana

A Machine Learning Framework for House Price Estimation 965
Adebayosoye Awonaike, Seyed Ali Ghorashi, and Rawad Hammaad

A Dedicated Temporal Erasable-Itemset Mining Algorithm 977
Tzung-Pei Hong, Hao Chang, Shu-Min Li, and Yu-Chuan Tsai

Denoising Hyperspectral Imageries with Split-Bregman Iteration Scheme 986
Satwinder Kaur, Bhawna Goyal, and Ayush Dogra

| | |
|--|------|
| iWAD: An Improved Wormhole Attack Detection System for Wireless Sensor Network | 1002 |
| Virendra Dani, Radha Bhonde, and Ayesha Mandloi | |
| Twitter People's Opinions Analysis During Covid-19 Quarantine Using Machine Learning and Deep Learning Models | 1013 |
| Wafa Alotaibi, Faye Alomary, and Raouia Mokni | |
| Estimation and Aggregation Method of Open Data Sources for Road Accident Analysis | 1025 |
| Sergey Savosin and Nikolay Teslya | |
| A Hybrid Approach for an Interpretable and Explainable Intrusion Detection System | 1035 |
| Tiago Dias, Nuno Oliveira, Norberto Sousa, Isabel Praça, and Orlando Sousa | |
| An IoT Based Home Automation System VIA Hotspot | 1046 |
| Raihan Uddin | |
| Genomic Variant Annotation: A Comprehensive Review of Tools and Techniques | 1057 |
| Prajna Hebbar and S. Kamath Sowmya | |
| Age Estimation and Gender Recognition Using Biometric Modality . . . | 1068 |
| Amal Abbes, Randa Boukhris, and Yassine Ben Ayed | |
| Towards a Historical Ontology for Arabic Language: Investigation and Future Directions | 1078 |
| Rim Laatar, Ahlem Rhayem, Chafik Aloulou, and Lamia Hadrich Belguith | |
| Optimized Evidential AIRS with Feature Selection and Genetic Algorithm | 1088 |
| Rihab Abdelkhalek and Zied Elouedi | |
| Predicting the Movement Intention and Controlling the Grip of a Myoelectrical Active Prosthetic Arm | 1098 |
| Jonatan Dellagostin, Anselmo Cukla, Fábio Bisogno, Raul Sales, Lucas Strapazzon, and Gregório Salvador | |
| An Evolutionary Approach for Critical Node Detection in Hypergraphs. A Case Study of an Inflation Economic Network | 1110 |
| Noémi Gaskó, Mihai Suciu, Rodica Ioana Lung, and Tamás Képes | |
| A Modified Technique Based on GOMASHIO Method for Mobile Nodes Localization in a WSN | 1118 |
| Omnia Mezghani | |
| Attitude Prediction of In-service Teachers Towards Blended Learning Using Machine Learning During COVID-19 Pandemic | 1129 |
| Pooja Manghirmalani Mishra, Rabiya Saboowala, and Niketa Gandhi | |

