Springer Series in Astrophysics and Cosmology

Elena Cuoco *Editor*

Gravitational Wave Science with Machine Learning

Springer

Springer Series in Astrophysics and Cosmology

The series covers all areas of astrophysics and cosmology, including theory, observations, and instrumentation. It publishes monographs and edited volumes. All books are authored or edited by leading experts in the field and are primarily intended for researchers and graduate students.

Elena Cuoco
Editor

Gravitational Wave Science with Machine Learning

Springer

Editor
Elena Cuoco
DIFA-Department of Physics
and Astronomy
Alma Mater Studiorum Bologna University
and INFN Bologna Unit
Bologna, Italy

ISSN 2731-734X ISSN 2731-7358 (electronic)
Springer Series in Astrophysics and Cosmology
ISBN 978-981-96-1736-4 ISBN 978-981-96-1737-1 (eBook)
https://doi.org/10.1007/978-981-96-1737-1

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

I want to express my gratitude to all the authors who enthusiastically contributed to this volume. Our journey began in 2018 with COST Action CA17137 and has progressed thanks to the dedication and energy of those seeking Machine Learning-based solutions for gravitational wave science. I dedicate this volume to everyone who passionately embraces innovative and forward-thinking ideas.

Preface I

In this collection, our objective was to curate diverse contributions highlighting innovative machine learning solutions to search and analyze gravitational waves. The past few years have witnessed a significant increase in the application of machine learning techniques within gravitational wave research, resulting in the groundbreaking use of new methodologies.

Through COST Action CA17137, we assembled a multidisciplinary European team of scientists from diverse backgrounds to explore and apply innovative machine learning solutions to emerging challenges in gravitational wave research. Our work encompassed noise analysis, control techniques, and the study of astrophysical signals.

This collection is designed to serve as a comprehensive resource for both newcomers and those eager to apply advanced techniques in the field. By gathering the latest advancements and insights, we aim to deepen understanding and inspire further innovation in gravitational wave science and demonstrate the transformative potential of machine learning in advancing gravitational wave research.

The book features a range of contributions from our collaborative efforts, showcasing the breadth of studies and results achieved.

Bologna, Italy — Elena Cuoco
September 2024

Preface II

The groundbreaking discovery of gravitational waves on September 14, 2015 was an achievement made possible by the synergy of techniques and expertise from various scientific disciplines. This historic event marked the dawn of a new era in astronomy and underscored the importance of interdisciplinary collaboration.

The use of Machine Learning, Deep Learning, and advanced data science techniques has surged across a wide range of disciplines, from the Social Sciences to the Natural Sciences. These fields are increasingly tackling challenges such as classification, data mining, and visualization to manage and analyze the massive and complex datasets characteristic of the "Big Data" era. With exponential increases in computing power and the development of advanced algorithms for rapid data analysis, Gravitational Wave Astronomy stands on the brink of a transformative revolution.

Specific areas of focus within gravitational wave science include the development of control and feedback systems for next-generation detectors, noise removal, data analysis, and data-conditioning tools. The discovery of gravitational wave signals from colliding binary black holes has revealed the existence of a previously unobservable population of massive, stellar-origin black holes. This has underscored the importance of analyzing low-frequency gravitational wave data, a mission that is central to gravitational wave science. The performance of Earth-based gravitational wave detectors at low frequencies is heavily influenced by the ability to suppress ambient seismic noise, making this a critical area of research.

During COST Action CA17137, we actively fostered collaboration and knowledge exchange across diverse fields to develop and enhance the techniques and tools essential for advancing gravitational wave detection and analysis.

Bologna, Italy
September 2024

Elena Cuoco

Acknowledgements

This publication is based upon work from COST Action CA17137, supported by COST (European Cooperation in Science and Technology).

