River Publishers Series in Communications

Artificial Intelligence for Digitising Industry Applications

Editors:
Ovidiu Vermesan
Reiner John
Cristina De Luca
Marcello Coppola

River Publishers

Artificial Intelligence for Digitising Industry

Applications

RIVER PUBLISHERS SERIES IN COMMUNICATIONS

Series Editors:

ABBAS JAMALIPOUR
The University of Sydney
Australia

MARINA RUGGIERI
University of Rome Tor Vergata
Italy

JUNSHAN ZHANG
Arizona State University
USA

The "River Publishers Series in Communications" is a series of comprehensive academic and professional books which focus on communication and network systems. Topics range from the theory and use of systems involving all terminals, computers, and information processors to wired and wireless networks and network layouts, protocols, architectures, and implementations. Also covered are developments stemming from new market demands in systems, products, and technologies such as personal communications services, multimedia systems, enterprise networks, and optical communications.

The series includes research monographs, edited volumes, handbooks and textbooks, providing professionals, researchers, educators, and advanced students in the field with an invaluable insight into the latest research and developments.

For a list of other books in this series, visit www.riverpublishers.com

Artificial Intelligence for Digitising Industry
Applications

Editors

Ovidiu Vermesan
SINTEF, Norway

Reiner John
AVL List, Austria

Cristina De Luca
Infineon Technologies, Germany

Marcello Coppola
STMicroelectronics, France

River Publishers

Published, sold and distributed by:
River Publishers
Alsbjergvej 10
9260 Gistrup
Denmark

www.riverpublishers.com

ISBN: 978-87-7022-664-6 (Hardback)
978-87-7022-663-9 (Ebook)

©The Editor(s) (if applicable) and The Author(s) 2021. This book is published open access.

Open Access
This book is distributed under the terms of the Creative Commons Attribution-Non-Commercial 4.0 International License, CC-BY-NC 4.0) (http://creativecommons.org/licenses/by/4.0/), which permits use, duplication, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, a link is provided to the Creative Commons license and any changes made are indicated. The images or other third party material in this book are included in the work's Creative Commons license, unless indicated otherwise in the credit line; if such material is not included in the work's Creative Commons license and the respective action is not permitted by statutory regulation, users will need to obtain permission from the license holder to duplicate, adapt, or reproduce the material.

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

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

Printed on acid-free paper.

Dedication

"Action is the real measure of intelligence."

- Napoleon Hill

"Intelligence is the ability to adapt to change."

- Stephen Hawking

"Intelligence is quickness in seeing things as they are."

- George Santayana

"Everything you can imagine is real."

- Pablo Picasso

Acknowledgement

The editors would like to thank all the contributors for their support in the planning and preparation of this book. The recommendations and opinions expressed in the book are those of the editors, authors, and contributors and do not necessarily represent those of any organizations, employers, or companies.

Ovidiu Vermesan
Reiner John
Cristina De Luca
Marcello Coppola

Contents

Preface **xix**

Editors Biography **xxi**

List of Figures **xxv**

List of Tables **xxxiii**

List of Abbreviations **xxxv**

1 AI Automotive **1**

1.0 AI Reshaping the Automotive Industry **3**
Daniel Plorin
1.0.1 Introduction and Background 3
1.0.2 AI Developments and Future Trends for AI Technologies . . 4
1.0.3 AI-Based Applications 6

1.1 AI for Inbound Logistics Optimisation in Automotive Industry **11**
Nikolaos Evangeliou, George Stamatis, George Bravos, Daniel Plorin and Dominik Stark
1.1.1 Introduction and Background 12
1.1.2 Requirements – User Journeys 12
1.1.3 Data Flow Principles and Architecture of the MPDSS 13
1.1.4 Preliminary Analysis of Data and Dataset 15
1.1.4.1 Data Pre-processing and Visualisation 16
1.1.4.2 Classification Models 16
1.1.5 Conclusion 18

1.2 State of Health Estimation using a Temporal Convolutional Network for an Efficient Use of Retired Electric Vehicle Batteries within Second-Life Applications **21**
Steffen Bockrath, Stefan Waldhör, Harald Ludwig, and Vincent Lorentz
1.2.1 Retired Electric Vehicle Batteries for Second-Life Applications . . . 22
1.2.2 State of Health of Lithium-Ion Batteries . . . 24
1.2.3 Data Measurement Using the Open-Source BMS foxBMS . 25
1.2.4 Temporal Convolutional Neural Network for State of Health Prediction . . . 27
1.2.4.1 Causal Convolutions and Receptive Field . . . 27
1.2.4.2 Dilated Convolutions . . . 28
1.2.4.3 Residual Block . . . 28
1.2.5 Results . . . 29
1.2.6 Conclusion . . . 31

1.3 Optimising Trajectories in Simulations with Deep Reinforcement Learning for Industrial Robots in Automotive Manufacturing **35**
Noah Klarmann, Mohammadhossein Malmir, Josip Josifovski, Daniel Plorin, Matthias Wagner and Alois C. Knoll
1.3.1 Introduction . . . 36
1.3.2 Background . . . 38
1.3.3 Methodology . . . 39
1.3.4 Conclusion and Outlook . . . 42

1.4 Foundations of Real Time Predictive Maintenance with Root Cause Analysis **47**
Franz Wotawa, David Kaufmann, Adil Amukhtar, Iulia Nica, Florian Klück, Hermann Felbinger, Petr Blaha, Matus Kozovsky, Zdenek Havranek and Martin Dosedel
1.4.1 Introduction and Background . . . 48
1.4.2 Foundations . . . 51
1.4.2.1 Model-based Diagnosis . . . 51
1.4.2.2 Machine Learning Based Diagnosis . . . 52
1.4.2.3 Artificial Neural Networks for Diagnostics . . . 53
1.4.3 Related Research . . . 54
1.4.4 Conclusion . . . 55

1.5 Real-Time Predictive Maintenance – Model-Based, Simulation-Based and Machine Learning Based Diagnosis **63**
Franz Wotawa, David Kaufmann, Adil Amukhtar, Iulia Nica, Florian Klück, Hermann Felbinger, Petr Blaha, Matus Kozovsky, Zdenek Havranek and Martin Dosedel
1.5.1 Introduction and Background 64
1.5.2 Application of Diagnosis Systems Based on Simplified DC e-Motor Model 65
1.5.2.1 Simplified DC e-Motor Model With Fault Injection Capabilities 66
1.5.2.2 Model-based Diagnosis for Simplified DC e-Motor . 67
1.5.2.3 Simulation-Based Diagnosis for Simplified DC e-Motor 71
1.5.2.4 Machine Learning for Diagnosis of Simplified DC e-Motor 75
1.5.2.5 Comparisons and Limitations 79
1.5.3 Conclusion 80

1.6 Real-Time Predictive Maintenance – Artificial Neural Network Based Diagnosis **83**
Petr Blaha, Matus Kozovsky, Zdenek Havranek, Martin Dosedel, Franz Wotawa, David Kaufmann, Adil Amukhtar, Iulia Nica, Florian Klück and Hermann Felbinger
1.6.1 Introduction and Background 84
1.6.1.1 AI-based Diagnosis of E-motors 84
1.6.1.2 Artificial Intelligence in Vibration Diagnosis 85
1.6.2 Artificial Neural Network for e-Motor Diagnosis 85
1.6.2.1 Acausal e-Motor Model with Faults Injection Capability 86
1.6.2.2 Artificial Neural Network for Inter-turn Short Circuit Detection 87
1.6.2.2.1 Selection of suitable condition indicator for fault detection 88
1.6.2.2.2 Network structure selection 90
1.6.2.2.3 Data pre-processing 90
1.6.2.2.4 Preparation of datasets 92
1.6.2.2.5 MNN training 92
1.6.2.2.6 MNN validation 92
1.6.2.2.7 MNN deployment 93

1.6.3 Artificial Neural Network based Vibration Diagnosis 93
1.6.3.1 Vibration Diagnosis of Rotating Machines 93
1.6.3.2 AI Approaches in Vibration Diagnosis 94
1.6.3.3 MLP implementable in device at the edge 95
1.6.4 Conclusion 99

2 AI Semiconductor 103

2.0 AI in Semiconductor Industry 105
Cristina De Luca, Bernhard Lippmann, Wolfgang Schober, Saad Al-Baddai, Georg Pelz, Andreja Rojko, Frédéric Pétrot, Marcello Coppola and Reiner John
2.0.1 Introduction and Background 106
2.0.2 AI Developments in Semiconductor Industry 106
2.0.3 Future Trends for AI Technologies and Applications in Semiconductor Industry 108
2.0.4 AI-Based Applications 109

2.1 AI-Based Knowledge Management System for Risk Assessment and Root Cause Analysis in Semiconductor Industry 113
Houssam Razouk, Roman Kern, Martin Mischitz, Josef Moser, Mirhad Memic, Lan Liu, Christian Burmer and Anna Safont-Andreu
2.1.1 Introduction and Background 114
2.1.2 Research Areas 115
2.1.2.1 FMEA and FMEA Consistency Improvement 117
2.1.2.2 Causal Information Extracting from Free Text 118
2.1.2.3 Failure Analysis Process, Failure Analysis Reports, and Ontologies 121
2.1.2.4 Knowledge Representation 123
2.1.2.5 Refinement Algorithm 123
2.1.3 Reflections 125
2.1.4 Conclusion 126

2.2 Efficient Deep Learning Approach for Fault Detection in the Semiconductor Industry 131
Liliana Andrade, Thomas Baumela, Frédéric Pétrot, David Briand, Olivier Bichler and Marcello Coppola

2.2.1 Motivation: The Wafer Fault Classification Problem 132
2.2.2 Related Works 133
2.2.3 Target Platform Requirements 135
2.2.4 HW/SW System and Methodology 137
2.2.4.1 Industrial HW/SW System for On-Device Inference 137
2.2.4.2 Neural Network Building and Training Using N2D2 139
2.2.4.3 Neural Network Export and FPGA Implementation Used for Inference 142
2.2.5 Conclusion 143

2.3 Towards Fully Automated Verification of Semiconductor Technologies 147
Matthias Ludwig, Dinu Purice, Bernhard Lippmann, Ann-Christin Bette and Claus Lenz
2.3.1 Introduction 148
2.3.2 Background: Interpreting Semiconductor Technologies . . . 150
2.3.2.1 Methodology: The Integrated Analysis Process . . . 152
2.3.2.2 Example Analysis: From the Image to the Feature Extraction 156
2.3.3 Conclusion 157

2.4 Automated Anomaly Detection Through Assembly and Packaging Process 161
Saad Al-Baddai, Martin Juhrisch, Jan Papadoudis, Anna Renner, Lippmann Bernhard, Cristina De Luca, Fabian Haas and Wolfgang Schober
2.4.1 Introduction and Background 162
2.4.2 Dataset Description and Defect Types 164
2.4.3 Methodology 168
2.4.3.1 Anomaly Detection 169
2.4.3.2 Pseudo Anomaly Detection 169
2.4.3.3 Convolutional Neural Networks 172
2.4.4 Results and Discussion 174
2.4.5 Conclusion and Outlooks 175

3 AI Industrial Machinery 177

3.0 AI in Industrial Machinery 179

Giulio Urlini, Janis Arents and Antonio Latella
3.0.1 Introduction and Background 179
3.0.2 AI Developments and Future Trends in Industrial Machinery 181
3.0.3 AI-based Applications 183

3.1 AI-Powered Collision Avoidance Safety System for Industrial Woodworking Machinery 187
Francesco Conti, Fabrizio Indirli, Antonio Latella, Francesco Papariello, Giacomo Michele Puglia, Felice Tecce, Giulio Urlini and Marcello Zanghieri
3.1.1 Introduction and Background 188
3.1.2 Review of Industrial-level Methods for Edge DNNs 189
3.1.2.1 Compression Techniques 189
3.1.2.2 Popular Frameworks and Tools 190
3.1.3 Materials and Methods 191
3.1.3.1 System Architecture 191
3.1.3.2 Dataset Collection 194
3.1.3.3 Detection Methods 196
3.1.3.4 Continual Learning Setup 198
3.1.4 Experimental Results 199
3.1.4.1 Evaluation Metrics 199
3.1.4.2 Continual Learning Scenario 200
3.1.4.3 Robustness Against Quantization 201
3.1.4.4 Latency, Energy and Memory Footprint on STM32H743ZI 202
3.1.5 Conclusion 203

3.2 Construction of a Smart Vision-Guided Robot System for Manipulation in a Dynamic Environment 205
Janis Arents, Modris Greitans and Bernd Lesser
3.2.1 Introduction and Background 206
3.2.2 Challenges of Enabling Robots to “See” 206
3.2.2.1 Modularity 206
3.2.2.2 Operability 207
3.2.2.3 Computer Vision Algorithms 207
3.2.2.4 Validation of Algorithms 207
3.2.3 Requirements 208
3.2.4 Proposed Solution 210
3.2.4.1 Hardware and Interface Components 211

3.2.4.1.1 Robot Interface 211
3.2.4.1.2 Industrial Robot 211
3.2.4.1.3 3D Cameras 211
3.2.4.1.4 Deep Edge Device 211
3.2.4.2 Software Components 212
3.2.4.2.1 Computer Vision Algorithms 212
3.2.4.2.2 Synthetic Data Generation 212
3.2.4.2.3 Object 3D Reconstruction 213
3.2.4.2.4 Validation Framework 213
3.2.4.2.5 Robot Control 214
3.2.4.3 Hardware/Software Partitioning 214
3.2.5 Demonstrator Setup and Initial Results 215
3.2.6 Conclusion and Future Work 218

3.3 Radar-Based Human-Robot Interfaces 221
Hans Cappelle, Ali Gorji Daronkolaei, Ing Jyh Tsang, Björn Debaillie and Ilja Ocket
3.3.1 Introduction and Background 222
3.3.2 Gesture Recognition Using a Machine Learning Approach . 223
3.3.2.1 Concept and Experimental Setup 223
3.3.2.2 Inference Pipeline, Training Algorithm 224
3.3.2.3 Data Recording and Results 227
3.3.3 Gesture Recognition Using a Spiking Neural Network 228
3.3.3.1 Concept and Experimental Setup 229
3.3.3.2 Inference Pipeline, Training Algorithm 229
3.3.3.3 Data Recording and Results 232
3.3.3.4 Discussion 234
3.3.4 Proof of Concept Demonstration 234
3.3.5 Comparison and Conclusion 235

3.4 Touch Identification on Sensitive Robot Skin Using Time Domain Reflectometry and Machine Learning Methods 239
Pawel Kostka, Anja Winkler, Adnan Haidar, Muhammad Ghufran Khan, Rene Jäkel, Peter Winkler and Ralph Müller-Pfefferkorn
3.4.1 Introduction and Background 240
3.4.2 State of the Art 240
3.4.3 Problem Definition 241
3.4.4 Concepts and Methods 242
3.4.5 Proof of Concept of the Novel Sensor System 243

3.4.5.1 Experimental Acquisition of Training Data 243
3.4.5.2 Training Procedure 244
3.4.6 Results 245
3.4.7 Conclusions 246

4 AI Food and Beverage 249

4.0 AI in Food and Beverage Industry 251
Rachel Ouvinha de Oliveira, Marcello Coppola and Ovidiu Vermesan
4.0.1 Introduction and Background 251
4.0.2 AI Developments in Food and Beverage Industry 252
4.0.3 Future Trends for AI Technologies and Applications 254
4.0.4 AI-Based Applications 256

4.1 Innovative Vineyards Environmental Monitoring System Using Deep Edge AI 261
Marcello Coppola, Louis Noaille, Clément Pierlot, Rachel Ouvinha de Oliveira, Nathalie Gaveau, Marine Rondeau, Lucas Mohimont, Luiz Angelo Steffenel, Simone Sindaco and Tullio Salmon
4.1.1 Introduction 262
4.1.2 Related Work 263
4.1.3 Edge Intelligence 265
4.1.4 Communication Technology – LoRaWAN 267
4.1.5 Environmental Monitoring System 273
4.1.6 Conclusion 276

4.2 AI-Driven Yield Estimation Using an Autonomous Robot for Data Acquisition 279
Lucas Mohimont, Luiz Angelo Steffenel, Mathias Roesler, Nathalie Gaveau, Marine Rondeau, François Alin, Clément Pierlot, Rachel Ouvinha de Oliveira and Marcello Coppola
4.2.1 Introduction 280
4.2.2 Artificial Intelligence for Grape Detection 281
4.2.3 Towards an Automated Protocol for Yield Estimation 282
4.2.4 Assessing the Vine Vitality Using an Embarked LiDAR . . . 285
4.2.5 Conclusions 286

4.3 AI-Based Quality Control System at the Pressing Stages of the Champagne Production **289**
Lucas Mohimont, Mathias Roesler, Angelo Steffenel, Nathalie Gaveau, Marine Rondeau, François Alin, Clément Pierlot, Rachel Ouvinha de Oliveira, Marcello Coppola and Philipe Doré
4.3.1 Introduction and Background 290
4.3.2 Methodology 291
4.3.3 Results and Discussion 295
4.3.4 Conclusion 296

4.4 Optimisation of Soybean Manufacturing Process Using Real-time Artificial Intelligence of Things Technology **301**
Ovidiu Vermesan, Jøran Edell Martinsen, Anders Kristoffersen, Roy Bahr, Ronnie Otto Bellmann, Torgeir Hjertaker, John Breiland, Karl Andersen, Hans Erik Sand, Parsa Rahmanpour and David Lindberg
4.4.1 Introduction 302
4.4.2 Soybean Production Process Description 303
4.4.3 Overall Manufacturing System Architecture and Platform . . 306
4.4.4 Process Parameters Monitoring 307
4.4.5 Edge Processing and AI-based Framework for Real-Time Monitoring 310
4.4.6 Embedded Intelligent Vision and Multi-sensors Fusion Approach 311
4.4.6.1 Embedded Vision IIoT Systems Evaluation 314
4.4.7 Experimental Setup 316
4.4.7.1 Experimental Evaluation and Results 319
4.4.8 Summary and Future Work 321

4.5 AI and IIoT-based Predictive Maintenance System for Soybean Processing **327**
Ovidiu Vermesan, Roy Bahr, Ronnie Otto Bellmann, Jøran Edell Martinsen, Anders Kristoffersen, Torgeir Hjertaker, John Breiland, Karl Andersen, Hans Erik Sand, Parsa Rahmanpour and David Lindberg
4.5.1 Introduction 328
4.5.2 Maintenance Foundations 330
4.5.3 Principles of Predictive Maintenance 333
4.5.4 Soybean Production Process and Maintenance Policies . . . 334

4.5.4.1 Vibration Analysis 337
4.5.5 AI-based Predictive Maintenance Framework Methodology 339
4.5.6 Industrial Integrated System for Soybean Production Equipment 342
4.5.7 Experimental Set-up and Implementation 345
4.5.8 Summary and Future Work 350

5 AI Transportation 353

5.0 Applications of AI in Transportation Industry 355
Mathias Schneider, Matti Kutila and Alfred Höß
5.0.1 Introduction and Background 355
5.0.2 AI Developments in Transportation Industry 356
5.0.3 Future Trends for Applications in Transportation Industry . . 358
5.0.4 AI-Based Applications 360

5.1 AI-Based Vehicle Systems for Mobility-as-a-Service Application 363
Mikko Tarkiainen, Matti Kutila, Topi Miekkala, Sami Koskinen, Jokke Ruokolainen, Sami Dahlman and Jani Toiminen
5.1.1 Introduction and Background 364
5.1.2 AI-Based 3D Object Detection and Tracking for Automated Driving 364
5.1.2.1 Camera and LiDAR Sensor Data Fusion 365
5.1.2.2 Experiments and Results 366
5.1.2.3 Evaluation of the Algorithm and Vehicle Integration 368
5.1.3 Autonomous Control Prototyping in Simulated Environments 369
5.1.3.1 Reinforcement Learning Control for Mobile Vehicles 370
5.1.3.2 The Architecture – Immediate Actions Time-Horizon 371
5.1.4 Conclusion 372

5.2 Open Traffic Data for Mobility-as-a-Service Applications – Architecture and Challenges 375
Mathias Schneider, Mina Marmpena, Haris Zafeiris, Ruben Prokscha, Seifeddine Saadani, Nikolaos Evangeliou, George Bravos and Alfred Höß
5.2.1 Introduction and Background 375

5.2.2 Data Acquisition 376
5.2.2.1 Bus Traces 376
5.2.2.2 Traffic Cameras 377
5.2.2.3 Loop Detectors 378
5.2.3 Data Processing at the Edge 379
5.2.3.1 Object Detection 380
5.2.3.2 Bus GPS Trace 380
5.2.4 Data Processing in the Cloud 381
5.2.4.1 Data Quality Monitoring 381
5.2.4.2 Data Quality Observations 382
5.2.5 Conclusion 384

List of Contributors **387**

Index **393**

Preface

Artificial Intelligence for Digitising Industry – Applications

Industry 4.0 has revolutionised the manufacturing sector by integrating several technologies, including cloud computing, big data, and cyber-physical systems. The goal of Industry 4.0 is to make the manufacturing industry "smart" by integrating machines and equipment that can be monitored and controlled throughout the life cycle.

Industry 5.0 extends technological advances to further facilitate intelligent machine-machine and human-machine collaboration. The goal is to combine the speed, precision, repeatability, and replicability of the operation of machines with the vision, decision-making, and critical and cognitive thinking of human beings. Industry 5.0 can significantly increase the efficiency of manufacturing by extending the use of AI technologies to create a versatile connection between humans and machines, enabling constant monitoring and interaction. This collaboration will enhance the speed and the quality of production by assigning repetitive tasks to intelligent robots and other machines and fostering critical thinking by human beings. Industry 5.0 is characterized by the convergence of technologies and integrates the industrial internet of things (IIoT) with AI-based solutions and digital twins to connect physical and virtual manufacturing environments. This convergence makes possible physical and virtual simulations and operating environments in which models based on predictive analytics and managed intelligence enable faster, more accurate and precise, and more reliable decisions. This approach may also provide greener solutions than those of current industrial facilities: end-to-end, environmentally friendly manufacturing solutions with a minimal CO_2 footprint.

AI is transforming industrial environments. Edge-based AI technologies mitigate operational risk, improve the safety and efficiency of manufacturing, optimise processes, and form more reliable and sustainable manufacturing facilities. Adopting AI technology across industrial sectors enables more accurate prediction of anomalies and malfunctions, better management of

resource consumption, and optimising of manufacturing processes. Artificial intelligence is expected to significantly impact global manufacturing and industrial development. Integrated with other technologies - like intelligent sensors, IIoT, digital twins, edge computing, augmented reality, intelligent wireless and cellular connectivity - AI optimises production in real time and facilitates vertical, horizontal, and end-to-end integration.

AI industrial applications harness artificial intelligence to enhance efficiency and sustainability while expediting digital transformations. By applying AI, machine learning, and deep learning, manufacturers can advance operational efficiency, dynamically control, and adapt product lines, customise product designs, and plan technological developments.

This book explores the research, practical results, and exchange of ideas between the representatives of forty-one organisations participating in the AI4DI project to develop the technological community. The concepts presented herein reflect interaction with other European and international projects addressing the research, development, and deployment of AI, IIoT, edge computing, digital twins, and robotics in industrial environments to strengthen and sustain a dynamic AI technology ecosystem. These concepts and research results shed light on steps in the evolutionary transition to Industry 5.0. The focus is on five industries: the automotive, semiconductor, industrial machinery, food and beverage, and transportation industries.

The AI4DI project is part of the Electronic Components and Systems for European Leadership Joint Undertaking (ECSEL JU) programme, and the applications and technologies developed by the project partners support the digital transformation of the industry. They are aligned with the priorities of the new European partnership for Key Digital Technologies (KDT). KDT aims to provide innovative electronic components and systems, software, and smart integration to digital value chains, providing secure and trusted technologies tailored to the needs of user industries and citizens to support and reinforce Europe's potential to innovate. The goal is to develop and apply these technologies to address significant global challenges in mobility, health, energy, security, manufacturing, and digital communications.

The alignment between research, innovation, and industrial policies by using collaborative approaches in mastering the drivers of innovation contributes to and strengthens Europe's scientific and technological bases.

Editors Biography

Dr. Ovidiu Vermesan holds a PhD degree in microelectronics and a Master of International Business (MIB) degree. He is Chief Scientist at SINTEF Digital, Oslo, Norway. His research interests are in the area of smart systems integration, mixed-signal embedded electronics, analogue neural networks, artificial intelligence (AI) and cognitive communication systems. Dr. Vermesan received SINTEF's 2003 award for research excellence for his work on the implementation of a biometric sensor system. He is currently working on projects addressing nanoelectronics, integrated sensor/actuator systems, communication, cyber–physical systems (CPSs) and Industrial Internet of Things (IIoT), industrial AI with applications in green mobility, energy, autonomous systems, and smart cities. He has authored or co-authored over 85 technical articles and conference papers. He is actively involved in the activities of the Electronic Components and Systems for European Leadership Joint Undertaking (ECSEL JU) and involved in technical activities to define the priorities for the new European partnership for Key Digital Technologies (KDT). He has coordinated and managed various national, EU and other international projects related to smart sensor systems, integrated electronics, electromobility and intelligent autonomous systems such as E^3Car, POLLUX, CASTOR, IoE, MIRANDELA, IoF2020, AUTOPILOT, AutoDrive, ArchitectECA2030, AI4DI, AI4CSM. Dr. Vermesan actively participates in national, H2020 EU and other international initiatives by coordinating and managing various projects. He is the coordinator of the IoT European Research Cluster (IERC) and a member of the board of the Alliance for Internet of Things Innovation (AIOTI). He is currently the technical co-coordinator of the ECSEL Artificial Intelligence for Digitising Industry (AI4DI) project.

Reiner John received his degree in Electrical Engineering from the Fachhochschule des Saarlandes (Germany) in collaboration with the University of Metz / Perpignan (France). In 1984 he started his career with the Siemens Semiconductor Group in Munich, where he worked in

automatic test system development. In 1989 he was responsible for the consultation and application of embedded control development tools in the Siemens Automation Group. After joining Siemens Corporate Research and Development in 1991, Reiner John researched knowledge-based embedded systems within the Fuzzy group. Moving to Regensburg to work for the Siemens Automotive Division three years later, he developed concepts and implementations for a real-time operating system to manage and control the engine and transmission system. In 1996 joined Siemens Semiconductors, the later IPO of Infineon Technologies, where he served in several management positions in the Quality and Production Department of the company. In 2000, he further pursued his career in Taiwan, where he set up and managed the Infineon Silicon Foundry Taiwan Office as the Head of Department for seven years. At present, Reiner John is working in AVL List GmbH, Austria, where he oversees the coordination of public-funded R&D projects in the area of trustable AI for industrial and electromobility applications.

Dr. Cristina De Luca received the *Laurea* degree in statistical and economic science from, University of Padova (Italy) and the PhD degree in mathematics from, University of Klagenfurt (Austria), 2003. She joined Infineon Technologies Austria AG in 2002. She has worked on a wide range of R2R control applications for lithography, CMP and CVD semiconductor production processes and rollout in Regensburg (Germany), Kulim (Malaysia) and Villach (Austria) and contributed to research on R2R for the epitaxy process. Her research interests included advanced process control, automation and statistical data analysis, production automation, predictive maintenance, virtual metrology and industry4.0 automation, model predictive control for semiconductor manufacturing. She was an external professor for statistical quality control at the "Fachhochschule Kärnten" for 2004-2008 in cooperation with Infineon Technologies Austria AG. She is certified in Project Management since 2008. In 2009, she became project manager for European projects, first ENIAC and then ECSEL JU. She followed projects at different levels and contributed to their preparation, implementation, and coordination. To cite some of the projects: IMPROVE, EPPL, EPT300, SemI40, PRODUCTIVE4.0, Arrowhead Tools, AI4CSM. She is currently the coordinator of the ArchitectECA2030 project (Automotive) and AI4DI project (artificial intelligence). In 2019 she joined Infineon Technologies AG, Munich (Germany), where she is Senior Manager Funding Projects and Coordination.

Marcello Coppola is technical Director at STMicroelectronics. He has more than 25 years of industry experience with an extended network within the research community and major funding agencies with the primary focus on the development of break-through technologies. He is a technology innovator, with the ability to accurately predict technology trends. He is involved in many European research projects targeting Industrial IoT and IoT, cyber physical systems, Smart Agriculture, AI, Low power, Security, 5G, and design technologies for Multicore and Many-core System-on-Chip, with particular emphasis to architecture and network-on-chip. He has published more than 50 scientific publications, holds over 26 issued patents. He authored chapters in 12 edited print books, and he is one of the main authors of "Design of Cost-Efficient Interconnect Processing Units: Spidergon STNoC" book. Until 2018, he was part of IEEE Computing Now Journal Technical editorial board. He contributed to the security chapter of the Strategic Research Agenda (SRA) to set the scene on R&I on Embedded Intelligent Systems in Europe. He is serving under different roles numerous top international conferences and workshops. Graduated in Computer Science from the University of Pisa, Italy in 1992.

List of Figures

Figure 1.0.1 Further development of core functions in the processes of the automotive industry through AI capabilities. 4
Figure 1.0.2 AI4DI – Development of AI solutions addressing different approaches. 6
Figure 1.1.1 High level data flow diagram. 13
Figure 1.1.2 MPDSS architectural diagram. 15
Figure 1.1.3 Multilayer perceptron neural network results. . . . 17
Figure 1.1.4 Random forest results. 17
Figure 1.1.5 Gradient boosted tree results. 17
Figure 1.1.6 Decision tree results. 18
Figure 1.2.1 Retired electric vehicle (EV) battery packs prognosis in GWh per year. 22
Figure 1.2.2 Measurement and data pipeline and the feedback loop into the BMS. 26
Figure 1.2.3 Comparison of a standard 1D convolution and a causal convolution. 27
Figure 1.2.4 Dilated convolutions visualised. 28
Figure 1.2.5 A residual block. 29
Figure 1.2.6 Constant current discharge profiles of a LIB. 30
Figure 1.2.7 SOH prediction using a TCN with a reference measurement for the whole lifetime of a LIB. . . . 31
Figure 1.3.1 (a) Manually predefined geometric model of the seam, (b) the path editor and the robot jog in tecnomatix process simulate for manual trajectory programming, (c) multiple training environments running in parallel and optimizing the trajectory for a car door, and (d) the robot following the learned trajectory. 38

Figure 1.3.2 DRL agent in the MDP formalization for optimising a TCP trajectory by minimizing the distance to a predefined moving target. 40
Figure 1.4.1 Four levels of maturity in predictive maintenance. . 49
Figure 1.4.2 Diagnosis system architecture. 50
Figure 1.5.1 Simplified DC e-motor circuit. 66
Figure 1.5.2 Simplified DC e-motor diagnosis observations used for model-based diagnosis. 68
Figure 1.5.3 Simulation-based diagnosis algorithm description. . 74
Figure 1.5.4 Simplified DC e-motor diagnosis observation with simulation-based diagnosis. 75
Figure 1.5.5 Box plot 10-fold cross validation. 77
Figure 1.5.6 Normalized confusion matrix - model testing/ verification. 78
Figure 1.6.1 Winding equivalent for extended motor model. . . . 87
Figure 1.6.2 Simulated/Emulated faults in extended motor model / experimental motor. 88
Figure 1.6.3 Phase motor currents (healthy/with fault). 89
Figure 1.6.4 Motor currents in *dq* coordinates (healthy/with fault). 89
Figure 1.6.5 Filtered data in $\alpha\beta$ coordinates (healthy/with fault). 90
Figure 1.6.6 MNN structure used for inter-turn short circuit detection. 91
Figure 1.6.7 Shallow dense NN. 96
Figure 1.6.8 Mean square error of MLP during training phase. . 97
Figure 1.6.9 CM of the testing process of MLP. 98
Figure 2.1.1 AI-based knowledge management concept system for risk assessment and root cause analysis in semiconductor industry. 116
Figure 2.1.2 Manual corpora annotation example. 119
Figure 2.1.3 BERT-ConvE workflow. 124
Figure 2.2.1 Example of classified wafer maps. 137
Figure 2.2.2 The platform hardware architecture. 138
Figure 2.2.3 The platform software architecture. 138
Figure 2.2.4 System design and optimization process using N2D2. 139
Figure 2.2.5 Resource occupation (FF and LUT) for 42x42 wafer maps. 143

Figure 2.2.6 BRAM occupation for 42x42 wafer maps. 143
Figure 2.3.1 Abstraction layers in typical computing systems, ranging from software, over hardware design, to the underlying manufacturing technology. 149
Figure 2.3.2 Equally scaled scanning electron microscope images of semiconductor device cross-sections, showing a 28nm, a 40nm, a 65nm, and 150nm process node. . 150
Figure 2.3.3 Example of a cross-section image which shows already interpreted objects on the left side and part of the raw image on the right side. 151
Figure 2.3.4 Overall framework of the project. Domain knowledge and AI methods were the enabler for the use-cases that are facilitated through the automated SEM image interpretation. 152
Figure 2.3.5 A. Normalized histogram per class. B. Various zoom levels of the same image, magnified 4310, 8650, 20940 and 72180 times, respectively. 153
Figure 2.3.6 Exemplified overview of the segmentation pipeline. 154
Figure 2.3.7 Examples of segmented images with yellow illustrating metal components, and green illustrating VIAs. 155
Figure 2.3.8 Example features of different semiconductor technologies. The four dimensions were arbitrarily chosen from more than hundred possible attributes defining a semiconductor front-end technology. . . 156
Figure 2.3.9 Example SEM CS with the grey-scaled SEM image (left) and the segmented image (right). 157
Figure 2.4.1 Curves for one experiment. 165
Figure 2.4.2 Left: Distribution of average curves distance to other samples. Right: the results are showed in left by using Wasserstein distance outliers. 167
Figure 2.4.3 Outliergram, an example of feature for device current traces. Outliers are detected by inspecting the relationship between MEI and MBD. 167

Figure 2.4.4 Shows samples of OOI use case. Top left: particle in lower right corner (bottom side). Top right: particle in centre of image (top side). Bottom left: particle in centre of image (top side). Bottom right: scratch in upper area of heatsink (top side). Note that Bottom side is larger than top side. 168
Figure 2.4.5 Show an example of outliers (anomaly) cluster which is clearly inconsistent with the rest of the dataset. 170
Figure 2.4.6 Shows the process flow for the whole process including PDA and supervised learning applied on optical images. 171
Figure 2.4.7 Anomalies exist at the marked area. In this study, anomaly detection with pre-trained algorithm Resnet was conducted. 172
Figure 2.4.8 Shows an example of clustering anomalies units. Left: shows the clustering according to PAD. Middle: shows clustering after review process by an expert. Right: shows an example of defect units which recognized as a tick by PAD. As it shown, names represent the real names of classes of labelled images of ImageNet dataset. 172
Figure 2.4.9 Shows the process flow of wire bonding use case. . 173
Figure 3.1.1 System architecture schema. 192
Figure 3.1.2 MCU firmware state machine. 194
Figure 3.1.3 Visualizations of selected samples of the dataset. . 196
Figure 3.1.4 Validation metrics at different augmentation factors. 200
Figure 3.1.5 ROC curves at different augmentation factors. . . . 201
Figure 3.2.1 Architecture of the proposed solution. 210
Figure 3.2.2 Hardware/Software partitioning. 215
Figure 3.2.3 Renderings with different light power levels and camera orientations. 216
Figure 3.2.4 Object detection and pick and place operation. . . 216
Figure 3.2.5 Setup for object 3D reconstruction. 217
Figure 3.2.6 Scene including the plastic bottle- and the reconstructed metal can 3D models (middle) together with the and corresponding depth image (left) and segmentation masks (right). 218

Figure 3.3.1 Human-robot interaction concept using 60 GHz radar. 223
Figure 3.3.2 Simplified radar block diagram. 225
Figure 3.3.3 Signal processing pipeline to provide input to the feature generator. 226
Figure 3.3.4 Feature extraction for the random forest classifier. . 226
Figure 3.3.5 SNN-based gesture demonstrator. 229
Figure 3.3.6 Range vs Doppler (left), flattened range-Doppler vs frames (right). 230
Figure 3.3.7 LSM network with trainable output layer. 231
Figure 3.3.8 Confusion matrix for a 90%-10% learning (1620 samples) and test (180 samples) split of the dataset. 233
Figure 3.3.9 Proof of concept radar/robot interface block diagram. 235
Figure 3.4.1 Overview of the HMI principle: Basic components of the sensing, computing and control of touch events. 243
Figure 3.4.2 Network performance during the training: a) accuracy of the force identification and b) loss of the position identification. 245
Figure 4.1.1 Value chain for champagne production. 262
Figure 4.1.2 Comparative range, data rate, energy efficiency characteristics of communications technologies. . . 269
Figure 4.1.3 LoRaWAN architecture. 270
Figure 4.1.4 Operation of the different classes of a LoRaWAN. 271
Figure 4.2.1 (a) Camera attached to the vehicle, (b) defoliated vine. 283
Figure 4.2.2 Example of grape segmentation. On the left: the original image. On the right: the segmented image. 284
Figure 4.2.3 Correlation between visible grapes and the total number of grapes. 284
Figure 4.2.4 Biomass estimation and vine cane diameters obtained from Physiocap©. 285
Figure 4.2.5 (a) Vineyard robot Bakus©, (b) Physiocap© LIDAR installed on the robot. 286
Figure 4.3.1 Image acquisition at the wine pressing sites. 292
Figure 4.3.2 Example of the image processing steps. 293
Figure 4.3.3 STM board and TPU accelerator used in this work. 294

Figure 4.3.4 Output of the models. 296
Figure 4.4.1 Soybean production process flow. 305
Figure 4.4.2 Soybean products. 305
Figure 4.4.3 The electromagnetic spectrum - regions of interest in the context of NIR spectroscopy. 308
Figure 4.4.4 Hardware architectures under evaluation. 315
Figure 4.4.5 Experimental setup. 317
Figure 4.4.6 Training and inference workflow. 318
Figure 4.4.7 Boundary tracking for samples with impurities and split soybeans (left) and cleaned soybeans and crushed fractions (right). 321
Figure 4.4.8 Image detection segmentation and processing for samples of cleaned soybeans (left), soybeans with impurities (middle) and crushed fractions (right). . 322
Figure 4.4.9 Soybeans images processed by a binary image filter. 322
Figure 4.5.1 Maintenance types. Adapted from EN 13306 Standard. 330
Figure 4.5.2 Failure and maintenance timing. 332
Figure 4.5.3 Comparison of maintenance cost and frequency of maintenance. 332
Figure 4.5.4 Parameters monitored during equipment life-time operation. Adapted from STMicroelectronics. . . . 336
Figure 4.5.5 AI-based predictive maintenance framework. . . . 340
Figure 4.5.6 Industrial integrated system for equipment maintenance. 342
Figure 4.5.7 Soybean production predictive maintenance system demonstrator. 343
Figure 4.5.8 Overall architecture. 345
Figure 4.5.9 Experimental set-up detailed. 346
Figure 4.5.10 Spectrum analysis Wi-Fi 2.4GHz. 347
Figure 4.5.11 MQTT subscriber with Node-RED flow (vibration sensor per topic). 348
Figure 4.5.12 MQTT subscriber with Node-RED results (vibration data streaming). 348
Figure 4.5.13 STWIN SensorTile wireless industrial node. 349
Figure 4.5.14 Real-time data from the three-axis MEMS vibration sensor. 350
Figure 5.0.1 Transportation research areas in AI4DI. 360

Figure 5.1.1 3D object clustering using a point cloud and 2D detection boxes. 366
Figure 5.1.2 3D detections on camera view. 368
Figure 5.1.3 3D detections in LiDAR point cloud. 368
Figure 5.1.4 Last-mile pod driving scenario. 370
Figure 5.1.5 Pod sensor view. 371
Figure 5.1.6 Immediate action control research architecture. . . 372
Figure 5.2.1 Open data tampere: design for data acquisition. . . 377
Figure 5.2.2 Bus GPS trace, Line 32 Ranta-Tampella to TAYS Arvo. 378
Figure 5.2.3 Traffic cameras and their field of view in Tampere. . 378
Figure 5.2.4 Loop detectors for traffic amount measurements using DATEX II. 379
Figure 5.2.5 Architecture for data preparation at the edge. 379
Figure 5.2.6 Traffic object detection (left) and hourly car quantity (right). 380
Figure 5.2.7 Refinement of GPS bus traces: (a) Raw GPS [blue] and planned bus route [green] (b) Snapped bus route to OSM road network [black] (c) Partitioned route according to bus stop vicinity [yellow/purple] (d) Map-matching GPS trace [red]. 380
Figure 5.2.8 Data storage and processing in the cloud. The processed features can be retrieved and visualized as time-series and used for training AI prediction models. 382
Figure 5.2.9 Weekly data from left to right, top to bottom: traffic amount, congestion, vehicle counts derived from traffic-camera images, queue length and travel-time durations for the segmented bus routes (bus-links). 383

List of Tables

Table 1.5.1 Simplified DC motor component state description. . . 67
Table 1.5.2 Diagnosis results obtained using model-based diagnosis. 71
Table 1.6.1 CatE device evaluation of occupied memory. 99
Table 2.3.1 Framework of the cross-section interpretation with the respective sub-processes. 152
Table 2.3.2 Results of measured features of the VIAs. In the right column, the deviation between the automated and the manual is shown. 157
Table 3.1.1 Validation metrics at different training stages. 202
Table 3.1.2 Deployment metrics on three model variants. 202
Table 3.2.1 First object detection results, where the model has been trained on syntheticaly generated data. 216
Table 3.3.1 Chirp/Frame (a) and scene (b) radar parameters. . . . 227
Table 3.3.2 Recall and precision statistics of the machine learning based detector. 228
Table 3.3.3 Chirp/Frame (a) and scene (b) radar parameters. . . . 232
Table 3.3.4 Recall and precision statistics of the SNN based detector. 234
Table 4.1.1 LPWAN technologies comparison. 273
Table 4.3.1 Results obtained with the STM32MP157C-DK2 board. 296
Table 4.4.1 Normal parameters measured on whole or cracked soybeans before drying. 320
Table 4.4.2 Normal parameters measured on expanded soybean flakes after drying. 321
Table 5.1.1 Percentage of estimated object cluster means correctly placed inside KITTI ground truth boxes. 369
Table 5.1.2 The statistics for the agent performance. 371

Table 5.2.1 Statistics for open traffic data in tampere (2021-5-31). (*) Traffic cameras images are available starting from November 2019 but are not stored in the MongoDB. 377
Table 5.2.2 Errors-to-Data Ratio (EDR) for five categories of traffic data collected for the week of February 19 to 25, 2021. EDR is given as a percentage before and after the first step of data cleaning, which involves eliminating erroneous observations. 383

List of Abbreviations

AI	Artificial intelligence
AI4DI	Artificial intelligence for digitizing industry
ANN	Artificial neural networks
API	Application programming interface
APMF	Approximate progressive morphological filter
ASIC	Application-specific integrated circuit
BLE	Bluetooth low energy
BMS	Battery management system
BRAM	Block RAM
CatE	Computing-at-the-Edge
CBM	Condition-based maintenance
CM	Confusion matrices
CNN	Convolutional neural networks
CPU	Central processing unit
CUDA	Compute unified device architecture
DC	Direct current
DFB	Data fusion bus
DL	Deep learning
DMA	Direct memory access
DNN	Deep neural network
ECSEL JU	Electronic components and systems for european leadership - joint undertaking
EDR	Errors-to-Data Ratio
EIS	Electrochemical impedance spectroscopy
EMF	Electro motive force
EOL	End of life
ERP	Enterprise resource planning
ETL	Extract, transform, load
EV	Electric vehicle
FF	Flip-flop

FFT	Fast fourier transform
FoV	Field of view
FPGA	Field programmable gate arrays
GPS	Global positioning system
GPU	Graphics processing unit
HDMI	High-definition multimedia interface
HIL	Hardware-in-the-Loop
HLS	High level synthesis
HMI	Human machine interface, human machine interaction
HTTPS	Hypertext transfer protocol secure
HW	Hardware
IDE	Integrated development environment
IIoIT	Industrial internet of intelligent things
IIoT	Industrial internet of things
IMU	Inertial measurement unit
ITS	Intelligent transport systems
JSON	JavaScript object notation
LIB	Lithium-ion battery
Lidar	Light detection and ranging
LoRa	Long range (modulation technique)
LoRaWAN	LoRa wide area network
LTE	Long-term evolution (standard for wireless broadband)
LSQ	Learned step size quantization
LUT	Lookup table
MaaS	Mobility as a service
MBD	Model-based diagnosis
MIR	Multimedia information retrieval
ML	Machine learning
MLP	Multilayer perceptron
MPDSS	Material planning decision support system
MQTT	Message queuing telemetry transport
MSE	Mean squared error
N2D2	Neural network design & deployment
NFR	Non-functional requirements
NN	Neural networks
OEM	Original equipment manufacturer
ONNX	Open neural network exchange
OPC	Open platform communication

OPC UA	OPC unified architecture
OpenDDS	Open data distribution service
OS	Operating system
OSM	Open street map
PdM	Predictive maintenance
PLC	Programmable logic controllers
PMSM	Permanent magnet synchronous motor
PTQ	Post training quantization
PvM	Preventive maintenance
QAT	Quantization aware training
R2F	Run to failure
RAM	Random access memory
RDBMS	Relational database management system
RESTful	Representational state transfer, world wide web services that satisfy the REST constraints is described as RESTful.
RGB	Red-Green-Blue
RL	Reinforcement learning
SAT	Scale-adjusted training
SCADA	Supervisory control and data acquisition
SCM	Supply chain management
SIRI	Service interface for real time information
SoC	System-on-Chip
SOH	State of health
SW	Software
TCN	Temporal convolutional network
TensorRT	TensorRT is an SDK for high-performance deep learning inference from NVIDIA
TPU	Tensor processing unit
UI	User interface
YOLO	You only look once

Section 1

AI Automotive

1.0

AI Reshaping the Automotive Industry

Daniel Plorin

AUDI AG, Germany

Abstract

This introductory article opens the section by giving an overview of the state-of-the-art Artificial Intelligence (AI) technologies in automotive manufacturing and the current AI development in areas such as quality optimisation and analytics and predictive maintenance. It presents future potential and opportunities for AI in the automotive manufacturing sector, covering trends of using AI, industrial internet of things (IIoT) and robotics technologies in production and logistics optimisation, quality, and maintenance. Finally, the article introduces the five contributions to this section, highlighting the use of AI and IIoT in various scenarios in automotive manufacturing processes and challenges and technological advancements.

Keywords: artificial intelligence (AI), industrial internet of things (IIoT), automotive production, automotive logistic, optimisation, predictive maintenance.

1.0.1 Introduction and Background

The automotive industry and its production and logistics processes are a complex network that must implement high planning, operation, quality, and security processes. To handle this complexity and ensure high productivity, processes have been optimised for several decades of

technological developments. The development of highly networked systems and intelligently supported processes offer a new era in automation and process optimisation. With the more recent developments in AI, new opportunities are being established to implement productions more efficiently, humanely, and with higher quality. Finally, AI also helps to make the processes and production systems more flexible and modular because the intelligence of the control system is implemented deeper into the individual production processes [1].

1.0.2 AI Developments and Future Trends for AI Technologies in Automotive Industry

The weak and light AI process already supports planning and production. Bots can trigger demand mediation; camera systems ensure the quality of the products, or intelligent algorithms optimise the demand control for the line supply. With the consistent implementation of sensors at the plant level, their intelligent and fast networking via service bus systems and the analysis of relevant data and AI algorithms, a new quality of data transparency and value is created. The development of core functions in the automotive industry processes through AI capabilities is presented in Figure 1.0.1.

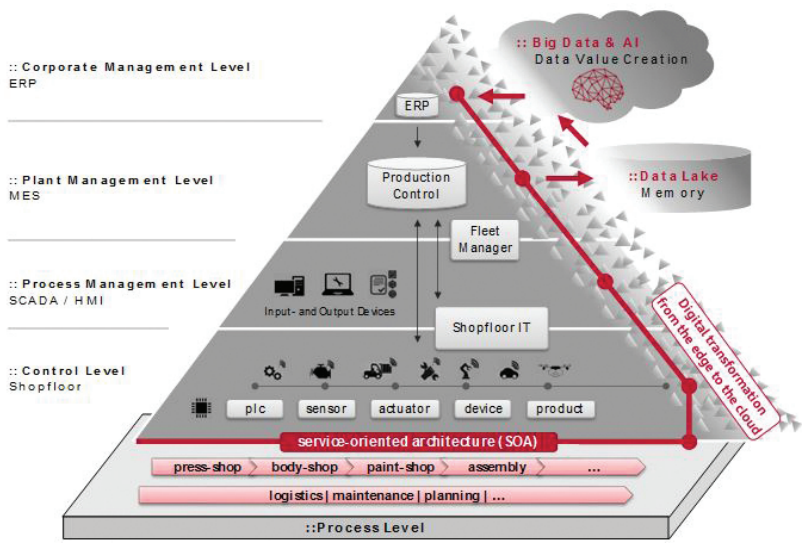


Figure 1.0.1 Further development of core functions in the processes of the automotive industry through AI capabilities [1].

This makes it possible to react to process and quality problems earlier or to automate the right decisions for the next process steps proactively. The use of new systems that automate routine tasks allows people to concentrate on the actual competencies of their function in the production system and being assisted by the intelligent system. In the past few years and during the AI4DI project, three essential AI topics have frequently emerged in the automotive industry: operational, prediction and detection intelligence. These topics are currently primarily developed and used in the productions and logistics of the automotive industry. A short description of these topics is given below.

- Operational intelligence relates to real-time dynamic process analytics that delivers visibility and insight into machines, process-generated data, streaming events, and business operations. These solutions run queries against streaming data feeds and event data to deliver analytic results as operational instructions. This provides the ability to make decisions and immediately act on these analytical insights through manual or automated actions.
- Prediction intelligence refers to solutions that use the knowledge gained from operational intelligence to determine the effects of real-time data using autonomous methods in forward-looking time series. Predictions are made regarding the behaviour of the process, or the product based on the learned historical comparisons.
- Detection intelligence addresses solutions that autonomously show deviations and abnormalities to defined as well as learned target states. These use various sensor technology options such as camera systems, sound sensors or other proximity sensors to detect objects, compare them with the system, and make statements about their condition. Sensing for object and status detection can be done by different senses – e.g., visually, acoustically, and sensitive.

Relevant AI technologies and methods for the implementation are suitable data clustering processes, neural networks, and intelligent sensing. Accordingly, ML and deep learning are essential areas of development, whereby physical objects play a significant role and process data that predict system behaviour. Therefore, pre-processing of the data with clustering algorithms (Gaussian mixture / K-Means) and time series prediction and anomaly detection with neural networks are primary fields of action to be further developed and implemented. However, the greatest challenge in the implementation is the secure integration in the clocked and complex processes, which must not be interrupted under any circumstances.

Implementing this in a wide-ranging legacy systems environment requires extensive protection and standardisation of the interfaces needed to source systems and sensor levels. Predictive maintenance and quality management represent an essential field of currently practical and efficient AI solutions.

1.0.3 AI-Based Applications

AI4DI project partners are developing AI and IIoT technologies with applications in different areas of the automotive industry sector, as illustrated in Figure 1.0.2.

The articles included in this section cover several demonstrators and actionable insights into how AI and IIoT are used in the automotive process and product applications. A brief overview of the articles in this section are presented in the following paragraphs.

The article "*AI for Inbound Logistics Optimisation in Automotive Industry*" addresses the challenges of the inbound supply process on production sites in the automotive industry (such as volatile supply chains) and argues for the use of AI- technology to manage its complexity and ensure the making of the right decisions in critical areas. A demonstrator use case of design and implementation of a Material Planning Decision Support System is presented, operating in the production site and attempting to optimise the complete inbound logistics process. The challenge is to fuse information dynamically from all sources into a single dataset and integrate it with user

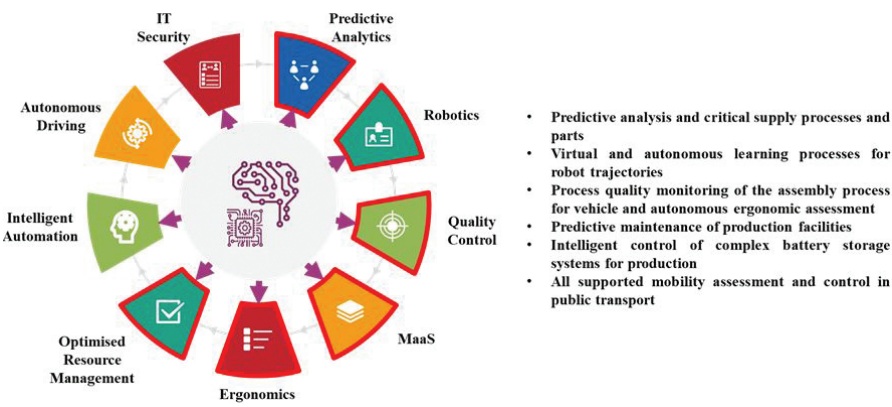


Figure 1.0.2 AI4DI – Development of AI solutions addressing different approaches [2].

requirements specified as short user journeys and label and integrate human experience-based knowledge for alternative courses of action.

The article "*State of Health Estimation using a Temporal Convolutional Network for an Efficient Use of Retired Electric Vehicle Batteries within Second-Life Applications*" addresses the need for state-of-health estimation algorithms to ensure safe and efficient usage of retired electric vehicle batteries (lithium-ion batteries) within second-life applications and proposes a data-driven approach, capable of overcoming the drawbacks of traditional less-robust estimation algorithms. The novel machine learning algorithm is based on a temporal convolution network and can deal with the highly nonlinear dependence on the changes of environment and working conditions during the operation. The network has been trained and tested with data gathered from commercial industry applications in energy storage, and the results show that it can predict the state of health with high accuracy.

The article "*Optimising Trajectories in Simulations with Deep Reinforcement Learning for Industrial Robots in Automotive Manufacturing*" presents a proof of concept for the applicability of reinforcement learning for industrial robotics by demonstrating a use case on automatic generation and optimisation of trajectories for applying the sealant material on car bodies (to prevent water intrusion and hence corrosion) using industrial manipulators. The Markov Decision Process (MDP) formalisation of an agent to reduce the amount of manual work involved in offline programming shows promising results. The methodology is yet to be verified and validated by comparing the agent solution with the hand-crafted trajectories and various degrees of involvement of human experts.

The article "*Foundations of Real Time Predictive Maintenance with Root Cause Analysis*" addresses the importance of autonomous systems to be equipped with a detection system to observe faulty behaviour in real time and predict failing operations. To avoid critical scenarios, finding the corresponding root cause is essential; hence, the focus of the article is on discussing the foundations behind diagnosis, i.e., the detection of failures and the identification of their root causes in the context of predictive maintenance. The article also explores the applicability of various diagnostic algorithms in real-time simulation environments, particularly artificial intelligence methods, including model-based diagnosis, machine learning and neural networks.

The article "*Real-Time Predictive Maintenance – Model-Based, Simulation-Based and Machine Learning Based Diagnosis*" addresses the importance of autonomous systems to be equipped with a detection system

to observe faulty behaviour in real time and outline its root cause. The underlying background is presented in a previous article "*Foundations of Real Time Predictive Maintenance with Root Cause Analysis*". This article explores the applicability of various diagnostic algorithms in real-time simulation environments. A simplified DC motor model with fault injection capability was developed, and three diagnostic methods (model-based, simulation-based and machine learning) were employed. The measurements have been compared, limitations identified and conclusions drawn. Preliminary results are promising, but more work is needed to address the challenges of efficiency and reliability of the diagnostic solutions.

The article "*Real-Time Predictive Maintenance – Artificial Neural Network Based Diagnosis*" addresses the importance of autonomous systems to be equipped with a detection system to observe faulty behaviour in real time and outline its root cause. The underlying background is presented in a previous article "*Foundations of Real Time Predictive Maintenance with Root Cause Analysis*". A second article "*Real-Time Predictive Maintenance – Model-Based, Simulation-Based and Machine Learning Based Diagnosis*" explores the applicability of three diagnostic methods in particular (model-based, simulation-based and machine learning) on a simplified DC motor model with fault injection capability. This article explores yet another method – artificial neural network (ANN) diagnostics – and its applicability with two use cases, one using an acausal six-phase e-motor model to simulate faults and the other for fault detection based on vibration measurements. Both simulation and measurement data are used for the ANN training. Two ANNs were designed, one for behaviour diagnosis and the other for the vibration sensor's microcontroller. Preliminary results are promising; the method can be applied to edge devices and can be implemented in real-time predictive maintenance applications.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] AI4DI Artificial Intelligence for Digitalizing Industry Project. (2020). "D1.2 - AI for digitizing industry road-mapping requirements ".

[2] AI4DI Project. (2020). "1st AI4DI Webinar: AI for Automotive Manufacturing and Mobility-as-a-Service." Available online at: https://ai4di.automotive.oth-aw.de/index.php/news/72-1st-ai4di-webinar-ai-for-automotive-manufacturing-and-mobility-as-a-service, 2020.

1.1

AI for Inbound Logistics Optimisation in Automotive Industry

Nikolaos Evangeliou[1], **George Stamatis**[1], **George Bravos**[1], **Daniel Plorin**[2] **and Dominik Stark**[2]

[1]Information Technology for Market Leadership, Greece
[2]Audi AG, Germany

Abstract

Artificial intelligence (AI) is playing an increasing role in the logistical aspects of a production site in an automotive industry. The pre-calculation of critical situations in the delivery of parts to the supplier network faces increasing disruptions which have an impact on delivery reliability. The planning and control processes are currently implemented by employees and consequently causes a lot of effort and sometimes incorrect decisions which are mostly based on the experiences of employees. The processing and learning AI component will assess the disruption risk caused by natural disasters such as earthquakes, hurricanes or through manmade political or social actions such as strikes and propose countermeasures and assure material availability. Automatic and permanent screening of external sources (newsfeed, weather forecast, traffic situation) determine potential influence of road conditions, natural disasters, strikes etc. on the expected reliability of material replenishment. Finally, the processing and learning component will assess different countermeasures based on a machine learning algorithm, which will be feed with data collected from the sensing component.

Keywords: artificial intelligence (AI), inbound logistics, optimisation, machine learning, real time analytics, data fusion bus, decision support system, scikit-learn.

1.1.1 Introduction and Background

The focus of demonstrator use case in AI4DI is the design and implementation of a Material Planning Decision Support System (MPDSS) that operates in an automotive production site and aims to optimize the complete inbound logistics process. Towards this direction, the work centres around the employment of advanced data-driven methods to collect and consolidate all relevant information and to use it for the identification of critical parts in the supply of AUDI's production lines.

This information refers to AUDI's internal data and information, AUDI's partner data and information (e.g., OEM's supplier's stock levels), public data information (e.g. weather conditions/forecast, road condition), as well as historical decisions and recommendations in similar situations. Finally, the MPDSS evaluates all possible measures for securing part supply via assessing all available data and collecting decisions and recommendations, and autonomously prioritises the applicable measures. Part autonomy is only delivered during decision on any critical part, as the user can always take the final decision of which countermeasure to apply based on given assessment parameters (e.g., cost, efficiency, CO_2 footprint, etc.). While the data collection (from local and publicly available sites) occurs at the edge, decision support offered by the MPDSS occurs at the cloud side. Training and inference of the ML algorithms happens centrally in the cloud. The AI methodology to follow is supervised training, with the main challenges being the learning prediction.

1.1.2 Requirements – User Journeys

The user requirements of the MPDSS will be presented below as short user journeys.

Data Collection and Consolidation: The MPDSS should make use of all available information to identify critical parts, while minimising the necessary actions for the manual collection and consolidation of data.

This is achieved by collecting (i) AUDI's internal data and information; (ii) AUDI's partner data and information; (iii) Public data and information (e.g. weather information, political situations affecting road conditions, etc); and (iv) decisions and recommendations.

Identification of critical parts: The MPDSS should show only those parts that are critical enough to cause a supply bottleneck in the production line.

To achieve this, the system should provide the best possible assessment of criticality by (i) categorising parts and determine critical ones; (ii) prioritising them according to supply capability (how probably it is to obtain this part on time); (iii) visualising critical parts and relevant background information (based on historical data).

Recommendation of measures: The MPDSS should leverage optimisation algorithms to prioritise the different applicable measures for securing part supply and recommend the best-suited measure, taking into consideration certain parameters (e.g., cost, effectiveness, CO_2 footprint).

Autonomous decision making: The MPDSS should autonomously decide which measures are applicable based on given conditions that can be defined by the user either in advance or after the user visualises the suggested countermeasures (partly autonomous decision making). This feature gives a flexible definition of conditions.

Continuous improvement: The user should be able to rate the recommendations given by the MPDSS and this rating should be used to improve the AI routines of the system in the future. This is achieved by comparing user's decision with MPDSS best-fit recommendation (when part autonomous operation).

1.1.3 Data Flow Principles and Architecture of the MPDSS

In this envisioned MPDSS, the data flow is depicted [1] in the following data flow diagram. A streaming platform collects information from AUDI internal data sources (such as warehouse databases) and external data streams (such as weather APIs, traffic condition APIs etc), and all information is fused

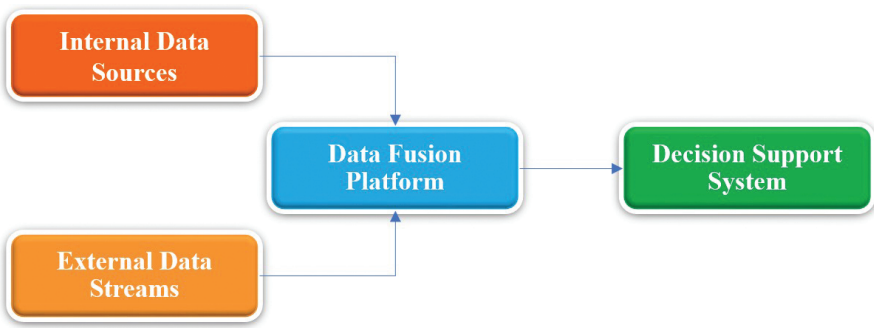


Figure 1.1.1 High level data flow diagram.

dynamically in a single dataset. To account for emergency situations such as traffic conditions or natural disasters, this dataset should be updated and queried continuously, so that decision support and alerting is provided in a real-time manner.

The Data Fusion Bus (DFB) is well-suited to account for the need of providing real time Machine Learning analytics. Brief reference to DFB and its rationale has been made below. DFB enables organizations in developing, deploying, operating, and managing a big data environment with emphasis on real-time applications. It combines the features and capabilities of several big data applications and utilities within a single platform.

The key capabilities of DFB [2] are:

- Real-time monitoring and event-processing, semantic fusion of events not coinciding in time.
- Data aggregation from heterogeneous data sources and data stores.
- Real time analytics, offering ready to use Machine Learning algorithms for classification, clustering, regression, anomaly detection.
- An extendable and highly customizable Interface REST API (and web app) for configuring analytics, manipulation, and filtering. It also includes functionality for managing the platform.

The technical architecture of MPDSS [3] will be a combination of well-known opensource tools and proprietary modules. ITML will leverage its in-house developed Data Fusion Bus, as depicted in Figure 1.1.2 below.

The main building blocks of the architecture are:

- **Support for multiple data streams and data stores**: Readily available interfaces are in place that allow for data acquisition for all well-established Relational Database Management System (RDBMSs), data streams (using MQTT), NoSQL databases, shared filesystems (HDFS Hadoop [4]. This functionality is supported by Kafka [5].
- **Data Fusion Bus**, comprised of the following sub-modules: (i) The Streaming Core of the platform is Apache Kafka. It relies on Kafka ‘s distributed messaging system to provide high fault-tolerance (Resiliency to node failures and support of automatic recovery) and elasticity - high scalability; (ii) Internal Store and Search Engine: When persistence of data within the platform is required, the Elastic stack (Elasticsearch and Logstash) is utilized. Data may flow either through Kafka connectors (usually in cases of stream data) or may be directly imported to Elasticsearch [6]. Elasticsearch also provides provide high fault-tolerance and scalability; and (iii) Identity management, authentication,

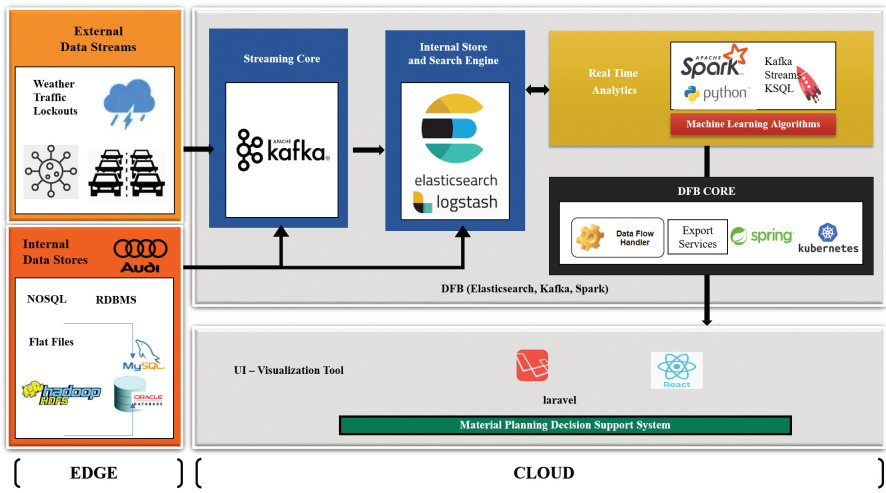


Figure 1.1.2 MPDSS architectural diagram.

authorization and accounting mechanisms that enhance the security of the platform. Moreover, the security mechanism includes dataset encryption and anonymization.

- **DFB Analytics Engine** supports batch processing and stream processing with Apache Spark [7], Kafka Streams & KSQL, Spark Streaming and python scikit-learn [8]. DFB can be used to perform supervised (classification and regression with algorithms such as RandomForest or neural networks) and unsupervised Machine Learning algorithms (e.g Clustering with Kmeans).
- **DFB Core** is responsible for providing business logic and managing all the data flows. It is a custom REST API (based on Java Spring). It exposes a configurable set of web services for providing Decision Support to external systems and managing/monitoring the whole platform.

1.1.4 Preliminary Analysis of Data and Dataset

Advanced data analysis will be applied in a dataset to detect critical parts (using a binary classification algorithm that return "1" when a part is critical and "0" when it is not), then assess and recommend countermeasures again based on calculations from input data, and finally perform decision making and take into consideration the final decision of the user for continuous

improvement. There is also consideration into extending the classification of parts into three classes: non-critical, critical and very critical part.

A preliminary analysis of that dataset to explore possible correlations among the various fields and the suitability of different machine learning algorithms has been performed. While this dataset is considered too small for reliable outcomes, some initial results and the methodology used is presented below.

1.1.4.1 Data Pre-processing and Visualisation

Data Understanding using descriptive statistics: Quantitative summary of raw data received as input using measures of central tendency and measures of variability. This process allows the identification of distinct values for each field and the distribution for the numeric values.

Handling missing values: If missing values are not handled properly an inaccurate inference about data might be drawn. Columns which had no values were removed.

Feature selection: Processing of input variables to select features with optimal contribution to the target variable. Removing redundant data helps in reducing data noise and improves model accuracy. This step is achieved with visualisation tools aiming at the detection of highly correlated variables.

Continuous vs categorical feature detection: Automatically identify which features are categorical and convert original values to category indices. This process improves the efficiency of the machine learning algorithms.

Categorical feature encoding: Transforming categorical variables to numbers by mapping each category to a binary vector denoting the presence or absence of the feature. This process also improves the efficiency of the machine learning algorithms.

1.1.4.2 Classification Models

Four different machine learning algorithms (Multilayer Perceptron Neural Network, Random Forest, Gradient Boosted Tree and Decision Tree) were used on the sample data. In every case 70% of the dataset was used for the training of the algorithm and 30% for testing. The model using the **multilayer perceptron neural network** had the higher accuracy. Other algorithms can be used in the future if needed as well. The results are presented below [9].

PerceptronTrainer

Confusion Matrix

	Non Critical	Critical	
Actual Value 0	91	6	**Sum row 97**
Actual Value 1	3	75	**Sum row 78**
	Sum column 94	**Sum Column 81**	

Figure 1.1.3 Multilayer perceptron neural network results.

RandomForestTrainer

Confusion Matrix

	Non Critical	Critical	
Actual Value 0	88	11	**Sum row 99**
Actual Value 1	8	63	**Sum row 71**
	Sum column 96	**Sum column 74**	

Figure 1.1.4 Random forest results.

Multilayer Perceptron Neural Network achieved an accuracy score of over 90% based on cross-validation results (Figure 1.1.3 and Figure 1.1.4). The confusion matrix that summarizes the proportion of correct vs incorrect classifications is as follows:

GBTTrainer

Confusion Matrix

	Non Critical	Critical	
Actual Value 0	81	9	**Sum row 90**
Actual Value 1	11	59	**Sum row 70**
	Sum column 92	**Sum column 68**	

Figure 1.1.5 Gradient boosted tree results.

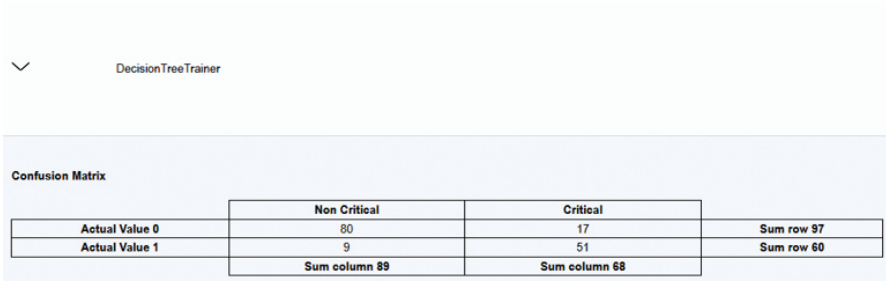


Figure 1.1.6 Decision tree results.

Gradient Boosted Tree achieved an accuracy score based on cross-validation results (Figure 1.1.5). The confusion matrix that summarizes the proportion of correct vs incorrect classifications is as follows:

Decision Tree achieved an accuracy score based on cross-validation results (Figure 1.1.6). The confusion matrix that summarizes the proportion of correct vs incorrect classifications is as follows:

1.1.5 Conclusion

The inbound supply process in the automotive industry is a complex structure of the availability of required material, calculable risks and unpredictable events which have a direct influence on the entire value added in the production of the vehicles but also on the supporting value creation processes. Dealing with these events and making the right decisions poses major challenges for every automotive manufacturer and supplier, especially since the supply chains in an international network of manufacturing units are very volatile. In this big data environment, artificial intelligence offers excellent technology to make this complexity manageable and to make the right decisions in critical areas. With the MPDSS system and the underlying architecture, the first major progress can be achieved at an early stage of implementation. The greatest challenge here is the integration of the right data with all the requirements described as well as the labelling and integration of human experience-based knowledge for alternative courses of action.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] AI4DI project, Deliverable D2.4: Report on hybrid intelligent system and sub-system level modelling and simulation, Public Report.
[2] Data Fusion Bus, Available online at: https://itml.gr/data-fusion-platform
[3] AI4DI project, Deliverable D2.3: Report on HW/SW partitioning and sub-system level key architecture designs, Public Report.
[4] APACHE KAFKA, Available online at: https://kafka.apache.org/
[5] APACHE hadoop, Available online at: https://hadoop.apache.org/
[6] Elastic, Available online at: https://www.elastic.co/
[7] APACHE Spark, Available online at: https://spark.apache.org/
[8] Scikit-learn, Available online at: https://scikit-learn.org/stable/
[9] AI4DI project, Deliverable D2.5: Report on IIoT/AI enabled system level platforms, Public Report.

1.2

State of Health Estimation using a Temporal Convolutional Network for an Efficient Use of Retired Electric Vehicle Batteries within Second-Life Applications

Steffen Bockrath, Stefan Waldhör, Harald Ludwig, Vincent Lorentz

Fraunhofer Institute for Integrated Systems and Device Technology IISB, Germany

Abstract

This paper presents an accurate state of health (SOH) estimation algorithm using a temporal convolutional neural network (TCN) for lithium-ion batteries (LIB). With its self-learning ability, this data-driven approach can model the highly non-linear behaviour of LIB due to changes of environment and working conditions all along the battery lifetime. The precise SOH predictions of the TCN are especially needed to ensure a safe and efficient usage of retired electric vehicle batteries within second-life applications. The provided network is trained and tested with data gathered from commercial industry applications in the domain of energy storage. It is shown, that even for dynamic load profiles, the TCN achieves a mean squared error (MSE) of less than 1.0 %. Using this approach, the uncertainty of the heterogeneous performances and characteristics of retired electric vehicle batteries can be drastically reduced.

Keywords: lithium-ion battery, battery management system, state of health, second-life, artificial intelligence, temporal convolutional neural network, retired electric vehicle battery, stationary battery system.

1.2.1 Retired Electric Vehicle Batteries for Second-Life Applications

According to the Paris Agreement signed in 2016, over 190 countries agreed to reduce their greenhouse gas emissions by at least 40 % until 2030 compared to 1990. To attain this objective, the usage of fossil fuels has to be drastically reduced, which is one reason why renewable energies are coming to the fore. For efficient and sustainable utilization of these intermittent energy sources, reliable and safe energy storage is an indispensable prerequisite. The lithium-ion battery (LIB) technology, with its high conversion efficiency, provides an efficient solution as dynamic energy storage. Thus, lithium-ion battery technology is a promising solution for sustainable transportation if the required energy comes from renewable energy resources. However, due to demanding operating conditions, an electric vehicle (EV) battery loses capacity and power over its lifetime. Typically, after 8 to 10 years of service, those batteries are retired due to capacity fade and power output that fails to meet range and performance requirements of modern EVs. In general, a retired battery of an EV can still provide 60-70 % of its initial energy storage capability at the end of its vehicular life. In Figure 1.2.1, three prognoses of retired EV battery packs are shown.

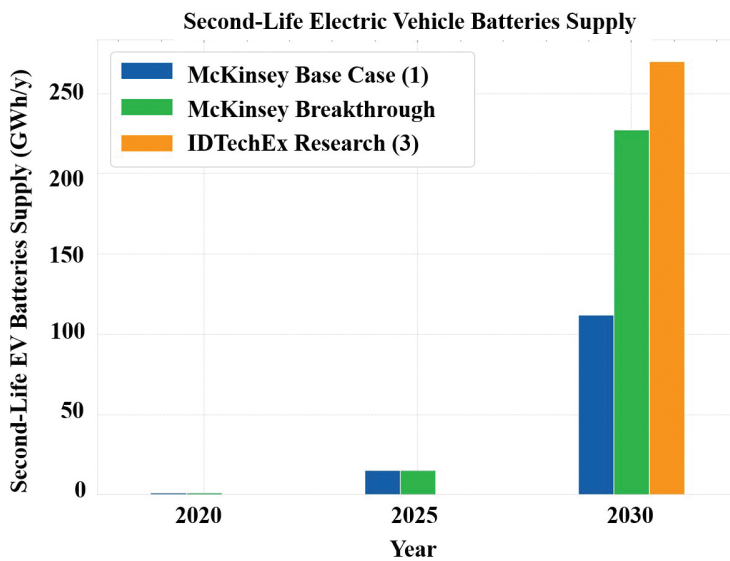


Figure 1.2.1 Retired electric vehicle (EV) battery packs prognosis in GWh per year [2][3].

According to IDTechEx research [2], by 2030 there will be over 6 million battery packs retiring from EVs per year. Since those batteries could contain 60-70 % of their initial energy storage capability, they can be further utilized in less-demanding applications such as stationary energy storage. However, there are still many challenges that have to be tackled in order to ensure a safe and economically valuable usage of retired EV batteries in second-life applications. In the following, four main challenges of second-life applications are described according to [1].

First, the competitiveness of second-life batteries with new generations of batteries is a big challenge. It is likely that when the worn-out EV batteries that are taken out of the car and could be used for second-life applications, there will be new generations of batteries with better quality and performance and at a lower price. Thus, the economical exploitation of second-life batteries will become even more challenging, while the CO_2 footprint of the battery manufacturing industry will have to be considered globally over the whole life-cycle. As a result, the cost competitiveness and the attractiveness of second-life batteries would be decreased, but the impact on the environment could become worse.

In addition, different regulations are a critical point. Second-life batteries are still not defined in the regulation in many countries. Since batteries are considered hazardous goods, the transportation requires special care and is, therefore, more expensive. Moreover, since the regulations of the electricity market in most countries are not fully open and transparent, the regulations of the battery storage for the energy market are not clear.

Another challenge is the design of the battery packs themselves. Battery packs are designed to optimally fulfill the requirements of the primary application they are used in, and that often requires technical and economical optimizations for the highest competitiveness on the market. Unfortunately, these optimizations are not optimal for repurposing the battery pack. Now, the vehicle manufacturers design and optimize the batteries only for being used in the vehicle, over 7-8 years. Battery repurposing cost is significantly affected by how the battery packs were initially designed. If components inside the battery pack are not compatible with stationary storage applications, additional costs for battery repurposing will result. For example, a car is designed for 300,000 km over 15 years and 10,000 h operation, while a stationary application is mostly requiring electronics supporting 24 h operation during 7 days in a week. A systemic design thinking that integrates the process of second-life repurposing into the initial battery pack design would simplify the repurposing procedure and reduce the repurposing costs

but would add with certitude costs on the implementation for the primary-life. Based on these considerations, it results that a regulatory way may be at least one enabler for second-life battery applications since the competitiveness in the primary application could be reduced significantly.

Finally, the spread and the uncertainty in the remaining battery lifetime and performance degradation in various energy storage applications is another main challenge. The lifetime and degradation of second-life batteries are quite heterogeneous and depending on a whole set of parameters (e.g., temperature, depth of discharge, current rates, mechanical vibrations), depending on how they were used in EVs and how they are going to be used during their second-life within stationary applications. Since each battery shows a different aging behavior depending on its chemistry (including the types and quantities of additives to the electrolyte), on its construction or its historical operating conditions within the vehicles, it is challenging to predict systematically the ageing behavior of the batteries during their second-life. A suitable evaluation and prediction of the second-life battery performance is essential for a safe and economically viable usage of retired EV batteries.

In the AI4DI project and with the demonstrator "autonomous reconfigurable battery system", the challenge of uncertain second-life battery performance is tackled. Retired batteries (i.e., modules of packs) with very heterogeneous performances and characteristics are combined within a single battery system [18]. For this purpose, it is essential to determine the state parameters, like the state of health (SOH) of each battery accurately.

In this paper, it is shown how a temporal convolutional network (TCN) can be used for accurately predicting the state of health of a lithium-ion battery. In the following section, the fundamentals of SOH of lithium-ion batteries are recapitulated. Subsequently, the data measurement using the open-source battery management system foxBMS is covered. Afterward the TCN is introduced, and its building blocks are explained. In the subsequent chapter, the results of the SOH prediction using a TCN are presented and analyzed. Finally, the conclusion and outline of this work are given.

1.2.2 State of Health of Lithium-Ion Batteries

The performance of lithium-ion batteries is decreasing with time (i.e., calendric aging) and with utilization (i.e., cyclic aging). The two most characteristic parameters for measuring the current performance capabilities are the total battery capacity and the internal series resistance of the battery. The

capacity is decreasing, and the internal resistance is increasing due to unwanted side reactions and structural deterioration. As a result, an aged LIB can store less energy and deliver less power compared to a new LIB of the same type. The current aging status, also known as the state of health (SOH), is defined from 0 % to 100 %, where the SOH of a new LIB is defined to be 100 %.

This work focuses on the SOH derived from the energy capacity fade of a LIB as stated in Equation 1.2.1.

$$\mathrm{SOH_Q_i} = \frac{\mathrm{Q_i}}{\mathrm{Q_0}} \tag{1.2.1}$$

where $\mathrm{SOH_Q_i}$ is the SOH after the *i*-th cycle, $\mathrm{Q_i}$ is the capacity after the *i*th cycle and $\mathrm{Q_0}$ is the initial capacity at the lithium-ion battery's start of life. The capacities $\mathrm{Q_0}$ and $\mathrm{Q_i}$ are determined using Coulomb Counting. The Coulomb Counting approach is a straightforward method that uses current integration. The capacity is computed by integrating the charge or discharge current over time. In order to realize the capacity computation and thus the SOH determination, the battery management system has to be introduced and how it is used for measuring battery usage data like the voltage, temperature, and current.

1.2.3 Data Measurement Using the Open-Source Battery Management System foxBMS

The battery management system (BMS) consists of the electronics and the embedded software to fulfil all tasks that ensure a safe, reliable and application specific optimal operation of the battery system. This includes measurement of all battery cell voltages in the battery pack, a use-case specific number of cell temperatures per battery module, and the battery pack current. Furthermore, additional measurement data can be used as input to ensure an optimal battery system operation, like e.g., pressure sensors or electrochemical impedance spectroscopy (EIS) measurements, with or without an additional sensor [8]. The BMS switches the electric power contactors of the battery system to ensure that the battery cells are not used outside of their safe operating limits. Pyro-fuses or electromagnetic fuses are used as last resort safety elements in the battery system to interrupt the battery current in case of a strong overcurrent or a short circuit. While the pyro-fuses are mostly actively controlled by the BMS, the electromagnetic fuses are triggered automatically by an overcurrent to ensure a shutdown if the battery is exposed to hazardous conditions.

In order to run a battery system in an application specific optimal operating window, battery models, ranging from cell to module and up to system models (e.g., equivalent circuit, physical- or heuristics-based ones) need to calculate battery state parameters, e.g., the previously mentioned SOH. The battery system must be able to perform the required model calculations and predict their output in real-time. Based on its own acquired measurement data, the implement application logic and the inputs from the higher-level control unit, the BMS can safely and optimal control the battery usage in the application.

To empower our partners and customers to build beyond state-of-the-art battery management systems, the Fraunhofer IISB has established a free, open and flexible Battery Management System R&D platform called foxBMS in 2016 [4, 5, 9]. In 2020, Fraunhofer IISB publicly announced that there is going to be the second generation of foxBMS [6] with enhanced safety and more data generation and connectivity possibilities, which then became available in 2021 [7]. foxBMS is a research and develop platform, which allows to rapidly development prototypes in the field of battery applications. These prototypes start from the simple implementation of drivers for innovative sensors, testing and benchmarking modern battery models on an embedded platform up to developing a full-customized battery system for a preproduction system, but also as starting point for advanced mobile and stationary battery powered products.

Whether battery usage data is generated in research projects, e.g., for an academic purpose for creating the most sophisticated and accurate models, or in a product, e.g., to increase the lifetime before end-of-life (EOL), it is mandatory to make the acquired measurement data available outside of the embedded system to learn from it, and feedback the gained knowledge. Figure 1.2.2 shows the information flow of the data pipeline.

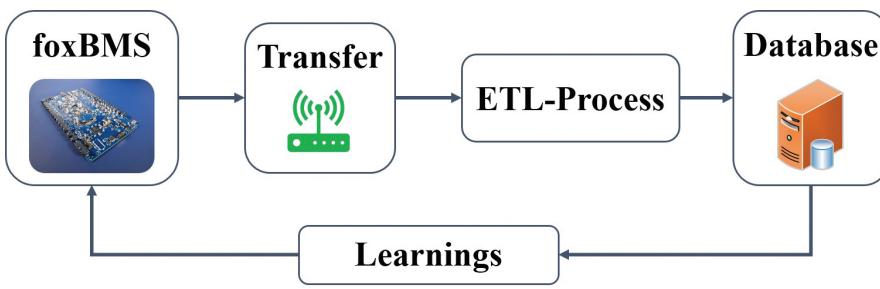


Figure 1.2.2 Measurement and data pipeline and the feedback loop into the BMS.

First, the measurement data is acquired by the BMS in the application. This raw data is then transferred and further processed in an ETL-process (*Extract, Transform, Load*) and stored in a database. This ETL-process is necessary, since the logged data stream in such a low-level system (e.g., CAN, Ethernet) cannot be directly used for modelling activities. Therefore, the output is converted and pre-processed in a data format that is reasonable for data analysis and model training.

After covering the fundamentals of the SOH and describing the data measurement using the open-source battery management system foxBMS, the data-driven approach for SOH prediction is introduced next.

1.2.4 Temporal Convolutional Neural Network for State of Health Prediction

Since the success of Deepmind's WaveNet [12], a so-called deep neural network (DNN), similar but simplified networks have been successfully applied to more and more problems. This architecture family was first named temporal convolutional network (TCN) by Lea et al. [13]. A TCN can be differentiated by the following characteristics:

1) Causal convolutions are used to prevent the "leakage" of information from the future to the past.
2) The output sequence has the same length as the input sequence.

1.2.4.1 Causal Convolutions and Receptive Field

In contrast to 1D convolutions, the TCN uses *causal* convolutions. These are convolutions that only consider the [t - k+ 1, t] data at time t, where k is the kernel size. To ensure that the output sequence will have the same length as the input sequence, (k - 1) data points have to be padded into the "past".

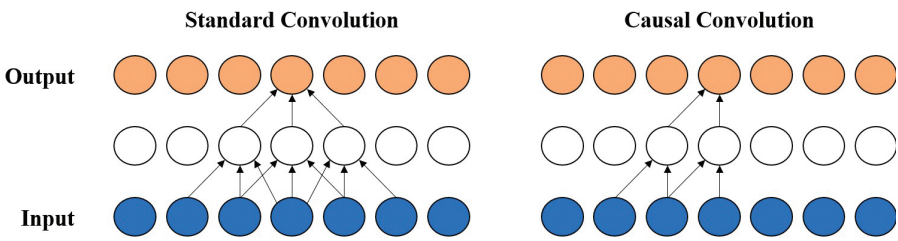


Figure 1.2.3 Comparison of a standard 1D convolution and a causal convolution.

The receptive field is the length of the input sequence of the TCN. When creating the model, the kernel size k and the receptive field R have to be specified. These two parameters then determine how many layers l are needed as it can be seen in Equation 1.2.2.

$$R = 2^l (k - 1) \tag{1.2.2}$$

1.2.4.2 Dilated Convolutions

The use of causal convolutions has the consequence that the network becomes deeper and deeper as the receptive field increases. As a result, not only the training duration but also the memory requirement increases. To counteract this problem, dilated convolutions are used. A dilation factor d indicates whether every data point is used (d = 1), only every second data point (d = 2), and so on. A too large dilation factor creates sparsity in the data, while a too small dilation factor does not solve the problems mentioned above. Therefore, the dilation factor is increased by a factor of two with each layer [12].

1.2.4.3 Residual Block

An additional method that ensures the stability and performance of deep networks is skip connections [14]. A skip connection does nothing more than adding the input to the output. In order to have a skip connection in a meaningful and useful way, a so-called residual block can be implemented. A residual block represents a layer of the network and ensures that local regeneration of a LIB can be captured [15] as it can be seen in Figure 1.2.5 at week 20.

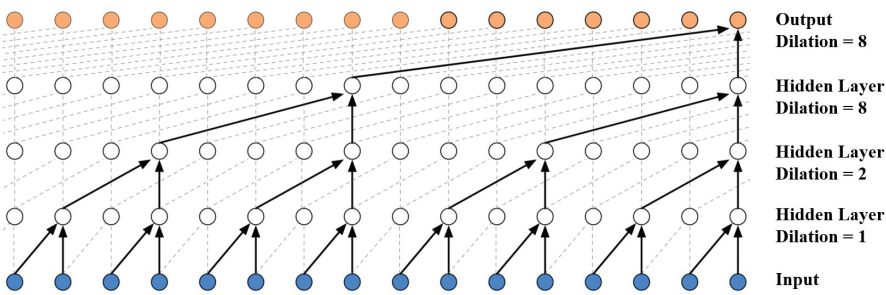


Figure 1.2.4 Dilated convolutions visualised [12].

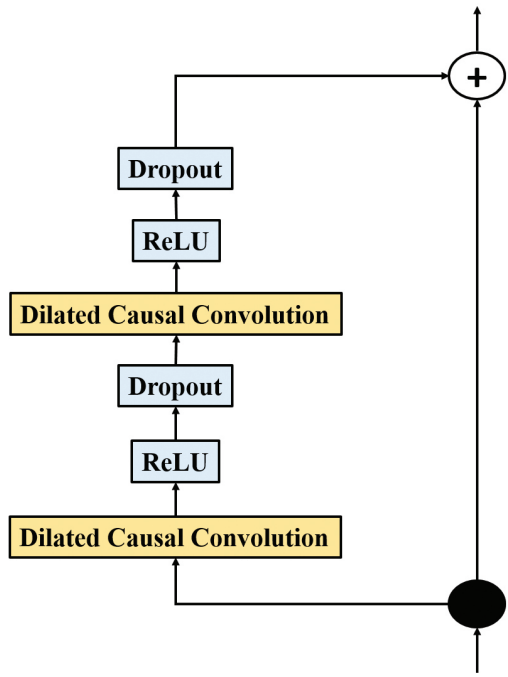


Figure 1.2.5 A residual block [12].

1.2.5 Results

In this chapter, it is shown how well the TCN performs the SOH prediction for a LIB. The TCN model was written in Python 3.8 and PyTorch 1.8. The training and experiments were run on a desktop PC with the following configuration: the CPU is AMD Ryzen 7 3700X, and the GPU NVIDIA GeForce RTX 3070. The TCN was trained on the public randomized battery usage data set from NASA Prognostics Center of Excellence [17]. This data set contains the data of 28 18650 lithium-cobalt-oxide cells with an initial capacity of 2.1 Ah. The battery cells are divided into seven groups of four cells each. Every group of cells was cycled with a different profile. A reference charge and discharge were carried out at regular intervals. Since the data set only contains time, current, voltage, and temperature the capacity and the resulting SOH were computed for each reference discharge. Then the calculated capacity and SOH were used to train the model. As input, the last 100 capacity values of a reference discharge were used and as output, the corresponding SOH was predicted. 22 cells were randomly picked as training

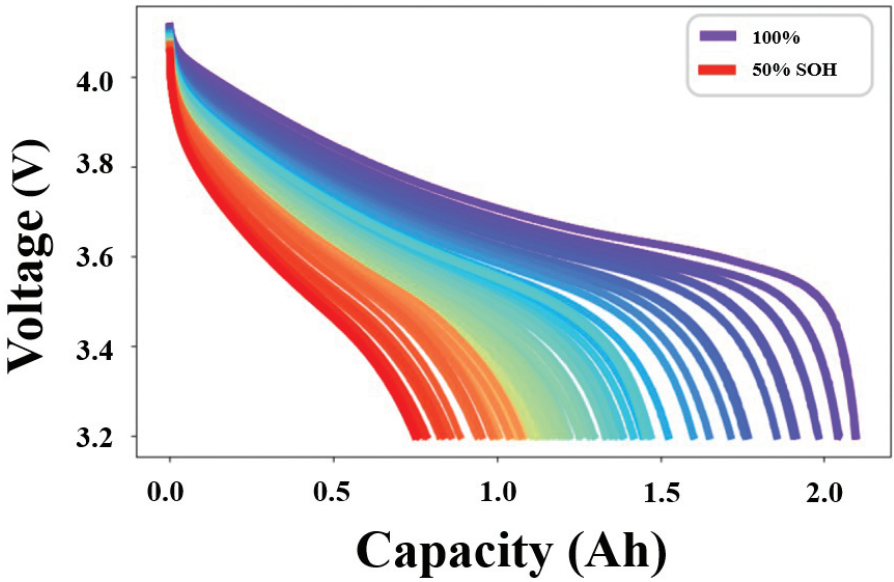


Figure 1.2.6 Constant current discharge profiles of a LIB [17].

data and the remaining six cells were used for testing purposes. The used model hyper parameters are a kernel size of 5, a dropout value of 0.2 and a batch size of 128. The model was trained for 2000 epochs.

In Figure 1.2.6, the reference discharge profiles are shown of a LIB is shown. The initial capacity of the LIB with 100 % SOH is 2.1 Ah. With increasing aging, the capacity and thus the SOH decreases. The neural network used in this work consists of three layers with seven neurons each. Furthermore, the TCN is trained by using Adam's optimizer, which is an adaptive learning rate optimization algorithm that is specifically designed for deep learning applications [10]. The input for the TCN contains the capacity profile of the LIB.

In Figure 1.2.7, the SOH estimated by the TCN, and the reference measurement are plotted. The TCN predicts the SOH very accurately for the whole lifespan of the LIB. The integral mean squared error (MSE) for all predictions is approximately 0.9 %.

Here the high adaptability and self-learning ability from neural networks are coming to the fore, especially for real-world data with dynamically changing environment and operating conditions. Therefore, the TCN can provide reliable SOH estimations for the whole lifetime of LIB.

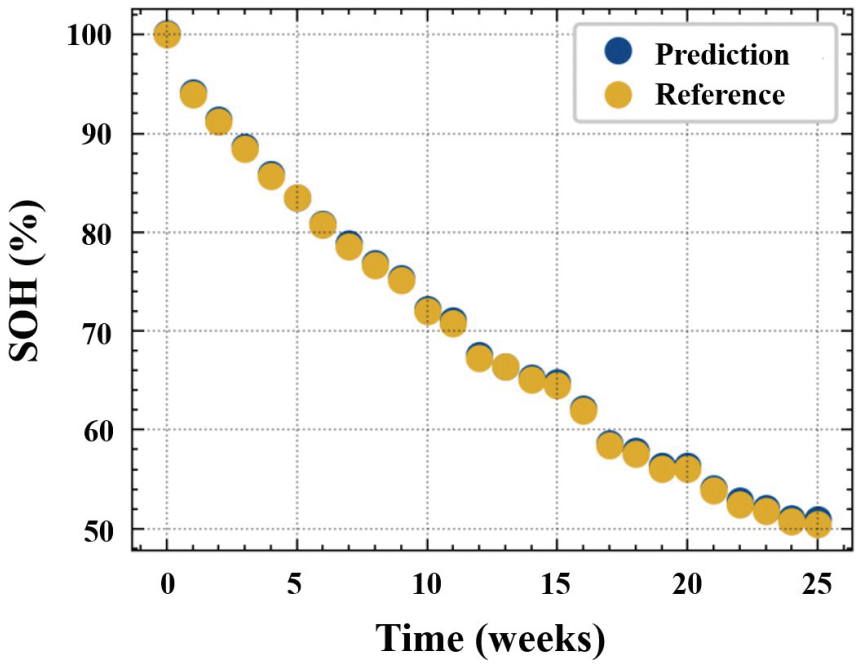


Figure 1.2.7 SOH prediction using a TCN with a reference measurement for the whole lifetime of a LIB.