| | |
|--|-------------|
| Driver Behavior Analysis: Abnormal Driving Detection Using MLP Classifier Applied to Outdoor Camera Images | 1142 |
| Wictor Gomes de Oliveira, Pedro Pedrosa Rebouças Filho, and Elias Teodoro da Silva Junior | |
| Supporting Reusability in the Scrum Process | 1153 |
| Oumaima Bhiri, Khaoula Sayeb, and Sonia Ayachi Ghannouchi | |
| Arabic Automatic Essay Scoring Systems: An Overview Study | 1164 |
| Rim Aroua Machhout, Chiraz Ben Othmane Zribi, and Saoussen Mathlouthi Bouzid | |
| Energy-Efficient Khalimsky-Based Routing Approach for K-Hop Clustered Wireless Multimedia Sensor Networks (WMSNs) | 1177 |
| Mahmoud Mezghani | |
| Academic Venue Recommendation Based on Refined Cross Domain . . . | 1188 |
| Abir Zawali and Imen Boukhris | |
| Typology of Data Inputs Imperfection in Collective Memory Model . . . | 1198 |
| Haithem Kharfia, Fatma Ghorbel, and Bilel Gargouri | |
| How Latest Computer Science Research Copes with COVID-19? | 1207 |
| Leila Bayoudhi, Najla Sassi, and Wassim Jaziri | |
| Using Machine Learning Approaches to Identify Exercise Activities from a Triple-Synchronous Biomedical Sensor | 1216 |
| Yohan Mahajan, Jahn timer Pinnamraju, John L. Burns, Judy W. Gichoya, and Saptarshi Purkayastha | |
| Intelligent Image Captioning Approach with Novel Ensembled Recurrent Neural Network Model | 1227 |
| L. Agilandeewari, Kuhoo Sharma, and Saurabh Srivastava | |
| Analysis of Six Different GP-Tree Neighborhood Structures | 1237 |
| Souhir Elleuch and Bassem Jarboui | |
| Ensemble Learning for Data-Driven Diagnosis of Polycystic Ovary Syndrome | 1250 |
| Subrato Bharati, Prajoy Podder, M. Rubaiyat Hossain Mondal, V. B. Surya Prasath, and Niketa Gandhi | |
| Tree Species Detection Using MobileNet – An Approach | 1260 |
| L. Agilandeewari, Aayush Jha, and Dhruv Gupta | |
| Dimensional Reduction Methods Comparison for Clustering Results of Indonesian Language Text Documents | 1271 |
| Siti Inayah Rizki Hasanah, Muhammad Ihsan Jambak, and Danny M. Saputra | |

**Gun Model Classification Based on Fired Cartridge Case Head
Images with Siamese Networks 1281**
Sérgio Valentim, Tiago Fonseca, João Ferreira, Tomás Brandão,
Ricardo Ribeiro, and Stefan Nae

**Image-based Android Malware Detection Models using Static
and Dynamic Features 1292**
Hemant Rathore, B. Raja Narasimhan, Sanjay K. Sahay, and Mohit Sewak

**A Fuzzy Logic Based Optimal Network System for the Delivery of
Medical Goods via Drones and Land Transport in Remote Areas 1306**
Shio Gai Quek, Ganeshsree Selvachandran, Rohana Sham, Ching Sin Siau,
Mohd Hanif Mohd Ramli, and Noorsiah Ahmad

The Menu Planning Problem: A Systematic Literature Review 1313
Dorra Kallel, Ines Kanoun, and Diala Dhouib

**Comparative Study on Deep Learning Methods for Apple Ripeness
Estimation on Tree 1325**
Raja Hamza and Mohamed Chtourou

Neuro-Fuzzy Systems for Learning Analytics 1341
Gabriella Casalino, Giovanna Castellano, and Gianluca Zaza

**Prediction of COVID-19 Active Cases Using Polynomial Regression
and ARIMA Models 1351**
Neji Neily, Boulbaba Ben Ammar, and Habib M. Kammoun

**Land Use/Land Cover Classification Using Machine Learning and
Deep Learning Algorithms for EuroSAT Dataset – A Review 1363**
Agilandeewari Loganathan, Suri Koushmitha,
and Yerru Nanda Krishna Arun

A Dynamic Rain Detecting Car Wiper 1375
Andeotum Roland, John Wejin, Sanjay Misra, Mayank Mohan Sharma,
Robertas Damaševičius, and Rytis Maskeliūnas

**Crude Oil Price Prediction Using Particle Swarm Optimization and
Classification Algorithms 1384**
Emmanuel Abidemi Adeniyi, Babatunde Gbadamosi,
Joseph Bamidele Awotunde, Sanjay Misra, Mayank Mohan Sharma,
and Jonathan Oluranti