For the Chap. 12 "Searching for Long-Duration Transient Gravitational Waves: Convolutional Neural Networks Applied to Glitching Pulsars", the authors thank the ULìege group from the STAR Institute for hosting L.M.M. during a g2net (CA17137) short-term scientific mission working on this project, and the members of g2net, the GRAVITY group at UIB as well as the Continuous Waves working group for fruitful discussions. This work was supported by the Universitat de les Illes Balears (UIB); the Spanish Agencia Estatal de Investigación grants CNS2022-135440, PID2022-138626NB-I00, RED2022-134204-E, RED2022-134411-T, funded by MICIU/AEI/10.13039/501100011033, the European Union NextGenerationEU/PRTR, and the ERDF/EU; and the Comunitat Autònoma de les Illes Balears through the Direcció General de Recerca, Innovació I Transformació Digital with funds from the Tourist Stay Tax Law (PDR2020/11—ITS2017-006) as well as through the Conselleria d'Economia, Hisenda i Innovació with grant numbers SINCO2022/6719 (European Union NextGenerationEU/PRTR-C17.I1) and SINCO2022/18146 (cofinanced by the European Union and FEDER Operational Program 2021-2027 of the Balearic Islands); and EU COST Actions CA18108 and CA17137. During the original work on the study published in [1], in addition, DK was supported by the Spanish Ministerio de Ciencia, Innovación y Universidades (ref. BEAGAL 18/00148) and cofinanced by the Universitat de les Illes Balears, LMM was supported by the Universitat de les Illes Balears, and RT was supported by the Spanish Ministerio de Ciencia, Innovación y Universidades (ref. FPU 18/00694). This material is based upon work supported by NSF's LIGO Laboratory, which is a major facility fully funded by the National Science Foundation. The authors gratefully acknowledge the computer resources at Artemisa, funded by the European Union ERDF and Comunitat Valenciana as well as the technical support provided by the Instituto de Física Corpuscular, IFIC (CSIC-UV), and computational resources provided by the LIGO Laboratory and supported by National Science Foundation Grants PHY-0757058 and PHY-0823459.

The work presented in the Chap. 14 "Detecting Gravitational Waves as Anomalies with Convolutional Autoencoders" was partially supported by the Polish NCN grants

no. 2016/22/E/ST9/00037, 2017/26/M/ST9/00978 and 2020/37/N/ST9/02151, as well as the European Cooperation in Science and Technology COST action G2Net (CA17137). The Quadro P6000 used in this research was donated by the NVIDIA Corporation. The production computations were supported in part by PL-Grid Infrastructure and by the MNiSW grant for expansion of the strategic IT infrastructure for science. This research has made use of data, software and/or web tools obtained from the Gravitational Wave Open Science Center (https://www.gw-openscience.org/), a service of LIGO Laboratory, the LIGO Scientific Collaboration and the Virgo Collaboration. LIGO Laboratory and Advanced LIGO are funded by the United States National Science Foundation (NSF) as well as the Science and Technology Facilities Council (STFC) of the United Kingdom, the Max-Planck-Society (MPS), and the State of Niedersachsen/Germany for support of the construction of Advanced LIGO and construction and operation of the GEO600 detector. Additional support for Advanced LIGO was provided by the Australian Research Council. Virgo is funded, through the European Gravitational Observatory (EGO), by the French Centre National de Recherche Scientifique (CNRS), the Italian Istituto Nazionale di Fisica Nucleare (INFN) and the Dutch Nikhef, with contributions by institutions from Belgium, Germany, Greece, Hungary, Ireland, Japan, Monaco, Poland, Portugal, Spain.

The work presented in the Chap. 17 "Detecting Gravitational Waves From Binary Black Hole Mergers Using Deep Convolutional Neural Networks and Quadratic Time-Frequency Distributions" was carried out within the framework of the EU COST action CA17137. This work has been partially supported by the University of Rijeka project uniri-mladi-tehnic-23-15 and the project line ZIP UNIRI of the University of Rijeka, for the project UNIRI-ZIP-2103-4-22.

The authors of the Chap. 1 "Neural Network Time-Series Classifiers for Gravitational-Wave Searches in Single-Detector Periods" acknowledge the partial support of the European Union's Horizon 2020 research and innovation program under grant agreement No 653477, the diiP (data intelligence institute of Paris), the IdEx Université de Paris, the ANR-18-IDEX-0001, the COST action G2net (CA 17137) and the ICSC—Centro Nazionale di Ricerca in High Performance Computing, Big Data and Quantum Computing, funded by European Union—NextGenerationEU. MB gratefully acknowledges partial support from the Polish National Science Centre grants no. 2016/22/E/ST9/00037 and 2021/43/B/ST9/01714, and Poland's high-performance Infrastructure PLGrid (ACK Cyfronet AGH). The authors are grateful for computational resources provided by the LIGO Laboratory and supported by National Science Foundation Grants PHY-0757058 and PHY-0823459. This work was granted access to the HPC resources of IDRIS under the allocation 2021- A0111012956 and 2021-AD011012279 made by GENCI. Some of the numerical computations were performed on the DANTE platform, APC, France. This work was partially supported by the PRIN project no. 202275HT58 by the Italian Ministry for Universities and Research.