1.2.6 Conclusion

For a safe, economically, and energetically efficient and sustainable utilization of retired EV batteries, reliable and accurate state parameter predictions are an indispensable prerequisite. To ensure a safe operation, an accurate prediction of the LIBs state of health (SOH) is essential. Traditionally, physical based SOH estimators are often limited due to their poor robustness regarding the highly non-linear dependence of the SOH on the changes of environment and working conditions during the operation. Data-driven approaches have shown their potential to overcome the drawbacks of traditional SOH estimation algorithms [16]. In the AI4DI project and its demonstrator "autonomous reconfigurable battery system", a novel machine learning algorithm called TCN was implemented that combines beneficial properties of long-short term memory recurrent neural networks while being computationally more efficient [17]. In this paper, it has been shown that using a TCN the SOH of a LIB can be accurately predicted with an MSE error over the whole LIB lifetime with less than 1%. As a result,

with this approach, the uncertainty of the heterogeneous performances and characteristics of retired electric vehicle batteries can be drastically reduced.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] Jiao, N., Evans, S. (2018), Business models for repurposing a second-life for retired electric vehicle batteries, In *Behaviour of Lithium-Ion Batteries in Electric Vehicles* (pp. 323-344), Springer, Cham.

[2] IDTechEx Research, Second-Life-Batteries for EV 2020-2030, *Online Access 2021/04/19*

[3] Engel, H., Hertzke, P., & Siccardo, G. (2019). Second-life EV batteries: The newest value pool in energy storage. *McKinsey Center for Future Mobility, Global Editorial Services*.

[4] M. Giegerich et al., "Open, flexible and extensible battery management system for lithium-ion batteries in mobile and stationary applications," 2016 IEEE 25th International Symposium on Industrial Electronics (ISIE), 2016, pp. 991-996. Available online at: https://doi.org/10.1109/ISIE.2016.7745026.

[5] M. Akdere et al., "Hardware and software framework for an open battery management system in safety-critical applications," IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, 2016, pp. 5507-5512. Available online at: https://doi.org/10.1109/IECON.2016.7793001

[6] S. Waldhoer, S. Bockrath, M. Wenger, R. Schwarz and V. R. H. Lorentz, "foxBMS - free and open BMS platform focused on functional safety and AI," PCIM Europe digital days 2020; International Exhibition and Conference for Power Electronics, Intelligent Motion, Renewable Energy and Energy Management, 2020, pp. 1-6.

[7] V. R. H. Lorentz, R. Schwarz, P. Kanzler, S. Waldhör, M. Wenger, M. Gepp, S. Koffel, S. Wacker, J. Wachtler, T. Huf, and A. Ochs, “foxBMS - The Most Advanced Open Source BMS Platform: foxBMS 2”, 2021, doi: 10.5281/zenodo.4727562

[8] R. Schwarz, K. Semmler, M. Wenger, V. R. H. Lorentz and M. März, “Sensorless battery cell temperature estimation circuit for enhanced safety in battery systems,” IECON 2015 - 41st Annual Conference of the IEEE Industrial Electronics Society, 2015, pp. 001536-001541. Available online at: https://doi.org/10.1109/IECON.2015.7392319.

[9] V. R. H. Lorentz, R. Schwarz, P. Kanzler, S. Waldhör, M. Wenger, M. Gepp, S. Koffel, S. Wacker, J. Wachtler, and T. Huf, "foxBMS - The Most Advanced Open Source BMS Platform: foxBMS 1", 2021, doi: 10.5281/zenodo.4720588

[10] Ilya, S. (2013). Training recurrent neural networks [Ph. D. dissertation]. *University of Toronto, Canada*

[11] Saha, B., & Goebel, K. (2007). Battery data set. *NASA AMES prognostics data repository*.

[12] Oord, A. V. D., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., & Kavukcuoglu, K. (2016). Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.

[13] Lea, C., Flynn, M. D., Vidal, R., Reiter, A., & Hager, G. D. (2017). Temporal convolutional networks for action segmentation and detection. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 156-165).

[14] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

[15] Zhou, D., Li, Z., Zhu, J., Zhang, H., & Hou, L. (2020). State of Health Monitoring and Remaining Useful Life Prediction of Lithium-Ion Batteries Based on Temporal Convolutional Network. *IEEE Access*, *8*, 53307-53320.

[16] Tian, H., Qin, P., Li, K., & Zhao, Z. (2020). A review of the state of health for lithium-ion batteries: Research status and suggestions. *Journal of Cleaner Production*, *261*, 120813.

[17] Zhou, D., Li, Z., Zhu, J., Zhang, H., & Hou, L. (2020). State of Health Monitoring and Remaining Useful Life Prediction of Lithium-Ion Batteries Based on Temporal Convolutional Network. *IEEE Access*, *8*, 53307-53320.

[18] Chiueh, T. C., Huang, M. C., Juang, K. C., Liang, S. H., & Ling, W. (2018). Virtualizing Energy Storage Management Using {RAIBA}. In 2018 {USENIX} Annual Technical Conference ({USENIX}{ATC} 18) (pp. 187-198).

[19] ECSEL AI4DI project. Artificial Intelligence for Digitising Industry. Available online at: https://ai4di.eu/

1.3

Optimising Trajectories in Simulations with Deep Reinforcement Learning for Industrial Robots in Automotive Manufacturing

Noah Klarmann[1*], Mohammadhossein Malmir[1*], Josip Josifovski[1*], Daniel Plorin[2], Matthias Wagner[2] and Alois C. Knoll[1]

[1]Technical University of Munich, Germany
[2]AUDI AG, Germany
[*]These authors contributed equally to this work

Abstract

This paper outlines the concept of optimising trajectories for industrial robots by applying deep reinforcement learning in simulations. An application of high technical relevance is considered in a production line of an autmotive manufacturer (AUDI AG), where industrial manipulators apply sealant on a car body to prevent water intrusion and hence corrosion. A methodology is proposed that supports the human expert in the tedious task of programming the robot trajectories. A deep reinforcement learning agent generates trajectories in virtual instances where the use case is simulated. By making use of the automatically generated trajectories, the expert's task is reduced to minor changes instead of developing the trajectory from scratch. This paper describes an appropriate way to model the agent in the context of Markov decision processes and gives an overview of the employed technologies. The use case outlined in this paper is a proof of concept to demonstrate the applicability of reinforcement learning for industrial robotics.

Keywords: deep reinforcement learning, automotive manufacturing, simulation, industrial robotics, virtual learning platform, trajectory optimisation, motion planning, offline programming, robot learning.

Video: A video clip demonstrating the proposed methodology is available at: https://vimeo.com/562948911.

1.3.1 Introduction

A concept for the automatic generation of trajectories for the control of industrial robots is presented in this work. Data-based robotic control has the potential to address two major shortcomings of conventional robot programming [1]:

(i) The classic programming of industrial robots is done manually by a specialist who precisely specifies the trajectory of the robotic arm *Tool Center Point* (TCP). To this end, the programmer is in close contact with the responsible plant engineer as well as the product owner to fulfill all demands, restrictions, and requirements.

(ii) Conventional programming is deterministic and thus prevents the flexible adaptation of the robot to changing environments (like varying products). A variable control that adapts to different conditions cannot be realized with classical programming.

Both shortcomings are addressed in this work by introducing a self-taught and automated method for robot control programming. By adopting the reinforcement learning methodology [2, 3], the control is learned based on sampled experience from the interaction with a virtual environment. In this context, a predefined reward function is optimised that steers the action policy towards the desired outcome of the robotic manipulation. Reinforcement learning combined with non-linear function approximators (e.g., neural networks – referred to as *Deep Reinforcement Learning* DRL) can generalize action policies on experience to solve problems with large and complex state spaces (e.g., learning from unstructured data such as a camera signal). Further advancements in the field of reinforcement learning are algorithms that can deal with continuous action spaces enabling robotic control [4, 5]. These developments have recently led to interesting milestones, such as learning the locomotion of a four-legged robot [6] or robots that have learned to open doors [7]. Well-known achievements in the field of reinforcement learning are based on environments that allow the efficient generation of an abundant amount of experience (e.g., video games [8] or board games [9, 10]). However, robots require physical interaction with the environment whereas training the agent in the real world is exceptionally resource intense. As mentioned in [11], an agent that learns a simple

grasping task requires experience from 800,000 episodes. The application of reinforcement learning to industrial robotics requires a significant amount of data due to the following two characteristics: (i) complex spaces for the state (e.g., learning based on unstructured data such as images) and action spaces (e.g., controlling the torques of each axis), and (ii) industrial requirements regarding safety and precision of the robotic control.

An appealing alternative to real-world training is to simulate the agent-environment interaction and transfer the control to the real world. For the use case described by Rusu et al., a considerable speed increase by the factor of 50 could be achieved when a robot is trained in parallel virtual environments instead of the real world [12]. Another advantage in the use of virtual methods is the avoidance of accidents that could occur during the training of robots in the real world [13].

In this work, DRL is employed to find the optimal trajectory of a robot that is applying PVC sealant on a door frame of a car body to prevent water intrusion and hence corrosion. Moreover, the robotic trajectory will be optimised with respect to the following aspects: (i) providing smooth velocity of the end effector, (ii) ensuring the optimal orientation of the end effector's nozzle to the car body surface, and (iii) avoiding collisions. Related work from other groups in which DRL is used for trajectory planning exists. In [14], a DRL agent learns to control a robot with six axes to solve the hot wire game that is seen as the first step towards industrial applications like welding, gluing, or cutting. A more practical application can be found in [15] where the authors propose a simulated environment for learning the optimal trajectories for applying paint on a car body. Besides learning a concrete task, DRL can also be used for the online optimisation of trajectories for robots with unknown/partially known dynamics that usually lead to control jumps [15–17].

Beyond the application to manipulators, DRL can also be used to generate trajectories for mobile robotics. The path planning for mobile robots with a known map can be found in [18], whereas the navigation with simultaneous map generation is proposed in [19–21]. The application of DRL to optimise the data collection for an agent that explores the environment can be found in [22]. Moreover, DRL to evaluate optimal trajectories for *Unmanned Air Vehicles* (UAVs) providing access points to end-users is presented in [23].

A unique feature of this work is the advanced simulation environment that can simulate the car body and robot with detailed geometry considering realistic physical behavior and a high-end rendering pipeline. In addition, the close collaboration with the industrial partner imposes high industrial

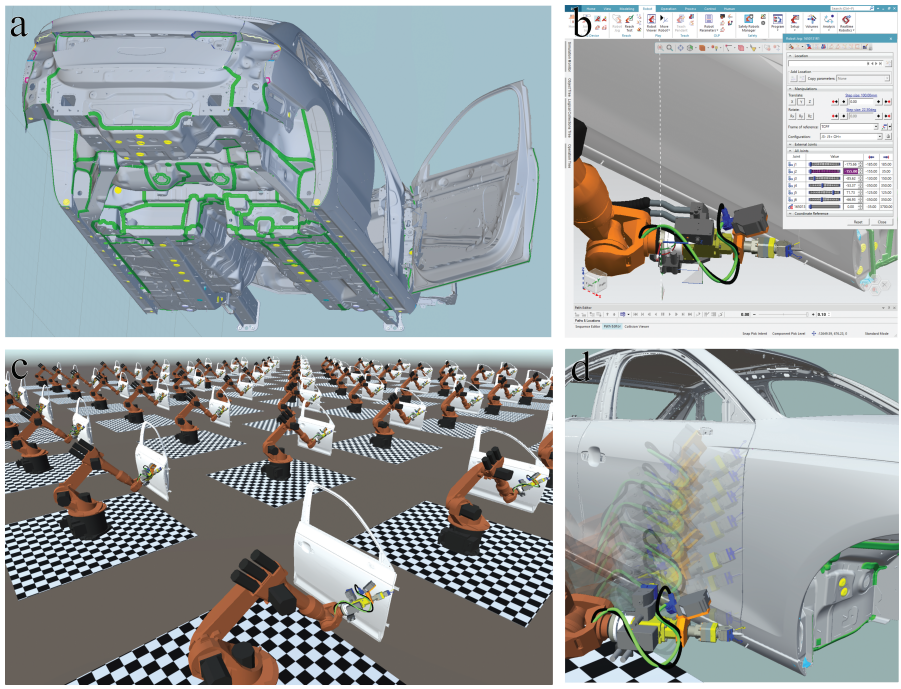


Figure 1.3.1 (a) Manually predefined geometric model of the seam, (b) the path editor and the robot jog in tecnomatix process simulate for manual trajectory programming, (c) multiple training environments running in parallel and optimising the trajectory for a car door, and (d) the robot following the learned trajectory.

requirements on this approach, and thus exceeds the usual academic proof-of-concept state.

1.3.2 Background

AUDI AG employs a fully automated process for applying the sealant material on car bodies using industrial manipulators. To program the manipulator, a conventional procedure is adopted in which fixed trajectories are specified by an expert. To this end, the creation of trajectories for the automation of a new car body requires the following steps: (i) A three-dimensional model of the seam that is supposed to be applied to the car body is designed manually (see the green markers in Figure 1.3.1a). (ii) The expert programs the final robot trajectories in a way that the manually predefined geometric model of the seam will be followed by the manipulator. It is worth

noting that there is a multitude of different nozzles and end effectors that vary according to the task and need to be specified before starting to program the robot. To create the robotic trajectory, the offline programming expert manually defines a sequence of parameterized motions using software tools (see Figure 1.3.1b). Two different ways are typically used to specify the movement of the end-effector along a free or constrained path towards a specific target. The first type is called *Point-to-Point* (PTP) motion that is used when the target pose should be reached as fast as possible by maximizing joint-level rotational speeds without specifying the TCP path. The second option is referred to as *Linear* (LIN) motion and is used whenever the target pose should be reached along a straight line with a specified velocity and acceleration. PTP motions are mainly used in two scenarios: (i) The robot's TCP should be moved to an initial pose configuration that is a suitable starting point and (ii) whenever there is a need to relocate or reorient the robot's TCP between the seam segments. The LIN motions are used whenever the nozzle tool is applying the sealant material as the priority is on accurate motions to guarantee a straight and uniform line.

It is worth noting that manual programming is a tedious and resource-consuming process as time-optimal and collision-free trajectories are complex and rely on many different parameters such as (i) quantitative metrics (e.g., how much time it takes to complete the seaming process), (ii) qualitative aspects (e.g., straight and uniform appearance of the seam), or (iii) expert-subjective considerations that are based on experience.

To reduce the complexity and workload of the task, the programmer can start with an existing trajectory from a different but similar scenario in which a solution already exists. An existing case from which the trajectory can be reused is called a *brown field*. By adopting a brown field, the manual effort is typically reduced tremendously as only minor modifications are necessary in most cases. In the so-called *green field* scenario, the car body is significantly different from any existing scenarios with the implication that the programmer must manually define the trajectories from scratch. In the following section, a procedure is described in which brown fields are generated by a DRL agent.

1.3.3 Methodology

The described problem is addressed by a DRL agent that finds an optimised trajectory in simulation. Figure 1.3.2 depicts the agent's state and action spaces as well as the reward function in the context of a *Markov Decision*

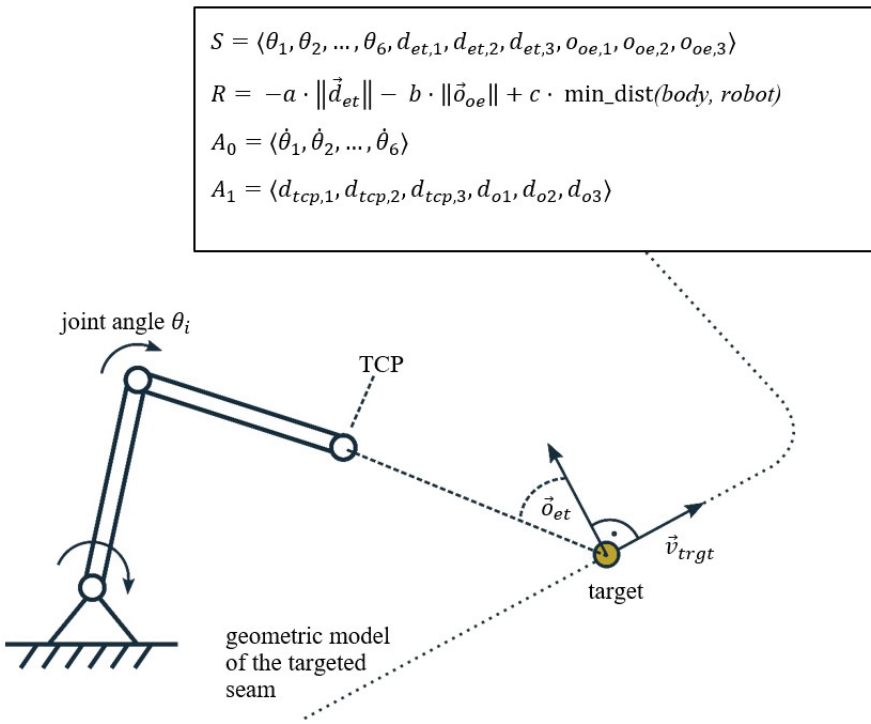


Figure 1.3.2 DRL agent in the MDP formalization for optimising a TCP trajectory by minimizing the distance to a predefined moving target.

Process (MDP). The geometric model of the targeted seam is given in the process as described in the previous section. An imaginary target location is modeled that travels along the predefined path at a specified velocity $\vec{v}_{trgt}$. The first term of the reward R is modeled as the negative distance from the TCP to the target location $\left\|\vec{d}_{et}\right\|$. By maximizing this term, the agent controls the robot's TCP to follow the target. When applying the sealant, the predefined target velocity $\vec{v}_{trgt}$ is required to be at a specific constant velocity to guarantee an optimally applied sealant. The application of the sealant is required to occur in an orientation that is orthogonal to the surface while aligned with the direction of the sealant line. Based on this, a reference end-effector orientation can be defined; the second term of the reward function penalizes the difference between the actual and the reference orientation of the end-effector (namely, the orientation error $\vec{o}_{oe}$, calculated based on the orientation of the target

surface normal w.r.t. the end-effector $\vec{o}_{et}$). The third term of the reward R measures the minimum distance of the manipulator to the car body to avoid collisions. Moreover, three weights (a, b, c) are used to balance each term individually.

Two different ways to define the action space are considered: (i) The robot is controlled on the joint level with the agent controlling the speed (or torque) of each joint individually. This results in a six-dimensional action space (action space A_0) for a robot that has six *Degrees of Freedom* (DoF). (ii) By employing kinematics, the TCP can be controlled directly in the operational space, e.g., by specifying a target delta for each Cartesian space dimension $d_{tcp,1}, d_{tcp,2}, d_{tcp,3}$ and for each orientation axis d_{o1}, d_{o2}, d_{o3} (see action space A_1). The state description comprises the relative position and orientation of the end-effector with respect to the current target point and the angles of all joints $\theta_1, \theta_2, \ldots, \theta_6$ of the manipulator. To simulate the robot interacting with the environment, a simulation developed with the commercial game engine *Unity3D* [24] is used (Figure 1.3.1c and d). *Unity3D* is a suitable choice for the planned undertaking as it provides advanced rendering capabilities (to potentially learn from pixels), the opportunity to write user-defined functions, and it comes with an efficient GPU-accelerated physics engine (*Nvidia PhysX* [25]).

To solve the MDP described above, the algorithm *Proximal Policy Optimization* [26] (PPO) is employed. PPO is a policy gradient method that trains both an actor and a critic function, whereas the policy update gradient is clipped to prevent stability issues. An important feature of PPO is the support of continuous action spaces that is achieved by training probability density functions instead of discrete actions. In this work, the PPO implementation of the *Stable Baselines* library is used (referred to as PPO2 [27]) that allows running multiple workers updating the same policies. Each simulation instance runs several robots at the same time (Figure 1.3.1c), whereas policy gradient updates are gathered in batches for the periodic update of the two policy networks.

The proposed methodology is applied to create brown fields. In a second step, the final control can be derived from the brown fields by the offline programmer. As mentioned before, a significant reduction of the workload can be achieved when the final programming is done based on a brown field provided by the agent. The simulation starts from scratch without any knowledge (starting from a green field). A disadvantage in taking a previous solution into account (e.g., starting the simulation from an existing but incompatible brown field) is a potential bias regarding the solution and might

lead to a local minimum. However, starting from another brown field comes with the high potential to reduce the training time and can be investigated in the future.

One might ask why brown fields are evaluated rather than directly learning the final robotic programming. While it is indeed possible to let the agent define the final trajectory, the human is kept in the loop as the approximative nature of DRL as well as the difference between the real world and the simulation (reality gap) occasionally lead to undesired control behavior.

1.3.4 Conclusion and Outlook

The concept of optimising trajectories with DRL for industrial robots in simulations is outlined in this paper. To this end, a possible MDP formalization of the agent that has the potential to considerably reduce the amount of the manual work that is involved in the offline programming of the industrial robots is presented. For the further course of this undertaking, an adoption of the methodology in three steps is envisioned: (i) The experts are performing plausibility checks by comparing hand-crafted trajectories with the solution from the DRL agent. (ii) The agent is used in production by creating brown fields from which the expert derives the final solution. (iii) The agent finds final robotic trajectories and the human experts verify the solution without modifying it.

Beyond the specific application that is outlined in the paper, an end-to-end learning platform is envisioned that satisfies the industrial requirements of industrial robotic applications. A high degree of generalization is targeted to address a wide variety of different tasks that are typical for manufacturing.

Acknowledgements

This work has been financially supported by AI4DI project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway. The authors would like to thank Khedr Kaddour and Luke Pugin for their help and technical support in preparing this publication and Bare Luka Zagar for reviewing the content.

References

[1] Knoll, A., and Walter, F., "Teleported AI," [online], 2020, mediatum.ub.tum.de/attfile/1575022/incoming/2020-Sep/221826.pdf.

[2] David Silver, "Reinforcement Learning: UCL Course," https://deepmind.com/learning-resources/-introduction-reinforcement-learning-david-silver, [retrieved 17 April 2020].

[3] Sutton, R. S., and Barto, A. G., Reinforcement learning. An introduction, 2nd edn., The MIT Press, Cambridge, Massachusetts, 2018.

[4] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D., "Continuous Control with Deep Reinforcement Learning," 10 Sep. 2015, http://arxiv.org/pdf/1509.02971v6, [retrieved 16 November 2020].

[5] Mahmood, A. R., Korenkevych, D., Vasan, G., Ma, W., and Bergstra, J., "Benchmarking Reinforcement Learning Algorithms on Real-World Robots," 2018, http://arxiv.org/pdf/1809.07731v1, [retrieved 10 May 2021].

[6] Levinson, J., Askeland, J., Becker, J., Dolson, J., Held, D., Kammel, S., Kolter, J. Z., Langer, D., Pink, O., Pratt, V., Sokolsky, M., Stanek, G., Stavens, D., Teichman, A., Werling, M., and Thrun, S., "Towards Fully Autonomous Driving: Systems and Algorithms," IEEE Intelligent Vehicles Symposium, 2011, pp. 163–168.

[7] Yahya, A., Li, A., Kalakrishnan, M., Chebotar, Y., and Levine, S., "Collective Robot Reinforcement Learning with Distributed Asynchronous Guided Policy Search," 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017.

[8] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D., "Human-Level Control Through Deep Reinforcement Learning," Nature; Vol. 518, No. 7540, 2015, pp. 529–533.

[9] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D., "Mastering the Game of Go With Deep Neural Networks and Tree Search," Nature; Vol. 529, No. 7587, 2016, pp. 484–489.

[10] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T., Hui, F., Sifre, L., van den Driessche, G., Graepel, T., and Hassabis, D., "Mastering the Game of Go Without Human Knowledge," Nature; Vol. 550, No. 7676, 2017, pp. 354–359.

[11] Levine, S., Pastor, P., Krizhevsky, A., and Quillen, D., "Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection," 7 Mar. 2016, http://arxiv.org/pdf/1603.02199v4, [retrieved 15 February 2021].

[12] Rusu, A. A., Vecerik, M., Rothörl, T., Heess, N., Pascanu, R., and Hadsell, R., "Sim-to-Real Robot Learning from Pixels with Progressive Nets," 14 Oct. 2016, http://arxiv.org/pdf/1610.04286v2, [retrieved 16 November 2020].

[13] Peng, X. B., Andrychowicz, M., Zaremba, W., and Abbeel, P., "Sim-to-Real Transfer of Robotic Control with Dynamics Randomization," 18 Oct. 2017, http://arxiv.org/pdf/1710.06537v3, [retrieved 16 November 2020].

[14] Meyes, R., Tercan, H., Roggendorf, S., Thiele, T., Büscher, C., Obdenbusch, M., Brecher, C., Jeschke, S., and Meisen, T., "Motion Planning for Industrial Robots using Reinforcement Learning," Procedia CIRP; Vol. 63, 2017, pp. 107–112. doi: 10.1016/j.procir.2017.03.095.

[15] Ota, K., Jha, D. K., Oiki, T., Miura, M., Nammoto, T., Nikovski, D., and Mariyama, T., "Trajectory Optimization for Unknown Constrained Systems using Reinforcement Learning," 2019, http://arxiv.org/pdf/1903.05751v2, [retrieved 10 May 2021].

[16] Akrour, R., Neumann, G., Abdulsamad, H., and Abdolmaleki, A., "Model-Free Trajectory Optimization for Reinforcement Learning," Proceedings of The 33rd International Conference on Machine Learning, Vol. 48, PMLR, New York, New York, USA, 2016, pp. 2961–2970.

[17] Tassa, Y., Erez, T., and Todorov, E., "Synthesis and stabilization of complex behaviors through online trajectory optimization," 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 07/10/2012 - 12/10/2012, pp. 4906–4913.

[18] Zhang, J., Springenberg, J. T., Boedecker, J., and Burgard, W., "Deep reinforcement learning with successor features for navigation across similar environments," 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 24/09/2017 - 28/09/2017, pp. 2371–2378.

[19] Botteghi, N., Sirmacek, B., Mustafa, K. A. A., Poel, M., and Stramigioli, S., "On Reward Shaping for Mobile Robot Navigation: A Reinforcement Learning and SLAM Based Approach," 2020, http://arxiv.org/pdf/2002.04109v1, [retrieved 20 May 2021].

[20] Tai, L., Paolo, G., and Liu, M., "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 24/09/2017 - 28/09/2017, pp. 31–36.

[21] Yu, J., Su, Y., and Liao, Y., "The Path Planning of Mobile Robot by Neural Networks and Hierarchical Reinforcement Learning," Frontiers in neurorobotics, published online 2 Oct. 2020; Vol. 14, 2020, p. 63.

[22] Kollar, T., and Roy, N., "Trajectory Optimization using Reinforcement Learning for Map Exploration," The International Journal of Robotics Research; Vol. 27, No. 2, 2008, pp. 175–196.

[23] Harald Bayerlein, Paul De Kerret, and David Gesbert, Trajectory Optimization for Autonomous Flying Base Station via Reinforcement Learning, IEEE, Piscataway, NJ, 2018.

[24] "Unity3d," https://unity.com, [retrieved 29 April 2020].

[25] "Nvidia PhysX," https://www.nvidia.com/de-de/drivers/physx/9_16_0318/physx-9-16-0318-driver-de/, [retrieved 29 April 2020].

[26] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O., "Proximal Policy Optimization Algorithms," 20 Jul. 2017, http://arxiv.org/pdf/1707.06347v2, [retrieved 16 November 2020].

[27] Hill, A., Raffin, A., Ernestus, M., Gleave, A., Kanervisto, A., Traore, R., Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., and Wu, Y., "Stable Baselines," GitHub, 2018 GitHub repository.

1.4

Foundations of Real Time Predictive Maintenance with Root Cause Analysis

Franz Wotawa[1], David Kaufmann[1], Adil Amukhtar[1], Iulia Nica[1], Florian Klück[2], Hermann Felbinger[2], Petr Blaha[3], Matus Kozovsky[3], Zdenek Havranek[3] and Martin Dosedel[3]

[1]Graz University of Technology, Austria
[2]AVL List GmbH, Austria
[3]Brno University of Technology CEITEC, Czech Republic

Abstract

Research on cyber-physical systems comes to the fore with the increasing progress of applications in the field of autonomous systems. Therefore, there is a growing interest in methods for enhancing reliability, availability, and self-adaptation of such systems in safety critical situations. Hence, it is essential that autonomous systems are equipped with a detection system to observe faulty behaviour in real time or to predict failing operations to avoid safety critical scenarios, which may harm people. To bring or hold a system within healthy conditions, not only detecting a faulty behaviour is important, but also to find the corresponding root cause.

In this article, we introduce different methods which make use of detecting unexpected behaviour in cyber-physical systems, for the localization of faults. The first approach, *model-based diagnosis* uses logic to represent a cyber-physical system to perform reasoning for computing diagnosis candidates. A second promising approach deals with *simulation-based diagnosis* systems, using *digital twin models* to produce faulty behaviour data in advance, and to find correlations with the original cyber-physical system's behaviour, for diagnosis. For the third method the focus is set on artificial intelligence (machine learning and neural networks), where

the goal is to utilize a huge amount of health and safety critical observations of the system for training to approximate the behaviour associated with faulty and safety critical states.

Keywords: model based diagnosis, model based reasoning, simulation based diagnosis, digital twin, AI based predictive maintenance, AI based diagnosis, abstract model, datacentre design, energy efficiency of datacentre, energy efficient metrics, datacentre carbon footprint computation.

1.4.1 Introduction and Background

Predictive analytics deals with forecasting the future progression of a situation and has a wide range of applications, including weather forecasting, epidemiology prediction, stock market prediction, and predictive maintenance. When implementing predictive maintenance, predictive modelling plays a major role. It aims to guarantee a robust prediction result, which can save considerable production downtime and either prevent or diminish economic loss. Considering information utilization and modelling mechanism, the predictive modelling techniques can be classified into three groups: *physics-based*, *data-driven*, and *model-based*.

The *physics-based* approach describes the physical behaviour of a system using the first principle as a series of ordinary or partial differential equations according to the law of physics [1][6]. However, the construction of a physics model is usually difficult since it requires detailed and complete knowledge about the system. Still, this kind of model lacks extensive failure samples to determine the model parameters in practice.

The *data-driven* approach constructs a model representing the underlying relationship of a system based on data mining techniques. The *data-driven* approach could be grouped into two categories including statistical and machine learning based methods. The typical statistical method used, include the autoregressive model and its variations, linear regression, Wiener process, and Gamma process among others. Machine learning based methods include algorithms such as artificial neural networks, clustering techniques, extreme learning machines, fuzzy logic, and deep learning models. However, the performance of the *data-driven* model is sensitive to the size and quality of the collected dataset. It is important to note that *data-driven* models are extremely domain specific. Therefore, the selection of such models is a crucial part of the process.

The *model-based* approach takes advantage of established physical knowledge and collected data to enhance the prediction performance. It typically involves two steps including model construction and model updating [2]. First, analytical models are built based on the physical or empirical model representing the situation evolving in a quantitative manner. These models are then updated with newly acquired information to predict the future progression of the situation based on inference. Comparing with the *data-driven* approach, the *model-based* approach requires less historical data to construct the models. The predicted value is associated with a confidence level, resulting from the uncertainty involved in the prediction process [3].

Over the past 30 years, predictive maintenance has been evolving from predicting failures based on periodic visual inspections to continuous real-time monitoring of assets and external data with alerts based on statistical techniques such as regression analysis for at least one important asset. Furthermore, the advent of Industrial Internet of Things (IIoT) technology has significantly optimized industrial operations management by connecting industrial assets with information systems and, hence, with business processes. Predictive Maintenance 4.0 (PdM 4.0) or simply Maintenance 4.0, is among the major focus points of IIoT. In [4] the authors identify four levels of maturity in predictive maintenance, depicted in Figure 1.4.1.

Many companies are combining the capabilities of IIoT and Big Data to predict equipment malfunctions. The accuracy of the forecast is further

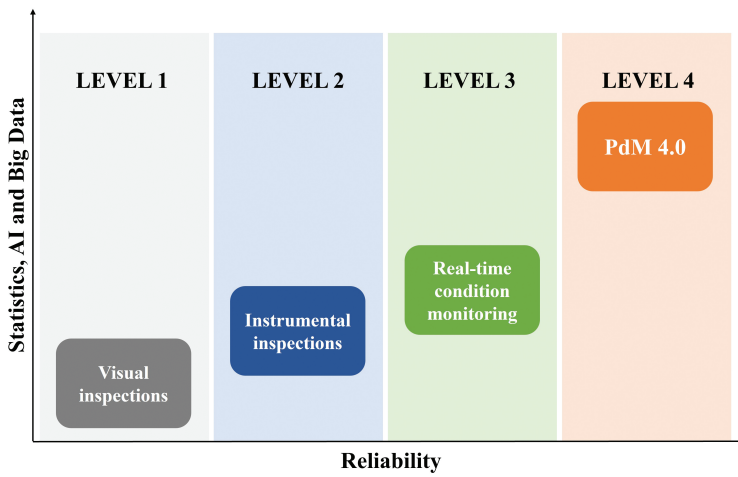


Figure 1.4.1 Four levels of maturity in predictive maintenance.

getting more precise with improved Artificial Intelligence (AI) techniques and machine learning tools.

As depicted in [5], Maintenance 4.0 forms a subset of smart manufacturing systems which are autonomous in their operation, capable of predicting failures and triggering maintenance activities. These systems consist of smart equipment in form of embedded or cyber-physical systems forming the digital twin of physical assets. To achieve near-zero defects, near-zero downtime and automated decision making based on condition monitoring, top diagnosis and prognosis techniques need to be implemented.

Finally, the most advanced form of maintenance is prescriptive maintenance which builds on PdM and provides further guidance on the maintenance task, including diagnosis capabilities. Prescriptive maintenance strategies extensively use advanced data processing and visualization techniques such as graph analysis, simulations, neural networks, complex event processing, heuristics, and machine learning. These tools can calculate the timing and the effect of failure, thus, deciding on the priority and urgency of the maintenance activity.

In Figure 1.4.2, we depict a simplified system architecture, showing how the different approaches contribute to diagnosis of systems. Simulation-based diagnosis as well as AI-based diagnosis, utilize models that are obtained in a pre-offline phase, depicted on the right. Model-based diagnosis makes use of

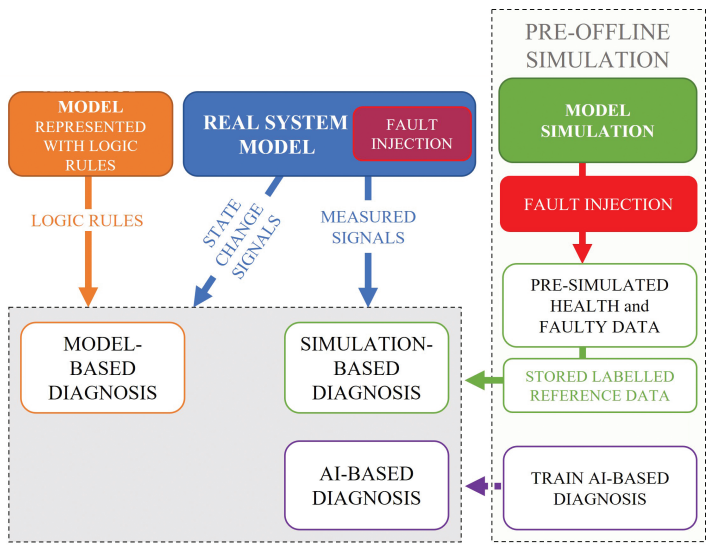


Figure 1.4.2 Diagnosis system architecture.

mainly abstract models for diagnosis directly not requiring an offline phase. In the following, we discuss the different approaches and their foundations in more detail.

1.4.2 Foundations

In the following, we discuss the foundations behind the diagnosis, i.e., the detection of failures and the identification of its root causes in the context of predictive maintenance. In particular, we focus on methods from artificial intelligence considering model-based diagnosis, machine learning, and specifically neural networks. Instead of a detailed discussion of the foundations, we briefly introduce underlying ideas and provide references to related literature for the interested reader.

1.4.2.1 Model-based Diagnosis

Model-based diagnosis or reasoning from the first principle has been developed in the 80s of the last century as an answer to challenges arising when using logic reasoning as a basis for applications like configuration and decision support. Instead of formalizing the knowledge-base in a way from observations to causes such that diagnosis can be directly derived using ordinary deduction, the idea was to formalize knowledge either in form of relations or as rules where causes imply their effects. Instead of deduction abduction or in a more general setting non-monotonic reasoning was used as an underlying reasoning mechanism (see [19], [20], [21]).

The idea behind model-based diagnosis is to take a model of a system, which is usually called a system description SD, and observations OBS for diagnosis computation. In this setup, SD comprises the structure of the system comprising interconnected components, and the behaviour of the components. For the latter, we explicitly introduce health states for components like working abnormally (ab), or correctly (i.e., not abnormal ($\neg ab$)). For example, the correct behaviour of components can be formalized using an implication, i.e., $\neg ab(C) \rightarrow behav(C)$, where C is a component and $behav(C)$ the behaviour of C. Whenever the component works as expected the behaviour is determined. However, if we assume C to be wrong, the implication does not allow us to determine behaviour. Hence, the component may work appropriately even when considered to be faulty. Note that this modelling allows also to specify a behaviour for any incorrect health state if required.

When assuming a model of a system comprising components $COMP$ and observations, we are able to compute diagnoses. Informally, a diagnosis explains a faulty behaviour. In the case of model-based diagnosis, we are interested in assigning a health state to every component in $COMP$ such that the observations are not in contradiction with the components' behaviour. Hence, diagnosis becomes searching for health states. In order to be applicable in practice, diagnosis reasoning often utilizes simplifications like searching only for diagnoses where one component is considered to be faulty, and all others are working as expected. Alternatively, diagnosis search may focus on parsimonious diagnoses, i.e., health assignments to components unequal to $\neg ab$, where we are not able to switch a component from being faulty to working correctly.

Model-based diagnosis computation in general is hard and requires a lot of computational resources. However, considering today's hardware, most recent algorithms, and the availability of fast theorem provers, diagnosis can be computed within a reasonable amount of time, i.e., within a fraction of a second even for larger systems (see [22]). For a more detailed discussion on model-based diagnosis, modelling, formal definitions, and its application to self-adaptive systems we refer to [23] and most recently [24].

1.4.2.2 Machine Learning Based Diagnosis

Machine learning algorithms have shown promising solutions and improved decision-making processes by analysing an enormous amount of data. The use of these algorithms has grown rapidly in the recent years which helps systems to act intelligently without being explicitly programmed [7]. Machine learning techniques are often used to detect faulty behaviours of the system [8], [9]. For example, [10] used Support Vector Machine (SVM), a machine learning algorithm to model linear and non-linear relationships, to model 9 fault states of the modular production system with different kernel functions namely Sigmoid, RBF, polynomial and linear kernel functions. The work presented a 100% classification rate on all kernel functions except for the sigmoid kernel (52.08% classification rate).

Machine learning algorithms are mainly divided into four categories explained below:

- *Supervised*: This type of learning typically learns a function based on the sample input and output pairs. The goal of the function is to classify/map a new input instance to the respective output [11]. Please note that the data samples provided during the training are labelled.

- *Unsupervised*: Unsupervised learning involves understanding the distribution of the data given the data is unlabelled [11]. These types of algorithms are mostly used for feature generation, dimensionality reduction, extracting hidden patterns, clustering/grouping data points, and exploratory analysis.
- *Semi-Supervised*: Data points could be rarely labelled in real world [12]. For example, in the fraud detection problem, there could be few occurrences of fraud transaction leaving too much non-fraud detection data. Thus, semi-supervised learning comes into play by generating new instances from the less seen (minatory output), often called synthetic data generation. It's a hybridization of "supervised" and "unsupervised" where the goal is to model better predictions given the data is highly unlabelled.
- *Reinforcement*: Reinforcement learning is an area of machine learning in which an agent is trained to learn the optimal behaviour for a given environment [13]. The goal of reinforcement learning is to find the best possible actions such that reward is maximized and the risk is minimized. Reinforcement learning is mostly useful for automation e.g., autonomous driving.

Based on the application, nature of the data and learning outcome, various machine learning algorithms can be chosen for fault diagnosis in complex systems. For this case study, we model the fault diagnosis problem with one of the supervised machine learning algorithms called Bootstrap Aggregation (Bagging).

1.4.2.3 Artificial Neural Networks for Diagnostics

Machine learning as well as deep learning techniques are very popular in many areas of engineer's work. The connection of the AI approach and technical diagnostics especially in the field of predictive maintenance of machines [14] is a very actual problem and directly addresses the Internet of Things as well as Industry 4.0 topics [15]. Big data processing algorithms, necessary for modern AI techniques application, are overviewed in, e.g., [16], standard machine learning approaches, mostly containing statistical algorithms [17] like SVM, k-NN, PCA, Mahalanobis-Taguchi strategy etc., are commonly used, but mainly using of powerful and very popular neural networks is currently growing. There exists a lot of NNs types used for diagnostics of the machines, but the convolutional neural network is one of the most recommended and also used types [18]. Mostly, NN algorithms

run on the dedicated and powerful hardware designed especially for such purposes.

The shift from cloud AI processing to local intelligence architecture is described in [25]. According to that paper, AI has a strong potential for sensor solutions in the future. Reasons are the increasing complexity of sensors, the increasing amount of generated raw data, and the requirement for straightforward data fusion from several sensors. The integration of wireless communication capabilities in smart sensors makes them usable also as an IoT device [26]. This process must be accompanied by the integration of safety- and privacy-aware functions.

1.4.3 Related Research

Predictive analytics intends to make predictions about future progressions, based on domain knowledge and historic data combined with physic-based, model-based or machine-learning modelling techniques. In the context of predictive maintenance (PdM), predictive modelling is used for failure prediction and prescription of operation and maintenance strategies. Here, the main objective is to obtain accurate and robust prediction results to avoid unexpected system downtime. Predictive maintenance is a condition-driven maintenance program that monitors the mechanical condition, system efficiency, and other indicators to determine the system's actual mean-time-to-failure or loss of efficiency. Considering the definition from [27], the three key steps of a PdM program are *data acquisition* to obtain data relevant to system health, *data processing* to handle and analyse the data or signals collected and *maintenance decision-making* to recommend efficient maintenance actions or adoptions of the operation strategy. Techniques for maintenance decision support in a PdM program can be divided into two main categories [27]: diagnostics and prognostics. Fault diagnostics focuses on detection, isolation, and identification of faults when they occur. In contrast, prognostics attempts to predict faults or failures before they occur. Jardine [28] reviewed and compared several commonly used PdM decision strategies such as trend analysis that is rooted in statistical process control (SPC), expert systems (ESs), and neural networks. Wang and Sharp [29] discussed the decision aspect of PdM and reviewed the recent development in modelling PdM decision support.

Various model-based diagnosis approaches have been applied to fault diagnosis of a variety of mechanical systems such as gearboxes [30][31], bearings [32][33][34], rotors [35][36] and cutting tools [37]. Hansen et al.

[38] proposed an approach to more robust diagnosis based on the fusion of sensor-based and model-based information. Vania and Pennacchi [39] developed some methods to measure the accuracy of the results obtained with model-based techniques aimed to identify faults in rotating machines. Two practical successful applications of maintenance programs using model-based approaches are: (i) an integrated framework for on-board fault diagnosis and failure prognosis of a helicopter transmission component and (ii) the TIGER system [40] that combines several artificial intelligence technologies, including qualitative model-based reasoning to perform condition monitoring of gas turbines. Here, the diagnostic mechanism is based on a fault manager and the three independent tools *KHEOPS [67], IxTeT* [40] and *CA-EN [69].* KHEOPS [41] is a high-speed rule-based system, used to express diagnostic rules in a classic rule-based formalism and allows the user to set pre-alarm limits for each parameter. IxTeT [40] is used to either describe the normal causal reaction or look for specific patterns resulting from known faults. CA-EN [42] is a model-based supervision system devoted to complex dynamic systems. CA-EN's representation formalism allows one to combine empirical causal knowledge and first principles of the domain.

The effectiveness of predictive maintenance depends on practical factors such as required planning time and implementation effort but especially on the achievable quality of condition monitoring, the behaviour of the deterioration process and system specific fault severity. For instance, vibration and oil debris monitoring is limited by the accuracy of the measuring instruments and can therefore be considered as imperfect [52]. In many cases, the imperfect condition information has been combined with deterioration processes, which were modelled as continuous stochastic processes. Kallen and Van Noortwijk [43] use a gamma deterioration process, Peng and Tseng [44] a linear trend with random coefficient plus a Brownian motion as a second random effect, Ye et al. [45] a Wiener process with positive drift, and Zio and Compare [46] a Randomized Paris-Erdogan fatigue crack growth model. Nevertheless, also here inspections have to be performed in order to obtain condition information. Given the effort and short comings, PdM should only be applied if the expected benefit outweighs the efforts and costs during the entire life cycle [47][48][49].

1.4.4 Conclusion

Predictive maintenance mechanisms are the major key to improve the availability, reliability and safety of cyber-physical systems in relation to

finding or predicting an unexpected behaviour before downtime, defects or harm to the environment occurs. In this article, we focus on different approaches for diagnosis, discuss their foundations, and also related research. However, there remains the questions which diagnosis methods to use and how to implement them to interact with a specific cyber-physical system. We elaborate on use cases in two separated articles of this book to answer these questions.

In these articles, we decided to focus on different diagnosis approaches based on two systems, a simplified DC e-motor model and a dual three-phase permanent magnet synchronous motor supported with detailed acausal e-motor model with the capability of fault injection. The use of model based and machine learning based approaches is demonstrated on a simplified DC e-motor model in the article "Real-Time Predictive Maintenance - Model Based and Machine Learning Based Diagnosis". The artificial neural network approaches are demonstrated on a dual three-phase motor diagnosis and on a diagnosis using smart vibration sensor which is described in article "Real-Time Predictive Maintenance – Artificial Neural Network Based Diagnosis".

In the mentioned articles, we discuss the applicability of diagnosis algorithms in real-time simulation environments by highlighting a specific case of how to implement the methods and perform diagnoses on unexpected behaviour. We obtained promising results encouraging for further research on the described diagnosis methods depending on the desired detection dimension, available resources, and model specifications. In addition, the diagnosis methods deliver the root cause affects which builds the basis for the research in self-adapting or self-healing systems to bring a system to a safe state if an unexpected behaviour is detected.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway. The work was co-funded by grants of Ministry of Education, Youth and Sports of the Czech Republic, by the Austrian Federal Ministry of Transport, Innovation and Technology (BMVIT) under the program

"ICT of the Future" between May 2019 and April 2022 (more information can be retrieved from https://iktderzukunft.at/en/). The work was also supported by the infrastructure of RICAIP that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 857306 and from Ministry of Education, Youth and Sports under OP RDE grant agreement No CZ.02.1.01/0.0/0.0/17_043/0010085.

References

[1] Wang, J., Liang, Y., Zheng, Y., Gao, R. X., and Zhang, F., "An integrated fault diagnosis and prognosis approach for predictive maintenance of wind turbine bearing with limited samples," Renewable Energy, vol. 145, pp. 642-650, 2020. Available online at: https://www.sciencedirect.com/science/article/pii/S0960148119309371

[2] Song, Z., Zhang, Z., Jiang, Y., and Zhu, J., "Wind turbine health state monitoring based on a bayesian data-driven approach," Renewable Energy, vol. 125, pp. 172-181, 2018. Available online at: https://www.sciencedirect.com/science/article/pii/S0960148118302404

[3] Wang, J., Wang, P., and Gao, R., "Enhanced particle filter for tool wear prediction," 2015.

[4] Sept. 2018, SURVEY Predictive Maintenance 4.0 Beyond the hype: PdM 4.0 delivers results, Mainnovation.

[5] Towards an open-standards based framework for achieving condition-based predictive maintenance, Kaur, Karamjit & Selway, Matt & Grossmann, Georg & Stumptner, Markus & Johnston, Alan, (2018) Proceedings of the 8th International Conference on the Internet of Things.

[6] Peng, Y., Dong, M., Zuo, M., "Current status of machine prognostics in condition-based maintenance: a review," 2010.

[7] I. a. F. M. a. N. R. Sarker, "AI-Driven Cybersecurity: An Overview, Security Intelligence Modeling and Research Directions," SN Computer Science, vol. 2, 2021.

[8] T. a. B. M. a. P. V. Ademujimi, "A Review of Current Machine Learning Techniques Used in Manufacturing Diagnosis," pp. 407-415, 2017.

[9] M. a. D. F. a. K. M. a. M. O. Barakat, "Self adaptive growing neural network classifier for faults detection and diagnosis," Neurocomputing, vol. 74, pp. 3865-3876, 2011.

[10] M. Demetgul, "Fault diagnosis on production systems with support vector machine and decision trees algorithms," The International Journal of Advanced Manufacturing Technology, vol. 67, 2012.

[11] J. a. K. M. a. P. J. Han, Data mining concepts and techniques, third edition, 2012.

[12] M. a. K. M. a. B. E. Mohammed, Machine Learning: Algorithms and Applications, 2016.

[13] L. P. a. L. M. L. a. M. A. W. Kaelbling, "Reinforcement learning: A survey," Journal of Artificial Intelligence Research, vol. 4, pp. 237-285, 1996.

[14] R. Liu, B. Yang and E. Zio: Artificial intelligence for fault diagnosis of rotating machinery: A review. Mech. Syst. Signal Process., vol. 108, aug 2018: p. 33–47, ISSN 10961216, doi:10.1016/j.ymssp.2018.02.016. URL https://linkinghub.elsevier.com/retrieve/pii/S0888327018300748.

[15] A. Chen, F. H. Liu and S. D. Wang: Data reduction for real-time bridge vibration data on edge. In Proc. - 2019 IEEE Int. Conf. Data Sci. Adv. Anal. DSAA 2019, Institute of Electrical and Electronics Engineers Inc., oct 2019, ISBN 9781728144931, p. 602–603, doi: 10.1109/DSAA.2019.00077.

[16] S. Yin, X. Li and H. Gao: Data-based techniques focused on modern industry: An overview. IEEE Trans. Ind. Electron., vol. 62, issue. 1, 2015: s. 657–667, ISSN 02780046, doi:10.1109/TIE.2014.2308133. URL https://ieeexplore.ieee.org/document/6748057/.

[17] S. Zhang, S. Zhang and B. Wang: Deep Learning Algorithms for Bearing Fault Diagnostics - A Comprehensive Review. IEEE Access, vol. 8, jan 2020: p. 29857–29881, ISSN 21693536, doi:10.1109/ACCESS.2020.2972859, 1901.08247. URL http://arxiv.org/abs/1901.08247

[18] H. Qiao, T. Wan and P. Wang: An Adaptive Weighted Multiscale Convolutional Neural Network for Rotating Machinery Fault Diagnosis under Variable Operating Conditions. IEEE Access, aug 2019: p. 118954–118964, ISSN 21693536, doi:10.1109/ACCESS.2019.2936625.

[19] Davis, R.: Diagnostic reasoning based on structure and behavior. Artificial Intelligence 24, pp. 347–410, 1984.

[20] Reiter, R.: A theory of diagnosis from first principles. Artificial Intelligence 32(1), pp. 57–95, 1987.

[21] de Kleer, J., Mackworth, A.K., Reiter, R.: Characterizing diagnosis and systems. Artificial Intelligence 56, 1992.

[22] Kaufmann, D., Nica, I., Wotawa, F.: Intelligent agents diagnostics - enhancing cyber-physical systems with self-diagnostic capabilities. Advanced Intelligent Systems, 2021. DOI https://doi.org/10.1002/aisy.202000218.

[23] Wotawa, F.: Reasoning from first principles for self-adaptive and autonomous systems. In: E. Lughofer, M. Sayed-Mouchaweh (eds.) Predictive Maintenance in Dynamic Systems - Advanced Methods, Decision Support Tools and Real-World Applications. Springer (2019). DOI 10.1007/978-3-030-05645-2.

[24] Wotawa, F.: Using model-based reasoning for self-adaptive control of smart battery systems. In: M. Sayed-Mouchaweh (ed.) Artificial Intelligence Techniques for a Scalable Energy Transition – Advanced Methods, Digital Technologies, Decision Support Tools, and Applications. Springer (2020).

[25] P. Jantscher: AI In Sensors For IoT [online] Cited 15.6.2021 Available from: https://siliconsemiconductor.net/article/106227/AI_In_Sensors_For_IoT.

[26] S. C. Mukhopadhyay, S. K. S. Tyagi, N. K. Suryadevara, V. Piuri, F. Scotti and S. Zeadally: Artificial Intelligence-based Sensors for Next Generation IoT Applications: A Review, in IEEE Sensors Journal, doi: 10.1109/JSEN.2021.3055618.

[27] A review on machinery diagnostics and prognostics implementing condition-based maintenance, Jardine, Andrew & Lin, Daming & Banjevic, Dragan, Mechanical Systems and Signal Processing, 2006.

[28] A.K.S. Jardine, Optimizing condition based maintenance decisions, in: Proceedings of the Annual Reliability and Maintainability Symposium, 2002, pp. 90–97.

[29] W. Wang, J. Sharp, Modelling condition-based maintenance decision support, in: Condition Monitoring: Engineering the Practice, Bury St Edmunds, 2002, pp. 79–98.

[30] I. Howard, S. Jia, J. Wang, The dynamic modelling of a spur gear in mesh including friction and a crack, Mechanical Systems and Signal Processing 15 (2001) 831–838.

[31] W.Y. Wang, Towards dynamic model-based prognostics for transmission gears, in: Component and Systems Diagnostics, Prognostics, and Health Management II, vol. 4733, Bellingham, 2002, pp. 157–167.

[32] D.C. Baillie, J. Mathew, Nonlinear model-based fault diagnosis of bearings, in: Proceedings of an International Conference on Condition Monitoring, Swansea, UK, 1994, pp. 241–252.
[33] K.A. Loparo, M.L. Adams, W. Lin, M.F. Abdel-Magied, N. Afshari, Fault detection and diagnosis of rotating machinery, IEEE Transactions on Industrial Electronics 47 (2000) 1005–1014.
[34] K.A. Loparo, A.H. Falah, M.L. Adams, Model-based fault detection and diagnosis in rotating machinery, in: Proceedings of the Tenth International Congress on Sound and Vibration, Stockholm, Sweden, 2003, pp. 1299–1306.
[35] A.S. Sekhar, Model-based identification of two cracks in a rotor system, Mechanical Systems and Signal Processing 18 (2004) 977–983.
[36] G.H. Choi, G.S. Choi, Application of minimum cross entropy to model-based monitoring in diamond turning, Mechanical Systems and Signal Processing 10 (1996) 615–631.
[37] W. Bartelmus, Mathematical modelling and computer simulations as an aid to gearbox diagnostics, Mechanical Systems and Signal Processing 15 (2001) 855–871.
[38] R.J. Hansen, D.L. Hall, S.K. Kurtz, A new approach to the challenge of machinery prognostics, Journal of Engineering for Gas Turbines and Power 117 (1995) 320–325.
[39] A. Vania, P. Pennacchi, Experimental and theoretical application of fault identification measures of accuracy in rotating machine diagnostics, Mechanical Systems and Signal Processing 18 (2004) 329–352.
[40] Dousson, C., Gaborit, P., & Ghallab, M. 'Situation recognition: representation and algorithms', Proc. 13th IJCAI, Chambery, France, 1993.
[41] Ghallab, M., & Philippe, H. 'A Compiler for Real-Time Knowledge-Based Systems'. Proc. IEEE International Symposium on AI for Industrial Applications, 287-293, 1988.
[42] Bousson, K., & Trave-Massuyes, L. 'Fuzzy Causal Simulation in Process Engineering'. IJCAI-93, Chambery, France. August-September 1993.
[43] Kallen MJ, Van Noortwijk JM. Optimal maintenance decisions under imperfect inspection. Reliab Eng Syst Saf 2005;90(2–3):177–85.
[44] Peng C-Y, Tseng S-T. Mis-specification analysis in linear degradation models. IEEE Trans Reliab 2009;58(3):444–55.

[45] Ye Z-S, Wang Y, Tsui K-L, Pecht M. Degradation data analysis using wiener processes with measurement errors. IEEE Trans Reliab 2013;62(4):772–80.
[46] Zio E, Compare M. Evaluating maintenance policies by quantitative modeling and analysis. Reliab Eng Syst Saf 2013;109:53–65.
[47] Maintenance management of wind power systems using condition monitoring systems – life cycle cost analysis for two case studies, Nilsson J, Bertling L., 2007.
[48] Asset life cycle management: towards improving physical asset performance in the process industry, Schuman CA, Brent AC, 2005
[49] Two probabilistic life-cycle maintenance models for deteriorating civil infrastructures, Van Noortwijk JM, Frangopol DM, 2004.
[50] Wang W, Christer AH. Towards a general condition based maintenance model for a stochastic dynamic system. J Oper Res Soc 2000;51(2): 145–55.

1.5

Real-Time Predictive Maintenance – Model-Based, Simulation-Based and Machine Learning Based Diagnosis

Franz Wotawa[1], David Kaufmann[1], Adil Amukhtar[1], Iulia Nica[1], Florian Klück[2], Hermann Felbinger[2], Petr Blaha[3], Matus Kozovsky[3], Zdenek Havranek[3] and Martin Dosedel[3]

[1]Graz University of Technology, Austria
[2]AVL List GmbH, Austria
[3]Brno University of Technology CEITEC, Czech Republic

Abstract

Predictive maintenance focuses on forecasting faulty or unwanted behaviour and defines appropriate countermeasures to be taken. Diagnosis, i.e., the detection of failures, the identification of faults, and repair provides useful foundations for predictive maintenance. In this article, we show how diagnosis, and in particular model-based, simulation-based and machine learning based diagnosis, can be used in practice. For this purpose, we introduce a simplified DC e-motor simulation model with the capability of fault injection to be used to show the efficiency of the introduced diagnosis methods based on the model's behaviour. A simulation run of the system under test with pre-defined injected faults during runtime is used to validate the results obtained by the diagnosis methods. The results outline a promising application of these diagnosis methods for industrial applications, since each algorithm shows a time efficient and reliable diagnosis in relation to find the root cause of an observed faulty behaviour within the model. Further, the root cause analysis, performed with the introduced diagnosis methods, offers an excellent starting point for future development of self-adapting systems.

Keywords: abstract model, AI based diagnosis, AI based predictive maintenance, digital twin, model-based diagnosis, machine learning, simulation-based diagnosis, reliability, validation, fault model, fault injection, root cause analysis.

1.5.1 Introduction and Background

In this article, we focus on the application of model-based and machine learning-based diagnosis outlined in the article "Foundations of Real-Time Predictive Maintenance with Root Cause Analysis" making use of an e-motor use case. In the foundations, we already discussed the underlying background, and an architecture of a real-time diagnosis tool for detecting root causes of faults based on different diagnosis methods, which distinguish in the applied methodologies.

Besides providing more information regarding the application of the different diagnosis methods, we want to solve the question of whether model-based reasoning can be used for obtaining explanations for the given models, a simplified DC motor model with the capability of fault injection was developed to capture the individual ideas of diagnosis tools.

The first approach, model-based diagnosis, considers an abstract model that can be represented as logical rules for diagnosis. This model captures the abstract values for quantities/signals. Using an abstraction function, it is possible to map given values to their abstract representation. The second approach, simulation-based diagnosis, utilizes simulation models directly. A pre-requisite is that the models not only capture the correct behaviour but also faulty behaviour like the influence of different parameters on the behaviour. The last approach, an AI-based diagnosis model also uses simulation models to gather information about the behaviour based on different parameters of the system. The produced knowledge base is further used to train a model. After the training, it is plausible to evaluate the feasibility of the diagnosis model in terms of decision process optimization in real time.

In summary, we deal with the following diagnosis methods in this article:

- Model-based diagnosis
    - Abstract model represented with logic rules.
    - Diagnosis based on state change observations.
    - Classify the model in components and identify a normal or abnormal behaviour of each.
- Simulation-based diagnosis

  - Detailed simulated system model ("digital twin") with the capability of fault injection.
  - Use of simulation models directly.
  - Generate labelled reference data by simulating the model with health and fault condition in different scenarios.
  - Real-time diagnosis of observed model values based on pre-simulated reference data.
- AI-based diagnosis
  - Machine learning to diagnose unexpected behaviour.
  - Artificial neural networks to diagnose and predict fault behaviour.
  - Physical model (digital twin) with the capability of fault injection to produce training data.
  - Train and evaluate an AI based model on collected labelled reference data to detect fault behaviour in real-time within cyber-physical systems.

For all diagnosis methods, the assumption is to find faults occurring at runtime. To show the architecture and applicability of these approaches, the focus is on describing the mechanism and show the results based on examples to highlight the idea, the problems, and solutions. In the following section a simplified DC e-motor model with the capability of fault injection is described and the obtained results based on different diagnosis method implementations are demonstrated.

1.5.2 Application of Diagnosis Systems Based on Simplified DC e-Motor Model

In this section we introduce a developed simplified DC e-motor model with fault injection capability in all used components which comprises the battery, switch, resistor, load on the motor and the e-motor parts. The ability of fault injection is used to discuss three different diagnosis algorithms to detect faults in a system based on the simplified DC e-motor. The first promising approach is the model-based diagnosis algorithm. The model-based diagnosis system uses logic to represent the e-motor to perform model-based reasoning to search for diagnosis candidates given an unexpected behaviour caused by faults in the system. The second introduced diagnosis system is based on simulation which uses digital twin models of the e-motor directly to simulate faults in advance to use the generated data to find correlations with

the original system. The last approach deals with machine learning using gathered fault data of the e-motor model to train the system to detect faulty behaviour.

1.5.2.1 Simplified DC e-Motor Model With Fault Injection Capabilities

The proposed use case of a DC e-motor comprises a battery, a switch for turning the motor on and off, a resistor, which we may use to adapt the voltage provided to the motor, the e-motor, and a load attached to the motor. In Figure 1.5.1, we find the schematics of the motor that also comprises the internals of the battery and the motor. We assume that the battery comprises an internal resistance, the motor resistance, inductance, as well as a part coupling the electric components to mechanic ones. For the model we consider a brushed e-motor comprising a wound rotor and a permanent-magnetic stator. The rotational speed of the motor is proportional to the voltage applied and its torque is proportional to the applied current. Table 1.5.1 shows a list of all components with the applicable health states including faults that can be set during runtime to simulate different behaviour of the e-motor. The DC e-motor simulation model is built with the equation-based language Modelica to simulate the complex physical system. For the diagnosis approaches based on this model, we use the simulated outputs

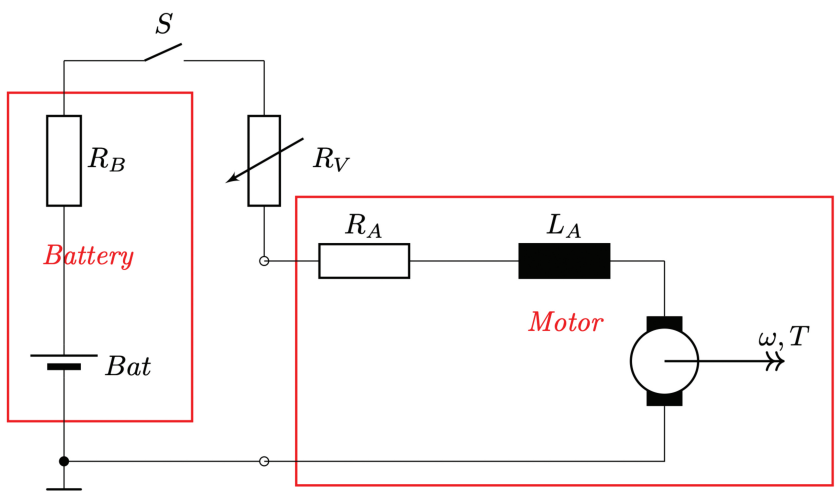


Figure 1.5.1 Simplified DC e-motor circuit.

Table 1.5.1 Simplified DC motor component state description.

Component	Health state	Description
Motor	*ok*	Ordinary behaviour of a motor given by its internal components and the equations provided for DC motors allowing to map electrical quantities to mechanical ones.
	f_1	In this fault mode we assume that 1/3 of the resistor and inductivity values is lost.
	f_2	In this fault mode we assume that 2/3 of the resistor and inductivity values is lost.
Load	*ok*	The load applied to the motor is set to its normal value.
	empty	There is no load anymore applied to the motor.
	f_1	The load is 50% higher than its normal value.
	f_2	The load is 50% lower than its normal value.

directly or a generated FMU (Functional Mockup Unit) from the model to be able to run simulations of the DC e-motor in other programming environments.

1.5.2.2 Model-based Diagnosis for Simplified DC e-Motor

In the following, we outline the use of model-based diagnosis for the identification of root causes. For this purpose, we discuss the necessary steps required to diagnose the simplified DC motor use case depicted in Figure 1.5.1. Specifically, we consider the following faulty case where a certain load is higher than expected, indicated as load fault f_1 in Table 1.5.1. In Figure 1.5.2 we depict the behaviour of the e-motor when switching it on without load (*empty)*, with the expected load (*ok*), and the higher load (f_1). We see that when switching on the motor during time 0.5 and 1.5 seconds, there is a deviation between the observed rotational velocity and the current drawn to drive the e-motor in all three cases. Although this deviation is not that high between the ordinary "normal load" scenario and the "high load" scenario, as it can be observed.

We require observations and a logic model for computing diagnoses. The observations in the case of model-based diagnosis are assumed to be available at certain points in time where we probe the system. In Figure 1.5.2, we consider 3 probing time points ❶, ❷, and ❸ at time steps 0.25, 1.00, and 1.75 seconds respectively. In ❶ there is no difference between the three observed signals. In ❷ we see that both the rotational velocity as well as the current are different when comparing "normal load" with the "high load" scenario. In the "high load" scenario, the velocity is lower and the absolute

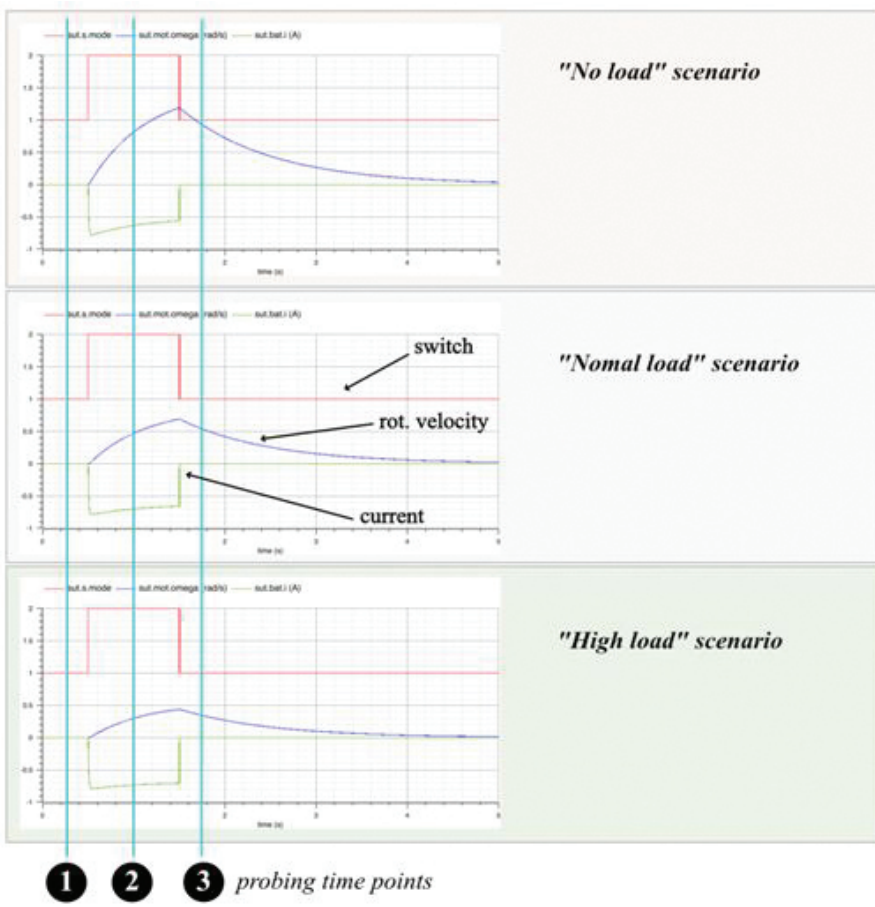


Figure 1.5.2 Simplified DC e-motor diagnosis observations used for model-based diagnosis.

value of the current is slightly higher. In time step ❸, only the velocity is still lower for "high load".

Such deviations can be obtained automatically comparing a simulation run considering the e-motor to work as expected with observations obtained from monitoring the real e-motor implementation. Deviations trigger diagnosis and the question is how a model of the e-motor example can be utilized for obtaining the root cause responsible for the behavioural differences observed. For diagnosis, we will map the deviations or values to their corresponding logic representation. But before discussing this issue, we have a look at modelling for diagnosis.

We have to formalize the behaviour of components and their interconnections. For the behaviour, we use rules of the form $\neg ab(C) \rightarrow behav(C)$. A battery component, for example, can be easily formalized stating that in case of correct behaviour, it is delivering power using the following logic rule:

$$\neg ab\,(C) \wedge type\,(C, battery) \rightarrow val(pow\,(C)\,, nominal) \tag{1.5.1}$$

The predicate $type$ is used to say that component C is of a type, e.g., $battery$. The predicate val is for stating a value, e.g., $nominal$, for a component port, e.g., pow. In addition, we may also formalize that a malfunctioning battery is not delivering any electricity, i.e.:

$$ab\,(C)\ \wedge type\,(C, battery) \rightarrow val(pow\,(C)\,, zero) \tag{1.5.2}$$

We can do the same for switches, resistors, and the motor. A switch if being switched-on provides electricity (but only if there is electricity at one port). If switched-off no electricity is provided. A resistor is for passing electricity, and a motor makes use of provided electrical power to speed-up its rotor. Depending on the load the velocity reached can be higher or lower, requiring less or more power.

$$\begin{gathered}
\neg ab\,(C) \wedge type\,(C, switch) \wedge on\,(S) \wedge val\,(inpow\,(C)\,, V) \\
\rightarrow val(outpow\,(C)\,, V) \\
\neg ab\,(C) \wedge type\,(C, switch) \wedge on\,(S) \wedge val\,(outpow\,(C)\,, V) \\
\rightarrow val(inpow\,(C)\,, V) \\
\neg ab\,(C) \wedge type\,(C, switch) \wedge off\,(S) \\
\rightarrow val(outpow\,(C)\,, zero) \\
ab\,(C) \wedge type\,(C, switch) \rightarrow val(outpow\,(C)\,, zero)
\end{gathered} \tag{1.5.3}$$

$$\begin{gathered}
\neg ab\,(C) \wedge type\,(C, resistor) \wedge val\,(inpow\,(C)\,, V) \\
\rightarrow val(outpow\,(C)\,, V) \\
\neg ab\,(C) \wedge type\,(C, resistor) \wedge val\,(outpow\,(C)\,, V) \\
\rightarrow val(inpow\,(C)\,, V) \\
ab\,(C) \wedge type\,(C, resistor) \rightarrow val(outpow\,(C)\,, zero)
\end{gathered} \tag{1.5.4}$$

$$\begin{gathered}
\neg ab\,(C) \wedge type\,(C, motor) \wedge val\,(inpow\,(C)\,, V) \rightarrow val(outpow\,(C)\,, V) \\
\neg ab\,(C) \wedge type\,(C, motor) \wedge val\,(inpow\,(C)\,, V) \rightarrow val(speed(C), V) \\
\neg ab\,(C) \wedge type\,(C, motor) \wedge val\,(speed\,(C)\,, V) \rightarrow val(inpow\,(C)\,, V) \\
ab\,(C) \wedge type\,(C, motor) \rightarrow \neg val(speed\,(C)\,, nominal) \\
ab\,(C) \wedge type\,(C, motor) \rightarrow \neg val(outpow\,(C)\,, nominal)
\end{gathered} \tag{1.5.5}$$

Note that in the above model we do not distinguish values to be higher or lower than expected. Instead, we state that the speed (power requested) is not allowed to be nominal in case of a fault in the e-motor. This formalization captures the faulty behaviour required for diagnosis in the mentioned use case. However, we are also able to come up with a model considering different faulty states (and not only ab).

The described model formalizes the behaviour of the components. What is missing, is the description of the structure of the system. In our case, we have 4 components, i.e., a battery b, a switch s, a resistor r, and a motor m, that are directly connected. We first, declare the components via stating logical facts:

$$type\,(b, battery) \wedge type(s, switch) \wedge type(r, resistor) \wedge type(m, motor) \tag{1.5.6}$$

Afterward, we define the connections between the components using the predicate $conn$:

$$\begin{array}{c} conn\,(inpow\,(s)\,, pow\,(b)) \wedge conn(outpow\,(s)\,, inpow\,(r)) \\ conn\,(outpow\,(r)\,, inpow\,(m)) \end{array} \tag{1.5.7}$$

To complete the formalization, we state that values are transferred via a connection (in both directions), and that it is not allowed to have different values on any connection:

$$\begin{array}{l} val\,(X, V) \wedge conn\,(X, Y) \rightarrow val(Y, V) \\ val\,(Y, V) \wedge conn\,(X, Y) \rightarrow val(X, V) \\ \neg(val\,(X, V) \wedge val\,(X, W) \wedge V \neq W) \end{array} \tag{1.5.8}$$

This logic model can be now used for diagnosis. Note that in this context a diagnosis is a setting of health states to components. Hence, we are interested in assigning either ab or $\neg ab$ to any component, e.g., in our case b, s, r, and m considering the given observations. In Table 1.5.2, we summarised the diagnosis results obtained when using the diagnosis engine described in [2] and the model introduced in this section. Based on the foundations elaborated in [2] we introduced a more complex physical system also taking the factor time into consideration for observation to show the efficiency of the developed diagnosis method for a broader field of application.

We see that in case of the second and third observations, we only obtain the motor being responsible for the deviation between the expected and the observed values. Note that this – because of the formalization – states that the load is higher than expected. The required diagnosis time was less

Table 1.5.2 Diagnosis results obtained using model-based diagnosis.

Section	Observation	Diagnosis
❶	$off\,(s)$ $\wedge\, val\,(pow\,(b)\,, nominal)$ $\wedge\, val\,(speed\,(m)\,, zero)$ $\wedge\, val(outpow\,(m)\,, zero)$	$\neg ab(b) \wedge \neg ab(s) \wedge \neg ab\,(r) \wedge \neg ab(m)$
❷	$on\,(s)$ $\wedge\, val\,(pow\,(b)\,, nominal)$ $\wedge\, \neg val\,(speed\,(m)\,, zero)$ $\wedge\, \neg val\,(speed\,(m)\,, nominal)$ $\wedge\, \neg val(outpow\,(m)\,, zero)$ $\wedge\, \neg val\,(outpow\,(m)\,, nominal)$	$\neg ab(b) \wedge \neg ab(s) \wedge \neg ab\,(r) \wedge ab(m)$
❸	$on\,(s)$ $\wedge\, val\,(pow\,(b)\,, nominal)$ $\wedge\, \neg val\,(speed\,(m)\,, zero)$ $\wedge\, \neg val\,(speed\,(m)\,, nominal)$ $\wedge\, val(outpow\,(m)\,, zero)$	$\neg ab(b) \wedge \neg ab(s) \wedge \neg ab\,(r) \wedge ab(m)$

than 0.0021 seconds for all observations. It is also worth noting that the observations represent our knowledge. For ❷ we know that the speed and the power consumption (the latter represented using the port $outpow$) are both not nominal and also not zero. Similarly, we represent the observations for ❸.

In summary, the provided model was able together with the given observations to come up with the expected solution. No other single fault diagnoses were obtained in any case. Modelling relied on the assumption of the particular fault case, and the transfer of power through the circuit. This simplified model may not be appropriate in all cases. Diagnosis time was very short making the approach feasible for this kind of application having a limited smaller number of components and taking care of simple models. Modelling, however, has always been an issue and more sophisticated models are maybe required for other application scenarios. The presented approach assumes that a simulation model (a-kind-of a digital twin) is running concurrently for allowing to generate observations.

1.5.2.3 Simulation-Based Diagnosis for Simplified DC e-Motor

The idea behind the simulation-based diagnosis is to make use of digital twin models to simulate pre-configured faulty behaviour and thus find correlations with the original cyber-physical system's measured values, which allows to

diagnose faults as well as fault combinations and the correlated root causes of a physical system.

To diagnose a physical system with a simulation-based approach we use a system to induce fault modes to measure relevant signals to gather information about the behaviour under these configurations. This can be performed on a real system or at least on a digital twin (simulation model) with the ability to perform fault injection and output all relevant measured signals. To use the knowledge about the behaviour under fault conditions is the main idea of the simulation-based diagnosis approach. This leads us to the question of how to use the measured information to detect a fault system and additional to diagnose the root cause of such a faulty behaviour?

The main part of the simulation-based diagnosis approach is a precise cyber physical model, or a simulation model of the physical system (digital twin) with the capability of fault injection. Besides that, the algorithm itself is categorized into three subsections. First the reference data, a pre-simulated fault data generation for the diagnosis, second the model signals processing, a preparation phase of the measured model signals on which the diagnosis is performed and last the diagnosis phase where the measured data is brought into comparison with the pre-simulated fault reference data to find the best correlation and compute diagnoses to explain the actual system behaviour.

To evaluate the simulation-based diagnosis approach on the DC e-motor model we use the FMU version of the simulation model running in a python environment instead of a real system to produce fault reference data for the diagnosis method. In addition, we used another instance of the simulated DC e-motor model as a system to be diagnosed. We set the focus on the diagnosis of the faults in regards to the motor and torque parameter. As stated in Table 1.5.1 the motor and load state can be set with different modes. However, for the validation we concentrate on the specific faults as *empty*, f_1 and f_2 for the motor as well as f_1 and f_2 for the torque. In addition, we use the *ok* state to diagnose a health system as a reference to a faulty system.

$$\begin{aligned} load_{state} &\in \{ok,\ empty,\ f_1,\ f_2\} \\ motor_{state} &\in \{ok,\ f_1, f_2\} \end{aligned} \tag{1.5.9}$$

To generate the reference data for diagnosis, the simulation is configured with the option to inject faults at runtime. To generate a reference dataset with a broad range of different scenarios and signal behaviour characteristics, the fault states f_x ($load_{state}$, $motor_{state}$) are injected at various time points and initial parameters of the DC motor model simulation. Besides the single fault injection, also all possible combinations of faults are considered to cover

most of the feasible fault diagnosis. While performing the simulations with fault injection, we measure the most significant signals $\overrightarrow{x_s}$ (1.5.10) of the e-motor model as the battery current i and voltage u, the motor rotation speed ω and angular acceleration α at a sampling rate of 0.001 seconds.

$$x_s\,(t) \in \{i(t),\ u(t),\ \omega(t),\ \alpha(t)\} \tag{1.5.10}$$

Next the observations $\overrightarrow{x_s}$ are processed with a moving average method (see equation (1.5.11)) on a time window Δw of 0.05 seconds. With the moving average we obtain an averaged signal value for each time step we simulate.

$$x_r\,(t) = \frac{1}{n}\sum_{i=0}^{n} x_s(t-i)\ ,\ n := \Delta w \tag{1.5.11}$$

The average is built since we perform the diagnosis based on an averaged time window Δw to avoid losing information during state changes and quick responses. The averaged reference data $\overrightarrow{x_r}$ (1.5.12) is stored in a table for later usage in the diagnosis algorithm. Since all fault states f_x are known for every measurement, we obtain a labelled dataset as reference data. The corresponding state information is appended to the reference data in the table.

$$\overrightarrow{x_r} = \left[\ x_{r0}\ \ \ldots\ \ x_{rn}\ \ ,\ \ \boldsymbol{f_{load}}\ \ \boldsymbol{f_{motor}}\ \right] \tag{1.5.12}$$

After generating the labelled reference data $\overrightarrow{x_r}$, we can run the DC motor simulation with the option to measure the signals $\overrightarrow{x_s}$, as mentioned before with a sampling rate of 0.001 seconds. For the diagnosis we constantly store the latest signal values within the same time window Δw length as selected for the reference data (0.05 seconds). By selection of an equal-sized time window, it is possible to make a direct comparison on the reference and measured data. The diagnosis is requested continuously within a time interval of 0.4 seconds. With every request, the latest measured signals are averaged at the request time point equal to equation (1.5.11) and result in $\overrightarrow{x_m}$. The generated averaged measured signal vector $\overrightarrow{x_m}$ is further used in the diagnosis process.

Within the diagnosis process the highest correlation between the averaged measured signals and the averaged reference data is searched. After the global minimum of the deviation is found, the related reference signal $\overrightarrow{x_r}$ is read out to get access to the parameter states f_x used as a label for the reference data. Finally, the identified states f_x (f_{load} and f_{motor}) are returned as the actual diagnosis. Figure 1.5.3 shows a detailed description of the complete diagnosis

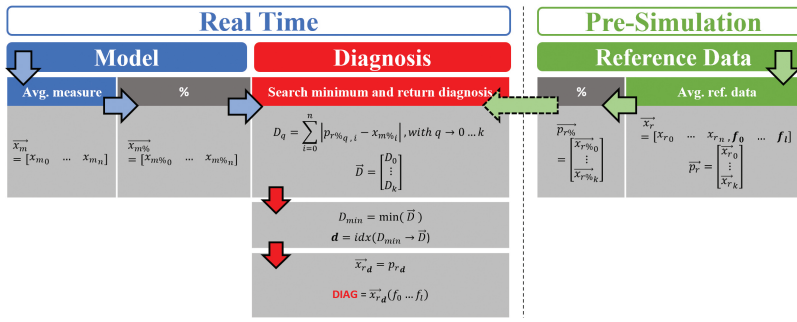


Figure 1.5.3 Simulation-based diagnosis algorithm description.

algorithm equations for the search process starting with a triggered diagnosis based on the averaged measured signals $\overrightarrow{x_m}$ and reference signals $\overrightarrow{x_r}$.

Figure 1.5.4 illustrates the e-motor simulation, where the first three graphs show the battery current flow, the motor angular velocity and the angular acceleration over a time of 5 seconds. In addition, the markers indicate a diagnosis request of the system, whereby a green dot depicts a health system and a red cross means that a fault is detected at this point by the diagnosis algorithm. The last graph describes the actual set of states f_x in the e-motor simulation (blue rectangle) and the system diagnosis (red arrow), whereby the diagnosis holds until a change in diagnosing is recognized.

We see that the system starts at health conditions (ok). After the first second the load fault f_1 (high load) is injected into the DC e-motor simulation. The algorithm recognizes the fault and returns the correct diagnosis. At the time of 2 seconds, the system is brought back to health conditions for 1 second when the motor state is set to fault f_2 (66% inductivity and resistor value lost). Since this fault is injected within a transient zone where no diagnosis request is triggered, the fault is detected with the next diagnosis request what explains the time delay of the diagnosis. With a higher diagnosis request rate, we obtain faster results, but this is limited in terms of real time diagnosis and the necessary computation time. The last fault injection into the system is a combined fault, it consists of a motor fault f_2 and a load fault f_1. The fault is again recognized and diagnosed correctly by the algorithm. We see again a delay between the injection and the computed diagnosis, due to the selected diagnosis interval of 0.4 seconds.

From this we conclude that the simulation-based diagnosis system is worth to be considered for further research since the overall algorithm is easy to implement and the system is robust in detecting different kinds of

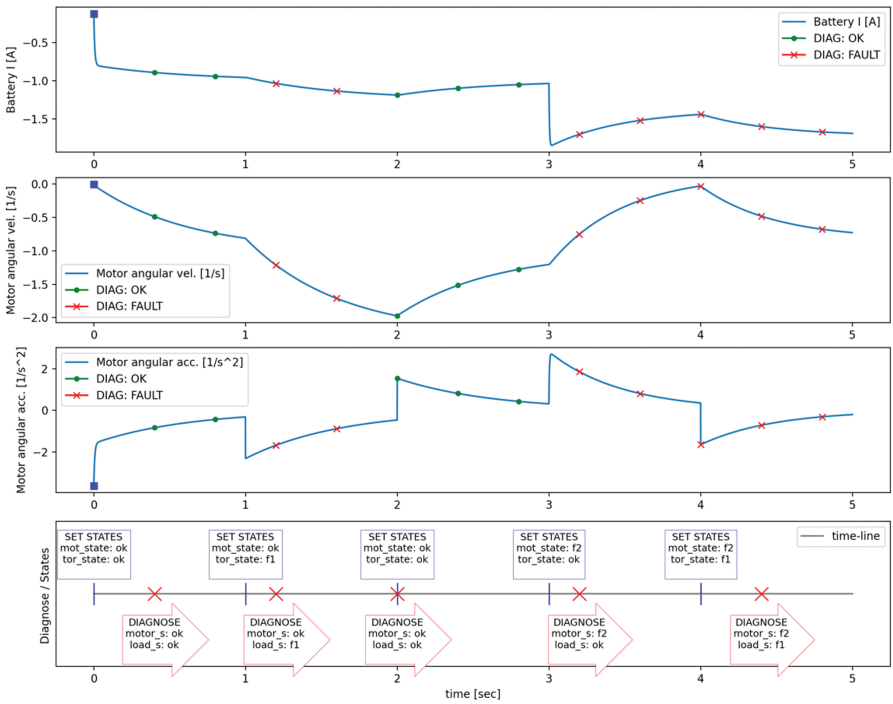


Figure 1.5.4 Simplified DC e-motor diagnosis observation with simulation-based diagnosis.

faults and fault combinations if the faults to be diagnosed are known and the digital twin is able to simulate the behaviour precisely enough. Weaknesses are zones where different faults can raise similar characteristics during the initial phase that may result in wrong diagnoses. The reference table can also cause problems in terms of storage space, if too many different faults and fault combinations need to be diagnosed, which also has a direct negative impact on the computation time. Since we are only interested in short sections starting with the fault injection and ending when signals reach a certain equilibrium level, the storage of reference data is minimized.

1.5.2.4 Machine Learning for Diagnosis of Simplified DC e-Motor

As already discussed, that machine learning algorithms are not unfamiliar with the domain of fault diagnosis. There exist classes of algorithms e.g., Support vector machines, Decision Tree, K-Means, etc., which can be utilized to solve a complex problem related to fault diagnosis. For this case study, fault

diagnosis of DC motor, we modelled the fault diagnosis problem with one of the ensembles-technique-based machine learning algorithms called *Bootstrap Aggregation (Bagging)*. We have multiple fault states of the simplified DC motor for the diagnosis. Hence, we will use a classifier model for the classification task. We already have faulty behaviour simulated for each faulty state, that's why we adopted a supervised methodology to train the model. Now we will discuss the methodology for the fault diagnosis using machine learning in more detail.

In a classification problem, machine learning models take input of predictor variables corresponding to the dependent variable. In this case study, we have nine predictor variables and two dependent variables namely $load_{state}$ and $motor_{state}$ of the e-motor system. Dependent variable $load_{state}$ and $motor_{state}$ have a set of categorical labels defined as shown in (1.5.9). In order to transform the problem into the multi-classification problem, we combined the $load_{state}$ and $motor_{state}$ and introduced a new variable called *target* defined as follows:

$$target{=}load_{state}\ .motor_{state} \tag{1.5.13}$$

Finally, the distribution of the dependent variable is almost equally divided into faulty and non-faulty categories i.e., 42% and 58% respectively. As the dependent variable has the sequence of values across predictor variables, therefore, a machine learning model can be trained to learn the underlying pattern associated with each state of the system. As there are more than two states, a multi-classification model is trained as opposed to binary classification. Furthermore, as machine learning models require data to be numeric, we encoded the dependent variable with a label encoder in order to train and evaluate the model performance. Label encoder simply assigns a unique numeric integer value to a categorical label.

As discussed earlier that we used *Bagging* algorithm for the modelling, Random Forest is one of the machine learning models from *Bagging* classifiers. Random Forest is a bagging technique that simply combines (average) the outcome of multiple models and makes more accurate prediction than one model.

Model selection is one of the important and crucial parts of the training. The main reason to select *Random Forest* is that it performs well on both large and small datasets, and it can select the best subset of features that perform better and adds more information into the modelling. There is a number of hyper parameters associated with most of the machine learning models which can be fine-tuned to achieve the best performance. In this case

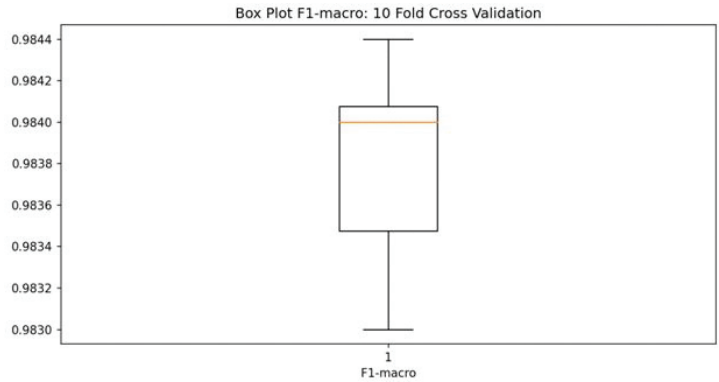


Figure 1.5.5 Box plot 10-fold cross validation.

study we used the important parameters for random forest i.e., *n_estmiators = 100, criterion = gini, min_samples_split = 2, min_samples_leaf = 1* etc. to train and evaluate the model. For the evaluation of the model, K-fold cross-validation is used where *k=10*. K-fold cross-validation is used to ensure that the model is not overfitting [1] and it generalizes well. In this setting, the dataset is randomly divided into *K* chunks, and *K* models are trained on each chunk. Each model is trained using *K-1* chunks and validated on the remaining dataset. Finally, as an evaluation metric, we used *F1-macro* (macro-averaged), used to assess the quality of the model for multiple classes. *F1-macro* is an average of label-wise F1 scores, whereas the F1 score is basically a harmonic mean of precision and recall. For each fold, the *F1-macro* is calculated, and then averaged score for 10-folds is used to evaluate the performance of the model. Once the model is passed through the validation process to estimate the overall performance, the final model is trained and tested on the test data. Please note that the test data was not part of the training and validation process.

Next, we will discuss the results obtained from the diagnosis using the machine-learning model. Figure 1.5.5. shows the distribution of *F1-macro* over the 10 folds. For each fold, our model performs well as there are no outliers. The average score for 10-fold cross-validation is **0.9838**, which shows that model was able to classify and detect the faults correctly, for most of the states.

Figure 1.5.6. shows the confusion matrix for the test data, where each cell along the diagonal represents the correct classification of diagnosis and the rest of the cells show the misclassifications predicted by the model. Results

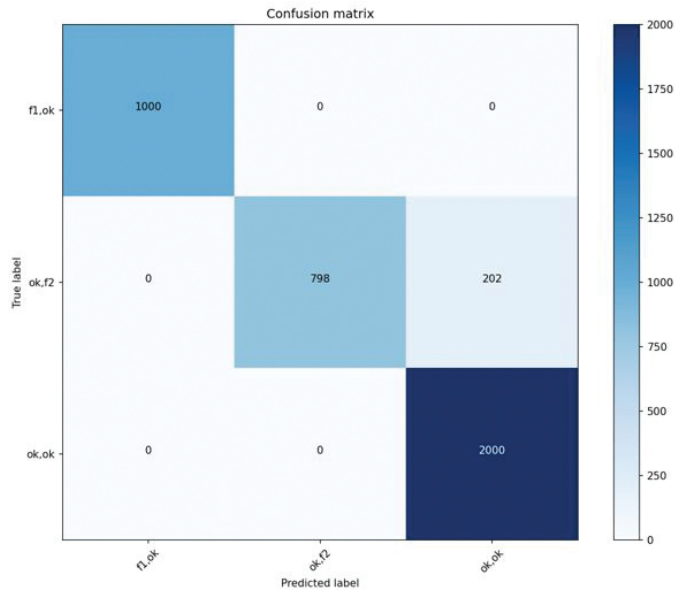


Figure 1.5.6 Normalized confusion matrix - model testing/verification.

suggest that the model was able to correctly diagnose two faults i.e. $\{f_1, ok\}$ and $\{ok, ok\}$, whereas we have very few misclassifications for faulty state $\{ok, f_2\}$. The *F1-macro* score for the test data is **0.9465**. Results indeed show that machine learning can be useful for the fault diagnosis of the e-motor.

From this, we conclude that we developed a machine learning algorithm to classify the faulty states of the DC e-motor, based on the sequence of variables. Results indeed suggest that machine learning algorithms have the potential to learn the fault behaviour of the DC e-motor system. Although, there are few misclassifications of the faulty states, but it is indeed part of the learning as learning cannot be perfect. Bagging algorithms take the decision from multiple models. Hence, these algorithms have the ability to perform well. It is important to note that learning highly depends on the quality of data, as the model learns from the underlying distribution of the data and correlations (if exist). Results also suggest that each faulty state has the learning curve associated with it, as a result, advanced machine-learning algorithms e.g., boosting, neural networks, and deep learning can be tested and evaluated for future work. As machine learning algorithms are not pre-programmed, it gives them the advantage over other traditional faulty diagnosis techniques.

1.5.2.5 Comparisons and Limitations

The used underlying methodology has some limitations due to the assumptions required. In the following section, the limitations and problems of each applied diagnosis algorithm on the DC e-motor model are summarized for comparison reasons.

The model-based diagnosis uses a detailed logic representation describing the model's components separately from the available simulation model. The logic models may be made for a particular purpose, e.g., hardware diagnose, and must be adapted to serve other goals, e.g., design diagnose of a system. Since the logic representation of complex cyber-physical systems is applied, the diagnosis is limited to observe state changes. Therefore, abstraction might not be that simple to map to real behaviour without ambiguity. Further, there is a lack of tools supporting the development and easily going through obtained results.

The simulation-based approach uses digital twin models directly to simulate healthy and faulty behaviour. The used parameter and obtained outputs from the simulation are analysed, processed, labelled and stored in a lookup table for further usage as a reference basis for the diagnosis search algorithm. The main requirement for this approach is to generate an accurate representation of the real system with the capability of fault injection. In addition, the faults and fault combinations must be previously defined to be able to diagnose them. With an increasing number of possible faults, limitation factors as computation time and storage space come to the fore. Another limitation is the adaptability to hardware or parameter changes in the real system, since a precise and realistic behaviour representation to obtain a correct diagnosis is needed.

Machine learning for diagnosis uses labelled reference data to train the system. Since the algorithm depends on the quality of the observed labelled data it is essential to have access to a precise simulated representation of the real system. Variability in the system hardware or parameter requires the machine learning model to be retrained which takes up an enormous amount of time. Models usually do not generalize well, and when deployed in real-time, results are affected by the data points which were not part of the training dataset. In addition, model selection is a crucial part of learning. Results may vary based on the model selected for the type of data, e.g., sequential and non-sequential and underlying distribution of the data. Further, if the labels of the fault type are not simulated properly the model will be biased towards the noise and the misclassification rate increases.