**An Efficient Thyroid Disease Detection Using Voting Based Ensemble
Classifier 1395**
L. Agilandeewari, Ishita Khatri, Jagruta Advani,
and Syed Mohammad Nihal

| | |
|---|------|
| A Cross-Entropy Based Feature Selection Method for Binary Valued Data Classification | 1406 |
| Zhipeng Wang and Qiuming Zhu | |
| Effective Music Suggestion Using Facial Recognition | 1417 |
| A. P. Ponselvakumar, S. Anandamurugan, K. Lokeshwaran, Suganneshan, Zubair, and Gokula Kannan | |
| A Survey on SLA Management Using Blockchain Based Smart Contracts | 1425 |
| Nawel Hamdi, Chiraz El Hog, Raoudha Ben Djemaa, and Layth Sliman | |
| Author Index | 1435 |



Open-Ended Automatic Programming Through Combinatorial Evolution

Sebastian Fix, Thomas Probst, Oliver Ruggli, Thomas Hanne,
and Patrik Christen^(✉)

FHNW University of Applied Sciences and Arts Northwestern Switzerland,
4600 Olten, Switzerland
`patrik.christen@fhnw.ch`

Abstract. Combinatorial evolution – the creation of new things through the combination of existing things – can be a powerful way to evolve rather than design technical objects such as electronic circuits. Intriguingly, this seems to be an ongoing and thus open-ended process creating novelty with increasing complexity. Here, we employ combinatorial evolution in software development. While current approaches such as genetic programming are efficient in solving particular problems, they all converge towards a solution and do not create anything new anymore afterwards. Combinatorial evolution of complex systems such as languages and technology are considered open-ended. Therefore, open-ended automatic programming might be possible through combinatorial evolution. We implemented a computer program simulating combinatorial evolution of code blocks stored in a database to make them available for combining. Automatic programming in the sense of algorithm-based code generation is achieved by evaluating regular expressions. We found that reserved keywords of a programming language are suitable for defining the basic code blocks at the beginning of the simulation. We also found that placeholders can be used to combine code blocks and that code complexity can be described in terms of the importance to the programming language. As in a previous combinatorial evolution simulation of electronic circuits, complexity increased from simple keywords and special characters to more complex variable declarations, class definitions, methods, and classes containing methods and variable declarations. Combinatorial evolution, therefore, seems to be a promising approach for open-ended automatic programming.

Keywords: Automatic programming · Combinatorial evolution · Open-endedness

1 Introduction

Genetic algorithms and evolutionary computation in general are widely used for solving optimisation problems [6]. Such algorithms follow the paradigm of biological evolution. They consist of a collection of virtual organisms, where every organism represents a possible solution to a given problem. Some fitness

measure is then calculated for each organism in an iterative process and it tries to find improved solutions by forming random mutations and crossovers on them.

In contrast to such evolutionary computation, *combinatorial evolution* as proposed by W. Brian Arthur [1, 2], makes no modifications to the organisms themselves. New solutions are formed through the combination of existing components which then form new solutions in later iterations with the goal of satisfying certain needs. The more useful a combination is, the higher is its need rating. Combining existing components to construct new components can be observed in the evolution of technology [1, 2]. For instance, the invention of radar was only possible through combining simpler electronic parts fulfilling functions like amplification and wave generation [3]. In order to investigate combinatorial evolution, Arthur and Polak [3] created a simple computer simulation, where electronic circuits were evolved in a combinatorial manner. Their simulation started by randomly combining primitive elementary logic gates and then used these simpler combinations for more complicated combinations in later iterations. Over time, a small number of simple building blocks was transformed into many complicated ones, where some of them might be useful for future applications. It was concluded that combinatorial evolution allows building some kind of library of building blocks for the creation of future and more complicated building blocks.

Intriguingly, combinatorial evolution is a key ingredient to achieve open-ended evolution [27, 29], that is the ongoing creation of novelty [4, 26]. This contrasts classical computational approaches where the aim is to converge towards a solution as fast as possible. Computational approaches according to open-ended evolution are therefore not more efficient but they are more creative since they generate ongoing novelty.