For the research covered in Chap. 19 "Convolutional Neural Networks for Signal Detection in Real LIGO Data", O.Z. thanks the Carl Zeiss Foundation for the financial support within the scope of the program line "Breakthroughs" and is supported

by the fellowship Lumina Quaeruntur No. LQ100032102 of the Czech Academy of Sciences. Further support has been provided by the COST network CA17137 "G2net". The computational experiments were performed on the ARA cluster at the Friedrich-Schiller-Universität Jena and the Atlas cluster financed by the Gottfried Wilhelm Leibniz Universität Hannover and the Max-Planck-Gesellschaft through the Albert-Einstein-Institut Hannover. We thank the Observational Relativity and Cosmology division for access. F.O. acknowledges support from the Max Planck Independent Research Group Program. Special thanks go to Marlin Schäfer for his great contribution through his work on organizing the MLGWSC-1 as well as discussions, and to all contributors to the challenge.

This research has made use of data or software obtained from the Gravitational Wave Open Science Center (https://www.gwosc.org), a service of the LIGO Scientific Collaboration, the Virgo Collaboration, and KAGRA. This material is based upon work supported by NSF's LIGO Laboratory which is a major facility fully funded by the National Science Foundation, as well as the Science and Technology Facilities Council (STFC) of the United Kingdom, the Max-Planck-Society (MPS), and the State of Niedersachsen/Germany for support of the construction of Advanced LIGO and construction and operation of the GEO600 detector. Additional support for Advanced LIGO was provided by the Australian Research Council. Virgo is funded, through the European Gravitational Observatory (EGO), by the French Centre National de Recherche Scientifique (CNRS), the Italian Istituto Nazionale di Fisica Nucleare (INFN) and the Dutch Nikhef, with contributions by institutions from Belgium, Germany, Greece, Hungary, Ireland, Japan, Monaco, Poland, Portugal, Spain. KAGRA is supported by Ministry of Education, Culture, Sports, Science and Technology (MEXT), Japan Society for the Promotion of Science (JSPS) in Japan; National Research Foundation (NRF) and Ministry of Science and ICT (MSIT) in Korea; Academia Sinica (AS) and National Science and Technology Council (NSTC) in Taiwan. The authors of the Chap. 5 "Denoising Gravitational-Wave Signals from Binary Black Holes with Dilated Convolutional Autoencoder" would like to acknowledge Tom Charnock and Florian Fuhrer for fruitful discussions, and Eric Chassande-Mottin for his contribution on the dataset preparation. This work was supported in part by the PLGrid infrastructure with the computing grant on the ACK Cyfronet AGH Prometheus cluster, the grant of the Polish Ministry of Science and Higher Education (MNiST) for the expansion of the IT infrastructure at the Nicolaus Copernicus Astronomical Center, and the National Science Centre grants no. 2016/22/E/ST9/00037 and 2021/43/B/ST9/01714. As authors of Chap. 7 "AI-Powered Charge Monitoring in Interferometers" we express our gratitude to the Virgo Collaboration, particularly Ettore Majorana, for providing the CAD model of the NE payload, the COMSOL Multiphysics license was generously supplied by the Istituto Nazionale di Fisica Nucleare (INFN), and the MATLAB license was graciously provided by the University of Genova (UNIGE).

Reference

1. Trovato, A. et al.: Neural network time-series classifiers for gravitational-wave searches in single-detector periods. Class. Quantum Grav. **41**, 125003 (2024)

Contents

Contributors

Marc Andrés-Carcasona Institut de Física d'Altes Energies (IFAE), The Barcelona Institute of Science and Technology, Bellaterra (Barcelona), Spain

Elena-Simona Apostol National University of Science and Technology Politehnica Bucharest, Bucharest, Romania

Federico Armato UNIGE, Department of Physics, Genova, Italy

Philippe Bacon INFN Sezione di Ferrara, Ferrara, Italy;
Nicolaus Copernicus Astronomical Center, Polish Academy of Sciences, Warszawa, Poland;
Université Paris Cité, CNRS, Paris, France;
Dipartimento di Fisica, Università di Trieste, Trieste, Italy;
INFN, Sezione di Trieste, Trieste, Italy