1.5.3 Conclusion

For the simplified DC e-motor, we introduced two types of components (motor, load) with the ability to inject faults as resistor and inductivity loss and a varying load factor. Based on this model, three methods, model-based diagnosis, simulation-based diagnosis and machine-learning diagnosis are introduced to be able to detect unexpected behaviour and outline its root cause. The model-based diagnosis method uses a logical representation of the simplified DC motor model to identify abnormal state changes. With this approach, we were able to come up with the expected solution in the particular case applying a high load fault during normal operation, still, the modelling complexity increases for more sophisticated models.

The simulation-based approach makes use of digital twin models directly to simulate normal and faulty behaviour to cover possible scenarios which are of interest for the diagnose part. The measurements and the corresponding state parameter are stored and used as reference data for the diagnosis search process during real-time observation of the simplified DC motor model. The simulation-based diagnosis approach delivers accurate diagnoses in real time with the limitation that only pre-simulated faults are considered to be diagnosed.

The last approach is the machine-learning diagnosis, which is capable to classify the faulty states based on the real-time measured signals of the model. As training data for the bagging algorithm, we use the simulated labelled reference data from a simulation-based approach which already covers different behaviours caused by fault injection. The machine-learning diagnosis model is validated with a 10-fold cross-validation method and the verification is done on unseen data which was not part of the validation set. We generated new instances of the system under test using the simulation-based approach architecture to run the DC e-motor model simulation to test the machine-learning diagnosis model.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI “Artificial Intelligence for Digitising Industry” project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union’s Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway. The

work was co-funded by grants of Ministry of Education, Youth and Sports of the Czech Republic, by the Austrian Federal Ministry of Transport, Innovation and Technology (BMVIT) under the program "ICT of the Future" between May 2019 and April 2022 (more information can be retrieved from https://iktderzukunft.at/en/). The work was also supported by the infrastructure of RICAIP that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 857306 and from Ministry of Education, Youth and Sports under OP RDE grant agreement No CZ.02.1.01/0.0/0.0/17_043/0010085.

References

[1] G. a. W. D. a. H. T. a. T. R. James, An Introduction to Statistical Learning: with Applications in R, Springer, 2013.

[2] Kaufmann, D., Nica, I., Wotawa, F.: Intelligent agents diagnostics - enhancing cyber-physical systems with self-diagnostic capabilities. Advanced Intelligent Systems, 2021. DOI https://doi.org/10.1002/aisy.202000218.

1.6

Real-Time Predictive Maintenance – Artificial Neural Network Based Diagnosis

Petr Blaha[1], Matus Kozovsky[1], Zdenek Havranek[1], Martin Dosedel[1], Franz Wotawa[2], David Kaufmann[2], Adil Amukhtar[2], Iulia Nica[2], Florian Klück[3] and Hermann Felbinger[3]

[1]Brno University of Technology CEITEC, Czech Republic
[2]Graz University of Technology, Austria
[3]AVL List GmbH, Austria

Abstract

In this article, we discuss the use of artificial neural networks for monitoring and diagnosis to be used in the context of real-time predictive maintenance. There are two use cases analysed here. As a first one, we discuss the motor model used for diagnosis in detail. In particular, we introduce a detailed acausal six-phase e-motor model to be used for different stator and inverter faults simulations. The inter-turn short circuit fault is targeted here. Simulation data and data measured on a real custom-made six-phase motor with the ability to emulate this fault are pre-processed based on the mathematical analysis of the fault. Such data are then used for modular neural network training. The trained modular neural network is optimized and deployed into the NVIDIA Jetson platform. The second ANN presented in this article is designed for bearing fault detection based on vibration measurements. The vibration data taken from publicly available datasets are transformed into suitable condition indicators which are analysed by the multilayer perceptron network running on a PC in MATLAB with the possibility to implement the resulting network into a small edge device. As such, two use cases are shown how artificial neural networks can be used on edge devices. Obtained results show that the approaches can be used in real setups.

Keywords: AI based diagnosis, acausal model, artificial neural network, computing-at-the-edge, modular neural network, multilayer perceptron, multiphase PMS motor, vibration diagnosis, inter-turn short circuit fault, reliability, validation.

1.6.1 Introduction and Background

This article focuses on the demonstration of Artificial Neural Network (ANN) based monitoring and diagnosis of e-motors and mechatronic systems implemented on the edge directly in embedded devices. It reveals a theoretical analysis of existing methods provided in the article "Foundations of Real Time Predictive Maintenance" and it can be viewed as its two use-cases.

1.6.1.1 AI-based Diagnosis of E-motors

There are many recent papers dealing with the AI-based diagnosis of e-motors [1], [2], [3], [4], [5] and others. The authors describe the design of the ANN and provide the success rate of the network evaluation, still they either do not deal with on the edge implementation or mention that the integration is in progress. This paper tries to reduce the complexity of the proposed networks by suitable data pre-processing to be able to classify the measured data on the edge platform represented with embedded AI hardware and tends to practical implementation and the operation in real-time. The integration of fault diagnosis and predictive maintenance algorithms as close as possible to the motor try to support this trend. This article demonstrates AI-based diagnosis and predictive maintenance for e-motor running on the edge. The diagnosis discovers the issues which are potentially dangerous for the operation if they are ignored. Diagnosis combined with the redundancy and integration of predictive maintenance tasks can substantially increase the reliability and the availability of the powertrain.

Various methods to detect faults and unexpected behaviour of cyber-physical systems were proposed [6]. These methods require a large amount of experimental data for the learning process, or well-known system behaviour described by the model. Modelling a healthy system is a relatively simple task, on the other hand, modelling the system under fault conditions can be challenging. For instance, commonly used causal modelling methods can be used to create a healthy motor model, however, modelling of fault behaviour of the electric motor using causal models is difficult [7]. For these reasons, an acausal modelling approach was selected since it brings many benefits [8].

This type of model can be created in MATLAB/Simulink using Simscape or other simulation methods and tools like Modelica.

The requirements on e-motor safety integrity levels are continuously increasing. It holds for the motor for fully or hybrid electric vehicles as well as for common industrial motors. For the e-motor, it is demanded by the braking capability of the e-motor which is good for the energy recuperation, and by the progression towards autonomous cars. In industrial applications, it is required due to a higher level of automation and precise production planning.

1.6.1.2 Artificial Intelligence in Vibration Diagnosis

Nowadays, the Artificial Intelligence (AI) approach to vibration diagnosis is growing significantly and machine learning as well as deep learning algorithms, including neural networks (NNs), are becoming a part of vibro-diagnosis [9]. Both approaches are used in practice – simple statistic-based machine learning algorithms as well as complicated NN structures. Examples of such methods can support vector machines, decision trees, Bayesian classifier, Mahalanobis-Taguchi system etc., as representants of the machine learning algorithms, and convolutional NN, recurrent NN, shallow dense NN, etc., as representants of the deep learning techniques. The functionality of the algorithms is mainly demonstrated on the publicly available datasets or on real captured data on minor occasions. Success rate of the classification is relatively high and reaches values over 98 %. Because of the lack of real data, even describing many failures of the concrete machine, transferred learning algorithms are in the scope of view of the scientific community in the last few years. This procedure allows the algorithm to be learned using one type of data captured on one machine, transfer the knowledge and classify the faults on the second machine without prior training using data of such machine.

1.6.2 Artificial Neural Network for e-Motor Diagnosis

This section provides the first use case of ANN for the inter-turn short circuit fault detection in a six-phase motor. It is composed of two subsections. The first one outlines acausal e-motor model, which is used to prepare training datasets with the fault, which are either not realisable on a real customised motor or prepared faster and complement datasets from the measurements on a real motor. The second one presents the steps from the selection of suitable condition indicators, through the data pre-processing, MNN design, training, validation, towards MNN deployment on NVIDIA Jetson Xavier platform.

1.6.2.1 Acausal e-Motor Model with Faults Injection Capability

This section outlines the development of an acausal e-motor model for the six-phase motor (connected as two three-phase sub-systems) which is capable to inject several typical Permanent Magnet Synchronous Motor (PMSM) stator faults. This model was parameterized for the correspondence with the real custom-made motor equipped with many windings taps enabling to emulate these faults. They both can serve as sources of datasets for the ANN training and validation which is capable to diagnose the inter-turn short circuit fault.

The Simscape allows building physical component models in Simulink in a fast and natural way. Components and physical connections are directly integrated within block diagrams and other modelling paradigms. Individual Simscape components interact with each other. Each Simscape block is represented by a set of equations that describe the physical behaviour of components. Equations are automatically processed during the model compilation process. The motor converts electrical energy into mechanical rotating energy. The mechanical rotating components as the moment of inertia or friction block can be used to create the simple model of a motor mechanical part. The motor connection to the complex mechanical model is also possible using Simscape.

The electrical part of the dual three-phase motor model can be described by equation (1.6.1).

$$\boldsymbol{u}_{abc_{12}} = \boldsymbol{R}_{abc_{12}}\boldsymbol{i}_{abc_{12}} + \frac{\mathrm{d}\boldsymbol{L}_{abc_{12}}\boldsymbol{i}_{abc_{12}}}{\mathrm{d}t} + \boldsymbol{e}_{abc_{12}} \tag{1.6.1}$$

Conversion of electrical energy into mechanical torque can be characterized using the equation (1.6.2).

$$T_e{=}Pp\left(\frac{1}{2}\boldsymbol{i}^{\mathbf{T}}_{abc_{12}}\frac{\mathrm{d}\boldsymbol{L}_{abc_{12}}}{\mathrm{d}\theta}\boldsymbol{i}_{abc_{12}}{+}\frac{\boldsymbol{i}^{\mathbf{T}}_{abc_{12}}\boldsymbol{e}_{abc_{12}}}{\omega_{\mathrm{e}}}\right) \tag{1.6.2}$$

The mentioned equation can be used to emulate healthy motor model behaviour. This model can be extended and the equation for some coils are split into the serial connection of two coils with mutual inductances. The serial connection of the coils has the same behaviour as the original one. The voltage potential of any place of the original coil can be subsequently used to simulate electrical fault. This approach is demonstrated in Figure 1.6.1.

The variable ***M*** represents mutual inductance between the coil ***L*** and other motor windings. ***R*** represents windings resistance. Variable ***e*** denotes the influence of back-EMF voltage in windings. Parameter σ represents a division

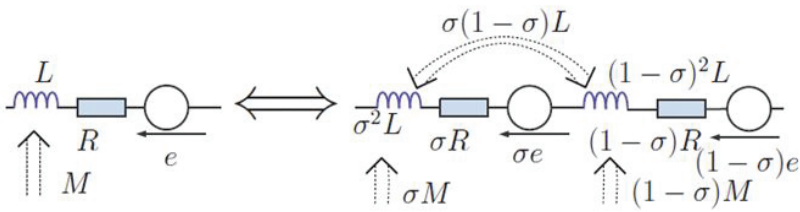


Figure 1.6.1 Winding equivalent for extended motor model.

ratio. The position of fault occurrence can be specified using this parameter. The coil splitting process is describable by equations (1.6.3).

$$u = Ri + \frac{\mathrm{d}Li}{\mathrm{d}t} + e \tag{1.6.3}$$

$$\begin{bmatrix} u_{1_1} \\ u_{1_2} \end{bmatrix} = \begin{bmatrix} R_{1_1} & 0 \\ 0 & R_{1_1} \end{bmatrix} \begin{bmatrix} i_{1_1} \\ i_{1_2} \end{bmatrix} + \frac{\mathrm{d}\begin{bmatrix} \sigma^2 L & \sigma(1-\sigma)L \\ \sigma(1-\sigma)L & (1-\sigma)^2 L \end{bmatrix} \begin{bmatrix} i_{1_1} \\ i_{1_2} \end{bmatrix}}{\mathrm{d}t} + \begin{bmatrix} e_{1_1} \\ e_{1_2} \end{bmatrix}$$

This description was used to create an acausal model of the dual three-phase machine able to emulate various internal motor faults. Internal short-circuits as well as disconnections in phases can be simply simulated. Figure 1.6.2 demonstrates various motor faults which can be simulated as well as emulated in the real motor. The model is used to generate important data sets for both healthy and faulty motors and for the transients from healthy to faulty states. These data sets can be used to train ANNs and for their validation.

1.6.2.2 Artificial Neural Network for Inter-turn Short Circuit Detection

This section shows the design of the DNN for inter-turn short circuit fault detection of PMSM. It starts with real experiments which were performed using the experimental motor with multiple windings taps which are capable to emulate this type of motor fault. The experiments helped with the selection of suitable condition indicator for fault detection. Further subsections describe data pre-processing, preparation of datasets and the process of training, validating and final deployment of DNN on embedded hardware.

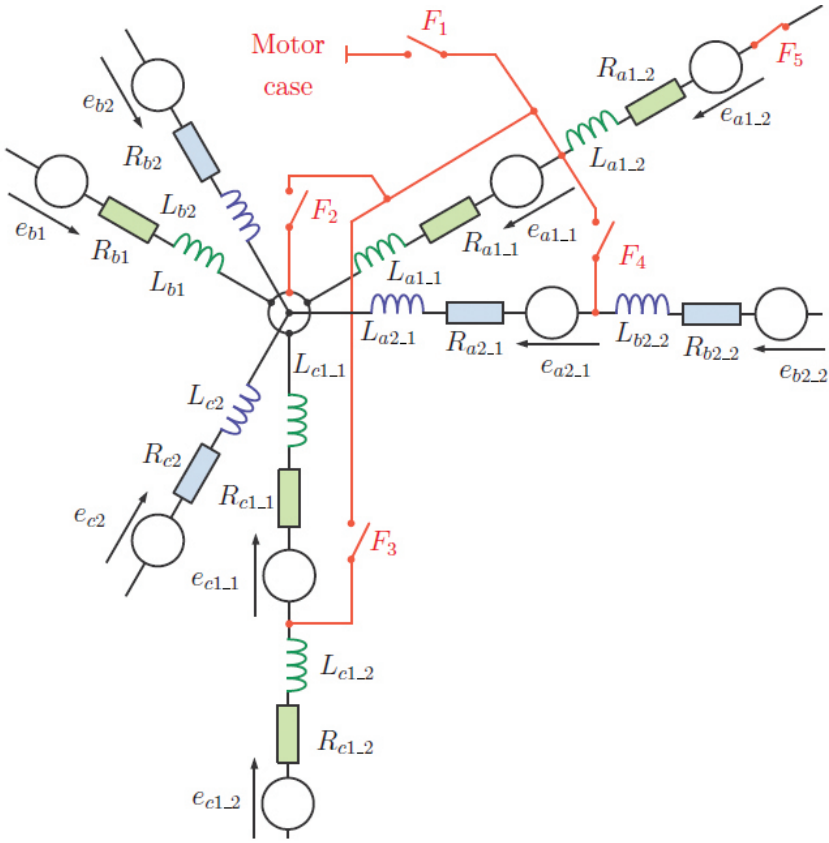


Figure 1.6.2 Simulated/Emulated faults in extended motor model / experimental motor.

1.6.2.2.1 Selection of suitable condition indicator for fault detection

Figure 1.6.3 demonstrates phase currents of both healthy and damaged motor sub-systems (only currents in the damaged sub-system are shown in this figure). Figure 1.6.4 shows phase currents transformed into *dq* coordinates. In this case, currents of both sub-systems are visible. As it can be observed, currents of damaged sub-system contain significant noise and distortion in a form of a significant second harmonic component. This is in accordance with the mathematical analysis of this fault as it is described e.g., in [8] and [11].

Phase currents or phase currents transformed into *dq* coordinates could be used as inputs to recurrent NN. This type of NN can filter the noise and consider not only the actual measurements but also previous ones.

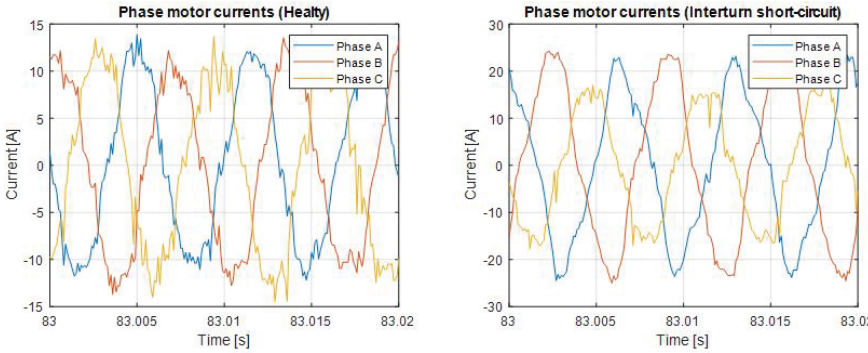


Figure 1.6.3 Phase motor currents (healthy/with fault).

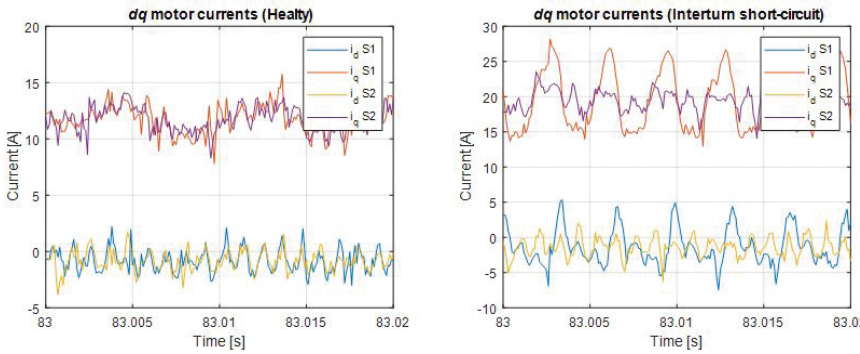


Figure 1.6.4 Motor currents in *dq* coordinates (healthy/with fault).

The computational complexity of such a NN would be high. On the other hand, linear NN would not be able to detect this fault properly using actual measurements as inputs due to high measurement noise. This problem can be overcome by suitable data pre-processing using the filtration method which is described later in this article.

Figure 1.6.5 shows low pass filtered motor current magnitudes in $\alpha\beta$ coordinates in both sub-systems. The magnitudes should be constant and independent of the motor electrical angle for the ideal motor and power inverter operating in steady-state; however, the sixth harmonic component is visible in both healthy and faulty waveforms. The sixth harmonics component is generated especially by the dead-time effect. Analysed inter-turn fault causes a significant increase of the second harmonic component which is nicely visible in $\alpha\beta$ current magnitude waveform.

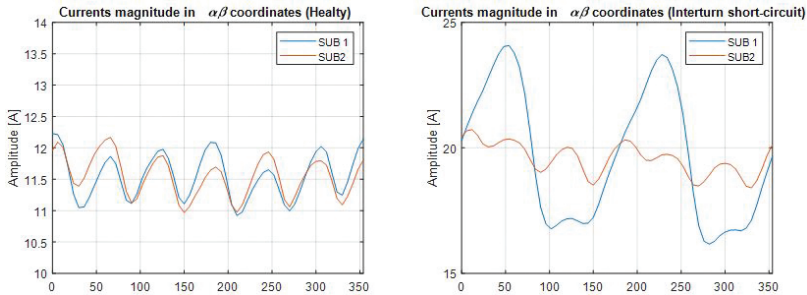


Figure 1.6.5 Filtered data in $\alpha\beta$ coordinates (healthy/with fault).

From the analysis above it is evident that the second harmonic component in filtered currents during one electrical period is a good condition indicator for the inter-turn short circuit fault.

The slight drawback is the fact that the number of measured data during one electrical period depends on motor speed. To suppress this drawback, the whole current waveform is converted to 60 data points per electrical period per sub-system. This fixes the length of the data buffer for its easier processing with ANN.

1.6.2.2.2 Network structure selection

The designed ANN is composed of several ANN modules, and as such, they form a Modular Neural Network (MNN). Filtered magnitudes of current waveforms in both sub-systems are used as inputs into the MNN. To increase fault classification precision, also filtered magnitudes of voltage waveforms are used as inputs. Currents represent the motor torque, while voltages carry the information about the rotational speed. Figure 1.6.6 shows the proposed structure of the MNN. The symmetry of the motor is reflected in the symmetry of data processing in MNN.

1.6.2.2.3 Data pre-processing

Inputs into the MNN consist of four buffers. Each buffer has 60 elements. The buffers are created from actual measured current/voltage magnitudes in both sub-systems.

Used filtration method is based on sixty IIR filters per MNN input. Only one filter with index i is active at a time depending on the actual motor position $\varphi_{e_{(k)}}$ in degrees.

$$i = \text{floor}(\varphi_{e_{(k)}}/60) \tag{1.6.4}$$

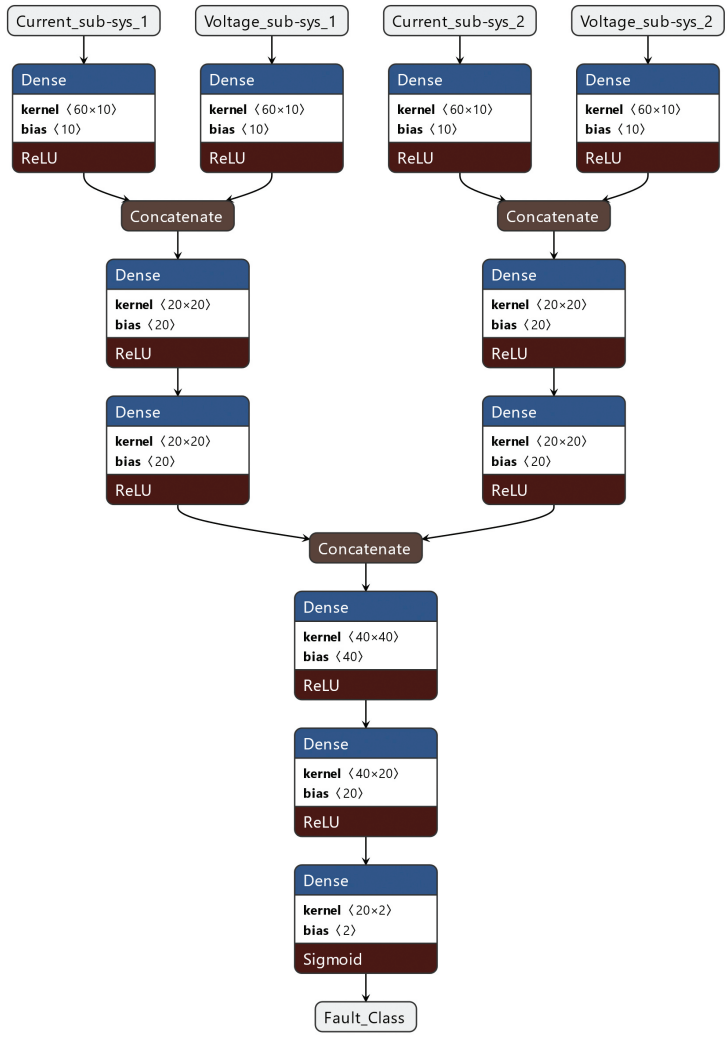


Figure 1.6.6 MNN structure used for inter-turn short circuit detection.

Input data updating occurs once per motor control period which is set to 100 micro us. Filters are described by the following formula:

$$y_{i_{(k)}} = K u_{i_{(k)}} + (1 - K)\, y_{i_{(k-1)}} \tag{1.6.5}$$

where the filtering constant K is set to 0.01, k is step related with the control period of the data generation and $u_{i_{(k)}}$ denotes one of the inputs.

Outputs of the filters are grouped into the buffer. This buffer is used as an input to MNN and it contains filtered signal along one electrical period.

1.6.2.2.4 Preparation of datasets

Fault symptoms that can appear in PMSMs depend not only on the emulated fault type but also on the motor operating point (motor speed, load torque, problematic stator phase). For this reason, a large amount of training data is required to cover all possible fault states in all operating conditions.

Datasets measured on the real motor were obtained under various motor speeds and torques. Randomly generated transients between randomly selected electrical speeds in the range from 200 to 3000 rad/s and with different torques in the range from 0 to 10 Nm (the breaking torques were not used for learning and not for validation) were used to generate training datasets. Simulated data using the motor model were used to generate complementary datasets under the fault condition because these experiments are time-consuming on a real motor. The fault current is high and causes fast local overheating of the motor and it is always necessary to let the motor cool down after such experiments. The experimental motor also does not enable to make the short-circuiting between an arbitrary couple of turns of the coil. And this is the second reason why simulated data are used to prepare missing datasets advantageously.

The motor symmetry was employed to extend datasets with the faults in different motor phases. Phase currents and voltages were re-grouped in different orders to prepare the training and validation data for the six-phase (two times three-phase) motor. This solution helped to prepare additional datasets for learning/testing without the necessity to simulate/experiment with each phase and each sub-system separately. This approach significantly reduced the time needed for the dataset preparation.

1.6.2.2.5 MNN training

The network from Figure 1.6.6 was trained from the mixture of real measurements and the data coming from the simulations using the acausal motor model on a workstation PC in the environment of MATLAB using pre-processed datasets as described in the previous two sections.

1.6.2.2.6 MNN validation

The capability of MNN to diagnose the inter-turn short circuit fault was validated using data from the real motor only. Three turns of the stator winding coil were short-circuited. It represents 3/7 of the stator coil in one

slot. Validation datasets were measured on the real motor in a similar way as the ones for training. Data used for training were not used for the validation at the same time. The fault was successfully classified with the probability of 99.92 %. When the fault depth was lower, the fault detectability was slightly reduced.

The fault detection below 200 rad/s is significantly less precise, but the severity of the fault is also lower, and it is usually not harmful for the motor.

1.6.2.2.7 MNN deployment

The designed and trained network was implemented in NVIDIA Jetson Xavier platform using GPU coder in MATLAB. This NVIDIA platform was connected with the inverter controller using Ethernet. The controller sends required voltages and currents. The data pre-processing can run in both, in the controller or in the NVIDIA platform.

After the fault injection into the model simulation, it requires only 1.1 ms for classification with a success ratio of 99.92%. The latency of the Linux running on the NVIDIA platform spans up to 100 μs with the provided JetPack software. Other operating systems designed for hard real-time like RedHawk Linux exist and significantly improve the latency issue.

1.6.3 Artificial Neural Network based Vibration Diagnosis

This section is devoted to the design of the second use case, which is ANN for vibration diagnosis for the bearing state of health monitoring. The first subsection deals in general with the vibration diagnosis of rotating machines. The second subsection analysis AI approaches in vibration diagnosis. The third section presents the developed MLP network.

1.6.3.1 Vibration Diagnosis of Rotating Machines

Vibration diagnosis of rotating machines is a commonly used technique in technical diagnosis and faults identification. Not only typical mechanical failures, such as unbalance, misalignment, gears, and bearings problems can be advantageously diagnosed, but also electrically caused failures may be simply found. Nowadays, electrical faults are diagnosed mainly using electrical quantities measurement, however, mechanical vibrations measurement can be very helpful in the detection of electrically hardly detected faults. On the other hand, vibration-based diagnosis is capable to

reveal the faults undetectable by measurement of only electrical quantities. There can be two reasons for this fact:

- Manifestation of a fault in the electrical signal domain is quite weak (e.g., in the initial phase of the fault) and cannot be correctly measured due to small signal to noise ratio, while the vibration signal provides successful information for sufficient detection of the fault.
- Given fault does not have an image in the electrical domain, thus the measurement of mechanically generated signals is helpful for successful fault diagnosis.

Thanks to the aforementioned aspects, vibration diagnosis is a widely used part of the diagnosis and predictive maintenance of rotating machines, including e-machines.

1.6.3.2 AI Approaches in Vibration Diagnosis

The algorithms, statistical procedures, and NNs are usually created, learned, and finally inferred on computers, both standard personal computers, and advanced powerful multicore computers with the support of dedicated graphic cards with multicore graphic processors. Also, specialized dedicated hardware such as the NVIDIA Jetson platform is commonly used thanks to relatively small dimensions and high computational performance compared to standard computers. A typical application area for this kind of hardware is Computing-at-the-Edge (CatE) nodes. Insignificant limitations in available memory and performance, rather typical for small size CatE sensors, shall also be taken into consideration. Finally, implementation of the NNs into small sensors or CatE nodes is a relatively challenging process because of limited resources, mainly available memory, computational performance, power consumption as well as the speed of inference of the algorithm. It is very common, that NN creation, learning, and validation process is done using a powerful computer, NN structure and parameters are exported from the IDE and imported into this small performance device as a functional and successfully learned algorithm. A NN is then executed on the target hardware with no need to learn the overall structure of the network. It is good to mention, that by the small performance device is understood a simple microprocessor with several kilobytes of read/write memory, max. a megabyte of program memory, core frequency of about several hundreds of MHz and typical performance of around 100 DMIPS (Dhrystone Million Instructions Per Second). For comparison, typical Jetson NANO hardware

has 128 core GPU, 4GB of internal RAM, and is capable of processing around 1600 FLOPS.

Because the performance of the system, as previously mentioned, can be somehow limited, it is good to reduce the amount of input data by pre-processing procedures. Not only the NN algorithm itself, but also other necessary code needs to be executed inside the processor to ensure the basic functionality of the system (e.g., communication with sensing elements, drawing graphics on display to communicate with the user, peripheral service routine, etc.). Signal pre-processing leads to reduction of the input data and in fact to the reduction of the size and execution time of the AI algorithm. In the vibration diagnosis, two types of extracted features are usually used:

- Time domain features – features calculated from the time signal, mainly statistic parameters like RMS value, standard deviation, kurtosis etc.
- Translated domain features – features calculated from translated domain. Frequency transform, Hilbert transform, Gabor transform, Z-transform, etc., are the most used transforms in the vibration diagnosis. It is good to mention, that not the whole e.g. frequency spectrum is used as an input for the algorithm, but only some particular frequency lines representing possible faults are led to the input of the NN. This brings a significant reduction of the input data and computational complexity of pre-processing algorithms.

1.6.3.3 MLP implementable in device at the edge

As an example of a simple and powerful NN algorithm for bearing faults classification, Multilayer Perceptron (MLP) can be considered. Simple shallow dense NN of a MLP type can be seen in Figure 1.6.7.

The network has one hidden layer and three layers in total (including input and output layer). The number of input neurons is equal to eight, representing eight input time-domain features. The number of output neurons is equal to five, representing five output classes. Therefore, the network is trained to distinguish between five faulty states of the input signal. Training dataset used for this network is represented by publicly available CWRU bearing data centre data. As this dataset is used by many scientists for evaluation of their bearing faults detection algorithms capabilities, accuracy of different neural networks can be found in the literature, e.g. [10], where maximal accuracy of 99,92 % can be found. Dataset, containing data of healthy bearing state and four degrees of bearing outer ring faults, was pre-processed and eight

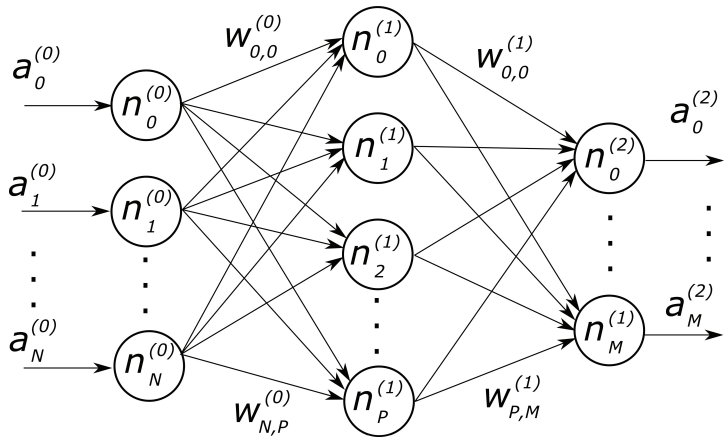


Figure 1.6.7 Shallow dense NN.

signal features have been extracted, namely RMS value, kurtosis, skewness, variance, standard deviation, mean value and min and max value.

Inference algorithm of the network, as well as weights modification procedure using back propagation method, have been implemented in MATLAB environment. The output value of each neuron can be calculated using equation (1.6.6).

$$y = f\left(\sum_{i=1}^{N} w_i x_i\right) \tag{1.6.6}$$

Where $\boldsymbol{w}_i$ is the vector of the individual weights $w_1, w_2, ..., w_N$, $\boldsymbol{x}_i$ is the vector of individual inputs $x_1, x_2, ..., x_N$ of the perceptron, and $\mathrm{f}(\cdot)$ is an activation function. In this case, sigmoid activation has been used.

It is necessary to adjust the initial weights values of the NN during the learning procedure. The commonly used approach is based on back propagation algorithm (gradient descend method). The goal is to adjust the weights according to equation (1.6.7)

$$w_j^0(t+1) = w_j^0(t) + \Delta w_j^0 \tag{1.6.7}$$

with the effort to minimize the output error, defined by subtraction between desired (d_j) and real ($a_j{}^{(2)}$) output values of the network (1.6.8).

$$E = \frac{1}{2}\sum_j \left(a_j^{(2)} - d_j\right)^2 \tag{1.6.8}$$

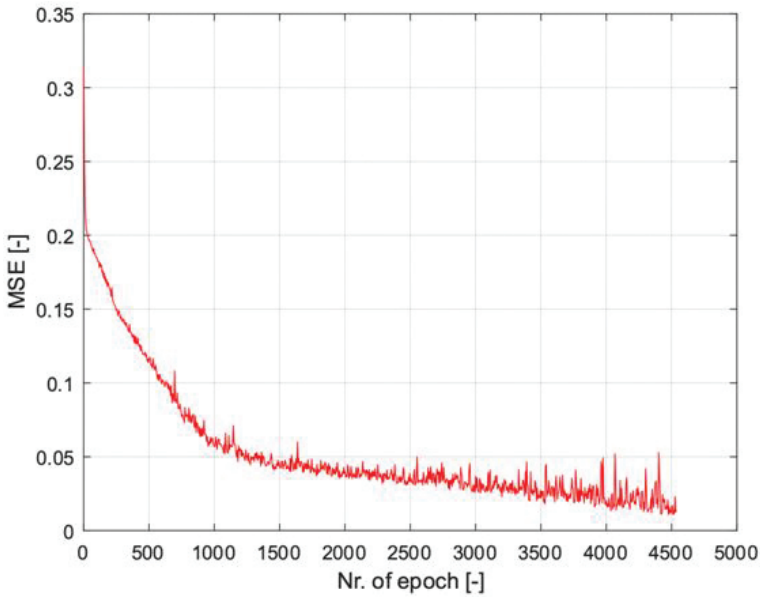


Figure 1.6.8 Mean square error of MLP during training phase.

The equation for final weights modification after partial derivatives of the aforementioned equations and using mathematical operations can be written:

$$\begin{aligned}\Delta w_j^0 = \frac{\partial E}{\partial w_j^0} =& \frac{\partial E_C}{\partial a_j^{(2)}} \cdot \frac{\partial a_j^{(2)}}{\partial z_k} \cdot \frac{\partial z_k}{\partial w_j^0} = \cdots \\ &= \left(a_j^{(2)} - d_j\right) \cdot a_j^{(2)} \left(1 - a_j^{(2)}\right) \cdot a_j^{(0)} \cdot \alpha\end{aligned} \tag{1.6.9}$$

where $\boldsymbol{a}_j^{(2)}$ is the output of the network, $\boldsymbol{a}_j^{(0)}$ is the input of the network, d_j is the desired output and α is the learning rate of the back propagation algorithm.

The final MLP ANN has been implemented in MATLAB and its classification accuracy has been evaluated using Confusion Matrices (CM). Mean square error (MSE) calculated according to Equation (1.6.8) during the learning phase can be observed in Figure 1.6.8.

As it can be seen, MSE reaches very low values (below 2 %) at the end of the learning phase, which has been confirmed by the CM obtained from the output acquired during the testing process. Mentioned CM can be seen in Figure 1.6.9.

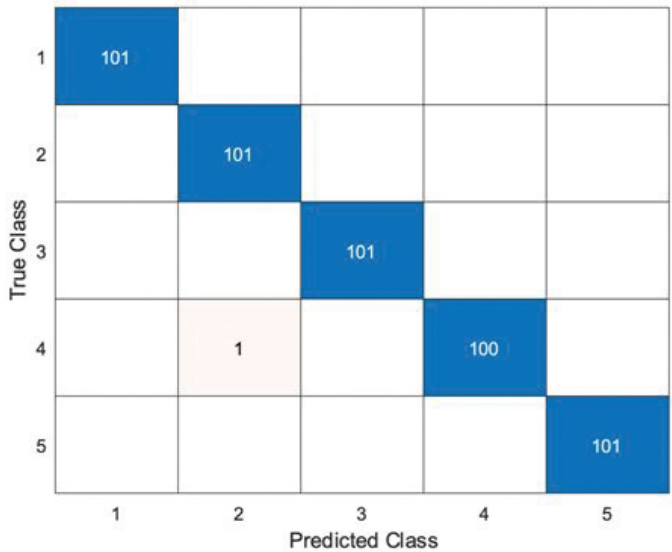


Figure 1.6.9 CM of the testing process of MLP.

Since the MLP was successfully implemented in MATLAB, there is a strong intention to import the network in the low-performance CatE device. Such a device can be represented by a small electronic sensor including a sensing element and microcontroller suitable for MLP inference (e.g. ARM based STM32 microcontroller). Once the structure of the network is created and the weights of the network are established by the training process, a file describing the structure of the network and weights values can be exported from MATLAB and imported by STM Cube. AI application directly into a microcontroller. Validation of the network is done on a PC within testing phase, while validation of the resulting network implementable into STM device is done within Cube.AI software. Afterwards, MLP will fully run inside the target STM device.

To fulfil the requirements of the limited resources of the microcontroller, a simple evaluation of the occupied memory has been done and it is listed in Table 1.6.1 (considering implementation of *float* data type using four bytes).

This amount of total occupied memory of ca. 1.2 kB can be smoothly implemented into small size memory of a microcontroller. Despite the small size of the MLP network, the classification accuracy of the network is satisfactory, as it can be seen in Figure 1.6.9 and the overall algorithm is very well suited for this simple case of bearing faults classification. It is good

Table 1.6.1 CatE device evaluation of occupied memory.

Layer	Memory
Input features	8 x 4 bytes
weights vector (between input and hidden layer)	162 x 4 bytes
weights vector (between hidden and output layer)	90 x 4 bytes
output layer	5 x 4 bytes
intermediate temporary variables	30 x 4 bytes
TOTAL	∼1.200 bytes

to mention, that the accuracy of the classification strongly depends on the learning phase given by the quality and size of the input training dataset.

1.6.4 Conclusion

Two ANNs were designed to detect unexpected behaviour of the e-motor and the bearing on the edge device to operate in real-time. For the inter-turn short circuit detection in the PMSM, MNN was utilized because of the motor symmetry. The highly detailed acausal e-motor model was used to substitute measurement in the operating points which were unreachable on a customized real motor and to reduce the number of required experiments on a real motor. A significant factor in the diagnosis of an inter-turn short circuit fault is the processing time. It was reduced with the used computational hardware to 1.1 ms which is promising and should be sufficient for real-time diagnostic of common e-motors. The second ANN prepared for abnormal vibrations analysis due to bearing faults is MLP designed in a way that the computation is prepared to be deployed directly on the vibration sensor's microcontroller. This is possible since a low-sized and efficient MLP network is applied, which delivers good results in the classification of bearing faults.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway. The work was also supported by the infrastructure of RICAIP that has received funding from the European Union's Horizon 2020 research

and innovation programme under grant agreement No 857306 and from Ministry of Education, Youth and Sports under OP RDE grant agreement No CZ.02.1.01/0.0/0.0/17_043/0010085.

References

[1] J. Han, D. Choi, S. Hong and H. Kim: Motor Fault Diagnosis Using CNN Based Deep Learning Algorithm Considering Motor Rotating Speed, 2019 IEEE 6th International Conference on Industrial Engineering and Applications (ICIEA), 2019, pp. 440-445, doi: 10.1109/IEA.2019.8714900.

[2] Y. Luo, J. Qiu and C. Shi: Fault Detection of Permanent Magnet Synchronous Motor Based on Deep Learning Method, 2018 21st International Conference on Electrical Machines and Systems (ICEMS), 2018, pp. 699-703, doi: 10.23919/ICEMS.2018.8549129.

[3] S. Wang, J. Bao, S. Li, H. Yan, T. Tang and D. Tang: Research on Interturn Short Circuit Fault Identification Method of PMSM based on Deep Learning, 2019 22nd International Conference on Electrical Machines and Systems (ICEMS), 2019, pp. 1-4, doi: 10.1109/ICEMS.2019.8921744.

[4] J. Qi and H. Wan: A Detection Method of Phase Failure of PMSM Based on Deep Learning, 2020 Chinese Automation Congress (CAC), 2020, pp. 5591-5595, doi: 10.1109/CAC51589.2020.9326708.

[5] F. Husari and J. Seshadrinath: Sensitive Inter-Tum Fault Identifcation in Induction Motors Using Deep Learning Based Methods, 2020 IEEE International Conference on Power Electronics, Smart Grid and Renewable Energy (PESGRE2020), 2020, pp. 1-6, doi: 10.1109/PESGRE45664.2020.9070334.

[6] Y. Chen, S. Liang, W. Li, H. Liang and C. Wang: Faults and Diagnosis Methods of Permanent Magnet Synchronous Motors: A Review. *Appl. Sci.* 2019, *9*, 2116. https://doi.org/10.3390/app9102116

[7] S. Foitzik and M. Doppelbauer: Simulation of Stator Winding Faults with an Analytical Model of a PMSM, 2018 IEEE International Conference on Power Electronics, Drives and Energy Systems (PEDES), 2018, pp. 1-6, doi: 10.1109/PEDES.2018.8707719.

[8] L. Otava, M. Graf, and L. Buchta: Interior Permanent Magnet Synchronous Motor Stator Winding Fault Modelling, IFAC-PapersOnLine, vol. 48, no. 4, pp. 324–329, 2015, doi: 10.1016/j.ifacol.2015.07.055.

[9] G. Zurita, V. Sanchez and D. Cabrera: A Review Of Vibration Machine Diagnostics By Using Artificial Intelligence Method, UPB -INVESTIGACIÓN & DESARROLLO, No. 16, Vol. 1: 102 – 114 (2016).

[10] S. Zhang, S. Zhang, B. Wang, and T. G. Habetler, "Machine Learning and Deep Learning Algorithms for Bearing Fault Diagnostics – A Comprehensive Review," Jan. 2019, doi: 10.1109/ACCESS. 2020.2972859.

[11] B. Mansouri, H.J. Idrissi and A. Venon: Inter-Turn Short-Circuit Failure of PMSM Indicator based on Kalman Filtering in Operational Behavior. Annual conference of the prognostics and health management society, pp. 1-7, 1 2019, doi: https://doi.org/10.36001/phmconf.2019.v11i1.831

Section 2

AI Semiconductor

2.0

AI in Semiconductor Industry

Cristina De Luca[1], Bernhard Lippmann[1], Wolfgang Schober[2], Saad Al-Baddai[2], Georg Pelz[1], Andreja Rojko[3], Frédéric Pétrot[4], Marcello Coppola[5] and Reiner John[6]

[1]Infineon Technologies AG, Munich, Germany
[2]Infineon Technologies AG, Regensburg, Germany
[3]Infineon Technologies Austria AG, Austria
[4]Univ. Grenoble Alpes, CNRS, Grenoble INP, TIMA, France
[5]STMicroelectronics, France
[6]AVL List. Austria

Abstract

This introductory article opens the "Applications of AI in the Semiconductor Industry" section by giving a holistic overview of the development of artificial intelligence (AI) technologies applied to the industry. Historically, the semiconductor industry has utilised complex automation for many tasks and areas, especially in repetitive work and uniform processes. The high need for flexibility in manufacturing, increased diversification of products, complexity, and demand for more autonomous operations, including human-machine interaction, have led to a strong push towards using AI technologies in semiconductor manufacturing. AI technologies are applied in semiconductor product development, digitised product definition (DPD), knowledge management system for risk assessment and root cause analysis, image recognition for inspection and defect classification in front end (FE) and back end (BE) applications for anomaly detection in process chains. Deep learning (DL) and Machine Learning (ML) techniques have given a new stimulus to semiconductor industry research to address the unique challenges for semiconductor manufacturing as the technologies nods are evolving and the number of process parameters to be controlled is increasing. In the end,

the article introduces the four contributions to this section, highlighting the use of AI, computer vision, neural networks (NNs) in various use cases in semiconductor manufacturing processes.

Keywords: artificial intelligence (AI), industrial artificial intelligence, semiconductor industry, manufacturing, image processing, computer vision, neural networks, pattern recognition, natural language processing.

2.0.1 Introduction and Background

Industrial AI integrates domain-specific know-how with the AI-based functions and capabilities into various AI-enabled applications in industrial sectors. AI technologies applied in the industry enables and accelerates the autonomous and semi-autonomous processes that run those operations, realising the vision of the self-optimising manufacturing facilities. AI plays a double role in the semiconductor industry: it acts as a key leverage element for digitising the manufacturing processes and provides the technology for semiconductor manufacturing to optimise the operations and control the process parameters as the technologies advance toward nanometre-scale semiconductor nodes. The primary goals of using AI technologies are to reduce costs, save time, improve quality, and increase the robustness of industrial processes. AI technologies are applied to increase the efficiency and effectiveness of industrial processes by mastering complex situations within the limitations of specified systems. The use of AI in industrial sectors represents a new opportunity for industrial stakeholders to optimise resources and increase profitability with a high economic impact.

Semiconductor companies are integrating AI, ML, expert systems, and other technologies to develop intelligent manufacturing environments to transform scheduling, dispatching, equipment productivity, process and equipment control, and robotic management. These technologies optimise quality, productivity, efficiency, and flexibility while maximising cost-effectiveness and accelerating overall innovation.

2.0.2 AI Developments in Semiconductor Industry

Today, the semiconductor manufacturing processes are based on the use of Advanced Process Control (APC) techniques. The availability and use of custom, off-the-shelf APC facilities in FAB are part of the production requirements. SEMI consortium [10] has issued the “The Process Control

System Standards" (SEMI E133) that defines communication between components to enable run-to-run (R2R) control, fault detection (FD), fault classification (FC), fault prediction (FP) and statistical process control (SPC). It is supported by SEMI specifications E125 and E134 on EDA (Equipment Data Acquisition).

The APC remains a fundamental pillar in semiconductor manufacturing supported increasingly by AI and Industrial Internet of Things (IIoT) technologies.

Today, the semiconductor manufacturing facilities experience more challenges due to high-mix/low-volume loads that result in shorter production cycles and frequent product mix changes, with increasing pressure on costs and quality.

In addition, the effect of Moore's law is expected to approach the limit of possible performances. Moore's law has been seen as the fundamental driver for innovation in the integrated circuit (IC) industry. The doubling of IC performance started to slow down due to the physical limitations of transistor shrinkage and quantum mechanical effects such as "quantum tunnelling" [1][2], which posed many challenges due to excess heat generation and power consumption. The phenomenon of "dark silicon" has posed other problems concerning the performance-cost perspective.

The development of new semiconductor technologies requires complex manufacturing facilities with advanced metrology systems. Each aspect of semiconductor processing, from lithographic design rule specifications to continuous yield analysis, essentially depends on accurate and reliable data for critical dimension (CD) lithographic patterning and material composition. The status of semiconductor metrology techniques and the opportunities for AI methods to provide the necessary breakthroughs to support future process node development is presented in [8][9].

The competitive pressures on semiconductor manufacturers are increased to reduce production time and costs, improve quality, shorten innovation cycles, and accelerate new technologies' ramp-up [3].

A list of few significant advances made by AI and IIoT in the semiconductor industry is presented below:

Analytics and optimisation used to eliminate repetitive processes and searches in content management for root cause analysis. Expert systems for root cause failure analysis and risk assessment in semiconductor production help access knowledge across all related content and support transferring domain knowledge from engineers' expertise into algorithms. Fast and reliable decisions are made using documents' data sources (e.g., failure

mode effect analysis, etc.). Performance improvement is made possible by performing multidimensional correlations analysis using highly nonlinear data through machine learning and deep learning techniques and discovering correlations where human experts need time and are prone to errors.

Sensing used in automated quality assurance by integrating AI-based capture systems such as image recognition to support the visual inspection and classification of defects at both the front-end (wafer fabrication) and back-end (assembly and test) manufacturing processes. Manual and other conventional quality inspections are unreliable, expensive, have a low detection rate, and are challenging to scale. The use of AI technologies increases the reliability and efficiency of these processes.

Packaging optimisation used to improve the assembly and packaging processes in the industry by applying AI solutions consisting of a combination of anomaly and deviation detection to increase reproducibility.

Digitalising product definition integrating AI technologies used to optimise the relationship between requirements and constraints. The complexity of the requirements stack requires optimisation techniques that AI provides. Significative development is expected by using AI integrated into manufacturing facility infrastructure to support the transition from document-based requirements to machine-readable formats.

2.0.3 Future Trends for AI Technologies and Applications in Semiconductor Industry

The global semiconductor market is projected to grow from $452.25 billion in 2021 to $803.15 billion in 2028 at a CAGR of 8.6% during 2021-2028 [4]. Globally, the long-term market trend for electronic components is expected to exceed US $1,000 billion by 2030. It is estimated that the research and development costs of developing circuits from a 65 nm node to a state-of-the-art 5 nm node have increased from $28 million to $540 million, and fab build costs for the same nodes have increased from $400 million to $5.4 billion [5]. By implementing AI and ML alone, the industry can gain $35-40 billion annually. Over a more extended timeframe of 3 to 4 years, it could double to almost 20% of the industry's current revenue [5].

AI is transforming the industrial semiconductor industry, moving from an "application-centric world" to a "data-centric world", where almost all data will be generated and consumed by machines. The industry's growth

is no longer limited by the ability of humans to create or consume data. New computing approaches emerge from processing the massive amounts of available data, and AI-based hardware and software are required to enhance productivity. Training AI computing becomes incredibly energy-intensive, so the industry must drive performance-per-watt improvements [6].

The AI technologies can be used to adjust tool parameters to achieve greater accuracy by deploying real-time tool-sensor data, metrology readings, and tool-sensor readings from earlier process steps, enabling ML algorithms to capture nonlinear relationships between process time and outcomes (e.g., etch depth). The data aggregated could include electric currents in the etching process, light intensity in lithography, and temperatures in baking. Optimal process times based on AI models can be provided for individual wafer or per-batch that decrease the processing time, improve yield, or both, thus reducing the cost and increasing throughput.

Computer vision and AI algorithms show their capabilities in the visual inspection of wafers to ensure quality by detecting defects in the front-end and back-end production process using cameras, microscopes, or scanning-electron microscopes. Optical inspection in the semiconductor manufacturing process for analysis and verification represents an area with considerable potential for AI research and can significantly improve the equipment's expected performance. In addition, combining different physical and electrical characterisation and measurements techniques with data mining and AI can provide better yield curves. AI-based wafer-inspection systems using DL and computer vision are trained/learned to automatically detect and classify defects on wafers with better accuracy than human operators. The use of dedicated hardware-based on graphics- and processing tensor-processing units and on-premises edge computing enables computer-vision algorithms to train and deploy in real-time in a scalable manner.

AI-based analytics can support the *automated yield learning in integrated circuit design and optimise* the iterations based on feedback from manufacturing. Deploying ML-based algorithms to identify patterns in component failures, predict likely failures in new techniques, and propose optimal layouts to improve yield and increase the design's efficiency.

2.0.4 AI-Based Applications

AI4DI partners [7] are developing AI and IIoT technologies with applications in different areas of the semiconductors sector. The articles included in this section cover four demonstrators and actionable insights into how AI and

IIoT are used in semiconductors applications, presenting challenges and technological advancements to accelerate the digitising process across the industry.

The article *"AI-Based Knowledge Management System for Risk Assessment and Root Cause Analysis in Semiconductor Industry"* proposes a new expert system concept for root cause failure analysis and risk assessment in the semiconductor industry. The knowledge representation of the expert system's main component is based on knowledge graphs created with knowledge extracted from various data sources and post-processed for better consistency. Queries to the expert system will provide known real-time risks of the production flow in semiconductor manufacturing. The paper concludes that integrating fast-developing natural language processing technologies and AI/ML methods seems the most promising way to digitalise FMEA documents and create this expert system that can support FMEA experts at their more complex tasks. Research conducted in AI4DI is also working toward accommodating industrial environment specifics to facilitate the integration of the FMEA tool in the real environment of industrial semiconductor manufacturing.

The article *"Efficient Deep Learning Approach for Fault Detection in the Semiconductor Industry"* investigates the use of high quantized artificial neural networks to be implemented on small industry-grade microcontrollers enhanced with hardware accelerators. The system proposes an automatic visual inspection and classification of defects in both the front- and back-end manufacturing processes in the semiconductor industry to increase yield and reduce costs. This is a considerable improvement of the current inspection performed by humans, primarily because of the high throughput in the production lines. Preliminary experiments indicate that when appropriately trained, quantized artificial neural networks can reach high accuracy, and their implementation using the interconnection of two hardware parts can be resource-efficient. It remains to be seen for the following steps to be applied on a larger scale.

The article *"Towards Fully Automated Verification of Semiconductor Technologies"* proposes an extension of the existing workflow with an automated device cross-section analysis to increase trust in semiconductor devices and their originality (i.e., combat rogues). Central to this approach is the confluence of knowledge from human domain experts and AI/ML experts input to automated image interpretation. The goal is to extract technological attributes and verify them against original design and specifications. By

applying state-of-the-art AI, the results are comparable to those of an operator's manual effort.

The article *"Automated Anomaly Detection through Assembly and Packaging Process"* highlights the importance of continuous optimising, using, and adjusting the assembly process in the semiconductor industry to achieve competitive advantages, mainly as its reproducibility depends on various distributed parameters. This demands the high accuracy of employed automatic inspection tools for visual defect detection. An AI solution consisting of a combination of anomaly detection (unsupervised learning) and supervised learning for detecting deviations is proposed, satisfying the demand, and required features. Two anomaly detection examples have been considered, and the results showed potential to be good alternatives to classical approaches.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] "Quantum Computers Explained - Limits of Human Technology". youtube.com. Kurzgesagt. 8 December 2017.
[2] "Quantum Effects at 7/5nm and Beyond". Semiconductor Engineering.
[3] Kern, E.-M. (2016). "Verteilte Produktentwicklung." In: Lindemann, U. (ed.) Handbuch Produktentwick - entwicklung, pp. 455-481. Hanser, München.
[4] Fortune Business Insights (2021). "The global semiconductor market is projected to grow from $452.25 billion in 2021 to $803.15 billion in 2028 at a CAGR of 8.6% in forecast period, 2021-2028". Available online at: https://www.fortunebusinessinsights.com/semiconductor-market-102365
[5] Göke, S., Staight, K., and Vrijen, R. (2021). "Scaling AI in the sector that enables it: Lessons for semiconductor-device makers." Available

online at: https://www.mckinsey.com/industries/semiconductors/our-insights/scaling-ai-in-the-sector-that-enables-it-lessons-for-semiconductor-device-makers

[6] Achutharaman, R. (2021). "Trends Accelerating the Semiconductor Industry in 2021 and Beyond". Available online at: https://blog.appliedmaterials.com/trends-accelerating-semiconductor-industry-2021

[7] AI4DI project. https://ai4di.eu/

[8] Dillinger T. (2020). "A Compelling Application for AI in Semiconductor Manufacturing." Available online at: https://semiwiki.com/semiconductor-manufacturers/287593-a-compelling-application-for-ai-in-semiconductor-manufacturing/

[9] Yi-hung Lin, Y-H. (2020). "Metrology with Angstrom Accuracy Required by Logic IC Manufacturing - Challenges From R&D to High Volume Manufacturing and Solutions in the AI Era", VLSI 2020 Symposium, Workshop WS2.3.

[10] SEMI. Available online at: https://www.semi.org/en/node/97711

2.1

AI-Based Knowledge Management System for Risk Assessment and Root Cause Analysis in Semiconductor Industry

Houssam Razouk[1,4], Roman Kern[3,4], Martin Mischitz[1], Josef Moser[1], Mirhad Memic[1], Lan Liu[3], Christian Burmer[2] and Anna Safont-Andreu[1]

[1]Infineon Technologies Austria AG, Austria
[2]Infineon Technologies AG, Germany
[3]Know-Center GmbH, Austria
[4]Graz University of Technology, Austria

Abstract

Due to the increasing technical complexity of products and market pressure, the demands in the semiconductor industry are rising with respect to quality, performance, and time to market. Root cause analysis and risk assessment are crucial elements for success in fulfilling these demands. As a result, there is an ever-growing number of technical documents, which potentially contain valuable information serving as a base to inform development and production. Experts need to cope with this large number of technical documents, for example, to generate new hypotheses to identify possible root causes of deviations or potential risks in the ramp-up and production phase of new products. Unfortunately, most of the technical documents are unstructured, making processing them even more tedious. New advances in computer science, specifically artificial intelligence (AI), open the door for a higher degree of automation of knowledge management tools to support experts. Knowledge bases such as knowledge graphs allow for representing complex information but need to be created for each domain. Novel state-of-the-art graph embedding algorithms showed promising results

in complementing knowledge bases with new relations. Complementary to knowledge base completion, language models trained on large textual corpora have demonstrated their ability to capture complex semantics. This paper proposes a new expert system concept for failure root cause analysis and risk assessment in the semiconductor industry, which leverages the advanced graph embeddings in combination with language models. The main challenges in this setting are the type of relations of interest, which are causal, and the language being used, which is highly domain-specific. Thus, we devised AI for consistency improvement of the data, predicting new links, and information extraction from unstructured data. The information extraction is conducted by levaraging domain specific ontologies and by focusing on presence of causal language.

Keywords: expert system, root cause analysis and risk assessment, knowledge representation, semiconductor industry, natural language processing, information extraction, knowledge graph, convolutional neural network, recurrent neural network, machine learning, link prediction, text classification, consistency improvement.

2.1.1 Introduction and Background

In the last decades, more than ever, high-tech microelectronic-based products consolidate as part of everyday life. Thus, expectations concerning functionality, reliability, and competitive prices are growing. As a response, more functions are integrated, facilitating products' performance continuous growth. Consequently, the technological complexity of microelectronic components and the amount of data are constantly increasing due to Industry 4.0 applications in the production facilities. Moreover, the fierce, competitive market situation for the industrial semiconductor companies, which are the leading supplier of such high-tech products, is inevitably increasing the time to market and price pressure. Therefore, knowledge and experience are necessary to enable innovation, stable production, and cope with market dynamics.

In the semiconductor manufacturing industry, knowledge and experience refer to domain-specific know-how in chip design, operation and control of highly sophisticated infrastructure, metrology, quality assurance, verification, and validation. An effective and efficient knowledge management system that, on-demand, applies and rolls out existing know-how and allows rapid learning from the failures has to be in place.

The necessary knowledge is usually accumulated over long periods and reflects in the practical experience of the human domain experts. It is common to document human domain experts' knowledge in the written form using the domain-specific language. One of the main challenges is how to make this knowledge continuously more accessible to all potential users, respectively, i.e., the engineering teams working in semiconductor manufacturing.

To guarantee the high quality of the products in the semiconductor industry, human domain experts thoroughly investigate deviations in the manufacturing process or in the products' characteristics. Mainly, two standard processes triggered after deviation identification to answer causal questions: (i) risk assessment: what will be the effect of the observed deviation? (ii) root cause analysis: what is the cause of the observed deviation? Therefore, it is highly intriguing to identify causal relations along the whole production process.

In the semiconductor manufacturing with many hundred subsequent process steps, it is effort-intensive and time-consuming to keep up with the detailed information required for successful semiconductor manufacturing.

The following chapters describe our concept system, which leverages recent development in computer science and artificial intelligence for automated information extraction methods from relevant text documents transferring it into a form that allows the inference of additional causal relationships.

2.1.2 Research Areas

This chapter proposes a knowledge management system for risk assessment and root cause analysis in the semiconductor industry. In specific, this chapter discusses the various system components and functionalities. Lastly, this chapter highlights the different challenges and research areas addressed for the proposed system's use cases.

The defined use case of risk assessment and root cause analysis relies on information about previous experiences. This information is documented in different data sources. The human experts' abilities to interpret various data sources, extract information, and intelligently combine the information, formulating hypotheses, are the fundamental motivation of the proposed system.

To address the motivation mentioned above, we opted for a star schema system design. The knowledge representation is the core component of the systems. The knowledge representation is responsible for the storage of

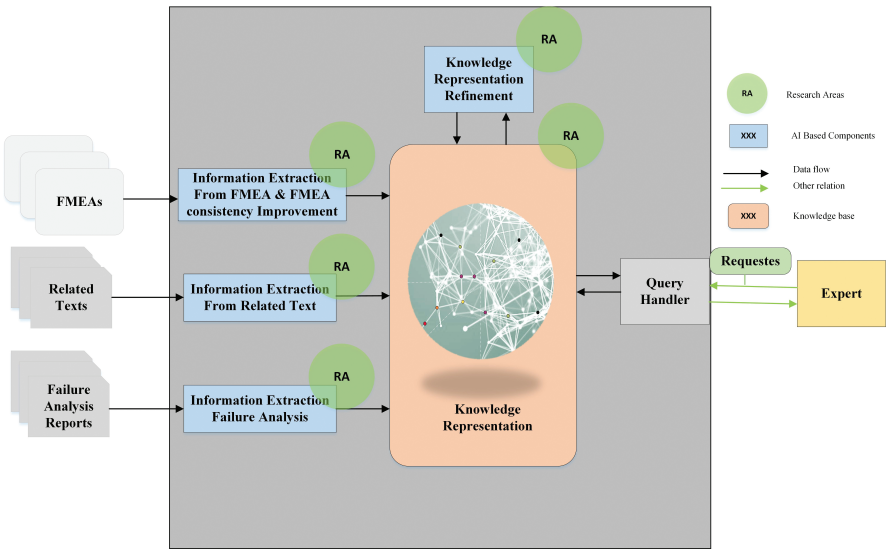


Figure 2.1.1 AI-based knowledge management concept system for risk assessment and root cause analysis in semiconductor industry.

information extracted from different sources. The failure mode effect analysis (FMEA) documents library, the Failure Analysis reports (FA reports), and other related text documents were selected as primary data sources for the proposed system. Different information extraction algorithms to account for the heterogeneity of the data sources were incorporated.

Finally, the refinement algorithms interact with the knowledge representation predicting new links that indicate causal relations, which are not present in the data sources. This is achieved by combining information extracted from different sources in an intelligent way that mimics the human experts' ability for reasoning and inference.

Figure 2.1.1 illustrates the general structure for the proposed system concept.

The proposed system concept is implemented as a demonstrator that uses three different types of documents as the data sources. However, the proposed system can be extended to accommodate more types of data sources.

Several challenges were identified and addressed as follows. The article addresses first the consistency improvement of the selected FMEA documents. Then, it presents the information extraction from FA reports. Third, the information extraction from free text is analysed. Fourth, the

knowledge representation of causal information that enables reasoning and inference is described, and finally, the refinement algorithm is addressed.

2.1.2.1 FMEA and FMEA Consistency Improvement

The well-known FMEA (failure mode effect analysis) process is performed by identifying a process map and visualizing components of the process with a multidisciplinary team. At each step in the process map, challenges that may lead to errors or potentially unsafe conditions are identified. Each of the ways that the workflow can fail is called a failure mode. A list of these failure modes and their possible causes and effects is compiled [1–3], formulating an FMEA document. In FMEA documents, relations between the columns (Effect/ Failure mode/ Root cause) are interpreted as causal relations, where designated cells in the same row are causally related, effectively representing a causal chain that identifies a single, known risk. The FMEA process supports the experts to document possible failure mechanisms reaching from the effects on the (final) product back to its potential root causes. Ideally, in a given cell, a short, descriptive text represents a single concept. In this case, a concept is a separable (identifiable) phenomenon that acts as either as (i) a root cause, (ii) an effect observed in the product characteristics, or (iii) an intermediate state in the causal chain. Since the FMEA documents are manually compiled by domain experts and the complex nature of causal relations (e.g., many to many relations with transitive perception [4]), inconsistencies are likely to occur. Crucially, any ambiguity anywhere in the FMEA documents will ultimately affect the whole causal chain due to the nature of causal relations. This is worsened by the collaborative manner in how such documents are being crafted. Typically, a group of experts from different fields (e.g., physicists, technology experts, designers) work together to formalize FMEAs, following techniques like brainstorming meetings and workshops. Due to the heterogeneity of these groups and their vastly different scopes, divergent interpretations of individual causal roles (i.e., the respective cells) are more likely to occur, contributing to the inconsistencies experienced in FMEA documents. Based on the analysis of existing FMEA documents, the majority of data inconsistencies is attributed to one of three main categories:

1. Cases of conflict in the direction of the relations, i.e., the direction of the causal effect relation, are reversed.
2. Cases of merged cells, wherein the short text of a single cell comprises multiple concepts or even relations between numerous concepts, e.g., a causal chain of multiple causes and effects.

3. Cases of missing information in the causal chain, where the documented relation describes a subsequent effect of the cause but skips its direct effect.

Additionally, anyone not closely familiar with the production process will struggle to interpret the documents, not only because of the presence of inconsistencies as mentioned above but also due to the language used in the short text, containing domain-specific terms including many abbreviations. To sum up, FMEA documents in their current state are designed to be created and maintained by experts and to be interpreted exclusively by experts. However, automatic data analysis methods lack the ability to correct for impairments in data quality [5], with merged and mixed-up data being a prime example. As such, methods for improving FMEAs data consistency and quality are highly required.

In our research, we found that just addressing the consistency purely via data-driven methods on a document level does not alleviate the problem, i.e., building a classifier to detect the content of the columns. Hence, domain knowledge needs to be considered to define a classification schema that mimics human domain experts' perception of the short text. As a response, we systematically defined: (i) a domain-specific model, consisting of concepts schema (i.e., types of cause/effects) and a relationship consistency scheme (i.e., valid relations between the concepts), and (ii) a model of inconsistencies, consisting of reverse direction of the causal relations, missing information, and merged cells. Based on real-world data, we found that the case of merged cells is in fact, the most prevalent cause of inconsistency. Moreover, the manual classification of the complete documents' library is time-consuming. Thus, our research provides a systematic study that addresses the effectiveness of variant AI-based approaches for short text classification in the semiconductors industry where domain-specific language is used. Also, our research extends the intersentential pattern mining algorithm presented in [6] to address the cases of merged cells.

Our research aims to transform the FMEAs from their current state of "experts only" to a more machine-friendly form that could achieve a higher automation degree of expert systems' methods for root cause analysis and risk assessment.

2.1.2.2 Causal Information Extracting from Free Text

Our research in causal extraction from the free text, contained in documents with no predefined structures (unstructured documents), was initiated via an initial literature survey. As a result, it has been found that there are two main

approaches, namely: (1) rule-based approaches, (2) machine-learning-based approaches. The first category is based on several hypotheses, which are briefly outlined below.

Hume's comments on causal relation: Based on the observation that cause and effect often co-occur and thus have a higher likelihood to be part of a causal relationship [7].

Transitivity: Preserving transitivity is an essential desideratum for an adequate analysis of causation [8]. For example, if e_i causes e_j, and e_j causes e_k, then the transitivity states that e_i causes e_k. Moreover, if there is another cause for e_j, e.g. e_l, it also follows that e_l causes e_k. This property is beneficial in a textual setting since causal statements are expected to be infrequent.

Suppes' probabilistic theory of causation [9]: If entity e_i causes e_j, then we will likely observe that the conditional probability given e_i exceeds the marginal probability of e_j: $P(e_j \mid e_i) > P(e_j)$.

In addition to these hypotheses, several lexical cues are indicative of causal relationships. Radinsky et al. [10] propose three types of lexical cues: (1) causal connectives (e.g., because, as, and after), (2) causal preposition (e.g., due to, because of), (3) periphrastic causative verbs (e.g., cause, lead to). Another key insight from literature is that such lexical cues are domain-dependent and thus are required to be specifically tailored towards the target text.

As an alternative to manually constructed rule-based methods, there are also machine-learning-based methods for extraction causality. However, corpora need to be annotated with ground truth for these methods to work, which is typically conducted as manual work by domain experts [11]. An example for such annotation is depicted in Figure 2.1.2, also highlighting the importance of additionally include hints for co-reference resolution (relation from "it" to "something").

Once sufficiently many sentences have been annotated with their contained causal relationships, methods from the field of supervised machine learning can be applied. Traditionally, this has been considered a sequence classification task and approached via Conditional Random Fields [12].

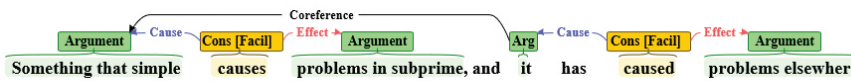


Figure 2.1.2 Manual corpora annotation example.

More recently, deep learning approaches have been adapted for these tasks, including LSTMs [13] and CNNs [14].

We identified two works as promising starting points for our setting, with the first work by Zhao et al. [15] considering extraction of pseudo causal relations from medical text. They make clear that without in-depth domain knowledge, one can only identify candidates that need to be screened by domain experts. Another aspect that makes this work of particular interest for our setting is the inclusion of intre-sentential relationships. Yu et al. [15] provided the second source of methods to extract causal relationships from the scientific literature. Here the language from scientific publications is expected to be closer to the text available as technical reports. They use the contextual word-embedding model BERT and classify sentences into four types of causal relationships as the first stage. The source code of their approach is also available for download[1].

In the semiconductor industry, many unstructured documents containing a significant amount of information are available. Extracting relevant structured information from such documents in a (semi-) automated fashion helps engineers in processing these documents more efficiently. Moreover, allow for more automation. However, the task is extremely challenging given the entirely different style in which each document is produced. Moreover, data annotating is highly effort and resource-intensive. Thus, our research aims to extract causal relations from a different source of unstructured documents in an unsupervised approach. The resulting cause and effect pairs will be used to populate the proposed system concept knowledge representation for further querying, reasoning, or inference. We consider the following sources of domain-specific unstructured texts, production tools manuals, handbooks, and PowerPoint presentations.

We devise a two-step approach, which combines rule-based and machine learning based approaches.

- Connective discovery: in this step, a rule-based approach is leveraged to distinguish sentences that contain causal cues. An initial test on publicly available causal IE dataset [16] shows that the rule-based approach achieves 80% accuracy.
- Entity extraction: in this step, a machine learning based approach is devised to identify the exact phrases for the cause and effect within a sentence that is extracted in the previous step. Specifically, we

[1]Detecting Causal Language Use in Science Findings, https://github.com/junwang4/causal-language-use-in-science

extract candidate phrases per sentence using part-of-speech tagging [17], followed by scoring using a pre-trained BERT model [18, 19]. Experiments on dataset [16] show that this simple approach achieves 40% hits@5.

The method has been tested on the real-life dataset. Initial analysis has revealed that the current assumption that a cause-effect pair will be mentioned within a single sentence is invalid. In addition, given the un-labelled data, a lack of automatic evaluation means is also an issue. Our follow-up work will address these two issues.

2.1.2.3 Failure Analysis Process, Failure Analysis Reports, and Ontologies

The Failure Analysis process aims to trace back detected failures in functional characteristics of a device to its corresponding physical defects. The Failure Analysis process is complex and requires significant knowledge about the device and the different diagnostic tools. Moreover, The Failure Analysis process includes performing experiments and analyzing their outcomes. Finally, the Failure Analysis process outcome is documented in a Failure Analysis report (FA report) report.

The FA report is an unstructured text document that summarizes the entire investigation process of a single device, i.e., the set of hypotheses, experiments, obtained measurements, and their implications. A set of possible hypotheses can be interpreted as the causal model that human domain experts rely on while conducting the Failure Analysis process. Our research aims to extract this causal information from experts and reports, boosting expert systems methods for the Failure Analysis process. The FA report primarily consists of unstructured text. Hence, human domain experts are free to report their work according to their personal preferences. However, the articulation of findings might vary considerably. For example, complete sentences, paragraphs, bullet points, and tables are commonly used in FA reports.

In the semiconductors industry, the acquisition of human domain experts' knowledge and its storage in a machine-readable form paves the way for applying AI methods that consider domain knowledge automatically. Various knowledge representation methods can be used to encode human domain experts' knowledge, e.g. standard definitions of terms used in the reports. In the Failure Analysis domain, we opted to formalize human domain experts'

knowledge as an ontology (a well-studied knowledge representation approach designed to store terminological definitions, allowing to structure them in a hierarchical manner) [20]. The Web Ontology Language (OWL), commonly used to define ontologies, allows for the articulation of human domain experts' knowledge. Moreover, given its formal logic-based semantics, OWL ensures that the formulated statements are interpreted unambiguously. Thus, the Failure Analysis ontology formulated using OWL language includes three main definitions:

1. Individuals describe real-world entities, like a job's integrated circuits sample or tools available in a lab.
2. Classes define parts of the Failure Analysis domain by summarizing properties of a collection of individuals, e.g., class OpticalMicroscope comprises all individual microscopes installed in the lab. Also, class IncomingInspection including all individuals describing applications of this method.
3. Properties determine relations between two individuals, e.g., property uses tool links class method, a superclass of class IncomingInspection, with class tools, a superclass of class OpticalMicroscope.

By analyzing a given FA report, human domain experts are able to trace all the laboratory processes, from the first visual inspections of the device to the key method leveraged to identify the fault. Moreover, the ontology provides complementary information that describes the human domain experts' knowledge (not existing in the FA reports). In consequence, the goal of the proposed artificial intelligence tool, which is used in the FA laboratory, is to map the written report to the conceptual knowledge contained in the ontology. We hypothesize that mapping the written report to the conceptual knowledge contained in the ontology allows for further AI-based algorithms. Moreover, as future work algorithms, which capture FA knowledge (reports mapped to the ontology), could be leveraged for diverse tasks such as:

1. Incorporate the language model for FA reports consistency improvements.
2. Offer a centralized search tool for all FA-related knowledge – previous reports, knowledge management database, etc.
3. Assess during the different stages of the FA job, applying the knowledge acquired from previous reports through suggestions and statistical knowledge.

2.1.2.4 Knowledge Representation

No-SQL databases, including graph databases, showed their effectiveness while working with distributed data [21]. Moreover, recent advances in graph embedding algorithms show promising results for downstream tasks such as node classification and link prediction [22]. Therefore, we opted for No-SQL databases in specific graph databases as a framework for the knowledge representation for the proposed system concept.

Moreover, the proposed system concept use case addresses risk Assessment and root cause analysis. In the proposed system concept use case, causal relations are the main relation types of interest. However, humans' ability to interpret causal information makes causal relations deceptively simple. Hence, causal relations are context-dependent [4]. Causal information representation is gaining more interest in many research disciplines to increase the understandability of automated decision-making systems [23].

In the semiconductor industry, the context information is highly variant from one product to another and from one process to another. Therefore, an event that occurs in multiple contexts (i.e., different manufacturing processes or different products) might have a completely different meaning. Consequently, the causal relation between two events might change or even disappear depending on the context. However, human domain experts are able to judge the possibility of the extension of the causal relations between the contexts.

Our research addresses causal knowledge representation in the semiconductor industry manufacturing studying the effectiveness of incorporating context information with regards to the inferencing and reasoning algorithms.

2.1.2.5 Refinement Algorithm

Real-life knowledge graphs, such as those constructed in this work, are often orders of magnitude sparser than benchmark knowledge bases like Freebase, especially if they are built via (automatic) extraction methods from textual corpora. The textual information stored in the text attributes is commonly used to compensate for the lack of structure in a sparse graph. For a recent example, see ref [22]. Language models such as BERT are preferred choices for generating low dimensional representations for the textual information, as they have been shown to consistently improve the performance of a large variety of Natural Language Processing (NLP) tasks [24].

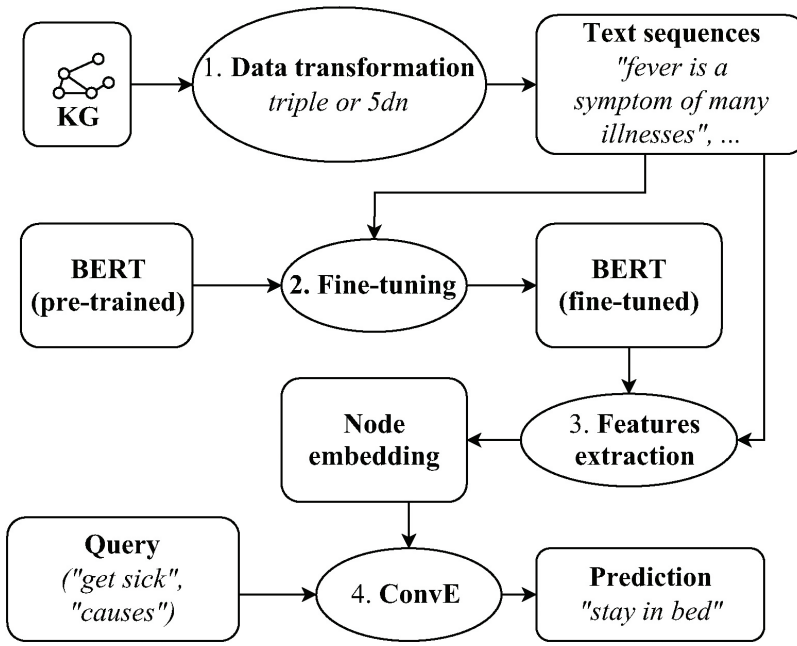


Figure 2.1.3 BERT-ConvE workflow.

Building upon existing work, this work also builds a BERT-based method (BERT-ConvE) to exploit transfer learning of BERT (fine-tuning) in combination with a convolutional network model ConvE [25]. See Figure 2.1.3 for an overview of the main components of the BERT-ConvE method, together with the general workflow. Experiments on ConceptNet [26] show that the proposed method outperforms strong baselines by 50% on knowledge graph completion tasks. The proposed method is suitable for sparse graphs as also demonstrated by empirical studies on ATOMIC [27] and sparsified-FB15k-237 datasets [28]. The next step is to apply the proposed method to the knowledge graph constructed in this work.

However, to our best knowledge, challenges in respect to the use of domain-specific languages, in specific the semiconductor industry, have not been addressed before. Moreover, context information plays a vital role in the correctness of the causal relation. The context information in the proposed use case indicates the settings in which the causal relation is present between two events (the cause and the effect). The prosed concept system addresses the prediction of causal relations, which is context-dependent. Thus, it introduces more sparsity to the resulted knowledge graph.

Our research addresses the causal knowledge graph completion challenge in industrial settings, considering the domain-specific language, structural information, and context information.

2.1.3 Reflections

We presented an AI-based knowledge management system for Risk Assessment and Root Cause Analysis in the Semiconductor Industry. The proposed system process various types of documents where causal relationships are being captured. As these documents were originally intended for interpretation by human experts and created by people of different backgrounds, these documents, in their current form, require additional information extraction algorithms.

Moreover, due to the manual creation of some of these documents and the nature of causal relation, some of these documents i.e., FMEAs, tend to contain a number of inconsistencies calling for consistency checks and automated means of quality improvements. As a response, we systematically defined: (i) a domain-specific model, consisting of a concept schema (i.e., types of cause/effects) and a relationship consistency schema (i.e., valid relations between the concepts), and (ii) a model of inconsistencies, consisting of mixed-up cells (including reverse direction), missing information, and merged cells.

Regarding the FA reports, the unstructured nature of their content (free text) requires a different approach than those utilized in FMEAs. Thus, the discovery of causal relations, among others, is handled with an ontology. The ontology, a formal set of descriptions of terms regarding the Failure Analysis work and the different links between them, is then utilized to map the reports. We hypothesize that this data structure will allow us to develop further AI-based algorithms such as language models.

Also, our research areas address causal knowledge representation in the semiconductor industry manufacturing and aim to study the effectiveness of incorporating context information regarding the hypotheses generation (i.e., predicting possible causal links in causal knowledge graph) methods. Moreover, it addresses the causal knowledge graph completion method in industrial settings. Hence, our research considers sparsity and noise introduced by automated information extraction, sparsity presented by the causal relations context information, and domain-specific language.

Finally, we devised a knowledge graph embedding method BERT-ConvE, that effectively exploits transfer learning and context-dependency of BERT

in combination with a convolutional network model, ConvE. Experiments on knowledge graph completion task on publicly available knowledge graphs (ConceptNet, ATOMIC, sparsified FB15k-237) has shown that BERT-ConvE is suitable for sparse knowledge graphs where structural information is limited and textural information is informative for reasoning over the graph.

2.1.4 Conclusion

In conclusion, while human domain experts remain the key source of knowledge, the proposed system aims to mimic their ability to extract information from different data sources and extend knowledge between different scenarios to support the experts in their repetitive tasks. Moreover, although the proposed system concept may appear simple in design, the proposed use case (Risk Assessment and Root Cause Analysis in Semiconductor Industry) pushes the boundaries of many states of the art methods of artificial intelligence and natural language processing. Furthermore, our research areas highlight novel approaches to address causal domain knowledge information extraction, representation, and completion, leveraging a combination of advances in computer science, artificial intelligence, and natural language processing.

Acknowledgments

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] L. Ashley and G. Armitage, "Failure mode and effects analysis: an empirical comparison of failure mode scoring procedures," *Journal of patient safety*, vol. 6, no. 4, pp. 210–215, 2010, doi: 10.1097/pts.0b013e3181fc98d7.

[2] M. Scorsetti *et al.,* "Applying failure mode effects and criticality analysis in radiotherapy: lessons learned and perspectives of

enhancement," *Radiotherapy and oncology: journal of the European Society for Therapeutic Radiology and Oncology*, vol. 94, no. 3, pp. 367–374, 2010, doi: 10.1016/j.radonc.2009.12.040.

[3] N. Viscariello *et al.,* "A multi-institutional assessment of COVID-19-related risk in radiation oncology," *Radiotherapy and oncology : journal of the European Society for Therapeutic Radiology and Oncology*, vol. 153, pp. 296–302, 2020, doi: 10.1016/j.radonc.2020.10.013.

[4] N. McDonnell, "Transitivity and proportionality in causation," *Synthese*, vol. 195, no. 3, pp. 1211–1229, 2018, doi: 10.1007/s11229-016-1263-1.

[5] Erik HOLLNAGEL, "Evaluation of Expert Systems," in *Studies in Computer Science and Artificial Intelligence, Topics in Expert System Design*, Giovanni GUIDA and Carlo TASSO, Eds.: North-Holland, 1989, pp. 377–416. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780444873217500193

[6] J.-L. Wu, L.-C. Yu, and P.-C. Chang, "Detecting causality from online psychiatric texts using inter-sentential language patterns," *BMC Medical Informatics and Decision Making*, vol. 12, no. 1, p. 72, 2012, doi: 10.1186/1472-6947-12-72.

[7] K. A. Rogers, "Hume on Necessary Causal Connections," *Philosophy*, vol. 66, no. 258, pp. 517–521, 1991, doi: 10.1017/S0031819100065165.

[8] L. A. Paul, N. Hall, and E. J. Hall, *Causation: A user's guide*: Oxford University Press, 2013.

[9] Julian Reiss, "Suppes' probabilistic theory of causality and causal inference in economics," *Journal of Economic Methodology*, vol. 23, no. 3, pp. 289–304, 2016, doi: 10.1080/1350178X.2016.1189127.

[10] K. Radinsky, S. Davidovich, and S. Markovitch, "Learning causality for news events prediction," in *Proceedings of the 21st international conference on World Wide Web*, Lyon, France: Association for Computing Machinery, 2012, pp. 909–918.

[11] J. Dunietz, L. Levin, and J. Carbonell, "Annotating Causal Language Using Corpus Lexicography of Constructions," in *Proceedings of The 9th Linguistic Annotation Workshop*, 2015, pp. 188–196. [Online]. Available: https://www.aclweb.org/anthology/W15-1622

[12] C. Mihăilă and S. Ananiadou, "Recognising discourse causality triggers in the biomedical domain," *Journal of bioinformatics and computational biology*, vol. 11, no. 06, p. 1343008, 2013.

[13] *Automatic extraction of causal relations from text using linguistically informed deep neural networks*, 2018.

[14] *Relation extraction: Perspective from convolutional neural networks*, 2015.

[15] *Causaltriad: toward pseudo causal relation discovery and hypotheses generation from medical text data*, 2018.

[16] I. Hendrickx *et al.,* "SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals," in *Proceedings of the 5th International Workshop on Semantic Evaluation*, 2010, pp. 33–38. [Online]. Available: https://www.aclweb.org/anthology/S10-1006

[17] M. Honnibal, I. Montani, S. van Landeghem, and A. Boyd, *spaCy: Industrial-strength Natural Language Processing in Python*: Zenodo.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 4171–4186. [Online]. Available: https://www.aclweb.org/anthology/N19-1423

[19] Thomas Wolf *et al., HuggingFace's Transformers: State-of-the-art Natural Language Processing*.

[20] D. Allemang and J. Hendler, *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL. Second Edition*, 2011.

[21] Z. Wei-ping, L. Ming-xin, and C. Huan, "Using MongoDB to implement textbook management system instead of MySQL," in *2011 IEEE 3rd International Conference on Communication Software and Networks*, 2011, pp. 303–305.

[22] D. Q. Nguyen, "A survey of embedding models of entities and relationships for knowledge graph completion," *arXiv: 1703.08098*, 2017.

[23] B. Schölkopf *et al.,* "Toward Causal Representation Learning," *Proceedings of the IEEE*, vol. 109, no. 5, pp. 612–634, 2021, doi: 10.1109/JPROC.2021.3058954.

[24] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 4171–4186. [Online]. Available: https://www.aclweb.org/anthology/N19-1423

[25] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2d knowledge graph embeddings," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[26] R. Speer and C. Havasi, "ConceptNet 5: A large semantic network for relational knowledge," in *The People's Web Meets NLP*: Springer, 2013, pp. 161–176.
[27] *Atomic: An atlas of machine commonsense for if-then reasoning*, 2019.
[28] J. Pujara, E. Augustine, and L. Getoor, "Sparsity and Noise: Where Knowledge Graph Embeddings Fall Short," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 1751–1756. [Online]. Available: https://www.aclweb.org/anthology/D17-1184

2.2

Efficient Deep Learning Approach for Fault Detection in the Semiconductor Industry

Liliana Andrade[1], Thomas Baumela[1], Frédéric Pétrot[1], David Briand[2], Olivier Bichler[2] and Marcello Coppola[3]

[1]Univ. Grenoble Alpes, CNRS, Grenoble INP, TIMA, France
[2]CEA-LIST, France
[3]STMicroelectronics, France

Abstract

The semiconductor industry is a very cost sensitive industry and yield is key to profitability. The ability to analyse and detect the faulty parts at several manufacturing steps is also very important to ensure the quality of the delivered integrated circuits. Several factors as alignments, shifts or masks rotations can lead to errors during the front-end step (wafer fabrication), or others causes such as fingerprints, scratches and stains can cause cosmetic damage during the back-end step (silicon packaging). Therefore, an automatic visual inspection is required to ensure that the parts are free of any defects. In this chapter, we focus specifically on classifying wafer maps according to predefined defaults. We propose a platform which aims at making the classification process more energy efficient, by means of the interconnection of two hardware parts. The first one, the microprocessor STM32MP1, is responsible for image pre-processing and for offloading inference to a dedicated hardware accelerator. The second one, the hardware accelerator, is implemented in a Xilinx Zybo Z7-20 FPGA and uses a quantized neural network model. Preliminary results show that, for this low throughput applications that has a limited number of classes, the solution presented in this article can classify in real-time with accuracy above 80% using limited resources.

Keywords: classification, wafer maps, deep learning, quantized neural networks, embedded artificial intelligence, HW/SW integration, hardware acceleration of AI, high-level synthesis, field-programmable gate array.

2.2.1 Motivation: The Wafer Fault Classification Problem

Defect inspection and classification are significant steps of most manufacturing processes in the semiconductor industry. These steps are needed during the front-end process, when wafers come out of the foundry, and during the back-end process, when packaging individual chips. Having a proper inspection to partition wafers, dies, or packages in correct vs incorrect items is needed not only to ensure delivering working packaged chips to customers, but also to improve the quality of the manufacturing process. Similarly, accurate classification of the defaults, be they during the front-end or back-end process, is key to high yield. Indeed, yield management and yield learning help the manufacturing process engineers in determining the causes of abnormal fabrication. To ensure a high-level of quality and yield, still today, many of the inspection phases are performed visually, by humans [1]. Given the throughput on the production lines, the actual inspection can only be done on samples, which leaves quite some room for improvement.

In this chapter, we focus on the front-end process, and more specifically on the detection and classification of wafer defects using deep neural networks (DNN), which have proven to be efficient for classifying images. Early detection of defects at the back-end process, during which wafers are produced and electrically tested, can help to readjust certain parameters in the production line, to increase yield and thus reduce costs. This explains why a lot of effort has been devoted to this topic since the infancy of mass production of integrated circuits.

Once wafers have been fabricated and the electrical inspection step carried out, different 2D images are generated indicating which dies are working properly and which are not. These images, known as wafer maps, must then be inspected to extract their features and classified into different categories. It will allow to determine if all chips on the wafer can go to packaging, in the case of lack of defects, or if the wafer presents a specific failure pattern indicating an issue during some of the production stages. Specific failure patterns may indicate the root cause of a problem. For instance, when several faulty dies appear randomly on a wafer without

following any specific pattern, the problem may come from the presence of dust particles transferred to the wafer surface; when a ring of faulty dies is present, the issue is due to a misalignment of several layers during photolithography; when defects are located in the centre of the wafer or following a donut pattern they may indicate a non-uniform application of forces to smooth and flatten, silicon wafer, a non-uniform temperature distribution or also a problem during oxidation; when some streaks run across the wafer surface it may be due to a human error in handling equipment or due to an issue during the chemical-mechanical polishing stage; or even when falling dies are located near to the edge of the wafer, this can indicate an issue during etching or a non-uniform cleaning.

Wafer data, because it could leak information on the process and possibly its yield, is very sensitive, and manufacturer are unwilling to share it. The work presented in this chapter will therefore be based on a public and open access dataset that has been "anonymized" and then donated to the community by TSMC: the WM-811K wafer map dataset [2]. The wafer production line throughput is low compared to general purpose computer vision applications, and the construction of the wafer maps is the result of a process which also involves test equipment [3]. However, since the machines are working 24/7, 365 days a year for continuous monitoring of the production quality, we seek solutions that are both accurate and low power. In addition to these constraints, and given the confidentiality of the data processed, being able to perform on-device classification instead of sending the data to a remote server is also of importance. To that end, we choose to investigate the use of highly quantized artificial neural networks to be implemented on small industry grade micro-controllers, possibly enhanced with hardware accelerators on FPGA. After having defined in this section the problem at hand, we organize this chapter as follows. Section 2.2.2 presents past works related to it while section 2.2.3 details the requirements and functional specifications of the system. Section 2.2.4 presents the method we propose and some preliminary results. Finally, section 2.2.5 wraps-up the chapter and presents possible extensions to this work.

2.2.2 Related Works

Recently, many authors have proposed different techniques for automatic detection and classification of failure wafer patterns, either using Machine Learning (ML) or Deep Learning (DL) techniques.

On the one hand, we find some approaches where a feature extraction is performed on wafer maps to obtain a reduced representation ready to be analysed and classified. In this context, different authors have highlighted the use of ML techniques, generally applied in computer vision. Some of these techniques allow for example the feature extraction using the Hough transform, the generation of probability distributions used to define specific-faulty regions, or the use of k-nearest-neighbour classifiers to distinguish faulty patterns. A key research implementing ML techniques is presented in [2]. It introduces a new set of features which, requiring low computation and storage, is used to obtain a reduced representation of wafer maps, to identify wafer maps failure patterns and to support recovery of similar failures in other wafer maps. The proposed approach applies support vector machines as classifier, preserves the rotation-invariant attribute in wafers maps and reduces the computational cost with respect to different approaches carrying spatial analysis between features maps. This work is considered as a reference because it is at the origin of the WM-811K dataset used for the experiments presented in this chapter.

On the other hand, we find several approaches implementing DL techniques, which have seen an exponential growth in the last years. For example, Alawieh et al. [4] propose a wafer map classification using deep selective learning and implement a reject technique where model refrain from predicting class label when the miss-risk is high. This can usually happen when during classification some wafers show default patterns that have never been seen during training. Also, Convolutional Neural Networks (CNN) have demonstrated great potential to recognize and classify patterns without carrying manual feature extractions. Using convolution layers, they can perform automatic feature extraction; using pooling layers they can summarize the last extraction by reducing the features maps size; and using fully-connected layers they can efficiently classify patterns into well-separated categories. As described below, several studies have been conducted to detect and classify wafer defect patterns using CNN. Kyeong and Kim [5] address the problem of detecting mixed-type defect patterns, this means to have different defect patterns combined in the same wafer. Authors propose a single approach building individual CNN-based classification models for each pattern and determining the final class by combining the results of multiple individual models. Jang et al. [6] implement a one-vs-one model that uses a CNN as base classifier. Their technique consists of determining a weighted mean score from failure bit count wafer maps (grey-scale images) and then, based on this score, they determine the presence

or absence of failures. Failure detection is performed calculating the score proximity in relation to a data group learned in a feature space. In principle to address a high-quality classification of wafers maps using CNN, having a set of examples containing a large quantity of labelled patterns is required. These patterns are the key to fit the parameters in DNN before inference. Sometimes, even having a large quantity of patterns is still not enough, but it is also required to have as much as possible a balanced dataset. That is, a set of examples where the proportion of wafer maps in each class is almost the same. As it is difficult to achieve, domain expert engineers are required for labelling data coming from the manufacturing process in the form of wafer maps. As this process represents a significant cost, several authors [7], [8], [9], [10], [11] have worked on different techniques to avoid the use of unbalanced datasets and to automatically increase the sets of examples reducing the intervention of experts.

Although many of the presented solutions achieve high performance, none of them have been specifically designed to be implemented on small embedded devices. We are interested to address this challenge by using quantized neural networks, which in our knowledge have never been used for classifying faulty wafer maps.

2.2.3 Target Platform Requirements

Based on these experiences and considering the increasing need to detect and classify defects in an automatic, real-time and power-efficient way, we define the requirements for an automatic wafer defect detection platform targeting high-power efficiency and real-time inference. While this platform should be generic enough to support different applications, it will primarily target detection and classification of faulty wafer maps. We rely on the requirements presented below to enable industrial scanning equipment to efficiently address the aforementioned problem.