Here we want to explore whether combinatorial evolution could be also applied to software development, more specifically to automatic programming to eventually make it open-ended [7]. An early idea of automatic programming was to implement high-level programming languages that are more human readable resulting in compilers, which produce low-level programs – down to machine code – from human readable syntax [8]. However, human input in some form was still needed and the programming task was simply transferred to a higher level. Furthermore, the software solution is limited by the programmer’s capabilities and creativity. Language therefore remains a barrier between programmers and computers. A way around this barrier would be to let the computer do the programming (also occasionally denoted as metaprogramming [10]), which might even lead to better programs. Koza [18] addressed this issue through genetic programming, where populations of computer programs are generated by a computer using genetic algorithms. The problem space consists of programs that try to solve (or approximately solve) problems. It has been demonstrated that random mutations and crossovers in source code can effectively contribute in creating new sophisticated programs [24].

Therefore, it seems possible to define a programming task and let a computer do the programming. However, looking at the process of software development, programming seems more comparable to technological rather than biological evolution. Existing libraries or algorithms are often integrated into new software

without the necessity of modifying them. Therefore, an automatic programming approach that creates new computer programs by means of combinatorial evolution might be an interesting alternative to genetic programming. Also, due to open-endedness, combinatorial evolution holds the promise to be more creative generating ongoing novelty. In the present study we investigate ways to define a programming task for automatic programming through combinatorial evolution including the evaluation of the generated code with a need rating. Our research question is whether it is possible to generate computer programs of increasing complexity using automatic programming through combinatorial evolution. Specifically, we ask what kind of basic code blocks are needed at the beginning? How are these code blocks implemented to allow them to combine? How can code complexity be measured?

2 Automatic Programming

Since the development of computers, it has been a challenge to optimise and adapt program code to access the potential performance of a computer. While the computational power of computers has been steadily increasing in recent years, program code is still limited by the ability of programmers to create efficient and functioning code. Programming languages have also evolved over the past decades. The development of programming languages has sought to provide programmers with abstractions at higher levels. However, this also led to limitations, especially regarding performance and creativity. It is thus intriguing to shift the programming to the computer itself. Most of the programming is currently done by human programmers, which often leads to a time-intensive and error-prone process of software development. The idea that computers automatically create software programs has been a long-standing goal [5] with the potential to streamline and improve software development.

Automatic programming was first considered in the 1940s describing the automation of a manual process in general and with the goal to maximise efficiency [22]. Later, automatic programming was considered a type of computer programming in which code is generated using tools that allow developers to write code at a higher level of abstraction [22]. There are two main types of automatic programming: *application generators* and *generative programming*. Cleaveland [9] describes the development of application generators as the use of high-level programming models or templates to translate certain components into low-level source code. Generative programming, on the other hand, assists developers in writing programs. This can be achieved, e.g. by providing standard libraries as a form of reusable code [10]. In generative programming it is crucial to have a domain model, which consists of three main parts: a problem space, a solution space, and a configuration knowledge mapping that connects them [11]. The problem space includes the features and concepts used by application engineers to express their needs. These can be textual or graphical programming languages, interactive wizards, or graphical user interfaces. The solution space consists of elementary components with a maximum of combinability and

a minimum of redundancy. The configuration knowledge mapping presents a form of generator that translates the objects from the problem space to build components in the solution space [10]. Most recently, automatic programming shifted towards higher level programming languages and incorporating even more abstraction [21].

While these kinds of automatic programming heavily depend on human interaction and thus the capabilities and creativity of programmers, genetic programming can be regarded an attempts to reduce this dependency and shift the focus to automation done by the computer itself. Koza [18] describes genetic programming as a type of programming in which programs are regarded as genes that can be evolved using genetic algorithms [15, 16]. It aims to improve the performance of a program to perform a predefined task. According to Becker et al. [5], a genetic algorithm takes, as an input, a set of instructions or actions that are regarded as genes. A random set of these instructions is then selected to form an initial sequence of DNA. The whole genome is then executed as a program and the results are scored in terms of how well the program solves a predefined task. Afterwards, the top scorers are used to create offspring, which are rated again until the desired program is produced. To find new solutions, evolutionary techniques such as crossover, mutation, and replication are used [23]. Crossover children are created by picking two parents and switching certain components. Another technique is mutation, which uses only one individual parent and randomly modifies its parts to create a new child. Sometimes parents with great fitness will be transferred to the next iteration without any mutation or crossover because they might do well in later steps as well.