Francesca Badaracco Dipartimento di Fisica, Genova, Italy

M. Bejger INFN, Sezione di Ferrara, Ferrara, Italy;
Nicolaus Copernicus Astronomical Center, Polish Academy of Sciences, Warsaw, Poland

Michał Bejger Nicolaus Copernicus Astronomical Center, Polish Academy of Sciences, Warsaw, Poland;
INFN Sezione di Ferrara, Ferrara, Italy;
Université Paris Cité, CNRS, Paris, France;
Dipartimento di Fisica, Università di Trieste, Trieste, Italy;
INFN, Sezione di Trieste, Trieste, Italy

Osman Tayfun Bişkin Department of Electrical-Electronics Engineering, Burdur Mehmet Akif Ersoy University, Burdur, Turkey

Bernd Brügmann Friedrich-Schiller-Universität Jena, Jena, Germany;
Michael Stifel Center Jena, Jena, Germany

Marco Cavaglià Institute of Multi-messenger Astrophysics and Cosmology, Missouri University of Science and Technology, Rolla, MO, USA

E. Chassande-Mottin Université Paris Cité, CNRS, Paris, France

Andrea Chincarini INFN, Sezione Genova, Genova, Italy

Roberto Corizzo Department of Computer Science, American University, Washington DC, USA

N. Courty Université Bretagne Sud, CNRS IRISA, Rennes, France

Elena Cuoco Department of Physics and Astronomy, University of Bologna and INFN Bologna, Viale Berti Pichat, Bologna, Italy

Francesco di Renzo Université Lyon, Université Claude Bernard Lyon 1, CNRS, Villeurbanne, France

Ali Çelik Department of Physics, Burdur Mehmet Akif Ersoy University, Burdur, Turkey

Maxime Fays Universite de Liege, Liege, Belgium

R. Flamary Ecole polytechnique, IP Paris, Palaiseau, France

José A. Font Departament d'Astronomia i Astrofísica, Universitat de València, Burjassot (València), Spain;
Observatori Astronòmic, Universitat de València, Paterna (València), Spain

Styliani-Christina Fragkouli Institute Of Applied Biosciences, Centre for Research and Technology Hellas, Thessaloniki, Greece

Franko Hržić Center for Artificial Intelligence and Cybersecurity, University of Rijeka, Rijeka, Croatia;
Faculty of Engineering, University of Rijeka, Rijeka, Croatia

Alberto Iess Scuola Normale Superiore, Pisa, Italy

David Keitel Departament de Física, Universitat de les Illes Balears, Palma, Spain

İsmail Kirbaş Department of Computer Engineering, Burdur Mehmet Akif Ersoy University, Burdur, Turkey

Alexandra Eleni Koloniari Department of Physics, Aristotle University of Thessaloniki, Thessaloniki, Greece

Jonatan Lerga Center for Artificial Intelligence and Cybersecurity, University of Rijeka, Rijeka, Croatia;
Faculty of Engineering, University of Rijeka, Rijeka, Croatia

Miquel Llorens-Monteagudo Departament d'Astronomia i Astrofísica, Universitat de València, Burjassot (València), Spain

Nikola Lopac Faculty of Maritime Studies, University of Rijeka, Rijeka, Croatia; Center for Artificial Intelligence and Cybersecurity, University of Rijeka, Rijeka, Croatia

Melissa Lopez Institute for Gravitational and Subatomic Physics (GRASP), Utrecht University, Utrecht, The Netherlands;
Nikhef, Amsterdam, The Netherlands

Antonio Marquina Departament de Matemàtiques, Universitat de València, Burjassot (València), Spain

Mario Martínez Institut de Física d'Altes Energies (IFAE), The Barcelona Institute of Science and Technology, Bellaterra (Barcelona), Spain;
Catalan Institution for Research and Advanced Studies (ICREA), Barcelona, Spain

Alexis Menéndez-Vázquez Institut de Física d'Altes Energies (IFAE), The Barcelona Institute of Science and Technology, Bellaterra (Barcelona), Spain

Lluïsa-Maria Mir Institut de Física d'Altes Energies (IFAE), The Barcelona Institute of Science and Technology, Bellaterra (Barcelona), Spain

Luana M. Modafferi Departament de Física, Universitat de les Illes Balears, Palma, Spain

Filip Morawski Nicolaus Copernicus Astronomical Center, Polish Academy of Sciences, Warsaw, Poland