- *Define a deep learning classification platform that can be programmed and its hardware partially reconfigured*: When we refer to deep learning, we evoke the new programming paradigm where humans provide input data and expected responses, and a layered system, better known as DNN, processes inputs and stores a meaningful representation that can be later used to perform tasks automatically, for example recognizing a set of images. The stage where the system transforms input data and stores a representation of it in form of parameters, also known as

weights, is called *learning*. The appropriate selection of the weights associated with each neural network layer is performed by first assigning random values and computing a temporal prediction of the network from a set of inputs, then comparing that prediction with respect to the expected response (through a loss function), and using a back-propagation algorithm (usually implemented by an optimizer) to adjust the weights in the correct direction [12]. Once the system has learned an enough representative input dataset, it can be used to perform automatic classification tasks also called *inference*. For learning, we will follow the approach in which the neural network parameters are computed and refined off-line, before implementing a HW/SW model in the industrial equipment via micro-controllers and small reconfigurable devices.

- *Design an efficient DNN model to be implemented in hardware*: We will focus on neural network models with small number of parameters and on quantization techniques to increase power efficiency during classification, without neglecting high-throughput. On modern high-end front-end equipment, the throughput in terms of wafer per hour is between 150 and 300. Assuming that the electrical characterization and test equipment is dimensioned to work at that same throughput, this means that the analysis must be performed in a 20 s to 40 s time frame. It is thus neither useful nor economically sensible to reach for throughputs like those required by general purpose video processing.
- *Use real images which undergo classical linear time pre-processing before being fed to a DNN implemented by the classification platform*: We will use the WM-811K public dataset provided by the Multimedia Information Retrieval (MIR) laboratory. It contains 811457 real wafer maps collected from 46393 lots of real-world fabrication. The 2D images provided in this dataset have different sizes and 172951 ($\sim$20%) of these images were manually labelled by domain experts using nine patterns (Figure 2.2.1): no-defects (85.2%), edge-ring (5.6%), edge-local (3.0%), center (2.5%), local (2.1%), scratch (0.7%), random (0.5%), donut (0.3%) or near-full (0.1%). As observed by different authors, the challenge with this dataset is that it is unbalanced, then image pre-processing and data augmentation will be required to improve classification accuracy.

2.2.4 HW/SW System and Methodology

2.2.4.1 Industrial HW/SW System for On-Device Inference

We propose a platform allowing the integration, in a reconfigurable device, of a neural network model trained upstream with a set of reference wafer maps, as well as the classification of new faulty wafers by means of a dedicated HW/SW architecture.

The hardware architecture of the platform consists of two main boards (Figure 2.2.2). One STM32MP1 board interfacing with the physical world (i.e., the wafer production line), and one Zybo-Z7 board for the wafer fault classification. The STM32MP1 is an industrial grade master board including an ARM dual-core Cortex-A7 and an additional Cortex-M4, DDR memory and a good set of peripherals, in particular a 1GBps Ethernet chipset, USB device connectors and an HDMI output connector. The Zybo-Z7 embeds the XC7Z020 SoC from Xilinx, featuring 667MHz dual-core Cortex-A9 processor, 1GB DDR3L memory, a 1GBps controller as well as an FPGA. Both boards communicate through a GBps Ethernet link, allowing the STM32MP1 to send input image to the Zybo-Z7 taking care of the inference and sending back the results through the Ethernet link. An inference cycle thus consists of: (1) receive an image from the test equipment, (2) apply scale and crop filters to get the image to the correct dimensions, (3) send it to the Zybo-Z7, (4) make the inference on the Zybo-Z7 and (5) get back the results to the STM32MP1. The Zybo-Z7 is the heart of the inference process. It integrates a hardware implementation of a neural network, taking advantage of the high parallelisation capabilities the FPGA offers. The network is integrated with a message-based interface consisting of a pair of RX/TX FIFOs connected to high-performance Scatter-Gather (S/G) DMA engine. These FIFOs are used by the network to receive configuration weights and inputs as well as send inference outputs. The DMA engine is used by the software to efficiently exchange those data and control and status commands to drive the neural network.

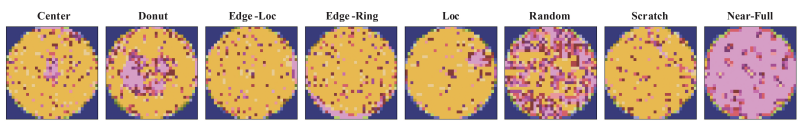


Figure 2.2.1 Example of classified wafer maps.

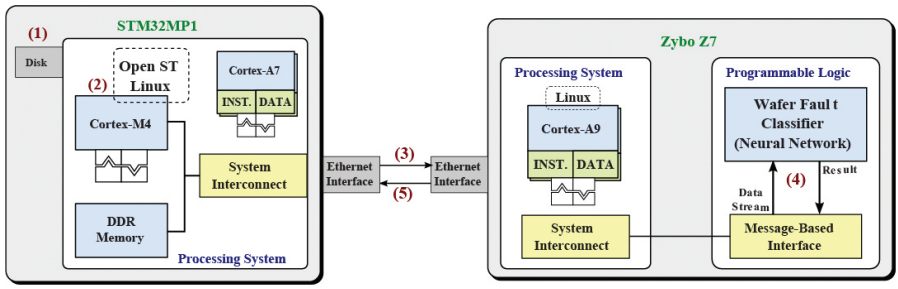


Figure 2.2.2 The platform hardware architecture.

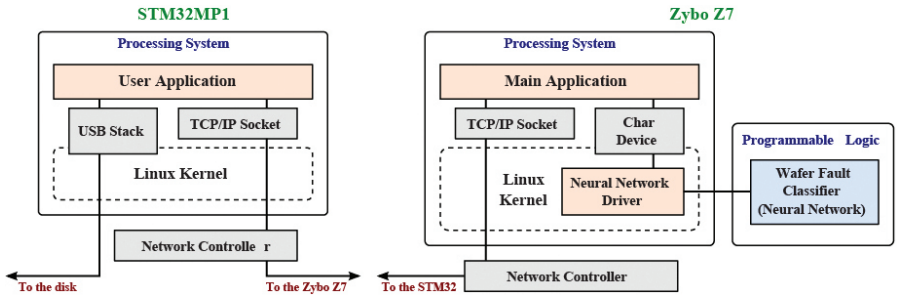


Figure 2.2.3 The platform software architecture.

The software architecture of the platform is also composed of two parts (Figure 2.2.3). The STM32MP1 runs a Linux operating system pre-configured by ST tools. It includes an application accessing the file system to send inputs to the Zybo-Z7 through the Ethernet link. It also configures the neural network by sending the weights to the Zybo-Z7. The Zybo-Z7 runs a minimal Linux operating system built using PetaLinux tools. It includes a custom driver integrating the neural network in the Linux environment, allowing user applications to control it. To do so, the driver provides a char device interface in which applications can read and write into to control the neural network. The driver implements those read and writes by driving the S/G DMA engine included in the neural network interface.

With the neural network accessible by user applications, the main application configures the Ethernet link with the STM32MP1. Once the communication with the STM32MP1 is established, the application forwards weights and inputs to the neural network and receives outputs from it.

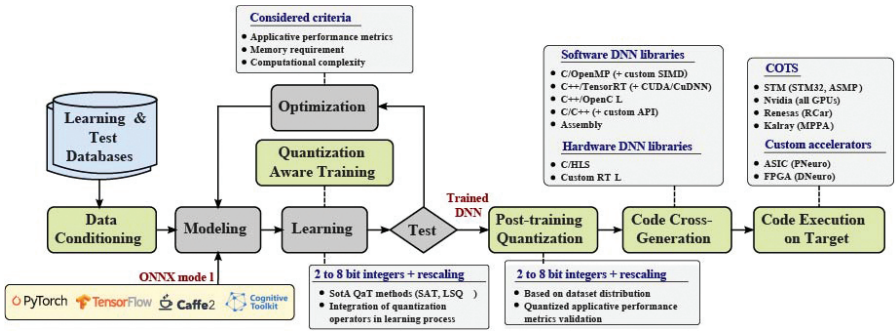


Figure 2.2.4 System design and optimization process using N2D2.

2.2.4.2 Neural Network Building and Training Using N2D2

Current research on deep neural networks extensively uses different frameworks, such as PyTorch [13] and Tensorflow [14], which allow the development of neural networks and ML algorithms. Some other frameworks, such as Keras [15], QKeras [16] or Larq [17], are interfaces to these frameworks. Unfortunately, once a network has been designed and trained, there is a gap towards its optimized deployment in an embedded and hardware constrained system. The recent development of libraries built to ease the deployment and optimization of DNN, such as TensorRT [18] or TFLite [14], reflects the rising trend of hardware integration requirements. The main weakness of these libraries remains the limitation to certain target platforms and the possible optimizations that can be applied, which are greatly linked to proprietary solutions. To train and generate a first neural model, we use the N2D2 deep learning framework [19], which reduces this gap by providing an innovative optimization method to the system designer.

N2D2 is hardware agnostic while being able to directly target most common computing architectures and parallel run-time software. As shown in the high-level view of the system design process enabled by N2D2 (Figure 2.2.4), this framework integrates a generic database handling and data processing dataflow.

The N2D2 learning core is close to the standard deep learning frameworks with the support of typical layers, operators and learning rules. Its execution on x86 and ARM processor is accelerated thanks to C++/OpenMP kernels while execution on NVidia GPUs is supported thanks to cuDNN and custom CUDA Kernels. Moreover, the N2D2 core also supports spikes simulations modelling. The input model representation of the N2D2 framework can be

made through an INI description file or thanks to the Open Neural Network Exchange (ONNX) format [20], allows the user to load a pre-trained neural network from another deep learning framework.

Among the key features of the N2D2 framework, the integrated quantization module remains one promising technique to optimize a deep learning model for a wide range of hardware accelerators. Quantization refers to the process of reducing the number of bits that represent a number, without performance degradation. In the context of deep learning, the predominant numerical format used for research and for deployment has so far been 32-bit floating point, or FP32. However, the desire for reduced bandwidth and compute requirements of models has driven research into using lower-precision numerical formats. It has been extensively demonstrated that weights and activations can be represented using 8-bit integers (or INT8) without incurring significant loss in accuracy. The use of even lower bit-widths, such as 4/2/1-bits, is an active field of research that has also shown great progress. The more obvious benefit from quantization is significantly reduced bandwidth and storage. For instance, using INT8 for weights and activations consumes 4x less overall bandwidth compared to FP32. Additionally, integer compute is faster than floating point compute. It is also much more area and energy efficient. Note that very aggressive quantization can yield even more efficiency. If weights are binary{-1, 1} [21], [22] or ternary{-1, 0, 1} [23], [24], then convolution and fully-connected layers can be computed with additions and subtractions only, removing multiplications completely. A lot of techniques have been proposed recently to quantize neural networks. These techniques can be classified into two types: Post Training Quantization (PTQ), which quantizes both weights and activations for faster inference, without requiring to re-train the model; Quantization Aware Training (QAT), which models quantization during training and can provide higher accuracies than post quantization training schemes. Both techniques are integrated into N2D2. However, QAT is currently the best technique to provide highest accuracies for heavily quantized networks, with bit-widths as low as 4/2/1-bits for weights and/or activations. N2D2 integrates both Learned Step Size Quantization (LSQ) [25] and Scale-Adjusted Training (SAT) [26] [27] state-of-the art QAT algorithms, the latter one being one of the most promising solutions, both in term of implementation complexity, flexibility and accuracy. The quantization aware training in N2D2 is performed by a full precision learning phase with weights clamping; and quantization learning phase, with the same hyperparameters

by using a transfer learning method from the previously clamped weights.

We use N2D2 to build and train a first neural network model. Three kinds of networks are tested, all based on convolutional layers, with various complexity. First, a simplified AlexNet made of 3 convolutional layers with MaxPooling, followed by 2 fully connected layers and a SoftMax activation layer. Second, a simplified VGG with 5 convolutional blocks (made of 2 or 3 convolutional layers) of increasing size, with MaxPooling, followed by 2 fully connected layers and a SoftMax activation layer. Third, a MobileNet V1, which uses depthwise separable convolutions in place of the standard convolutions to provide lighter models. The version used here has 27 layers (26 groups of 'convolution + batch normalization' and 1 fully connected + softmax layers).

Before training, the images in the WM-811K dataset are homogenized. They are rescaled to a common size. Sizes of 42x42 pixels and 64x64 pixels were tested. As mentioned above, the dataset is very unbalanced, as the 'no defect' class has much more images than the others. We then decided to limit the scope of the application to the first eight classes and discard this last class, for a total of 17625 training images and 7894 test images. We help the network converge to a correct solution, by applying data augmentation (Random rotation and horizontal/vertical image flipping) during the training phase. To decrease the memory usage, images are also transformed to Grayscale and normalized (colour range moved from 0-255 to 0-1) before applying the data augmentation strategy.

After training, we observed that although the topology of the simplified AlexNet is much simpler, it uses much more weights and biases than the other networks for a total of 2,478,632 numbers to store. The performances were not better, so this network was abandoned. The simplified VGG network requires around 600,000 parameter storage (depending on the presence of batch normalization layers) which is much lighter. In comparison, the MobileNet V1 requires a little bit more with 823,752 parameters. The best performances were obtained with the VGG network with 98.2% recognition on the training set and 81.0% on the test set (1 hour and 38 minutes training and 2000 training epochs). After applying post-training quantization (8-bits), the performance on the test set remained at 80.4% (0.6% loss) for images size of 42x42 pixels.

2.2.4.3 Neural Network Export and FPGA Implementation Used for Inference

The inference platform is implemented in hardware, targeting an FPGA based platform (the Zybo-Z7 board in our proposed platform). The implementation is performed using Vivado-HLS, Xilinx's High-Level-Synthesis tool from C++ programs. The neural network implementation process is divided in two phases. First, a C++ export, which could be for example the reference N2D2 export. This export is expected to provide an implementation in which the structural parameters of the layers, e.g., loop boundaries, are passed as template parameters. This allows heavy compile-time inlining, optimization and loop unrolling. Second, the neural network FPGA implementation, which consists of modifying the C++ implementation to make it fit for HLS synthesis. Indeed, the C++ implementation does not target HLS due to language and library limitations such as dynamic memory allocation, printfs, file system access. In addition, a set of pragmas must be added to guide the synthesis tool in order to get a properly optimized network. The most important pragmas are array modifiers and loop unrolling directives. Array modifiers (called *array_map*, *array_partition* and *array_reshape* in Vivado-HLS) are used to optimize the structure of arrays by splitting them in smaller arrays. It also allows to merge small words, for instance our 2-bit wide weights, into bigger words allowing to fetch several small words in the same cycle. Having efficiently structured arrays, in particular the one storing weights in Block RAMs (BRAMs), allows to completely unroll some loops as data contained in these arrays can be accessed in one clock cycle by all the parallelized iterations. Unrolling loops is performed using the *unroll* pragma. It requires reordering the nested loops in a way that will allow the synthesis tool to properly unroll and take advantage of the array structures we defined. Optimizing the usage of BRAM is key to make a network fit entirely in an FPGA, even with low parallelization settings, to avoid time and power consuming external memory accesses.

We report the resource usage for a preliminary experiment on a fully-connected input layer for 42x42 wafer maps (Figure 2.2.5). These results show the evolution of resource usage against the bit-size of weights. The important point enlighten by these experiments is that lowering the size of weight is key to make a network fit inside a given FPGA. Of course, reducing the weight size leads to a loss of accuracy, though this loss can be mitigated by increasing the number of neurons in the network. For instance, reducing the weight size on a 100-neuron network with 8-bit weights to a 2-bit weights

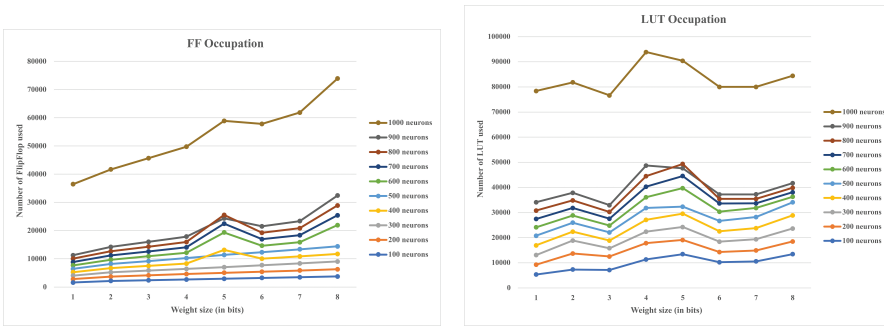


Figure 2.2.5 Resource occupation (FF and LUT) for 42x42 wafer maps.

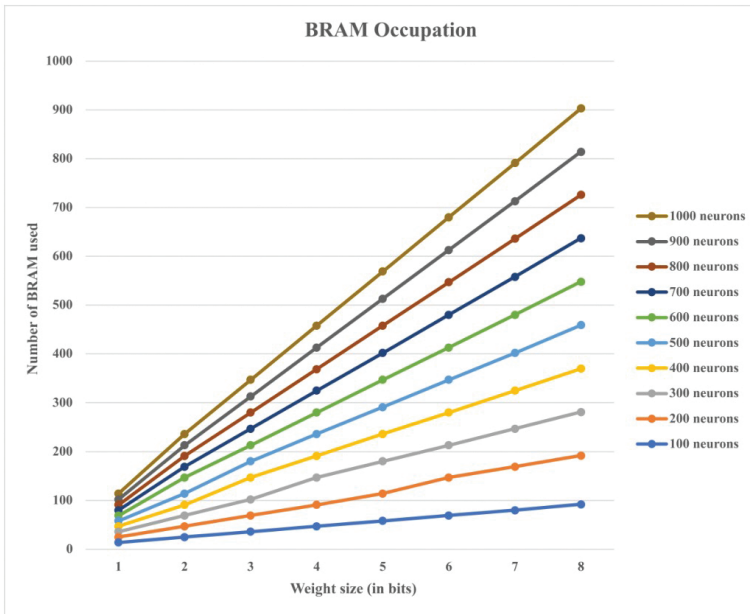


Figure 2.2.6 BRAM occupation for 42x42 wafer maps.

network allows to increase the number of neurons to 400 while occupying the same amount of BRAMs (Figure 2.2.6).

2.2.5 Conclusion

Process control in the semiconductor industry is a major issue. In this article, we present the approach we propose, that is suited to the low throughput of the

wafer production line. It is an AI hardware/software based solution running on a small industry grade device which aims at analysing wafer maps in real-time. We report preliminary experiments showing first that highly quantized neural networks, when trained appropriately, can reach high accuracy, and second, that the hardware implementation of these networks can be very resource efficient.

The next step is to smooth the integration between the tools and generalize hardware support to larger classes of network layers.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] M.-J. Wang and C.-L. Huang, "Evaluating the Eye Fatigue Problem in Wafer Inspection," IEEE Transactions on Semiconductor Manufacturing, vol. 17, no. 3, pp. 444-447, 2004.

[2] M.-J. Wu, J.-S. Jang and J.-L. Chen, "Wafer Map Failure Pattern Recognition and Similarity Ranking for Large-Scale Data Sets," IEEE Transactions on Semiconductor Manufacturing, vol. 28, no. 2, pp. 1-12, 2015.

[3] F. Duvivier, "Automatic detection of spatial signature on wafermaps in a high volume production," in International Symposium on Defect and Fault Tolerance in VLSI Systems, Albuquerque, NM, USA, 1999.

[4] M. B. Alawieh, D. Boning and D. Z. Pan, "Wafer Map Defect Patterns Classification Using Deep Selective Learning," in ACM/EDAC/IEEE Design Automation Conference, Virtual Event, USA, 2020.

[5] K. Kyeong and H. Kim, "Classification of Mixed-Type Defect Patterns in Wafer Bin Maps Using Convolutional Neural Networks," IEEE Transactions on Semiconductor Manufacturing, vol. 31, no. 3, pp. 395-402, 2018.

[6] J. Jang, M. Seo and C. O. Kim, "Support Weighted Ensemble Model for Open Set Recognition of Wafer Map Defects," IEEE Transactions on Semiconductor Manufacturing, vol. 33, no. 4, pp. 635-643, 2020.

[7] T. Nakazawa and D. V. Kulkarni, "Wafer Map Defect Pattern Classification and Image Retrieval Using Convolutional Neural Network," IEEE Transactions on Semiconductor Manufacturing, vol. 31, no. 2, pp. 309-314, 2018.

[8] M. Saqlain, Q. Abbas and J. Y. Lee, "A Deep Convolutional Neural Network for Wafer Defect Identification on an Imbalanced Dataset in Semiconductor Manufacturing Processes," IEEE Transactions on Semiconductor Manufacturing, vol. 33, no. 3, pp. 436-444, 2020.

[9] J. Shim, S. Kang and S. Cho, "Active Learning of Convolutional Neural Network for Cost-Effective Wafer Map Pattern Classification," IEEE Transactions on Semiconductor Manufacturing, vol. 33, no. 2, pp. 258-266, 2020.

[10] R. Wang and N. Chen, "Defect Pattern Recognition on Wafers using Convolutional Neural Networks," Quality and Reliability Engineering International, vol. 36, no. 4, pp. 1245-1257, 2020.

[11] T. -H. Tsai and Y. -C. Lee, "A Light-Weight Neural Network for Wafer Map Classification Based on Data Augmentation," in IEEE Transactions on Semiconductor Manufacturing, vol. 33, no. 4, pp. 663-672, Nov. 2020. Available online at: https://doi.org/10.1109/TSM.2020.3013004.

[12] F. Chollet, Deep Learning with Python, USA: Manning Publications Co., 2017.

[13] A. Paszke, S. Gross et al., "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in Advances in Neural Information Processing Systems, Vancouver, Canada, 2019.

[14] M. Abadi et al., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," 2015. Available online at: http://tensorflow.org/.

[15] "Keras: The Python Deep Learning API." Available online at: https://keras.io/. [Accessed 2021].

[16] "QKeras: A Quantization Deep Learning Library for Tensorflow Keras," Available online at: https://github.com/google/qkeras. [Accessed 2021].

[17] "Larq: Python Library for Training BNN," 2021. Available online at: https://larq.dev/.

[18] "NVidia TensorRT: Programmable inference accelerator." Available online at: https://developer.nvidia.com/tensorrt.

[19] O. Bichler, D. Briand et al., "N2D2-neural network design & deployment," CEA LIST, 2017. Available online at: https://github.com/CEA-LIST/N2D2/raw/master/manual/manual.pdf.

[20] J. Bai, F. Lu et al., "ONNX: Open Neural Network Exchange." Available online at: https://github.com/onnx/onnx. [Accessed 2021].

[21] I. Hubara, M. Courbariaux et al., "Quantized Neural Networks: Training Neural Networks with Low Precision Weights and Activation," J. Mach. Learn. Res., vol. 18, no. 1, p. 6869–6898, 2017.

[22] M. Rastegari, V. Ordonez et al., "XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks," in European Conference in Computer Vision, 2016.

[23] L. Fengfu and L. Bin, "Ternary Weight Networks," 2016. Available online at: https://arxiv.org/abs/1605.04711.

[24] H. Alemdar, V. Leroy et al., "Ternary Neural Networks for Resource-Efficient AI Applications," in 30th International Joint Conference on Neural Network, Training code available at https://github.com/slide-lig/tnn-train, 2017.

[25] S. K. Esser, J. L. McKinstry et al., "Learned Step Size Quantization," 2019. Available online at: http://arxiv.org/abs/1902.08153.

[26] J. Qing, Y. Linjie and L. Zhenyu, "Towards Efficient Training for Neural Network Quantization," 2019. Available online at: https://arxiv.org/abs/1912.10207.

[27] J. Qing, Y. Linjie and L. Zhenyu, "Rethinking Neural Network Quantization," 2020. Available online at: https://openreview.net/forum?id=HygQ7TNtPr.

2.3

Towards Fully Automated Verification of Semiconductor Technologies

Matthias Ludwig[1], **Dinu Purice**[2], **Bernhard Lippmann**[1], **Ann-Christin Bette**[1] **and Claus Lenz**[2]

[1]Infineon Technologies AG, Munich, Germany
[2]Cognition Factory GmbH, Munich, Germany

Abstract

In an ever more connected world, semiconductor devices are at the heart of every technically sophisticated system. Safety and security in operation, on which many times vital personal or business data or our lives depend on, is critical. The market for semiconductors is tremendous, and rogues also to get their share by selling counterfeit products which potentially jeopardize that very safety and security. Trust into semiconductor devices can be created by securing the supply chain or by verifying the electrical characteristics, the physical layout and the manufacturing technology against the design and specifications. The objective of this work is to propose a verification pipeline for semiconductor devices utilizing their technological features computed by the means of an automated device cross-section analysis. The emphasis lies on the confluence of an established industrial analytic process with novel possibilities provided by the advances in data processing and machine learning. This framework, its technical implementations, and exemplary results of our proposed autonomous technology analytics approach are presented in this work. Furthermore, the results are compared against a manual expert's measurement which underline the high performance of the framework and its effective multi-stage realisation.

Keywords: industrial artificial intelligence, failure analysis, anti-counterfeiting, hardware trust, verification and validation, semiconductor technology analysis, image processing, convolutional neural network, semantic segmentation, pattern recognition, supervised learning, deep learning.

2.3.1 Introduction

Trust into microelectronics [2], [3] can be generated by the validation and verification [12] of its originality. With today's world-wide distributed supply chains of microelectronics manufacturing, validating the safety, security, and trustworthiness of these devices is a highly complex task. Still, it is of paramount importance: electronics span every aspect of our daily lives and range from applications such as the (industrial) internet of things, over consumer electronics, to connected vehicles.

A way to check a product's originality is through physical inspection techniques, such as cross-sectioning. Through a sub-sequent analysis of the cross-sections, the integrity of the manufacturing technology [11] can be verified. To achieve this, all technological properties can be used in a verification process. In the case of cross sections, these are geometric shapes and dimensions, or material-related properties. Each technology can be interpreted as a unique fingerprint, so that deviations from specifications can be reported as suspicious. Nonetheless, physical inspection techniques must keep up with the continuously growing complexity of advanced semiconductor manufacturing nodes, and automation is another requirement in demand.

Cross-section (CS) images from scanning electron microscopes (SEM) are acquired at the failure analysis or process control labs and are a standard analysis process in the semiconductor industry. By the usage of SEM-integrated software tools, the technological features are manually measured and evaluated by engineers. Due to the expenditure of human labour, this process is costly and domain knowledge is required to fully interpret a sample or to detect anomalies in a set of images. The data is already available today, with datasets being produced at the sites. The utilization of data intensive analysis methods opens the possibility to create additional value by saving analysis expenses - and in the end overall cost - with an automated interpretation and measurement approach.

Figure 2.3.1 shows the second important aspect of the inspection flow: the full abstraction stack – ranging from software applications down to the

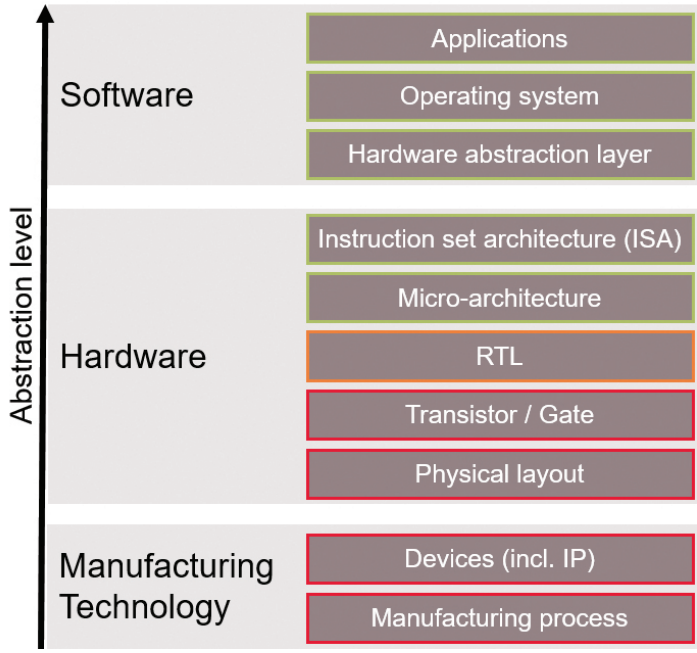


Figure 2.3.1 Abstraction layers in typical computing systems, ranging from software, over hardware design, to the underlying manufacturing technology.

physical implementation – of complex computing systems is illustrated. More and more software layers and manufacturing technology-agnostic layers can be investigated through published methods for verification of security [4] and functional safety (IEC 61508). Yet, the lower layers remain proprietary with no way to verify the integrity of their design. There have been several publications addressing the integrity checking of physical properties of semiconductor packages [5], [6], [7], and supply chain security related approaches [8]. Summaries about the detection and avoidance schemes of counterfeit electronics can be found in [1], [9], [10]. This work aims to push the boundaries of the state-of-the-art of automated physical inspection by the enablement of an automated detection of suspicious devices through SEM cross-section analysis.

In this work, academic and industrially relevant topics are discussed: First, the technology related characteristic – providing methods to secure the integrity of integrated devices. And second, the implementation of an industrial automation use-case – integrated into a complex established

environment – which can be seen paradigmatic for the challenges and possibilities of the entire project.

2.3.2 Background: Interpreting Semiconductor Technologies

The tremendous manufacturing improvements of past years and decades for semiconductor devices are shown in Figure 2.3.2. The CS images show four different technology *stacks*, from 150nm (introduced in the early 2000s) down to a more advanced 28nm (introduced in the early 2010s) process node. On these equally scaled CS images it is shown that the size of critical dimensions (CDs) has been continuously shrunk. On the other hand, the total number of processing steps and subsequently the number of visible objects is increasing.

The *stacks* visible in the images can be interpreted as a unique fingerprint for each manufacturing technology and its measured properties allow an inference to the specified and designed technological features. Specifically from these images, the thickness for each deposited layer and the minimum dimensions of each lithographic pattern found for each layer can be extracted. The set of identified technological parameters then enables the identification of production technologies. The innovative novelty of our approach can be explained via Figure 2.3.3: In the current reverse engineering process, the input is a known or unknown device, with the target to analyse its physical properties (geometrical and material-related) and consequently its manufacturing process.

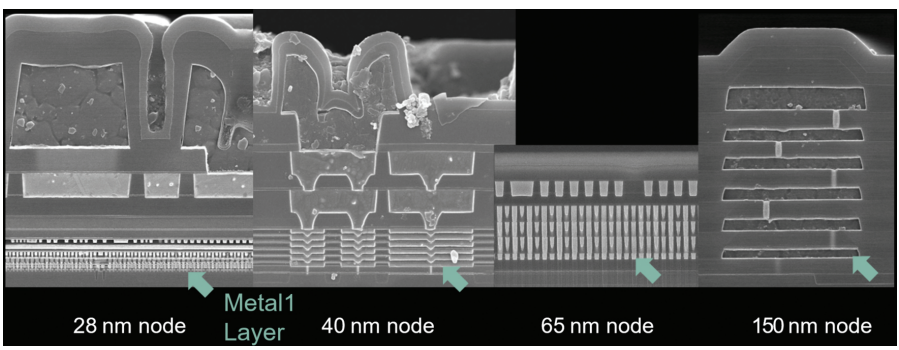


Figure 2.3.2 Equally scaled scanning electron microscope images of semiconductor device cross-sections, showing a 28nm, a 40nm, a 65nm, and 150nm process node (from [12]).

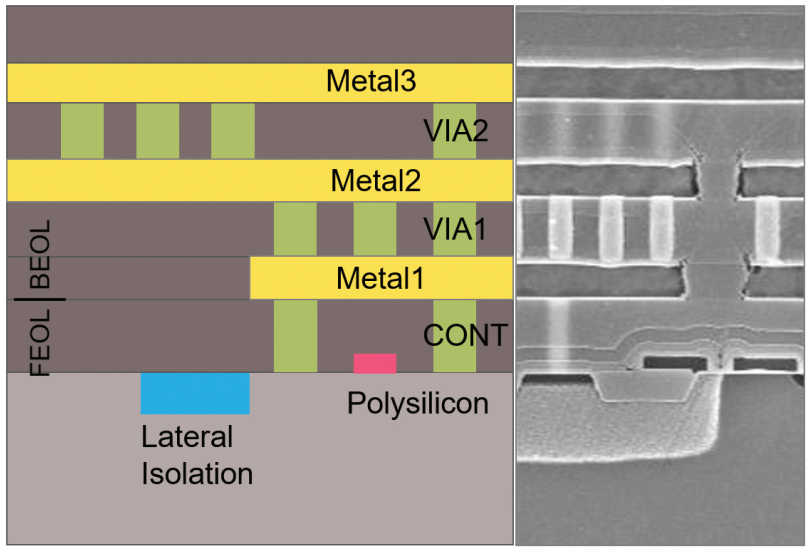


Figure 2.3.3 Example of a cross-section image which shows already interpreted objects on the left side and part of the raw image on the right side.

The application of the aforementioned principles for the purpose of counterfeit identification is even more challenging when vast numbers of features must be correlated and interpreted against known technology definitions. An automated processing of this data has been enabled by the advances in image processing and automated feature extraction.

The integration of technology domain knowledge and AI methods into a well-controlled industrial process (see Figure 2.3.4) is a fundamental prerequisite of the project. Considering the challenges of a supervised deep learning approach to interpret the SEM images, contributions from both fields were needed to produce the labelled dataset. Yet due to the high complexity of the task and the non-availability of methods to analyse the complex data structures, it was not possible to provide a fully automated approach. This missing link between AI methods and domain knowledge and the use cases is worked out by the proposed approach of SEM image interpretation and presented in this work. The second stage of overlap between the application and AI fields then comes into play during the segmentation result interpretation process. During this process, the segmentation accuracy does not linearly translate into overall technology prediction accuracy. This is explained by the fact that certain features identified by the deep learning (DL)

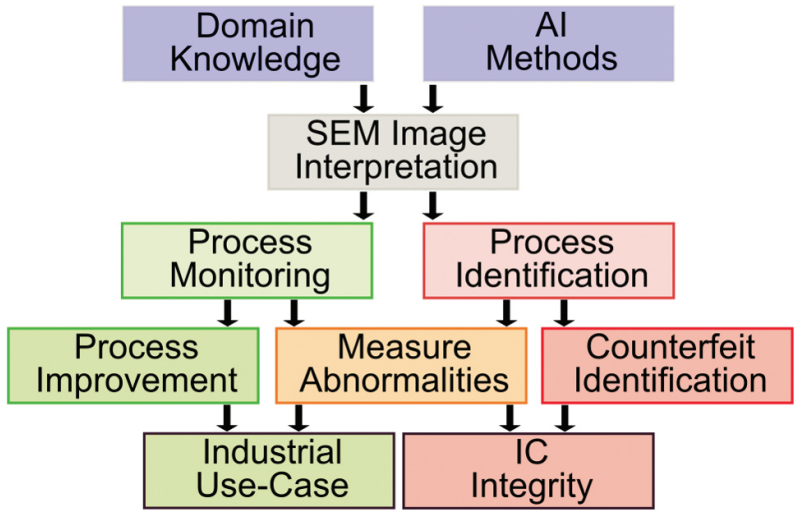


Figure 2.3.4 Overall framework of the project. Domain knowledge and AI methods were the enabler for the use-cases that are facilitated through the automated SEM image interpretation.

methodology have a larger impact on subsequent calculation than others. Consequently, a looped, iterative development approach was followed to ensure the AI component of the overall process is trained adequately. An emphasis is put on the most relevant and critical features, instead of the more common approach of maximizing a pre-defined accuracy metric.

2.3.2.1 Methodology: The Integrated Analysis Process

A conspectus of the whole analysis process is shown in Table 2.3.1, where the established laboratory process is extended via two software (data processing) steps. The entire process is outlined in detail in this chapter.

Table 2.3.1 Framework of the cross-section interpretation with the respective sub-processes.

	Sub-Process	Sub-Steps	Intermediate Results
Process Flow ↓	Lab work	Established analysis process: • Physical sample preparation • SEM image acquisition	Grey-scaled images
	Feature extraction	• Image segmentation • Object vectorisation	Vectorised images, objects per class
	Feature processing	• Feature measurement • Technology determination	Technology features, technology platform

Sample preparation and image acquisition: Even though cross-sectioning is considered a standard process, mastering the physical process can take several years. Two main methods for cross-sectioning exist: The first is a deposition of the sample in epoxy and subsequently an abrasive grinding of it. Moreover, the cross-sectioning can be performed on a glass grinding wheel, after devices package has been detached. The last step in the laboratory is the image acquisition via SEMs [13].

Image Processing: The goal of the image processing step is to provide fast, reliable, and accurate segmentations based on SEM images. The images are grey-valued with ambiguous intensity values for different object classes, as indicated in Figure 2.3.5. Furthermore, the task difficulty is boosted by the various zoom levels and variability of the regions of interest sizes.

The overlaps between different classes represented in Figure 2.3.5. A challenge the use of classical computer vision segmentation techniques such as thresholding, region-growing, or histogram-based methods. Nevertheless, these approaches are useful to supplement the segmentation pipeline in pre- and post-processing steps. The nature of the SEM images bears high similarity to medical images, particularly computed tomography and magnetic resonance images, where AI based techniques are becoming increasingly investigated to solve segmentation challenges. Therefore, a set of experiments aimed at comparing various DL state-of-the-art fully convolutional methods were conducted, comparing architectures such as U-net [15], PSPNet [18], FPN [19], GSCNN [20], Siamese-based [23]. It is concluded that overly complex architectures overfit specific tasks and often underperform on high-variance data, and while being commonly used as a benchmark, the U-net basis for the CNN architecture can outperform

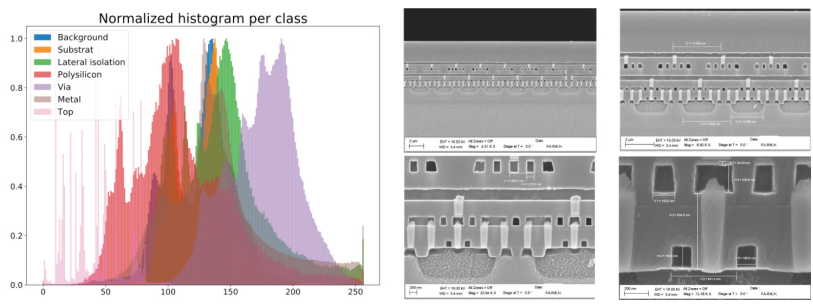


Figure 2.3.5 A. Normalized histogram per class. B. Various zoom levels of the same image, magnified 4310, 8650, 20940 and 72180 times, respectively.

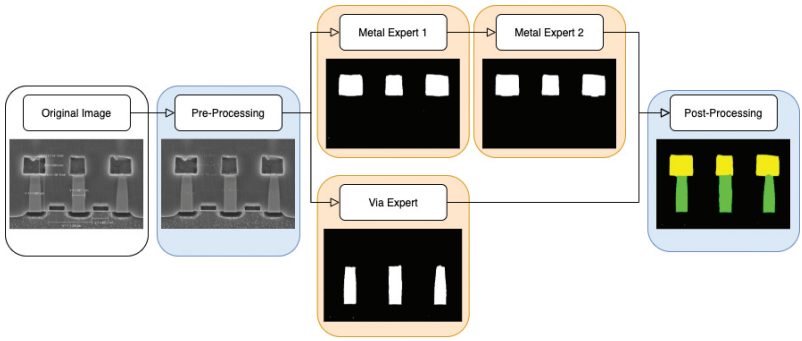


Figure 2.3.6 Exemplified overview of the segmentation pipeline.

other architectures assuming proper pre- and post-processing techniques [14]. Subsequently, a cascade U-net based architecture is concluded to be most suitable for the task at hand.

A dataset of around 500 images was created and labelled in pixel-wise accuracy, and dedicated networks were trained for metal and VIA (vertical interconnection access) segmentation (further called "experts"). First level experts segment the down-sampled image, and pass the resulting segmentation (one-hot encoded) to the second level expert along with the input image, who produces a more accurate output, much less vulnerable to outliers. Due to the varied nature of the labels of interest it was concluded that such a cascaded approach is beneficial for metal segmentation, while providing negligible improvements for VIAs, which were subsequentially segmented by a single "expert".

The issue of the relatively small dataset was tackled using image augmentation including horizontal flips and small rotations. Segmentation problems involving high intra- and inter-class imbalance (as is the case in question) have shown to be solved most successfully using Dice-based loss functions [16]. Therefore, several candidates were investigated as hyper-parameter options, with metal segmentation benefitting most from LogCoshDSC Loss [22], and VIA segmentation from Focal Tversky Loss [21], respectively. The high number of hyperparameters were tuned using a population-based approach. The evolutionary nature of the approach ensured high confidence in the obtained parameters and better final performance while keeping computational time requirements within reasonable limits [17]. The obtained results yield a 24 % increase in accuracy compared to the baseline version, and obtained an overall Dice score of 0.90. Examples of resulting segmentations are presented below in Figure 2.3.7.

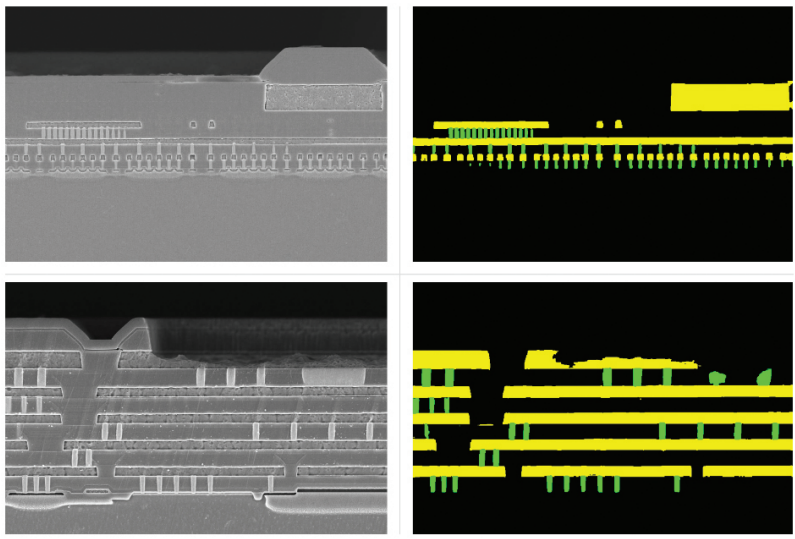


Figure 2.3.7 Examples of segmented images with yellow illustrating metal components, and green illustrating VIAs.

Image Measurement: The segmented images are calibrated via SEM meta-data or pattern matching of the dimensional bars and then vectorised into polygons of the different classes (e.g.: metal, VIAs, etc.). Polygons enable the utilization of their inherent attributes like the centroid, the circumference, or the area. An innovative – completely unsupervised - usage of these attributes is used for pattern recognition purposes. Established clustering methods [24] are linked with the properties of manufactured semiconductor devices. From these clusters the geometrical features are determined.

Technology Determination: The target is to evaluate the correct technology platform via the computed process feature vector. This vector will have dozens of measured attributes which are correlated against the known technology definitions (see example in Figure 2.3.8). In our implementation, distance metrics (Euclidean, rectilinear distance) between measured and defined values have been shown to yield good prediction results. A further improvement will be gained through assessment of individual feature importance by variable selection techniques.

In the example in Figure 2.3.8, three random features – metal 1 thickness, contact height, contact minimum pitch, and the total number of metal layers (colour coded) – are plotted for several dozen possible technology

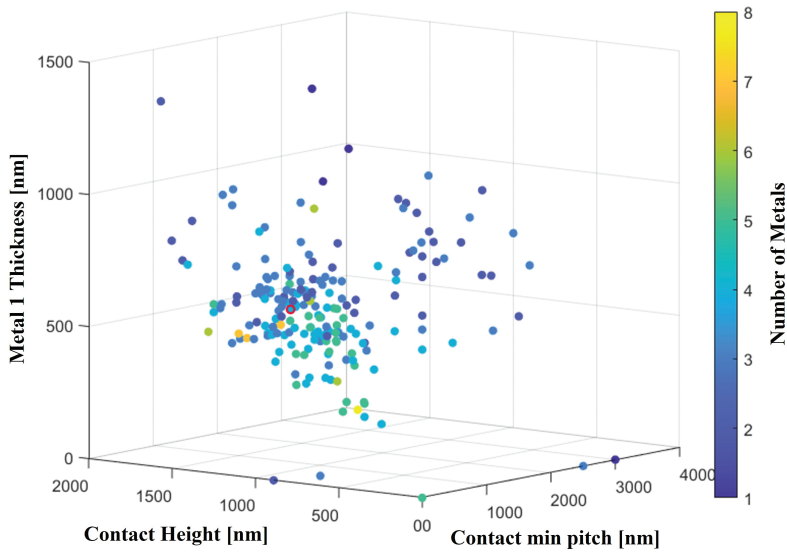


Figure 2.3.8 Example features of different semiconductor technologies. The four dimensions were arbitrarily chosen from more than hundred possible attributes defining a semiconductor front-end technology.

specifications. These characteristics are also of importance for a correct determination. The red mark shows an example measurement and the closest distance to adjacent data-points yields the most likely technology match. These three dimensions are extended to a higher dimensional space in the application.

2.3.2.2 Example Analysis: From the Image to the Feature Extraction

To conclude our work, the technological attributes of the VIAs of a sample are extracted. The VIAs are shown in the grey-scale image of Figure 2.3.9 and indicated through red boxes. After their semantic segmentation, the VIA objects appear in green and the metal lines in yellow. A visual inspection shows that all VIAs have been neatly extracted. The same applies to the metal, except for the top metal which shows a minor tear in the middle section. The measurement of the geometrical features (pitch and height) is shown in Table 2.3.2 and the automated measurement is compared against the manual measurement of an expert operator. The deviation on the right column shows the feasibility of an autonomous analysis which can also be done with

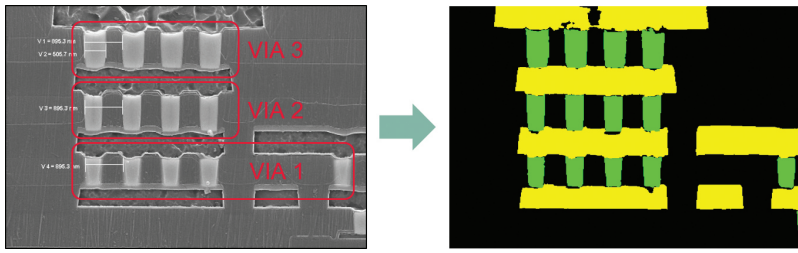


Figure 2.3.9 Example SEM CS with the grey-scaled SEM image (left) and the segmented image (right).

Table 2.3.2 Results of measured features of the VIAs. In the right column, the deviation between the automated and the manual is shown.

Measurement	Auto	Manual	Dev. [%]
VIA 1 Pitch [nm]	917	895	2.4
VIA 1 Height [nm]	675	711	5.1
VIA 2 Pitch [nm]	912	895	1.9
VIA 2 Height [nm]	700	742	5.7
VIA 3 Pitch [nm]	910	895	1.7
VIA 3 Height [nm]	779	806	3.3

other measurable features. Due to the high accuracy of the measurement, the technology platform determination for this example was successful.

2.3.3 Conclusion

The possibility of applying state-of-the-art AI approaches has enabled us to extend the existing workflow by an automated technology analysis. It has been shown that an extraction of technological attributes from SEM CS images in a fully autonomous manner is possible, with results comparable to an operator's manual effort. The most challenging part was the confluence of the knowledge of both domain experts and AI/ML experts.

The presented framework allows an automated check of the inferred technological parameters for verification and validation against specifications. Additionally, emphasis is put on a modular design of the sub-tools. This allows a migration to other applications and an extension of the presented status with other classes for segmentation is not overly complex. In summary, this contribution is a steps towards improved physical inspection for hardware assurance. A future task will be the application of the framework on real world examples.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] U. Guin, K. Huang, D. DiMase, J. M. Carulli, M. Tehranipoor, and Y. Makris, "Counterfeit integrated circuits: A rising threat in the global semiconductor supply chain," Proceedings of the IEEE, vol. 102, no. 8, pp. 1207–1228, 2014.

[2] T. Hoque, P. SLPSK, and S. Bhunia, "Trust issues in microelectronics: The concerns and the countermeasures," IEEE Consumer Electronics Magazine, vol. 9, no. 6, pp. 72–83, 2020.

[3] B. Liu, Y. Jin, and G. Qu, "Hardware design and verification techniques for supply chain risk mitigation," in 2015 14th International Conference on Computer-Aided Design and Computer Graphics (CAD/Graphics), 2015, pp. 238–239.

[4] O. Demir, W. Xiong, F. Zaghloul, and J. Szefer, "Survey of approaches for security verification of hardware/software systems." IACR Cryptology ePrint Archive, vol. 2016, p. 846, 2016. http://dblp.uni-trier.de/db/journals/iacr/iacr2016.html

[5] P. Ghosh and R. S. Chakraborty, "Recycled and remarked counterfeit integrated circuit detection by image-processing-based package texture and indent analysis," IEEE Transactions on Industrial Informatics, vol. 15, no. 4, pp. 1966–1974, 2019.

[6] R. Hammond. Counterfeit electronic component detection. ERAI, Inc. [Online]. Available: https://www.aeri.com/counterfeit-electronic-component-detection/

[7] A. Kanovsky, P. Spanik, and M. Frivaldsky, "Detection of electronic counterfeit components," in 2015 16th International Scientific Conference on Electric Power Engineering (EPE), 2015, pp. 701–705.

[8] C. E. Shearon, "A practical way to limit counterfeits," in 2019 Pan Pacific Microelectronics Symposium (Pan Pacific), 2019, pp. 1–7.

[9] G. Caswell, "Counterfeit detection strategies: When to do it / how to do it," International Symposium on Microelectronics: FALL 2010, vol. Vol. 2010, no. No. 1, pp. 227–233, 2010.

[10] M. M. Tehranipoor, U. Guin, and D. Forte, Counterfeit Integrated Circuits: Detection and Avoidance. Springer Publishing Company, Incorporated, 2015.

[11] Y. Nishi and R. Doering, Handbook of Semiconductor Manufacturing Technology. CRC Press, 2017. https://books.google.de/books?id=PsVVKzhjBgC

[12] B. Lippmann, N. Unverricht, A. Singla, M. Ludwig, M. Werner, P. Egger, A. Duebotzky, H. Graeb, H. Gieser, M. Rasche, and O. Kellermann, "Verification of physical designs using an integrated reverse engineering flow for nanoscale technologies," Integration, vol. 71, pp. 11 – 29, 2020. http://www.sciencedirect.com/science/article/pii/S0167926019302998

[13] M. Vogel, Handbook of Charged Particle Optics, 2nd ed., J. Orloff, Ed. Contemporary Physics, 2010, vol. 51, no. 4.

[14] Isensee, F., Petersen, J., Klein, A., Zimmerer, D., Jaeger, P. F., Kohl, S., Wasserthal, J., Koehler, G., Norajitra, T., Wirkert, S., & Maier-Hein, K. H. (2018). nnU-Net: Self-adapting Framework for U-Net-Based Medical Image Segmentation. Informatik Aktuell, 22. http://arxiv.org/abs/1809.10486

[15] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (Vol. 9351, pp. 234–241). https://doi.org/10.1007/978-3-319-24574-4_28

[16] Milletari, F., Navab, N., & Ahmadi, S.-A. (2016). V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation. 2016 Fourth International Conference on 3D Vision (3DV), 565–571. https://doi.org/10.1109/3DV.2016.79

[17] Jaderberg, M., Dalibard, V., Osindero, S., Czarnecki, W. M., Donahue, J., Razavi, A., Vinyals, O., Green, T., Dunning, I., Simonyan, K., Fernando, C., & Kavukcuoglu, K. (2017). Population Based Training of Neural Networks. http://arxiv.org/abs/1711.09846

[18] Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid scene parsing network. Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017-Janua, 6230–6239. https://doi.org/10.1109/CVPR.2017.660

[19] Li, X., Lai, T., Wang, S., Chen, Q., Yang, C., & Chen, R. (2019). Weighted feature pyramid networks for object detection. Proceedings - 2019 IEEE Intl Conf on Parallel and Distributed Processing with Applications, Big Data and Cloud Computing, Sustainable Computing and Communications, Social Computing and Networking, ISPA/BDCloud/SustainCom/SocialCom 2019, 1500–1504. https://doi.org/10.1109/ISPA-BDCloud-SustainCom-SocialCom48970.2019.00217

[20] Takikawa, T., Acuna, D., Jampani, V., & Fidler, S. (2019). Gated-SCNN: Gated shape CNNs for semantic segmentation. In Proceedings of the IEEE International Conference on Computer Vision (Vols. 2019-October). https://doi.org/10.1109/ICCV.2019.00533

[21] Abraham, N., & Khan, N. M. (2019). A novel focal tversky loss function with improved attention u-net for lesion segmentation. Proceedings - International Symposium on Biomedical Imaging, 2019-April, 683–687. https://doi.org/10.1109/ISBI.2019.8759329

[22] Yeung, M., Sala, E., Schönlieb, C.-B., & Rundo, L. (2021). A Mixed Focal Loss Function for Handling Class Imbalanced Medical Image Segmentation. http://arxiv.org/abs/2102.04525

[23] Martin, K., Windunga, N., Sani, S., Massie, S., & Clos, J. (2017). A convolutional siamese network for developing similarity knowledge in the SelfBACK dataset. CEUR Workshop Proceedings. https://rgu-repository.worktribe.com/output/246838/a-convolutional-siamese-network-for-developing-similarity-knowledge-in-the-selfback-dataset

[24] S. Wierzchon and M. Kłopotek, "Modern algorithms of cluster analysis," Springer International Publishing, vol. 34, 01 2018.

2.4

Automated Anomaly Detection Through Assembly and Packaging Process

Saad Al-Baddai[1], Martin Juhrisch[2], Jan Papadoudis[1], Anna Renner[2], Lippmann Bernhard[1], Cristina De Luca[1], Fabian Haas[1] and Wolfgang Schober[1]

[1]Infineon Technologies AG, Germany
[2]Symate GmbH, Germany

Abstract

In the semiconductor industry the desired quality and effectiveness in the process of assembling integrated circuits is nowadays at the limit and without safety margin. To achieve important competitive advantages, this process must be continuously optimized and adjusted. Such process is indeed strongly dependent on parameters that are distributed among various control technology assemblies, materials, and the environment. However, the current inspection tools deployed for defect detection through assembly and packaging process are mainly based on rigid and simple rules. The latter are handcrafted by engineers, which can only extract shallow features. Therefore, the accuracy of classification by tools is quite low, which provides incomplete information for root cause investigation and can cause yield-loss costs due to over reject. Hence, automatic inspection tools for visual defect detection, acting as final quality gate before shipping to end customers is very demanding. Therefore, a deviation detection model based on machine learning is developed. On the other side, due to the lack of existing labelled images, an anomaly detection is proposed, in some cases as an assistant tool for collecting defect images with less effort. Results show that artificial intelligent (AI) solutions can achieve a better performance than the classical tools and overcome the human ability in detecting the deviation in the data.

Hence, AI can be used for decreasing the yield-loss, improving quality of the product and greatly reduce labour intensity.

Keywords: artificial intelligence, semiconductor industry, image classification, wirebonding, deep learning, anomaly detection.

2.4.1 Introduction and Background

Semiconductor manufacturing has an increasing complexity and demand on quality requirements, as electronics increasingly become an important part of modern society. In principle, semiconductor manufacturing is equipped with lots of sensors to monitor the processes, but it lacks a suitable way to make use of this data. Thus, new methods are needed to support quality and engineering personal at finding deviations during production to avoid costly production losses or even worse, complaints by customers. Machine learning based anomaly detection (AD) can be a powerful tool to indicate single outliers, but also systematic changes in processes and / or materials. In a next step those deviations can be analysed to label the data indicating a root cause for the different types of deviations. Therefore, one of the success factors in optimizing the industrial processes is either automatic anomaly detection, supervised learning or both, which leads to prevention of production flaws, improving the quality, increasing yields and making more benefits.

The most popular way of performing anomaly detection in many industrial applications is by adjusting digital camera parameters or sensors during the collection of either images or time series data. This is basically an image or signal anomaly detection problem that is searching for patterns that are different from normal data later on at test phase [9]. By assumption, humans can easily manage such tasks by recognizing normal patterns, but this is not as easy for machines. Unlike other classical approaches, image anomaly detection faces some of the following difficult challenges: class imbalance, quality of data, and unknown anomaly [9]. A prevalence of abnormal events is generally an exception, whereas normal events account for a significant proportion. Some techniques usually handle the anomaly detection problem as a "one-class" problem. Here models are learnt by using the normal data as truth ground and afterwards are evaluated whether the new data belongs to this ground truth or not, by the degree of similarity to the ground truth [18]. In the early applications of surface defect detection, the background is often

modeled by designing handmade features on defect-free data. For example, Bennatnoun et al. used blobs technique [5] to characterize the correct texture and to detect deviations through changes in the charter ships of generated blobs. While Amet et al. [1] used wavelet filters to extract different scales of defect-free images, then extracted the informative features of different frequency scales of images. However, most of these methods can work with homogeneous data of good quality and would require prior knowledge. But in most of real applications, this is not the case. Here, the deep learning approaches are used. One variant of common deep learning, which is used for anomaly detection, is the auto encoders (AEs), as they have unique reconstruction property.

The latter can map the input data non-linearly into a low-dimensional latent space and reconstruct it back into the data space. These models are then learned in an unsupervised fashion by minimizing input and output errors [3, 4, 12]. For time series data, the anomaly detection has a similar goal and issues alike:

- Difficulties connected to definition of normal regions, especially in regions close to boundaries.
- In many domains, normal behaviour develops gradually, and an ongoing position of normal pattern cannot guarantee its usage as sufficient proxy on another time step.
- Depending on application field, different parameter fluctuations are considered as normal, so there is no universal pattern or system, which does directly allow using techniques developed for one application to another.
- Absence of labelled data.
- Challenges connected to removing noise from data, which could be mistaken as anomalies [7].

Due to these above-mentioned challenges unsupervised anomaly detection on multi-dimensional data is a very important problem in machine learning and business applications [13].
In this article we will show two examples, how we make use of AD to

1. Detect deviations and
2. Generate further benefit by applying AD such as:
    a. Setup control
    b. Material control
    c. Labelling deviations for supervised learning

d. Compare different equipment regarding process stability and matching

The first example is based on sensor data from the wire bonding process and the second is based on images of the product. For both examples, different approaches were evaluated regarding accuracy and usability in production. First implementations showed that relevant outliers can be found, labelled, and used for subsequent supervised modelling. Additionally, the anomaly detection helped the production and engineers to find systematic influences and derive process improvements based on the new data insights from the anomaly detection. The defect would happen either in early processes or after the chip completed all the process including wafer fabrication, assembly and final test. Technically, the recorded data during sequence processes is collected in a time series fashion for some process or as images for others. Such data has fluctuations, noise, bad quality and high resolution. However, the defect is relatively small and hard to detect even manually. Unfortunately, the built-in software algorithm has a poor classification performance due to rigid and simple rules. So, the specification for inspection is very tight because no defective chips are allowed to ship to customers. As a result, a huge amount of good chips is scrapped, causing unnecessary yield loss cost. Moreover, there is another challenge for defect detection in productive environment if the production environment is dynamic, which means that the data quality is always strongly inconsistent. But also, to detect new defect types which have not been seen before is challenging but important for production.

In summary, the following section will describe the development of an IT infrastructure for anomaly detection in process chains. The aim is to develop an industrialised solution for the detection and visualization of anomalies in different process – using wire bonding and optical outgoing inspection (OOI) as examples. If necessary, with subsequent notification of the user about critical analysis results via e-mail/output signals. Basis of the development and visualization in anomaly detection is the work on wire bonding and OOI image data as well as further demo data.

2.4.2 Dataset Description and Defect Types

For wire bonding data, the data consists of a set of 369 experiments, each of which is described via 432 features (coming from 3 different sensors) during 143 timestamps. However, the features are highly repetitive

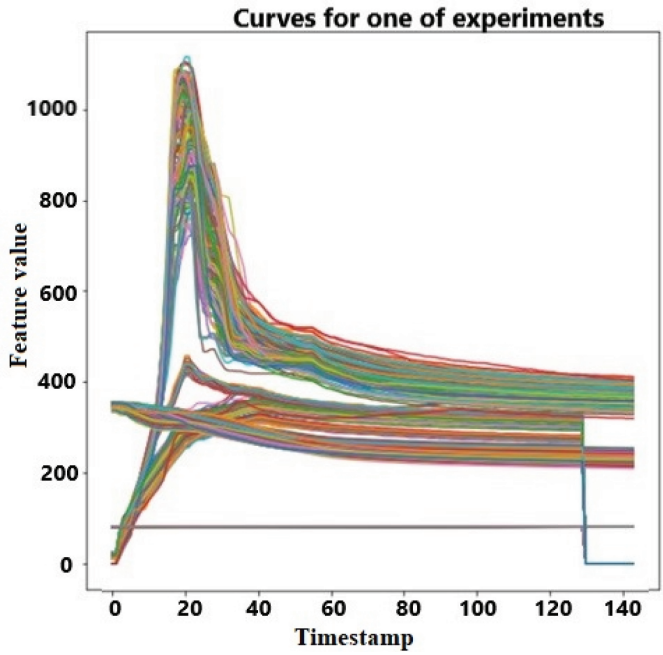


Figure 2.4.1 Curves for one experiment.

(see Figure 2.4.1). This is because there are multiple bond connections on one device, which share the same process parameters and behave quite similar. The three sensors are a current sensor, located at the transducer, a displacement sensor measuring the deformation of the wire and a frequency sensor, also located at the transducer of the wire bonder.

Changes in the raw data can have multiple reasons and are not necessarily known prior. However, most prominent are defects based on contamination of the device or a misadjusted machine, which can cause misaligned or deformed bonds. Some of the defects are shown in the following figures.

Already here enough deviations were found and labelled to enable a supervised training, which will be tested on new and historical data. Further developments were carried out based on Outliergram. It is also based on comparing the shapes of functions. Intuitively, the idea is to inspect how much time the curves spend above and between other curves from the dataset. The outliers are detected by inspecting the relationship between those two values for each of the curves. The results are presented in Figure 2.4.3.

The methods described require the pairwise comparison of all samples in the dataset. In some cases, this may be too expensive. If those methods produce meaningful results, they can be used to filter datasets before training an outliers-sensitive model, e.g., PCA or autoencoder on the rest of the dataset. Furthermore, the reconstruction error from those models could be used to detect outliers as it is less expensive to compute than the pairwise methods.

The second example is dealing with images which are basis for decision if a product has critical deviations and should be scrapped. As the availability of labelled images in a high yield manufacturing is low, AD can help to find critical devices. The further down presented procedure is in principle the same as for the wire bonding, however the used methodologies are more adapted to image data.

The last production step before packing is always the electrical test and a final optical outgoing inspection (OOI) to check that the product is free of visible defects. In the given use-case, a semiconductor power module needs to be inspected from two sides using two monochrome cameras and multiple light sources. The task of the inspection is to check the module at three areas: Leads, mold body and heatsink. Leads and mold body are very consistent in their optical appearance and the images can be checked using classical, rule-based algorithms. These are not considered in this use-case.

The biggest challenge of the optical inspection is the defect detection on the heatsink, see Figure 2.4.4, which consists of a rough copper surface. It needs to be inspected for scratches, metal, or mold particles as well as for mechanical damage like imprints. However, this surface shows a very high variety in appearance, as it is oxidized during preceding high temperature testing steps. Hence, the inspection cannot be carried out using rule-based algorithms, as the oxidized areas cannot be distinguished clearly from true defects by a rule-based algorithm. In this context, trained personnel took care of the heatsink inspection and was used to label the image data for supervised learning. The image data consists of four images per module and side, recorded with a different combination of light sources. Coaxial and diffuse lighting are used to highlight contaminations and particles on the heatsink whereas low-angle lateral lighting is used for detecting mechanical defects such as scratches or imprints in the surface, see Figure 2.4.4.

Also, for visualization purpose, two metrics are used: modified band depth (MBD) and modified epigraph index (MEI). The outliergram visualises

the relationship between these two metrics. The normal curves define a parabola in the two-dimensional space, see Figure 2.4.3. With some thresholds regarding quantiles, some outliers, which are too far away from the parabola (see Figure 2.4.2) can be identified.

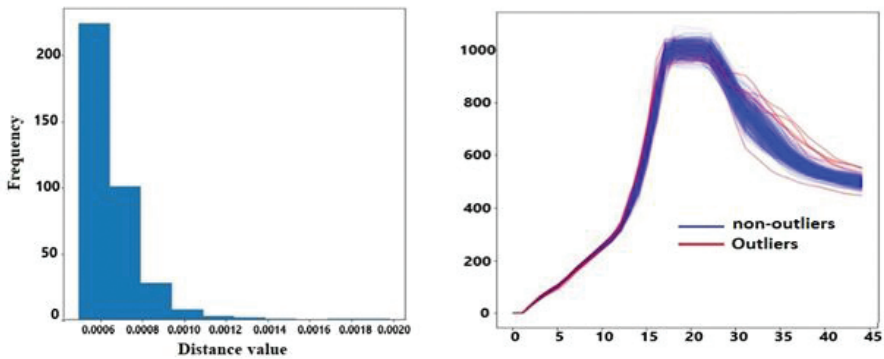


Figure 2.4.2 Left: Distribution of average curves distance to other samples. Right: the results are showed in left by using Wasserstein distance outliers.

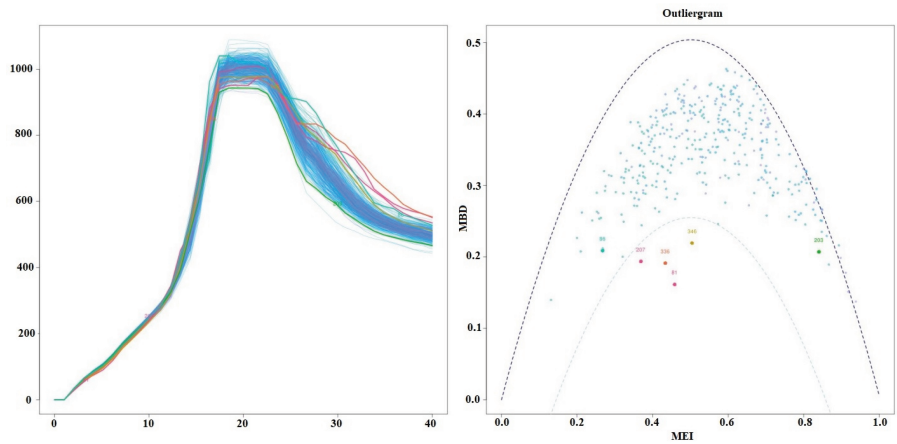


Figure 2.4.3 Outliergram, an example of feature for device current traces. Outliers are detected by inspecting the relationship between MEI and MBD.

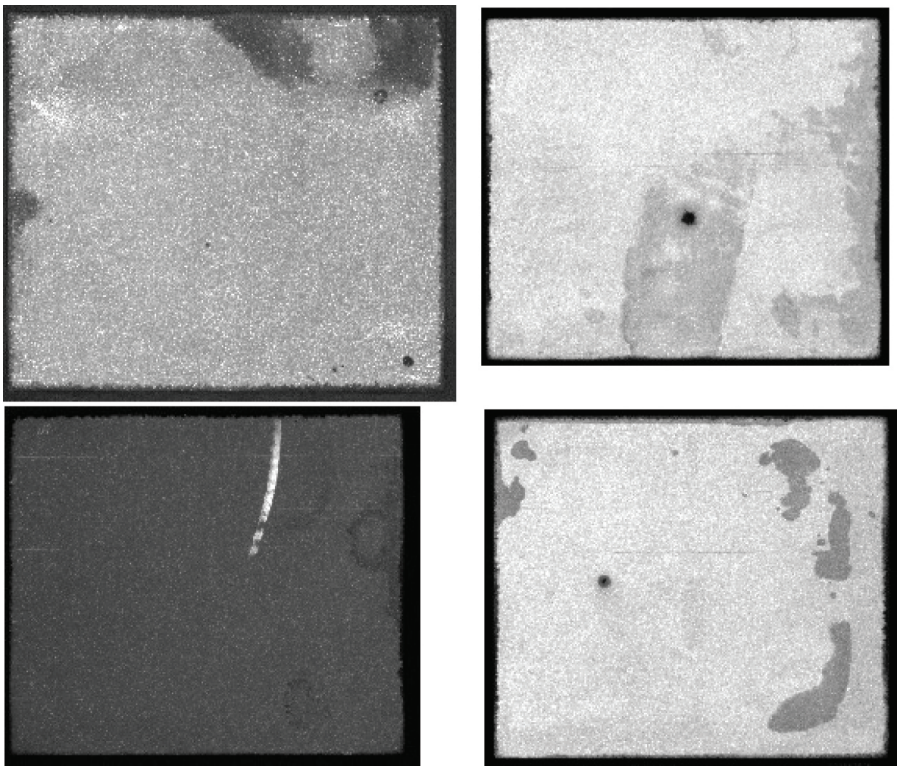


Figure 2.4.4 Shows samples of OOI use case. Top left: particle in lower right corner (bottom side). Top right: particle in centre of image (top side). Bottom left: particle in centre of image (top side). Bottom right: scratch in upper area of heatsink (top side). Note that bottom side is larger than top side.

2.4.3 Methodology

In this work, we used absolutely pure anomaly detection for the first use case and combined AD with supervised learning for the second use case. Hence, we apply the following scenarios:

- For wire bonding use case, Warstein distance outlier is applied.
- For optical outgoing inspection (OOI), two approaches are considered:
    a. Anomaly detection, using pre-trained DL algorithms, was used first in order to reduce effort of labelling data.
    b. Afterwards, the labelled data were used for training a convolutional neural network (CNN).

2.4.3.1 Anomaly Detection

Anomalies are defined as events that deviate from the standard, rarely happen, and don't follow the rest of the "pattern", see Figure 2.4.5. In general, anomaly detection algorithms (ADA) can be classified into two types:

- **Outlier detection**: In this case the dataset consists of both good and abnormal units. Here ADA tries to find the optimal region boundaries of the training data, where the good units are most concentrated and therefore isolating the abnormal units. Such algorithms are often trained using unsupervised learning [6] (i.e., without labels). This type of detection can detect global outliers [2], contextual outliers [8, 10], or collective outliers [8]. However, sometimes, such methods could be used as a pre-process for datasets before applying additional machine learning techniques [11].
- **Novelty detection**: Unlike outlier detection, which includes examples of both normal and abnormal units, novelty detection algorithms have only the normal units (i.e., no anomaly events) during training phase. These algorithms are trained with only labelled examples of good units (semi-supervised learning). At inference phase, novelty detection algorithms must detect when an input data point is far (deviate) off to the good ones.

Generally speaking, outlier detection and novelty detection is a form of unsupervised learning. In this study we introduce a new version of anomaly detection called pseudo anomaly detection (PAD). The latter is indeed a supervised learning algorithm, which can be employed to do unsupervised learning (anomaly detection).

2.4.3.2 Pseudo Anomaly Detection

Following the definition of AD, the idea behind PAD is to simply follow the same definition by using an existing pre-trained algorithm like Alex [16], Resnet [17], GoogleNet [18] etc. Those pretrained algorithms are already trained on a benchmark called the ImageNet dataset [14]. The latter has labels of up to 1000 classes. To cluster the unlabelled data into different categories, under the assumption that prevalence of the defects is very low with respect to the whole population, the expected outputs is to map the good images (majority) to a specific category (one or subset of 1000 classes), within they should have some similarity. On the other hand, the scrapped images (minority) would be distributed over other categories. Such scrapped images

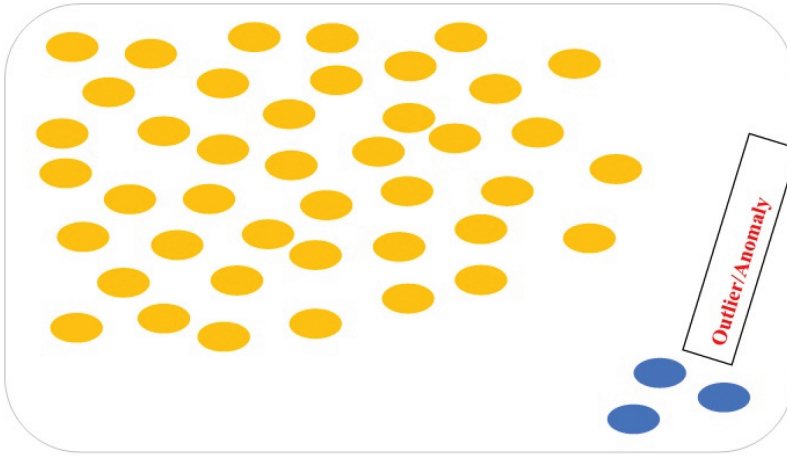


Figure 2.4.5 Show an example of outliers (anomaly) cluster which is clearly inconsistent with the rest of the dataset.

will show up but will happen with an incredibly small probability. Here, these images are reviewed by an expert. In this way the effort for labelling images was reduced by roughly 85%. Please note that names of classes as you can see in Figure 2.4.4 and Figure 2.4.5, represent the original names of the classes, which was used during training of such algorithms as supervised learning (names of real objects). However, in this work, we employ such algorithm as unsupervised algorithms for our data if they don't belong to any of these classes. As, a result, we suppose most good units have similar patterns and would map to a one or few real classes. However, from a machine learning perspective, this makes detecting anomalies hard — by definition, in case we have massive amounts of good images and few bad images of "anomaly" units, but which have a uniform distribution in our dataset. How are anomaly detection algorithms, which tend to work optimally with balanced datasets, supposed to work when the anomalies we want to detect might only 0.2% based on prevalence assumption? Luckily, in our case PAD could figure out the similarity within good images and map them to only a few categories. This is very helpful to reduce the effort for labelling defect images, see Figure 2.4.5.

For wirebonding, a method was developed for the detection of possible outliers. First attempts were done using dimensionality reduction techniques and tests of new approaches, which could smooth out possible anomalies and then, search for new approaches to analyse each feature separately.

Only the results for single feature are presented, however if the adopted approach provides meaningful results, it could be extended to the whole dataset. The planned methodological approach was to find the curves that had different shapes than the others. To compare the shapes of curves we utilised Wasserstein distance which estimates how much work should be done to transform one distribution into another. For each curve in the dataset, we computed the average of its distances to all the other curves.

Based on the histogram in Figure 2.4.2, a threshold value is selected (threshold = $1.2e^{-3}$) to detect the curves that differ much from the other.

On the other hand, the anomaly detection for the wire bonding process was integrated into the process monitoring system from IFX with an additional visualization to quickly see the status of the machine in the anomaly detection. The machines were sending the data via the SECS/GEM interface to a central IFX system which combines different data sources to a unified format and sends the data to the IFX APC-System. The anomaly detection can access this data and calculate the anomaly score. The result of the anomaly detection system is then also stored in the IFX APC-System. This is done by creating a file with the appropriate unified format containing the anomaly detection result and storing in on a network share, where the APC-System access the data and integrates it. In this stage the visualization process can be done by accessing the data independent from the anomaly detection calculation. The data flowchart for wire bonding case can be seen in Figure 2.4.9.

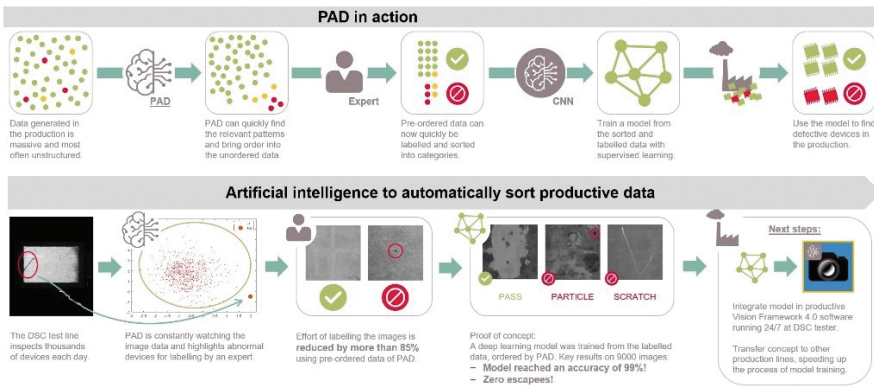


Figure 2.4.6 Shows the process flow for the whole process including PDA and supervised learning applied on optical images.

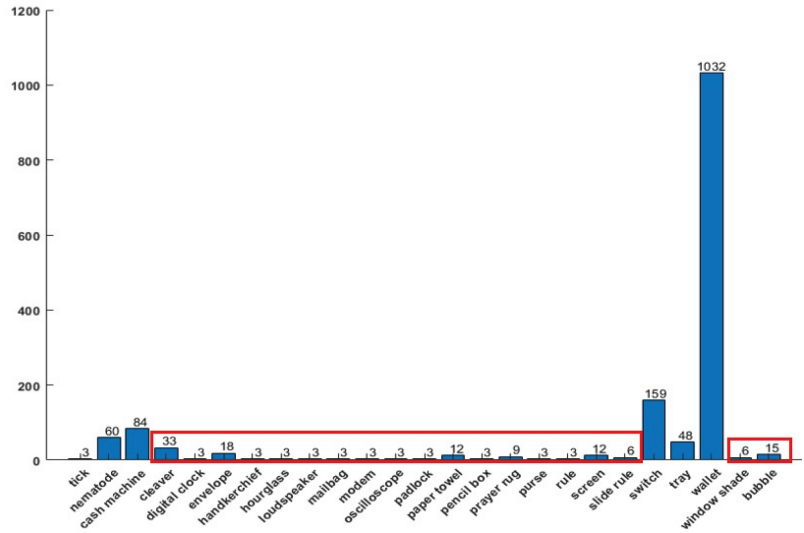


Figure 2.4.7 Anomalies exist at the marked area. In this study, anomaly detection with pre-trained algorithm Resnet was conducted.

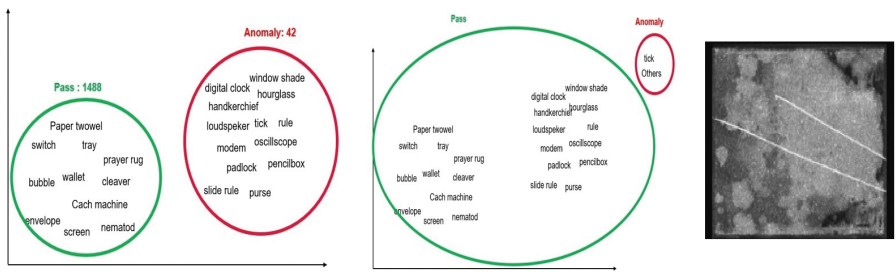


Figure 2.4.8 Shows an example of clustering anomalies units. Left: shows the clustering according to PAD. Middle: shows clustering after review process by an expert. Right: shows an example of defect units which recognized as a tick by PAD. As it shown, names represent the real names of classes of labelled images of ImageNet dataset.

2.4.3.3 Convolutional Neural Networks

Recently, deep neural networks (DNNs) have shown superior performance in a wide range of image processing tasks. We shortly summarize the most common variant of deep learning algorithms, which is called sequential convolutional neural network (SCNN): The primary purpose of the sequential convolution operation is to extract local features from the input image at various spatial scales. Convolution preserves the spatial relationship between

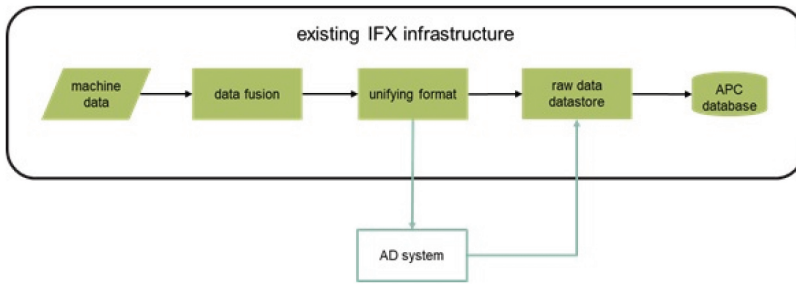


Figure 2.4.9 Shows the process flow of wire bonding use case.

pixels by learning image features using small patches of the input data. In CNN terminology, a 4×4 image patch, is called, for example, a captive field or filter kernel or feature detector, and the matrix formed by sliding the local filter over the whole image and computing the dot product of the filter weight with the input image intensity is called the convolved feature or activation map or the feature map. Each such feature map acts as input to the subsequent convolutional layer. It is important to note that filters act as feature detectors extracting various features from the original input image. As a result, the most relevant features are kept and less relevant features are suppressed. Let us suppose that an image $\mathbf{X}$ is defined by the following mapping:

$$\mathbf{X} : \{1, \ldots, M\} \times \{1, \ldots, N\} \to W \in \mathfrak{R}, (i, j) \to \mathbf{X}_{i,j} \tag{2.4.1}$$

Such an image $\mathbf{X}$ is represented by an array of size $M \times N$. Given a filter $\mathbf{F} \in \mathfrak{R}^{(2k_1+1)\times(2k_2+1)}$ the convolution of the image $\mathbf{X}$ with the filter kernel $\mathbf{F}$ is computed as:

$$(\mathbf{X}*\mathbf{F})_{r,s} := \sum_{u=-k_1}^{k_1} \sum_{v=-k_2}^{k_2} F_{u,v}\, X_{r+u,s+v} \tag{2.4.2}$$

Where the filter $\mathbf{F}$ is given by

$$\mathbf{F} = \begin{pmatrix} F_{-k_1,-k_2} & \cdots & F_{-k_1,k_2} \\ \vdots & F_{0,0} & \vdots \\ F_{k_1,-k_2} & \cdots & F_{k_1,k_2} \end{pmatrix} \tag{2.4.3}$$

However, in addition to convolution layers there are several common layers, which can be used with CNN such as rectified linear units (ReLU), pooling layers (either max or average) and fully connected layers. The latter is

corresponding to the traditional multi-layer perceptron network and is conventionally applied in the last stage of the CNN.

In this study for OOI use case, a CNN structured was created from scratch with 170 layers and 3 branches. A common set of hyperparameters as follows: number of epochs =3, initial learning rate (ILR) = 0.0001, mini-batch size = 64, and the stochastic gradient descent with momentum (SGDM) optimizer is employed.

2.4.4 Results and Discussion

For wire bonding use case, the anomaly detection system was running in parallel to production for several weeks. As it is difficult to validate the anomaly detection during production, since a difference in the raw data might result in a wide range of different impacts on the product, two different approaches to validate the system were made. The first one was to simply calculate the percentage of devices which showed an anomaly in the dataset and compare this to the process yield. If these percentages align, this is a good indicator that the anomaly detection represents the product quality. During our tests this was the case. As a second approach we gathered multiple devices which showed a high anomaly score and examined them thoroughly. In all of the cases different influences could be found on the device, like a contaminated device, reduced shear value or input material which was out of specifications. However, score indicating how different the raw data is from normal, an important aspect of the used anomaly detection was that the result is an anomaly and not a Boolean indication anomaly / no anomaly. Thus, it is necessary to find a threshold on which the difference in the raw data influences the quality of the product. It might be possible to find this threshold automatically if labelled data is available.

For OOI use case, PDA was running on roughly 12000 images. From this historical data PAD could reduce effort for labelling by more than 85%. This enabled an expert to go through only the rest of suspicion images and categories this portion to the real defects and real over-reject (good images). Roughly 130 images were recognized as defect images. Here, the same number of good images was used for training the CNN model to avoid imbalance issues during training process. Furthermore, relative few defect images were available during the training process, a strict regularization was considered to avoid the over-fitting issue by adding a dropout layer with 0.5 parameter. However, remaining of good images were used for test purposes. But first, we split the data into 80% for training and 20% for validation.

The accuracy was 99% for sensitivity as well as for specificity. That means only 1% should be historically reviewed but also periodically during run the model in productive data. Importantly, to follow zero defect philosophy, which means that only images without any defect are sent out to a customer. The threshold of the confidence level is set higher than 95% in order to report good images. On the other hand, this leads to an increase in the over-reject rate to roughly 2.5%. In this way, the model was tested on productive data with roughly 24000 images. An expert also manually reviewed the latter. The accuracy was robust with 98% and zero escapee. Overtime, more defect images are collected, and the model is updated to reduce the over-reject. Moreover, the model was transferred to run on the BOT side of the same product. Here, no available labelled images of this side are used for training. But there is sort of similarity between both sides. Only a bit of adaptation was done as a pre-processing on BOT images due to the difference in terms of reference points and resolution. The accuracy on BOT side was 97% as well.

2.4.5 Conclusion and Outlooks

In summary an AI solution consisting of a combination anomaly detection (unsupervised learning) and supervised learning are used for detecting deviations in semiconductor processes. In this work, it was demonstrated how AI can efficiently solve real-world problems in the industrial setting. The results are promising and would be a good alternative for classical approaches. As a results yields will be increased significantly, the quality will be improved, and the effort will be reduced as well. The next steps is to monitor, optimise and validate both solutions over time, but also integration of AI models into the productive environment. Additionally, the long-term goal is not only find the deviation but also to detect the exact type of defects like scratch, particle in case of images and to point out the root cause in case of wirebonding.

Acknowledgment

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union’s Horizon 2020 research

and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] A. Amet, A. Ertuzun, and A. Ercil. Texture defect detection using subband domain co-occurrence matrices. pages 205–210, 05 1998.

[2] F. Angiulli and F. Fassetti. Dolphin: An efficient algorithm for mining distance-based outliers in very large datasets. *TKDD*, 3, 01 2009.

[3] C. Baur, B. Wiestler, S. Albarqouni, and N. Navab. Deep autoencoding models for unsupervised anomaly segmentation in brain mr images. 04 2018.

[4] P. Bergmann, S. Löwe, M. Fauser, D. Sattlegger, and C. Steger. Improving unsupervised defect segmentation by applying structural similarity to autoencoders, pp. 372–380, 01 2019.

[5] A. Bodnarova, M. Bennamoun, and K. Kubik. Automatic visual inspection and flaw detection in textile materials: A review, pp. 194–197, 01 2001.

[6] A. Boukerche, L. Zheng, and O. Alfandi. Outlier detection: Methods, models, and classification. *ACM Computing Surveys*, 53:1–37, 06 2020.

[7] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

[8] D. D. and D. Sasidhar Babu. Methods to detect different types of outliers. pages 23–28, 03 2016.

[9] T. Ehret, A. Davy, J.-M. Morel, and M. Delbracio. Image anomalies: a review and synthesis of detection methods. 08 2018.

[10] N. Liu, D. Shin, and X. Hu. Contextual outlier interpretation. pages 2461–2467, 07 2018.

[11] S. Rao, N. Shah, and H. Patil. Novel pre-processing using outlier removal in voice conversion. 09 2016.

[12] D. Zimmerer, S. Kohl, J. Petersen, F. Isensee, and K. Maier-Hein. Context-encoding variational autoencoder for unsupervised anomaly detection – extended abstract. 07 2019.

[13] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. 2018.

[14] J. Deng, W. Dong, R. Socher, L.-J Li, K. Li and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pp. 248–255.

Section 3

AI Industrial Machinery

3.0

AI in Industrial Machinery

Giulio Urlini[1], **Janis Arents**[2] **and Antonio Latella**[3]

[1]STMicroelectronics, Italy
[2]EDI - Institute of Electronics and Computer Science, Latvia
[3]SCM Group, Italy

Abstract

This introductory article opens the section on "Advancing Artificial Intelligence in Industrial Machinery Applications". It gives an overview of the state-of-the-art AI technologies in industrial machinery and the current AI development in efficiency improvement, personnel safety, automation, and human-machine interaction. It also presents future potential and opportunities for AI in the sector, covering trends of using AI, IIoT technologies, and advanced actuation and sensing techniques, safety/quality, maintenance, waste reduction, and environmental sustainability. Finally, the article introduces the four contributions to this section.

Keywords: artificial intelligence (AI), industrial internet of things (IIoT), industrial automation, predictive maintenance, human-machine interaction, smart manufacturing, edge computing, smart robot.

3.0.1 Introduction and Background

Today, AI is a powerful source of disruption and a tool to achieve a competitive advantage in industrial manufacturing. The manufacturing companies that neglect to recognise the importance of AI are expected to lose their competitive edge. Many industrial manufacturing facilities are implementing AI across the value chain, but still, many are only using

AI in core functions such as engineering, product development, assembly, and quality testing. The main reasons for implementing AI technologies in industrial environments are driven by the need to assist in making decisions or acting, automate manual and cognitive tasks, and augment decision-making through continuous machine learning (ML) [5]. There has been a rapid growth in AI development and deployment in the last decade. Machines already complete 29% of simple or complex tasks today [7].

Modern production processes in the manufacturing industry and the process industry have reached a critical level of complexity. Stable operation and constantly high product quality are maintained only through continuous monitoring, inspection, and adaptation. This applies in particular to the industrial landscape in Europe, which has a strong focus on customisable products and highly specialised processes rather than standardised mass production. New business models (e.g., lot-size one production) and intense competition from outside Europe require increasing speed and reducing complexity overhead. Through automation, artificial intelligence (AI) and ML are key technologies for managing this increasing complexity in the future manufacturing and process industry. Examples include plant reconfiguration on demand in Industry 5.0, automatic proactive online adaption, optimisation of process parameters, and predictive production planning.

Integrating AI/ML in future manufacturing lines and processes heralds a new era where interactions between people and machines become more integrated, and the decision-making process is driven by data and AI. In other words, the current fragmentation within and outside the manufacturing lines will evolve towards a system where manufacturing processes are connected, and decisions are taken accordingly by the data analysis coming from different sources. Implementing an AI/ML method in a production system can address both the actual production (physical level) and the monitoring and planning of the production (abstract level). Skilled human workers will continue to play an essential role at both levels and are therefore the most critical factor that needs to be considered in the automation process. AI4DI addresses this challenge in its third pillar. While pillars one and two covers the technological and methodological challenges involved in rolling out AI/ML in a production environment, pillar three is vital for the final system's acceptance and efficient and novel human-machine interactions.

3.0.2 AI Developments and Future Trends in Industrial Machinery

With AI entering the manufacturing floor, starts the use of digital technology to replace not only "muscles" but also "brains". In the last few years, AI has become deeply embedded across industrial and other applications, with initial use cases using AI in manufacturing representing niche applications and expanding into mainstream production.

The current adoption rate of AI in manufacturing is relatively low, and the prevalence of AI is expected to increase significantly by 2030 [5].

Industrial machinery is changing alongside society, and everyday life, while digitisation is rapidly becoming de facto. Jobs that are repetitive, tedious, and do not require high skills are slowly being replaced by smart manufacturing systems. AI-based approaches are internationally accepted as the main driver [1] for digitisation and transformation of factories since flexibility and deep understanding of complex manufacturing processes are becoming the critical advantage to raise competitiveness [2]. By looking at smart manufacturing and digitisation trends [3], the factories of tomorrow will be multi-purpose and able to adapt to new designs in a very short time. Similarly, smart industrial robot control methods will allow robots to adapt to the stochastic environment, enabling more human-like performance by completing tasks that have not been directly programmed to the robot or intuitively interact and collaborate with humans.

IIoT and AI-based real-time monitoring in industrial machinery can optimise production, tracking the different production steps and identify changes in the production parameters. Supervised and unsupervised ML algorithms can interpret real-time data from multiple production shifts and identify unknown patterns in processes, products, and production workflows.

In robotics, vision systems support the development of collaborative robots and cobots. Cobots are used to collaborate with humans in terms of helping or relieving the human operators of repetitive tasks and are expected to evolve and provide automated tasks and connected in a network of intelligent IIoT devices.

Operating, checking, and improving functioning and efficiencies in industrial pieces of machinery requires AI-based solutions designed with embedded technical robustness and safety. The industrial AI systems must be assessed to withstand potential attacks (along with unexpected functioning in new environments) and have fallback plans and similar general safety mechanisms in place. The use of AI solutions has the potential in autonomous

system monitoring to improve safety and efficiency and provide new performant human-machine interfaces [6].

The accomplishments in the field of AI are contributing to innovative industrial robot control trends. The AI usage in robotic systems is firmly becoming one of the main areas of focus as the industrial machinery requires increased performance, adaptability to product variations, increased safety, reduced costs etc. Still, these requirements are neither feasible nor sustainable to be achieved by standard control methods.

Applications of AI are progressing in different areas of industrial machinery manufacturing with a focus on improving quality control/assessment, energy efficiency, safety, maintenance, and process optimisation. A number of these areas where AI technologies have the potential to expand in industrial machinery are listed below [4].

Operational simulation and optimisation are application segments for AI in machinery. Dynamic simulation and optimisation of processes enable end-users to plan the use of the machine/equipment effectively, plan the flow of materials, dynamically supply, and predict possible anomalous scenarios. Key drivers of growth in this segment are the need for end-users to lower overall operating costs and the rise of physics-based AI solutions. The demand for AI solutions that address operational simulation and optimisation grow since more manufacturing lines become more complex plus integrated with the supply chain and processes.

Quality control is increasingly important in industrial machinery production due to stringent quality requirements for industrial products. The AI-based techniques can bring new intelligent quality inspection solutions in the industrial machinery space that support quality control applications across several industrial and machinery segments. AI-based computer vision for quality inspection is used in advanced equipment manufacturing lines with increasing demand for intelligent systems for quality control in all production steps. The developments further advance the evolution of embedded AI at the different micro, deep, meta edge levels.

Maintenance is one of the critical applications for AI in industrial machinery manufacturing that evolves from preventive toward predictive and prescriptive maintenance using AI-based techniques. Increasing machine/equipment efficiency and minimise/eliminate unplanned downtime requires new predictive maintenance solutions. The solutions are based on ML using supervised or unsupervised learning to detect failure patterns for

parts from machine data and predict when the subsequent machine part failure can occur.

Energy management and energy efficiency are significant concerns in industrial machinery/equipment design and their manufacturing processes. AI-based methods used in the industry can support the efficient use of energy in manufacturing facilities, optimising the energy management for various production lines and manufacturing plants. AI-based solutions can predict precise the energy need and type of energy available at the time of use to optimise the integration and use of various energy sources (renewable, fossil) in the production processes.

3.0.3 AI-based Applications

AI4DI partners are developing AI and IIoT technologies with applications in different areas of industrial machinery. The articles included in this section cover several aspects: sensing the environment, making independent decisions, and acting according to the machinery.