3 Combinatorial Evolution

With combinatorial evolution, new solutions build on combinations of previously discovered solutions. Every evolution starts with some primitive, existing building blocks and uses them to build combinations. Those combinations are then stored in an active repertoire. If the output satisfies a need better than an earlier solution, it replaces the old one and will be used as the building block in later iterations. Building blocks are thus not modified, they are combined together creating new building blocks. The result is a library of functionalities that may be useful for a solution in the future [1, 2].

As Ogburn [20] suggested, the more equipment there is within a material culture, the greater the number of inventions are. This is known as the Ogburn's Claim. It can therefore be inferred that the number and diversity of developed components as well as their technological developments matters because next generation components build upon the technological level of the previous, existing components. To investigate this, Arthur and Polak [3] created a simple computer simulation to 'discover' new electronic circuits. In their simulation, they used a predefined list of truth tables of basic logic functions such as full adders or n-bit adders. Every randomly created combination represented a potential satisfaction of a need, which was then tested against this list. If the truth table

of a newly created circuit matched one from the predefined list, it is added to the active repertoire as it fulfils the pre-specified functionality. Sometimes, it also replaced one that was found earlier, if it used fewer parts and therefore would cost less. New technologies in the real world are not usually found by randomly combining existing ones nor do they exist in a pre-specified list to be compared against. Nevertheless, their needs are generally clearly visible in economics and current technologies [3].

Combinatorial evolution is in general an important element of evolutionary systems. Stefan Thurner and his colleagues developed a general model of evolutionary dynamics in which the combination of existing entities to create new entities plays a central role [27–29]. They were able to validate this model using world trade data [17], therefore underlining the importance of evolutionary dynamics in economic modelling in general and combinatorial interactions in particular. The model shows punctuated equilibria that are typical for open-ended evolutionary systems [27–29].

4 Code Complexity

Genetic algorithms have been used for automatic programming already, however, a large number of iterations are required to significantly increase code complexity in order to solve more complex problems [14]. It therefore seems beneficial to use combinatorial evolution in which complexity seems to increase in fewer steps and thus less time.

Code complexity has been measured in this context with different approaches. The cyclomatic complexity of a code is the number of linearly independent paths within it [12]. For instance, if the code contains no control flow elements (conditionals), the complexity would be 1, since there would be only a single path through the code [30]. If the code has one single-condition IF statement, the complexity would be 2 because there would be two paths through the code – one where the IF statement evaluates to TRUE and another one where it evaluates to FALSE [30]. Two nested single-condition IFs (or one IF with two conditions) would produce a complexity of 3 [19, 30]. According to Garg [13], cyclomatic complexity is one of the most used and renowned software metrics together with other proposed and researched metrics, such as the number of lines of code and the Halstead measure. Although cyclomatic complexity is very popular, it is difficult to calculate for object-oriented code [25].

5 Methods

5.1 Development Setup and Environment

We used the programming language Java though other programming languages would have been feasible as well. The development environment was installed on VirtualBox – an open source virtualisation environment from Oracle. Oracle Java SE Development Kit 11 was used with Apache Maven as build automation

tool. To map the existing code with a database, Hibernate ORM was used. It allows mapping object-oriented Java code to a relational database. Furthermore, code versioning with GitHub was used.