Conor Muldoon Department of Computing and Mathematics, Manchester Metropolitan University, Manchester, UK

Luca Naticchioni INFN sez. di Roma, Rome, Italy

Paraskevi Nousi Swiss Data Science Center, ETH, Zürich, Switzerland

Frank Ohme Max-Planck-Institut für Gravitationsphysik, Albert-Einstein-Institut, Hannover, Germany;
Leibniz Universität Hannover, Hannover, Germany

Luigia Petre Faculty of Science and Technology, Åbo Akademi University, Turku, Finland

Jade Powell OzGrav, Centre for Astrophysics and Supercomputing, Swinburne University of Technology, Hawthorn, Melbourne, Australia

Stefano Schmidt Institute for Gravitational and Subatomic Physics (GRASP), Utrecht University, Utrecht, The Netherlands;
Nikhef, Amsterdam, The Netherlands

Nikolaos Stergioulas Department of Physics, Aristotle University of Thessaloniki, Thessaloniki, Greece

Anastasios Tefas Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece

Rodrigo Tenorio Departament de Física, Universitat de les Illes Balears, Palma, Spain

Alejandro Torres-Forné Departament d'Astronomia i Astrofísica, Universitat de València, Burjassot (València), Spain;
Observatori Astronòmic, Universitat de València, Paterna (València), Spain

A. Trovato Dipartimento di Fisica, Università di Trieste, Trieste, Italy;
INFN, Sezione di Trieste, Trieste, Italy

Agata Trovato INFN Sezione di Ferrara, Ferrara, Italy;
Nicolaus Copernicus Astronomical Center, Polish Academy of Sciences, Warszawa, Poland;
Université Paris Cité, CNRS, Paris, France;
Dipartimento di Fisica, Università di Trieste, Trieste, Italy;
INFN, Sezione di Trieste, Trieste, Italy

Ciprian-Octavian Truică National University of Science and Technology Politehnica Bucharest, Bucharest, Romania

Eftim Zdravevski Faculty of Computer Science and Engineering, University Ss. Cyril and Methodius, Skopje, North Macedonia

Ondřej Zelenka Astronomical Institute of the Czech Academy of Sciences, Prague, Czech Republic;
Friedrich-Schiller-Universität Jena, Jena, Germany;
Michael Stifel Center Jena, Jena, Germany

Acronyms

1D	One-Dimensional
2D	Two-Dimensional
3G	Third Generation
ADAM	ADAptive Moment estimation algorithm
AE	Auto Encoder
ASD	Amplitude Spectral Density
BBH	Binary Black Hole
BH	Black Hole
BJD	Born-Jordan Distribution
BNS	Binary Neutron Star
BUD	Butterworth Distribution
CBC	Compact Binary Coalescence
CCSN	Core-Collapse Supernova
CNN	Convolutional Neural Network
CW	Continuous Wave
cWB	Coherent Wave Burst
CWD	Choi-Williams Distribution
DAE	Denoising Auto Encoder
DL	Deep Learning
EOB	Effective-One-Body
ET	Einstein Telescope
ETG	Event-Trigger-Generator
FAR	False Alarm Rate
FPR	False Positive Rate
GAN	Generative Adversarial Networks
GP	Gaussian Process
GPS	Global Positioning System
GPU	Graphics Processing Unit
GW	Gravitational Wave
GWOSC	Gravitational Wave Open Science Center
IFAR	Inverse False Alarm Rate

IMF	Initial Mass Function
IT	Inception Time
LB	Large Baffle
LF	Low Frequency
LIGO	Laser Interferometer Gravitational-wave Observatory
LISA	Laser Interferometer Space Antenna
LRSDL	Low-Rank Shared Dictionary Learning
LSTM	Long Short-Term Memory
LVK	LIGO, Virgo and KAGRA Collaborations
ML	Machine Learning
MSE	Mean Square Error
NE	North End
NEMO	Neutron Star Extreme Matter Observatory
NN	Neural Network
NN	Newtonian Noise
NOMF-SNR	Network Optimal Matched-Filter Signal-to-Noise Ratio
NS	Neutron Star
O1	First joint observation run of LIGO and VIRGO, 12.09.2015-19.01.2016
O2	Second joint observation run of LIGO and VIRGO, 30.11.2016-25.08.2017
O3	Third joint observation run of LIGO and VIRGO, 01.04.2019-30.09.2019
PAY	Payload
PC	Principal Component
PCA	Principal Component Analysis
PR AUC	Area Under the Precision-Recall Curve
PSD	Power Spectral Density
PWVD	Pseudo Wigner-Ville Distribution
ReLU	Rectified Linear Unit
RH	Ring Heater
RID	Reduced-Interference Distribution
RIDB	Reduced-Interference Distribution with a Kernel Based on the First Kind Bessel Function
RIDBN	Reduced-Interference Distribution with a Kernel Based on Binomial Coefficients
RIDH	Reduced-Interference Distribution with a Kernel Based on the Hanning Window
RIDT	Reduced-Interference Distribution with a Kernel Based on the Triangular Window
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
ROC AUC	Area Under the Receiver Operating Characteristic Curve
SASI	Standing Accretion Shock Instability
SDL	Sparse Dictionary Learning