The article *"AI-Powered Collision Avoidance Safety System for Industrial Woodworking Machinery"* addresses the challenge of applying AI-technology to safety-critical industrial equipment: it cannot be certified, as, although safety standards do exist for both product and process, they are likely not yet to include innovative algorithms. At the same time, its inclusion in the current certification schemes waits for the technology to become mature enough to trigger industry engagement. The paper attempts to demonstrate by using a prototype (based on ultrasound sensors and coupled with a temporal convolutional network-TCN algorithm) that AI technology can meet safeguards such as halting machinery's operation and bringing it to a safe state when certain conditions are met. The prototype can detect when a person is within a certain distance from the industrial machine with high sensitivity and specificity.

The article *"Construction of a Smart Vision-Guided Robot System for Manipulation in a Dynamic Environment"* addresses the challenges of enabling industrial robots integrated into manufacturing processes to "see" in dynamic environments. The article presents an innovative vision-guided robot system capable of collecting and processing data from various edge devices and adaptive decision-making. Promising preliminary results have been obtained based on synthetic training and validation data generated by open-source software building blocks, easily adaptable and extendable for

other industrial applications. It is yet to be seen results and performance when combining synthetic with real training data sets.

The article *"Radar-Based Human-Robot Interfaces"* addresses the need to ensure robots' safety and active control as they interact more closely with humans in different types of settings. The current vision-only approaches are no longer sufficient and must be improved, for example, using hand gesture recognition capabilities. Two implementations of the radar-based human-robot interface have been explored (one using traditional machine learning classification techniques and the other using spiking neural networks). The implementations are compared in terms of their strengths and weaknesses, and the results are presented and discussed. Finally, some preliminary conclusions on performance trade-offs, gesture set choice, ergonomics are provided. Both implementations successfully detect gestures using a single radar, but more work is needed to improve the detection performance.

The article *"Touch Identification on Sensitive Robot Skin Using Time Domain Reflectometry and Machine Learning Methods"* presents the proof of concept of a novel sensor system for robotic human-machine interface (HMI) applications, mimicking the human sense of touch. The system is enabled by implementing an artificial sensitive skin consisting of a robust and straightforward part of the sensing hardware mounted on the robot combined with adaptive AI algorithms to recognise touch events. A measurement concept based on electrical time domain reflectometry (TDR) allows identifying/remembering touch events, localising them on the sensor surface, and determining the touch-force magnitudes. The information collected from the sensor is pre-processed and then used for training and validation of artificial neural networks to obtain high-accuracy: regressive deep neural networks (DNNs) for identification of the touch positions and forces and classification DNNs for discrete force level identification. The results demonstrate that a high-level accuracy is obtained, and more work is needed to reduce the gap between training and validation accuracy.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research

and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] Probst L., Pedersen B., Lefebvre V., Dakkak L. (2018). USA-China-EU plans for AI: where do we stand. Digital Transformation Monitor of the European Commission. 2018.

[2] Arents J, Abolins V, Judvaitis J, Vismanis O, Oraby A, Ozols K. (2021). "Human-Robot Collaboration Trends and Safety Aspects: A Systematic Review". Journal of Sensor and Actuator Networks. 2021 Sep;10(3):48.

[3] Evjemo L.D., Gjerstad T., Grøtli E.I., Sziebig G. (2020). Trends in smart manufacturing: Role of humans and industrial robots in smart factories. Current Robotics Reports. 2020 Jun;1(2):35-41.

[4] Güldner, F. (2020). The State of Artificial Intelligence in Machinery. Available online at: https://www.arcweb.com/blog/state-artificial-intelligence-machinery

[5] PwC (2020). An introduction to implementing AI in manufacturing. Available online at: https://www.pwc.com/gx/en/industrial-manufacturing/pdf/intro-implementing-ai-manufacturing.pdf

[6] Graham, N. and Sobiecki, M. (2020). Artificial intelligence in manufacturing. Available online at: http://www.businessgoing.digital/artificial-intelligence-in-manufacturing/

[7] World Economic Forum (2018). Future of Jobs Report. Available online at: http://reports.weforum.org/future-of-jobs-2018/

3.1

AI-Powered Collision Avoidance Safety System for Industrial Woodworking Machinery

Francesco Conti[1], Fabrizio Indirli[2], Antonio Latella[3], Francesco Papariello[4], Giacomo Michele Puglia[5], Felice Tecce[5], Giulio Urlini[4] and Marcello Zanghieri[1]

[1]Alma Mater Studiorum – Università di Bologna, Italy
[2]Politecnico di Milano, Italy
[3]SCM Group, Italy
[4]STMicroelectronics, Italy
[5]DPControl, Italy

Abstract

Applying Artificial Intelligence technology to safety-critical industrial equipment requires preliminarily studies on the efficacy and limitations of such technology, to enable the definition of normative certification frameworks. In this chapter, we present the prototype of an ultrasound-based collision avoidance system for industrial woodworking machinery. Using a single ultrasound sensor, the prototype can identify the presence of an operator in less than 13 milliseconds, with high sensitivity (97.3%) and specificity (98.6%) also in the presence of noise. The solution presented is able to leverage increasing amount of data over time to increase accuracy, improving the model while always keeping the inference adequate for the memory, power and latency constraints of real-time execution on an embedded microcontroller unit.

Keywords: electro-sensitive protective equipment, ultrasound processing, time series analysis, machine learning, deep learning, deep neural networks, temporal convolutional networks, safety-critical, embedded systems, TinyML.

3.1.1 Introduction and Background

Every day, industry workers operate on complex pieces of machinery for tasks ranging from woodworking, car construction, circuit soldering, clothing fabrication, etc. Virtually all such machinery requires trained and skilful operation, and it is often hazardous if operated out of well-defined practices. Safeguarding the health of operators without disrupting operativity is therefore a paramount concern in the design of such pieces of machinery, prompting the specification of industrial standards for functional safety of equipment: safeguards that halt the operation of machinery and bring it to a safe state when certain conditions are met, e.g., a person is detected in the trajectory of a moving part by a non-contact Electro-Sensitive Protective Equipment (ESPE) sensor, such as a photodiode.

To ensure a shared set of rules that can be used as a normative framework, safety equipment in industrial machinery must typically be certified under international standards such as IEC 61508 and EN/IEC 61496 to achieve a given Safety Integrity Level (SIL). The former standard deals in general with all programmable electronic equipment, while the latter is more specific on requirements for designing, building, and verifying the operation of non-contact ESPE systems. A downside of international standards is that newly introduced technology that is not covered by the current version cannot be certified – and this includes innovative algorithms. Even technology improving the worker's safety or convenience must wait for inclusion in a new version of the standards, which only happens if the technology itself is mature enough to trigger the industry's interest in its inclusion.

In the fields of machine vision and data analytics, algorithms based on artificial intelligence – and particularly, Deep Learning (DL) – have recently become mainstream, both when executed in the cloud and directly *at the edge*, i.e., on the same computing devices that perform data collection. The most famous family of algorithms, Deep Neural Networks (DNNs), is now considered a mature technology, showing extremely good results for many data analytics tasks. DNNs would be a promising technology for improving the performance of safety equipment such as ESPE systems by extracting more relevant information out of sensor data, for example from the correlation

between multiple streams of information. Edge Deep Learning techniques enable an analysis such as this to be performed directly near the sensors themselves, ensuring high reliability and ultra-low-latency response to critical situations. Despite this promise, applications of AI to ESPE systems are not yet considered by any industrial standard, and industry interest would have to be gathered by means of advanced prototypes demonstrating the effectiveness of this idea in the field.

In this chapter, we showcase a prototype collision avoidance system for industrial woodworking machinery that is based on cheap ultrasound sensors – like the ones used for park assist in cars – coupled with an algorithm based on Temporal Convolutional Networks (TCNs), a sub-family of lightweight DNNs specifically dedicated to time series data analytics. The goal of the collision avoidance system is to detect the presence of a human body in front of the sensor array using only ultrasound information, possibly in presence of intrusive noise coming from other operations in a busy manufacturing environment.

3.1.2 Review of Industrial-level Methods for Edge DNNs

In recent years, DNNs became leading solutions in a broad variety of computational tasks, and they are being extensively integrated into digital industries. The rapid proliferation of pervasive Internet of Things (IoT) devices and of ubiquitous cognitive computing is pushing the industry towards performing Machine Learning (ML) inference on edge devices, which enables real-time processing of data and reduces strain on Cloud networks. These embedded platforms, however, pose stringent constraints on power consumption, latency, and memory footprint. Moreover, some devices are equipped with hardware accelerators that require specialized programming and mapping endeavours 1D be fully exploited These new challenges led to the development of Tiny Machine Learning (TinyML), a novel and rapidly growing field that aims to enable 1he porting of ML algorithms onto embedded computational platforms, characterised by strict requirements in terms of memory and power envelope.

3.1.2.1 Compression Techniques

In the modelling and training stages, several compression techniques can be used to trim the size of the network and reduce the number of computations.

Quantization approximates real (floating-point) values to integers with lower bit widths, enabling reduced-precision computation. In fact, normally most networks are trained using FP32 numbers, but smaller representations can greatly optimize the memory utilization and the inference performance with little loss of accuracy [1]. Model designers can experiment with different numerical precisions for the weights and/or the activations of each layer, building mixed-precision models to achieve the desired trade-off between prediction loss and model size. Although quantization is more effective when applied during training (*quantization-aware training*), also *post-training quantization* schemes are widely used.

Vector compression schemes focus on reducing the constants' size by clustering and sharing the weights and the biases, using algorithms such as k-means or hash functions.

Pruning removes redundant parameters or neurons that do not significantly contribute to the accuracy of results, for example, because they are 0. It can be performed at training time (*static pruning*) or at runtime (*dynamic pruning*) and it can either target the *neurons* or the *connections* between them.

3.1.2.2 Popular Frameworks and Tools

Most of the major deep learning frameworks support some compression schemes and other techniques tailored for Deep Learning at the Edge [2].

TensorFlow Lite is a lightweight framework for on-device inference, based on the popular TensorFlow (by Google). It supports post-training quantization targeting half-precision float (FP16) and INT8 datatypes. Moreover, *quantization-aware training* and pruning can be performed in TensorFlow and Keras. The models produced with these tools can be used also in **TensorFlow Lite Micro**, a runtime framework designed to perform inference on microcontrollers.

Apache TVM is an open compiler stack that provides end-to-end compilation of neural networks (modelled in TensorFlow, ONNX, Keras, or MXNet) to several backend frameworks and hardware targets. It supports quantization up to $1\sim4$ bits, as well as block-sparsity. Moreover, the **microTVM** extension allows targeting small bare-metal devices using a minimal C runtime.

ONNX provides an open format for AI models, aiming to simplify networks exchange between different frameworks, tools, and target hardware. It supports INT8 quantization, both at runtime and during training, for Convolutional, MatMul, and Activation layers.

STM32CubeAI is a software extension pack for the STM32CubeMX code-generation tool, which provides a user-friendly GUI to quickly configure STM32 microcontrollers to run Neural Networks inference. It supports ONNX and TFlite models and can perform post-training compression on them. The generated code provides APIs to use multiple models in the same codebase and to accelerate their execution using ARM CMSIS kernels.

In addition to the popular frameworks listed above, **novel specialized tools** implement more advanced compression techniques and finer-grained quantization settings: some examples are QKeras [3], Larq [4], and Brevitas [5].

3.1.3 Materials and Methods

3.1.3.1 System Architecture

The collision avoidance system that we propose exploits ultrasonic (US) sensing to detect the presence of a person or object within a certain distance from the industrial machine; in case of detection, a STOP signal is immediately conveyed to the machine control logic. The system works in a conceptually simple way: a set of transducers emit an ultrasonic pulse; if the pulse hits a person, it will produce an echo that can be sensed by the transducers themselves.

To work correctly, the system must operate with ultra-low latency, and, at the same time, it has to deal with many possible noise sources that pollute the signal and make it harder to achieve high sensitivity and specificity. To increase the system's resiliency against interfering waves in the US spectrum, we process the acquired data using a Neural Network to discern the US echo from other unwanted noises and detect the presence of a worker in the machine's trajectory.

As shown in the Figure 3.1.1, the main components of this system are:

- The ultrasonic sensors and their drivers
- A Lattice FPGA
- An STM32-H7 microcontroller board

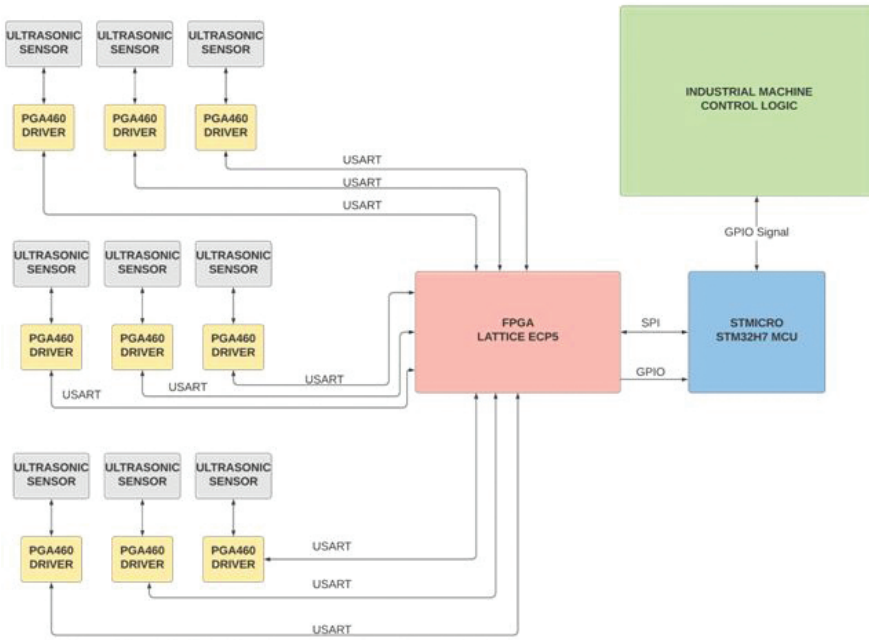


Figure 3.1.1 System architecture schema.

The data acquisition pipeline of the final system will start with a 3×3 matrix of 9 MULTICOMP MCUSD14A58S9RS-30C **ultrasonic ceramic transducers**, while the demo prototype uses only one sensor. Each of these devices acts both as an emitter and as a microphone for sound waves with a frequency around 50 kHz. When the emitted ultrasonic pulses encounter an obstacle, they get reflected towards the sensor, which translates the mechanical vibrations of the echo into a variable electrical tension. By analysing this signal, the distance between the sensor and an object, if any, can be easily derived.

Each sensor is driven by a dedicated Texas Instruments PGA460 **ultrasonic signal processor and driver**, that integrates a low-noise amplifier, a programmable time-varying gain stage, an (up to) 12-bit ADC and a DSP. In the proposed system, the ADC is configured with a resolution of 8 bits that matches the input bit width of the Neural Network. The sampling frequency is 1 MHz and the sampling period for each data chunk is 20.48 ms, yielding an output data rate of 8000 kbps for each driver.

A low-power Lattice ECP5 LFE5U-85F **FPGA** is used to aggregate the data streams (channels) coming from each of the PGA460 devices; moreover,

at start up the FPGA configures the drivers' resolution, sampling rate, and other parameters. The devices communicate using the USART protocol, in which the programmable logic acts as master, while the ultrasonic drivers are slaves. In particular, the communication happens through synchronous USART (clocked at 8 MHz) with a packet size of 8 bits and a baud rate of 8 Mbps.

After collecting all the data of a sampling window from the drivers, the FPGA performs subsampling, reducing the 20480 per-channel samples by a $10\times$ factor; then, all the channels' data is packed to be sent to the STM32 microcontroller. A data package produced by the FPGA contains 2048 8-bit samples for each channel, for a total size of 147.46 Kb. The communication between the two devices happens mainly via the SPI protocol (Mode 0), with a serial clock of 5 MHz, and is initiated by the MCU which acts as the master. An additional GPIO line is asserted by the FPGA to inform the microcontroller when it has finished its tasks.

The STMicroelectronics **STM32H743ZI2 board** (STM32-H7 for short), belonging to the Nucleo-144 family, features a high-performance ARM Cortex-M7 processor (with double-precision FPU) operating at 480 MHz, 2 MB of Flash memory, 1 MB of SRAM (including 192 KB of tightly-coupled scratchpad memory for real-time tasks), 4 DMA controllers, and several communication peripherals including UART/USART, SPI, USB-OTG, Ethernet, and GPIO lines.

Its main duties are to acquire the data from the Lattice device, perform the Neural Network's inference and send control signals to the industrial machine's PLC and to the FPGA (which, in turn, controls the ultrasonic drivers and transducers).

In particular, the MCU prototype firmware implements a state machine (depicted in the Figure 3.1.2) composed of five main states:

(1) *Waiting FPGA configuration*
(2) *Waiting Acquisition*
(3) *Data Transfer*
(4) *Inference*
(5) *Halted*

The system starts in state (1), in which the FPGA configures the transducers drivers and communicates with the MCU. When ready, the FPGA sends a signal to the MCU, which replies with a "*Start Data Acquisition*" command and waits in the state (2). When all the data needed for one inference have been acquired and pre-processed, the FPGA sends a "*Data_ready"* signal

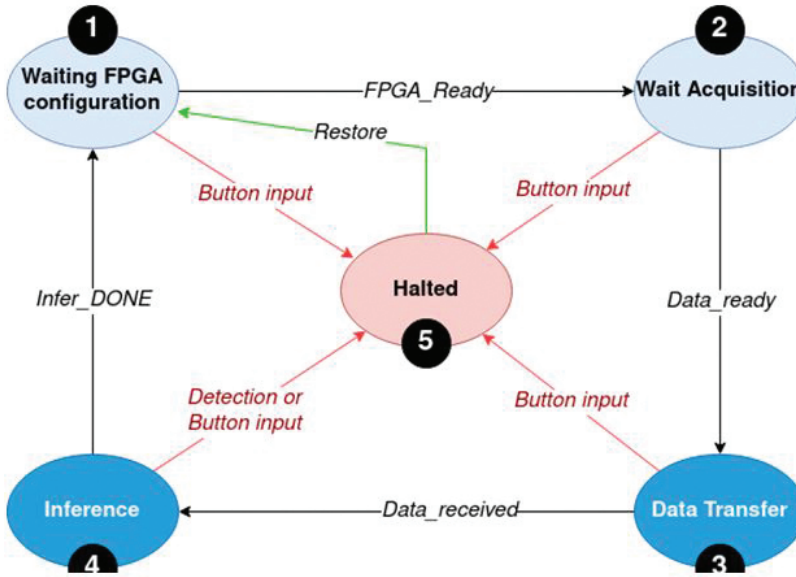


Figure 3.1.2 MCU firmware state machine.

which triggers the data transfer to the STM32 over SPI, happening in state (3). When the payload is received, the MCU informs the FPGA (which will prepare for the next acquisition) and then it runs the Neural Network inference in state (4). If an obstacle is detected, the MCU will assert a GPIO line connected to the industrial machine's controller, and the firmware will move to state (5); otherwise, the system will go back to state (1). The system can be halted also by pressing a specific button on the control panel, to allow the user to report false negatives: this information is stored and could be used to improve the model accuracy in the future. A dedicated command can be used to exit from the halting state and restore the normal execution from state (1).

3.1.3.2 Dataset Collection

To address the targeted use case, an ultrasound (US) dataset was collected with an acquisition setup reproducing the working conditions of the industrial machinery of interest.

Framing the problem as a binary classification task, where the goal is detecting the presence of a human inside the area of interest, US windows were recorded with and without a person originating a US echo.

Working in an anechoic chamber, bursts of US at 50kHz were sent toward a predefined region by utilizing the TI PGA460 illustrated in the previous section. Ultrasounds were sampled at 1MHz, producing UINT8 data; and a window of 28ms was recorded in each acquisition.

In addition to the presence or absence of a person, corresponding to the positive and negative class respectively, two conditions were varied to explore the variability of the real working conditions:

- The person's distance from the sensor, which was varied from 0.5m to 2.0m.
- The pressure level of a compressed-air jet used to reproduce the environmental noise of the machinery's room, which was varied from 0bar to 3bar (applied for the negative class as well).

A total of 227 US windows (85 negatives, 142 positives) were collected with different combinations of the described settings.

Four examples of the acquired US signals are shown in the Figure 3.1.3 (for clarity, only the first 20ms are displayed). All windows include the final part of the US burst, which contains no information but can be easily cut since it has the same timing in every recording. As it can be seen, the information needed for classification is strictly related to the reception of a US echo.

The shown recordings exemplify the motivation for addressing the task with Machine Learning (ML), and with Temporal Convolutional Networks (TCN) in particular. In the ML/DL paradigm, the modelling of the relationship between the signal pattern and the class is completely entrusted to the training of the algorithm, which learns a discriminant function that is entirely *data-driven*, i.e., fully independent from any feature extraction handcrafted ad hoc. Thus, there is no need to develop an analytical description of the positive echo, which would require a huge amount of trial and error. Moreover, TCNs (defined in detail in the next Subsection, *Detection Methods*) are deep models able to work directly on raw signals. In our use case, this prerogative is paramount, since it allows to automatize the discrimination between the signal of interest and the background noise due to the compressed-air jet pressure, making this aspect data-driven as well and avoiding the need for any manual calibration.

For the subsequent Deep Learning setup, data were kept UINT8, and under-sampled to 100kHz and randomly split into a training set and a test set with a 50%-50% proportion. The US burst was discarded from all recordings,

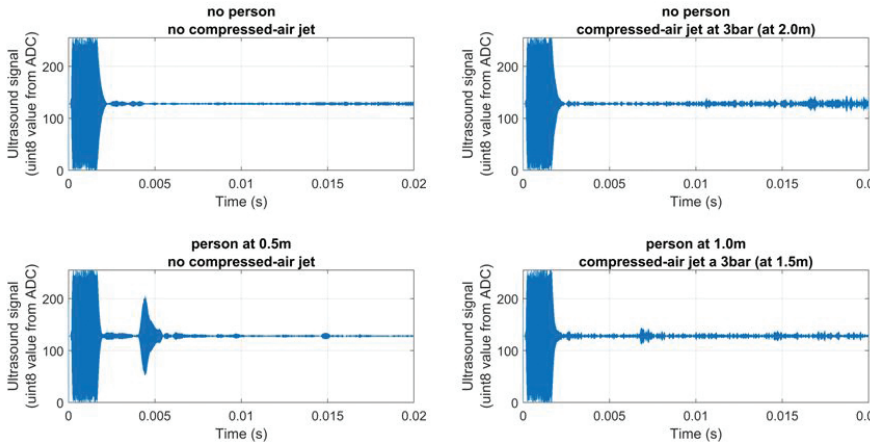


Figure 3.1.3 Visualizations of selected samples of the dataset.

and only 2048 samples (20.48ms at 100kHz) were considered for each window.

3.1.3.3 Detection Methods

Temporal Convolutional Networks (TCNs) are a recent category of deep models that have surged to State of The Art (SoA) in many tasks involving time sequences, surpassing Recurrent Neural Networks (RNNs) for trainability and accuracy [6–8].

The two features of TCNs are in the 1D convolutions, applied along the temporal direction:

- *Causality*: filters only cover a left-neighbourhood of each input sample, to exclude future samples.
- Dilation: a fixed step d is inserted between the kernel's input samples, to enlarge the receptive field while keeping the model size fixed.

The TCN presented in this work is inspired by TEMPONet, a topology that is SoA in real-time classification and regression of bio signals [9]. In this work, we impose a reduction of model size compared to the original TEMPONet structure: we strongly reduce the feature maps (as explained below), obtaining a net that is more compact as to both parameters and activations. Moreover, we use no activation (i.e., we set $d = 1$ for all convolutions), since we experimentally found that dilation provides no increase in accuracy.

The model architecture is made up of 3 Convolutional Blocks, each containing:

- 2 causal convolutions with filter size $k = 3$ and full padding;
- 1 convolution with filter size $k = 5$, variable stride s, followed by average pooling (filter 2, stride 2).

The 3 convolutional blocks have stride $s = 1, 2, 4$, respectively. Searching for a net as compact as possible, we lower the number of feature maps of the 3 convolutional blocks compared to the original model of [9]. We reduce Block I's maps from 32 to 2, reduce Block II's maps from 64 to 4, and reduce Block III's maps from 128 to 4.

After the convolutional blocks, 3 Fully Connected (FC) layers execute the classification. FC I has 8 units, FC II has 4 units, and FC III has 1 unit, representing the estimated probability of the input sequence belonging to the positive class.

All layers, except FC III, have ReLU non-linearity as activation function and are equipped with Batch-Normalization to counter internal covariate shift [10]. FC I and FC II are trained with dropout with $p_{drop} = 0.5$, to help regularization [11].

The optimized TEMPONet proposed in this work processes a 2048-sample input US window (20.48ms at 100kHz) with less than 1000 parameters. The model directly works on raw signals (UINT8 data), without the need for any pre-processing or feature extraction, which would cause overhead before inference.

The TCN model was implemented in Python 3.8 using PyTorch 1.6. Trainings were performed with Binary Cross-Entropy loss (computed in a class-balanced way, i.e., weighting equally the positive and negative class), and Stochastic Gradient Descent (SGD) with AdaM optimizer, initial learning rate $\lambda_0 = 0.001$, and minibatch size 128. SGD was applied for 8 epochs at FP32 precision, followed by Post-Training Quantization (PTQ) to 8bit (i.e., INT8). Then, 8 epochs of quantization-aware training were performed, applying PArameterized Clipping acTivation (PACT) [12] as implemented in the library NEMO (NEural Minimizer for tOrch [13], [14], an open-source package to quantize CNN for deployment on memory-constrained ultra-low power platforms, such as the STM32-H7 MCU targeted in this work. These further 8 epochs of quantization-aware training are a fine-tuning which mitigates the initial drop of accuracy due to PQT (as shown in the Section Results).

On the edge device, different variants of this TCN were tested (for example, with or without dilations), but all the presented experiments are based on the TCN without dilations.

Since an STM32 microcontroller is being used, the models are fed to the X-CUBE-AI extension (version 6.0) for STM32CubeMX, which configures the MCU and generates the code of the inference application. The generated API contains functions to manage multiple models and to run the inference on the data of an input buffer.

Three versions of the chosen TCN were produced, tested, and compared:

- The original model *(TCN FP32),* using FP32 inputs and no compression or quantization: it consists of 251224 MACC operations, 3.57 KiB of weights, and 24 KiB of activations.
- The compressed model *(TCN FP32 compressed)*, using FP32 inputs and X-CUBE-AI's weight-sharing compression based on *k*-means clustering (with compression level = 8): it consists of 251224 MACC operations, 1.84 KiB of weights, and 24 KiB of activations.
- The quantized model *(TCN UINT8 Quantized)*, which uses UINT8 inputs, outputs, activations, and weights: it consists of 226509 MACC operations, 1.02 KiB of weights and 6.05 KiB of activations.

3.1.3.4 Continual Learning Setup

To simulate a realistic scenario of Continual Learning, where the new data received by the TCN are also stored and used for periodic retraining, we conducted experiments applying an increasing data augmentation to the original US data. Data augmentation allows increasing the variability of the data seen by a model during training, thus improving its generalization capability. Exploring data augmentation provides insight into the model's ability to leverage an increasing training set, improving its learning over time.

In our experiments, data were augmented by a factor F_{augm} ranging from 50 to 1000. From each original US data window, F_{augm} synthetic windows were produced via a two-step transformation:

- Scaling by a random factor s uniformly drawn between 0.5 and 1.5, followed by clipping and rounding to recover UINT8 values.
- Time-shift by a random interval Δt uniformly drawn between -1.5ms and +1.5ms.

Since this augmentation injects randomness from the very beginning of our training pipeline, $N_{\mathrm{rep}} = 30$ repetitions of the experiment were performed for

each explored value of F_{augm}; each repetition involved augmenting the data from scratch and training a TCN from scratch.

3.1.4 Experimental Results

3.1.4.1 Evaluation Metrics

In the setup we propose, we target both *classification metrics*, regarding the goodness of the recognition provided by the algorithm, and *deployment metrics*, to assess the suitability of the model and of the chosen STM32-H7 MCU to the edge-inference workload.

The classification metrics we target are three, and are typical of binary classification in unbalanced settings:

- *Sensitivity* (also called *True Positive Rate* - TPR, or *recall*): the fraction of actual positives the net correctly classifies, formally TP / (TP + FN);
- *Specificity* (or *True Negative Rate* - TNR): the fraction of actual negatives the net correctly classifies: TN / (TN + FP);
- *Balanced accuracy* (or *macro-average accuracy*): the average between sensitivity and specificity.

In contrast with unbalanced accuracy, these three metrics are independent of class imbalance. Moreover, recourse to sensitivity and specificity allows characterizing the model's behaviour exploring different sensitivity-specificity tradeoffs, which is methodologically interesting as it allows to tune the model's detection threshold based on the application-specific relative importance of False Positives and False Negatives.

The deployment metrics we use are three, to assess the satisfiability of the main requirements of real-time on-edge computations:

Memory footprint: amount (and percentage of the available quantity) of RAM and FLASH memory used by the firmware (including the neural network); since dynamic memory is not used, this metric is available at compile-time.

- *Latency:* amount of time or clock cycles needed to complete one inference, once the input data is ready.
- *Power consumption:* average power draw of the MCU during a sequence of inferences.

The *memory footprint* metric is particularly important in a resource-constrained scenario and provides interesting insights on which alternative devices could be used for the computation. In terms of performance

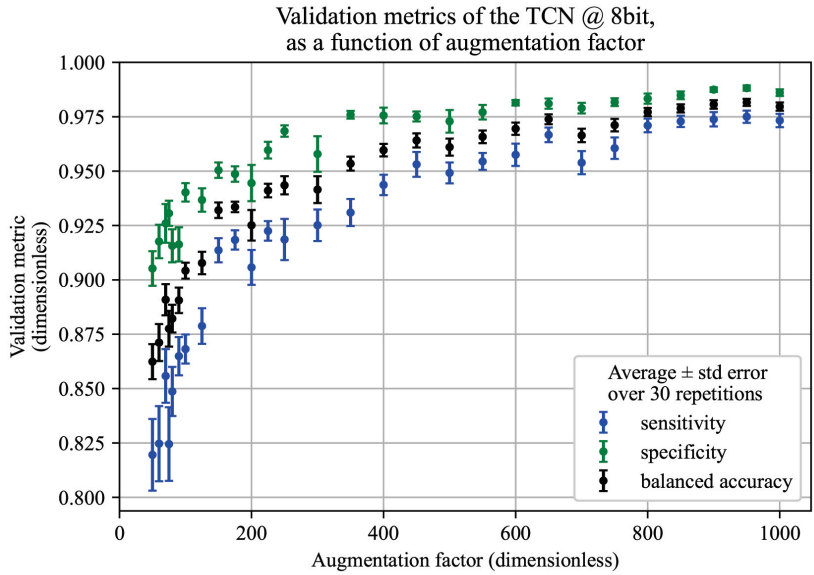


Figure 3.1.4 Validation metrics at different augmentation factors.

assessment, *throughput* was not considered since this system is mainly sequential and does not exploit parallelism or pipelining.

3.1.4.2 Continual Learning Scenario

Figure 3.1.4 shows the validation metrics of the proposed TCN quantized to 8bit, after training on data augmented by a factor F_{augm} ranging from 50 to 1000. The observed trend is a clear learning curve, which highlights that our setup is able to leverage an increasing amount of data to improve the recognition.

The same behaviour is shown by a further characterization, namely the Receiver Operating Characteristic (ROC) curve for the validation metrics, reported in the Figure 3.1.5. As it can be seen, the model is able to exploit the larger training set, produced by stronger augmentation, to improve its sensitivity-specificity Pareto frontier.

Our setup is thus well-suited for a Continual Learning scenario, where the classifier is required to improve its goodness by exploiting new data seen in periodic re-training.

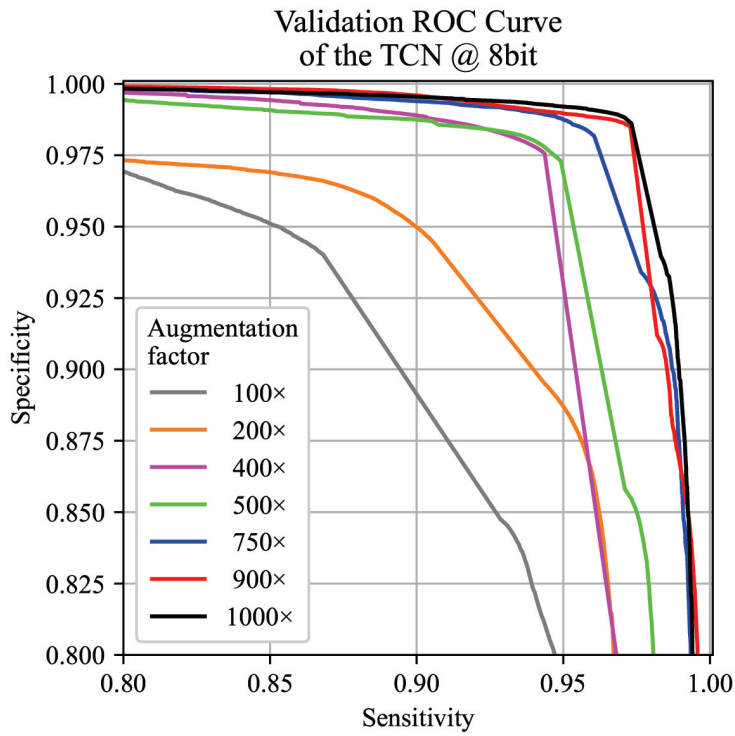


Figure 3.1.5 ROC curves at different augmentation factors.

3.1.4.3 Robustness Against Quantization

Table 3.1.1 reports the validation metrics at different stages of training, obtained for augmentation factor $F_{\text{augm}} = 1000$ and averaged over the $N_{\text{rep}} = 30$ repetitions run. Each validation is performed as an inference over the test set. The best recognition is achieved by the model in FP32 format. However, this numeric type is not hardware-friendly and is not suitable for deployment. After Post-Training Quantization to 8bit and 1 epoch of quantization-aware training, the model's goodness at validation drops by 1.75%, 0.74%, and 1.25% in sensitivity, specificity, and balanced accuracy, respectively. After 8 epochs of quantization-aware training, these degradations (at validation) are reduced to 1.02%, 0.14%, and 0.58% respectively.

This demonstrates that 8bit fine-tuning is capable to recover the accuracy drop following quantization. The final negligible deterioration proves that the proposed TCN is robust against quantization to 8bit.

Table 3.1.1 Validation metrics at different training stages.

Training level	Validation metric		
	Sensitivity	**Specificity**	**Balanced accuracy**
8 ep. FP32	0.9835 (baseline)	0.9875 (baseline)	0.9855 (baseline)
8 ep. FP32 + 1 epoch INT8	0.9660 (- 0.0175)	0.9801 (- 0.0074)	0.9730 (- 0.0125)
8 ep. FP32 + 8 epoch INT8	0.9733 (- 0.0102)	0.9861 (- 0.0014)	0.9797 (- 0.0058)

3.1.4.4 Latency, Energy and Memory Footprint on STM32H743ZI

The three TCN variants described in Section 3.4.1 were deployed and tested on the STM32 MCU using the X-CUBE-AI platform. To perform the measurements, the framework's "System Performance" application template was used, and the board was connected to a PC through its microUSB port, which served also as power source.

Table 3.1.2 reports the measured values of the chosen deployment metrics.

For each model, the execution latency is the average value over 16 inferences with random input data.

The memory footprint was computed statically by the compiler, based on the size of the program's segments and their location according to the linker script: in these experiments, the *.text* segment was allocated in the Flash memory, while *.data*, *.bss* and *.stack* were saved in the RAM. The values of RAM and Flash utilization reported in the Table 1.1.2 refer to the models' data structures, as reported by X-CUBE-AI.

The power consumption was measured using a USB power meter connected between the board and the computer. For each model, we report the average power draw over a 30-seconds interval, during which a continuous cycle of inferences was being executed on the board.

Despite the baseline model (TCN FP32) already meeting the selected requirements, the results show that compression and quantization techniques can successfully reduce the memory footprint and the power consumption on this workload. However, limitations in the current version of the framework prevented latency optimizations. Our future work will focus on optimizing

Table 3.1.2 Deployment metrics on three model variants.

Model version	**Latency**	**RAM Util.**	**Flash Util.**	**Power Draw**
TCN FP32	11.412 ms	32 KiB	3.57 KiB	1.639 W
TCN FP32 Compressed	11.418 ms	32 KiB	1.84 KiB	1.637 W
TCN UINT8 Quantized	12.768 ms	8.11 KiB	1.02 KiB	1.621 W

the inference time on top of the platform, to combine the quantized model's higher memory efficiency with performance gain.

3.1.5 Conclusion

In this work, an ultrasound-based and AI-powered collision avoidance system for industrial machinery was presented. Its development required engineering of the hardware acquisition and processing pipeline in terms of hardware components, system integration, and firmware development, as well as dataset collection, models optimization and deployment. The resulting 1-sensor prototype is able to accurately sense the presence of a person in the working area with low latency.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] T. Liang, J. Glossner, L. Wang, and S. Shi, "Pruning and Quantization for Deep Neural Network Acceleration: A Survey," Jan. 2021, Accessed: Jun. 07, 2021. [Online]. Available: http://arxiv.org/abs/2101.09671.

[2] M. G. S. Murshed, C. Murphy, D. Hou, N. Khan, G. Ananthanarayanan, and F. Hussain, "Machine Learning at the Network Edge: A Survey," Jul. 2019, Accessed: May 31, 2021. [Online]. Available: http://arxiv.org/abs/1908.00080.

[3] C. N. Coelho et al., "Automatic deep heterogeneous quantization of Deep Neural Networks for ultra low-area, low-latency inference on the edge at particle colliders," Jun. 2020, Accessed: Jun. 17, 2021. [Online]. Available: http://arxiv.org/abs/2006.10159.

[4] T. Bannink et al., "Larq Compute Engine: Design, Benchmark, and Deploy State-of-the-Art Binarized Neural Networks," Nov. 2020, Accessed: Jun. 17, 2021. [Online]. Available: http://arxiv.org/abs/2011.09398.

[5] Alessandro Pappalardo, "Xilinx Brevitas." Zenodo, doi: 10.5281/zenodo.3333552.

[6] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal Convolutional Networks for Action Segmentation and Detection," Jul. 2017, doi: 10.1109/CVPR.2017.113.

[7] S. Bai, J. Z. Kolter, and V. Koltun, "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling," Mar. 2018, Accessed: Jun. 17, 2021. [Online]. Available: http://arxiv.org/abs/1803.01271.

[8] J. L. Betthauser, J. T. Krall, R. R. Kaliki, M. S. Fifer, and N. V. Thakor, "Stable Electromyographic Sequence Prediction during Movement Transitions using Temporal Convolutional Networks," in International IEEE/EMBS Conference on Neural Engineering, NER, May 2019, vol. 2019-March, pp. 1046–1049, doi: 10.1109/NER.2019.8717169.

[9] M. Zanghieri, S. Benatti, A. Burrello, V. Kartsch, F. Conti, and L. Benini, "Robust Real-Time Embedded EMG Recognition Framework Using Temporal Convolutional Networks on a Multicore IoT Processor," IEEE Trans. Biomed. Circuits Syst., vol. 14, no. 2, pp. 244–256, Apr. 2020, doi: 10.1109/TBCAS.2019.2959160.

[10] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in 32nd International Conference on Machine Learning, ICML 2015, Feb. 2015, vol. 1, pp. 448–456, Accessed: Jun. 17, 2021. [Online]. Available: https://arxiv.org/abs/1502.03167v3.

[11] N. Srivastava, G. Hinton, A. Krizhevsky, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," 2014. Accessed: Jun. 17, 2021. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html.

[12] J. Choi, Z. Wang, S. Venkataramani, P. I.-J. Chuang, V. Srinivasan, and K. Gopalakrishnan, "PACT: PARAMETERIZED CLIPPING ACTIVATION FOR QUANTIZED NEURAL NETWORKS," 2018.

[13] "GitHub - pulp-platform/nemo: NEural Minimizer for pytOrch." https://github.com/pulp-platform/nemo (accessed Jun. 17, 2021).

[14] F. Conti, "Technical Report: NEMO DNN Quantization for Deployment Model," Apr. 2020, Accessed: Jun. 17, 2021. [Online]. Available: http://arxiv.org/abs/2004.05930.

3.2

Construction of a Smart Vision-Guided Robot System for Manipulation in a Dynamic Environment

Janis Arents[1], **Modris Greitans**[1] **and Bernd Lesser**[2]

[1]EDI - Institute of Electronics and Computer Science, Latvia
[2]VIF - Virtual Vehicle Research GmbH, Austria

Abstract

This article outlines the construction of universal, user-friendly smart robot system for manipulation in a dynamic environment through AI-based vision system which incorporates processing on the edge. To successfully perform complex tasks in changing conditions, robots require both intelligence for adaptive decision-making and the ability to accurately perceive the environment and interface with it. The proposed system is built in a way that maximizes the modularity of the system. And thus, improves the ease at which the system can be modified to other specific goals after it has been operationalized. In this work, these characteristics are achieved by the use of synthetically generated data and Robot Operating System (ROS) as a middleware software. The first results prove the feasibility of training object detection networks on synthetically generated data sets. And also a combination of a 3D camera and industrial robot provides a convenient way for adding new objects to the database.

Keywords: edge computing, artificial intelligence, smart robot, smart manufacturing, synthetic data generation, robot operating system, computer vision, object detection, verification and validation.

3.2.1 Introduction and Background

Over the last decades, robots are increasingly used to off-load the physical labour of workers and to perform tasks more efficiently and accurately. This has resulted in a significant increase in productivity and quality of the performed tasks and manufactured products [1]. At first, robots were mainly used to take over the repetitive tasks of the workers. Today AI-based robotic systems are becoming an increasingly important part of manufacturing processes [2], [3], but industrial robots lack abilities to tackle the dynamic environment of today's manufacturing.

Manufacturing processes are equipped with a wide variety of sensors and cameras for quality control, safety fields, object detection in the 2D environment etc. Most of these processes are manually programmed for one specific task with only little tolerance for changes or adaption to different environments. Industrial robots in these systems are capable to manipulate with objects very precisely and repeat tasks with high accuracy. However, traditionally working in dynamic environments (especially with randomly distributed objects) still requires either human resources or dedicated sorting hardware. The latter one is usually spacious, expensive, and costly/time consuming to adjust if product assortment changes.

Changes in the marketplace translate into uncertainty for the manufacturing and end user mobility services. The way for business to succeed is by being flexible, smart, and effective in the manufacturing process [4]. However today many factories are still effectively designed for single purpose, that means there is little or no room for flexibility in terms of product design changes. In this article we propose a universal, user-friendly, and modular system that enables robots to "see" and work with randomly dropped objects that are overlapping with each other in a pile.

3.2.2 Challenges of Enabling Robots to "See"

The attention to picking and placing of arbitrarily placed objects that are overlapping each other in a pile has increased in the last years [4], especially in the context of Industry 4.0 and smart manufacturing [5]. The described problem is not only challenging to solve but also to be adapted and deployed to different manufacturing sectors.

3.2.2.1 Modularity

Different industries or manufacturing sectors have diverse conditions especially in the context of vision, such as lightning conditions,

environmental elements, and most importantly varied object types. The challenge within modularity includes efficient adaption to changes in the environment by respective component change such that a replacement of one component has no (or only minimal) impact on other components within the system.

3.2.2.2 Operability

Even though the modularity is a critical characteristic of a universal and flexible system, the operability plays an important role as well. Whereas the challenge with respect to operability includes user-friendliness and easy adaptability to other specified goals should be doable/manageable without any deep specific knowledge of the underlying target technology.

3.2.2.3 Computer Vision Algorithms

Two computer vision problems - object detection and instance segmentation - are sufficient to automate many tasks of an industrial robot. The detection indicates where in the camera's frame an object is located, and which class does this object belong to. Whereas segmentation determines which class does every pixel of an image belongs to. Instance segmentation is a type of segmentation that differentiates among pixels belonging to different instances of the same class. With this information, one can acquire a visible shape of a specific object and use it to determine an object's pose, which in turn is handy for picking up and manipulating the object. However, detection and segmentation are challenging tasks in the case of randomly piled objects. The objects are often only partially visible, and when the pile consists of similar or even the same object types, the similar features that could be used to detect the unobstructed objects are scattered all over the pile [6]. Similarly, an instance segmentation algorithm might struggle to distinguish similar and partially overlapping objects. Thus, the AI-based methods depend on annotated training data, where each new object requires numerous new training examples of pile images and labelling of such data is a tedious and very time-consuming manual labour, especially in the case of image segmentation tasks.

3.2.2.4 Validation of Algorithms

Since the advent of deep AI algorithms not only the performance but also the complexity of the respective algorithms has drastically increased.

The increased complexity of these algorithms in turn poses a unique set of challenges to both system designers and system validators. Since laboratory-based testing and user validation often forms a very time-costly and expensive task, simulation-based testing and user validation are often a preferred method in this aspect to (a) shorten development cycle times and (b) to reach a higher level of system maturity before testing and validating the system under laboratory- and real-world conditions.

Simulation- as well as laboratory testing and validation methods in general face both, a significant state space explosion problem as well as a gap to the real-world environment. For vision-based systems, this may arise from many aspects, for instance light conditions or dirty or distorted lenses or sensors etc. It is therefore crucial to system testers and validators to design their experiments not only as close as possible to the real-world conditions, but they must also be aware about the coverage of representative corner conditions and border cases which might affect the system in the field. Only in this way experiments can be designed to address many issues as possible beforehand, and to report valuable feedback to the systems designers during development cycles.

3.2.3 Requirements

To successfully perform complex tasks in changing conditions, robots require both intelligence for adaptive decision-making and the ability to accurately perceive the environment and interface with it. Enabling robots to "see" in terms of ability to work with objects that are different and unstructured in piles where industrial robot movements cannot be pre-programmed can support many workers in challenging working environments. However, this ability requires specifically designed solutions which addresses the stated challenges.

These needs introduce a sufficient degree of modularity to the system as a strong requirement to keep a high operability in both changing environments but also when changing system components: goal is to keep both the effort as well as cost as low as possible when adapting the system to e.g. other types of objects, changed environmental lighting conditions, but also when adapting the system to use other hardware components like other types of sensors, (which might feature different characteristics in terms of resolution, accuracy and even sensor failures like lens distortions, which might have to be handled differently for different sensors). To alleviate the training data acquisition

process and simplify the use of computer vision methods in industry the solutions require more efficient data preparation.

The extent to which a system can be decomposed into independent interacting modules that can be separately understood is beneficial not only with respect to Verification and Validation (V&V) efforts. Thus, to reduce complexity, independent and interchangeable system component-modules have to be defined that can be separately implemented, tested and validated to achieve a specific functionality. I.e. when the robot should be retrained to handle different kinds of objects under different environmental conditions, it should not be required to redesign or revalidate the perception system itself; when the perception system is being exchanged for another one, it should not be required to redesign and revalidate the module handling the user interaction or the module performing high level decision making. However, in such a case it is still important to validate the proper integration of the new module to provide assurance about the system as a whole.

The design of representative experiments featuring a high coverage of potential issues arising in the field requires a comprehensive standardization of the experiments with simultaneous preservation of degrees of freedom for adapting the experiments to modified use case requirements as well as to similar application domains. Thus, with respect to the proposed validation framework, we aim at accompanying the AI algorithm design- and training phase by providing a toolbox that allows for efficiently creating standardized representative experiments while being easy to handle and by being as intuitive as possible to the user.

Being based on existing, publicly available open source software building blocks, the validation framework should form a software abstraction layer easing the handling of a synthetic image generator to setup and conduct a variety of simulation scenarios (including physics simulation), the rendering of realistic image scenes as well as the generation of required ground truth annotation information (i.e. segmentation- and depth images/information) without having to directly deal with the complexity of the different underlying lower level software modules.

Furthermore, the toolbox shall come with a set of helper methods trying to ease (a) the generation of synthetic experiments (i.e. generated benchmark datasets), (b) the evaluation of the system-under-test's performance on generated datasets, while supporting the usage of hardware accelerators (i.e. GPUs) and to allow for parallel data processing using remote- and distributed computing. In addition, by using generic interface types, the framework shall be extendable with little effort to also incorporate other software modules

later on, like, for instance style transform networks to further decrease the simulation-reality gap or to also allow for applying adversarial dataset generation techniques.

3.2.4 Proposed Solution

The architecture of the proposed solution, Figure 3.2.1, is built in such a way that maximizes the modularity of the system and improves the ease at which the system can be modified to other specific goals after it has been operationalized. The experimental system is intended to be fully functional by the use of at least two 3D cameras and respective edge devices in a combination with an industrial robot. Moreover, the designed architecture does not preclude adding an additional camera-edge-robot blocks to supplement a pick and place process in scenarios when different objects are mixed in one pile and requires different grasping strategies. In following the sections hardware and software components will be described in more detail.

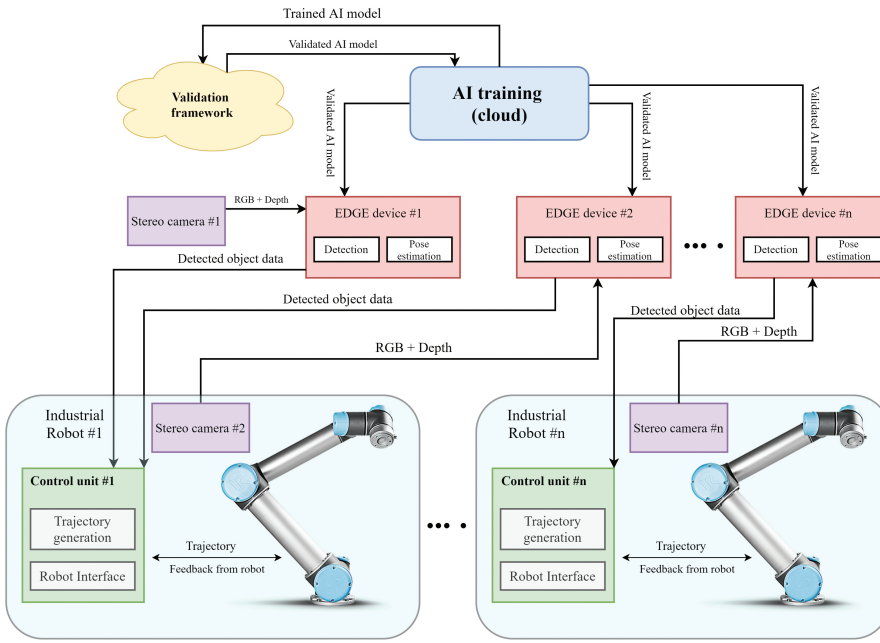


Figure 3.2.1 Architecture of the proposed solution.

3.2.4.1 Hardware and Interface Components

3.2.4.1.1 Robot Interface

Several interfaces are available to communicate with a robot [7]. The robot operating system (ROS) [8] is a popular abstraction layer to interface with a robot, and we propose that ROS is used for our proof-of-concept demonstrator. Using ROS for interfacing allows easier re-use for other robot types. A limitation of ROS is that no hard real-time constraints are supported. This will be addressed in ROS-2, a major rewrite of the ROS code. For a proof-of-concept, both ROS and ROS-2 are valid options, and over time it is expected that ROS-2 will become the preferred option for commercial products. ROS is supported for different operating systems – with Ubuntu Linux the main supported operating system.

3.2.4.1.2 Industrial Robot

Using ROS as an abstraction layer, a wide range of devices can be supported. For our proof-of-concept demonstrator, a Universal Robots UR5, 6 DOF (degreed-of-freedom) robotic arm is considered as a hardware platform for a smart robot. The maximum payload of UR5 can reach up to 5kg and the default reach is 850mm. Thus, the reach of the robot can be improved by gripper modifications. The repeatability of the UR5 robot is +/- 0.1mm.

3.2.4.1.3 3D Cameras

To enable robots to "see" we propose the usage of at least 2 3D cameras. One camera is statically mounted above the robot, perceives the environment around it and locates the region of the object of interest. The second camera is mounted on the robot as gripper modification for closer and more precise data acquisition from the object of interest. In this work, we have evaluated two different cameras for this task: a *Zivid One M* stereo camera [9] and an *Intel RealSense D415* stereo camera [10]. The *Zivid One M* stereo camera uses structured light as 3D technology, features a resolution of 1920x1200 and a common point precision of 60 μm and operates at up to 12 frames per second. The main parameters of the *D415* stereo camera are a resolution of 1280x720 for depth images and a frame rate of up to 90fps.

3.2.4.1.4 Deep Edge Device

The data acquired from the 3D cameras is then processed on deep edge devices. The evaluation board *ZCU102* [11] was utilized for prototyping and verification purposes and to define the necessary parameters and interfaces

for the carrier board. The carrier board is necessary for the "brains" of the edge processing unit (image and ML algorithm – System-on-a-Module). In essence, SoM is a bare-minimum board with all the necessary peripherals (e.g., RAM, power supplies, etc.) forming a stand-alone system. The SoM is connected to a carrier board, which has the necessary peripheral devices and connections for the use-case. By utilizing a SoM, it is possible to reuse the "brains" of the operation in different systems or to easily upgrade them in case there is an increased demand for performance. Therefore, the computing unit and its periphery is separated and can be upgraded independently. In our use case, the aim is to perform image and ML algorithm processing at the edge, therefore the developed carrier boards dimensions must be minimized. To interface with the stereo camera and other USB peripherals, an USB3 hub has been integrated into the carrier board. To forward the processed data to the robot control unit, the carrier board has gigabit ethernet connectivity to ensure a low latency connection with the control unit.

3.2.4.2 Software Components

3.2.4.2.1 Computer Vision Algorithms

The perception software module consists of an image and depth-map processing AI that segments images, detects pickable objects, and determines the orientation of the detected object. The detection is accomplished by a YOLO deep neural network architecture [12] and the segmentation is done by a Mask R CNN instance segmentation model [13]. To detect objects using YOLO the data from the camera must be pre-processed. The main step of pre-processing is object scaling regarding to the trained model, so that the object proportions are the same. YOLO detects all the objects in one frame on which the model has been trained on and then selects the best pickable object by the highest confidence rating. Additionally, the object's pick position is determined in the same frame and a name/ID is defined for the object. Thus, not all detected objects can potentially be picked by the robot arm, as the object could be too close to the side of the container, or the approach angle is too high. Therefore, different parameters are applied, and initial collision checking is done to decrease the *pick&place* cycle time and increase the picking success rate.

3.2.4.2.2 Synthetic Data Generation

Data preparation for machine-learning tasks plays an important role and according to Cognilytica [14] on average more than 80% of time spent on AI

projects are based on the collection and the processing of the data. The data collection techniques can be distinguished in several methods. The reviewed techniques published in [15] varies for different use cases. For example, in applications such as every-day object detection or machine translation there are publicly available data sets that could be reused and adapted for one's needs for model training [16]. In the context of smart factories, the situation is different, where product variety is changing more quickly, and algorithms must be repeatedly trained on new data sets. In these cases, re-usability of existing data sets is fairly low and manual labeling methods cannot meet the requirements of agile production as it is time-consuming, expensive and usually requires expert knowledge of the specific field. The most promising technique in terms of flexibility and comparatively low cost is synthetic data generation where time consumption is reduced depending on processing power and how optimized the generation algorithms are implemented.

Our proposed data generator itself consists of a python library encapsulating the 3D render engine specific commands for setting up and rendering the synthetic images and forms an easy-to-use abstraction layer supporting an application-specific set of image generation parameters. Since a vast number of synthetic images will be generated, the data generator library further provides hardware accelerator support (i.e. GPUs) wherever applicable as well as support for remote deployment and execution to ease massively parallel remote data generation. The current development version is implemented using Python 3.7 [17] and interfaces the open-source 3D render engine Blender® [18] v2.92.

3.2.4.2.3 Object 3D Reconstruction

The synthetic data generation framework is intended to be used with object 3D models, whereas in some cases the 3D models of the objects of interest are not available. For such situations object scanning software tools using depth cameras are being developed. The scanning software uses a camera mounted on the robot arm, capturing data from different viewpoints, to gather data from all sides of the object. The point clouds gathered from different viewpoints are aligned using the camera position information from the robot system. The alignment is fine-tuned by estimating a transformation for point to plane distance. After alignment, a 3D mesh is generated representing the object.

3.2.4.2.4 Validation Framework

The purpose of the validation framework is to automate the standardized procedure of performance evaluation (i.e., systematically carry out a vast

number of deterministic test inferences using the AI algorithm under test) on the generated datasets and to perform corresponding bookkeeping about the achieved object detection results. The results further serve as valuable information for the designers of the AI-based object detection algorithm during repeatedly improved training- and testing iteration cycles.

The validation framework is being implemented in Python 3.7 featuring a distributed master-worker architecture. It interfaces the synthetic data generator and the AI algorithm under test, a bookkeeping module for storing the validation results and an easily expandable plug-in system for applying specific analytics to the device under test.

3.2.4.2.5 Robot Control

After the object's pick position and orientation is determined a collision-free trajectory is generated, including different pose generation for approaching any detected object and for successfully picking it. A time optimal trajectory generation is used for the generation of trajectories with smooth and continuous velocity profiles. Moreover, several common ROS packages are used for ensuring the modularity with different sensor types and robots. ROS-Industrial is used to extend the advanced capabilities of ROS software to industrial relevant hardware and applications. For example, for interfacing with Universal Robot UR5 driver enabling ROS operation of UR robots is used. Moreover, the MoveIt! [19] motion planning framework that runs on top of ROS is utilized for robot arm navigation, motion and trajectory planning, robot interaction etc.

3.2.4.3 Hardware/Software Partitioning

The main functions of the proposed solution - object detection and pose estimation - will be performed on the deep edge device. Moreover, the pre-processing will be done using an application processing unit APU, but neural network models will be deployed on DPU. Furthermore, the object pose, and type will be sent to the control unit for trajectory generation to pick up the detected object. An additional application server will be used for application interface/GUI, training and validation supervision, edge configurations and implementation of the trained AI model and other applications. The object 3D reconstruction utilizes an industrial robot in a combination with stereo camera to precisely acquire data of the object of interest. The detailed HW/SW partitioning can be seen in Figure 3.2.2.

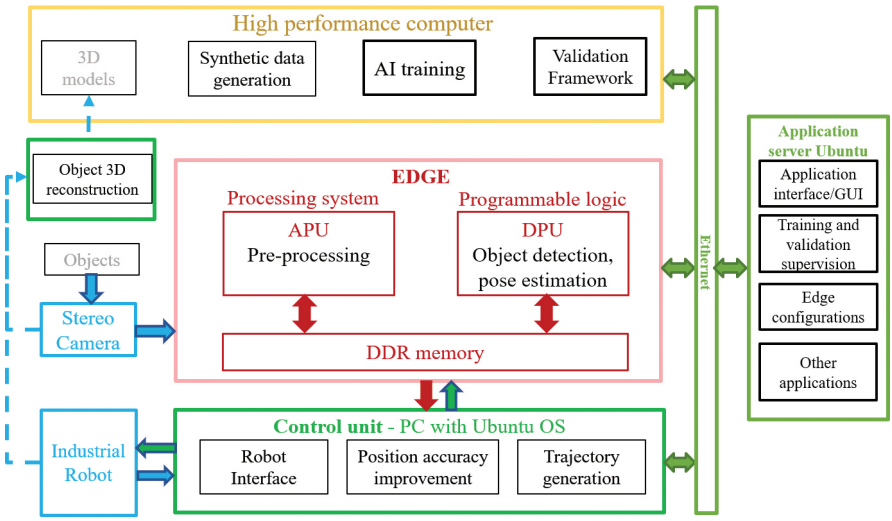


Figure 3.2.2 Hardware/Software partitioning.

3.2.5 Demonstrator Setup and Initial Results

Firstly, the following dataset is generated: for every individual (independent) scene, we fill an initially empty box with 50 randomly placed objects by making use of Blender's physics simulation engine to achieve realistic positioning and orientation. We use textures and Blender's principled BSDF shader nodes to achieve realistic renderings of the scenes including reflections. After filling the box with the objects, we create a series of (data dependent) renderings both varying 4 different light power levels and the orientation of the camera, which orbits around the box and renders the scene from 16 different angles, which can be seen in Figure 3.2.3. For every camera orientation, we also generate a depth image and the segmentation images of the individual objects (Figure 3.2.6) as seen by the camera and labelled by the object ID. We further generate an annotation file for every camera perspective which contains the individual object's orientation and rotation in camera coordinates together with the object's visibility percentage.

First results of a successful implementation on EDGE device of the trained YOLO model on synthetically generated data (Table 3.2.1) is achieved by using only one object type, where the model has been trained to detect only fully visible objects that are not obstructed by other objects. The test data consists of real 300 images.

Figure 3.2.3 Renderings with different light power levels and camera orientations.

Table 3.2.1 First object detection results, where the model has been trained on syntheticaly generated data.

mAP@0.90	mAP@0.75	mAP@0.50	Synthetic data set		Best result
			Total	Augmented	(steps)
44.99 %	94.35 %	96.73 %	4000	16000	6400

Accordingly, the whole workflow of the system has been tested, as it can be seen in Figure 3.2.4, where the AI is used to analyse the data of the Intel RealSense camera, which incorporates processing on the edge (FPGA based SoC). Currently the object detection is done on the edge device and then the further processing such as: object segmentation, pose estimation and communication with the robot is done separately on another processing unit (application server).

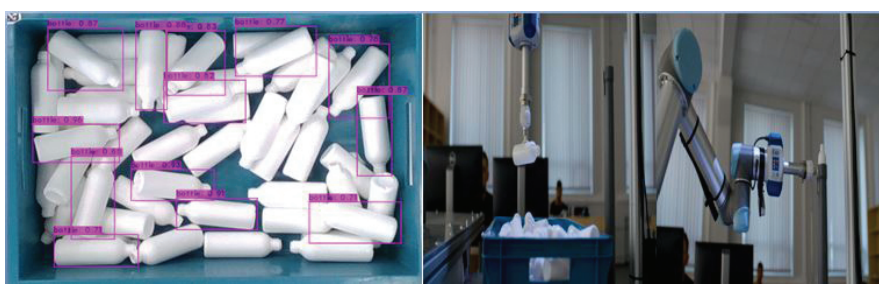


Figure 3.2.4 Object detection and pick and place operation.

Figure 3.2.5 Setup for object 3D reconstruction.

Furthermore, the first results on adding new objects to the scene have been also achieved. The Zivid 3D camera mounted on the robot arm is utilized to reconstruct the 3D model of the object of interest (marked in red in Figure 3.2.5). Depending on the dimensions of the object, the robot moves in a certain distance and angle around the object to precisely acquire a point cloud and generate a 3D model.

The reconstructed 3D model is then added to the synthetic data generation software by mixing different kind of object types in a pile. The next version of the dataset will use 2 different 3D models of objects for filling the box, where half of the objects are matt white plastic bottles, and the other half are shiny aluminium metal cans Figure 3.2.6.

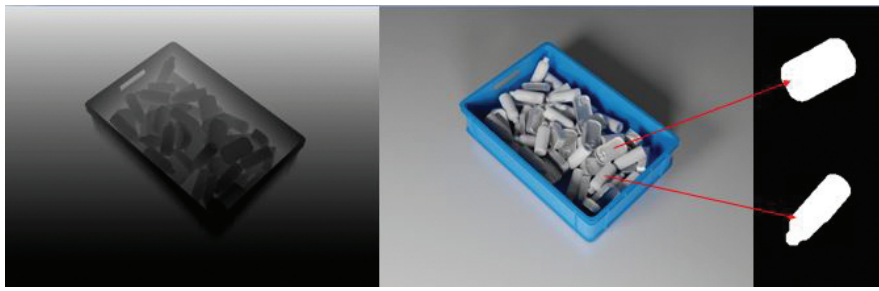


Figure 3.2.6 Scene including the plastic bottle- and the reconstructed metal can 3D models (middle) together with the and corresponding depth image (left) and segmentation masks (right).

3.2.6 Conclusion and Future Work

The proposed system consists of several elements that tackles the challenges of enabling robots to "see". The first results prove the feasibility of the proposed system and form the basis for further developments within the AI4DI project. The proposed hardware and software components leverage the modularity, operability, and the functional correctness of the system. Even though the results of using only synthetic data for training AI-based computer vision algorithms are promising, different combinations of synthetic and real training data sets will be explored as well. Furthermore, future work will include improvements of the object detection algorithms and continued improvements on the used pose-estimation methods. In scenarios where various objects are mixed in one pile different grasping strategies could be required, a case for which multi-robot collaborations methods are being explored and will be implemented during the project.

With respect to the synthetical data generation framework, in this project we are developing a set of open-source software building blocks to automatically generate a large amount of photorealistic training- and validation data for our robotic bin picking use case. The dataset images are fully annotated with position- and rotation information, including depth images, a labelled segmentation mask as well as a visibility score for every object visible in the scene. We believe that our set of software building blocks can be easily adapted or extended and allow for a rapid creation of similar datasets also for other industrial applications. The datasets created during this project will finally be made publicly available on the project website.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] Lee, K. "Artificial intelligence, automation, and the economy." Executive Office of the President of the USA 20 (2016).

[2] Carbonero, Francesco, Ekkehard Ernst, and Enzo Weber. "Robots worldwide: The impact of automation on employment and trade." (2020).

[3] Evjemo, Linn D., et al. "Trends in Smart Manufacturing: Role of Humans and Industrial Robots in Smart Factories." Current Robotics Reports 1.2 (2020): 35-41.

[4] Alonso M, Izaguirre A, Graña M. Current research trends in robot grasping and bin picking. In The 13th International Conference on Soft Computing Models in Industrial and Environmental Applications 2018 Jun 6 (pp. 367-376). Springer, Cham

[5] Buchholz D. Bin-picking: new approaches for a classical problem. Springer; 2015 Nov 29.

[6] Buls E, Kadikis R, Cacurs R, Ārents J. Generation of synthetic training data for object detection in piles. InEleventh International Conference on Machine Vision (ICMV 2018) 2019 Mar 15 (Vol. 11041, p. 110411Z). International Society for Optics and Photonics.

[7] Arents J, Cacurs R, Greitans M. Integration of Computer vision and Artificial Intelligence Subsystems with Robot Operating System Based Motion Planning for Industrial Robots. Automatic Control and Computer Sciences. 2018 Sep;52(5):392-401.

[8] Quigley M, Conley K, Gerkey B, Faust J, Foote T, Leibs J, Wheeler R, Ng AY. ROS: an open-source Robot Operating System. InICRA workshop on open-source software 2009 May 12 (Vol. 3, No. 3.2, p. 5).

[9] Zivid One technical specification 2019. Available from: https://www.zivid.com/hubfs/files/SPEC/Zivid%20One%20Plus%20Datasheet.pdf

[10] Intel RealSense Product Family D400 Series, Datasheet, 2020. Available from: https://www.intelrealsense.com/wp-content/uploads/2020/06/Intel-RealSense-D400-Series-Datasheet-June-2020.pdf

[11] ZCU102 Evaluation Board, User Guide, June 12, 2019 Available from: https://www.xilinx.com/support/documentation/boards_and_kits/zcu102/ug1182-zcu102-eval-bd.pdf

[12] Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: Unified, real-time object detection. InProceedings of the IEEE conference on computer vision and pattern recognition 2016 (pp. 779-788).

[13] He K, Gkioxari G, Dollár P, Girshick R. Mask r-cnn. InProceedings of the IEEE international conference on computer vision 2017 (pp. 2961-2969).

[14] Cognilytica. Data engineering, preparation, and labeling for AI. 2019

[15] Roh Y, Heo G, Whang SE. A survey on data collection for machine learning: a big data-ai integration perspective. IEEE Transactions on Knowledge and Data Engineering. 2019 Oct 8.

[16] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. OpenML: networked science in machine learning. SIGKDD Explorations 15(2), pp 49-60, 2013.

[17] Python Software Foundation. Python Language Reference, version 3.7. Available at http://www.python.org

[18] Community BO. Blender - a 3D modelling and rendering package [Internet]. Stichting Blender Foundation, Amsterdam; 2018. Available from: http://www.blender.org

[19] Coleman D, Sucan I. A., Chitta, S, Correll, N. Reducing the Barrier to Entry of Complex Robotic Software: a MoveIt! Case Study, Journal of Software Engineering for Robotics, 5(1):3–16, May 2014. doi: 10.6092/JOSER_2014_05_01_p3.

3.3

Radar-Based Human-Robot Interfaces

Hans Cappelle[1], Ali Gorji Daronkolaei[1], Ing Jyh Tsang[1,2], Björn Debaillie[1] and Ilja Ocket[1]

[1]IMEC, Belgium
[2]University of Antwerp, Belgium

Abstract

In this article, two implementations of a radar-based human-robot interface are presented. These implementations represent two classes of inference approaches that are investigated in the radar group at imec. The first class exploits traditional machine learning classification techniques. The second class uses spiking neural networks. The machine learning classification system presented in this article supports nine gestures and achieves a gesture classification accuracy of 93%. This compares to an accuracy of 98% for our spiking neural network system operating on four gestures. Based on public data sets, the accuracy of the spiking neural network approach exceeds the published state of the art. Misclassification is however significant, which is still precluding safety critical interactions when using a single radar sensor. As proof-of-concept, a discrete control of a robot will be demonstrated by means of radar-based gesture recognition using five gestures. We present the main concepts of this demonstrator. For pre-validation, we use emulation of the gesture recall statistics and timing characteristics to model the radar part.

Keywords: gesture recognition, 60-GHz radar, machine learning, random forest classification, neuromorphic computing, spiking neural networks, micro-Doppler, human-robot interaction, discrete robot control.

3.3.1 Introduction and Background

In tomorrow's factories, production robots and cobots will need to interact more closely with humans in different types of settings, ranging from advanced assembly lines to the use of exoskeletons to enhance worker capabilities. To ensure safety and active control of those robots, advanced sensors will need to be integrated both on the robots as in the fixed factory infrastructure. These sensors must be reliable and fast while being able to operate in harsh conditions. Often, vision-only approaches will be found to be vulnerable to failure in low visibility conditions.

Millimeter wave radar has the advantage of operating under visually difficult conditions such as darkness, smoke, and dust. Moreover, radar enables to measure the surrounding including the speed of approaches and receding targets. Therefore, radar is excellently suited for collision avoidance. No wonder that this technology is intensively applied in automotive. Radar also enables to use the temporal velocity changes (so-called micro-Doppler patterns) to identify/classify the target. As such, road users can be identified [1], or different hand gestures can be distinguish as demonstrated by Google in their Soli project [2].

Many different approaches can be explored to create a radar-based robot interface. To enhance the intuitive interaction between the operator and the machine, a contact/touchless interface via hand gestures is preferred. These hand gestures could, for example be used to select from a menu (= discrete gestures), or the hand movements could be tracked at real-time to operate the robot (continuous control). Although real-time interaction can be perceived very natural , it comes with significant technical challenges and security risks. Therefore, in this work, we opt for detecting the hand discrete gestures, and we construct a vocabulary allowing the operator to control discrete robot actions. Our envisioned proof-of-concept will control a robot arm taking pictures of an object from different positions and angles. This will enable 3D object modelling.

In this article, we consider two hand gesture recognition implementations using the same 60 GHz radar platform (TI IWR6843 [3]), as well as their suitability for the proof-of-concept demonstrator for interfacing with a robot arm. In our first implementation, the hand gesture type is identified before augmenting it based on the hand location.

This implementation relies on traditional classification techniques based on engineered features [4].

The second implementation does not rely on segmenting the space in quadrants, but only aims to recognize gestures independent on the hand location. This implementation uses a spiking neural network organized as a liquid state machine (LSM) in combination with a trained output classifier layer [5]. For pre-validation of the demonstrator the radar part is modelled by means of gesture command recall statistics and timing behavior.

In the following sections, we start with describing both interface implementations separately. Then we compare both implementations in terms of performance and implementation complexity, as well as how they can be integrated in the proof-of-concept use case.

3.3.2 Gesture Recognition Using a Machine Learning Approach

3.3.2.1 Concept and Experimental Setup

Although the radar system [3] offers only a moderate angular resolution (3x4 MIMO with singe patch antennas), the angular dimensions (both azimuth and elevation) can be exploited to determine the hand position. This provides an extra degree of freedom to design the control interface. The implemented concept is depicted in Figure 3.3.1, showing the training process (in red) and the inference process (in blue).

The result of the radar inference is then used to control a robot arm (in green).Specifically:

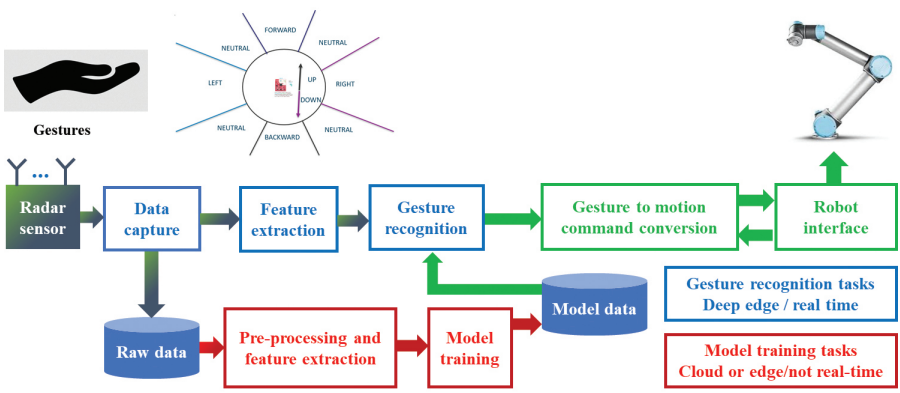


Figure 3.3.1 Human-robot interaction concept using 60 GHz radar.

- For training, indicated in red, raw data is collected. Next radar signal pre-processing is applied, and features are extracted. These features are then used to train the model, and the machine learning model data is saved. These tasks do not require real-time processing and can be done on the edge or in the cloud.
- For gesture recognitions, indicated in blue, features are extracted, and the model is used to recognize the gesture by means of a classifier that uses the machine learning model data.
- The recognized gestures are communicated to the robot part, indicated in green. Gestures are transformed into robot commands which are sent to the robot interface.

The proof-of-concept setup consists of an upwards facing radar mounted in front of the robot operator. The center of detection is approximately 50 cm above the radar sensor. The operated can perform the following hand gestures:

- Palm/hand wave. If a waving hand is detected, then the zone of this gesture is also detected (left/right/front/back/up/down) relative to the detection center. These zones are between 20 and 30 cm away from the center. For example: doing a palm wave at 70 cm (50 cm + 20 cm) above the radar sensor is detected as a "palm-wave/up".
- Pinch. This corresponds to pinching the thumb and index finger.
- Thumbs down. This corresponds to a "thumbs down" gesture with the thumb facing to the radar sensor.
- Tick. This corresponds to making a "V" check movement in the air.

During the measurement campaign, also other gestures were recorded. These were used to model unknown gestures for testing the robustness of the classifier.

3.3.2.2 Inference Pipeline, Training Algorithm

Radar systems exploit electromagnetic waves to detect and locate objects in their environment.

A radar system comprises a transmitter, receiver, and signal processing modules. The implementation uses an FMCW radar [3].

Figure 3.3.2 illustrates an FMCW radar with one transmitter and one receiver, depicting the linear sawtooth and digital signal processing to recover range and Doppler information after the ADC convertor.

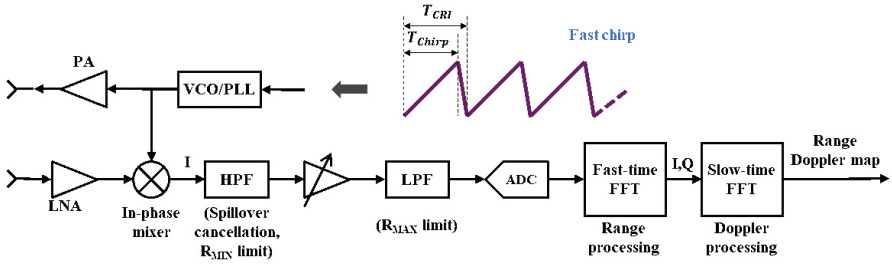


Figure 3.3.2 Simplified radar block diagram.

Transmit Part

The transmit signal is a sinusoid whose frequency is swept linearly from a start frequency to a stop frequency, forming a chirp with a duration of T_{Chirp}. These chirps are repeated with a chirp repetition interval T_{CRI}. A voltage-controlled oscillator (VCO) is used to steer a phased locked loop (PLL) which produces the output signal, which is amplified with a power amplifier (PA) and sent to the transmit antenna.

Receive Part

A reflected signal is picked up by the receive antenna and amplified with a low noise amplifier (LNA). It is mixed with the transmitted signal producing a beat signal that has a frequency that depends on the range (delay) of the reflected signal. A high pass filter (HPF) is used to removed unwanted signal components at low frequencyies, determining the minimum range and cancelling spillover from the transmit signal. Next the signal is amplified by a second amplifier and passed through a low pass filter (LPF) to limit the maximum range. Next, the signal is digitized with an analog to digital convertor (ADC) and a range Doppler map is produced by doing a fast-time fast Fourier transform (FFT) to recover range and slow time FFT to produce Doppler information. Note that by doing so only moving objects can be observed.

In this implementation, all data after the ADC is processed on a laptop, using a data capture board [6]. Angular information can be obtained by combining multiple transmitters with multiple receivers. Figure 3.3.3 shows the signal processing to obtain angular and micro-Doppler information, used to generate feature data. To generate a point cloud a constant false alarm rate (CFAR) detector is used to identify targets, and the MUSIC algorithm is used

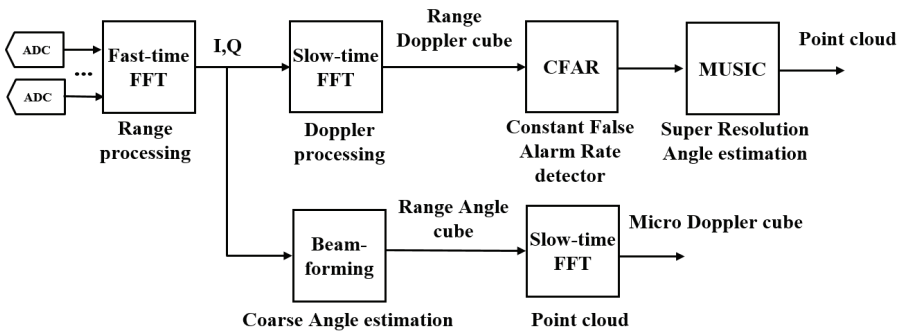


Figure 3.3.3 Signal processing pipeline to provide input to the feature generator.

to annotate identified targets with angular information. To generate micro-Doppler data, beamforming is applied to do a coarse angle estimation, and a slow time FFT to generate a micro-Doppler cube.

To train and evaluate a random forest classifier, feature data is extracted from these signal processing blocks, as illustrated in Figure 3.3.4. We extract ten features, subdivided in four classes [4]:

1. MD: micro-Doppler features:
   - RAW: a sub-sampled micro-Doppler cube
   - ENV: a curve fit of the micro-Doppler envelopes
2. RD_ROI: range-Doppler region of interest features. This corresponds to a denoised and subsampled version of the range Doppler information.
3. POINT: point cloud features. This tracks the average, mode and standard deviation of range (RNG), elevation (ELV), azimuth (AZM) and Doppler (DOP) over several radar frames (of 90 milliseconds each).

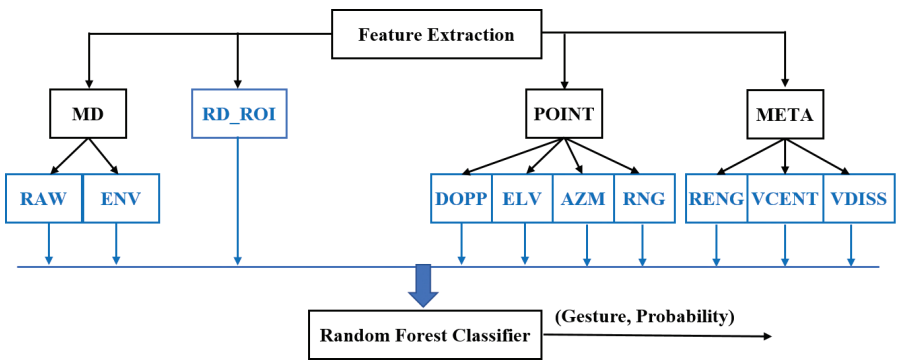


Figure 3.3.4 Feature extraction for the random forest classifier.

4. META. From the range Doppler cube, the following meta parameters are derived:
    - RENG: range instantaneous energy
    - VCENT: centroid velocity
    - VDISS: dispersion of velocity

3.3.2.3 Data Recording and Results

The machine learning approach was based on a supervised framework. First, the setup collected a dataset of gestures for the learning phase. Table 3.3.1 gives the radar parameter used for these measurements [3].

The TI DCA1000 [6] data capture board was used to obtain raw data samples. While the maximum unambiguous range is 11.3 meter, the maximum range was restricted to 1.5 m since larger heights are not relevant for the upward facing radar.

To train the machine leaning model, 22 different test subjects with two types of gesture were recorded.

- In 6 zones (left, right, up, down, backwards, forward) measurements were done for a palm wave gesture under different conditions (normal speed, fast speed, left arm, right arm).
- Six gestures were done in the central position: **pinch**, thumb-up, **thumb-down**, cross, **tick**, palm tilt with different speeds and hand. After analysis it was decided to retain only the pinch, thumb-down and tick gestures.

Table 3.3.1 Chirp/Frame (a) and scene (b) radar parameters.

Start Frequency (GHz)	60.2
Slope (MHz/us)	60.0
Samples per chirp	280
Chirps per frame	128
Sampling rate (Msps)	5.47
Sweep bandwidth (GHz)	2.70
Frame period (msec)	90
Transmit antennas	3

(a)

Range resolution (cm)	4.46
Maximum unambiguous range (m)	11.3
Maximum radial velocity (m/s)	1.75
Radial velocity resolution (m/s)	0.0273
Azimuth resolution (Degrees)	14.5

(b)

Table 3.3.2 Recall and precision statistics of the machine learning based detector.

Percentage	Tick	Pinch	Thumb down	Left	Right	Up	Down	Forward	Backward	Unknown
Recall	87.3	74.4	78.7	96.1	93.6	94.8	96.8	93.6	89.4	83.2
Precision	87.3	84.2	76.6	80.3	81.0	88.5	86.0	83.0	90.8	89.7

Labeling the data took most effort, and unsupported or poorly executed gestures were labeled as unknown. For the palm-wave, some gestures were relabeled to another type if the test subject made the gesture in the wrong zone.

For training the model a 5-fold cross validation was used, doing 5 runs using 80% users for training and 20% for testing the model. To assess the performance, we look at detector statistics, timing, and real time inference performance.

Machine Learning Detector Statistics

Achieved detection accuracy is 86.1% with 13.8% misdetections. The detection rate is significantly impacted by using unknow data for input stimuli. If this data is not included, then detection performance increases to 92.8% with 7.2% misdetections.

Table 3.3.2 shows the achieved recall and precision statistics of the detected gestures. Recall shows the probability that a gesture is detected correctly, while precision indicates the percentage that a reported gesture is correct.

Machine Learning Real Time Inference

We use an Intel Core i7-8750H @ 2.20GHz based laptop to run all signal processing after ADC, feature extraction and classification in python on an Ubuntu 16.04 operating system. Critical parts are optimized in C or C++. While 12 cores are available, we use no explicit multi-threading. Real-time performance is achieved, and processing delay is less than 120 milliseconds.

3.3.3 Gesture Recognition Using a Spiking Neural Network

For the second implementation, a spiking neural network (SNN) approach was used for the radar-based hand gesture recognition (HGR). For this implementation, the same FMCW millimeter-wave radar was used. After

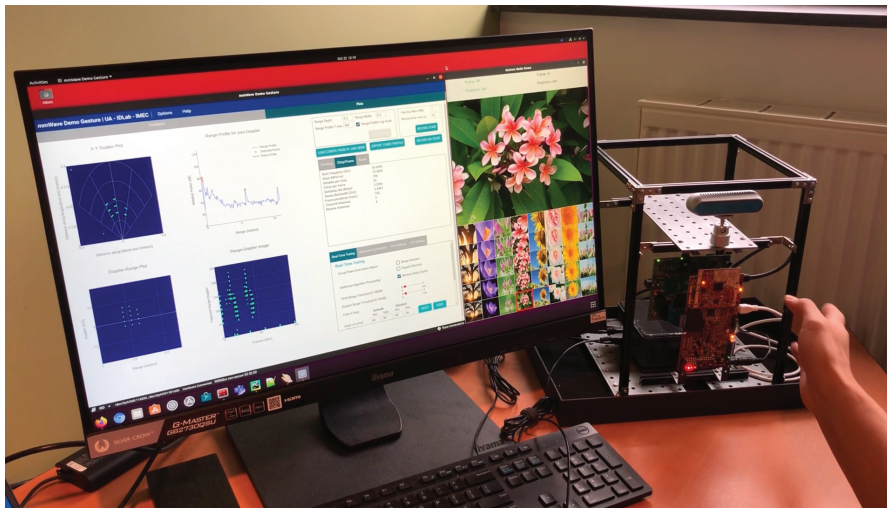


Figure 3.3.5 SNN-based gesture demonstrator.

pre-processing the range-Doppler radar signal, we use a signal-to-spike conversion scheme that encodes radar Doppler maps into spike trains. The spike trains are fed into a spiking recurrent neural network, a liquid state machine (LSM). The readout spike signal from the SNN is then used as input for a logistic regression which is used as a classifier in a supervised learning machine learning framework.

3.3.3.1 Concept and Experimental Setup

The proof-of-concept setup of the second implementation is shown in Figure 3.3.5. This implementation differs from the previous one in two ways. Firstly, the hand is now placed at a more or less fixed distance to the radar. No attempt is made to identify where the hand is positioned in space in front of the sensor. Secondly, the demonstration focuses on accurately identifying the gesture the person is making. The gesture vocabulary is “Swipe left”, “Swipe right”, “Zoom out” and “Zoom in”, allowing the user of the demonstrator, e.g., to navigate through a series of pictures and zoom in/out on each of them.

3.3.3.2 Inference Pipeline, Training Algorithm

The radar system collects the range-Doppler frames, representing the velocity and distance of the reflected object (i.e., hand). These frames were mapped as

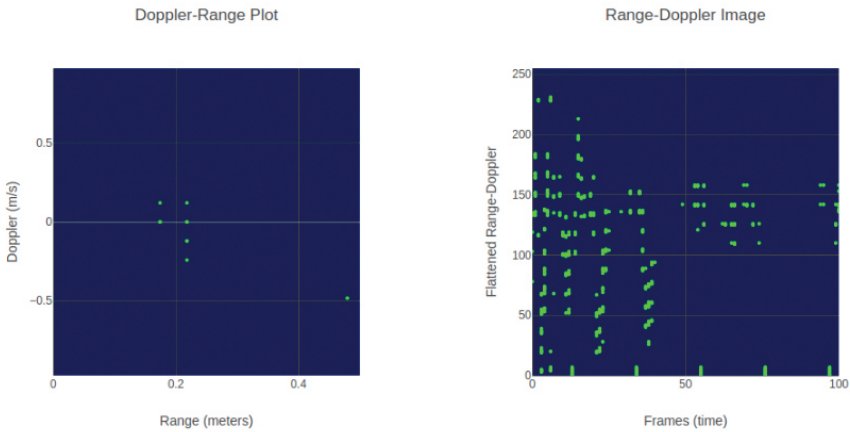


Figure 3.3.6 Range vs Doppler (left), flattened range-Doppler vs frames (right).

16×16 images (Figure 3.3.6 left), where a sequence of frames represents the complete gesture. Each gesture has different time durations, thus different amounts of frames. Each frame was unrolled and vertically stacked for the data representation, creating a frame versus flattened range-Doppler matrix (Figure 3.3.6 right).

The flattened range-Doppler can be seen as a unique pixel location of the 16×16 range-Doppler images. The frame vs. pixel representation captures the information in time, which is ideal for the signal-to-spike neural encoding to produce the spike train input for the LSM network.

The LSM is a type of reservoir computer capable of universal function approximation [7]. The basic formulation of LSM maps an input function $u(\cdot)$ onto a filter, or liquid neurons, L^M while the output $x^M(t) = \left(L^M u\right)(t)$ is fed to a second component, a readout map f^M, which is task-specific and generates the output $y(t) = f^M\left(x^M(t)\right)$.

The readout maps in our context will be a classifier that receives a state as input. Different classifiers can be used for this second component, such as logistic regression, random forest, or support vector machine. For simplicity and ease of in hardware implementation, we focus on the logistic regression in these experiments.

Figure 3.3.7 shows the LSM and how it has been used to build an end-to-end system for gesture recognition.

The top part of Figure 3.3.7 depicts the timing and how each gesture is sampled in the LSM. T_{s0} and T_{s1} are the boundaries of the time interval

reserved for a gesture, wherein the spike train of a gesture can have a variable stimulus length duration.

After the end of the stimulus, a readout delay t_r determines the readout window interval, during which the state of the liquid is measured and stored or passed to the classifier, depending on whether it is used in a real-time online or offline learning and inference system.

When mapping to the LSM, each sample had a different stimulus length. As a result, the readout window varies according to the sample frame duration.

The conversion from spike at pixel position i, of frame n to spike $s_i\,(t)$ at time t, is a direct map from n to t, i.e., if frame n has a spike at pixel i, then $s_i\,(t)$ has a spike at $t = n$. An alternative way to map the LSM was to normalize the frames to a predefined fixed stimulus length, whereas all the samples have the same readout window duration.

For every pixel position i, we convert the spike in frame n to a relative time regarding a fixed stimulus length S_l . Thus, the spike train sequence is given by:

$$s_i(t) = \frac{f_n * S_l}{f_l}$$

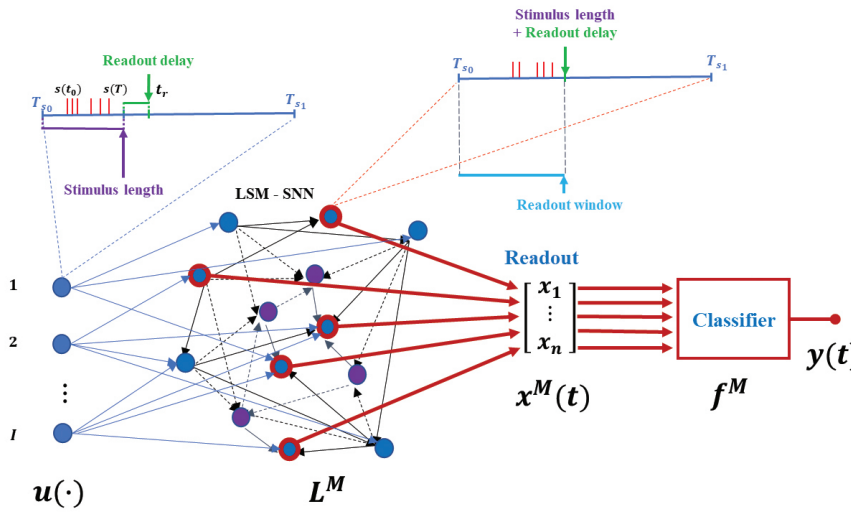


Figure 3.3.7 LSM network with trainable output layer.

Where:

S_l= predefined fixed stimulus length;

f_n= frame number,n that contains a spike;

f_l= length of the particular sample in number of frames.

In constructing the LSM, we focus on achieving the most compact and simpler to implement network without sacrificing accuracy. Each pixel i will produce a spike train as an input to the LSM, and each input is randomly connected to C_{inp} excitatory neurons.

All excitatory neurons are used for readout. For the neuron unit, we used a leaky integrate-and-fire neuron model with exponential postsynaptic currents with the associated synaptic model, based on [8].

3.3.3.3 Data Recording and Results

The SNN approach was based on a supervised framework. First, we collect a dataset of gestures for the learning phase.

Table 3.3.3 details the radar parameter used for the radar [3] in this demo setup. Notice that while it was configured for 32 chirps per frame, we rescaled to a 16 x 16 range-Doppler image to compose the frame versus pixel representation, reducing to a total of 256-pixel channels as input to the LSM.

The range depth, width and resolution were configured to around 0.5 m, thus only the reflected signal directly in front of the radar receivers were captured for the range-Doppler frames.

Table 3.3.3 Chirp/Frame (a) and scene (b) radar parameters.

Start Frequency (GHz)	60.0
Slope (MHz/us)	70.0
Samples per chirp	256
Chirps per frame	32.0
Sampling rate (Msps)	5.21
Sweep bandwidth (GHz)	3.44
Frame period (msec)	100
Transmit antennas	2

(a)

Range resolution (cm)	4.36
Maximum unambiguous range (m)	8.93
Maximum radial velocity (m/s)	0.974
Radial velocity resolution (m/s)	0.122
Azimuth resolution (Degrees)	14.5

(b)

Ideally, the more diverse and generic the learning dataset, the better generalization can be achieved by the machine learning framework.

Conversely, a personalized dataset can be used to tune the system for a specific user. In this case, we have collected four gestures from a single person. Collecting data from multiple persons will be done in the next phase.

Each gesture was collected in 30 separate sessions, wherein each session, a gesture was repeated 15 times.

The learning set contained then 450 samples for each of the four gestures. Figure 3.3.8 shows the confusion matrix of a 90%-10% learning (1620 samples) and test (180 samples) split of the dataset.

The LSM consisted of 600 neurons and a normalized stimulus length $S_l = 20$.

Table 3.3.4 summarizes recall and precision statistics of the SNN based detector, for the confusion matrix in Figure 3.3.8.

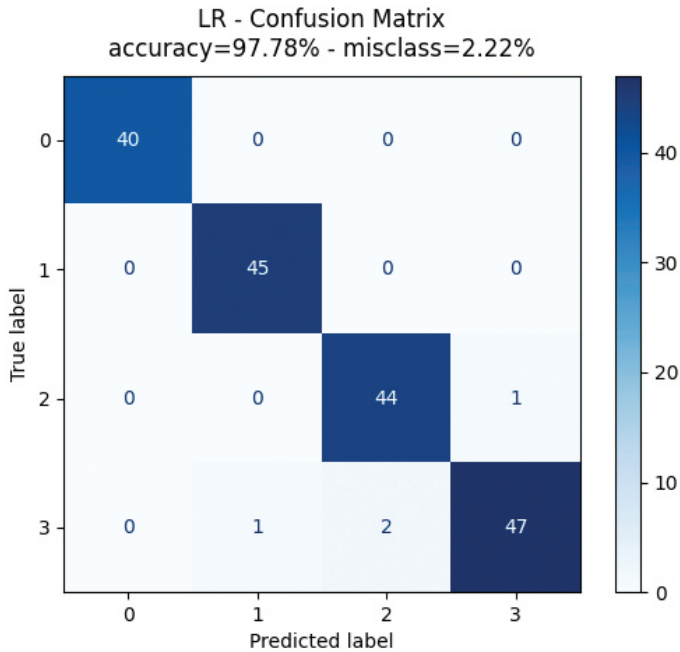


Figure 3.3.8 Confusion matrix for a 90%-10% learning (1620 samples) and test (180 samples) split of the dataset.