5.2 Simulation

Simulations are initialised by adding some basic code building blocks into a repository. The first simulation iteration then starts by randomly selecting code blocks from this repository. Selected blocks are then combined into a new code block, which subsequently gets analysed for its usefulness and complexity. Based on this analysis, the code block is assigned a value. Nonsense code, which is the most common result when randomly combining keywords of a programming language, are assigned a value of 0 and not used any further. Only code blocks with a value greater than 0 are added to the repository and consequently have a chance of being selected in a later iteration.

5.3 Code Building Blocks

Preliminary experiments in which code snippets with placeholders were predefined showed that this approach would limit the creativity and complexity of the automatic programming solution by the predefined snippets. The simulation would only create program logic that is already given by the basic set of code blocks.

To overcome this limitation, we defined basic code building blocks according to keywords and special characters of the Java programming language, e.g. the keywords `int`, `for`, `class`, and `String` as well as the special characters `&`, `=`, `;`, and `{`. Additionally, we defined three more extra code blocks: First, `PLACEHOLDER` to define where blocks allow other code blocks to be combined and integrated. This is particularly important for nesting certain code elements, such as methods that must be nested into a class construct to be valid Java code. Second, `NAME` to name something, e.g. classes, methods, and variables. And third, the special keyword `main` in the main method definition.

5.4 Selecting and Combining Code Blocks

During the selection process, new source code is generated based on combinations of existing code blocks from the repository. The chance that a particular code block is selected depends on its classification value (see next section). In a first step, a helper function defines a random value of how many code blocks are taken into consideration in the current iteration. There is a minimum of two code blocks required to generate a new code block. The maximum number can be predefined in the program. Arthur and Polak [3] combined up to 12 building blocks. To reduce the number of iterations needed for receiving valid Java code, a maximum of eight blocks turned out to be a good limit. After randomly defining the number of code blocks to be combined, the weighted random selection of code blocks

based on their classification value follows. Instead of simply chaining all selected code blocks together, there is also the possibility to nest them into a placeholder if available. A random function decides whether a code block is nested into the placeholder, or simply added to the whole code block. This procedure is important because program code usually exhibits such nested structures.

5.5 Code Analysis and Building Block Classification

After the selection and combination process, the newly generated source code is passed into the classification function where it gets analysed. The classification process is required to weight the different code blocks according to their relevance in the Java programming language and to see whether the code evolved with respect to complexity. This is achieved with regular expression patterns, which allow identifying relevant Java code structures such as classes and methods that can be weighted with predefined classification values for these code structures. Basic structures such as variable declarations are assigned a value of 1. More elaborate structures such as classes have a value of 2 and even more complicated structures such as methods have a value of 3. If a structure contains several of these substructures, their classification values is added. An important structure in many programming languages is the declaration of a variable. With the following regular expression, any declaration of the value types `boolean`, `byte`, `char`, `double`, `float`, `int`, `long`, and `short` are detected:

```
(PLACEHOLDER(?!PLACEHOLDER))?  
(boolean|byte|char|double|float|int|long|short) NAME;  
(PLACEHOLDER(?!PLACEHOLDER))?
```

Other important elements are brackets. E.g. they are used in methods and classes specifying the body. The syntax is given by the programming language. Placeholders inside brackets are important, they allow new code to be injected into existing code blocks in future combinations. We therefore created the following regular expression:

```
^(\\{PLACEHOLDER\\}|\\(PLACEHOLDER\\))$
```

As already shown in the simple simulation with electronic circuits [3], one needs a minimal complexity of the initial building blocks to be able to generate useful and more complex future combinations. Classes and methods are essential to build anything complex in Java. Therefore, regular expressions were implemented to identify valid classes and methods. Valid means, the element is closed and it successfully compiles. Variable declarations and methods are allowed to be nested in the class structure. The following regular expression to detect classes was developed:

```
(protected|private|public) class NAME \\{  
((boolean|byte|char|double|float|int|long|short) NAME;  
|(protected|private|public) void NAME\\(  
((boolean|byte|char|double|float|int|long|short) NAME)?\\) \\{  
((boolean|byte|char|double|float|int|long|short) NAME;
```