SFT	Short Fourier Transform
SNR	Signal-to-Noise Ratio
SP	Spectrogram
SPWVD	Smoothed Pseudo Wigner-Ville Distribution
SSIM	Structural Similarity Index Measure
STFT	Short-Time Fourier Transform
TCN	Temporal Convolutional Network
tCW	Transient Continuous Wave
TFD	Time-Frequency Distribution
TM	Test Mass
TPR	True Positive Rate
Virgo	Virgo Gravitational Wave Observatory
WGAN	Wasserstein Generative Adversarial Network
WVD	Wigner-Ville Distribution
ZAMD	Zhao-Atlas-Marks Distribution

Part I
Machine Learning for Gravitational Wave Detector Noise Analysis

One of the primary challenges in gravitational wave science is the presence of noise with diverse characteristics, which can complicate the detection of signals of astrophysical origin. Consequently, substantial efforts have been devoted to enhancing the understanding of detector noise, identifying its non-stationary and nonlinear nature, and pinpointing the major sources of this noise.

The data we analyze from gravitational wave detectors are time series sampled at frequencies high enough to allow for the detection of the most probable signal sources. Among the most troublesome types of noise are the so-called glitches, transient signals caused by noise sources that can resemble the signals we aim to detect. These glitches can, as has occurred in the past, overlap with genuine gravitational wave signals, posing significant challenges.

For this reason, the initial applications of machine learning techniques within our community focus on the detection, classification, and potential removal of these transient noise signals. Through machine learning, we aim to develop robust methodologies that can distinguish between genuine astrophysical signals and noise, thereby improving the accuracy and reliability of gravitational wave detections.

In addressing these challenges, our research endeavors to create a comprehensive framework that integrates advanced machine learning algorithms with traditional data analysis techniques. This interdisciplinary approach not only enhances our ability to identify and mitigate noise but also paves the way for the discovery of new gravitational wave sources and phenomena. The ongoing collaboration among physicists, data scientists, and engineers is crucial in driving forward the capabilities of gravitational wave observatories and in unraveling the mysteries of the universe.

In the following, you will find a series of chapters dedicated to the study of noise, the analysis of glitches, their classification, and proposals for innovative analytical methods to be employed in future scientific runs of terrestrial gravitational wave detectors.

Each chapter delves into specific aspects of noise characterization, offering a comprehensive examination of the various sources and types of noise that can affect gravitational wave data.

The chapters on classification present advanced techniques for distinguishing between true astrophysical signals and noise.

Additionally, this collection features forward-looking proposals for new analytical approaches. These innovative techniques are designed to address the evolving challenges faced by the next generation of terrestrial gravitational wave detectors. By integrating these novel methods, we aim to improve the sensitivity and accuracy of future scientific runs, enabling the discovery of new gravitational wave sources and deepening our understanding of the universe.

Through this comprehensive resource, we hope to provide valuable insights and practical tools for researchers in the field of gravitational wave science, fostering continued advancements and breakthroughs in this exciting area of study.

Chapter 1
Neural Network Time-Series Classifiers for Gravitational-Wave Searches in Single-Detector Periods

A. Trovato, E. Chassande-Mottin, M. Bejger, R. Flamary, and N. Courty

Abstract This chapter contains a summary of the paper in Ref. [1], in which the challenges in detecting gravitational-wave (GW) signals, especially when only one detector is operating, are discussed. The single detector case is particularly difficult to analyse since a very useful tool to distinguish astrophysical signals from instrumental glitches cannot be used, i.e. the temporal coincidence between detectors. Neural network classifiers are explored, including convolutional neural networks, temporal convolutional networks, and inception time, specifically tailored for time-series data processing. The classifiers are trained on a subset of data from the LIGO Livingston detector during the first observing run (O1) to identify segments containing binary black hole merger signatures. Their performances are evaluated and compared. Subsequently, these trained classifiers are applied to the remaining O1 data, particularly focusing on single-detector times, with the most promising candidate from this search identified at the time 2016-01-04 12:24:17 UTC.