Table 3.3.4 Recall and precision statistics of the SNN based detector.

Percentage	Zoom out (label 0)	Zoom in (label 1)	Swipe left (label 2)	Swipe right (label 3)
Recall	100	100	97.8	94.0
Precision	100	97.8	95.7	97.9

3.3.3.4 Discussion

The setup was based on an Intel NUC Core i7-10710U (12MB Cache, 1.10GHz) with 32 GB DDR4 RAM. A complete capture, classification, and image cursor movement took between 0.5 to 1 sec for inference. The recognition rates reflect the confusion matrix shown in Figure 3.3.8, depending on the hand's relative position to the radar. The learning phase is relatively fast as the SNN was designed to be compact and efficient, considering the possibility of being deployed on an embedded system. The learning process was performed when launching the program, and uses a few minutes. The collection of the learning dataset was the most time-consuming element.

Dataset personalization shows that the system can be tuned specifically to the user operating the robot or other device to be controlled. Generalization to many users or a generic user base depends on the learning dataset. Moreover, extending the dataset to recognize more gestures can be easily done. In both cases, other classification schemes, such as support vector machine (SVM) or random forest, can be applied and integrated straightforwardly into the system. The spiking neural network algorithms were also validated on public data sets [5], achieving a gesture detection accuracy of 98%, which is better than the published state of the art. Still there is a misclassification chance of 2%. This restricts gesture input to non-safety-critical applications.

3.3.4 Proof of Concept Demonstration

We envisioned a proof-of concept demonstrator with five gestures to control the position of a robot arm:

- The robot arm positions a camera at discrete locations around an object. The robot arm can be moved to the left or to the right of the object, following a pre-defined trajectory. Also, the camera can be tilted up and down.
- A picture can be made of a 3D object and stored for post processing.

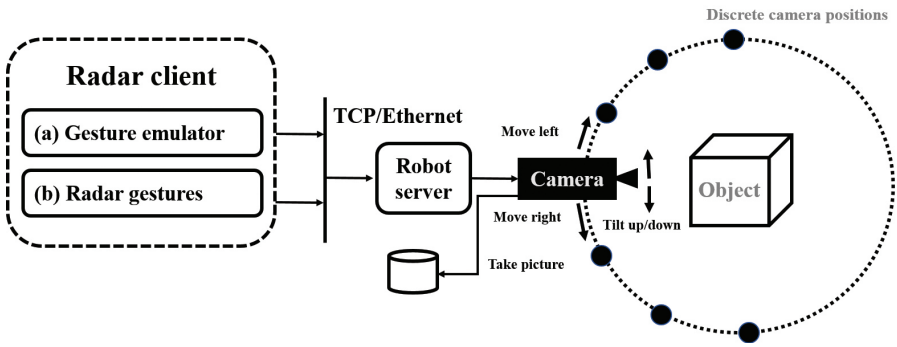


Figure 3.3.9 Proof of concept radar/robot interface block diagram.

These interactions are shown in Figure 3.3.9. The gesture-detecting radar system and the robot arm will interface over TCP/IP via an Ethernet link. A radar client collects gestures and transmits them to a robot server, which converts the gestures to robot commands to manipulate the camera location/position. There is a risk that a gesture is misclassified, resulting in a faulty recall. Such errors need to be corrected by the operator. This can be done without an increased safety risk. We still need to decide on the process to take the pictures and to handle the associated risks for sub-optimal captured data.

The radar client can either be emulated (a) or use a radar part (b) as shown in Figure 3.3.9. Either one of these modes is used. In emulation mode, it is sufficient to model the recall statistics of the different gestures together with their latency. This allows pre-validation of the robot part of the use case, without requiring a radar part or classification. For demonstration the emulation part is replaced by a suitable radar and classification. The TCP/IP communication scheme stays identical.

For the radar machine learning approach, sufficient gestures are available to support this use case. A logical choice is to use gestures with the best recall statistics. At least one gesture needs to be added to the spiking neural network if this approach is chosen.

3.3.5 Comparison and Conclusion

Both implementations (the machine learning and the spiking neural network) successfully detect gestures using a single radar. We observe that the SNN implementation achieves a better detection performance of 97.8%. The main reason is that larger training sets are used for a single user. For the

machine learning implementation, we observe that the classifier sometimes generates valid gestures for unseen data. Excluding this (for fair comparison), the detection performance improves from 86.1% to 92.8%. Although the obtained detection performances are rather high, the current implementations are not yet suited for safety critical applications. Both approaches require some time to detect a gesture, which may exceed half a second. This enables discrete control of a robot but precludes real time control. We envision a proof-of-concept demonstrator to illustrate the interaction between the radar system and the robot arm. This system will control the position/location of a camara mounted on the robot arm based on hand gestures detected by the radar system. A statistical model of the radar system is being created to allow early evaluation. This model combines gesture recall statistics with latency characteristics. The proof-of-concept demonstrator will be fully developed within the frame of the AI4DI project together with other consortium partners.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] Dimitrievski, M., Shopovska, I., Van Hamme, D., Veelaert, P., & Philips, W. (2020). Weakly supervised deep learning method for vulnerable road user detection in FMCW radar. In 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), Proceedings. Rhodes, Greece. https://doi.org/10.1109/ITSC45102.2020.9294399

[2] Wang, S., Song, J., Lien, J., Poupyrev, I.., Hilliges, O. (2016). Interacting with Soli: Exploring Fine-Grained Dynamic Gesture Recognition in the Radio-Frequency Spectrum. In Proceedings of the 29th Annual Symposium on User Interface Software and Technology, Tokyo, Japan, 16–19 October 2016; Association for Computing Machinery: New York, NY, USA, 2016; pp. 851–860.

[3] Texas Instruments, “IWR6843 intelligent mmWave overhead detection sensor (ODS) antenna plug-in module”, https://www.ti.com/tool/IWR6843ISK-ODS.

[4] A. Gorji, A., Khalid, H. U. R., Bourdoux A., and Sahli, H. (2021). “On the Generalization and Reliability of Single Radar-Based Human Activity Recognition,” in IEEE Access, vol. 9, pp. 85334-85349, 2021. https://doi.org/10.1109/ACCESS.2021.3088452

[5] Tsang, I.J., Corradi, F., Sifalakis, M., Van Leekwijck, W., Latré, S. (2021). Radar-Based Hand Gesture Recognition Using Spiking Neural Networks. Electronics 2021, 10, 1405. https://doi.org/10.3390/electronics10121405

[6] Texas Instruments, “DCA1000EVM: Real-time data-capture adapter for radar sensing evaluation module”, https://www.ti.com/tool/DCA1000EVM.

[7] Maass, W., Natschläger, T., Markram, H. (2002). Real-Time Computing Without Stable States: A New Framework for Neural Computation Based on Perturbations. Neural Comput. 2002, 14, 2531–2560.

[8] Tsodyks, T., Uziel, A., Markram, H. (2020). Synchrony Generation in Recurrent Networks with Frequency-Dependent Synapses. J. Neurosci. 2000, 20, RC50

3.4

Touch Identification on Sensitive Robot Skin Using Time Domain Reflectometry and Machine Learning Methods

Pawel Kostka, Anja Winkler, Adnan Haidar, Muhammad Ghufran Khan, Rene Jäkel, Peter Winkler and Ralph Müller-Pfefferkorn

Technische Universität Dresden, Germany

Abstract

The article presents the proof of concept of a novel sensor system for robotic HMI applications, mimicking the human sense of touch. An artificial sensitive skin, consisting of a robust and simple part of the sensing hardware based on electrical TDR, is mounted on the robot. In combination with adaptive AI algorithms, it enables for localisation of touch events on the sensor surface as well as determination of the touch-force magnitudes. Sensor data, obtained from a robotised test stand, are utilised to train and validate regressive DNNs for touch position recognition and classification DNNs for discrete force level classification. The results demonstrate that a high level of accuracy can be obtained, but some additional work is needed to reduce the gap between training and validation accuracy.

Keywords: human machine interaction, sensitive robot skin, touch control, collision detection, sensor development, artificial intelligence methods, artificial neural networks, deep learning, machine learning, training and validation, electrical time domain reflectometry.

3.4.1 Introduction and Background

Robot-based processes have been indispensable in many branches of industry for a long time. In addition to the typical application in autonomous production lines, an increasing trend to use robot co-workers in interaction with humans is currently recognizable. The robot assistance aims at relieving the strain on physically strenuous, repetitive or particularly precise work steps, enabling a significant improvement of working conditions, accelerating of workflows, and enhancing the product quality.

While the kinematic, dynamic, and performance characteristics of today's robots are suitable for supporting a wide range of human activities, the biggest challenge remains an appropriate control of the robots. Working hand in hand between a person and a robot requires a high degree of compatibility not only in terms of motor skills, but also in terms of communication capabilities. This work addresses the communication-related aspect of the human-machine interaction (HMI). It presents some ideas and development steps of an artificial sensitive skin that – in combination with suitable AI algorithms – enables a kind of tactile sense for robots. The goal is, on the one hand, to provide the interacting human with a communication channel for issuing commands through simple touches. On the other hand, the robot should be able to recognize its environment and react accordingly, e.g. stop in case of a collision.

3.4.2 State of the Art

Several projects use sensor systems based on the well-known capacitive measurement principle to detect the approach and contact between humans and objects with spatial resolution [1]. Furthermore, optical systems based on Bragg grating sensors [2] or on the measurement of electrical resistance changes [3] are often used to fulfill the same function. New sensors are under development that originate from the field of elastic circuits. Such sensors consist of multilayer micro channels in an elastomer matrix, which is filled with a conductive liquid to detect multi-axial strains and contact pressures [4]. Other scientific projects are analyzing the robot cell by multiple high-resolution cameras that capture images from different directions and continually create a three-dimensional representation of the scene [5]. Recent advances in environmental modeling and navigation are in many ways connected to the developments of high-precision laser or ultrasound scanning systems [6], [7]. Such scanners are based on the time-of-flight principle, in

which a transmitted light or sound pulse is reflected by an obstacle and the echo is detected by the receiver.

Identification of user interactions and associated intentions is an important task that is solved by interpreting raw sensor signals. In the field of robot HMI, solutions for collision detection based on signals from joint force sensors of smaller robots are known. The used algorithms range from analytic or empirical approaches to the use of AI methods such as artificial neural networks (ANN) and deep learning training algorithms [8], often referred to as machine learning (ML). The goal is to detect collision events with relatively low forces under constant presence of variable process forces. In this context, the proposed large area touch sensor represents an input device that outputs signals containing an implicit information about the contact position and force. In literature, similar applications are mentioned where AI methods are used for information extraction from sensor signals. An example is the use of ANNs to detect touch position and force in multi-channel piezo-based touch panels with intrinsic channel crosstalk [9]. Other works focus on AI-based identification of more abstract features of the HMI with the goal of implementing a running user authentication [10].

3.4.3 Problem Definition

High development and integration costs of the above mentioned sensor systems, often coupled with inherent drawbacks such as dead zones (laser scanning or ultrasonic systems) still prevent the widespread use of sophisticated HMI concepts. Thus, the presented work addresses the development of a touch sensor based on electrical time domain reflectometry (TDR). TDR is a well-established measurement method that enables a spatially resolved measurement of the electrical properties of a transmission line based on propagation times and reflection characteristics of electrical signals fed in at the beginning of the line [11]. The underlying idea for the proposed touch sensor principle arises from the observation that physical deformations of an elastic transmission line can cause significant local changes of its electrical impedance that are well-measurable by means of TDR. Such a solution promises several important advantages compared to conventional touch sensor principles. A single, standard shielded electrical connection is sufficient for interrogation of the sensor signal. The sensor structure is simple and inexpensive to manufacture, and it shows high mechanical robustness and electromagnetic compatibility, which is especially important under harsh industrial conditions.

The development of a functioning touch sensor according to the outlined principle comprises two main tasks. The first task focuses on the elaboration of an elastically compressible patch sensor with suitably designed and distributed transmission lines. The distribution of the lines and the elastic properties of the entire sensor structure should allow deformations related to the touch force over the entire range of expected HMI forces. Moreover, the deformations should be reliably detectable in the TDR signal, enabling the identification of both touch position and touch force.

The second task concerns the reconstruction of touch positions and forces from the TDR signals. The periodically triggered TDR measurement provides a vector of discrete values describing the impedance profile along the electromagnetic (EM) waveguide at each measurement. Because of the complex path of the waveguide, even simple contacts can produce multiple deformations. Due to the complexity of the wave phenomena and partly unknown system parameters, the determination of an empirical or analytical inverse model that converts the TDR vectors into contact positions and forces would be very challenging. As a possible solution, an AI based approach is developed, which achieves the preprocessing of TDR vectors by means of established signal analysis methods and an identification using ANNs.

3.4.4 Concepts and Methods

Figure 3.4.1 shows a generic application scenario of the focused touch sensor, where it represents a component of the robot's control loop. The combination of signal processing algorithms with an AI-based recognition of touch events and collisions enables a flexible and application-adapted behavior of the robot when interacting with humans.

The structure-installed touch sensor is a purely passive part of the system containing the compressible transmission lines. Once excited by the radio frequency (RF) generator, it responses with EM wave reflections that are analog-digital converted by the RF digitizer and carry information sufficient for:

- Detection of touches and collisions,
- Identification of the touch force magnitudes,
- Geometric localization of touch points on the sensor surface.

Achieving of these functionalities depends on the information content of the output signals, which in turn results from mechanical and geometrical properties of the sensor. The required elastic, electrical and dielectric

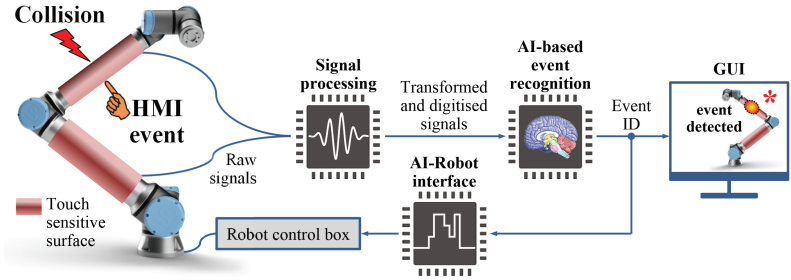


Figure 3.4.1 Overview of the HMI principle: Basic components of the sensing, computing and control of touch events.

properties of the constitutive materials as well as the layout of the transmission lines are determined in an iterative, model-based process. The developed multi-physical model features a time-domain simulation of TDR signals that takes into account the characteristics of the RF electronic modules.

The AI-based signal analysis allows the above-mentioned detection and localization of touch events as well as determination of the touch force magnitude. The applied ML concepts assume the supervised training of ANN based on labelled, experimentally acquired signal sequences.

3.4.5 Proof of Concept of the Novel Sensor System

In this section, the implementation and validation of the sensor system functions described in the chapter 4 is shown step by step.

3.4.5.1 Experimental Acquisition of Training Data

The acquisition of training data is typically an important challenge in the implementation of AI-based applications. For this purpose, an experimental approach has been designed to capture TDR vectors that result from artificial touch events occurring at different locations and force levels. A gantry robot, adapted for this purpose, automatically carries out test series in which a custom end-effector equipped with a soft, finger-like tip touches the sensor surface in a force-controlled manner. A specially developed software runs defined touch sequences whilst controlling the robot itself, triggering the TDR device, and storing the TDR data together with labels identifying the touch coordinates and forces as ground truth for the later learning stage.

The stored raw signal sequences are preprocessed (averaging and filtering) in order to reduce the noise content. A further pre-processing step is a resampling of the averaged and filtered TDR vectors in order to reduce the data dimensionality. The processed experimental data become training data by labeling them using the information about touch coordinates and applied forces. In the investigations carried out so far, different labeling schemes were applied, which enable the training of both, regressive deep neural networks (DNN) for continuous touch positions, and classification DNNs for discrete force levels.

The presented approach allows the acquisition of large experimental data sets, which are needed to obtain high quality training data. It would be impossible to get an appropriate amount of data by a manual approach. A further advantage is the high and reproducible precision of the generated touch events in terms of contact coordinates and forces.

3.4.5.2 Training Procedure

In the proof-of-concept phase, a TensorFlow-based training procedure is used on a data set generated from a thin and elastic surface sensor applied to a flat metallic component. The data set consists of 6380 TDR sequences, each containing 1000 values known as data set features and was labelled by six labels (0 N, 5 N, 6 N, 7 N, 8 N, 9 N) for the force identification task and two labels (x and y coordinates values) for the position identification task.

Before model training, the data was normalized, so that a distribution with a mean of zero and a standard deviation of one results. Then the data set is divided into three parts with 70% training, 15% validation, and 15% testing data, respectively. Once the data set is divided, a DNN model is trained to identify touch force and position. For the touch force identification task, two hidden layers are used with Relu [12] as an activation function. Furthermore, the Softmax [13] function is used in the output layer with the categorical-cross entropy-based loss function. The first hidden layer contains 128 neurons, and the second hidden layer contains 64 neurons. The network weights and biases are updated using stochastic gradient descent (SGD) [14] based backpropagation algorithm.

Position identification is a regression task due to the continuous nature of x and y coordinate values. Here, two hidden layers were used with Relu as an activation function. The two hidden layers contain 128 and 64 neurons, respectively. Moreover, in the output layer linear activation was used to predict the x and y coordinate values and the mean absolute error (MAE) [15]

is used for loss calculation. Here, a SGD based backpropagation algorithm is used for biases and weight optimization of the network also.

Due to intensive investigations in regard to different network architectures using different numbers of hidden layers and various numbers of neurons, the architectures described above have found to be appropriate to predict force as well as the position of occurring touch events.

The presented approach involves a single training process that bases on all available training data. Further development steps should include procedures for a continuous retraining and validation based on consecutively acquired user feedback, leading to the improvement of the sensor functionalities.

3.4.6 Results

Selected results of the force and position prediction models acting as a proof of concept are shown in Figure 3.4.2. There the results for force prediction, especially the model accuracy [16] during the training procedure is presented (Figure 3.4.2a). The accuracy is equal to the fraction of correctly predicted instances.

Using the mentioned approaches for force and position determination, an overall accuracy of 99.7% for training and 83.9% for validation are obtained. The considerable gap between training and validation accuracy indicates slight overfitting [17], which may be critical in a real-time human-machine collaboration application. Currently, experiments using dropout layers, batch normalization and the use of other deep learning architectures (temporal CNN [18], LSTM [19]) are under consideration in order to reduce the gap between training and validation accuracy.

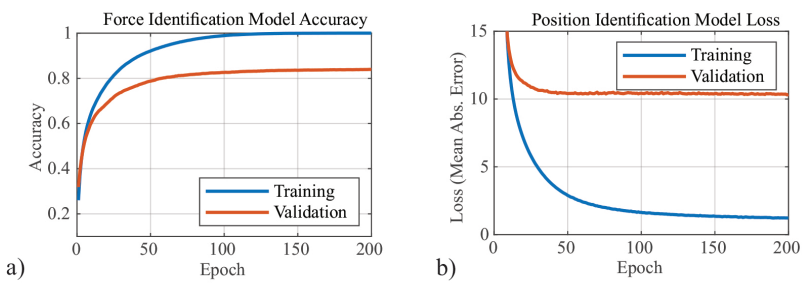


Figure 3.4.2 Network performance during the training: a) accuracy of the force identification and b) loss of the position identification.

Results for position identification are shown in (Figure 3.4.2b), where the MAE loss is plotted as a function of the training epoch. This graph is used as evaluation criteria rather than accuracy [16], because the discrete comparison between actual and predicted coordinate (x, y) values are not possible. It demonstrates that a significant decrease in both training and validation loss is achieved between 1 and 50 epochs. At higher training epochs, there is no significant decrease in validation loss, whereas training loss still show a significant decrease. The final difference in loss between actual and predicted validation cases is approximately 10.33 mm, which could be reduced by predicting the region rather than the specific coordinate position, which is currently under investigation.

3.4.7 Conclusions

The contribution reports on the concept, methods and early results of a large-area touch sensor for robotic HMI applications. A robust and simple part of the sensing hardware mounted on the robot enables in combination with adaptive AI algorithms the implementation of an artificial sensitive skin that mimics the human sense of touch. The results presented provide proof of concept of the novel sensor system through a basic training and validation of the touch position and force detection capability. This functionality can be extended depending on the application – for example by means of incremental learning – enabling a new quality of communication with collaborating robots.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060 and from the Federal Ministry of Education and Research and the Free State of Saxony under grant number 16ESE0341S. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] Goeger, D., Blankertz, M. and Woern, H. (2010). "A tactile proximity sensor," SENSORS, 2010 IEEE, 2010, pp. 589-594, doi: 10.1109/ICSENS.2010.5690450

[2] Park, Y.-L., Ryu, S. C., Black, R. J., Chau, K. K., Moslehi, B. and Cutkosky, M. R. (2009). "Exoskeletal Force-Sensing End-Effectors With Embedded Optical Fiber-Bragg-Grating Sensors". In: IEEE Trans. Robot. 25 (6), S. 1319–1331. DOI: 10.1109/TRO.2009.2032965.

[3] Fritzsche, M., Elkmann, N. and Schulenburg, E. (2011). „Tactile sensing: a key technology for safe physical human robot interaction". In Proceedings of the 6th international conference on Human-robot interaction (HRI '11). Association for Computing Machinery, New York, NY, USA, 139–140. DOI: https://doi.org/10.1145/1957656.1957700.

[4] Park, Y.-L., Chen, B.-R. and Wood, R. J. (2012). "Design and Fabrication of Soft Artificial Skin Using Embedded Microchannels and Liquid Conductors". In: IEEE Sensors J. 12 (8), S. 2711–2718. DOI: 10.1109/JSEN.2012.2200790.

[5] Kuhn, S. and Henrich, D. (2007) "Fast vision-based minimum distance determination between known and unkown objects," 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2186-2191, doi: 10.1109/IROS.2007.4399208

[6] Siciliano, B. and Khatib, O. (2008). Springer Handbook of Robotics. Berlin, Heidelberg: Springer Science+Business Media. http://site.ebrary.com/lib/alltitles/docDetail.action?docID=10284823.

[7] Dannemann, M., Holeczek, K., Modler, N., Winkler, A., Starke, E., Weiß, M. and Rupitsch, S. J. (2018). „Development of Material-Integrated Actuator-Sensor-Arrays for Obstacle Sensing". In: Adv. Eng. Mater., S. 1. DOI: 10.1002/adem.201800475.

[8] Heo, Y. J., Kim, D., Lee, W., Kim, H., Park, J. and Chung, W. K. (2019). "Collision Detection for Industrial Collaborative Robots: A Deep Learning Approach," in *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 740-746, doi: 10.1109/LRA.2019.2893400.

[9] Gao, S., Dai, Y., Kitsos, V., Wan, B. and Qu, X. (2019). "High Three-Dimensional Detection Accuracy in Piezoelectric-Based Touch Panel in Interactive Displays by Optimized Artificial Neural Networks," Sensors 2019, 19, 753. https://doi.org/10.3390/s19040753

[10] Silvelo, A., Garabato, D., Santoveña, R. and Dafonte, C. (2020). "A First Approach to Authentication Based on Artificial Intelligence for Touch-Screen Devices," Proceedings, 54, 1. https://doi.org/10.3390/proceedings2020054001

[11] Dominauskas, A., Heider, D. and Gillespie, J. W. (2007). „Electric time-domain reflectometry distributed flow sensor," Composites Part A: Applied Science and Manufacturing 38, 1, 138 - 146.

[12] Agarap, A.F. (2018). „Deep Learning using Rectified Linear Units (ReLU)," ArXiv, abs/1803.08375

[13] Prasad, P.S., Pathak, R., Gunjan, V.K. and Ramana Rao, H.V. (2020). "Deep Learning Based Representation for Face Recognition," In: Kumar A., Mozar S. (eds) ICCCE 2019. Lecture Notes in Electrical Engineering vol 570.Springer, Singapore. https://doi.org/10.1007/978-981-13-8715-9_50

[14] Cruciani, F., Vafeiadis, A., Nugent, C. et al (2020). "Feature learning for Human Activity Recognition using Convolutional Neural Networks," CCF Trans. Pervasive Comp. Interact. 2, 18–32. https://doi.org/10.1007/s42486-020-00026-2

[15] Sammut C. and Webb G.I. (2011). "Mean Absolute Error," Encyclopedia of Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-30164-8_525

[16] Behera, B., Kumaravelan G. and Kumar, P. (2019). "Performance Evaluation of Deep Learning Algorithms in Biomedical Document Classification," In: 2019 11th International Conference on Advanced Computing (ICoAC), pp. 220-224, doi: 10.1109/ICoAC48765.2019.246843.

[17] Garbin, C., Zhu, X. and Marques, O. (2020). "Dropout vs. batch normalization: an empirical study of their impact to deep learning," Multimed Tools Appl 79, 12777–12815, https://doi.org/10.1007/s11042-019-08453-9

[18] Pelletier, Ch., Webb, G.I. and Petitjean, F. (2019). "Temporal Convolutional Neural Network for the Classification of Satellite Image Time Series," Remote Sens. 11, no. 5: 523. https://doi.org/10.3390/rs11050523

[19] Kumar, J., Goomer, R. and Kumar Singh, A. (2018). "Long Short Term Memory Recurrent Neural Network (LSTM-RNN) Based Workload Forecasting Model For Cloud Datacenters," Procedia Computer Science, Volume 125 , Pages 676-682, ISSN 1877-0509.

Section 4

AI Food and Beverage

4.0

AI in Food and Beverage Industry

Rachel Ouvinha de Oliveira[1], **Marcello Coppola**[2] **and Ovidiu Vermesan**[3]

[1]Champagne Vranken-Pommery, France
[2]ST Microelectronics, France
[3]SINTEF AS, Norway

Abstract

This introductory article opens the section giving an overview of the state-of-the-art AI technologies in the food and beverage industry and the current AI development in areas such as quality optimisation and analytics and predictive maintenance. It also presents future potential and opportunities for AI in the sector, covering trends of using AI and IIoT technologies in production optimisation, safety/quality, maintenance, waste reduction, environmental sustainability, and packaging. Finally, the article introduces the five contributions to this section, providing highlights on the use of AI and IIoT in various scenarios in champagne production and soybean manufacturing processes and challenges and technological advancements.

Keywords: artificial intelligence (AI), industrial internet of things (IIoT), champagne production, soybean manufacturing, optimisation, predictive maintenance.

4.0.1 Introduction and Background

The food and beverage industry is undergoing a significant transformation to adopt new technologies, accelerate automation, increase efficiency, safety and avoid production disruption.

Artificial intelligence (AI) and the industrial internet of things (IIoT) enable digitising industries. The advancement in technology brings more intelligence at the edge that empowers IIoT devices with smarter decision-making, high performance, low power processing, and built-in security to create more intelligent and adaptive industrial applications.

The deployment of AI, IIoT and robotics solutions in the food and beverage industry has supported overcome significant issues related to production and execution by reducing the possible chance of human errors and by increasing the automation process while moving manual labour to specific tasks that are crucial for the quality of the final product.

AI and IIoT fuel change in food and beverage production and packaging to reach user expectations concerning quality and associated impact on the cost. To achieve the desired trade-off between quality and price, manufacturing stakeholders actively leverage AI and IIoT technologies' potential across various applications, like product design, quality control, maintenance, and user engagement.

4.0.2 AI Developments in Food and Beverage Industry

The integration of AI technology has transformed the productivity in the food and beverage industry, with increased efficiency, significant decreases in downtime, repair costs, and added labour requirements and costs.

Companies in the food and beverage production and manufacturing industry leverage AI's benefits by using AI methods such as neural networks (NNs), machine learning (ML) techniques, and advanced analytical tools, like speech and text analysis linked with computer vision and voice recognition technologies to optimise time and improve the overall user experience. The food manufacturing facilities use AI to automatically sort, clean, and dispose of products like fruits and vegetables. Manual labour is automated using cameras, sensors, and actuators integrated into autonomous machines. These improved monitoring abilities can reduce millions of tons of food waste. Food quality and safety are monitored using IIoT devices, supported by arrays of sensors, wireless devices, and edge technology, while AI-based food safety solutions help identify food risks in food products.

AI technologies monitor potential problems through various supply chain levels, supporting food manufacturing to become safer, healthier, and more efficient. Precise inventory management is a base of the food and beverages production and manufacturing industry, ensuring production lines

are stocked with the equipment, ingredients, and supplies necessary to run an effective and profitable business. AI helps remove the uncertainty from inventory management. Strategies like intelligent forecasting can utilise sales data, consumer behaviour, and seasonal information to predict how to keep warehouses stocked accurately.

In most food and beverage applications, AI and IIoT interpret data from sensors, detect patterns or anomalies and identify when action is needed. Sensors generate the data that is aggregated, classified, and significant data points are analysed using AI techniques. These technologies are used to detect anomalies, such as early warning signs that an asset may fail or require maintenance at food and beverage manufacturing facilities. AI technology is used to distinguish patterns, expand the knowledge base, recognise cause-and-effect relationships, use analytics insights related to likely outcomes or the next data point in the trend's curve.

Food and beverage manufacturing facilities are utilising capital-intensive machinery and improving and optimising the use of these machines; their energy consumption and efficiency are critical for staying competitive in the industry. The industry is an integrated chain of suppliers, vendors, utilities, labour, stakeholders, ancillaries and manufacturing, and the increase in efficiency in each part of the supply chain improves the overall productivity.

Predictive quality analytics and predictive maintenance are areas in the food and beverage industry where AI and IIoT are used to detect machine failures and anomalies, predict faults and abnormalities, redefine/define error classes and find factors that impede productivity.

IIoT devices and their digital twins provide benefits for predictive maintenance solutions in food and beverage processing and manufacturing combined with AI, including deep learning (DL) and NNs. Advanced and accurate detection of faults, predicting the remaining useful life of an asset given an operational context, can be simulated in an environment where accurate digital twin models of IIoT devices are used. The intelligent IIoT digital twin represent a continuously learning system that is updated automatically to mirror the changes and parameters of the physical IIoT devices. The digital twins can predict asset behaviour and deliver results within given parameters and cost constraints. The equipment is constant functioning, and the digital twins provide information about the physical processes to achieve the targeted outcome.

4.0.3 Future Trends for AI Technologies and Applications in Food and Beverage Industry

In the food and beverage market, AI has a value at USD 3.07 billion in 2020 and is foreseen to attain USD 29.94 billion by 2026 at a CAGR of over 45.77% during the period (2021 - 2026) [1][2][3]. Shifts in consumer needs by preferring fast, affordable, and easily accessible food options have led to a transformation in the food and beverage industry, with many companies leveraging advanced technologies, such as AI, ML, IIoT and robotics to scale operations and help corporations stay competitive in a dynamic market environment. The future trends indicate several areas in food of beverage that are impacted by AI, IIoT and automation, and provide opportunities for expanding AI technologies' development, increasing efficiency and profitability. The AI and IIoT technologies are focusing on addressing process optimisation, predictive maintenance, and production efficiency.

In the following paragraphs a short overview is provided covering the trends of AI, IIoT technologies and applications used in areas such as food and beverage production optimisation, safety/quality, hygiene, maintenance, waste reduction, environmental sustainability, and packaging.

Production Optimisation - AI and IIoT technologies have the most potential to optimise production and reveal manufacturing facilities' best operating points to meet and even exceed the production facility nominal performance.

The production optimisation allows to address all the productions issues related to the climate change introducing a more rigorous monitoring systems and more agile production changeovers, decreasing the amount of time needed to switch from one product to another and recognising production bottlenecks before they grow into a problem. IIoT devices, AI algorithms and actuators can be used together with AI trained models to calibrate production automatically, improving output quality and speed.

Safety and Quality - AI-based systems with the support of IIoT devices provide performant solutions for detecting safe and quality issues in production. These technologies deliver safer, more accurate production lines resulting in higher speed and more consistency than humans. AI-based detection on the factory floor has the potential to keep employees and equipment safer, identifying possible risks.

Hygiene - AI technologies have the potential for optimising the hygiene and cleaning tasks that are critical for food and beverage facilities by using self-optimising cleaning systems, where AI-based multi-sensor IIoT systems recognise food residue and microbial debris on equipment to determine the optimal length of cleaning time.

Maintenance - Food and beverage processing covers the whole value chain from planting and growing, harvesting, receiving materials to production, quality assurance and inspection, and the packing and dispatching of final products. In each step of the value chain, the processes happen in a particular environment (hot, cold, harsh, humid, etc.) that requires constant maintenance of equipment, storage, and workspaces. IIoT and AI, DL are applied to understand data, make predictions, and suggest recommended actions without explicit human guidance. Predictive maintenance brings benefits, including shortened maintenance time, streamlined equipment reconfiguration, avoid downtime, reduced failures, including maintenance costs. The AI-based maintenance in food and beverage includes production line sensors, equipment, motors, manufacturing assets and quality inspection controls to smart connections with electronic records and manufacturing execution systems (MES).

Waste Reduction - AI and IIoT are effectively used in optimisation and provide novel approaches to measuring and monitoring production input and output materials and significantly impacting waste reduction. AI analytics use IIoT real-time monitoring to identify anomalies in production outputs as soon as they occur concerning each batch or cycle and check the production quality.

Environmental Sustainability - The food and beverage process optimisation using AI and IIoT provides an indirect way of optimising energy and water consumption, creating immediate advantages for operating costs and margins while positively impacting the environment. The raw materials utilised as input to the production (e.g., fruit, grapes, vegetables, beans) differ significantly in size, shape, colour, moisture, and texture, adding a layer of complexity to the production line. Implementing AI-based computer vision and pattern recognition techniques combined with parameter measurements using sensors can easily recognise variances, removing contaminants without wasting whole batches and continually adjusting water and energy usage according to process requirements. The entire process operating 24-7,

including robotics and IIoT devices, can be fully automated using AI-based solutions across the production line.

Environment sustainability is achieved by reducing waste, pollution, carbon footprint and cutting electricity consumption using AI-based forecasting, alerts, and energy management tools using predictive ML algorithms to help facility managers to identify issues before they become problems, reducing costly downtime.

Packaging - Automation using AI-driven robotics, 3D cameras, IIoT devices is an area that is evolving fast for applications such as packing and picking demands for fast and efficient delivery. The food and beverage industry processes offer unique potential for intelligent automation by reducing complexity and automating the labour-intensive process, reducing cost, increasing efficiency, accuracy, and work at scale. AI is used in supply chain management through logistics, predictive analytics, and transparency. AI is used to analyse the supply chain data and better understand variables in the supply chain by anticipating future scenarios by reducing the time to market and establishing an agile supply chain capable of foreseeing and dealing with uncertainties.

The high cost of large-scale deployment of AI-based solutions in the food and beverage sector restricts the market growth, and the trend is to develop AI, IIoT technologies that are cost-effective, scalable, and energy-efficient and applied to several layers in the food and beverage supply chains.

Feedstock in the food processing industry can be increasingly made uniform, considering that the food storage is done with the help of AI-based automated solutions used in sorting, which can decrease the labour cost, increase speed, and improve yields.

4.0.4 AI-Based Applications

AI4DI partners are developing AI and IIoT technologies with applications in different areas of the food and beverage sector. The articles included in this section cover five demonstrators and actionable insights into how AI and IIoT are used in food and beverage applications, presenting challenges and technological advancements to accelerate the digitisation process across the industry.

The article "*Innovative Vineyards Environmental Monitoring System Using Deep Edge AI*" presents a novel environmental monitoring system, demonstrating how to connect science (AI) engineering (IIoT) and design

to improve the quality of products and increase the efficiency of their industrial processes by better tracking the production flow. IoT nodes provide real-time data related to weather, soil, crop water status and soil salinity. Connecting many sensors with different sensing technologies to each IIoT node allows for the generation of many and best-fit use cases in champagne production. Sensor data is accessed rapidly and at a relatively low cost by using LoRaWAN wireless technology. In the study, ML is deployed on IIoT nodes, and two architectural pattern solutions were investigated: one, where deep neural networks (DNNs) are executed on the end device with no AI on the cloud, and the other, where DNNs are implemented on both edge and cloud in a complementary manner. The results show that with proper hardware and automatic conversion of pre-trained NNs to fit within the limited resources, moving computation to the edge solves the business and power consumption constraints and addresses the privacy and security requirements.

The article "*AI-Driven Yield Estimation Using an Autonomous Robot for Data Acquisition*" explores automated and non-destructive methods for detection and counting grapes to overcome the drawbacks of the traditional techniques based on automated data acquisition and AI. The conventional techniques are both manual and destructive and have often been uncertain regarding the results' precision and repeatability /reproducibility. Most automated processes based mainly on the analysis of 2D images have drawbacks linked to detecting hidden grapes and estimating the number of berries. LiDAR combined with non-linear modelling can achieve better performances. The extra modelling step can determine hidden parts on the 2D images, such as grapes hidden by leaves. The LiDAR sensor installed on a vineyard robot and the image acquisition cameras used for grape detection transform the robot into a fully automated tool for yield forecasting.

The article "*AI-based Quality Control System at the Pressing Stages of the Champagne Production*" discusses computer vision algorithms/models to automatically classify grapes containers in terms of the average quality of contained grapes. The system detects grapes and unwanted elements (green or ripen grapes, leaves, stones, tools) for quality estimation before the delivery of the grapes to the press, as well as the challenges of deploying the trained models into the field, namely, on small edge devices with limited capabilities. The paper proposes using converters rather than rewriting the models in low-level languages to reduce the size and resources. Thus, trained models developed with high-end API (such as TensorFlow) can be deployed on various boards, allowing for exploring trade-offs between performances and inference time. A deep neural network with an encoder-decoder architecture

has been developed for this purpose. The architecture's performance is evaluated based on three parameters (inference time, the model's overall size, and the intersection over union score) and in three board configurations: without quantised, quantised without an accelerator, and quantised with an accelerator. The results obtained are promising, showing that it is possible to deploy the converted model in a real-time context while limiting the performance losses due to its conversion.

The article "*Optimisation of Soybean Manufacturing Process Using Real-time Artificial Intelligence of Things Technology*" presents a soybean process optimisation solution using real-time artificial intelligence of things (RT-AIoT) technology based on data collected from - and transmitted to different types of industrial IoT sensors, cameras, and actuators, using several wired and wireless protocols. Implementing intelligent vision locally on IIoT edge devices solves several issues faced by deploying it to the cloud and brings further challenges posed by deep learning on resource-constrained edge devices. Data is analysed using AI-based algorithms to improve the utilisation of the raw material, increase the yields and end-product quality, and optimise energy consumption reduction by supporting and/or replacing manual work and existing systems. The overall target is an analysis system that monitors the production line and offers information and analytics on production adjustments to preserve or increase the quality and utilisation. With multi-image sensors, IIoT devices under evaluation, the proposed production optimisation system is interfaced with the existing industrial SCADA system, processes and analyses the IIoT sensor data at different edge computing granularity levels. By applying analytics and AI-based approaches based on data, it is possible to obtain interpretive results for strategic decision making for process optimisation, cost reduction and energy-efficient process tuning.

The article "*AI and IIoT-based Predictive Maintenance System for Soybean Processing*" presents a creative and innovative approach to bringing artificial intelligence to edge devices with various levels of resources, demonstrated for an industrial soybean processing AI and IIoT-based predictive maintenance system. The system implements an architecture integrated at micro, deep and meta edge levels, based on a heterogeneous wireless sensor network that consists of sensor nodes and IIoT devices with different communication interfaces (BLE, LoRaWAN, Wi-Fi). This allows for exploring various combinations of computing power, sensing range, and AI-based processing capabilities to identify the parameter changes that occur before a failure and predict a future period in which these parameter changes

appear and thus identify when a failure might occur. The experimental results are promising, showing that it is possible to plan maintenance actions to reduce the number of production stops for single maintenance actions and thus minimise the downtime of the soybean production line.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] Mordor Intelligence (2021). Artificial Intelligence (AA) In Food & Beverages Market - Growth, Trends, COVID-19 Impact, and Forecasts (2021 - 2026). Available online at: https://www.mordorintelligence.com /industry-reports/artificial-intelligence-in-food-and-beverages-market

[2] Technavio report. (2021). "Global Artificial Intelligence (AI) Market in Food and Beverage (F&B) Industry 2017-2021". Online at: https://www.technavio.com/report/global-artificial-intelligence-market-in-food-and-beverage-industry?utm_source=pressrelease&utm_medium=bw&utm_campaign=t17_wk6&utm_content=IRTNTR15511&tnplus

[3] BlueWeave Consulting, (2020). "AI in Food & Beverage market is projected to reach US$ 12.58 Billion by the year 2026". Online at: https://www.blueweaveconsulting.com/ai-in-food-&-beverage-market

4.1

Innovative Vineyards Environmental Monitoring System Using Deep Edge AI

Marcello Coppola[1], Louis Noaille[2], Clément Pierlot[3], Rachel Ouvinha de Oliveira[3], Nathalie Gaveau[4], Marine Rondeau[4], Lucas Mohimont[4], Luiz Angelo Steffenel[4], Simone Sindaco[5] and Tullio Salmon[5]

[1]ST Microelectronics, France
[2]TechNext, France
[3]Groupe Vranken-Pommery, France
[4]University of Reims Champagne-Ardenne, France
[5]Alma Mater Studiorum - Università di Bologna, Italy

Abstract

With a turnover of more than 4.2 billion euros in 2020 and a 20% share in the value of the French wine industry's exports, the champagne industry represents a considerable weight in the French economy. In this context of significant economic development, the issue of climate change has been added, calling into question the practices and means of production of the sector. The challenges related to global warming and an ever-increasing demand for yield can be addressed using the Internet of Things (IoT) and Artificial Intelligence (AI) technologies to benefit champagne production and answer these challenges.

This article presents a solution to optimise Vranken Pommery products' quality and make environmentally friendly decisions by using intelligent sensors distributed as close as possible to the production and storage facilities to collect data. These sensors use LoRaWAN technology and protocol to communicate. The system integrates components capable of hosting artificial intelligence algorithms and using advanced microcontrollers that allow for intelligent communication network implementation while reducing power consumption and deployment costs.

Keywords: artificial intelligence, internet of things, AI-based microcontrollers, deep edge AI, LoRaWAN, vineyards, champagne.

4.1.1 Introduction

The 21st century has brought a digital transformation in the industrial sector in which the boundaries between the physical and digital worlds are blurring to create what we called Industry 4.0. Industry 4.0 will be the place where employees, machines and products interact, bringing a new set of technologies to enable the Internet of Things (IoT) and, more specifically, the Industrial Internet of Things (IIoT).

Industry 4.0 began in manufacturing but has become essential for all industrial markets such as the food and beverage markets. Like any business, those within food and beverage manufacturing, such as Champagne manufacturers, must respond quickly and effectively to change to keep up with competitors. Industry 4.0 applied to Champagne is a challenge since today the work in vineyards in Champagne still involves many manual tasks such as counting grape berries for yield forecasting or visual inspection of vines for disease detection. These tasks are essential because the quality of Champagne naturally depends on the quality of the raw material, i.e., grapes. In addition to the agricultural imperatives, the Champagne is the result of a long and rigorous industrial manufacturing process, as shown in Figure 4.1.1. This process starts with the pressing and the first fermentation, continues with the assembly and the second fermentation, the ageing in the cellar, and ends with bottling and sending to the end customer.

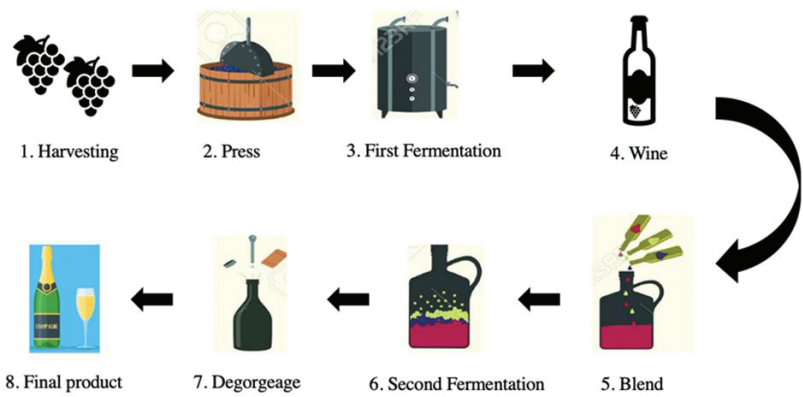


Figure 4.1.1 Value chain for champagne production.

Smart manufacturing leveraging on IoT and Cyber-Physical Systems (CPS) enables different physical sensors, actuators, and controllers to be locally interconnected and globally connected to cloud computing servers, forming complex online systems.

The use of IIoT can have an impact all along the manufacturing process of champagne (and more generally of wine). Indeed, thanks to sensors distributed in vineyards, it is possible to collect numerous data such as humidity, temperature or soil parameters: moisture, temperature, and electrical conductivity. The analysis of these data helps winemakers better managing and controlling the growth of their cultures. Besides, with the help of AI, specialised analytics allow growers to continually monitor soil, plant, and atmosphere to adjust irrigation and fertilisation in response to the environment. For example, by comparing current data with historical ones, the creation of predictive models on the best harvest period is now a reality. Furthermore, beyond the vineyard itself, IIoT can be used in wine cellars to monitor the ageing and the conservation of the champagne. Temperature is particularly important as even slight fluctuations impact the oxidation of the wine, which strongly affects the quality. Thanks to the IIoT, vintners are able to understand when tiny fluctuations occur and correct them before any damage is done. Thus, IIoT can help winemakers to achieve more successful harvests, better control production, and ensure ideal quality during transit and storage.

With these ideas in mind, this article presents a new environmental monitoring system enhanced by AI for yield forecasting, disease detection, fertiliser/pesticide optimisation, quality estimation, etc. This document aims to explain how the solution works, from the communication part to the intelligence part, and give insights on how this solution will help champagne manufacturers.

The article is structured as follows. Section 4.1.2 describes the current state of the art. Section 4.1.3 introduces the edge intelligence concept. Section 4.1.4 describes the LoRaWAN system architecture. Section 4.1.5 presents the monitoring system along with the architecture of the end nodes enhanced by AI. Section 4.1.6 concludes the work.

4.1.2 Related Work

Agriculture is seeing fundamental changes due to IoT and AI. In today's global warming environment and growing demographics, connected objects and artificial intelligence are an advantage. Their use allows farmers to

manage their farms better. Collecting data on the state of crops, weather forecasts, or even parameters such as temperature or humidity is at the heart of the intelligent farming concept.

The main contribution of AI and IIoT in agriculture is helping the industry players make decisions, allowing them to optimise their production and, therefore, their yield. For example, Farmwave [1] will enable farmers in the decision-making process concerning their farms. Using vision-based algorithms and edge AI, this solution can identify pest damage and disease through photos. Plantix [2] is also a solution to help farmers and agricultural workers increase their productivity. Thanks to a mobile application, farmers can take pictures of their crops and get information about them. Plantix can diagnose infected crops and diseases and propose appropriate treatments.

Unlike solutions such as Farmwave or Plantix, which rely on images, some use data collection and AI to provide models and predictions to help farmers know how to optimise the productivity of their crops. This is the case of Cropx [3], a solution that measures moisture, temperature, and electrical conductivity in the soil. Cropx helps farmers monitor their crops and ensure increased productivity by providing crop-specific recommendations. Thanks to AI, Cropx uses crop models to learn and understand the behaviour of its supported crops, depending on the region. Cropx also provides aerial imagery, topography maps, and soil mapping to help the farmer in the decision-making process.

Another example could be Microstrain [8] which is a wireless environmental detection system that monitors vineyards' key growth episodes. Information such as soil and leaf moisture, solar radiation and temperature are collected and merged to monitor vineyards remotely and alert growers to critical situations.

In addition to providing an answer to purely economical questions, AI and IoT are being used to provide solutions to more complex problems. Adapting production methods to climate change is, for instance, one of the challenges of smart agriculture. The solution aWhere [7] uses AI to give insights about the weather to help farmers, companies, governments, or agencies adapting to climate change. More than 1.9 million virtual weather stations are deployed to turn climate insights into action (as pest and disease modelling, fertiliser timing recommendations, optimal planting dates, etc.) and create powerful maps to monitor the weather in a specific area (global to local scale).

The issues raised by the concept of sustainable development also integrates a social dimension, and some solutions try to respond to this

problem. For example, PlantVillage Nuru [4] helps farmers from developing countries diagnose crop diseases, even without an internet connection. Developed with the UN FAO (Food and Agriculture Organization of the United Nations) and the CGIAR (Consultative Group on International Agricultural Research), Nuru is an AI assistant that can diagnose multiple diseases in Cassava, fall armyworm infections in African Maize, potato disease and wheat disease. An essential part of the PlantVillage Nuru solution is also the share of knowledge between smallholder farmers.

Many projects belonging to smart farming concept are based on servomechanism systems such as robots, drones, or satellites rather than scattered sensors. For example, Precisionhawk [5] is a solution based on drones, sensors, and AI. Drones collect high-quality data through sensors to survey, map, and image farmland. The results are then provided to a web application.

The Blue River Technology project [6] has developed robots that can accurately distinguish between "weeds" and cultivated plants using AI. Based on image processing algorithms, this solution allows farmers to limit spraying to weeds only, thereby reducing pesticide use.

Finally, Taranis [9] helps farmers monitor their fields. Using satellites, planes, and drones with vision-based AI, this solution allows workers to detect and prevent crop loss due to insects, crop disease and weeds. Data are assembled in reports, graphs, maps, or insights to make the decision-making process easier for the worker.

4.1.3 Edge Intelligence

AI has started to widen the application potentiality of IoT and CPS, enriching them with intelligent services used by many users. Deployment of standalone localised CPS such as the one offered by the ISA-95 model based on supervisory control and data acquisition (SCADA) system offers an inefficient solution due to resource wastage, prohibitive costs with the significant disadvantage of the distributed system nature of data itself. Thus, centralised approaches based on the cloud have tried to address these problems by combining data distribution and robust central services. A significant number of sensor data can be analysed and consolidated in synthetic format by modern dashboards In these approaches dashboards are updated in real-time or near real-time to understand the adequate status of manufacturing processes better.

Compared to the previous approaches, cloud-based solutions enable to monitor the actual working conditions of machines and analyse data to understand what is happening. When deviations occur, using this approach, it is possible to identify the reasons for variation compared to a standard procedure. This transparency implies the possibility of subsequent forecasting events and thus anticipating possible dangerous situations for the efficiency of the manufacturing production lines. To implement an efficient correct forecast, it is important to analyse a considerable amount of data collected during a long period. Then, applying AI with the most appropriate ML algorithms that model the behaviour of machines, it is possible to anticipate the future event of the machines and decide the most appropriate actions. For instance, depending on these events, it is possible to predict the time of preventive maintenance. Another advantage of these cloud-based approaches is implementing the digital twin of one machine or an entire manufacturing line, enabling without human controls to activate the most appropriate corrective actions within the manufacturing process. Cloud-based monitoring solutions allow for the improvement of the operative efficiency of a manufacturing line by decreasing machine downtime and reducing maintenance periods. The core of the cloud-based monitoring system is to have an efficient communication infrastructure for each machine and the overall manufacturing line. Such communication infrastructure must send efficiently data coming from the sensors towards the cloud. Cloud-based monitoring systems require smart sensors that include functionalities of communication and data signal processing.

Data signal processing is required to transform the physical monitored variable into something meaningful that can be transmitted to the cloud. For such reasons nowadays, such sensors include Micro Controller Unit (MCU), analogue and digital interfaces, memories, and communication hardware. The degree of smartness is related to its decentralised computation capabilities to perform operations that may include data from many probes connected to the same smart sensors. Considering the Moore law, it is possible to implement smart sensors with smaller and more powerful MCUs such as the STM32.

These MCUs can process data from several probes and apply algorithms more and more complex, including AI. A direct implication of this trend is that smart sensors are becoming the hub of many probes, thus reducing the costs associated with communication, processing, latency, and energy.

Communication costs can be reduced since data can be combined, so less data will be transmitted. Reducing data communication also implies a reduction of energy since most of the energy of the end node is consumed

during the data transmission. Time-critical applications imply real-time/near-real-time computation. These requirements cannot be met using the standard cloud approach due to the broad latency introduced by the network. With the increase of computation, it is now possible to move part of the computation from the cloud to the smart sensors: aka the edge nodes.

Moving computation to the edge, we also address privacy and security. Data privacy is guaranteed since the MCU can now decide the form of data to be transmitted to the cloud. Instead, the security will be reinforced leveraging the hardware security mechanisms provided by modern MCU. Several research papers focused on the possibility of bringing artificial intelligence to devices with limited resources [13][14][15][16]. To bring an AI model to MCU, ML developers should deal with the proper hardware, ML accelerator and memory set up to fit with the limited resources.

Therefore, to implement ML, two solutions may be used. The first one is called on-device computation, where Deep Neural Networks (DNNs) are executed on the end device with no AI on the cloud. The second is referred to as hierarchical computation, where DNNs are executed on the device and then on the cloud. In the second solution, the DDNs executed on the device and the cloud are complementary. Implementing an AI algorithm on MCU is challenging. And it is still a young technology.

As a result, engineers often must rely on a lot of different tools and complex workflows. For such reason, tools are essential. An example of a good tool that enables simple implementation of a DNN on a MCU is the X-CUBE-AI [17], suitable only for STMicroelectronics MCUs. It is an expansion of the STM32CubeMX environment that extends the tool's potential, allowing an automatic conversion of pre-trained NNs to low resource hardware. X-CUBE-AI also optimises libraries by modifying layers and reducing the number of weights to make the network more memory friendly.

4.1.4 Communication Technology – LoRaWAN

The numerous IoT applications impose constraints on the choice of the network architecture to be implemented. Depending on the use of a connected object, the organisation of the communication network will be different. To meet the required specifications and use cases, a network using IoT must find a compromise between the following four constraints:

- Range

- Data transmission rate
- Power consumption
- Cost of deployment

There are many different technologies available for this purpose.If the communication must be done over short distances (a few metres to a hundred metres), it is possible to use Wi-Fi, Bluetooth, RFID or Zigbee connectivity. These technologies allow sending data at a fast rate with reasonable energy consumption, but the communication can only be done at short range [10].

If the use case requires sending data over a hundred metres, then cellular connectivity technologies (2G, 3G, 4G or 5G) seem more appropriate. Cellular technologies allow for the transmission of large amounts of data over vast distances, which can be advantageous in the industrial sector.

However, there are IoT use cases where these technologies are not adapted. Indeed, these technologies are energy-intensive and have a high deployment cost. In some applications, such as in the field of connected agriculture or smart cities, the connected devices used need to transmit little data over large distances but are powered by simple batteries that do not provide much energy [10].

LPWAN (Low Power Wide Area Network) technologies are designed to transmit over large distances and maintain sound signal propagation even in more challenging environments. In an open environment, communication can be established over several tens of kilometres. In a more constrained environment (e.g., in urban areas), the range of LPWAN technologies is a few kilometres. LPWANs consume very little energy and allow devices to reach a lifetime of 10 years or more depending on the battery used. In addition, LPWANs allow covering a large area with few communicating devices. Indeed, the long range of LPWAN technologies and the network structure itself allows deploying fewer devices than cellular technologies while maintaining optimal efficiency.

Finally, since LPWANs do not have to handle complex waveforms (such as a voice call, for example), the transmit/receive module does not have to be very elaborate, which saves on hardware and production techniques.

Thus, the exponential growth of the IoT and the possibilities offered by LPWAN technologies are very interesting for enterprises.

The number of deployed connected objects (excluding phones, tablets, and computers) was indeed 7 billion in 2018 and is expected to reach 21.5 billion in 2025. Of all these devices, 25% belonged to LPWAN deployments [10].

Therefore, some companies have invested in establishing LPWAN networks and offer their own technology solution.

The Figure 4.1.2 summarises the characteristics of different communication technologies [11][12].

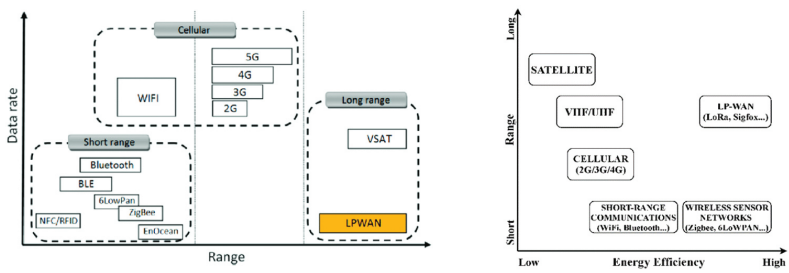
The LoRaWAN protocol was born under the impetus of the LoRa Alliance, which brings together various players in the IoT. It allows realising an LPWAN network that benefits from the advantages of LoRa technology while providing a solution to some IoT requirements, such as mobility and a large capacity of module connections.

The LoRaWAN uses a star-of-stars topology in which gateways relay messages between LoRa modules and a LoRa server. Figure 4.1.3 shows the overall architecture of a LoRaWAN network, which can be broken down into four parts.

The *End Nodes* part groups all the LoRa modules that communicate with the gateways. These are the ones that contain all the sensors necessary for data acquisition. They have a LoRa radio that allows them to send the collected data to all the gateways within the communication range. The data transmission is done using LoRa technology.

The *Concentrator/Gateway* part gathers all the gateways that have been deployed. They ensure the link between the connected devices and the LoRa server. They listen to all the communication channels. They convert LoRa frames into messages understandable by the server and vice versa. They can handle many LoRa modules, giving the LoRaWAN network a high load capacity.

The *Network Server* receives, via TCP/IP communication, the messages transmitted by the LoRa gateways. It also manages incoming and outgoing



a) Data rate/range comparison of communication technologies

(b) Range/energy efficiency comparison of communication technologies

Figure 4.1.2 Comparative range, data rate, energy efficiency characteristics of communications technologies [11][12].

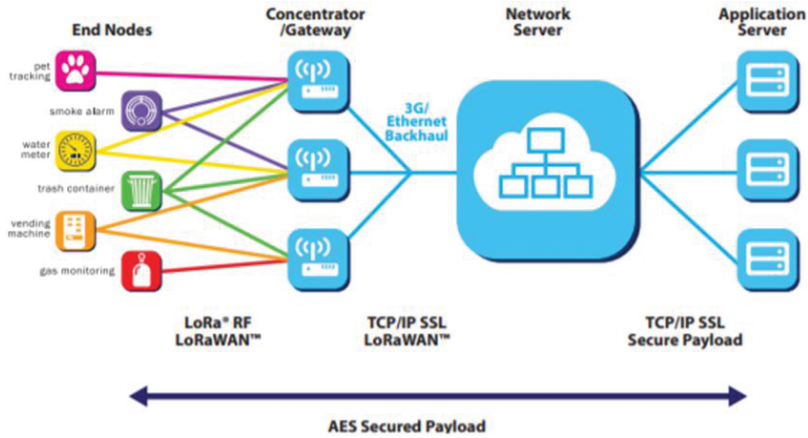


Figure 4.1.3 LoRaWAN architecture [19].

communications between the application part of the network and the gateways. For example, it will delete messages received in duplicate (several gateways can send the same data if they are in the range of the same LoRa node) and will take care of the authentication of data sent and received by LoRa nodes.

The *Application Server* takes care of the encryption and decryption of messages passing through the network. In most cases, the Application Server is followed by a Web Application part, grouping the web applications that will use the data collected by the LoRa modules. This part does not belong to the LoRaWAN protocol and is implemented by the user, but one of the roles of the Application Server is to dissociate the different web applications that want to connect to the network and transmit the instructions coming from them to the LoRa terminals.

Communication within a LoRaWAN network is bidirectional. It can be uplink (from the terminals to the server) or downlink (from the server to the endpoints). Most transmissions in a LoRaWAN network are uplink. It is also possible to realise a LoRaWAN network implementing only uplink connections to reduce the complexity of the network if the use case allows it.

In addition to the energy benefits of LoRa technology, the LoRaWAN protocol has implemented a class system to reduce network consumption. Thus, a LoRa module can be class A, B or C depending on its ability to communicate in the downlink as presented in Figure 4.1.4.

All LoRa devices must be able to implement class A. This mode is the least power consuming. At each transmission of the terminal, two reception windows are opened to receive downlink communications.

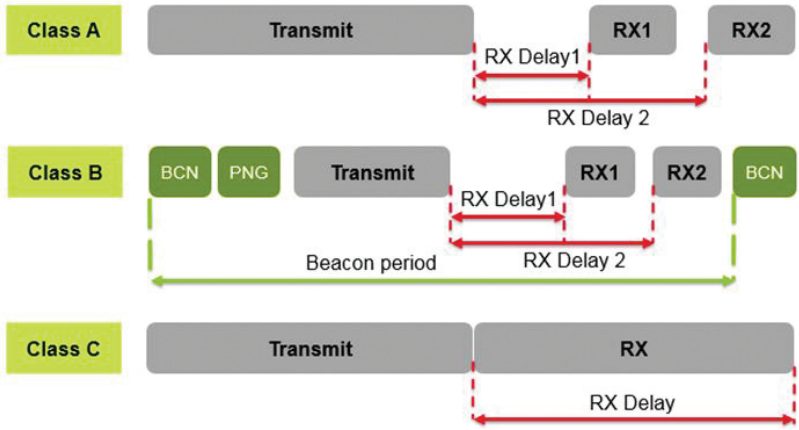


Figure 4.1.4 Operation of the different classes of a LoRaWAN [18].

These reception windows depend on a fixed duration, frequency, and data rate. If the device receives communication in the RX1 window, then the second window is not opened, and the device goes back to standby.

A downlink communication can only be done after an uplink transmission has been done. This mode consumes very little energy, as the device is mainly on standby but imposes a significant gateway/module communication latency.

Class B is a mode that seeks a compromise between energy consumption and downlink communication latency. It has the same operation as class A (2 reception windows after each transmission) and implements reception windows that open periodically. To allow synchronising the reception windows between the LoRa module and the concentrator, the concentrator must send a beacon and a ping. The LoRa device can therefore receive instructions without having first sent a message. This mode reduces the latency of downlink communications but increases the terminal's power consumption.

Class C is a mode adapted to specific LoRa modules. Indeed, in this mode, the terminal continuously listens to downlink communications, except when it transmits. Class C eliminates any latency in the transmission but is not energetically viable for a battery-powered device. It is, therefore, suitable for modules connected to the mains.

One of the major drawbacks of LoRa technology is the lack of means to secure communication. The LoRaWAN protocol offers a solution to overcome this problem. Any communicating object wanting to join a network

must be identified. To achieve this identification, it is necessary that the device is activated.

The security within a LoRaWAN network is ensured using three essential elements:

- Device Address (DevAddr): address of the device on the network, acts as an IP address
- Network Session Key (NwkSKey): AES128 key shared between the terminal and the Network Server, used for authentication
- Application Session Key (AppSKey): AES128 key shared between the terminal and the Application Server, used for data encryption
- Each module knows three elements necessary for its identification by the LoRa server:
- Device EUI (DevEUI): defines the device ID
- Application EUI (AppEUI): defines the ID of the application to which the device is attached
- Application Key (AppKey): a key that allows deriving the security keys

The transmission data rate depends on two parameters: the Spreading Factor (SF) and the Bandwidth (BW). The LoRaWAN protocol normalises the associations of these two parameters and names Data Rate (DR) an SF/BW pair. LoRaWAN lists seven DRs (from DR0 to DR6) for a LoRa modulation.

As the LoRaWAN protocol is based on LoRa technology, communication is carried out in the same frequency bands (from 863 MHz to 870 MHz in Europe). The LoRaWAN server defines several channels that can be used for uplink and downlink communications within this band. The LoRaWAN protocol requires the LoRa device to know the channels 868.1 MHz, 868.3 MHz, and 868.5 MHz from DR0 to DR5. LoRaWAN protocol also implements an algorithm named Adaptative Data Rate (ADR) that allows the Network Server to automatically calibrate the optimal DR for communication with the device, using *Signal to Noise Ratio* (SNR) and *Received Signal Strength Indication* (RSSI) [20].

Thus, LoRaWAN provides a suitable answer to most of the issues raised by the IoT. Its range of several kilometres, its energy efficiency and the robustness of its communications make LoRaWAN one of the most used solutions in the LPWAN market. The Table 4.1.1 summarises the characteristics of different LPWAN technologies.

Table 4.1.1 LPWAN technologies comparison.

	Sigfox	**LoRaWAN**	**NB-IOT**	**LTE-M**
Modulation	UNB, GFSK	CSS	QPSK	16QAM
Flow	100 bps *uplink* 600 bps *downlink*	0,25 to 50 kbps	100 kbps	1 Mbps
Range (open environment)	To 50 km	To 20 km	To 10 km	To 5 km
Cost	€	€€	€€€	€€€
Lifetime	More than 10 years	More than 10 years	To 10 years	Less than 10 years
Payload (Bytes)	12 *uplink* 8*downlink*	Up to 250	1600	More than 1000
Security	None	AES128	LTE	LTE
Quality of Service	None	Definable but complicated	Definable	Definable
Latency	*Downlink* communication limited	Depends on the class used	1 second	10 milliseconds
Mobility and localization	No	Yes	Limited mobility, no localization	Mobility, no localization
Deployment	Sigfox operator	Private operators and networks	Operators	Operators

4.1.5 Environmental Monitoring System

It is widely recognised that the digitalisation of French wine and champagne grape production can bring significant economic, environmental, and social benefits. The future of the Champagne and Wine sector implies an exponential increase to observe and monitor key aspects of production cost effectively. For a company like Vranken Pommery, the production starts at the vineyards and ends at the bottling. At each step, the data sources are diverse, spanning from simple environmental data to complex images. The environmental monitoring system manage the production operations and to reduce the waste by improving Vranken-Pommery operational efficiency.

Fungi cause the most common vine diseases. Different species can infect grapevines. Black rot (Guignardia bidwellii), Powdery mildew (Uncinula necator), and Grey mold (Botrytis cinerea) are examples of diseases that can affect grape quality. Each fungus develops under certain environmental conditions.

The environmental monitoring system is based on data collected by different industrial sensors (e.g., TEROS, STMicroelectronics, etc.) connected to STM32WL enhanced by a machine learning core enabling continuous monitoring of the environment, the soil, meteorological conditions, and/or plant performances. The STM32WL System-On-Chip integrates both a general purpose microcontroller and a sub-GHz radio on the same chip. Built on Arm® Cortex®-M4 and Cortex®-M0+ cores (single- and dual-core architectures available), STM32WL microcontrollers support multiple modulations- LoRa®, (G)FSK, (G)MSK, BPSK - to ensure flexibility in wireless applications with LoRaWAN®, Sigfox, W-MBUS, mioty® or any other suitable protocol in a fully open way. Sensors will be able to acquire and merge underground and climate data. Many sensors are today available on the market but in order to accurately understanding the percentage of water in a soil has been a complex, costly, and laborious process. Soil moisture is highly variable over short distances, at different depths in the soil profile, and in different soil types and densities. Today only few of sensors provide the right degree of precision and low percentage of sensor-to-sensor variability in their measurements. In the environmental monitoring system in order to meet the functional and not functional requirements provided by Vranken-Pommery for the soil moisture sensors, the TEROS12 sensor from METER Group has been selected since it provides sensor-to-sensor variability (less than 1%), at a reasonable cost. Thus, the TEROS12 sensors along with other types of sensors are used to make precise, informed decisions and better manage Vranken-Pommery, labour, equipment, and chemical usage. Technological advancements introduced by the STM32WL enables ML and efficient communication directly at the edge. To improve the power efficiency an innovative approach has been chosen: to enrich with a machine learning core to the STM32WL. The adopted solution give the possibility to implement ML directly to the STM32WL and/or to the machine learning core. The Machine Learning Core provided by the LSM6DSOX comprises a set of configurable parameters and decision trees able to implement AI algorithms in the sensor itself. The kinds of algorithms suitable for the Machine Learning Core can be implemented by following an inductive approach, which involves searching patterns from observations.

The idea behind the Machine Learning Core is to use the accelerometer, gyroscope, and external sensor data (readable through the I^2C master interface) to compute a set of statistical parameters selectable by the user (such as mean, variance, energy, peak, zero crossings, etc.) in a defined time

window. In addition to the sensor input data, some new inputs can be defined by applying some configurable filters available in the device.

The Machine Learning Core parameters are called "Features" and can be used as input for a configurable decision tree that can be stored in the device. The decision tree, which can be stored in the LSM6DSOX, is a binary tree composed of a series of nodes. A statistical parameter (feature) is evaluated against a threshold to establish the evolution in the next node and this in each node. When a leaf (one of the last nodes of the tree) is reached, the decision tree generates a readable result through a dedicated device register. Using this innovative architecture, we can target from 10 to 1000 times energy saving.

The environmental monitoring system exploits the range of State-of-the-art IoT sensor nodes and communication protocols to deliver data to Vranken Pommery to aid the decision-making process. As described above, the IoT sensor node provided includes different sensing technologies to provide real-time data related to weather, soil, crop water status, soil salinity. With the latest development of wireless communication technologies, sensor data can be accessed rapidly and at a relatively low cost, saving Pommery potentially significant amounts of time and money.

Since IoT sensor nodes are battery-powered, the right combination of low-power sensors and communication networks is imperative for the environmental monitoring system. In addition, the sensors used in this demo require low bandwidth due to the small size of the transmitted data packets. Thus, LPWANs are the best suited wireless communication protocols for this demo due to their low power consumption and long communication distance. LoRaWAN is one well-established protocol in the LPWAN family, it uses Long-Range (LoRa) modulation in its physical layer, and it is characterised by extended and significant coverage and low data rate with low complexity assuring optimal power consumption. Using LoRaWAN, a large volume of data from multiple sensor types installed in multiple vineyards of Vranken-Pommery are generated. Therefore a data management system composed of a distributed data system formed by the IIoT nodes previously described and a centralised data system collecting sensor data from the distributed data system and providing access to data via ad-hoc methods is required. This system aims to enable time-series data collection, processing, and storage. In order to have a user-friendly approach to managing the acquired data, it is also crucial to present and visualise data via a complete end-to-end infrastructure based on Grafana. Using Grafana, we can pull data from the database, allowing us to create customised and attractive charts and graphs. Dashboards

provide the real value of the monitoring parameters and use computational models and algorithms to translate data to useful information to Vranken-Pommery to make actionable decisions. The introduction of an efficient and scalable data management system allows managing larger datasets that may cover multiple Vranken-Pommery vineyards. Managing the collected datasets effectively makes it possible to exploit further prediction (AI) opportunities in the Cloud that are infeasible with smaller siloed datasets.

4.1.6 Conclusion

This article presents a monitoring system demonstrating how an AI-based energy-efficient IIoT solution using LoRaWAN connectivity can be used in Champagne production. The trends in moving computation from the cloud to the edge are summarised, and the implication of IIoT end nodes design and architecture is discussed. It is crucial to connect many sensors to each IIoT end node to give the flexibility to address several use cases in champagne production. We have also proposed to deploy machine learning on IIoT end nodes. The article described the way to enable the execution of machine learning models on hardware with low performances based on STM32 MCU to reduce the network data transmission by allowing computations to be performed close to the sensor data sources, preserving privacy in uploading data, and reducing power consumption for continuous wireless communication to cloud servers. Finally, the article describes the deployment of a system monitoring infrastructure based on LoRaWAN for the monitoring of environmental conditions within the vineyards of Vranken Pommery.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] Farmwave. Available online at: https://www.cadre-ai.com/agriculture
[2] Plantix. Available online at: https://plantix.net/en/
[3] Cropx. Available online at: https://cropx.com/

[4] PlantVillage Nuru. Available online at: https://plantvillage.psu.edu/
[5] Precisionhawk. Available online at: https://www.precisionhawk.com/agriculture
[6] Blue River Technology Project. Available online at: https://bluerivertechnology.com/
[7] aWhere. Available online at: https://www.awhere.com/about/
[8] Microstrain. Available online at: https://www.microstrain.com/applications/sensorcloud-enables-condition-based-agriculture-shelburne-vineyard
[9] Taranis. Available online at: https://taranis.ag/about-us/
[10] Raza, U., Kulkarni P. and Sooriyabandara, M.. (2017). "Low Power Wide Area Networks: An Overview," in IEEE Communications Surveys & Tutorials, vol. 19, no. 2, pp. 855-873, Second quarter 2017. https://doi:10.1109/COMST.2017.2652320
[11] Sanchez-Iborra R, G. Liaño I, Simoes C, Couñago E, Skarmeta AF. (2019). Tracking and Monitoring System Based on LoRa Technology for Lightweight Boats. Electronics. 2019; 8(1):15. https://doi.org/10.3390/electronics8010015
[12] Mekki, K.; Bajic, E.; Chaxel, F.; Meyer, F. A comparative study of LPWAN technologies for large-scale IoTdeployment. ICT Express 2019, Vol. 5, Issue 1, pp. 1–7. Available online at: https://www.sciencedirect.com/science/article/pii/S2405959517302953?via%3Dihub
[13] Lee D.D., Seung H.S. (1999). WOES'99: Proceedings of the Workshop on Embedded Systems on Workshop on Embedded Systems. USENIX Association; Berkeley, CA, USA: 1999. Learning in intelligent embedded systems; p. 9.
[14] Haigh K.Z., Mackay A.M., Cook M.R., Lin L.G. (2015). Machine Learning for Embedded Systems: A Case Study. BBN Technologies; Cambridge, MA, USA: 2015. Technical Report.
[15] Chen J., Ran X. (2019). Deep Learning with Edge Computing: A Review. Proc. IEEE. 2019; 107:1655–1674. https://doi:10.1109/JPROC.2019.2921977
[16] Sze V., Chen Y.H., Emer J., Suleiman A., Zhang Z. (2017). Hardware for machine learning: Challenges and opportunities; Proceedings of the 2017 IEEE Custom Integrated Circuits Conference (CICC); Austin, TX, USA. 30 April–3 May 2017; pp. 1–8.
[17] X-CUBE-AI—AI Expansion Pack for STM32CubeMX—STMicroelectronics. Available online at: https://www.st.com/en/embedded-software/x-cube-ai.html#overview

[18] Polonelli, T.; Brunelli, D.; Marzocchi, A.; Benini, L. (2019). Slotted ALOHA on LoRaWAN-Design, Analysis, and Deployment. Sensors 2019, 19, 838. https://doi.org/10.3390/s19040838
[19] LoRaWAN Architecture. Available online at: https://air.imag.fr/index.php/File:Network.png
[20] LoRaWAN. The Things Network. Available online at: https://www.thethingsnetwork.org/docs/lorawan/

4.2

AI-Driven Yield Estimation Using an Autonomous Robot for Data Acquisition

Lucas Mohimont[1], Luiz Angelo Steffenel[1], Mathias Roesler[1], Nathalie Gaveau[1], Marine Rondeau[1], François Alin[1], Clément Pierlot[2], Rachel Ouvinha de Oliveira[2] and Marcello Coppola[3]

[1]University of Reims Champagne-Ardenne, France
[2]Champagne Vranken-Pommery, France
[3]ST Microelectronics, France

Abstract

The quality of the harvest depends significantly on the quality of the grapes. Therefore, winemakers need to make the right decisions to obtain high-quality grapes. One of the first problems is estimating the yield of the crops. It allows winemakers to respect the specific norms of their appellation (yield quota, alcohol levels, etc.). It is also necessary to organise the logistics of the harvest (start date, human resources required, transportations, etc.).

Traditionally, the yield estimation is performed by collecting grapes and berries over small, randomised samples, a destructive and laborious task. This work explores how automatic data acquisition combined with artificial intelligence can drive an automated and non-destructive yield estimation, adapted to the characteristics of each vine parcel.

Keywords: yield estimation, precision viticulture, image segmentation, fruit counting, deep learning, LiDAR sensor, vine balance.

4.2.1 Introduction

Winemakers use yield estimation to get decisive information for the organisation of the harvest and the business's economy. Therefore, it is necessary to estimate the yield for the organisation of the harvest, whether in the field or at the wine press.

Today, most winemakers estimate their yield using Equation (4.2.1) on a given land parcel. Traditionally, the counting is visually performed by an operator, leading to uncertainties on the precision and repeatability; in addition, the weight of the grapes requires them to be harvested. Estimations using historical data are also used; however, important variations can skew the predictions. The number of grapes and their weight varies from year to year. Despite the method, a variation of 30% can be found between the estimations and the reality [2].

$$\text{Yield (in kg per hectare)} = \frac{\text{nb of vine plants} \times \text{nb of grapes} \times \text{average grape weight}}{\text{parcel's area}} \quad (4.2.1)$$

Artificial Intelligence (AI) allows a clear improvement of work conducted in businesses using decision aids. AI can be better than humans in repetitive, time consuming, and tedious tasks. For example, automating the counting of grapes and fruits, in general, is one of the central problems in precision agriculture.

Many methods have been proposed these past years. Some methods are based on a classic image analysis approach, which consists of developing algorithms for segmentation, shape recognition, and problem-specific feature extraction. This method has been applied for the detection of oranges [8] and peppers [12]. Another approach is based on deep learning and convolutional neural networks. This type of neural network can solve multiple tasks like classification, segmentation, or object detection by automatically learning the correct representations needed for the job. This approach requires a large quantity of raw data rather than subjective criteria and specialised algorithms developed by humans. Deep learning has been used a lot since 2012 [6] and is now state of the art for classifying and detecting objects and fruits [7].

This paper aims to summarise the different methods of fruit detection and counting applied to viticulture for yield estimation.

4.2.2 Artificial Intelligence for Grape Detection

Grape counting is one of the yield estimation methods used by winemakers today. Grapes are harvested among a random sample allowing an estimation of the number of grapes by vine, the number of berries per grape, and the weight of the berries. These three components make up for 60%, 30%, and 10% of the variability in the yield, respectively [1]. However, this method is destructive, hence limiting the number of samples. An alternative consists of using images to estimate the components to allow for the automation of the tasks and limiting the biases due to the perception of the human eye.

The detection algorithm must take an image containing grapes as input and output an image that includes the location of the grapes and their number. This task is potentially difficult for several reasons: (*i*) there are many sources of variations in images taken in natural conditions (lighting, distance, background), (*ii*) the leaves can hide some of the grapes, and (*iii*) the leaves are green and share a similar colour to the grapes before they ripen.

Several classic methods or using deep learning can be used for the detection of grapes. A first naive approach is to use a threshold. One or several thresholds are chosen and applied on each pixel to separate the areas of fruit from the rest of the image [4]. These algorithms are fast, but they have several limitations that render them difficult to use in the field unless the lighting is controlled. In rare cases, this technique can be employed in natural conditions, however, only in simple cases where the berries are ripe, of a black variety, and the vine has been trimmed [3].

A second approach uses a segmentation method with active contours. It has been used for the detection of white grapes for automatic harvesting [14]. However, it remains limited to being used at night with a controlled light source, which can erase the image background (sky, ground, and the vine rows).

A third approach uses classic machine learning to develop methods for grape detection that are more robust to the variations in natural lighting conditions. They perform a segmentation pixel by pixel using a pixel's neighbourhood, or block, as an input to the classification model. The model produces a binary output (grape or not grape) which is then applied to the central pixel or the entire block. These methods do not work with the raw image; the extraction of features is a necessary step before analysing each block. The average of the RGB channels of a block is an example of simple features. This method suffers from several limitations, including sensitivity to colour (grape variety) and a high execution time (potentially long).

Deep learning has been recently applied to help solve the problem of detecting and counting grapes. A naive approach consists of a block-by-block (or pixel by pixel) classification with a convolutional neural network. The usage of CNNs allows for simplifying the detection algorithm because the model will learn the appropriate features from the data. Several popular object detection models, Faster R-CNN, R-FCN, and SSD, have been applied to the problem of detecting grapes and counting them using videos [5]. In addition, the model Mask R-CNN, which allows for simultaneous object detection and object segmentation, has also been used [10].

4.2.3 Towards an Automated Protocol for Yield Estimation

The detection of grapes is the first step for automated yield estimation. It requires converting the counting into an assessment in kilograms per vine or kilograms per hectare. The benefits of image analysis are that it can rapidly process large quantities of data to avoid random selection and destructive methods in the field. However, most automated methods have drawbacks linked to detecting hidden grapes and the estimation of the number of berries using 2D images.

Since 2019, we have performed image collection campaigns on parcels of the Vranken-Pommery domain in Reims for the project H2020 AI4DI[15], using different cameras and methods, for example, with a GoPro fixed on a picket or embarked on a tractor (Figure 4.2.1a). Approximately 400 pictures have been taken in the 2019 campaign, from which 322 photos have been labelled to train the segmentation models. The model is a UNet encoder-decoder with a ResNet-34 backbone. At the end of the training process, the generated model has an IoU (Intersection over Union) score of 0.69 and an F1 score of 0.8. The IoU is limited due to the lack of precision in the labelling. However, the model allows detecting nearly 100% with a false positive rate near 0% (Figure 4.2.4). The main problem is that grapes on the background can also be counted, reducing the counting precision. Several filtering methods, including the suppression of areas that are too small or morphological openness, have been studied to control this problem. The model was then applied to 200 images taken in 2020, allowing for the total count of the number of grapes (hidden and visible) and the precise location of each visible grape.

In addition, [9] carried out a systematic tracking of several rows in the vineyard, allowing calibrate our deep learning algorithms for automated yield estimation. This tracking, performed over four rows (200 vines) at

Figure 4.2.1 (a) Camera attached to the vehicle, (b) defoliated vine.

different phenological stages, has included the counting of classic organs (vine, flowers/grapes) thanks to other strategies (random counting or by the sampling of the parcels) as well as sampling the berries to estimate their volume and ripeness.

The counting of the grapes has also been done by unveiling hidden grapes by defoliation. Hence, the operator first counts the visible grapes in the plant and, after defoliation, takes a second picture (Figure 4.2.1b). Around 30 images were taken in this way and were then labelled and used to help identify partially hidden grapes. An example of the comparison between automated and manual counting where hidden grapes have been exposed is illustrated in Figure 4.2.4.

Thanks to the manual and automated counting data, a linear regression model has been generated for each row then a cross-examination of each model is done using the three other rows. Although the error rate varies from 0% to 31%, depending on the model and the row, we obtain an average error rate of 14%. This is better than current error rates with the traditional approach but remains perfectible. Hence, the improvement of this analysis is based on a better distinction between grapes in the foreground and grapes in the background as well as using non-linear regression models and other variables such as the porosity of the canopy.

Another improvement relates to the average weight of the grapes. Indeed, the measurements performed by [9] show a high dependency on the pluviometry before the readings. Also, the vitality of the vines varies each year, leading to a high variability when comparing with historical averages. The following section details one approach that may help our algorithms to compensate for this variability.

Figure 4.2.2 Example of grape segmentation. On the left: the original image. On the right: the segmented image.

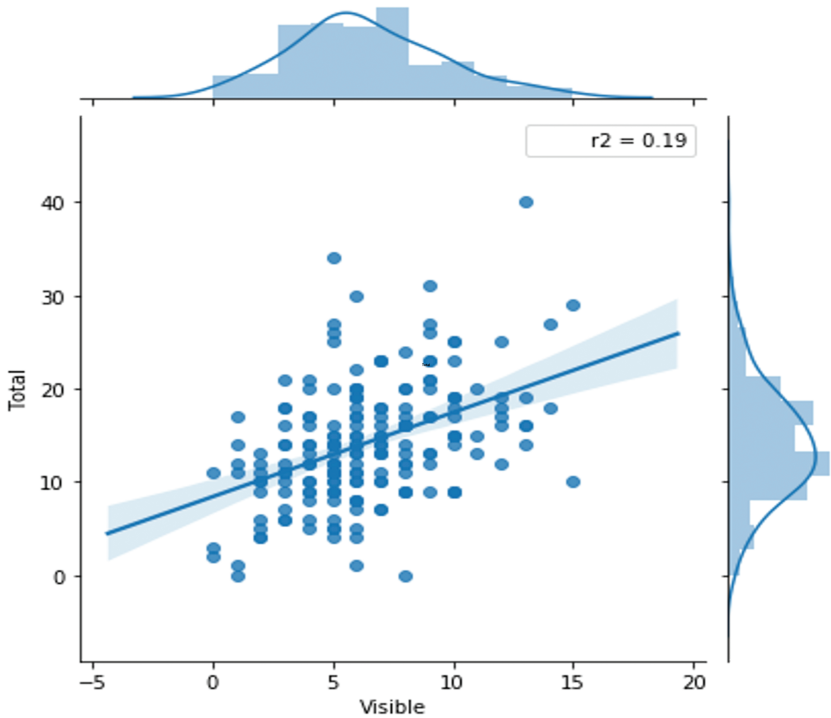


Figure 4.2.3 Correlation between visible grapes and the total number of grapes.

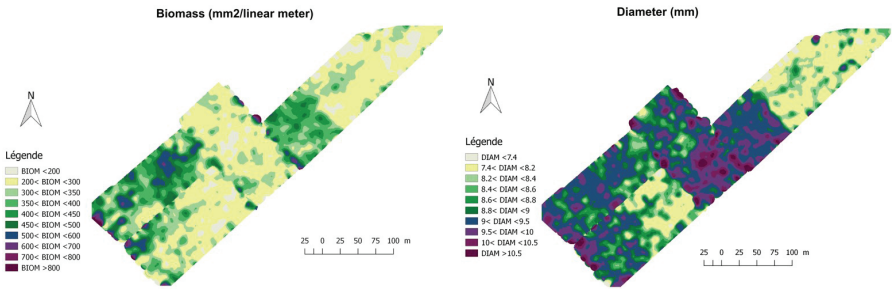


Figure 4.2.4 Biomass estimation and vine cane diameters obtained from Physiocap©.

4.2.4 Assessing the Vine Vitality Using an Embarked LiDAR

The yield estimation depends not only on the grape count but also on the relative weight of the grapes and berries. As a result, estimation divergences may appear between different vineyard parcels. Indeed, vineyard management practices and terroir characteristics may influence the fruit quality and quantity. Also, the vine balance can affect grape content as sugar, acids, and flavours concentrations [11]. Therefore, vine vigour evaluation is an important tool to estimate the vine balance, which is directly linked to the number of branches per linear meter and the diameter of these branches.

The traditional measurement technique is based on counting and weighing the winter dormant canes manually. This method is time-consuming, and the accuracy can be compromised by manual sampling, which does not consider all vineyard densities. An alternative for the traditional approach is the mapping of the dormant canes using a 2D laser scanner LiDAR sensor before pruning to assess variability in vine vigour within vineyards. This scanner allows to create charts representing the vineyard parcels, as seen in Figure 4.2.4. Previous works suggest that laser scanners offer great promise to characterise field variability in vine performance [13].

In addition, the LiDAR sensor can be installed on a vineyard robot, allowing a fully autonomous measurement of all the vineyard areas with few human interventions (Figure 4.2.5). Using a robot is a safe, more ecological, and less time-consuming support than a straddle vine tractor. Coupled with the image acquisition cameras used for grape detection, the robot becomes a fully automated tool to improve the yield forecasting for the winemakers.

Figure 4.2.5 (a) Vineyard robot Bakus©, (b) Physiocap© LIDAR installed on the robot.

4.2.5 Conclusions

The work that has been developed since 2019 shows an interest in deep learning for the detection and counting of grapes in natural conditions. These approaches have greater flexibility with respect to classic methods based only on image analysis. Indeed, deep learning has achieved better results for fruit and flower detection by avoiding the subjective selection of the algorithms and features.

These good performances have only been evaluated for grape count estimation. Yet, yield estimation requires an extra modelling step to determine the hidden part of the fruits: number of grapes hidden by the leaves, number of berries per grape, etc. Therefore, better performances are expected with non-linear modelling using additional information such as the vine vitality.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway. We also

would like to thank the ROMEO Computing Center of the University of Reims Champagne-Ardenne.

References

[1] P.R. Clingeleffer, S.R. Martin, G.M. Dunn, M.P. Krstic, 'Crop development, crop estimation and crop control to secure quality and production of major wine grape varieties: a national approach'. Final report to Grape and Wine Research and Development Corporation, Australia, 2001.

[2] I. Dami, 'Methods of crop estimation in grape'. Department of Horticulture and Crop Science at the Ohio State University, 2011.

[3] S.F. Digennaro, P. Toscano, P. Cinat, A. Berton, A. Matese, 'A Low-Cost and Unsupervised Image Recognition Methodology for Yield Estimation in a Vineyard', Frontiers in Plant Science 10, p. 559, 2019. DOI: 10.3389/fpls.2019.00559

[4] G.M. Dunn, S.R. Martin, 'Yield prediction from digital image analysis: A technique with potential for vineyard assessments prior to harvest'. Australian Journal of Grape and Wine Research 10(33), 196-198, 2004. DOI: 10.1111/j.1755-0238.2004.tb00022.x.

[5] K. Heinrich, A. Roth, L. Breithaupt, B.Möller, J. Maresch, 'Yield prognosis for the agrarian management of vineyards using deep learning for object counting'. Wirtschaftsinformatik 2019 Proceedings, p. 15, 2019. https://aisel.aisnet.o rg/wi2019/track05/papers/3

[6] A. Krizhevisky, I. Sutskever, G.E Hinton, 'ImageNet Classification with Deep Convolutional Neural Networks'. Communications of the ACM, vol 60, n. 6, 2017. DOI: 10.1145/3065386.

[7] X. Liu, S.W. Chen, C. Liu, S.S. Shivakumar, J. Das, C.J. Taylor, C.J. Underwookd, V.Kumar. 'Monocular Camera Based Fruit Counting and Mapping With Semantic Data Association', IEEE Robotics and Automation Letters 4, no 3, p. 2296-2303, 2019. DOI: 10.1109/LRA.2019.2901987

[8] W. Maldonado, J.C Barbosa, 'Automatic green fruit counting in orange trees using digital images'. Computers and Electronics in Agriculture 127, 572-581, 2016. DOI: 10.1016/j.compag.2016.07.023.

[9] L. Rossignon, 'Vers une méthode optimale d'estimation du rendement de la vigne basée sur l'intelligence artificielle', MSc. Thesis report, AgroParisTech, 2020.

[10] T.T. Santos, L.L de Souza, A.A. dos Santos, S. Avila, 'Grape detection, segmentation, and tracking using deep neural networks and three-dimensional association'. Computers and Electronics in Agriculture 170, 105247, 2020. DOI: 10.1016/j.compag.2020.105247

[11] P. Skinkis, A. Vance. 'Understanding Vine Balance: An Important Concept in Vineyard Management', Oregon State University Extension Manual EM9068, 2013.

[12] Y. Song, C. Glasbey, G. Horgan, G. Polder, J. Dieleman, G. van der Heijden, 'Automatic fruit recognition and counting from multiple images'. Biosystems Engineering 118, 203–215, 2014. DOI: 10.1016/j.biosystemseng.2013.12.008.

[13] A.C. Tagarakis, S. Koundouras, S.Fountas, T. Gemtos. 'Evaluation of the use of LIDAR laser scanner to map pruning wood in vineyards and its potential for management zones delineation', Precision Agric 19, 334–347, 2018. DOI: 10.1007/s11119-017-9519-4

[14] J. Xiong, Z. Liu, R. Lin, R. Bu, Z. He, Z. Yang, C. Liang. 'Green Grape Detection and Picking- Point Calculation in a Night-Time Natural Environment Using a Charge-Coupled Device (CCD) Vision Sensor with Artificial Illumination', Sensors (Basel, Switzerland) 18, no 4, p. 17, 2018. DOI: 10.3390/s18040969

[15] ECSEL AI4DI project. Artificial Intelligence for Digitising Industry. Available online at: https://ai4di.eu/

4.3

AI-Based Quality Control System at the Pressing Stages of the Champagne Production

Lucas Mohimont[1], Mathias Roesler[1], Angelo Steffenel[1], Nathalie Gaveau[2], Marine Rondeau[1], François Alin[1], Clément Pierlot[2], Rachel Ouvinha de Oliveira[2], Marcello Coppola[3] and Philipe Doré[4]

[1]University of Reims Champagne-Ardenne, France
[2]Champagne Vranken-Pommery, France
[3]ST Microelectronics, France
[4]CEA-LIST, France

Abstract

Deep learning (DL) is a hot trend for object detection and segmentation, thanks to Deep Neural Networks (DNNs). Image recognition is a powerful tool for precision viticulture, having strong potential in yield estimation and automatic quality estimation of the grapes. However, developing the models is one part of the problem; deploying them in the field, at the edge of the network, is another problem that comes with its own constraints. This paper studies the use of embedded devices to run DNN algorithms for real-time grape segmentation at the wine press. The results show that it is possible to use edge devices while respecting a real-time context with little detection quality losses.

Keywords: grape detection, precision viticulture, deep learning, edge computing, computer vision, object detection, Tensorflow-Lite.

4.3.1 Introduction and Background

Computer vision has helped automate tasks that once required intensive manual labour. For example, it has been used to automatically count fruits and vegetables such as peppers [14] or oranges [11]. Applying this to viticulture is a more challenging problem because each fruit, i.e., the grape, is made of several berries with colours that can vary depending on the variety (white or red) or even resemble the colour of the foliage before the grapes ripen. Nonetheless, detecting grapes automatically is a necessary step for solving other, more complex problems such as yield estimation. Dunn et al. [5] were the first to propose a method for detecting grapes in images. Since then, many methods have been developed to achieve better detection rates and be used on large scales.

Indeed, several approaches can be used to identify the location of grapes on an image. The most intuitive way is by looking at the colour of each pixel, as proposed by Dunn et al. [5]. Unfortunately, this method is sensitive to the lighting condition and cannot be used for different grape varieties: red or white.

Another approach to detecting grapes consists in trying to detect the individual berries as first proposed by [7], which uses the reflection properties of light on each berry. It will produce a specular reflection pattern that follows a Gaussian distribution that can be used to isolate the individual berries that compose the grape. Although it has been implemented in the field and evaluated on a large scale, this approach requires additional equipment (flash or lamp) and works best at night. Therefore, it is not a practical method for use in the field.

Machine learning methods have been proposed to create a binary estimation on each pixel or pixel block of an image. In this case, the selected pixel or block is classified as either a grape or not a grape. Some methods require a feature extraction process before the classification [4][10] and others, based on deep learning, combine the feature extraction and the classification within the same model [3][2]. In this first case, many different features can be used, for example, the average of the RGB channels in the pixel block [10] or the colour histogram [9]. Several classifiers are possible as well, such as the Multi-Layer Perceptron (MLP) [2], Support Vector Machines [4] and AdaBoost [10]. Each method will be a combination of these two different algorithms - feature extractor and classifier. One of the main problems with this approach is that the quality of the classification results depends on the choice of the feature extractor, which in turn depends on

the researcher's choice. One of the main problems with this approach is that the quality of the classification results depends on the choice of the feature extractor, which in turn depends on the researcher's choice.

Convolutional Neural Networks can overcome this problem by combining the feature extraction process and the classification of the extracted features in the same algorithm. Different architectures have been examined with the objective of detecting grapes using transfer learning which yields good results [3]. Some other models have also been explored, such as the Mask R-CNN by Santos et al. [13], Faster R-CNN, R-FCN, and SSD [8]. These methods detect the location of grapes in the image with bounding boxes. Other approaches use deep learning for semantic segmentation to detect individual berries [6][15] or grapevine flowers [12], which we use in this work.

The deployment of deep learning for industrial applications is a challenge. Current deep learning models are trained on powerful GPUs. The challenge is to convert these models for real-time inference on the field. One way to deploy the algorithms is to use specific hardware such as embedded devices, essentially small computers that can operate in remote places like vineyards or wine presses. These small devices have limitations, most notably in computing power and available memory. These constraints must be acknowledged to choose the most suitable tools and algorithms for onboard applications. These constraints include the inference time that must be low enough for practical use. Luckily, a wide range of readily available boards with various capabilities can be used for deploying grape detection algorithms. The option of creating a board for a specific application is always possible.

This paper focuses on detecting unwanted elements (green or ripen grapes, leaves, stones, tools) before delivering the grapes to the press. This paper will be looking at the deployment of a deep neural network for semantic segmentation on a readily available embedded device, enabling AI inference at the edge of the network. Different versions of the model will be tested, and the performances will be analysed based on these three criteria: inference time, performance loss when compared to the original model, and model size. In addition, results must be obtained in less than 15 seconds not to impact the winery production chain.

4.3.2 Methodology

Our team acquired the images for training and testing at the wine press, using a GoPro camera mounted over the weighting device, as illustrated

Figure 4.3.1 Image acquisition at the wine pressing sites.

in Figure 4.3.1. Each high-resolution image (4000 by 3000 pixels) covers four crates at the wine press containing grapes and some of the surrounding environment. However, the model can only accept image blocks of 224 by 224 pixels as input. Therefore, the training and validation set images were split into smaller blocks that correspond to 12713 image blocks for the training set and 3250 image blocks for the validation set. The model was implemented in Python using TensorFlow's Keras API and saved in the *hdf5* format. All the weights and biases are stored as 32-bit floating-point numbers (float32 datatype) for this model. Figure 4.3.2 illustrates the image analysis workflow, where a segmentation mask is devised and used to select only the classes we are interested in (in this case, it applies a binary mask to select only grapes).

Embedded devices are not necessarily powerful enough to run an AI model written in Python. For efficiency reasons, the applications that are run on these kinds of devices are usually programmed in a compiled language such as C or C++ instead of an interpreted language like Python. However, to avoid having to rewrite entire models in one of these languages and yet still deploy them on smaller, embedded devices, Tensorflow has created a conversion process to optimise an *hdf5* model developed using their API. The process converts the model into an optimised FlatBuffer by, for example, fusing layers when possible. This conversion aims at reducing the overall model size while trying to maintain the performance of the original model. Different options are available when converting a model from the *hdf5* format

(a) Input image (b) Segmentation mask (c) Segmented image

Figure 4.3.2 Example of the image processing steps.

to the *tflite* format. A commonly used one is the quantisation of the weights and biases to optimise the model. From the original encoder-decoder model, two variants were generated using this converter. Both variants were created with the TOCO converter provided by TensorFlow and saved in the *tflite* format. The first model was converted in the most straightforward manner using the API provided by TensorFlow's version 2.2. No datatype conversion was performed on the weights, biases, or activations for this model, and they maintain their original datatype of float32.

While also being converted using TensorFlow's version 2.2, the second variant took advantage of the post-training integer quantisation process to convert the datatype of the constant tensors (i.e., weights and biases) and the variable tensors (i.e., activations) from float32 to int8. This quantisation process reduces the model's size and memory usage while increasing inference speed, allowing it to run on smaller devices. However, it will inevitably decrease the global performances of the model due to rounding errors that will occur during the conversion. To convert the variable tensors (the output of the intermediate layers), a representative dataset must be provided to estimate the range of the floating-point tensors by running a few inference cycles. A specific dataset does not need to be created for this process; therefore, the representative dataset was generated using the images in the validation set from the original model. Several of the images were cut into blocks of the same size as the model's input. A total of 179 images of 224x224 pixels were included in the representative dataset. Before being used for the quantisation of the model, the images were normalised. This conversion operation is necessary for being able to use the model on a TPU. The accelerator can only run layers that have been converted beforehand. If the entire model is not quantised, then the operations that have not been affected by the process will be run on the CPU.

In this case, all layers were successfully converted, except the first and the final one. As the images are normalised before input, the value of each pixel is a floating-point number between 0 and 1. If the input layer only accepted integers, then the pixel values would be rounded off, and the input image would only contain values of 0, creating a black image and rendering the inference pointless. For the final layer, even though the model's output is a binary mask with integer values for pixels — 1 represents a pixel belonging to a grape and 0 a pixel that does not – the performances significantly deteriorated if the output of the final layer was of type int8. Therefore, the first and final layers were not converted using post-training quantisation.

The device used to run the AI models is an STM32MP157C-DK2 board produced by STMicroelectronics. It has two processors, a dual-core Cortex-A7 32 bits and a Cortex-M4 32 bits. The latest version of the X-Linux-AI package, created explicitly by STMicroelectronics to run AI models on their devices, was installed on the board. This package comes with TensorFlow Lite 2.4.1 and the necessary support libraries for using Coral Edge TPU accelerators. Since it cannot have any version of TensorFlow installed on it, the STM32MP157C-DK2 can only run *tflite* models. Because this board has no dedicated GPU for artificial intelligence, inferences can only be performed using its CPU, which can be pretty slow. Therefore, we equipped the board with a Google Coral USB accelerator (Figure 4.3.3). This tensor processing unit (TPU) is an ASIC processor specifically designed to accelerate the inference of artificial intelligence models, provided as TensorFlow Lite models.

Figure 4.3.3 STM board and TPU accelerator used in this work.

To evaluate the performances, three criteria were used: the inference time, the model's overall size, and the intersection over union (IOU) score. The inference time is used to compare the hardware and software options concerning the real-time constraint that has been set, 15 seconds in this case. The size of the model is given to show how efficient the compression is during the conversion process. Finally, the IOU score was calculated relative to the results obtained with the DGX server. Using this metric, we aim to assess how the inferences from different model versions differ from the original one. Hence, an IOU score of 1 means that the variants' performances are the same as those of the baseline model.

4.3.3 Results and Discussion

The obtained results are presented in Table 4.3.1. Also, Figure 4.3.4 shows an example of the output for each model variant on the STM32MP157C-DK2 board against the baseline output obtained on the DGX server. The inference time and the IOU score presented here were obtained by averaging the individual inference time and IOU score of all the images in the test set. Three tests were run with different models. The STM32MP157C-DK2 board can only run the *tflite* versions of the model (quantised and not quantised) because the board does not support TensorFlow but only the TensorFlow Lite runtime environment. Therefore, the quantised model was run twice, once without using the TPU accelerator and the second time with the accelerator.

The results show that the chosen device is not powerful enough to run the *tflite* model using only the CPU, whether quantised or not and fit the requirements. The quantisation process does allow for a slight decrease in the inference time, of a factor of 1.1 only. This improvement is far insufficient to satisfy the real-time requirements that were set. The only viable solution is to use the TPU accelerator, as the inference time is reduced by a factor of 13 when comparing it with the same model without using the accelerator. Using the accelerator has a drawback as it forces the model to be quantised, inducing a performance degradation as shown by the relative IOU score of 0.93. Considering the significant reduction in the inference time, the slight deterioration of the performances is justified, especially since it is the only scenario that fills the real-time requirements. However, it is interesting to note that since the IOU score is 1 for the non-quantised model, the conversion from the *hdf5* format to the *tflite* does not impact its performances and the only effect is to reduce its overall size. The compression factor between each model is approximately three. More precisely, it achieves a factor of

Table 4.3.1 Results obtained with the STM32MP157C-DK2 board.

	IOU score	Inference (sec.)	Size (in MB)
Baseline	N/A	0.4	93
tflite not quantised	1	137	31
tflite quantised without TPU	0.93	117	8.7
tflite quantised with TPU	0.93	11	8.7

(a) Baseline (b) *tflite* model not quantised (c) quantised without TPU (d) quantised with TPU

Figure 4.3.4 Output of the models.

3 between the original and the *tflite* version and a factor of 3.6 between the non-quantised and the quantised version. Even if the quantisation process impacts the performances, it still allows for complex and heavy models to run on devices with limited resources.

4.3.4 Conclusion

Deep neural networks require large amounts of resources to operate. This is not a problem when deployed on various servers with powerful GPUs; however, this impedes deploying trained models on embedded devices with limited capabilities. To tackle this problem and avoid having to rewrite models in programming languages better suited for smaller devices such a C or C++, different converters exist to reduce the size and the necessary resources for the models to run. These converters allow models initially developed using high-end APIs such as TensorFlow to be easily deployed on boards such as the STM32MP157C-DK2.

In this paper, a deep neural network with an encoder-decoder architecture for semantic segmentation was converted to the *tflite* format, allowing it to run on two small devices. The evaluation proposed in this paper compares three criteria: the inference time, the IOU score relative to the non-converted

model, and the model size. The obtained results are very encouraging. They show that deploying the converted model in a real-time context is possible while limiting the performance losses due to its conversion. Furthermore, the time constraints at the wine press are relatively light, allowing the exploration of model architectures that are not necessarily conceived for real-time applications, such as the original encoder-decoder architecture used in this case study. Nonetheless, this paper gives some insights into the trade-off between performances and inference time when deploying models to smaller devices.

Still, other alternative converters have not been studied here. For instance, the N2D2 platform [1] developed by the CEA-List can convert models from an ONNX format to various targets, including the STM32MP157C-DK2 board. Using N2D2 would bypass the use of Python and TensorFlow Lite by creating a specifically designed project in C. Using this converter may provide better inference time while maintaining the same performances and will be explored in the future.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

We also would like to thank the ROMEO Computing Center[1] of the University of Reims Champagne-Ardenne, where the original model was developed and trained.

References

[1] CEA-LIST/N2D2, https://github.com/CEA-LIST/N2D2, original-date: 2017-01-06, Apr. 2021.

[2] N. Behroozi-Khazaei, M.R. Maleki, 'A robust algorithm based on color features for grape cluster segmentation', Computers and Electronics in Agriculture 142, pp. 41-49, 2017. DOI: 10.1016/j.compag.2017.08.025

[1] https://romeo.univ-reims.fr

[3] H. Cecotti, A. Rivera, M. Farhadloo, M.A. Pedroza, 'Grape detection with convolutional neural networks'. Expert Systems with Applications 159, 2020. DOI: 10.1016/j.eswa.2020.113588

[4] R. Chamelat, E. Rosso, A. Choksuriwong, C. Rosenberger, H. Laurent, P. Bro, 'Grape detection by image processing'. IECON 2006 - 32nd Annual Conference on IEEE Industrial Electronics. pp. 3697-3702, 2006. DOI: 10.1109/IECON.2006.347704

[5] G.M. Dunn, S.R. Martin, 'Yield prediction from digital image analysis: A technique with potential for vineyard assessments prior to harvest'. Australian Journal of Grape and Wine Research 10(33), 196-198, 2004. DOI: 10.1111/j.1755-0238.2004.tb00022.x

[6] J. Grimm, K. Herzog, F. Rist, A. Kicherer, R. Töpfer, V. Steinhage, 'An adaptable approach to automated visual detection of plant organs with applications in grapevine breeding'. Biosystems Engineering 183, 170-183, 2019. DOI: 10.1016/j.biosystemseng.2019.04.018

[7] M. Grossetete, Y. Berthoumieu, J.P. Da Costa, C. Germain, O. Lavialle, G. Grenier, 'Early estimation of vineyard yield: Site specific counting of berries by using a smartphone'. In: International Conference on Agriculture Engineering (AgEng), 2012, https://hal.archives-ouvertes.fr/hal-00950298

[8] K. Heinrich, A. Roth, L. Breithaupt, B. Möller, J. Maresch, 'Yield prognosis for the agrarian management of vineyards using deep learning for object counting'. Wirtschaftsinformatik 2019 Proceedings, p. 15, 2019. https://aisel.aisnet.o rg/wi2019/track05/papers/3

[9] S. Liu, S. Marden, M. Whitty, 'Towards automated yield estimation in viticulture'. Proceedings of the Australasian Conference on Robotics and Automation, Sydney, Australia p. 9, 2013.

[10] L. Luo, Y. Tang, X. Zou, C. Wang, P. Zhang, W. Feng, 'Robust grape cluster detection in a vineyard by combining the AdaBoost framework and multiple color components'. Sensors 16(1212), 2016. DOI: 10.3390/s16122098

[11] W. Maldonado, J.C Barbosa, 'Automatic green fruit counting in orange trees using digital images'. Computers and Electronics in Agriculture 127, 572-581, 2016. DOI: 10.1016/j.compag.2016.07.023

[12] R. Rudolph, K. Herzog, R. Töpfer, V. Steinhage, 'Efficient identification, localisation and quantification of grapevine inflorescences in unprepared field images using fully convolutional networks'. arXiv:1807.03770 [cs] pp. 95-104, 2018.

[13] T.T. Santos, L.L de Souza, A.A. dos Santos, S. Avila, 'Grape detection, segmentation, and tracking using deep neural networks and three-dimensional association'. Computers and Electronics in Agriculture 170, 105247, 2020. DOI: 10.1016/j.compag.2020.105247

[14] Y. Song, C. Glasbey, G. Horgan, G. Polder, J. Dieleman, G. van der Heijden, 'Automatic fruit recognition and counting from multiple images'. Biosystems Engineering 118, 203–215, 2014. DOI: 10.1016/j.biosystemseng.2013.12.008

[15] L. Zabawa, A. Kicherer, L. Klingbeil, R. Töpfer, H. Kuhlmann, R. Roscher, 'Counting of grapevine berries in images via semantic segmentation using convolutional neural networks'. ISPRS Journal of Photogrammetry and Remote Sensing 164, 73–83, 2020. DOI: 10.1016/j.isprsjprs.2020.04.002

4.4

Optimisation of Soybean Manufacturing Process Using Real-time Artificial Intelligence of Things Technology

Ovidiu Vermesan[1], Jøran Edell Martinsen[2], Anders Kristoffersen[2], Roy Bahr[1], Ronnie Otto Bellmann[2], Torgeir Hjertaker[2], John Breiland[3], Karl Andersen[3], Hans Erik Sand[3], Parsa Rahmanpour[4] and David Lindberg[4]

[1]SINTEF AS, Norway
[2]DENOFA AS, Norway
[3]NXTECH AS, Norway
[4]INTELLECTUAL LABS AS, Norway

Abstract

In this article, a soybean process optimisation solution using real-time artificial intelligence of things (RT-AIoT) technology at the edge is presented. Image classification, object detection and recognition are machine vision techniques implemented into industrial internet of things (IIoT) devices to determine variations in the morphological features in soybeans. Evaluating soybean features, such as moisture and temperature combined with other measurements, such as colour, size, shape, and texture, can improve the utilisation of the raw material and the quality of the derived products, thus reducing energy consumption. Implementing intelligent vision locally on IIoT edge devices solves several issues faced by deploying it to the cloud and brings further challenges posed by deep learning on resource-constrained edge devices. Most deep neural networks are too complex to be created and trained on most nowadays microcontrollers, but if optimised in terms of memory, processing, and power capabilities, they can run on them. With multi-image sensors, and IIoT devices under evaluation, the

proposed production optimisation system is interfaced with the existing industrial SCADA system, and analyses the IIoT sensor data at different edge computing granularity levels. With the preliminary findings and results, we show that the RT-AIoT, including machine vision technology, is now possible on all micro, deep and meta edge levels with the advent of AI.

Keywords: production optimisation, artificial intelligence, smart sensors systems, edge computing, industrial internet of things, industrial internet of intelligent things, soybeans manufacturing, machine vision, machine learning, deep learning, SCADA, PLC, real-time artificial intelligence of things (RT-AIoT).

4.4.1 Introduction

The digitising industry brings about the integration of the physical and digital systems of the production environments. It allows the collection of vast amounts of information using supervised control and data acquisition (SCADA) systems comprising programmable logic controllers (PLC), sensors/actuators and industrial internet of things (IIoT) devices [1][2]. These devices are connected to different equipment located in various production facilities, measure and monitor several parameters and process the data in on-premises servers and the cloud. The new technologies integrate people, machines, and products, enabling faster and more targeted information exchange. The information insights and analytics are increasing in value by implementing artificial intelligence (AI) techniques and methods collected by IIoT systems and processing at the edge close to the industrial production line. The data intelligent edge processing can bring valuable information and knowledge from the manufacturing process and system dynamics. By applying analytics and AI-based approaches based on data collected from IIoT devices, it is possible to obtain interpretive results for strategic decision making for process optimisation, cost reduction and energy-efficient process tuning.

Food processing and manufacturing include all processes intended to transform raw food materials into products suitable for consumption, cooking or resale. Implementing AI, IIoT and robotics solutions in the food processing and manufacturing sector can assist in overcoming critical issues related to production and execution by eliminating the possible chance of human errors and reducing the work redundancy being performed by manual labour.

Furthermore, innovation in production optimisation, production parameters tuning, and equipment maintenance can be fuelled by AI.

In soybean production facilities, the benefits of AI can be leveraged by using IIoT, neural networks (NNs), machine learning (ML) techniques, advanced analytical tools, image, and pattern recognition technologies to optimise production, equipment maintenance timely and less costly and overall production flow. With AI and IIoT, the data received from sensors are interpreted and recognised when action is needed. Aggregated data are generated and sorted, and significant data points are identified by sensors and AI techniques. These technologies are used to optimise processes, spot anomalies, such as early warning signs that equipment or motors may fail or require maintenance. AI technology is used to recognise patterns, expand the knowledge base, identify cause-and-effect relationships, and use insights related to likely outcomes or the next data point in the curve of the trend.

The Real-time Artificial Intelligence of Things (RT-AIoT) is the combination of AI technologies with the IIoT devices and infrastructure to achieve more efficient real-time IIoT operations, improve human-machine interactions and enhance data management and analytics.

In this article, an approach to optimising an industrial soybean manufacturing process using AI-based methods and RT-AIoT technology is presented.

The article is organised as follows. This section provided the introduction and the background for this research and innovation activity. The next three sections give an overview and a description of soybean production process, reference architectural conceptual framework, and process parameters monitoring techniques. The micro, deep and meta edge concepts are described in the next section. Afterwards, the section on embedded intelligent vision and multi-sensors fusion approach describes the system requirements, including an overview of relevant hardware architectures. The experimental set-up section depicts the overall architecture and workflow, the specific experiments performed and results. Finally, the last section concludes and highlights the next steps.

4.4.2 Soybean Production Process Description

For the use case presented in this article, the soybean production starts with 30 000 tonnes of soybeans arriving at the manufacturing facility on ships every three to four weeks.

The ships are unloaded in 3–4 days into a flat storage container, where the soybeans are stored until they are processed.

From the storage, the soybeans are transported on a conveyor belt into the cleaning area of the plant. Here, the coarse fraction and dust from the soybeans are cleaned out.

The cleaned soybeans are moved through a weight in which the budget capacity is 59 tonnes per hour.

In the next step, the soybeans are cracked between two pairs of cracker rolls, where each bean is broken into 6–8 pieces. The cracked soybeans are transported in closed conveyors and through a conditioning phase where the soybeans are heated and dried before flakers. The soybeans become more elastic in this process, so they do not crack in the flaking step. The flakers press the soybeans into thin flakes between a pair of hydraulic rollers.

The next phase is the expander process, where direct steam is added to the flakes, pressing the soybean flakes to a conus with a high-pressure screw to expand the oil cells in the soybeans. After the expander phase, the water content is increased due to the added direct steam, and the expanded material is dried with hot air before extraction.

In the extraction process, soybean oil is extracted from soybeans with hexane. Then, the hexane and soybean oil mixture is pumped to the distillation, and the soybean meal is transported to the toaster and heat treatment.

During distillation, hexane is evaporated from the soybean oil in three steps. After the hexane is removed, the soybean oil is pumped into a degumming phase. Here, water is added to separate lecithin from soybean oil.

After separation, the two products are pumped into separate dryers to evaporate the water, and then the products are pumped into storage tanks.

The soybean meal must be toasted and heated to evaporate hexane, eliminate bacteria, and make the meal digestible. After toasting, the meal is hot air-dried and transported to a storage container.

The soybeans production flow is presented in Figure 4.4.1.

The products resulted after different phases of the production are illustrated in Figure 4.4.2.

The soybeans are shipped from Brazil and Canada, with temperatures fluctuating from 5°C to 35°C. With the variation in raw material, product yields and energy consumption in soybean production are affected.

Using sensors, IIoT devices, and AI-based techniques makes it possible to control variations throughout the process to optimise product yields and

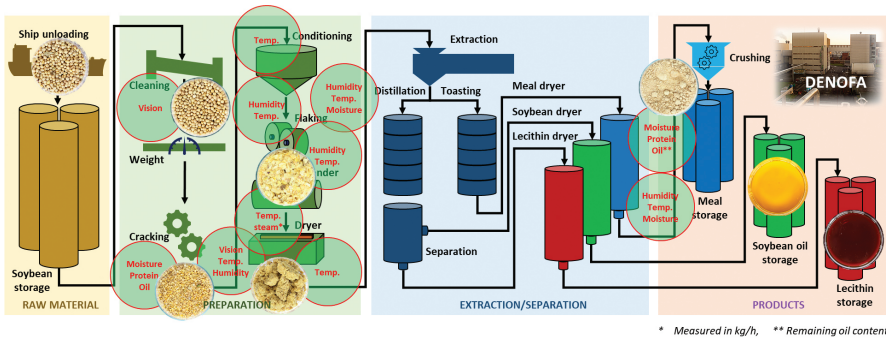


Figure 4.4.1 Soybean production process flow.

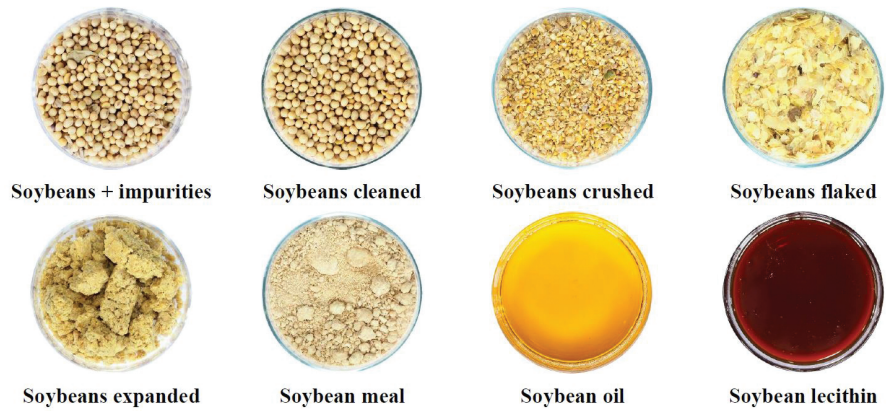


Figure 4.4.2 Soybean products.

energy consumption. Typical parameters monitored during the manufacturing process are temperature, moisture, colour, texture, weight, and volume.

Water content, also known as moisture, is the most critical parameter in preparing soybeans before the extraction phase. If the water content is too high, the residual oil in the soybean meal will increase, and the oil yield will be reduced.

This process is crucial for optimisation, and suitable locations are identified for instrumentation in cleaning, cracking and after-drying production areas.

The cleaning and preparation environment are dusty and challenging for moisture measurement and monitoring. Therefore, unique solutions must be considered for implementation.

4.4.3 Overall Manufacturing System Architecture and Platform

The soybean process optimisation solution is developed in the AI4DI (ECSEL JU) project [3]. The AI4DI reference architecture is defined at a high-level abstraction with various functional domains that include different devices, equipment on several communications networks, processing and storage capabilities at the edge and in the cloud, and training/learning embedded in different layers. The reference architectural conceptual framework includes different views, functional domains, system properties, cross-cutting functions, the description of interfaces and interactions between these elements and the features located outside the reference architecture [4].

This article uses the proposed implementation of the optimisation procedure for an industrial soybean manufacturing process using AI-based methods and RT-AIoT technology for mapping it into the functional domains. The high-level reference architecture includes six functional domains. A short description of each functional domain is provided in the next paragraphs.

The physical systems domain consists of physical components such as IIoT devices operating within soybean industrial manufacturing.

The control domain interfaces the physical systems using sensing and actuation in soybean industrial manufacturing, and implements necessary communication and means of execution. This domain includes the communication function (e.g., abstraction of different types of physical/link layer/networking technologies, Bluetooth, LoRa, Wi-Fi) with which the different sensors, actuators, and support infrastructure (gateways, controllers, routers, etc.) connect to exchange data, messages, and information.

The operations domain encompasses the provisioning, management, monitoring, diagnostics and optimisation of sets or groups of devices in the control domain, ensuring the continued operations of single devices and the associated control systems for soybean manufacturing. The domain includes provisioning, deployment, management, monitoring, diagnostic, predictive and optimisation functions implemented in on-premises edge computing facilities.

The information domain implements the collection, system-level data fusion, transformation, storage, optimisation, and analysis of data from several domains, and implements AI techniques and methods for intelligence fusion at the system level during different soybean manufacturing and

production stages. The analytics function includes data modelling, processing and analysis and the rule engines for different feature implementations.

The application domain uses case-specific logic, rules, integration, human interfaces, and models to deliver the system-wide optimisation of operations and relies on intelligence from the information domain. The APIs/UI function presents the application's capabilities in the form of APIs for dashboards or use by other applications.

The business domain integrates information from applications, business system enterprises, human resources, customer relationships, assets, service lifecycle, billing and payment, work planning and scheduling to achieve the desired business objectives. The business domain for soybean manufacturing implements the functionality for the integration of AI/IIoT-specific functions and standard enterprise business support systems such as Enterprise Resource Planning (ERP), Product Lifecycle Management (PLM), Supply Chain Management (SCM).

4.4.4 Process Parameters Monitoring

The following section gives a short description of the measurement techniques under evaluation for measuring soybean parameters and ambient conditions.

The techniques and methods [8][9] evaluated for moisture, protein, oil measurements, soybean colour, texture and pattern analysis, and ambient parameters (e.g., temperature, pressure) are presented below.

Microwave non-destructive testing (MNDT) is a non-invasive, non-destructive measurement technique in which microwaves penetrate a material and can thus be used to measure its water content (moisture). The dielectric constant of water changes with temperature and frequency, and is typically 20 times higher than that of other materials [5] at around 78.4 at ambient condition of 25°C and 1GHz [6]. This is resulting in a relatively strong interaction between microwaves and water, which is measured as attenuation and phase shift. The dielectric constant of soybeans thus influences microwaves, and the water content can be accurately determined. The equipment must use weak microwave power (typically 0.1mW) [7] so that the soybeans themselves are not heated or altered.

Near-infrared (NIR) spectroscopy is a non-invasive, non-destructive technique based on the absorption of electromagnetic radiation. NIR

spectroscopy instruments produce a large amount of data, and chemometric methods are used to extract useful information. Like many other measurements, those for NIR spectroscopy rely on standard calibration methods to achieve good results, and the instruments therefore need to be calibrated for the specific measurements that you want to perform—typically, a wide range of measurements, including highest and lowest levels of water, oil, or protein content, are needed. The possible bottlenecks of calibration versus the benefits to the demonstrator of the measurements are currently under investigation and there is yet no conclusion.

NIR light is a portion of the electromagnetic spectrum close to visible red, at about 750 to 2500 nanometres as illustrated in Figure 4.4.3, and can be used to detect the chemical bonds between atoms in organic compounds such as soybeans. Infrared absorption is caused by several effects, but the most important is the transfer of electromagnetic energy into chemical bond vibration, and absorption features may be related to specific molecular structures [10].

Soybeans absorb, reflect, and transmit varying amounts of near-infrared electromagnetic waves based on their composition. Each compound (e.g., oil, lecithin, and water) responds to a particular NIR wavelength, which can be measured to estimate the oil, water (moisture) and protein content in soybeans used to produce soy oil, lecithin, and meal. The quality of soybeans can be determined by their colour, shape, and chemical composition, and NIR technology can therefore help to identify and distinguish soybean quality based on chemical, oil, lecithin, and water composition.

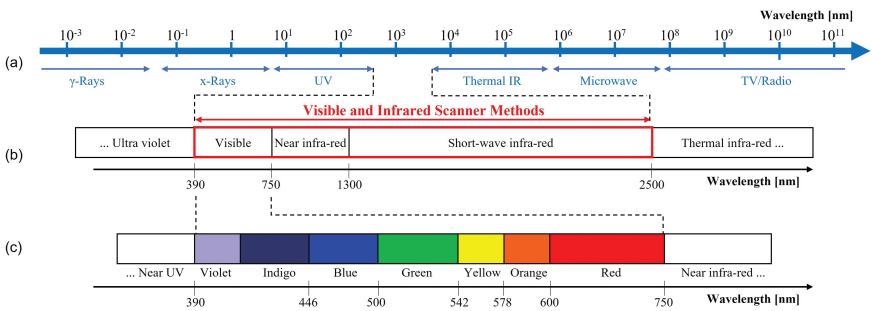


Figure 4.4.3 The electromagnetic spectrum - regions of interest in the context of NIR spectroscopy. Adapted from [10].

Hyperspectral image analysis is a non-invasive, non-destructive technique that is based on imaging of the electromagnetic spectrum, dividing it into many bands, and can be extended to a wide range of wavelengths beyond the visible range. Hyperspectral imaging measures continuous spectral bands and depends on relatively high computing power to transform the acquired pixelated images into readable data at an acceptable wavelength resolution. Optical filters and light sources are optimised for the wavelengths (bands) in the spectrum that reflect the levels of water, oil, and protein in soybeans. This method is being evaluated considering the complexity, cost, and calibration features.

Capacitive sensing is a non-destructive technique based on the same principle as a capacitor, measuring the electric field between two electrodes using a material placed between them as the dielectric. Applying an excitation voltage (DC or AC) to the electrodes creates an electric field, and the current flow in that field will change based on the conductivity of the material between the electrodes. The current is measured and transformed into values based on a physical model to give the moisture level of the material. This method is being evaluated for accuracy and other features.

Cameras can capture images in visible, infrared, near-infrared, hyperspectral spectrum to monitor the sizes and colours of whole soybeans and the texture of crushed soybeans. The image processing of crushed soybeans is more challenging than that of whole soybeans, thus requiring different types of cameras. Solutions for good lighting conditions are also needed.

Temperature and pressure sensors are used to measure the temperature in the different areas of the soybean processing line using wired/wireless sensors temperature sensors with a temperature range of 0°C to 55°C. The indoor ambient temperature and pressure vary according to location, weather conditions and seasons, and depends on the process steps performed in that area. Temperature and humidity are therefore critical parameters to measure and consider when analysing data from different points in a soybean production line.

AI and IIoT rely upon data generated at the sensor level, and data must be consistent, accurate and reliable. Sensors must have the required precision and embedded connectivity to pass measurement data for process optimisation purposes to the edge computing data system.

A common historian where information is aggregated for AI-based analytics, reporting and visualisation is needed to aggregate the data from the SCADA system and made it available for the edge server.

The value of AI and IIoT is limited by the ability to capture data from sensors in the soybean manufacturing process. The wired/wireless sensors must accurately measure moisture, temperature, humidity and other visual constituents, and the system should provide a way to confirm that the readings are accurate. This requires that the sensors are calibrated, and able to provide information when their battery life is low and a diagnose action is needed.

The process parameters monitoring includes a modular design for reliable sensing solutions in the harsh soybean manufacturing environment.

The sensor and related electronics are adequately packaged and placed in secure locations so they are not exposed to overrated temperature, humidity, dust, and other ambient conditions that can degrade and/or damage the sensors, IoT devices, gateways prematurely.

Power consumption is critical for the lifetime of the wireless sensors, the measurement precision of the sensors, and the AI-based algorithms applied to them. Therefore, viable power monitoring and energy-efficient communications capabilities must be integrated into the design.

4.4.5 Edge Processing and AI-based Framework for Real-Time Monitoring

Intelligent edge computing architectures accelerate the move to more processing and the value-creating process-optimisation use cases associated with the edge. The approach used in this work for soybean process optimisation addresses the granularity of the edge by providing intelligence to the micro, deep and meta edge. A description of the micro, deep and meta edge concepts are provided in the following paragraphs.

The micro edge describes the intelligent sensors, machine vision and IIoT devices that generate data and are implemented using processors and microcontrollers (e.g., ARM Cortex M4) due to constraints related to costs and power consumption. The distance from the compute resource is minimal, as the compute resources operate on the data they generate. The hardware devices of the micro-edge physical sensors/actuators generate data and/or actuate based on physical objects. Integrating AI-based elements into these devices and running AI-based techniques for training/learning and inference

on these devices brings the intelligence and analytics closest to the physical parameters measured.

The deep edge comprises intelligent controllers PLCs, SCADA elements, machine vision connected embedded systems, networking equipment (IIoT gateways) and computing units that aggregate data from the sensors/actuators of the IIoT devices generating data. Deep edge processing resources are implemented with performant processors and microcontrollers (e.g., Intel i-series, Atom, ARM M7+, etc.) that include components such as CPUs, GPUs, TPUs or ASICs.

The meta edge integrates processing resources, typically located on premises, implemented with embedded high-performance computing units, edge machine vision systems, edge servers (e.g., high-performance CPUs, GPUs, FPGAs, etc.) that are designed to handle compute-intensive tasks, such as processing, data analytics, AI-based functions, networking, and data storage.

The edge analytics applications presented in this work enable new use cases that rely on low-latency and high-data throughput. The demonstrator developed use intelligent sensors, embedded machine vision and IIoT devices integrated with edge computing to implement learning and inference on-premises in the soybeans manufacturing facility.

4.4.6 Embedded Intelligent Vision and Multi-sensors Fusion Approach

Image classification, object detection and recognition are machine vision techniques using information collected from IIoT sensors. With such information, it is possible to determine morphological features such as colour, size, shape, texture, and moisture in soybeans for monitoring and improving the utilisation of the raw material and the quality of the derived products, thus reducing energy consumption. With the advent of AI, this capability is now possible on all micro, deep and meta edge levels.

Intelligent devices are enabled by machine vision to grasp the visual surroundings. Machine vision is integrated into the perception systems in industrial sectors, including autonomous vehicles, food processing, semiconductors and more, and is one of the areas that has benefitted the most from the rapid advances in AI/ML. ML algorithms enable high performance

in image segmentation, object detection, image classification, object tracking, pattern and object recognition, image generation, and more.

Deep Learning (DL), a subset of ML, allows machines, robots and intelligent IIoT devices to recognise objects with close to human-like ability. At the lower levels, ML algorithms perform processing techniques on the image, extract features from the image, access and intertwin multiple views. At the higher level, they perform more advanced tasks, such as image classification - making inferences about whether the object in the image belongs to a specific class of objects. It is at the highest level that DL is employed to build intelligent, scalable machine vision systems that can recognise/identify and react/respond to objects in images and videos.

Convolutional neural network (CNN) is a class of DL networks and has become increasingly powerful in large-scale image recognition on IIoT devices by combining the feature extraction process and classifying the extracted features in the same algorithm, relying on extracted features. When DL technology is deployed in IIoT devices, it relies on pretrained DL models, and transfer learning techniques are employed to retrain an existing image classifier into a custom classifier by retraining a small image dataset using minimal resources. CNN is under evaluation along with other DL models and techniques.

Edge sensors and IIoT devices are increasingly becoming more intelligent, generating a massive amount of data, often creating latency, reliability, and privacy concerns. A shift in AI processing from the cloud to the edge was triggered by such developments, made possible by recent advances in microcontroller architectures and algorithm design. By deploying intelligent vision locally on IIoT edge devices, most concerns related to deploying to the cloud are addressed and answered:

- Bandwidth: ML algorithms need lots of data and transferring large amounts to the cloud is costly and demands bandwidth. Therefore, severe reductions must be applied, affecting the performance and accuracy of the results from the algorithms. When algorithms run on IIoT edge devices, the amount of data processed is limited only by IIoT edge device capabilities.
- Latency: ML models on IIoT edge devices can respond in real-time to inputs (as the round-trip to the cloud is no longer involved) enabling real-time edge nodes to run in real-time and meet deadlines.
- Costs: By processing data on-device, the costs of transmitting data over a network and processing it in the cloud are reduced. The cost of running ML in the cloud can be expensive due to the complex infrastructure.

- Reliability: Systems controlled by on-device models are inherently more reliable, not least because they are no longer affected by outages in the cloud.
- Privacy: User privacy is protected when data are processed locally on an embedded system and are not transferred to the cloud.

Nonetheless, other concerns are posed by ML on machine vision IIoT edge devices. Most deep NNs are too complex to be created and trained on most nowadays microcontrollers, but if optimised in terms of memory, processing, and power capabilities, they can run on them. The optimisation can be done either by rewriting the models in low-level languages or by quantising to improve the latency and the model size.

It is envisaged that it will be more common for machine vision IIoT edge devices to embed deep NNs and other AI techniques in the future. For now, thanks to interoperability efforts, tools and methods are available to optimise deep NN that have been trained on standard platforms to do specific tasks. Therefore, they can run on IIoT edge devices with limited capabilities. It is a matter of balancing the goals of obtaining the most significant reduction in the size of the original code with a minor accuracy loss.

Real-time monitoring and control are essential criteria for optimising process parameters and maximising soybean manufacturing production outcomes. The proposed process optimisation is built on an industrial real-time data acquisition AI-based system (intelligent sensors and machine vision IIoT devices) implemented into an on-premises edge computing environment integrated with existing industrial SCADA system. The remote soybean parameters are measured and collected by the intelligent data acquisition and control system through reliable protocols and communication networks, providing an interface with the existing SCADA system through a common historian entity.

In this context, multi-sensor fusion is the process of achieving multi-objective optimisation by combining data from multiple sensors, which, taken separately, can only provide local optimums.

The data aggregation functionalities are integrated into the edge platform components, whereas the IIoT gateway handles edge data collected from different IIoT devices. The IIoT hardware platform and devices are integrated with the existing SCADA system and open platform communications server (OPC), interfaced with the ERP manufacturing facility and web and mobile App solutions.

Monitoring the moisture of soybeans before processing is critical, and three process sub-systems are identified as possible locations in the processing workflow and contain component sensor instrumentation according to the sensor tag system developed for unique identification.

The moisture measurements and other monitoring measurements are seen in conjunction with temperature and image analyses. The targeted measurement parameters monitored are the moisture of soybeans before processing at different locations in the processing workflow (e.g., on the conveyor belt before cleaning, on the conveyor belt after cleaning and before weight, and after crackers before conditioning). The aim is to measure soybean water content, temperature and "quality of cracking," to control the changes and adjust the conditioning according to the variations.

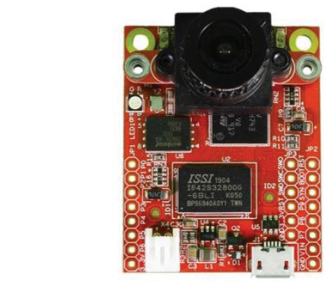
Different communication protocols and gateways are used (BLE, LoRa and Wi-Fi), depending on data rate, bandwidth, application, etc. Even in harsh environments, communication with edge devices is facilitated by the seamless integration of wireless connectivity, ensuring data storage, pre-processing in real-time, visualisation, and possibilities to change parameters or effectuate other necessary actions.

4.4.6.1 Embedded Vision IIoT Systems Evaluation

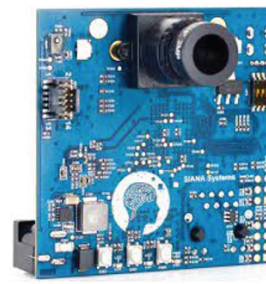
A broad spectrum of hardware architectures is available with various trade-offs to deploy machine vision NN models. Several architectures are under evaluation in terms of suitability for different machine vision applications and placement on the three edge levels. They are illustrated in Figure 4.4.4 and briefly presented in the following paragraphs.

OpenMV [11] is a small camera module on a microcontroller board that can be programmed in Python to implement applications using machine vision in the real world. It can detect colour and shape, frame differencing, face detection and more. For the experimental setup, the webcam capabilities have been enhanced with infrared and global shutter camera modules.

The former is to easily interface with the flare left in thermal imaging sensors for thermal vision applications. Combining machine vision with thermal imaging allows for better identifying objects to measure the temperature with great accuracy. Because of the modular design, the swapping of the standard lens for the long-range infrared imager can be done easily. The latter module allows the OpenMV Cam to capture high-quality greyscale images and not be affected by motion blur. The module can take snapshots on-demand with high frame-per-second speed.



OpenMV [11]



MPCam [12]

STM32MP157F-DK2 [13]

STM32H747I-DISCO(VERY) [14]

Figure 4.4.4 Hardware architectures under evaluation.

MPCam [12] is an intelligent camera system designed to bridge the gap between the development and rapid deployment of machine vision applications. The camera has, in addition to a Dual Arm® Cortex®-A7 core running up to 800 MHz and Cortex®-M4 at 209 MHz combined with a dedicated 3D graphics processing unit (GPU) and MIPI-DSI display interface and a CAN FD interface, an accelerator module that balances performance and cost and is therefore suitable for lab experiments as well as in the production line.

STM32MP1 [13] is a multiprocessor system that allows independent firmware to run on two computer cores (MasterArm Cortex-A7 running Linux based operation system and Arm Cortex-M4running RTOS). The latest Linux includes TensorFlow Lite (TFLite), so the development kit can run TFLite models. It has no dedicated computer unit for AI, and as such, inferences can only be performed using its CPU unit. However, the board can add an accelerator (such as Coral USB accelerator) to speed up the inferences of AI models. STM32MP1 is compatible with the Deep Learning STM32Cube.AI ecosystem.

STM32H747I-DISCO kit [14] is designed with STM32Cube.AI, an extension pack of the STM32CubeMX configuration and code generation tool and function packs for high performance AI applications. It is now possible to map and run pretrained networks on the board of the microcontroller using several AI solutions. The function pack for computer vision features examples of computer vision applications based on CNN, including an application for food recognition.

Regardless of the type of hardware architecture, the solution allows the import of trained neural networks and convert them into microcontroller code and run the inference directly on the microcontroller on edge. This reflects the AI paradigm shift, going from the cloud approach with high bandwidth, high centralised processing power, high latency, to more distributed AI, with lower bandwidth and reduced centralised computing power, more real-time response, and improved privacy.

4.4.7 Experimental Setup

In the first phase of the soybeans production optimisation, the specific objective is to evaluate variability in the morphological features of soybeans and classify soybeans according to selected features. The concept is to build an embedded intelligent vision system integrated into the production line as part of an advanced IIoT concept that can detect soybeans (wholes and fractions) and analyse their morphological features. The system can be used to detect variations that can lead to production process adjustments to improve final product quality and optimise the process in terms of energy reduction.

The embedded vision system is a flexible machine vision platform integrated into the IIoT system that will instantly, when powered on, display interactive results in real-time.

The OpenMV-based experimental setup currently under evaluation is illustrated in Figure 4.4.5. The system consists of multiple OpenMV nodes acting as machine vision IIoT devices. The OpenMV comes with a removable camera module, making the interface with different vision sensors possible. Some nodes are equipped with a global shutter camera module to capture fast action and eliminate motion blur, while others use the infrared camera module for thermal machine vision.

The nodes are mounted strategically on the production line (before and after the soybeans are cleaned out and after they are crushed). The machine

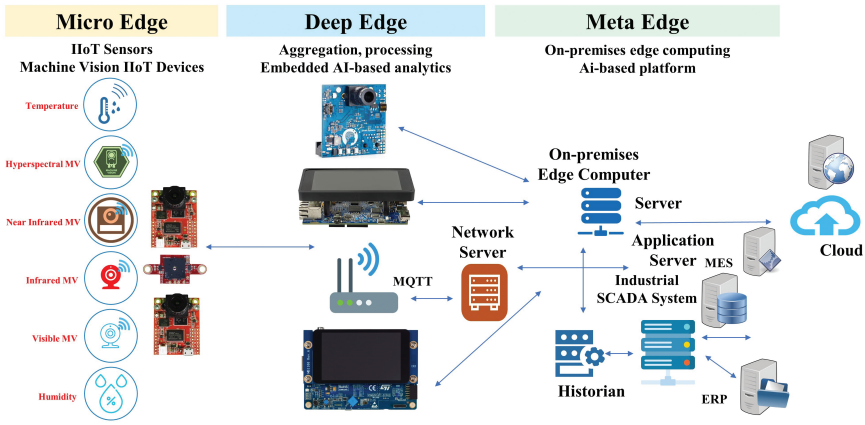


Figure 4.4.5 Experimental setup.

vision IIoT devices will be placed over the conveyor belt or in places that view the crushed soybeans.

The OpenMV machine vision IIoT devices are used not only as image sensors but also as AI-based processing nodes. The OpenMV IDE includes a Python-based interface to develop application code and programme the machine vision functions. The IDE is a robust editor and offers a frame buffer viewer to see what the camera sees, a serial terminal for debugging, and a histogram display for making object detection and tracking easy. The application is then sent as a script to the camera module, which is running MicroPython.

The OpenMV machine vision IIoT devices can run NNs on images, and deep learning NNs can run inference layers. As such, they do not need a network connection for inferences for the AI functionality. Some nodes are equipped with Wi-Fi modules using limited bandwidth to transmit via MQTT all protocol-specific measurements and results to the higher edge layers (deep and meta edge) for multi-sensor fusion and further processing.

The OpenMV offers competitive performance at low power consumption. Still, the nodes have limited flash memory (2 MB) required to store the firmware, and the NNs files. The memory can be expanded using an SD card, resulting in a slower inference output.

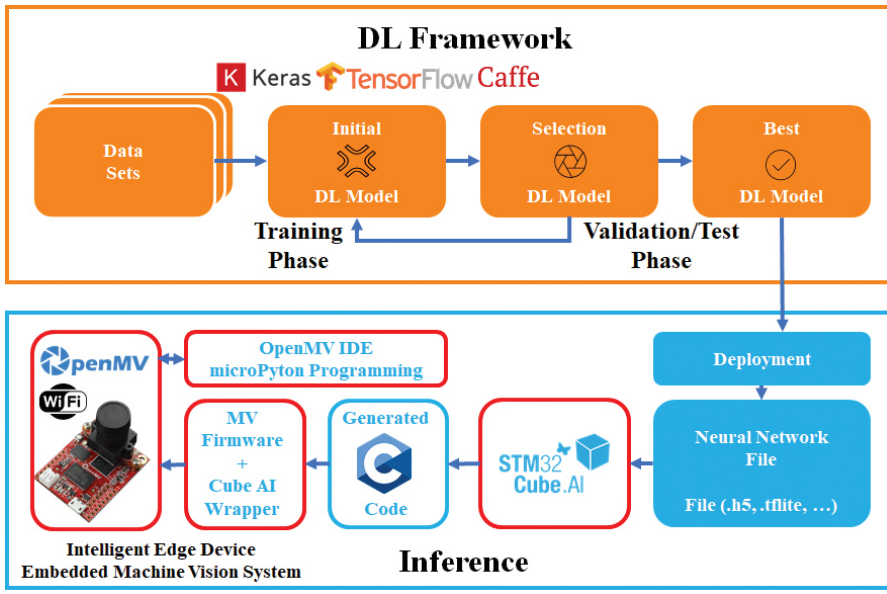


Figure 4.4.6 Training and inference workflow.

As the soybean application is relatively large, the approach is to optimise the size, and the optimisation flow under evaluation is presented in Figure 4.4.6.

The NN model's creation, training, and validation are performed using ML frameworks, and several tools are under evaluation (Keras, TensorFlow and Cafee). The trained NN model is then input to the STM32Cube.AI module and converted into optimised C code. Next, the firmware wrapped with the generated files and NN library is compiled, and the binary file is flashed onto the OpenMV target using IDE. The model is then used to programme the board (using microPython) and call the NN prediction function. The advantage of this workflow is that it performs the hardware level optimisation, and it also provides access to the software stack.

Notably, if the resulting optimised code still does not follow the hardware capabilities, the optimisation process is repeated with more compression. The process is about reaching a balance between avoiding opting for more performant hardware (resulting in increased costs) and not jeopardising the application's performance (e.g., results accuracy). Although not shown in the Figure 4.4.6, it is envisaged to use the above flow in a feedback loop, where

relevant runtime data is sent back to the framework to retrain the NN model and redeploy it in real-time back onto the microcontroller.

4.4.7.1 Experimental Evaluation and Results

In real-time, soybeans will move in a bulk fashion on the conveyor belt under machine vision IIoT devices. Both bulk and individual soybean samples must be considered. The preliminary experiments were conducted mainly on soybean samples to sense soybean colour, shape, and soybean amounts. The following guidelines govern the machine vision objectives:

Origin - Soybean size in the same load can vary, for example, due to different suppliers. There are relatively large variations in seed shape, size, and colour. Shape varies from almost spherical to flat and elongated. Seed size ranges from 5-11 mm and seed weight from 120-180 mg/seed. Soybean hulls can be yellow, green, brown, or black, either all one colour or a pattern of two colours [15]. For the use case presented in this article there are mainly two types of soybeans (originating from Canada and Brazil), and the former tends to be slightly larger than the latter.

Dockage fractions - A load also contains dockage fractions (including split soybeans due to breakage) that must be removed during the cleaning process. The percentage (%) of these fractions is an important indicator of soybean quality. The amount of broken soybeans smaller than halves should be determined.

Colour - Colour differences may relate to a moisture content variation.

Moisture - Investigating the impact of moisture content on the morphological feature classification of soybeans, individual and bulk, is important at different moisture content levels.

Crushed fractions - Soybeans are crushed and analysed. Each targeted fraction present in the sample should be distinguished based on images. Currently, this is performed manually based on three target values (3.36mm, 1.69mm, 0.84mm), resulting in four fractions: $>$ 3.36 mm, $>$ 1.69 mm, $>$ 0.84 mm, $<$0.84 mm. The results provide a measure of the decrease in soybean oil quality with increasing soybean breakage.

The challenge is the variation in soybeans' morphological features, which are extracted as attributes for classification using image processing techniques and neural networks. Around 50 data sets are collected with a fixed number of soybeans (60) randomly arranged in an imaginary cell size of 80 x 80 mm.

All images were captured with the OpenMV device and pre-processed within the IDE before saving them. The camera can capture up to 320x240 RGB565 images. The saved images are split into training and testing data sets and fed to training and validation. Various machine vision functions, including neural networks, are performed, on the images, including the following:

Boundary detection - This technique uses the Canny Edge Detector algorithm and simple high-pass filtering followed by thresholding. Boundary detection indicates the presence of dockage fractions before cleaning.

Colour tracking - The OpenMV device can detect up to 32 colours simultaneously in an image, and each colour can have any number of distinct blobs. OpenMV Cam will then determine the position, size, centroid, and orientation of each blob. Using colour tracking, the OpenMV device is programmed to track the soybeans on the belt, with colours set using the Threshold Editor.

Colour classifier - Although distinct colour variances between soybeans with different moisture content can be seen, preliminary results indicated that colour classification alone does not adequately describe the variations among different moisture content. NIR measurement is also needed.

Thermal and NIR water content analysis - The setup measures whole or cracked soybeans before drying, using infrared and NIR cameras. An infrared camera classifies soybeans at different moisture content levels using the thermal approach. The moisture content effects on the classification capability of colour, morphology, and textural features of imaged soybeans are evaluated. An NIR camera classifies soybeans at different moisture content levels using absorbance of water in the NIR spectrum.

The normal parameters measured on whole or cracked before drying and expanded soybeans flakes after drying are presented in Table 4.4.1 and Table 4.4.2, respectively..

Table 4.4.1 Normal parameters measured on whole or cracked soybeans before drying.

Parameter	Value
Water content in soybeans	11,0 – 13,5 %
Oil content in soybeans	18,0 – 21,5 %
Accuracy of measure	+/- 0,2 %
Temperature in the soybeans	5 – 30 °C

Table 4.4.2 Normal parameters measured on expanded soybean flakes after drying.

Parameter	Value
Target water content	9,5 %
Water content in soybeans	9,0 – 10,5 %
Oil content in soybeans	18,0 – 21,5 %
Accuracy of measure	+/- 0,2 %
Temperature in the soybeans	55 – 65 °C

Classification to detect variations on the production line - A TensorFlow NN for image classification has been trained, optimised, and deployed on the OpenMV. A convolutional NN trained on the collected image data set for detecting soybeans is investigated. This approach can give robust results even with significant variations. CNN are exponentially more accurate and efficient than traditional computer processing models for AI use cases like recognition, identification, and classification tools.

The results of the machine vision functions applied on various soybeans samples are shown in Figure 4.4.7 and Figure 4.4.8.

The soybeans images processed by a binary image filter are presented in Figure 4.4.9.

4.4.8 Summary and Future Work

The soybean production flow is complex, and the many process steps of soybeans impact the quality of the derived products and energy consumption. These steps can be improved and optimised by monitoring morphological features, such as moisture, size, shape, texture, and colour in soybeans and using variations in these features to adjust in real-time.

Figure 4.4.7 Boundary tracking for samples with impurities and split soybeans (left) and cleaned soybeans and crushed fractions (right).

Figure 4.4.8 Image detection segmentation and processing for samples of cleaned soybeans (left), soybeans with impurities (middle) and crushed fractions (right).

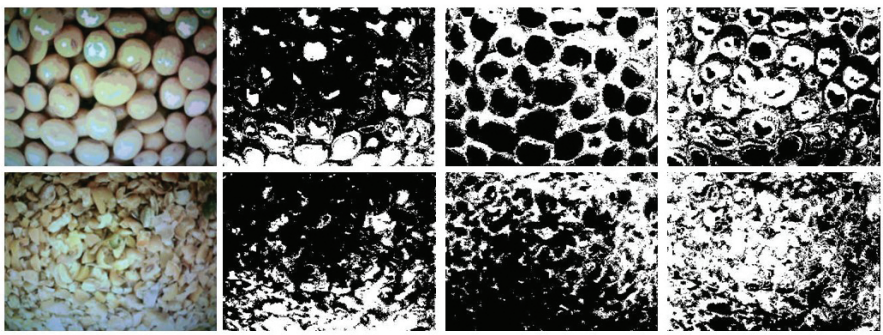


Figure 4.4.9 Soybeans images processed by a binary image filter.

This optimisation is made possible by employing RT-AIoT (a combination of AI technologies with sensing and machine vision IoT devices integrated into industrial infrastructure) to achieve more efficient real-time IIoT operations. With such an integration, human-machine interactions are improved, enhancing data management and analytics.

The system proposed for soybean process optimisation, based on RT-AIoT, includes a flexible machine vision embedded platform that displays results interactively into the IIoT system.

A broad spectrum of hardware architectures is available with various trade-offs to deploy machine vision IIoT devices at the edge. Several architectures are under evaluation concerning the suitability for different machine vision functions for the soybean optimisation process, such as boundary detection, colour tracking, thermal analysis, classification, and appropriateness for placement on the three edge levels.

Machine vision IIoT devices are used as image sensors, AI-based processing nodes and communication devices to run neural networks on images and transfer the information to the industrial process system.

The creation of the model, training and validation are performed using standard ML frameworks. The generated models can run on the microcontrollers if optimised in memory, processing, and power capabilities. It is a matter of balancing the goals of obtaining the most significant reduction in the size of the original code with a minor accuracy loss.

In preliminary results, it is assumed that by placing machine vision IIoT devices at different locations in the processing workflow (e.g., on the conveyor belt before cleaning, on the conveyor belt after cleaning and before weight, and after crackers before conditioning), better sensor and AI functionality can be obtained. In turn, an improvement in product quality and process efficiency can be achieved with such a procedure.

Preliminary experiments are being conducted on an experimental test bench, mainly on soybean samples, to sense temperature, moisture, colour, weight and volume. The following steps are envisaged to adopt the same AI functions for soybean bulk samples, validating the proposed machine vision IIoT system and further integrating it into the soybean industrial process. Another possible activity is identifying the optimal soybean moisture measurement method considering precision, ease of calibration, size, robustness, processing capabilities and cost. Thermal imaging for moisture detection in soybeans to increase production efficiency and reduce energy consumption is a challenging issue and will be explored in the next steps. The camera functions like a microbolometer, i.e. multiple heat-detecting sensors sensitive to infrared radiation from 700 nm to 1000 nm wavelength. By setting a maximum and minimum temperature range, the thermal camera can be programmed in the IDE to function as a sensor for seeing objects of a particular temperature. It is important to note that the camera does not really “see” moisture in soybeans; it can detect slight temperature differences and patterns that reveal the existence of water.

Finally, as the soybeans are moved in a bulk fashion on the conveyor belt, further work will focus on ensuring that the system is equipped with high-speed imaging cameras. Global shutter cameras, which are recording all image data simultaneously, are used to take pictures of soybeans on a conveyor belt. Preliminary simulations were performed with the OpenMV global shutter. Provided the exposure is short enough, the image has no motion blur on moving objects. However, the trend is to increase the exposure time to obtain more lighting on the camera and the best signal to noise ratio. The choice of the camera requires reaching a balance between increasing exposure time as much as possible (resulting in slightly higher levels of noise) and preserving the image accuracy, resolution and reliability, also allowing the algorithm to be programmed within the IDE.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] Vermesan, O., and Bacquet, J. (2020). Internet of Things – The Call of the Edge. Everything Intelligent Everywhere. Gistrup: River Publishers. Available online at: https://doi.org/10.13052/rp-9788770221955

[2] Vermesan, O., and Bacquet, J. (2018). Next Generation Internet of Things - Distributed Intelligence at the Edge and Human Machine-to-Machine Cooperation. Gistrup: River Publishers. Available online at: https://doi.org/10.13052/rp-9788770220071

[3] ECSEL AI4DI project. Artificial Intelligence for Digitising Industry. Available online at: https://ai4di.eu/

[4] Report on hybrid reference system level architecture design for the digitizing industry. AI4DI, D2.1, June 2020.

[5] Microwave technology. Moisture, dry substance and concentration - Microwave measurements, (Rev.02). Berthold Technologies GmbH & Co. October 2018.

[6] Jusoh, M.A., Abbas, Z., Hassan, J. Azmi, BZ, and Ahmad, A.F. A Simple Procedure to Determine Complex Permittivity of Moist Materials Using Standard Commercial Coaxial Sensor. ResearchGate, Measurement Science Review ů April 2011.

[7] Okamura, S. (2000). Microwave Technology for Moisture Measurement. Subsurface Sensing Technologies and Applications, 1, 205–227 https://doi.org/10.1023/A:1010120826266

[8] Pandey, T., Bhuiya. T.K., Singh, R., Singh, B., Harsh, R. (2012). A Review on Microwave Based Moisture Measurement System for Granular Materials. IOSR Journal of Electronics and Communication Engineering (IOSR-JECE), Volume 3, Issue 2 (Sep-Oct. 2012), 37-41.

[9] Zambrano, M.V., Dutta, B., Mercer, D.G., MacLean, H.L., Touchie, M.F. (2019). Assessment of moisture content measurement methods of dried food products in small-scale operations in developing countries: A review. Trends in Food Science & Technology, Volume 88, 484-496. Available online at: https://doi.org/10.1016/j.tifs.2019.04.006

[10] A. Kerr, H. Rafuse, G. Sparkes, J. Hinchey, and H. Sandeman. "Visible/Infrared spectroscopy (VIRS) as a research tool in economic geology: Background and pilot studies from Newfoundland and Labrador". Newfoundland Department of Natural Resources, Geological Survey, Report 2011-1, pp. 145-167, 2011. Available online at: https://www.gov.nl.ca/iet/files/mines-geoscience-publications-currentresearch-2011-kerrvirs-2011.pdf

[11] OpenMV. Available online at: https://openmv.io/

[12] MPCam. Available online at: https://www.siana-systems.com/mpcam

[13] STM32MP157F-DK2. Available online at: https://www.st.com/en/evaluation-tools/stm32mp157f-dk2.html

[14] STM32H747I-DISCO(VERY). Available online at: https://www.st.com/en/evaluation-tools/stm32h747i-disco.html

[15] Soybean seeds. (2009) Feedipedia. Available online at: https://www.feedipedia.org/node/42

4.5

AI and IIoT-based Predictive Maintenance System for Soybean Processing

Ovidiu Vermesan[1], Roy Bahr[1], Ronnie Otto Bellmann[2], Jøran Edell Martinsen[2], Anders Kristoffersen[2], Torgeir Hjertaker[2], John Breiland[3], Karl Andersen[3], Hans Erik Sand[3], Parsa Rahmanpour[4] and David Lindberg[4]

[1]SINTEF AS, Norway
[2]DENOFA AS, Norway
[3]NXTECH AS, Norway
[4]INTELLECTUAL LABS AS, Norway

Abstract

This article presents an industrial predictive maintenance (PdM) system used in soybean processing based on artificial intelligence (AI) and Industrial Internet of Things (IIoT) technologies. The PdM system allows for the continuous monitoring of relevant production equipment/motor parameters, such as vibration, sound/noise, temperature, and current/voltage. It is designed to identify abnormalities and potentially break down situations to prevent damage, reduce maintenance costs and increase productivity. Condition monitoring is combined with AI-based methods and edge processing to identify the parameter changes and unusual patterns that occur before a failure and predict impending failure modes well before they occur. The PdM demonstrator currently under evaluation is planned to integrate intelligent IIoT-based sensors to measure parameters, convolutional neural network and Wi-Fi, LoRaWAN, Bluetooth low energy (BLE) technologies for intelligent communication.

Keywords: predictive maintenance, artificial intelligence, smart sensors systems, edge computing, industrial internet of things, industrial internet of intelligent things, soybeans manufacturing, vibration analysis, condition monitoring, machine learning, deep learning.

4.5.1 Introduction

Artificial intelligence (AI), Industrial Internet of Things (IIoT) and edge computing combined with intelligent sensors and actuators are enablers for digitising industry and driving the development of new technologies for the Industrial Internet of Intelligent Things (IIoIT). The advancement in these technologies brings additional intelligence at the edge that empowers IIoIT devices with more intelligent decision making, high performance, low power processing and built-in security to create more intelligent and adaptive industrial applications.

As the intelligent capabilities of the IIoT devices expand, industrial systems become more efficient, interactions become more seamless and IIoT devices become capable of detecting anomalies and potential failures sooner.

The manufacturing infrastructure, equipment and industrial products integrate novel components (e.g., CPUs, GPUs, AI accelerators, neuromorphic processors) that support AI operations and capabilities, allowing intelligence to be moved to the edge. Integrating edge distributed intelligent sensors/actuators and AI methods and techniques into industrial process flows accelerates the digitising of industry and improves manufacturing processes (i.e., lower cost, less energy consumption, higher yield, and quality).

Furthermore, the progress in equipment monitoring accelerates the transition of maintenance operations from preventive maintenance (PvM) towards predictive maintenance (PdM). These developments further advance cost reduction, machine fault reduction, repair stop reduction, spare parts inventory reduction, spare part life increasing, increased production, operator safety, repair verification and overall profit.

Soy is a predominant ingredient in the food industry. Soybean production and the maintenance of equipment in the soybean production line can be improved through optimisations and reductions in downtime, repair costs and additional labour costs and requirements.

Predictive quality analytics using AI is helping soybean production facilities gain control over the equipment. Predictive analytics substantially helps to:

- Detect production equipment/motor anomalies and failures.
- Predict abnormalities and faults.
- Redefine and define error classes.
- Find factors that hamper productivity.

IIoT and intelligent sensors/actuators integrated with different AI techniques offer benefits to PdM solutions in soybean processing and manufacturing. These benefits include detecting faults early and accurately, predicting the remaining useful lifetime of an equipment/motor given an operational context or even prescribing guidance on work scope for the field service team with recommendations regarding the parts and personnel skills desired to service them.

The equipment and motors of the soybean production facility have little or no communication with the SCADA control system. As a result, it is challenging to determine the actual fault that causes a stop to the equipment without remote monitoring. Combined with AI-based techniques, placing various sensors and IIoT devices on the equipment to monitor critical parameters help identify abnormalities and potentially break down situations that reduce the production's unforeseen downtime. Some of the typical parameters to monitor are vibration, sound/noise, temperature/thermography, and current/voltage. In addition to the real-time measurements of these parameters, an analysis of the rate of change of the machine condition can provide valuable information for estimating warning levels and absolute limits before failure. Sensor data collection can be carried out in parallel for similar machines by building an AI/ML/DL model that can predict how much the mechanical machine components have deteriorated. The model can also determine which sensors should get the most attention to increase the sampling frequency and/or length of sampling interval. Several measurement technologies also reduce the possibility of false positives and false negatives (i.e. an indication for machinery when it is not deteriorated and no indication when a warning should have been received from the inference).

The article is organised as follows. The next sections present the elements describing the maintenance foundations in industrial production facilities and principles of PdM. Soybean production process and maintenance policies are described in the next section, followed by the description of the AI-based PdM framework methodology. Afterwards, the section on

integrated industrial system for the maintenance of soybean production equipment describes the current approaches and elements. The experimental set-up section depicts the overall architecture, the specific experiments performed and results. Finally, the last section concludes and highlights the next steps.

4.5.2 Maintenance Foundations

Maintenance is defined by the standard European Standard EN 13306 as "the combination of all technical, administrative and managerial actions performed during the life cycle of an item intended to retain it in or restore it to, a state in which it can perform a required function" [1]. A maintenance management plan is required to perform the maintenance operation.

Maintenance management is defined as the sum of all the management activities that determine the maintenance objectives, strategies and responsibilities, and implementation through maintenance planning, maintenance control, and improvement of maintenance activities. Regular maintenance is critical to keep the equipment/motors and the work environment safe and reliable. Several types of maintenance are defined by the EN 13306 standard and illustrated in Figure 4.5.1.

The classification includes the following maintenance types [1]:

Reactive Maintenance (RM) is a run-to-failure maintenance management method, offering maximum production output of the equipment by using it to its limits. The maintenance action for repairing equipment is performed only when the equipment has broken down or been run to the point of failure. The cost of repairing or replacing a component would potentially be more than the production value received by running it to failure.

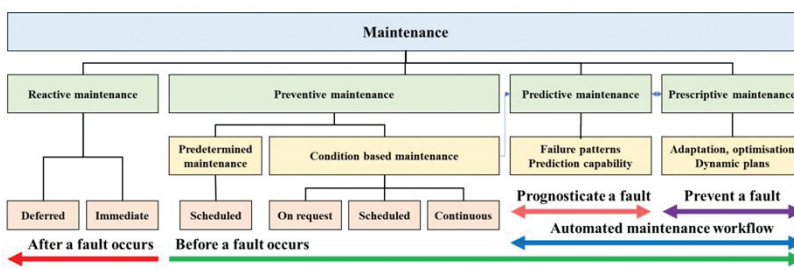


Figure 4.5.1 Maintenance types. Adapted from EN 13306 Standard [1].

Preventive Maintenance (PvM), time-based, or scheduled is a maintenance procedure conducted periodically with a planned schedule in time or process iterations to anticipate process/equipment/motors failures. The main aim is to improve the efficiency of the equipment/motors by minimising the failures in production preventive maintenance works usually under a pre-existing maintenance schedule provided by the equipment's manufacturer The procedure is used in many different industrial processes to avoid failures. However, the method requires several corrective actions that can lead to increased operating costs.

Condition-based Maintenance (CBM) is defined as a method based on constant equipment/motors monitoring or their behavioural health that can be performed when they are necessary and not planned. The maintenance actions can be performed when the actions on the process are taken after one or more conditions of degradation of the equipment/motors.

Predictive Maintenance (PdM) is based on the continuous monitoring of the equipment/motors to detect trends in the health of a machine and using prediction tools, models, and algorithms to predict when failure occur and estimate when such maintenance actions are required, and maintenance scheduled.

Prescriptive Maintenance (PsM) uses sensors, data, and advanced analytics to determine the root cause of a potential failure so specific corrective action can be prescribed. A fully proactive/prescriptive maintenance implements the following tasks. The workflow starts with *detection* – measuring machine vibrations and making comparisons to the baseline or previously measured data to determine changes in condition. If significant changes occur, the data are analysed to identify problems and prepare maintenance timelines. *Analysis* involves evaluating the relationship between phase, frequency and amplitude in the data collected from various sensors to deduce the symptoms that identify the root problem. If necessary, maintenance or corrective repairs are scheduled. Additional measurements are taken to *Verify* that the problem has been fixed. Finally, data history is studied to determine the *Root Cause* so that it can be avoided if the problem is recurrent. PsM solutions are in the first stages of evaluation and the implementations still require increased complexity and costs.

The solution presented in this article focusses on a PdM system approach that does not involve the prescriptive part, which is envisaged in the next development steps. Figure 4.5.2 illustrates the comparison between PdM,

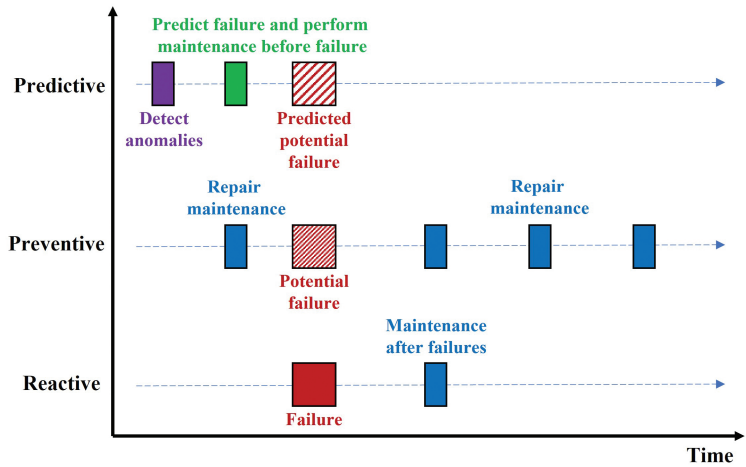


Figure 4.5.2 Failure and maintenance timing. Adapted from [8].

PvM and RM types in terms of maintenance plans and intervention timing and Figure 4.5.3 shows the comparison in terms of cost.

The maintenance types display different trade-offs between repair cost and prevention cost. PvM has the lowest repair cost – due to well-scheduled downtime – but has the highest prevention cost, while RM has the lowest

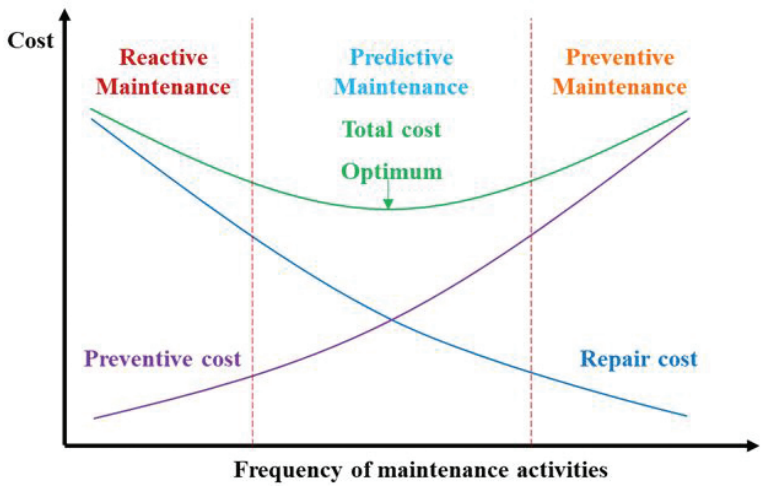


Figure 4.5.3 Comparison of maintenance cost and frequency of maintenance. Adapted from [3].

prevention cost – due to using run-to-failure management – but has the highest repair cost. PdM's goal is to predict when the equipment is likely to fail and to decide which maintenance activity should be performed. Therefore, PdM can achieve the optimal trade-off.

Although PdM supersedes both PvM and RM, the total cost of its condition monitoring devices (e.g., sensors, IIoT devices) is often higher. PdM systems become increasingly complex to detect failures in early stages, using real-time IIoT data, historical data, prediction tools such as AI, machine learning (ML) methods, feature extraction from sensor and monitoring analysis, model-based condition analysis, statistical inference approaches and engineering techniques.

For the soybean production the focus is on predictive maintenance using condition-based monitoring and AI-based techniques for the prediction algorithms.

4.5.3 Principles of Predictive Maintenance

PdM collects data from IIoT sensors and devices connected to machines and processes the data through predictive algorithms to discover trends and identify when the equipment needs to be repaired or retired.

The principle of PdM is to use the actual operating condition of systems and components to optimise operation and maintenance – neither running the equipment to failure, nor replacing it when it still functions. Maintenance is conducted only when necessary. The motors requiring maintenance are identified in time and are shut down only just before imminent failure occurs, a decision that reduces the time and money spent on maintenance, minimising the production hours lost to maintenance as well as the cost of spare parts and supplies. Maintenance is scheduled when specific conditions are met and before the equipment/motors break down.

When PdM is used in industrial processes, such as soybean production, maintenance is performed by observing specific parameters or components (e.g., equipment, motors) of the system or production line. The advantage of this procedure is that the system is controlled in real time based on the monitored parameters. The equipment/motors in the production systems have an operating curve that is well defined by the manufacturer.

PdM has the possibility of detecting potentially critical situations with the equipment/motors that lead to serious consequences situations before they arise. A cost-benefit analysis is conducted before deciding if PdM is profitable and preferred for a specific motor.

The performance of a PdM system – deciding which machines to keep running and which to schedule for maintenance – depends on the accuracy of the information gathered from various sensors, IIoT devices and the algorithms' ability to interpret that information i.e., the system's intelligence. CBM enables real-time evaluation of machine health and triggers alarms (e.g., by indicating excess vibration or temperature) so that immediate corrective action can be taken to avert failure.

There is a dependency between PdM and CBM and PsM. CBM can be standalone without a PdM in place, but PdM relies on CBM in collecting, comparing, and storing measurements that determine a machine's health. Also, PdM is part of a proactive (prescriptive) maintenance approach but is not necessarily a fully proactive/prescriptive system: it does not guarantee that the root causes of problems and failures are eliminated.

4.5.4 Soybean Production Process and Maintenance Policies

The predictive maintenance policies for soybean production are centred on improving the efficiency of the equipment/motors utilisation, reducing the down time, estimating the remaining useful lifetime of the equipment/motors, and reducing the overall maintenance costs.

The approach used for the soybean production process is based on condition-based monitoring implemented using various sensors, IIoT devices that allow a continuous monitoring process of relevant equipment/motors sensor parameters. Condition based monitoring is combined with AI-based methods and edge processing to identify the parameter changes that occur before a failure and predict a future period in which the parameter changes appear, and thus the failure might happen.

The policies adopted are based on the production manufacturing goals, the selected category for conditioning monitoring, the maintenance scope, fault detection categories, manufacturing system size, predictive AI-based techniques, data handling and the evaluation approach. A short description of these different categories selected for the soybean production process and maintenance policies is presented in the following paragraphs.

The predictive maintenance system combining CBM, and AI is aimed to minimise the downtime of the soybean production line as it allows to plan maintenance actions and group-specific maintenance actions to reduce the number of production stops for single maintenance actions.

Minimising downtime helps reducing costs and increase productivity. The goals are aligned with the non-functional requirements (NFRs) for the implementation, such as reliability, compatibility, and maintainability according to the standard ISO/IEC 25010 (SQuaRE - Systems and software Quality Requirements and Evaluation) [4].

The selection of NFRs such as maintainability and reliability emphasises the importance of preserving the system's capabilities over the operational lifetime. The reliability aims in improving individual components and providing redundancy. Maintainability enhances the maintenance measures to implement and improve preventive maintenance, apply predictive maintenance measures, and increase repair capability and speed.

The soybean production PdM system aims to evolve from inspection-based monitoring to sensor-based continuous online monitoring in real-time. With sensor-based monitoring, various IIoT sensors and devices monitor vibration, temperature/thermography, sound/noise, current/voltage parameters and collect the relevant data. Continuous collection of relevant monitoring data is used to identify the running state and estimate the useful life of the equipment/motors.

Figure 4.5.4 shows the operative curve slope of the machine condition dependency on the life cycle of the equipment that is typical for industrial motors and hence applicable in our case. At the failure inception point, the machine's condition starts to deteriorate, and various sensor modalities (such as vibration, temperature, sound and current) can reveal conditions that indicate the machine's potential for failure. The combinations of parameter measurements associated with specific failure modes, such as motor imbalance, misalignment, loose coupling and degraded bearings, are valuable data. These data sets are used as input to supervised learning algorithms (such as decision trees or neural-network models) to later predict those failure modes from real-time sensor data collected from motors.

As illustrated in Figure 4.5.4, a vibration analysis is typically an indicator of machine health. It enables the early detection of a sudden failure and helps to eliminate downtime due to such a failure. A well-designed PdM system uses a combination of several sensor modalities to determine the time elapsed from the detection of deterioration symptoms to the failure of the equipment. For example, increased current consumption, noise or heat typically suggests a shorter potential to failure time interval for most motors.

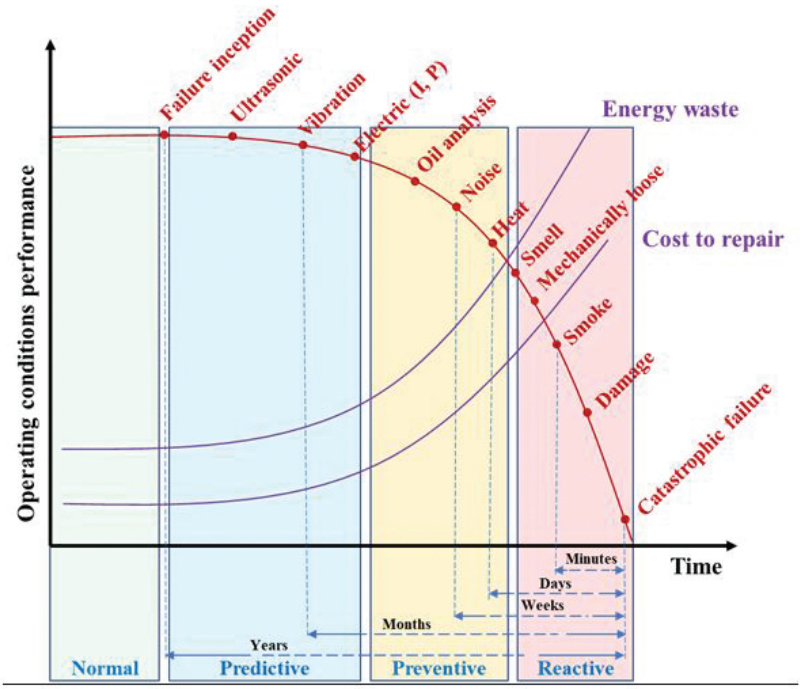


Figure 4.5.4 Parameters monitored during equipment life-time operation. Adapted from STMicroelectronics.

In the case of soybean production, vibration, sound, temperature, and current modalities, are monitored as part of the CBM and integrated into the PdM system.

The soybean production predictive maintenance system focuses on the prediction of the future conditional state of equipment/motors to schedule maintenance activities in an appropriate way and scope, and provide fault detection, attempting to predict the remaining life of the equipment/motors and in the future to identify the root cause of the failure based on the collected data.

The future activities are targeting the processing of acquired monitoring data to reveal the reasons for future failure. The feasibility and accuracy of a fault detection approach depend on the monitoring activity level, which means that the more equipment/motors parts and components are monitored separately, the better can be identified where the root cause for a future failure is.

Manufacturing system size for the soybean production PdM is applied to single-component equipment/motors. Further work could focus on multi-component systems. The implementation of PdM system for these multi-component systems requires increasing the number of monitoring devices and processing data and dependencies between the equipment/motors' components.

PdM incorporates a combination of monitoring techniques such as ultrasonic, vibration, noise/sound, temperature/thermography, and motor parameters (e.g., current, voltage, load).

The selection of the sensors is important as the sensors can detect certain faults and failures. Several types of parameters and transducer types are considered for monitoring the equipment/motors:

- Vibration is measured using accelerometers based on piezo transducers with low noise measuring frequencies up to 30kHz to identify bearing conditions, gear meshing, misalignment, imbalance, and load conditions. MEMS accelerometers offer low cost/power/size solutions for vibration measurements up to 20kHz.
- Sound pressure is measured using low cost/power/size microphones detecting sound with frequencies up to 20kHz for identifying bearing conditions, gear meshing, pump cavitation, misalignment, imbalance, load conditions or ultrasonic microphones detecting sound with frequencies up to 30kHz.
- Currents up to 150 A are measured using a clamp-on transformer with wireless capabilities (e.g., LoRaWAN).
- Temperature is measured using thermocouple or thermistor sensors (e.g., temperature range from 0 to 85℃. An infrared camera is considered for taking thermographic images of the motors.

4.5.4.1 Vibration Analysis

Predictive maintenance incorporates a combination of monitoring techniques, such as vibration, noise/sound, temperature/thermography, and motor parameters (e.g., current, voltage and load). Vibration analysis stands out due to the multitude of problems that can be discovered and rectified through it.

Vibration is adaptable to various machines and indicates overall machine condition and problem severity, and analysis provides information on specific faults.

A vibration analysis is the best indicator of the condition of motors with rotating parts. It is considered a direct measurement for detecting and monitoring imbalance, misalignment, and looseness of a rotating part. Machines/motors vibrate regularly when operated, and a low vibration level normally indicates that the equipment is running correctly. When vibration begins to increase, the machine may be about to fail.

Vibration amplitude can be expressed in units of displacement, velocity or acceleration. Displacement is a measurement of the linear movement in the signal as the machine oscillates back and forth. Velocity is the speed of the signal as the machine oscillates back and forth. Acceleration is usually compared to the gravitational acceleration in the signal at the instant the oscillation changes direction. In summary, displacement is the peak-to-peak movement of the vibrating part. The velocity is the speed at which displacement occurs, and acceleration is the rate of velocity change.

Vibration is the motion of machine components caused by dynamic forces. It refers to the mechanical oscillations around an equilibrium point. The oscillations may be periodic, random, or transient. Transient vibration appears, for example, when pump cavitations occur due to an improper system line-up.

Vibration is described by amplitude (typically velocity), time, frequency, and phase. Vibration is measured by transducers that convert vibration motion (e.g., an accelerometer to measure g-force on a 3-axis and then convert speed and frequency into an electrical signal for processing), vibration meters that detect only amplitude (no frequency components) and vibration analysers that convert amplitude versus time to amplitude versus frequency (spectrum analysis).

Faults can be detected early using a full signature (spectrum) analysis, frequency analysis parameter sets and overall vibration levels (no specific faults can be detected via this method). Types of faults that can be detected include misalignment, looseness, bearing defects or wear, unbalance, internal component rubbing and resonant structural conditions.

Different vibration sensors are under evaluation for the implementation including two high-performance accelerometer-based sensors with very low noise operation (45 $\mu g/\sqrt{Hz}$ $\pm$2g, and $\pm$10g), 3D accelerometer + 3D Gyro inertial measurement sensor, and an ultra-wide bandwidth (up to 6 kHz) low-noise 3-axis digital vibration sensor.

4.5.5 AI-based Predictive Maintenance Framework Methodology

The AI-based predictive maintenance framework includes the design and development of an intelligent multi-sensors wireless system that comprise the following steps:

- Define the system architecture.
- Find sensors that measure and collect the required physical parameters with the correct accuracy and stability at the right price and availability.
- Determine the required processing microcontroller specifications, including computational power, memory, interfaces, and AI-based capabilities.
- Choose the connectivity and communication protocols technologies.
- Design the power management, form factor and integration into the industrial system.
- Outline the edge integration strategy and the overall collection of information flow.
- Develop the AI-based models and algorithms.
- Implement the required analytics and characterise the system.
- Validate the AI-based system in the real application scenario.

The AI-based algorithms are fed with data gathered to monitor the motors/equipment parameters and train models to identify possible anomalies.

The architecture used for predictive maintenance allows the implementation of edge machine learning, using intelligent capabilities of IIoT devices, which can be deployed to run AI models directly on the motors/equipment.

The IIoT devices collect the information from sensors in the edge node in real-time, allowing continuous monitoring of the motors/equipment operations. Data is processed in the cloud and locally at the edge from machine learning predictive models, detecting anomalies.

The overall AI-based predictive maintenance framework used for the soybean production is illustrated in Figure 4.5.5. The AI-based models can be deployed at different micro, deep and meta-edge levels as illustrated in Figure 4.5.6. The work described in this article consider the case of the deployment of the AI-based models at the meta-edge level.

AI-based PdM refers to the ability of a PdM system to use knowledge and sensor data to anticipate and address potential issues before they lead to breakdowns in operations, processes, services, or systems. In the context of the soybean production demonstrator, three AI-based techniques have been

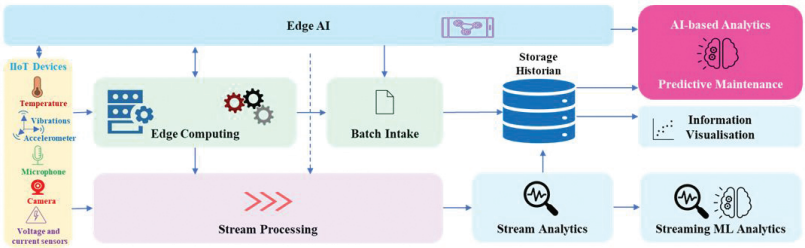


Figure 4.5.5 AI-based predictive maintenance framework.

explored: knowledge-based, ML-based and DL-based approaches. A more comprehensive survey of current AI approaches for PdM can be found in [9].

Knowledge-based approaches make use of domain expert knowledge and deductive reasoning, of which expert systems and model-based reasoning are two representative examples.

Expert systems typically consist of a knowledge database and an inference engine. The knowledge database contains the human domain expert knowledge represented in a form that can be processed by machines. For example, rules are structured in an "if A, then B" format. The inference engine consists of algorithms, which, via step-by-step inferences, draw deductions based on the knowledge rules.

Model-based approaches are applicable in cases where physical processes that have an impact on the health of the equipment can be simulated using mathematical models. The advantage of these approaches is that they are effective and accurate, and models can be reused. However, complex systems cannot always be approximated precisely using explicit mathematical models.

Knowledge-based approaches are feasible when there is a lot of human expert knowledge and experience that can be modelled but not enough data. On the downside, knowledge bases take time to acquire and represent on computers, and if some knowledge is missing or incomplete, a less reliable result will be produced.

Machine learning (ML)-based approaches are useful when domain knowledge and experience is scarce, but vast amounts of data are available, allowing ML algorithms to search for large patterns and extract useful knowledge. ML algorithms developed for the context of PdM include Artificial Neural Network (ANN), decision tree (DT), Support Vector Machine (SVM), k-Nearest Neighbours (k-NN).

These approaches typically involve feeding a neural network with data (images, vibration, audio, etc.), and the network thus trained would then

be able to accurately guess the motor diagnosis when fed with real-time sensor data.

The advantage of ML-based approaches is that they do not rely on a domain expert's knowledge and are able to handle large amounts of real-time sensor data, thus allowing for automation. ML-based approaches also have limitations depending on the use case application of the PdM. A survey of ML methods applied to PdM, their challenges and opportunities can be found in [2].

Deep learning (DL)-based approaches have been proven to be superior to ML-based approaches in the field of PdM. For example, Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) are widely applied.

It is technologically possible to combine the above approaches to obtain the best trade-offs. For the demonstrator, a combination of regression methods and statistical techniques, along with CNN or RNN, is under evaluation.

The choice of DL algorithms for the demonstrator is justified by their abilities in feature learning and prediction involving multilayer nonlinear transformations. CNN can extract local features of the input data and combine them layer by layer to generate high-level features. The CNN structure consists mainly of input layer, convolution layer, pooling layer, and fully connected layer [9][7][10]. The steps involved are data collection, the data pre-processing, the data transformation [6], and the CNN model creation [5].

Preparation of the training data requires analysing the following information sources: real-time data from the IIoT monitoring devices, the motors/equipment fault history, including the description of the error events, the failure scheme that contains a sufficient number of failure cases, motors/equipment maintenance/repair history including information about replaced components, predictive maintenance tasks performed, and the motors/equipment conditions to estimate the life-time. The data collected should contain time-varying functions that acquire ageing patterns or any anomaly that could cause performance reduction.

For training/learning the data pre-processing requires to create/construct the datasets, create the features, and the anomalies and normalise the data sets that they can be used for training. For supervised anomaly detection ML models, creation of data sets for training and testing are both needed.

The DL models can identify an anomaly, and the edge device sends a notification to signal that was recognised a different function

pattern (e.g., different current consumption, increased operating temperature, alternative operational state, different vibration, and sound patterns, etc.).

Whereas model training is primarily done in cloud, model inference is performed at the edge and on the devices to allow for information to be captured and analysed without transferring across network communication protocols or storing in cloud infrastructure.

4.5.6 Industrial Integrated System for Soybean Production Equipment Maintenance

The architecture proposed takes into consideration that the IIoT sub-systems are connected to different edge gateways, and then the information is aggregated to an on-premises edge server as presented in Figure 4.5.6. The edge computing solution proposed is to improve the performance, security, operating cost, and reliability of IIoT and AI-based platform, applications, and services.

The system design is based on a heterogeneous wireless sensor network that consists of sensor nodes and IIoT devices with different communication interfaces (e.g., BLE, LoRaWAN, Wi-Fi), computing power, sensing range and AI-based processing capabilities.

The system implements an architecture integrated at micro, deep and meta-edge levels, allowing heterogeneous wireless sensor networks to communicate with the various gateways while integrating the information

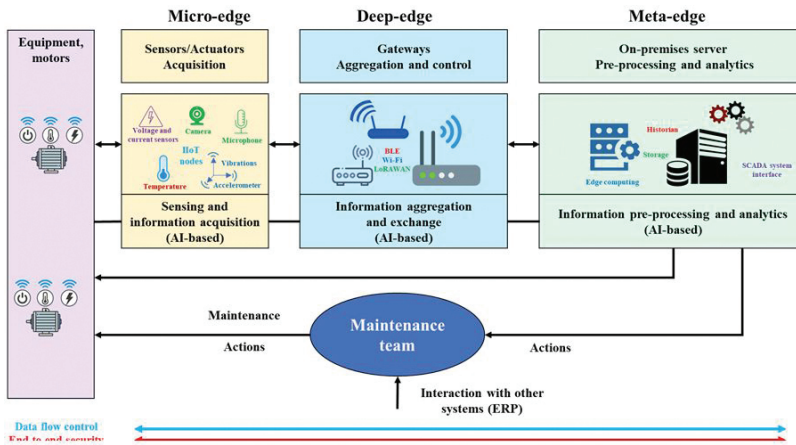


Figure 4.5.6 Industrial integrated system for equipment maintenance.

from heterogeneous nodes in a shared on-premises edge server application and a shared database. The network architecture allows for interfacing with the existing SCADA system and providing a secure link to external cloud applications.

The micro-edge implementation increases the information acquisition from the intelligent edge sensors placed on equipment/motors and allows end-users to build predictive maintenance solutions based on advanced anomaly detection algorithms.

The heterogeneous architecture provides the ability to retrieve data from LoRaWAN and Wi-Fi wireless sensor nodes using for example the MQTT protocol. The architecture has several advantages related to integrating data from heterogeneous sensor nodes and providing a mechanism for their transmission to an on-premises edge computing server and creating geographically distributed wireless sensor nodes over the production facility.

LoRaWAN network is deployed in a star topology, where the end nodes communicate with the LoRA gateways. Information received by LoRa gateways is sent to the LoRA network server and the application server using an IP-based backhaul network. The components integrated into meta-edge are detailed in Figure 4.5.7.

AI models can run on the edge server, considering the ability to use the Kubernetes platform on-premises.

Two or three nodes (servers/processing units) can be connected to the master node running the "brain" of the Kubernetes. A cluster of nodes/resources need to be created virtually by sharing the available CPUs and RAMs using the on-premises edge server.

The edge server interfaces include protocols such as MQTT, HTTPS using RESTful API, OPC UA, etc.

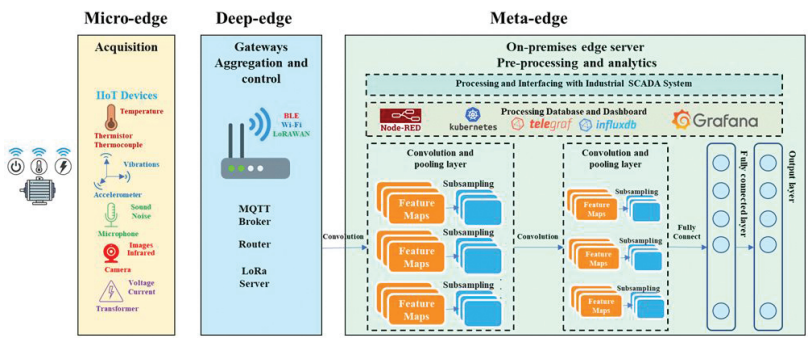


Figure 4.5.7 Soybean production predictive maintenance system demonstrator.

The soybean manufacturing facility produces soy oil, lecithin, and meal. In the soybean production line, the hammer mill located in the crushing area of the plant is defined as one of the most critical equipment to monitor and prioritised in the predictive maintenance use case.

The lumps in the soybean meal are crushed in two parallel hammer mills, and the meal is transported to the soybean meal storage. The hammer mills are only inspected visually by operators two times a day and have no communication back to the control system. The hammers and shafts get worn repeatedly, and trained operators can sometimes hear the weariness before a breakdown. The hammer mills sound picture is an important characteristic that can indicate an imminent accident.

The experimental set-up comprises of the physical/field part that includes the equipment/motors and the IIoT sensors and actuators. The control layer includes different types of network and domain controllers, PLCs, communication IIoT gateways, etc. The operation and information layers include backbone network, clients (e.g., OPC), edge server(s), and data storage.

Predictive maintenance demonstrator performs maintenance based on the motors/equipment health status indicators. The IIoT-based sensors are used to measure unusual patterns of motors parameters, such as motor's vibration level, temperature, current consumption, and, based on experience, failures are preceded by an unusual pattern of these parameters. A convolutional neural network (CNN) DL technique capable of extracting data representation is planned for the demonstrator that is integrated in the AI-based model and the algorithms developed.

CNN deep learning is proposed due to its shared weights and the ability of local field representation to extract the input sensors/IIoT data features and combine them layer by layer to generate high-level features. The CNN structure consists of the input layer, convolution layer, pooling layer, and fully connected layer.

CNN can extract valuable and robust features from IIoT and sensor monitoring data such as raw vibration signals to identify fault types. For example, the vibration signals can be converted to discrete frequency spectrum via Fast Fourier Transform (FFT) and use CNN to analyse the spectrum-principal-energy-vector and obtains a series of eigenvectors. Next, a CNN model can be used for regression prediction. RNN deep learning is another method evaluated in the project that includes feedback connections in ANN architecture, accounting for past input state influence to the current

network output. Compared with the simple feedforward architectures, the training of the RNN architecture is a much more complex task.

The edge computing approach is integrated and interfaced with industrial SCADA infrastructure and linked through the historian component.

Consolidating the historian, SCADA, and HMI applications alongside new containerised functions for PdM using AI-based models and algorithms alongside an IIoT stack supports the processing at the edge and the AI deployment.

4.5.7 Experimental Set-up and Implementation

The overall architecture and role of the different components, technologies and protocols that constitute the PdM system is depicted in Figure 4.5.8. The evaluation approach for the soybean production PdM system is based on specific experiments conducted to validate the approach. The experiments are limited to test setups suitable to demonstrate the concept. The aim is to scale up the system as the experiments validate the different solutions.

As vibration analysis is the most common technique of PdM programs in the industry, this section focuses on experimentation related to the vibration parameter. The deployed HW/SW predictive maintenance solution measure and analyse vibrations to detect abnormal behaviours of the motors/equipment, with AI-based techniques for detecting operating anomalies before a failure occurs.

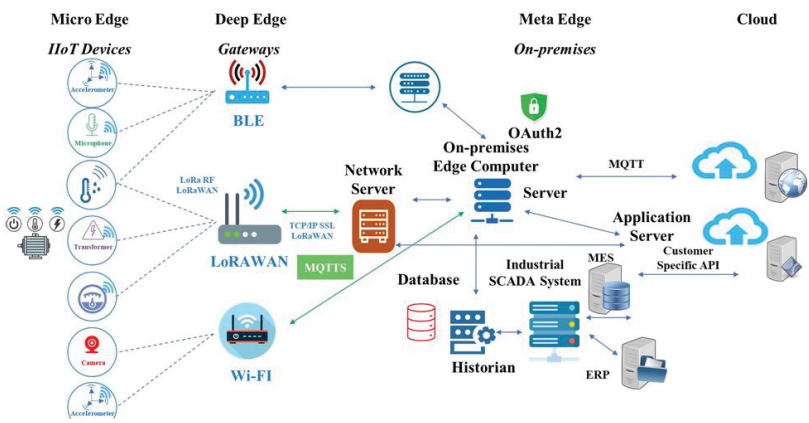


Figure 4.5.8 Overall architecture.

The inference is applied as well to a class of devices based on microcontrollers (e.g., STMicroelectronics microcontrollers) that have AI-based components and can implement semi-supervised learning engine that aggregates data from sensors, identify and create a reference behavioural profile of the motor/equipment, then detects and acts upon anomalous/abnormal behaviour.

Every machine component produces a specific type of vibration signal, which, when displayed in the vibration spectrum, often forms characteristic patterns. Pattern recognition is a key part of vibration analysis, but significant training and experience are necessary to read patterns.

The experimental setup uses a four item software stack: Node-RED [15] and MQTT [16] to collect vibration measurements from two IIoT devices placed on an AC test motor; and InfluxDB [11] and Grafana [13] to store the data into a database and query the database to build dashboards and create visualisations of the data in the form of charts, graphs and more.

Node-RED implements various automation logic, while InfluxDB is preferred over other databases (such as MySQL); a timestamp is automatically added when data are pushed into the database. The experimental setup is detailed in Figure 4.5.9.

An MQTT broker (e.g., Mosquitto [12]) is installed on a separate server. The IIoT devices are connected to the broker using a 2.4 GHz band Wi-Fi connectivity protocol. An analysis of the load on different channels was conducted before selecting the channel for the IIoT sensor node. As illustrated in Figure 4.5.10, some channels are loaded more than others depending on the

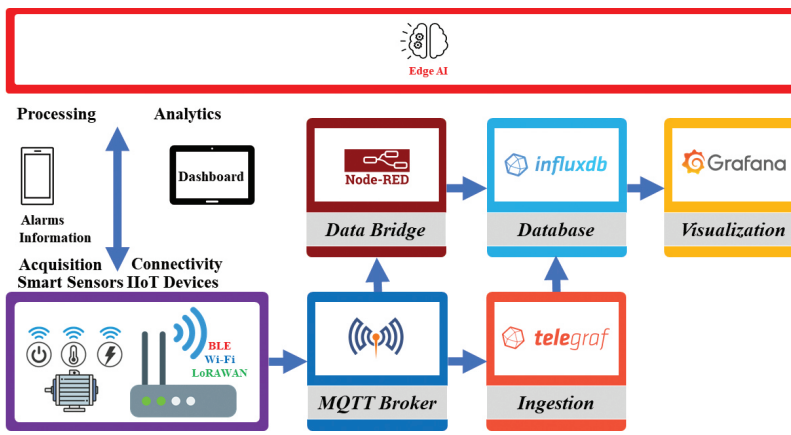


Figure 4.5.9 Experimental set-up detailed.

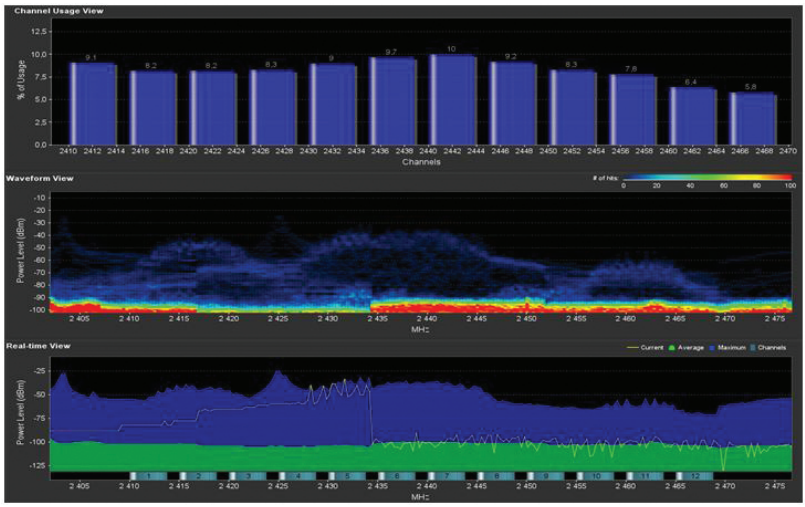


Figure 4.5.10 Spectrum analysis Wi-Fi 2.4GHz.

Wi-Fi devices operating in the area where the sensors are installed. For the setup the channel with the lower load was selected (channel 12).

The IIoT devices are set up to use MQTT and have five channels (three for the accelerometer and two for inclinometer) that are active and enabled for static or streaming mode. During the former, measurements are published to separate topics, while during the latter all measurements are published to a single topic.

The data is parsed on the Node-RED. The Node-RED flow connects to the MQTT broker and subscribes to each sensor's measurement topic, waits for the payloads, checks the acquisition mode, parses the payload, extracts measurements, displays the data on live dashboards, stores data into the database and displays in Grafana. A Node-RED flow is visualised in Figure 4.5.11 and a Node-RED dashboard in Figure 4.5.12.

The Payload Parsing node function receives as input the payload sent from the IIoT device and captured by the MQTT broker nodes. The payload is sliced into sections, each section will hold only the payload of each field of the MQTT frame content and return them in array object.

The Measurement Threshold Alarm function listens to measurement values and send notification if a defined threshold is reached. SMTP or SMS to mobile are used for notifications. Other functions nodes are preparing the data for the database and for the Node-RED live dashboard.

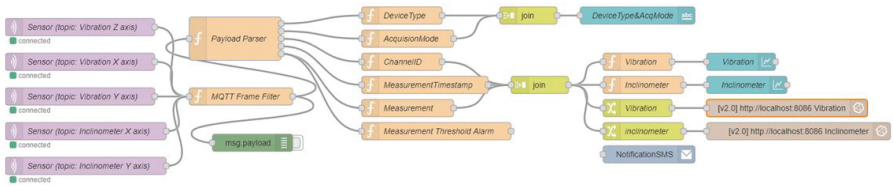


Figure 4.5.11 MQTT subscriber with Node-RED flow (vibration sensor per topic).

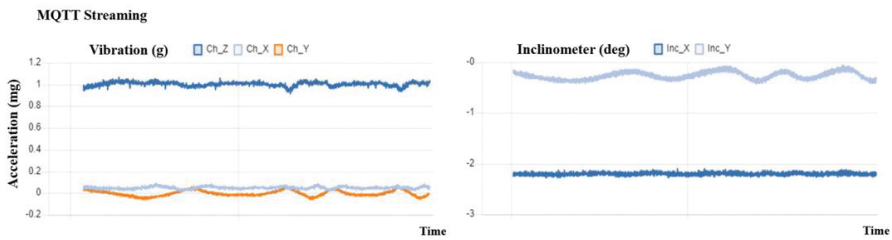


Figure 4.5.12 MQTT subscriber with Node-RED results (vibration data streaming).

During the preliminary vibration analysis, techniques, such as K-means clustering to organise the measurements into useful clusters and non-linear regression to make predictions inside and outside of the training sets area, have been employed. The sensor data were also processed visually as 3D points and inputted to a K-means clustering and non-linear regression together with data to test against the learning data.

This configuration is implemented on meta-edge server, where MQTT, Node-RED, InfluxDB and Grafana are setup to work together. This workflow can also be implemented on deep edge. Furthermore, Node-RED and possibly also MQTT can be replaced with Telegraf [14] in case of simpler flows with less automation logic, thus reducing the software stack.

In addition to the above flow, a second flow has been deployed using the STWIN SensorTile Wireless Industrial Node [17], which is a complete sensor-to-edge and cloud ecosystem with environmental sensing, vibration monitoring and sound/ultrasound detection. It also features a debugger, embedded signal processing libraries running on an ARM Cortex-M4 microcontroller and an ultra–low-power accelerometer to preserve battery life during monitoring. As shown in Figure 4.5.13, it has digital and analogue microphones, inertial sensors and temperature and pressure sensors connected through wired or wireless options.

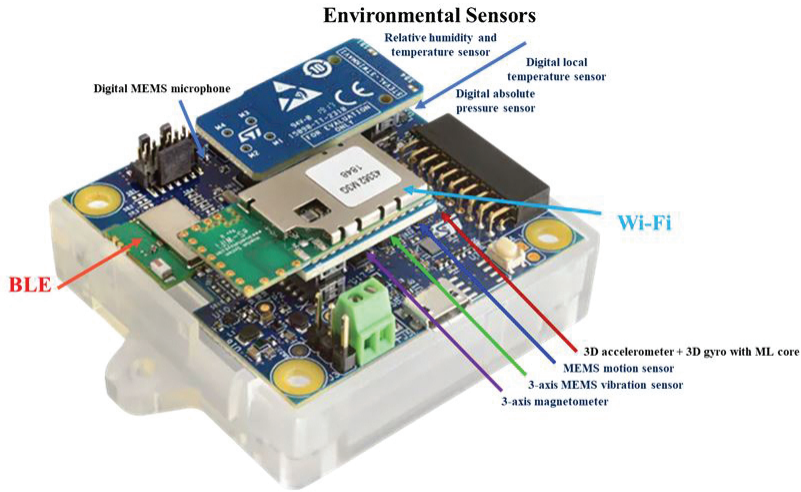


Figure 4.5.13 STWIN SensorTile wireless industrial node.

Data from all the sensors are sent to a USB at the maximum data rate for analysis by the computer; it can also be stored on an SD card or transmitted using BLE or Wi-Fi capabilities.

The device can be connected directly to the cloud through a secure connection using certificates. On the cloud side, it is also possible to collect data from various devices that are located at disparate places and/or from devices using different connectivity. AI methods can be employed to collect data at the edge and in the cloud.

The test setup includes several IIoT devices that take various measurements using BLE, Wi-Fi or LoRaWAN communication protocols and an AC low power e-motor installed on a lab test bench.

The data collection flow was also tested by IIoT devices connected directly to the cloud through the Wi-Fi expansion to verify the STWIN device's capabilities and fitness to the purpose. The end-to-end solution includes access to a dashboard for the predictive maintenance application that allows data from the IIoT devices to be collected and visualised. The sensor parameters have thresholds that trigger alarms and warnings.

Another IIoT device was connected via Bluetooth to an Android tablet with the BLE app installed. The app recognised the board after the board with the preloaded software was powered up. The data collected from the sensors are illustrated in Figure 4.5.14.

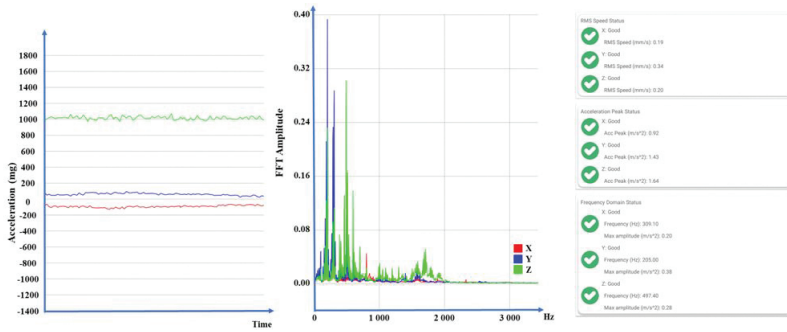


Figure 4.5.14 Real-time data from the three-axis MEMS vibration sensor.

The various graphs show the live data coming from the three-axis MEMS vibration sensor and the FFT applied to the signal received from the vibration sensor that indicates the main frequencies in the spectrum.

As illustrated in Figure 4.5.14, thresholds have been added to the sensors, and unbalanced situations were purposely created to trigger the alarms and warnings associated with the vibration parameters. The data logs were downloaded for further analysis and processing.

4.5.8 Summary and Future Work

This article presented an AI- and IIoT-based PdM for soybean processing and its implementation approaches. The PdM foundations described in the article are followed to develop a PdM concept that fits the process requirements of soybean manufacturing. The maintenance scope assumes that every maintenance action conducted at the equipment/motors restores functionality and durability to their original level. The PdM solution for the soybean production system targets individual equipment. Future activities could address the possibility of grouping maintenance actions that may lead to an overall cost reduction for maintenance activities.

The proposed edge computing solution improves the performance, security, operating cost and reliability of IIoT and AI-based platform, applications and services.

The system design is based on a heterogeneous wireless sensor network consisting of sensor nodes and IIoT devices with different communication interfaces (e.g., BLE, LoRaWAN, Wi-Fi), computing power, sensing range and AI-based processing capabilities. The network architecture allows for

interfacing with the existing SCADA system and providing a secure link to external cloud applications.

Future work will focus on data fusion and filtering to integrate multiple sensor data and generate data that are more reliable than individual sensor data. The proposed convolutional neural network DL technique to extract data representation will be further evaluated to integrate the AI-based model and the algorithms developed. The vibration signals collected will be converted to a discrete frequency spectrum via Fast Fourier Transform and further analysed to improve the PdM model.

The AI- and IIoT-based PdM concept will be further developed for edge processing at different levels by combining micro, deep and meta-edge with local data access and storage.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] EN 13306:2017, Maintenance Terminology. European Standard. CEN (European Committee for Standardization), Brussels, 2017.

[2] T.P. Carvalho, F.A.A.M.N., Soares, R. Vita, R. da P. Francisco, J.P. Basto, and S.G. S. Alcalá, (2019). A systematic literature review of machine learning methods applied to predictive maintenance, Computers & Industrial Engineering, Volume 137. Available online at: https://doi.org/10.1016/j.cie.2019.106024

[3] Netto A.C., A.H. de Andrade Melani C.A. Murad, M.A. de Carvalho Michalski, G.F. Martha de Souza, S.I. Nabeta (2020) A Novel Approach to Defining Maintenance Significant Items: A Hydro Generator Case Study. Energies. 2020; 13(23):6273. Available online at: https://doi.org/10.3390/en13236273

[4] ISO/IEC 25010:2011. Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models.

[5] W. Silva, M. Capretz, (2019). "Assets Predictive Maintenance Using Convolutional Neural Networks," in 20th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)

[6] Z. Wang, T. Oates (2015). "Imaging time-series to improve classification and imputation," 2015, arXiv:1506.00327. Available: http://arxiv.org/abs/1506.00327

[7] K. S. Kiangala and Z. Wang, "An Effective Predictive Maintenance Framework for Conveyor Motors Using Dual Time-Series Imaging and Convolutional Neural Network in an Industry 4.0 Environment," in IEEE Access, Vol. 8, pp. 121033-121049, 2020, Available online at: https:// doi: 10.1109/ACCESS.2020.3006788.

[8] R. K. Mobley, An introduction to predictive maintenance. Elsevier, 2002

[9] Ran, Y., Zhou, X., Lin, P., Wen, Y. and Deng, R. (2019). "A Survey of Predictive Maintenance: Systems, Purposes and Approaches". IEEE Communications Surveys & Tutorials, 20, pp. 1-36. Available online at: https://arxiv.org/pdf/1912.07383.pdf

[10] L. Jing, M. Zhao, P. Li, and X. Xu, "A convolutional neural network based feature learning and fault diagnosis method for the condition monitoring of gearbox," Measurement, Vol. 111, pp. 1–10, 2017.

[11] InfluxDB Platform. Available online at: https://www.influxdata.com/products/influxdb-overview/

[12] Eclipse Mosquitto. Available online at: https://mosquitto.org/

[13] Grafana. Available online at: https://grafana.com/grafana/

[14] Telegraf. Available online at: https://www.influxdata.com/time-series-platform/telegraf/

[15] Node-RED. Available online at: https://nodered.org/

[16] MQTT. Message Queuing Telemetry Transport. Available online at: https://mqtt.org/

[17] STWIN SensorTile Wireless Industrial Node. Available online at: https://www.st.com/en/evaluation-tools/steval-stwinkt1.html

Section 5

AI Transportation

5.0

Applications of AI in Transportation Industry

Mathias Schneider[1], **Matti Kutila**[2] **and Alfred Höß**[1]

[1]Ostbayerische Technische Hochschule Amberg-Weiden, Germany
[2]VTT Technical Research Centre of Finland Ltd., Finland

Abstract

This introductory article opens the section on "Applications of AI in Transportation Industry", giving a broad overview of the latest AI technologies in the transportation industry, with an additional focus on the developments enabling automated Mobility-as-a-Service (MaaS). It presents future capabilities and opportunities for AI, together with covering state-of-the-art Intelligent Transport Systems (ITS) trends, including advancements on the vehicle, infrastructure, and management level. Finally, the article outlines the two papers included in this section, highlighting concepts and challenges of using AI for automated, optimised, and individual passenger transport.

Keywords: intelligent transport systems (ITS), mobility-as-a-service (MaaS), advanced driver assistance systems (ADAS).

5.0.1 Introduction and Background

Transportation industry is a crucial element to guarantee our daily lives. Following the previous trends of the last decade, the transportation industry has pioneered by digitising its processes by introducing extensive data

systems and automated agents, spanning from the vehicle up to traffic systems.

To understand and control this data, it became mandatory to optimise processes on the micro- and macroscopic level in this complex, ever-changing ecosystem. However, since data alone does not enable higher efficiency, safety or automation, the demand for data processing is constantly increasing. Thereby, specific use cases, e.g., in the field of automated driving, require high demands in terms of latency. Decentralised, intelligent systems leveraging efficient AI models and suitable edge computation platforms are currently being investigated to close the gap. These developments will contribute to the European Commission long-term strategies "*Vision Zero*" (reduce road fatalities to almost zero) and "*European Green Deal*" (climate-neutrality), which should be reached by 2050.

In this introduction article, we will introduce the state-of-the-art for automated passenger transport. Thereby, we will elaborate on recent trends on AI-enabled automated MaaS in the field of ITS and envision possible opportunities. Finally, the article outlines ongoing activities concerning the *AI4DI* project that are presented in two separate articles.

5.0.2 AI Developments in Transportation Industry

In recent years, AI progressively became an imperative approach for processing ITS related data. This trend, reinforced by wide industrial support, establishes a solid foundation to build an efficient MaaS architecture. Accordingly, the latest progress for Machine Learning (ML) applications is discussed based on the survey by Yuan et al. [1]. The authors of this paper structure ML applications in three primary tasks: perception, prediction, and management. This differentiation corresponds to the processing architecture for automated driving, namely perception, planning, and control, which by itself is an expansive research field [2] [3].

Perception – Nowadays, due to the broad usage of different sensors such as cameras, LiDARs, and radars, traffic perception data's variety and quantity increased exponentially. Accordingly, ML approaches are progressively leveraged as a first step to process this data to retrieve valuable information. Perception aspects deal with the physical world (road, vehicles, and pedestrians) and the monitoring of the digital components (reliability and security of the communication network).

Whereas earlier work for object classification, detection, and segmentation leverages mainly supervised ML algorithms such as Support Vector Machines (SVM) utilising hand-crafted features, recent trends aim to harness deep-learning (DL) models, capable of embedding features in their neural network architecture. Common approaches include Convolutional Neural Network (CNN), and implementations such as YoloV4 [4]. In contrast to traditional algorithms, these models tend to be more versatile (resolution, orientation, scene) and robust against anomalies or external conditions (daylight or weather). Besides perception algorithms relying on a single sensor-type input, data-fusion approaches are currently under development. These operate either low-level (a single model uses all raw sensor inputs for inference) or high-level (multiple networks are used, and outputs are concatenated at a later stage) [5] and further improve the overall reliability of the perception module. Moreover, perception algorithms fusing the output of multiple agents generating HD-maps and digital twins [6] are research fields.

Prediction – Diverse ML approaches are investigated for ITS to fulfil prediction purposes, including anticipating traffic, travel times, vehicle behaviour, and road occupancy. These methods improve the decision-making fleet management, e.g., regarding the last mile support use case. Traffic flow prediction methods are applied based on the results of the presented perception models and are used to determine travel times for vehicles and passengers. Subsequently, the results are leveraged to eventually optimise the vehicle and route selection on a global scale. Since these tasks require the model of temporal-spatial changes, Recurrent Neural Network (RNN) architectures and derivates, such as Long Short-Term Memory (LSTM) [7], are employed.

Management – ML for management tasks is considered to raise efficiency on vehicle-, infrastructure-, and resource-level. This includes control of traffic lights and a trajectory or route selection for the automated fleet. Secondary tasks, such as networking and computation problems, are tackled, comprising resource management for V2X communication [8] and mobility-aware edge computing offloading [9].

In contrast to the previous domain, ML often investigates deep reinforcement learning (DRL) techniques for management decisions. For instance, Deep Q-Learning (DQN) is considered to optimise traffic light management to minimise queue waiting times [10]. Besides, Proximal

Policy Optimization (PPO) is leveraged for steering and speed control of an automated vehicle [11].

5.0.3 Future Trends for Applications in Transportation Industry

The following paragraphs elaborate on two future applications utilising the introduced ML technologies in detail.

Automated driving – In recent years, AI has been used commercially in passenger cars' Advanced Driver Assistance Systems (ADAS). In addition, lately, AI has also been used in the development of automated driving functionalities. CNN and DRL are the most common deep learning methodologies, which have been successfully applied to automated driving solutions. Developing a reliable and robust fully Automated Driving System (ADS) often needs that several AI methods are used together.

Training data is one of the essential requirements and challenges to develop deep learning solutions. Many ADS developers have done the collection of large data sets for autonomous driving and environment perception. Luckily, more and more open data sets have been published for the research community. One of the best-recognized data sets for ADS development is the KITTI benchmark suite [12], which includes several data sets to evaluate various ADS functions [3]. There are also other similar open data sets such as Waymo Open data set [13], Cityscapes [14], Berkeley DeepDrive [15], etc. The training data is always limited as it is impossible to cover all scenarios that an automated vehicle could encounter in the real world. However, the rapid progress in collecting larger and larger data sets will enable more advanced deep learning systems on automated vehicles.

The environment perception and scene understanding around the vehicle is crucial for automated driving. This includes detection of other road users, road markings and other road furniture. Deep neural networks, such as CNNs, are today very accurate for detecting, tracking, and classifying various road user types, including cars, trucks, busses, pedestrians, cyclists, etc. A breakthrough has been achieved in pedestrian detection solutions with deep learning [1]. However, there are still some challenges in the pedestrian detection task from camera data, such as substantial occlusions and bad weather conditions. Deep learning-based methods are also widely used for detecting and tracking positions and geometries of moving obstacles (e.g., other vehicles) based on camera data [16]. Image segmentation is used to

classify the pixels of an image into the road and non-road parts [1]. Road marking detection and recognition involves detecting the marking positions and recognizing their types (e.g., lane markings, road markings, messages, and crosswalks) [16]. Other road furniture detection includes, for example, traffic sign recognition.

AI-based environment perception algorithms utilize only two dimensions (2D). However, 2D models are not enough in all cases to describe 3D real-world objects. The 3D perception is based on LiDAR or stereo cameras. 3D tracking and behaviour prediction of other road users is required in automated driving. Vehicle behaviour corresponds to braking, steering, lane change and moving trajectory [1]. Pedestrian behaviour includes actions like running or crossing the street [1]. In future years, AI and ML will gradually enable better prediction of the behaviour and intent of other road users.

Traffic flow and public transport travel time prediction – Various combinations of AI algorithms have been used in predicting traffic flow and travel time. Travel time predictions enable, for example for vehicle routing, guide vehicle dispatching, as well as congestion and traffic management. Forecasting traffic flows and travel time is a complex and challenging problem, which is affected by diverse factors, including spatial correlations, temporal dependencies, and external conditions (e.g., events, holidays, weather, and traffic lights) [1]. For travel time prediction, there are segment and path-based estimation approaches. Lately, integrated DL methods, which utilize both segment-based and path-based approaches, have also been studied. Recently researchers have also combined deep learning with traditional methods with some success [1].

One problem with AI-based prediction development is that training data is not readily available as most road networks are not equipped with traffic measurement sensors. Traffic data can be collected from mobile devices, and this data is often available from global map data providers such as Google or Here. In many cases, multiple data sources are used together to get better results. High-quality public data sets from the real-world are essential for accurate traffic forecasting. These are progressively available from some cities in Europe as open public data. For example, the open public transport data from a city may provide many opportunities to develop new AI-based tools. Today, most public transport vehicles are fitted with positioning systems (e.g., Global Navigation Satellite System - GNSS), which provides accurate real-time information about the current location and movements of the vehicles. Typically, open public transport data from a city includes vehicle

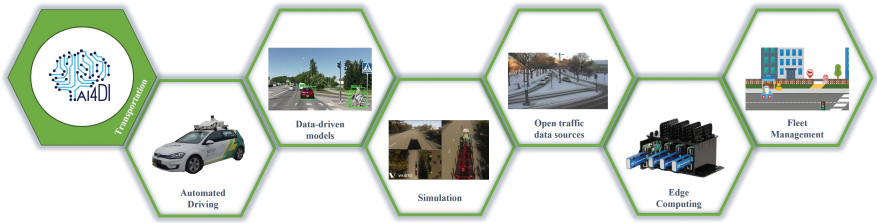


Figure 5.0.1 Transportation research areas in AI4DI.

positions, public transport schedules and route identifiers, etc. This kind of continuous open data stream has enabled the development of Estimated Times of Arrival (ETA) prediction methods utilising ML. Recently, in many studies, several external data sources such as weather, traffic and information about the passengers have been combined for machine learning model development [17].

5.0.4 AI-Based Applications

AI4DI partners are developing AI and Industrial Internet of Things (IIoT) technologies with applications in different areas of the transportation industry sector. This section introduces two articles covering how AI and IIoT are used in the transportation sector. They present challenges and technological developments for perception, prediction, and management in the context of automated MaaS.

The article "*AI-Based Vehicle Systems for Mobility-as-a-Service Application*" describes the safe operation of automated vehicles in urban environments, attempting to improve the environmental perception to detect other road users by proposing a novel method for data fusion between an in-vehicle camera and a LiDAR sensor. Accurate 3D object detection and tracking is achieved by employing deep models (high-level, deterministic, supervised, and reinforcement learning). The KITTI benchmark suite has been used for development and validation, with promising results. The gap between simulated and real environments continuously diminishes with the rapid advances in autonomous control technology that offer improved visual and physical experiences.

The article "*Open Traffic Data for Mobility-as-a-Service Applications - Architecture and Challenges*" addresses the need for high-quality public data sets from the real world with advancing digitisation in the domain of ITS and hence the need for data pre-processing from multiple sources, including raw

sensor data, to prepare for AI-based modelling. While current pre-processing is often implemented as a cloud solution, a system architecture is proposed where computations are scaled and distributed to different layers in the edge–cloud continuum. A set of data refinement strategies has been developed to improve data quality and integrity, which refine the data into becoming more suitable for AI-based MaaS applications.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] T. Yuan, W. Borba da Rocha Neto, C. Rothenberg, K. Obraczka and C. Barakat, "Harnessing Machine Learning for Next-Generation Intelligent Transportation Systems: A Survey," hal-02284820, 2019.

[2] S. Grigorescu, B. Trasnea, T. Cocias and G. Macesanu, "A Survey of Deep Learning Techniques for Autonomous Driving," *Journal of Field Robotics*, no. Machine Learning (cs.LG); Robotics (cs.RO), 2019.

[3] S. Kuutti, R. Bowden, Y. Jin, P. Barber and S. Fallah, "A Survey of Deep Learning Applications to Autonomous Vehicle Control," arXiv:1912.10773, 2019.

[4] A. Bochkovskiy, C.-Y. Wang and M. H.-Y. Liao, "Yolov4: Optimal speed and accuracy of object detection," arXiv preprint arXiv:2004.10934, 2020.

[5] M. Aeberhard and N. Kaempchen, "High-level sensor data fusion architecture for vehicle surround environment perception," in *8th International Workshop Intelligent Transport*, 2011.

[6] O. El Marai, T. Taleb and J. Song, "Roads Infrastructure Digital Twin: A Step Toward Smarter Cities Realization," *IEEE Network*, vol. 35, no. 2, pp. 136–143, 2020.

[7] C. Siripanpornchana, S. Panichpapiboon and P. Chaovalit, "Travel-time prediction with deep learning," in *2016 IEEE Region 10 Conference (TENCON)*, Singapore, 2016.

[8] J. Schmid, M. Schneider, A. Höß and B. Schuller, "A Deep Learning Approach for Location Independent Throughput Prediction," in *IEEE International Conference on Connected Vehicles and Expo (ICCVE)*, Graz, 2019.

[9] Q. Qi, J. Wang, Z. Ma, H. Sun, Y. Cao, L. Zhang and J. Liao, "Knowledge-Driven Service Offloading Decision for Vehicular Edge Computing: A Deep Reinforcement Learning Approach," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 4192–4203, 2019.

[10] X. Liang, X. Du, G. Wang and Z. Han, "A Deep Reinforcement Learning Network for Traffic Light Cycle Control," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1243–1253, 2019.

[11] A. Folkers, M. Rick and C. Büskens, "Controlling an Autonomous Vehicle with Deep Reinforcement Learning," in *IEEE Intelligent Vehicles Symposium (IV)*, Paris, 2019.

[12] A. Geiger, P. Lenz, C. Stiller and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[13] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai and B. Caine, "Scalability in perception for autonomous driving: Waymo open dataset," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Seattle, 2020.

[14] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *IEEE conference on computer vision and pattern recognition*, 2016.

[15] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan and T. Darrell, "BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, 2020.

[16] C. Badue, R. Guidolini, R. V. Carneiro, P. Azevedo, V. B. Cardoso, A. Forechi, L. Jesus, R. Berriel, T. M. Paixão, F. Mutz, L. d. P. Veronese, T. Oliveira-Santos and A. F. De Souza, "Self-driving cars: A survey," *Expert Systems with Applications*, vol. 165, no. 1, 2021.

[17] T. Reich, M. Budka, D. Robbins and D. Hulbert, Survey of ETA prediction methods in public transport networks, arXiv preprint arXiv:1904.05037, 2019.

5.1

AI-Based Vehicle Systems for Mobility-as-a-Service Application

Mikko Tarkiainen[1], Matti Kutila[1], Topi Miekkala[1], Sami Koskinen[1], Jokke Ruokolainen[2], Sami Dahlman[2] and Jani Toiminen[2]

[1]VTT Technical Research Centre of Finland Ltd., Finland
[2]Vaisto Solutions Ltd, Finland

Abstract

Achieving sufficient safety measures is among the major challenges in developing automated vehicles that can operate safely in an urban environment. Data fusion between an in-vehicle camera and a LiDAR sensor can be used for detection and tracking of other road users in an automated vehicle. In addition, simulated environments together with high-level deterministic, supervised and reinforcement learning-based autonomous control could provide traffic safety benefits in the future. These AI-based technologies have been studied in the AI4DI project to enable the Mobility as a Service (MaaS) operators fleet management of automated vehicles. The development and testing of these methods are presented in this chapter with the first promising results. The Camera - LiDAR fusion algorithm provided very good results with the accuracy evaluation using the KITTI dataset. The real-time applicability of the fusion algorithm was also successfully verified.

Keywords: automated driving, sensor data fusion, 3D object detection and tracking, reinforcement learning, simulation, CNN, training, real-time systems, neural networks.

5.1.1 Introduction and Background

This article focuses AI solutions to be used in Mobility as a Service (MaaS) with fleet management of automated driving "last mile" vehicles. Automated vehicles could bring improved efficiency to the MaaS, but road safety is a must. There are several factors affecting the decision making of the automated vehicle. The automated vehicles must be aware of the surrounding road users and obstacles and possess quick reaction times in case unexpected movements or behaviour should occur. This presents the problem of 3D object detection and tracking, which is a major topic of research with automated or autonomous vehicles. Having knowledge of the accurate physical locations of other road users is integral to decision making. In addition, estimation of the speeds and headings of other road users is required to predict possible dangerous situations. Therefore, accurate 3D detection and tracking are needed.

Coming up with the best possible predictions and consequent actions is mission critical requirement for the automated vehicle. A mission critical system is a system that is essential to the survival. The selected action depends on many factors in complex traffic situations. The faster the vehicles are moving, the quicker the cycle of prediction and action selection must be. This creates a critical role for system components of the automated vehicle system including the software components.

This paper presents a novel method for data fusion between an in-vehicle camera and a LiDAR sensor, which enables the vehicle to map 2D image coordinates to a 3D environment and vice versa. This is utilised for detection and tracking of other road users in an automated vehicle. In addition, this paper compares the deterministic, supervised and reinforcement learning-based autonomous control development possibilities on a high level. An autonomous control solution blueprint for a control pipeline that can be trained in a simulation environment is presented. This is done by combining reinforcement learning control planning capability with complementary supervised, learning-based observation metadata detection collection and deterministic safety measures for avoiding collisions and casualties.

5.1.2 AI-Based 3D Object Detection and Tracking for Automated Driving

With the increased computing power, there are more possibilities of implementing AI-based solutions for automated driving systems, which

require real-time processing rates. A convolutional neural network (CNN) is one popular solution for 2D object detection. While the CNNs for image processing are getting more and more accurate, some additional methods are required to make use of these networks in accurate 3D object tracking.

5.1.2.1 Camera and LiDAR Sensor Data Fusion

Data fusion between a camera and a LiDAR sensor enables the vehicle to map 2D image coordinates to a 3D environment and vice versa. With accurate inter-sensor calibrations, the 2D detections provided from image data by a CNN can be transformed into the 3D coordinate system of the LiDAR. This enables 3D location estimation of objects, while taking advantage of the accuracy of a 2D CNN. The 3D points provided by a LiDAR sensor can be projected onto a corresponding 2D image, if the Lidar and the camera providing the images are calibrated to each other.

The method of combining the point cloud and 2D detection boxes to obtain object clusters is described in Figure 5.1.1. The output of a 2D image object detector defines the locations of the objects in the 2D space using rectangular bounding boxes. The point cloud provided by the LiDAR sensor can be projected to the same 2D image, and the pixel coordinates of the projected points can be compared to the detection boxes. The point projections that are inside a detection box boundary can be tagged so that the original 3D point is marked as residing inside a detection box in the 2D perspective. This allows the examination of 3D spatial information of the 2D detection box. A challenge in this method, however, is raised by the fact that the 2D detection boxes are usually drawn so that the entire object is contained inside it, which results in the background area being included especially in the corner areas of the detection box. This means that many of the 3D points that are projected and considered to be inside a detection box, actually originate from the background terrain, and falsify the 3D spatial information related to the actual object. This can be solved by altering the 2D detection box size to make it smaller, and focus it on a certain area of the original detection box. This detection box focusing can eliminate the false point projections originating from the background. This process results in accurate 3D spatial information of the 2D detected objects. With this information, the original 3D point cloud from the LiDAR can be processed to obtain the 3D point cluster representation of the 2D object mapped onto the image by an image object detector.

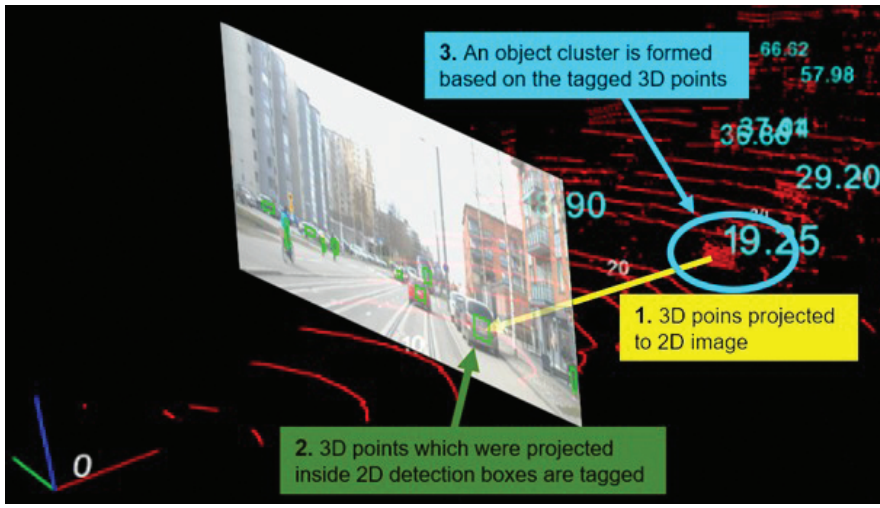


Figure 5.1.1 3D object clustering using a point cloud and 2D detection boxes.

5.1.2.2 Experiments and Results

The presented method of 3D object detection was implemented as a functional real-time system into one of the test vehicles of the VTT automated vehicles research team. The vehicle was equipped with a 32-beam RoboSENSE LiDAR and a 16-beam Velodyne LiDAR for point cloud capturing. The image capturing was performed on a Basler Ace2-series RGB camera, and the images were processed on an Nvidia Jetson Xavier AGX embedded deep learning device. On an actual vehicle integration, the time delays between the data capture events of the LiDAR and the camera must be addressed. To synchronise the point clouds to the captured image more precisely, the odometry data of the vehicle was also captured using a combination of an inertial measurement unit (IMU) and a Global Positioning System (GPS) sensor. The velocity and angular turn rate of the vehicle were used together with the time delays between the camera and the LiDAR to rectify the point cloud to better match the 2D image, and therefore keep the point projections more accurate, even with the vehicle in motion.

This sensor setup was integrated into the vehicle as three separate data capture and distribution modules running on separate computers. The LiDAR-capturing computer collected point clouds, transformed them into the vehicles' common coordinate system, and the modified point cloud was published on an OpenDDS (Open Data Distribution Service) network. The

Jetson device processed the images captured by the Basler camera using the YOLO v4 network [1], and published the images and the neural network 2D detections on the in-vehicle OpenDDS network together with the movement data of the vehicle from the odometry module.

All the processed data was received from the OpenDDS network by the fourth computer, performing the sensor data fusion. The algorithm first received the latest 2D camera-based detections, transformed the LiDAR point cloud and the odometry data. The point cloud was filtered using an Approximate Progressive Morphological Filter (APMF). It is a simplified version of the Progressive Morphological Filter [2], which removes the ground points of the cloud in real-time. The ground points are unneeded, and they are even likely to add error to the later calculations.

The algorithm first applied the detection box focussing, and processed each of the point cloud 3D points. For each 3D point, the delay correction was applied using the odometry and capture time delays, and then the projection was performed onto the 2D image. Then it was checked whether the projected point was placed inside a detection box. This point cloud processing operation was multi-threaded with the sensor fusion computers' processor cores to significantly decrease the computing times.

After matching the 3D points to the 2D detection boxes, the algorithm processed each of the detection boxes to find the objects' 3D location from the LiDAR point cloud. This was done by sorting the points tagged to a detection box based on distance, and choosing the median 3D point as the estimated location of the object. Choosing the median point helps remove any possible noise that might still be caused by some background 3D points, and even occluding obstacles of smaller sizes, which may partially cover the detection box in the image. Another option instead of choosing the median point is to average all of the 3D points which have been tagged to the detection box. This, however, leads to more inaccuracies and makes the estimation much more susceptible to noise from occluding obstacles, for example.

With the estimations of the locations of the objects in 3D, they can be extracted from the ground-filtered point cloud. This was done by cropping the approximate point cloud area containing the object. The 3D crop dimensions depend on the predicted class of the object. For a pedestrian, the cropping is much smaller than for a car, for example. This operation was performed on every object detected by YOLO v4, optimally resulting in the true 3D locations and point representations of the objects, see Figure 5.1.2 and Figure 5.1.3.

Figure 5.1.2 3D detections on camera view.

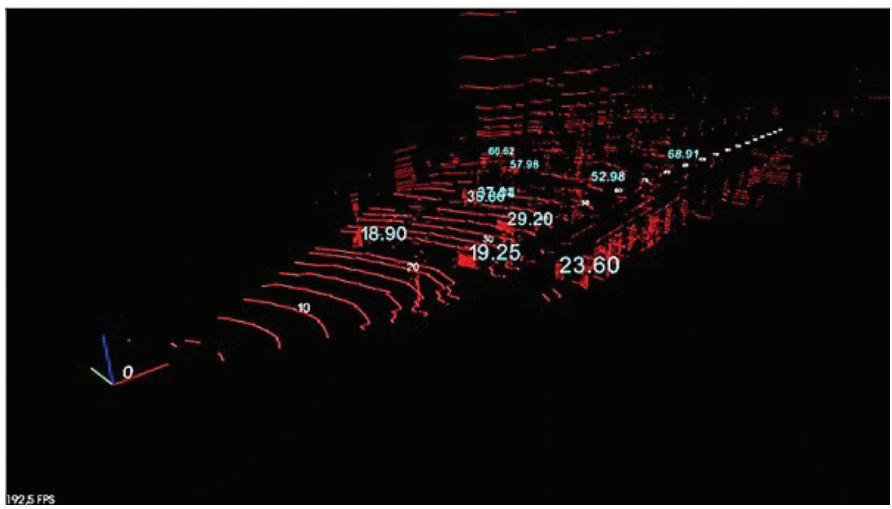


Figure 5.1.3 3D detections in LiDAR point cloud.

5.1.2.3 Evaluation of the Algorithm and Vehicle Integration

The performance of the algorithm was evaluated on the KITTI 3D object detection dataset [3]. Instead of using the standard KITTI evaluation threshold, a custom method of accuracy evaluation was used to focus more on the accuracy of the projection-based location matching. The evaluation was done by estimating the 3D point clusters representing the objects, and

Table 5.1.1 Percentage of estimated object cluster means correctly placed inside KITTI ground truth boxes.

DIFFICULTY	TOTAL	CAR	PEDESTRIAN
Easy	97.91%	99.29%	93.94%
Moderate	92.28%	92.58%	90.72%
Difficult	87.64%	87.50%	88.50%

then comparing whether the mean of the points in a single cluster is found inside a KITTI 3D ground truth detection box. If the average point is inside a ground truth box, it is counted as a success, otherwise it is a failure. Only one match per ground truth box is allowed. The accuracy was evaluated with the 'easy', 'moderate' and 'hard' difficulty thresholds KITTI, with the 'car' and 'pedestrian' classes included, see Table 5.1.1.

The vehicle integration was the main goal of the algorithm implementation. For the practicality of the system, real-time operating speed was critical. The YOLO v4 module was able to operate at a rate of 16 Hz with the Jetson utilising the TensorRT library to accelerate deep learning operations. The amounts of the odometry data were comparably very small, and it was streamed in the OpenDDS network at a rate of 20 Hz. The LiDAR point clouds were captured at 10 Hz, which were the largest data stream in the system. Based on the field of view (FoV) of the Basler camera, the points that were clearly out of the camera image frame were ignored to speed up the algorithm. Additionally, the OpenMP multiprocessing library was utilised to parallelise the data fusion operations, increasing the total speed of the integrated system to real-time levels. The inference times were measured in a 728-second-long test in an urban driving environment, resulting in operation rates of 7-10 Hz for the full system.

5.1.3 Autonomous Control Prototyping in Simulated Environments

Autonomous control fascinates technology enthusiasts and engineering teams all over the world. Public focus is on autonomous road vehicles for bringing improved efficiencies and safety on the road. In more controlled and restricted operating environments, autonomous work machines and robots have already been able to tirelessly perform cycles of work under human operator surveillance for some time. The ambition and need for research remain clear as more advanced autonomous control seems achievable.

5.1.3.1 Reinforcement Learning Control for Mobile Vehicles

The potential to use simulated environments for purposes of training reinforcement learning (RL)-based control agents for mobile machines has been studied in the project by Vaisto. This topic has been studied actively in recent years, see [4] and [5]. Contrary to supervised learning methods, which cover the potential action state space of the targeted operational domain only partially, RL has the theoretical potential to have comparably higher finite action-value state space coverage [6]. Then again, it is a known challenge that applying RL control in the real world is challenging [7].

This higher state coverage brings with it the promise that RL can handle more corner cases, if the RL training process and reward scheme considers the state space coverage as one key performance indicator. There's no certainty that what the action state coverage is and how well an RL agent can adapt to slightly different environments and observations. Research focus remains on measuring the potential actions and state space in each operational domain and then be able to measure that and identify potential pitfalls.

The RL-agent controlled last-mile pod has been trained for project demonstrators in a simulation environment. A short route from the bus stop to the nearby coffee shop has been highlighted in Figure 5.1.4.

The RL agent can drive the pod along any arbitrary and continuous routes in the simulation environment, but in the case wherein the control model was overfitted to be able to collect statistics related to *reward scheme obedience*. The reward scheme monitors the agent's actions, and based on fitting actions, a reward was granted to the agent. If the agent was violating the reward rules, the training episode was ending. The training for this measurement

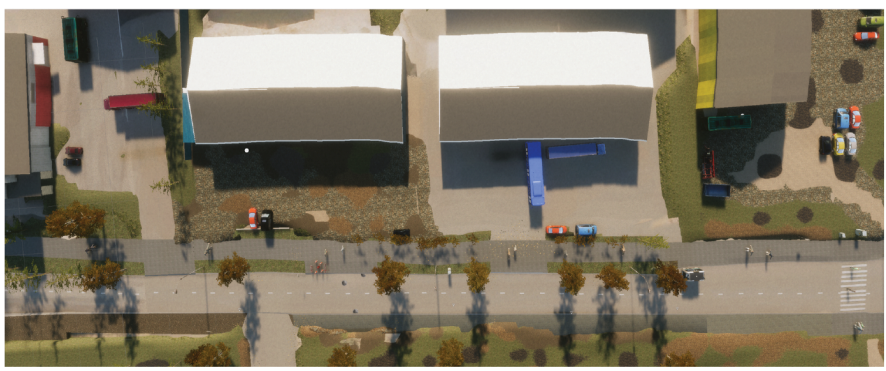


Figure 5.1.4 Last-mile pod driving scenario.

Table 5.1.2 The statistics for the agent performance.

Episode end reasons:		
Maximum (60m) distance reached	16601	68.68%
Episode end checkpoint reached	4135	17.11%
Collision with "dummy" car	2362	9.77%
Pod was off from the GPS line by 1.5m+	617	2.55%
Collision with pedestrian	113	0.47%
The next checkpoint was not reached within 10 sec	338	1.40%
The Pod flipped more than 45 degrees	4	0.02%
TOTAL	24,170	100.00%

Figure 5.1.5 Pod sensor view.

was performed on a laptop workstation with Intel i7 CPU. Agent was trained in simulation over 10 nights (∼90 hours). The statistics for the agent's performance are shown in Table 5.1.2.

The training for this measurement example is not complete, but the statistics clearly show that "Maximum (60m) distance reached" starts to be in the majority and "Episode end checkpoint reached" has increased to 17.11%, so pod is able to complete the route successfully. Based on our experience the reward obedience continues to improve as training continues. Also, the dummy cars and pedestrian are not naturally behaving at all times and they cause some of the episodes to end. As shown in Figure 5.1.5, the sensors are facing forward and thus they currently leave blind spots around the vehicle.

5.1.3.2 The Architecture – Immediate Actions Time-Horizon

Potential solution architecture for autonomous mobile vehicles based on a focus on reinforcement learning was presented above. The main control functions would be handled by a hierarchy of RL agents as shown in Figure 5.1.6. On the top level is a multi-armed bandit agent that would

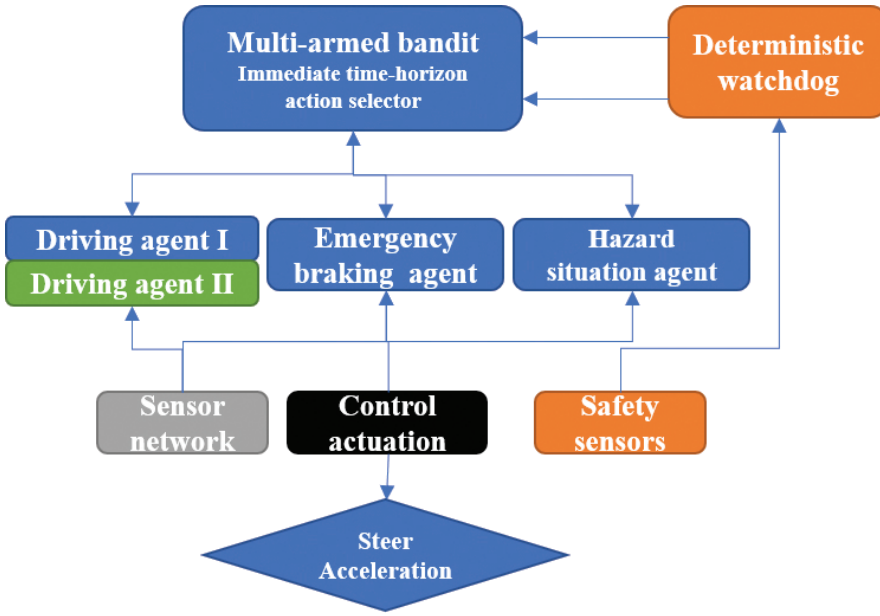


Figure 5.1.6 Immediate action control research architecture.

have the highest system level control authority. Driving agents would be complementing each other and handling specific parts of driving. Then there would be peer agents performing overlapping primary functions and if the actions proposed by the agents would be equal, then the action is approved for actuation.

The multi-armed bandit would be trained to stochastically select the right agent for proposing action in real-time. The object-detecting environmental sensing would be performed by neural networks trained with finite datasets. The time horizons beyond two seconds can rely on various deterministic and machine learning approaches. A reinforcement learning agent can learn to follow, for example, position breadcrumbs, but the actual planning and optimisation is beyond the scope of this article.

5.1.4 Conclusion

This chapter introduced novel methods for data fusion between an in-vehicle camera and a LiDAR sensor for detection and tracking of other road users as well as high-level, deterministic, supervised and reinforcement learning-based autonomous control development possibilities.

The Camera – LiDAR fusion algorithm provided very good results with the accuracy evaluation, being mostly able to locate even the more challenging objects in the KITTI dataset as seen in Table 5.1.1. The real-time applicability of the algorithm was also verified. The developed algorithm makes a valuable contribution to the development of the automated vehicles' environment perception. In addition, the real-time operating speed of the algorithm in the test vehicle was quite fast. However, occasional performance drops also occurred for single frames. In future work, the operating rates could be stabilised by further developing the multiprocessing of the data fusion module.

Reinforcement Learning can be used for developing autonomous driving control in a simulated environment. RL was applied in continuous action space, so the control agents learn to approximate parametric action-value control functions that correspond to real-world needs. A method was presented whereby reinforcement learning is complemented by other machine learning methods or even deterministic safety methods in building a flexible autonomous driving control system. Based on the study, it was concluded that RL potentially plays a role in autonomous control development. Simulation environments are as yet neither visually nor physically on par with the real world, but the gap is getting smaller every year and autonomous control models are already a viable method of product development.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland and Norway.

References

[1] A. Bochkovskiy, C.-Y. Wang, H.-Y. M. Liao, (2020), YOLO v4: Optimal Speed and Accuracy of Object Detection. arXiv 2020, arXiv:2004 10934.

[2] K. Zhang, S.-C. Chen, D. Whitman, M.-L. Shyu, J. Yan, C. Zhang, (2003), A Progressive Morphological Filter for Removing Nonground Measurements From Airborne LiDAR Data. Appl. Opt. 56, 9359-9367 (2017)

[3] A. Geiger, P. Lenz, C. Stiller, R. Urtasun, (2013), Vision meets Robotics: The KITTI Dataset. The International Journal of Robotics Research, 32(11), pp. 1231–1237. doi: 10.1177/0278364913491297.
[4] Praveen Palanisamy (2019): Multi-Agent Connected Autonomous Driving using Deep Reinforcement Learning. arXiv 2019, arXiv: 1911.04175.
[5] Alhawary, Mohammed (2018) Reinforcement-learning-based navigation for autonomous mobile robots in unknown environments. Available from http://essay.utwente.nl/76349/
[6] M. Mitchell Waldrop with Matthew Botvinick: PNAS/What are the limits of deep learning? Proceedings of the National Academy of Sciences Jan 2019, 116 (4) 1074-1077; DOI: 10.1073/pnas.1821594116
[7] Peter Almasi, Robert Moni, Balint Gyires-Toth: Robust Reinforcement Learning-based Autonomous Driving Agent for Simulation and Real World. arXiv 2020, arXiv:2009.11212.

5.2

Open Traffic Data for Mobility-as-a-Service Applications – Architecture and Challenges

Mathias Schneider[1], Mina Marmpena[2], Haris Zafeiris[2], Ruben Prokscha[1], Seifeddine Saadani[1], Nikolaos Evangeliou[2], George Bravos[2] and Alfred Höß[1]

[1]Ostbayerische Technische Hochschule Amberg-Weiden, Germany
[2]Information Technology for Market Leadership, Greece

Abstract

Data-driven approaches will be a pivotal tool to interpret traffic data and to optimise operations to enable more efficient, individual, public transport. Whereas nowadays data remain a proprietary resource, Finland pioneered an open ecosystem. In this work, we present an architecture to acquire heterogeneous data sources and different data refinement strategies at the edge-level, such as a map-matching approach for inaccurate vehicle GPS traces. Finally, data quality monitoring at the cloud-level is highlighted by introducing and applying an *Errors-to-Data Ratio (EDR)* metric.

Keywords: mobility-as-a-service, edge computing, cloud computing.

5.2.1 Introduction and Background

Mobility-as-a-Service (MaaS) is set to revolutionize urban transport by enabling the orchestration of multiple means of transportation [1]. Thereby, Artificial Intelligence (AI) is a key technology capable of transforming vast volumes of historical and real-time data generated by edge devices, such as vehicles, traffic sensors and cameras to valuable knowledge for MaaS [2]. The utilization of traffic data at scale is a critical factor for training predictive

AI systems. They will power a MaaS operator to successfully manage a fleet of automated driving vehicles for real-time, multi-modal and on-demand transportation [3]. Traffic data are collected from heterogeneous sources, and they come in large volumes, diverse formats, and different rates of speed. To unlock the full potential of the traffic data and make them applicable for training AI algorithms suitable for Intelligent Transportation Systems (ITS), we conceptualised and implemented a complete data management stack that entails processing pipelines applied both at the edge and the cloud. Data processing at the edge involves raw data acquisition, pre-processing for feature engineering and the utilisation of an unstructured database for storage. Data management is resumed in the cloud with pipelines that include structuring, further processing, data quality monitoring and storing in a time-series database.

5.2.2 Data Acquisition

Initiated by the strategic Open Tampere program in 2012, the City of Tampere, Finland, is publishing several data sources under the Open Data licence [4]. Traffic-related data are maintained by the ITS Factory Community [5] and InfoTripla [6]. They comprise information of public transport positioning [7], traffic cameras [8] and loop detectors, measuring traffic amount, congestions, and queue lengths [9].

Data scrapers extract, synchronize, and retain data for each of the sources, as illustrated in Figure 5.2.1. Whenever applicable, existing data formats are kept, including the Service Interface for Real Time Information (SIRI) [10] for public transport vehicle activity, as well as DATEX II [11] for traffic amount measurements. Utilising standardised data formats increase the reusability of subsequent processing components. Raw data is stored in an unstructured MongoDB database. Table 5.2.1 presents database statistics, including the amount of data and sampling rates of the different sources. Thereby, bus traces comprise around 3000 traces of about 150 bus lines. As indicated in the table, traffic cameras capture images with different frequencies.

5.2.2.1 Bus Traces

ITS Factory's public transport Application Programming Interface (API) allows to continuously monitor active vehicles with an overall sampling rate between 0.5 Hz and 1 Hz. Utilizing information of the related bus route,

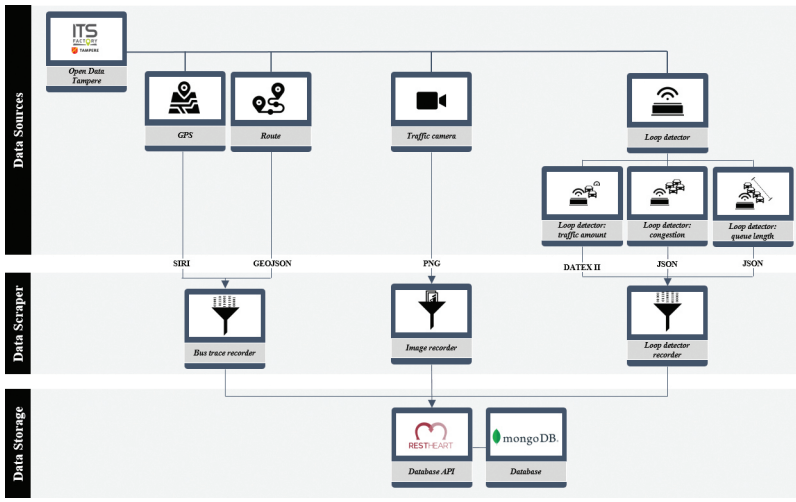


Figure 5.2.1 Open data tampere: design for data acquisition.

Table 5.2.1 Statistics for open traffic data in tampere (2021-5-31). (*) Traffic cameras images are available starting from November 2019 but are not stored in the MongoDB.

	# Samples	Total size [GB]	Avg. size [KB]	Start date	Measurements per day/sensor	# Sensors
Traffic amount	104,616	65.24	653.88	2020-11-18	∼1,440	∼510
Congestion	97,584	12.18	130.88	2020-11-18	∼1,440	∼480
Queue length	59,994	7.24	126.5	2020-11-18	∼720	∼300
Bus traces	597,251	107.21	188.22	2020-11-17	∼3,000	∼150
Traffic camera	7,163,364	657.46	96.24	2021-01-15*	96/192/1,440	∼140

Global Positioning System (GPS) traces are used to generate durations spent from one bus stop to another. They provide valuable information about the traffic flow in general by deriving metrics such as average speed and stop times. Since the GPS accuracy varies especially in urban regions, the trace is subsequently processed to match the true track.

5.2.2.2 Traffic Cameras

About 140 traffic cameras are available around Tampere. Due to privacy reasons, only images are publicly accessible (maximal one per minute). While certain parts of the image are censored (buildings, etc.), the view of the camera focuses on the street and intersections. Image resolutions vary (e.g.,

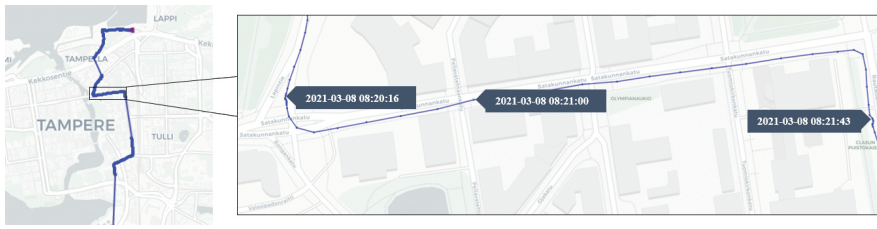


Figure 5.2.2 Bus GPS trace, Line 32 Ranta-Tampella to TAYS Arvo.

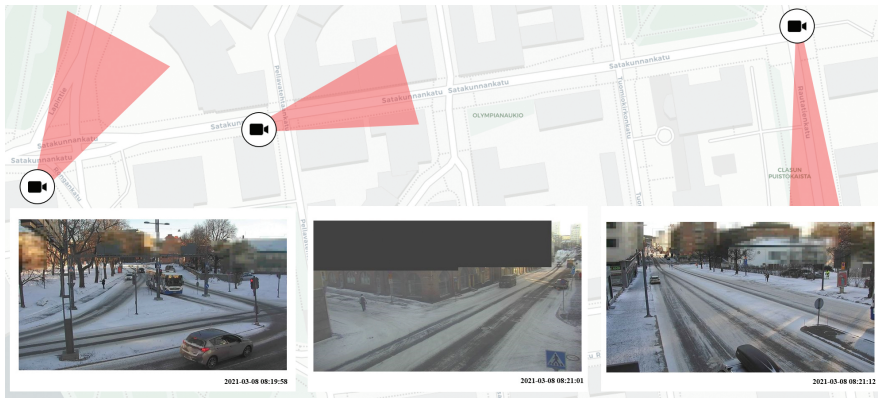


Figure 5.2.3 Traffic cameras and their field of view in Tampere.

640 x 360 px, 704 x 576 px) and objects are largely distorted due to the large perspective. Background objects tend to become very small (less than ten pixels wide) and are often partially occluded. As shown in Figure 5.2.2 and Figure 5.2.3, traces and cameras are roughly synchronized as the passing bus is visible on the images corresponding to its GPS position.

5.2.2.3 Loop Detectors

Tampere provides a vast amount of loop detector measurements, including metrics for traffic amount, congestions, and queue lengths. Data are updated each minute. The spatial information of each sensor is documented separately for each traffic intersection as shown in Figure 5.2.4. Whereas congestions and queue lengths are formatted in JavaScript Object Notation (JSON), traffic amounts are structured using DATEX II standard developed by the European Committee for Standardization (CEN/TC 278).

Figure 5.2.4 Loop detectors for traffic amount measurements using DATEX II.

5.2.3 Data Processing at the Edge

Depending on the data source, raw sensor data are not yet suitable for scaling AI-based MaaS applications. This subsection presents data refinement strategies as illustrated in Figure 5.2.5. The architecture comprises object detection for traffic camera images to condense valuable information related to the traffic flow as well as map-matching algorithms to normalize travel times from bus GPS traces. Whereas this kind of pre-processing is nowadays often implemented as a cloud solution, our architecture leverages heterogeneous edge platforms to orchestrate the required computations. Since the edge platforms cannot be physically deployed to the test field in Tampere, a dedicated hardware-in-the-loop (HIL) laboratory cluster is set up for this task.

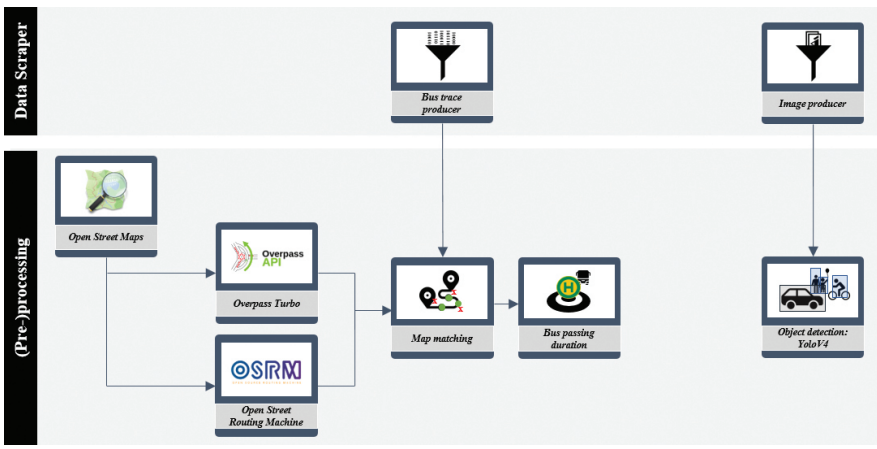


Figure 5.2.5 Architecture for data preparation at the edge.

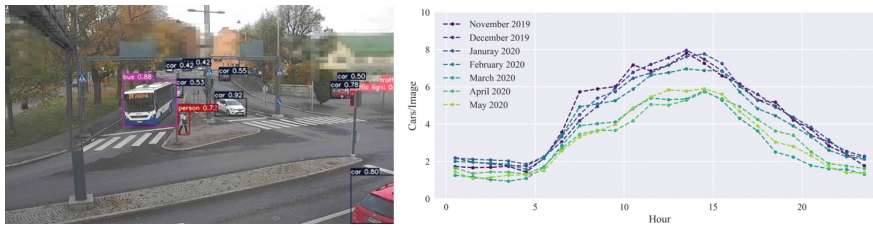


Figure 5.2.6 Traffic object detection (left) and hourly car quantity (right).

5.2.3.1 Object Detection

Object detection is applied to reduce raw, traffic camera image footage to the number of different road objects. Therefore, a YOLOv4 network [12], trained on the MS COCO dataset [13], is leveraged to detect six different types of road users (car, truck, bus, bicycle, motorbike, and person), as well as traffic lights. Although improvements can be introduced to increase the quality of the detection (e.g., excluding parking cars), a first evaluation reveals that it is capable to outline the traffic situation (Figure 5.2.6): whereas the accumulated cars-per-image metric is stable between November 2019 and February 2020, a decline can be observed starting March 2020, likely influenced by the effects of the COVID-19 pandemic.

5.2.3.2 Bus GPS Trace

Bus GPS traces contain a high amount of information about the current traffic state and are utilised to estimate travel times between bus stops and timings for the passenger transfer at a station. Since coordinates are imprecise as shown in Figure 5.2.7, multiple processing steps are conducted to increase the quality of this data source.

Figure 5.2.7 Refinement of GPS bus traces: (a) Raw GPS [blue] and planned bus route [green] (b) Snapped bus route to OSM road network [black] (c) Partitioned route according to bus stop vicinity [yellow/purple] (d) Map-matching GPS trace [red].

In our approach, the given route provided by the bus API is first snapped to the Open Street Map (OSM) road network. Based on the bus stop positions and a predefined radius, the aligned route is split into segments which allow differentiating a segment between two bus stops and a segment in the vicinity of a stop. GPS coordinates are mapped to this aligned route while applying additional consistency checks, e.g., filtering positions too far away from the route, or physically impossible heading deviations introduced by the inaccuracy of raw GPS. This transformation rectifies timings for each segment and further enables to augment additional OSM-based information, e.g., road segment IDs [14] or amenity characteristics [15].

5.2.4 Data Processing in the Cloud

Historical traffic data stored in MongoDB are further processed to extract structural time series features which can be used for machine learning algorithms. Data quality metrics are monitored before and after the final cleaning and imputation to improve the integrity and inherit information value of the training features. The data extraction is performed with Dask, a Python library for parallel computation. The final features are stored in an InfluxDB, a time-series database optimized for fast, high-availability storage and retrieval of time series data. For high-quality visualizations, Grafana, an open-source monitoring and observability platform, is configured to run queries on InfluxDB data.

5.2.4.1 Data Quality Monitoring

In the context of AI-based MaaS applications, data management processes can be influenced by principles that are quite different from those ruling more traditional computing environments. Cloud deployments, streaming data, data volume, volatility and heterogeneity pose new challenges for data-driven analytics. Moreover, the limited explainability of many broadly used AI models adds another layer of ambiguity, since performance issues can be attributed to various factors (e.g., model selection, implementation, data quality). Therefore, data quality assessment and improvement are the first steps in an iterative process of designing, building and evaluating AI solutions. Even after deployment, continuous monitoring of data distributions is critical for detecting data shifts and promptly enact retraining to avoid performance deterioration. To improve data quality and integrity, we defined, quantified, and monitored four classes of errors: 1) duplicate data, 2) missing

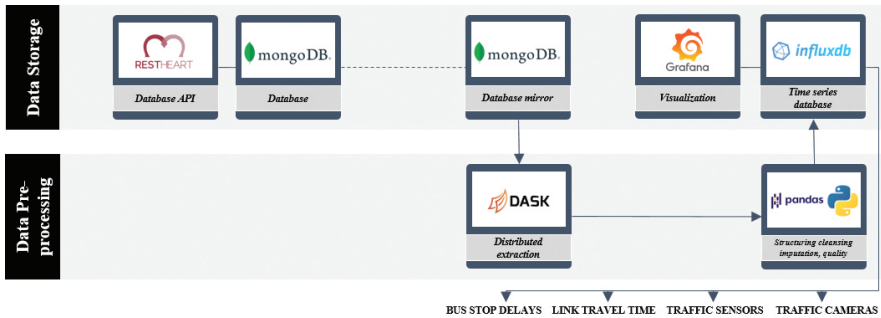


Figure 5.2.8 Data storage and processing in the cloud. The processed features can be retrieved and visualized as time-series and used for training AI prediction models.

data, 3) inconsistent values (e.g., outliers for traffic sensors and cameras, or negative values for travel-time durations, and 4) incomplete items (e.g., bus route segments with less than two GPS traces, or sensor measurements with a count period less than the one defined in the specifications). All these types of errors are considered of critical importance for obtaining a high-quality dataset to train machine learning models [16]. For each error class and each category of traffic data, we calculated the Errors-to-Data Ratio (EDR), i.e., the number of errors divided by the total number of items. To derive an overall data quality metric for each traffic data category, we used the unweighted EDR average across all error classes in the category. EDRs have been calculated before and after removing erroneous measurements. For missing data in the categories of sensors and cameras, the elimination was applied sensor-wise, only for those sensors that exceeded 50% of missing values. The remaining missing values are dealt with imputation by interpolation through time. The threshold was decided to retain a balance between losing information and injecting imputation related bias into the dataset.

5.2.4.2 Data Quality Observations

This section presents some of the preliminary observations obtained from applying the cloud-based data management pipelines on data collected for the week of February 19 to 25, 2021. Data observability is the first step to troubleshoot, understand, and explore the data. Figure 5.2.9 presents the weekly traffic data and bus traces stored as time-series in InfluxDB as they are captured in Grafana dashboards. Expected patterns of seasonality or

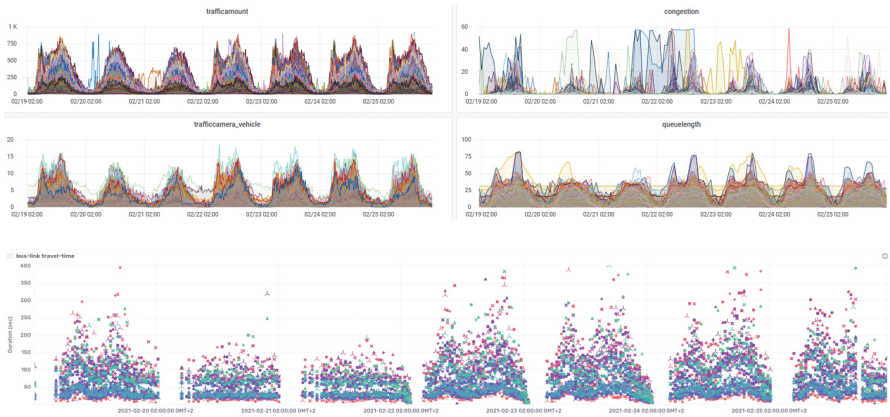


Figure 5.2.9 Weekly data from left to right, top to bottom: traffic amount, congestion, vehicle counts derived from traffic-camera images, queue length and travel-time durations for the segmented bus routes (bus-links).

unexpected outliers can be readily detected to assess the maturity of the data components and decide on further actions.

Subsequently, the EDR metrics were calculated for each category of traffic data and error type, before and after eliminating erroneous samples. Table 5.2.2 presents the ratios and the mean EDR reduction percentage in each category of traffic data. In addition, the number of total measurements is shown before and after the elimination. Our data quality monitoring strategy improves the data by reducing the errors by 26.95% and up to 100%. While the total number of measurements is only reduced by 14.91%, data quality

Table 5.2.2 *Errors-to-Data Ratio* (EDR) for five categories of traffic data collected for the week of February 19 to 25, 2021. EDR is given as a percentage before and after the first step of data cleaning, which involves eliminating erroneous observations.

	EDR (%) pre / post-processing				% EDR	# Measurements
	Duplicates	**Missing**	**Impossible**	**Incomplete**	**Reduction**	**pre/post processing**
Traffic amount	6.93 / 0	15.19 / 12.54	0.9 / 0	0 / 0	45.5	5,836,320 / 5,554,080
Congestion	0 / 0	15.4 / 12.58	2.88 / 0	0 / 0	31.18	5,473,440 / 5,090,400
Queue length	1.82 / 0	51.09 / 39.04	0.51 / 0	0 / 0	26.95	3,376,800 / 1,975,680
Bus traces	0 / 0	13.16 / 0	0.004 / 0	1.63 / 0	100	494,716/ 426,629
Traffic camera	0 / 0	56.64 / 3.97	0 / 0	0 / 0	93	223776 / 60,480
				Measurements Reduction		14.91%

analysis reveals a higher loss of information in the category of 'queue length', in which the lowest EDR reduction is recorded. This observation indicates that this category of data might be of low quality as a feature and needs to be further assessed to decide if it has to be excluded.

5.2.5 Conclusion

With advancing digitisation in the domain of ITS exploiting generated data becomes a key challenge to optimise operations to establish greener and more resource-efficient mobility. In this work, we presented a system architecture to acquire and process open traffic data which will allow AI-based modelling. Our architecture addresses two major challenges for such a system - data volume and quality. To compensate for a high data quantity and related communication overhead, computations are scaled and distributed to different layers in the edge-cloud continuum. Further, the presented monitoring strategies improve the quality of training data sets that are required by data-driven approaches. In future work, we will leverage the data to develop MaaS applications, such as predicting the estimated time of arrival (ETA) for public transport, optimising passenger transfer timing in a last mile use case.

Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland, Norway.

References

[1] G. Smith, J. Sochor and M. Karlsson, "Mobility as a Service: Implications for future mainstream public transport," in International Conference Series on Competition and Ownership in Land Passenger Transport (Thredbo), Stockholm, 2017.

[2] J. Wu, L. Zhou, C. Cai, J. Shen and S. K. Lau, "Data Fusion for MaaS: Opportunities and Challenges," in IEEE 22nd International Conference on Computer Supported Cooperative Work in Design (CSCWD), Nanjing, 2018.

[3] C. O. Cruz and J. M. Sarmento, ""Mobility as a Service" Platforms: A Critical Path towards Increasing the Sustainability of Transportation Systems," Sustainability 2020, vol. 16, no. 6368, 7 August 2020.
[4] City of Tampere, "Avoin data -lisenssi," [Online]. Available: https://www.tampere.fi/tampereen-kaupunki/tietoa-tampereesta/avoin-data/avoin-data-lisenssi.html. [Accessed 24 3 2021].
[5] ITS Factory, "ITS Factory - Innovative Tampere Site," [Online]. Available: https://itsfactory.fi/. [Accessed 24 3 2021].
[6] "infoTripla - Smart Mobility," [Online]. Available: https://infotripla.fi/. [Accessed 24 3 2021].
[7] City of Tampere, [Online]. Available: https://lissu.tampere.fi/timetable/. [Accessed 2021 March 24].
[8] City of Tampere, "Tampere traffic camera API," [Online]. Available: https://traffic-cameras.tampere.fi/.
[9] City of Tampere, "Trafficlightdata Service - API," [Online]. Available: http://trafficlights.tampere.fi/. [Accessed 24 3 2021].
[10] VDV Die Verkehsunternehmen, "CEN TS 15531 Service Interface for Real time Information (SIRI)," [Online]. Available: https://www.vdv.de/siri.aspx. [Accessed 24 3 2021].
[11] DATEX II, "DATEX II version 3 documentation portal," [Online]. Available: https://docs.datex2.eu. [Accessed 24 3 2021].
[12] A. Bochkovskiy, C.-Y. Wang and M. H.-Y. Liao, "Yolov4: Optimal speed and accuracy of object detection," arXiv preprint arXiv:2004.10934, 2020.
[13] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, L. Zitnick and P. Dollár, "Microsoft COCO: Common Objects in Context," arXiv:1405.0312, 21 Feb 2015.
[14] J. Schmid, P. Heß, A. Höß and B. Schuller, "Passive monitoring and geo-based prediction of mobile network vehicle-to-server communication," in 14th International Wireless Communications & Mobile Computing Conference (IWCMC), Limassol, 2018.
[15] J. Schmid, M. Schneider, A. Höß and B. Schuller, "A Comparison of AI-Based Throughput Prediction for Cellular Vehicle-To-Server Communication," in 15th International Wireless Communications & Mobile Computing Conference (IWCMC), Tangier, 2019.
[16] V. N. Gudivada, A. Apon and J. Ding, "Data Quality Considerations for Big Data and Machine Learning: Going Beyond Data Cleaning and Transformations," International Journal on Advances in Software, vol. 10, no. 1 & 2, 2017.

List of Contributors

Al-Baddai, Saad, *Infineon Technologies AG, Regensburg, Germany*

Alin, François, *University of Reims Champagne-Ardenne, France*

Amukhtar, Adil, *Graz University of Technology, Austria*

Andersen, Karl, *NXTECH AS, Norway*

Andrade, Liliana, *Univ. Grenoble Alpes, CNRS, Grenoble INP, TIMA, France*

Arents, Janis, *EDI - Institute of Electronics and Computer Science, Latvia*

Bahr, Roy, *SINTEF AS, Norway*

Baumela, Thomas, *Univ. Grenoble Alpes, CNRS, Grenoble INP, TIMA, France*

Bellmann, Ronnie Otto, *DENOFA AS, Norway*

Bette, Ann-Christin, *Infineon Technologies AG, Munich, Germany*

Bichler, Olivier, *CEA-LIST, France*

Blaha, Petr, *Brno University of Technology CEITEC, Czech Republic*

Bockrath, Steffen, *Fraunhofer Institute for Integrated Systems and Device Technology IISB, Germany*

Bravos, George, *Information Technology for Market Leadership, Greece*

Breiland, John, *NXTECH AS, Norway*

Briand, David, *CEA-LIST, France*

Burmer, Christian, *Infineon Technologies AG, Germany*

Cappelle, Hans, *IMEC, Belgium*

Conti, Francesco, *Alma Mater Studiorum – Università di Bologna, Italy*

Coppola, Marcello, *ST Microelectronics, France*

Dahlman, Sami, *Vaisto Solutions Ltd, Finland*

Daronkolaei, Ali Gorji, *IMEC, Belgium*

De Luca, Cristina, *Infineon Technologies AG, Munich, Germany*

de Oliveira, Rachel Ouvinha, *Champagne Vranken-Pommery, France*

Debaillie, Björn, *IMEC, Belgium*

Doré, Philipe, *CEA-LIST, France*

Dosedel, Martin, *Brno University of Technology CEITEC, Czech Republic*

Evangeliou, Nikolaos, *Information Technology for Market Leadership, Greece*

Felbinger, Hermann, *AVL List GmbH, Austria*

Gaveau, Nathalie, *University of Reims Champagne-Ardenne, France*

Greitans, Modris, *EDI - Institute of Electronics and Computer Science, Latvia*

Höß, Alfred, *Ostbayerische Technische Hochschule Amberg-Weiden, Germany*

Haas, Fabian, *Infineon Technologies AG, Germany*

Haidar, Adnan *Technische Universität Dresden, Germany*

Havranek, Zdenek, *Brno University of Technology CEITEC, Czech Republic*

Hjertaker, Torgeir, *DENOFA AS, Norway*

Indirli, Fabrizio, *Politecnico di Milano, Italy*

Jäkel, Rene, *Technische Universität Dresden, Germany*

John, Reiner, *AVL List, Austria*

Josifovski, Josip, *Technical University of Munich, Germany*

Juhrisch, Martin, *Symate GmbH, Germany*

Kaufmann, David, *Graz University of Technology, Austria*

Kern, Roman, *Know-Center GmbH, Austria, Graz University of Technology, Austria*

Khan, Muhammad Ghufran, *Technische Universität Dresden, Germany*

Klück, Florian, *AVL List GmbH, Austria*

Klarmann, Noah, *Technical University of Munich, Germany*

Knoll, Alois C., *Technical University of Munich, Germany*

Koskinen, Sami, *VTT Technical Research Centre of Finland Ltd., Finland*

Kostka, Pawel, *Technische Universität Dresden, Germany*

Kozovsky, Matus, *Brno University of Technology CEITEC, Czech Republic*

Kristoffersen, Anders, *DENOFA AS, Norway*

Kutila, Matti, *VTT Technical Research Centre of Finland Ltd., Finland*

Latella, Antonio, *SCM Group, Italy*

Lenz, Claus, *Cognition Factory GmbH, Germany*

Lesser, Bernd, *VIF - Virtual Vehicle Research GmbH, Austria*

Lindberg, David, *INTELLECTUAL LABS AS, Norway*

Lippmann, Bernhard, *Infineon Technologies AG, Munich, Germany*

Liu, Lan, *Know-Center GmbH, Austria*

Lorentz, Vincent, *Fraunhofer Institute for Integrated Systems and Device Technology IISB, Germany*

Ludwig, Harald, *Fraunhofer Institute for Integrated Systems and Device Technology IISB, Germany*

Ludwig, Matthias, *Infineon Technologies AG, Munich, Germany*

Martinsen, Jøran Edell, *DENOFA AS, Norway*

Memic, Mirhad, *Infineon Technologies Austria AG, Austria*

Müller-Pfefferkorn, Ralph, *Technische Universität Dresden, Germany*

Malmir, Mohammadhossein, *Technical University of Munich, Germany*

Marmpena, Mina, *Information Technology for Market Leadership, Greece*

Miekkala, Topi, *VTT Technical Research Centre of Finland Ltd., Finland*

Mischitz, Martin, *Infineon Technologies Austria AG, Austria*

Mohimont, Lucas, *University of Reims Champagne-Ardenne, France*

Moser, Josef, *Infineon Technologies Austria AG, Austria*

Nica, Iulia, *Graz University of Technology, Austria*

Noaille, Louis, *TechNext, France*

Ocket, Ilja, *IMEC, Belgium*

Pétrot, Frédéric, *Univ. Grenoble Alpes, CNRS, Grenoble INP, TIMA, France*

Papadoudis, Jan, *Infineon Technologies AG, Germany*

Papariello, Francesco, *STMicroelectronics, Italy*

Pelz, Georg, *Infineon Technologies AG, Munich, Germany*

Pierlot, Clément, *Champagne Vranken-Pommery, France*

Plorin, Daniel, *Audi AG, Germany*

Prokscha, Ruben, *Ostbayerische Technische Hochschule Amberg-Weiden, Germany*

Puglia, Giacomo Michele, *DPControl, Italy*

Purice, Dinu, *Cognition Factory GmbH, Germany*

Rahmanpour, Parsa, *INTELLECTUAL LABS AS, Norway*

Razouk, Houssam, *Infineon Technologies Austria AG, Austria, Graz University of Technology, Austria*

Renner, Anna *Symate GmbH, Germany*

Roesler, Mathias, *University of Reims Champagne-Ardenne, France*

Rojko, Andreja, *Infineon Technologies Austria AG, Austria*

Rondeau, Marine, *University of Reims Champagne-Ardenne, France*

Ruokolainen, Jokke, *Vaisto Solutions Ltd, Finland*

Saadani, Seifeddine, *Ostbayerische Technische Hochschule Amberg-Weiden, Germany*

Safont-Andreu, Anna, *Infineon Technologies Austria AG, Austria*

Salmon, Tullio, *Alma Mater Studiorum – Università di Bologna, Italy*

Sand, Hans Erik, *NXTECH AS, Norway*

Schneider, Mathias, *Ostbayerische Technische Hochschule Amberg-Weiden, Germany*

Schober, Wolfgang, *Infineon Technologies AG, Regensburg, Germany*

Sindaco, Simone, *Alma Mater Studiorum – Università di Bologna, Italy*

Stamatis, George, *Information Technology for Market Leadership, Greece*

Stark, Dominik, *Audi AG, Germany*

Steffenel, Luiz Angelo, *University of Reims Champagne-Ardenne, France*

Tarkiainen, Mikko, *VTT Technical Research Centre of Finland Ltd., Finland*

Tecce, Felice, *DPControl, Italy*

Toiminen, Jani, *Vaisto Solutions Ltd, Finland*

Tsang, Ing Jyh, *University of Antwerp, Belgium*

Urlini, Giulio, *STMicroelectronics, Italy*

Vermesan, Ovidiu, *SINTEF AS, Norway*

Wagner, Matthias, *AUDI AG, Germany*

Waldhör, Stefan, *Fraunhofer Institute for Integrated Systems and Device Technology IISB, Germany*

Winkler, Anja, *Technische Universität Dresden, Germany*

Winkler, Peter, *Technische Universität Dresden, Germany*

Wotawa, Franz, *Graz University of Technology, Austria*

Zafeiris, Haris, *Information Technology for Market Leadership, Greece*

Zanghieri, Marcello, *Alma Mater Studiorum – Università di Bologna, Italy*

Index

3D object detection and tracking 358, 361, 362
60-GHz radar 221

A

abstract model 48, 51, 64
acausal model 84, 87
advanced driver assistance systems (ADAS) 353, 356
AI based diagnosis 48, 50, 64, 65, 84
AI based predictive maintenance 48, 64, 337, 338
AI-based microcontrollers 261
anti-counterfeiting 148
anomaly detection 5, 105, 141, 169, 172, 339
artificial intelligence (AI) 251, 300, 326
artificial intelligence (AI) methods 239
artificial neural networks (ANN) 239, 241
automated driving 354, 356, 341, 362, 374
automotive logistic 3
automotive manufacturing 1, 7, 35
automotive production 1, 12

B

battery management system 21, 24–27

C

champagne production 251, 257, 261, 262, 275, 287
champagne 251, 256, 257, 261, 262, 287
classification 14, 16, 52, 76, 93, 132
cloud computing 262, 347
CNN 120, 134, 153, 168, 172, 197
collision detection 239, 241
computer vision 106, 109, 133, 153, 205, 207
computing-at-the-edge 84, 94
condition monitoring 50, 55, 325, 326, 331
consistency improvement 114, 116, 117, 122
convolutional neural network 21, 27, 53, 114

D

data fusion bus 11, 14
datacentre carbon footprint computation 48
datacentre design 48
decision support system 6, 11, 12
deep edge AI 256, 261
deep learning (DL) 105, 133, 151, 188, 253, 287, 300, 326
deep neural networks 132, 139, 172, 188, 257, 294
deep reinforcement learning 7, 35, 355

digital twin 47, 50, 64, 71, 253, 355

E

edge computing 109, 179, 205,258, 287, 300, 326, 343
electrical time domain reflectometry (TDR) 184, 239, 241
electro-sensitive protective equipment 188
embedded artificial intelligence 132
embedded systems 188, 364
energy efficiency of datacentre 48
energy efficient metrics 48
expert system 54, 106, 110, 118, 338

F

failure analysis 107, 121, 148
fault injection 8, 56, 63, 64, 66
fault model 64
field-programmable gate array 132
fruit counting 277

G

gesture recognition 184, 221, 223, 228
grape detection 257, 279, 287

H

hardware acceleration of AI 132
hardware trust 148
high-level synthesis 132, 142
human machine interaction 105, 179, 239
human-robot interaction discrete robot control 221
HW/SW integration 132

I

image classification 162
image processing 106, 148, 153, 265
image segmentation 152, 207, 277, 357
inbound logistics 6, 11
industrial artificial intelligence 106, 148
industrial automation 149, 179
industrial internet of intelligent things 300
industrial internet of things (IIoT) 49, 179, 251, 300, 325, 326
industrial robotics 7, 35, 37
information extraction 114, 116, 126, 241
intelligent transport systems (ITS) 353
internet of things 3, 49, 148, 189, 261
inter-turn short circuit fault 83, 85, 90, 92

K

knowledge graph 110, 113, 124, 125
knowledge representation 110, 114–117, 121, 123

L

LiDAR sensor 257, 277, 283, 358
link prediction 114, 123
lithium-ion battery 21, 25
LoRaWAN 257, 261, 267, 269–274

M

machine learning (ML) 105, 133, 180, 195, 221, 223, 300, 326
machine vision 188, 219, 299, 300, 308
manufacturing 3, 35, 105, 125, 150

micro-doppler 221, 225
mobility-as-a-service (MaaS) 353
model based diagnosis 47, 51, 67, 68, 71
model based reasoning 48, 55, 65, 338
modular neural network 83, 90
motion planning 35, 214
multilayer perceptron 16, 17, 95
multiphase PMS motor 84

N

natural language processing 106, 110, 114, 123
neural networks 5, 15, 47, 53, 106, 134, 172
neuromorphic computing 221

O

object detection 205, 214, 216, 278, 358
offline programming 7, 35, 42
optimisation 3, 11, 251, 255, 370

P

pattern recognition 106, 148, 155, 255, 344
PLC 193, 300, 342
precision viticulture 277, 287
predictive maintenance 3, 63, 83, 179, 251, 325, 329
production optimisation 251, 254, 299, 300, 314

Q

quantized neural networks 132, 135

R

random forest classification 221
real time analytics 11, 14
real-time artificial intelligence of things (RT-AIoT) 258, 299, 300, 301
real-time systems 335
recurrent neural network 31, 114, 196, 229, 339
reinforcement learning 7, 35, 320, 361
reliability 8, 108, 189, 333, 355
retired electric vehicle battery 21
robot learning 35
robot operating system 205
root cause analysis and risk assessment 113, 114, 118
root cause analysis 47, 63, 105, 113

S

SCADA 258, 265, 300
scikit-learn 11, 15
second-life 7, 21, 23
semantic segmentation 148, 156, 289, 294
semiconductor industry 105, 106, 108, 113, 116
semiconductor technology analysis 148
sensitive robot skin 184, 239
sensor data fusion 361, 363, 365
sensor development 239
simulation based diagnosis 48, 71, 74, 75
simulation 7, 35, 63, 92, 182
smart manufacturing 50, 179, 181, 205
smart robot 179, 205, 211
smart sensor systems 300, 326
soybean manufacturing 251, 300, 326
spiking neural networks 184, 221, 228

state of health 21, 24, 27, 31
stationary battery system 21
supervised learning 53, 111, 148, 162, 168, 171, 333
synthetic data generation 53, 205, 212

T
temporal convolutional networks safety-critical 188
temporal convolutional neural network 21, 27
Tensorflow-lite 287
text classification 114, 118
time series analysis 188
TinyML 188, 189
touch control 239
training and validation 77, 86, 92, 184, 218, 239, 290
training 8, 38, 92, 97, 99, 139
trajectory optimization 35

U
ultrasound processing 188

V
validation 64, 77, 84, 200, 202, 207, 213
verification and validation 114, 157, 205, 209
vibration analysis 326, 335, 343, 346
vibration diagnosis 85, 93, 94
vine balance 277, 283
vineyards 256, 261, 272, 283
virtual learning platform 35

W
wafer maps 131, 132, 143, 144
wirebonding 162, 170

Y
yield estimation 257, 277, 280, 287

Artificial Intelligence for Digitising Industry Applications

**Ovidiu Vermesan, Reiner John,
Cristina De Luca, Marcello Coppola**

This book provides in-depth insights into use cases implementing artificial intelligence (AI) applications at the edge. It covers new ideas, concepts, research, and innovation to enable the development and deployment of AI, the industrial internet of things (IIoT), edge computing, and digital twin technologies in industrial environments. The work is based on the research results and activities of the AI4DI (ECSEL JU) project, including an overview of industrial use cases, research, technological innovation, validation, and deployment.

This book's sections build on the research, development, and innovative ideas elaborated for applications in five industries: automotive, semiconductor, industrial machinery, food and beverage, and transportation.

The articles included under each of these five industrial sectors discuss AI-based methods, techniques, models, algorithms, and supporting technologies, such as IIoT, edge computing, digital twins, collaborative robots, silicon-born AI circuit concepts, neuromorphic architectures, and augmented intelligence, that are anticipating the development of Industry 5.0.

Automotive applications cover use cases addressing AI-based solutions for inbound logistics and assembly process optimisation, autonomous reconfigurable battery systems, virtual AI training platforms for robot learning, autonomous mobile robotic agents, and predictive maintenance for machines on the level of a digital twin.

AI-based technologies and applications in the semiconductor manufacturing industry address use cases related to AI-based failure modes and effects analysis assistants, neural networks for predicting critical 3D dimensions in MEMS inertial sensors, machine vision systems developed in the wafer inspection production line, semiconductor wafer fault classifications, automatic inspection of scanning electron microscope cross-section images for technology verification, anomaly detection on wire bond process trace data, and optical inspection.

The use cases presented for machinery and industrial equipment industry applications cover topics related to wood machinery, with the perception of the surrounding environment and intelligent robot applications.

AI, IIoT, and robotics solutions are highlighted for the food and beverage industry, presenting use cases addressing novel AI-based environmental monitoring; autonomous environment-aware, quality control systems for Champagne production; and production process optimisation and predictive maintenance for soybeans manufacturing.

For the transportation sector, the use cases presented cover the mobility-as-a-service development of AI-based fleet management for supporting multimodal transport.

This book highlights the significant technological challenges that AI application developments in industrial sectors are facing, presenting several research challenges and open issues that should guide future development for evolution towards an environment-friendly Industry 5.0.

The challenges presented for AI-based applications in industrial environments include issues related to complexity, multidisciplinary and heterogeneity, convergence of AI with other technologies, energy consumption and efficiency, knowledge acquisition, reasoning with limited data, fusion of heterogeneous data, availability of reliable data sets, verification, validation, and testing for decision-making processes.

ISBN 978-87-7022-664-6

River Publishers