A. Trovato (✉)
Dipartimento di Fisica, Università di Trieste, 34127 Trieste, Italy
e-mail: agata.trovato@units.it

INFN, Sezione di Trieste, 34127 Trieste, Italy

E. Chassande-Mottin
Université Paris Cité, CNRS, Astroparticule et Cosmologie, 75013 Paris, France

M. Bejger
INFN, Sezione di Ferrara, 44122 Ferrara, Italy

Nicolaus Copernicus Astronomical Center, Polish Academy of Sciences, ul. Bartycka 18, 00-716 Warsaw, Poland

R. Flamary
Ecole polytechnique, IP Paris, CMAP, 91120 Palaiseau, France

N. Courty
Université Bretagne Sud, CNRS IRISA, 35042 Rennes, France

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_1

1.1 Introduction

The LIGO, Virgo and KAGRA Collaborations (LVK) have already detected 90 GW events during the first 3 observing runs (called O1, O2 and O3) and more detections are expected during the analysis of the data from the fourth observing run (O4). The sources of these GW detections are mergers of Binary Black Holes (BBH) or Binary Neutron Stars (BNS) or mixed systems with one Black Hole (BH) and one Neutron Star (NS). The search for these kind of signals, for which accurate waveform models are available, usually employs the matched filtering technique [2], which consists in checking the correlation between the data and a set of template waveform models. The presence of non-Gaussian transient noises, called "instrumental glitches", that mimic astrophysical signals, represent a challenge for these searches. The use of temporal coincidence between detectors is useful to reduce contamination from these instrumental glitches but during periods with only one operational detector, coincidence-based techniques are unavailable, posing difficulties in distinguishing signals from glitches and measuring their statistical significance.

This concern is relevant because interferometers used to detect gravitational waves are challenging instruments to operate and maintain in observing mode. Consequently, it often occurs that only one detector is operational. During the O1, O2 and O4a observing runs, when only the two LIGO detectors were active, single-detector periods comprised roughly 30% of the observation time. With the addition of the Virgo detector to the network during O3, this fraction decreased to about 15% in O3a and 11% in O3b. In total, from O1 to O4a, the single-detector periods sum up to almost 8 months.

In the last observing runs, methods to assess the significance of single-detector triggers are starting to be used directly by pipelines like `GstLAL` and `PyCBC` normally employed by LVK to detect GW events [4, 5]. Concerning in particular O1, the authors of Ref. [6] discuss two candidate events found in single-detector periods, one on 2015-12-25 04:11:44 UTC observed with LIGO Hanford and the other on 2016-01-04 12:24:17 UTC observed with LIGO Livingston. This last event is ruled out as a possible gravitational-wave source because of the excess power observed in the residual after subtraction of the best-fit waveform.

On the other hand, as this book demonstrates, machine learning methods are beginning to be widely used in the field of GW astronomy, showing particular promise in terms of signal identification in the data. Previous studies in this direction have typically employed simulated Gaussian noise, or if real noise was used, they failed to achieve a low enough false alarm rate to be practically useful. The goal of the work described here is to detect GW signals in real noise, aiming for a false alarm rate on the order of two false alarms per day, which represents the minimum threshold for issuing public alerts [3].

The methodology used is described in detail in the following sections. Three different neural network architectures are trained and tested using data from one month of observations by the LIGO Livingston detector (L1) during O1, where no GW signals were detected (Sect. 1.2). Then, the best classifiers are applied to the

three remaining months of O1 where only for one data segment all the classifiers agrees that it contains a BBH signal (Sect. 1.3). This segment, which is consistent with the time of the candidate event found by [6] is further investigated and it is compatible with an astrophysical origin.

1.2 Training and Testing on One Month of O1 L1 Data

1.2.1 Datasets Description

The data used for training and testing are those recorded by the LIGO Livingston (L1) detector in the one month between November 25, 2015 (GPS time 1132444817) and December 25, 2015 (GPS time 1135036817). These data, publicly available via the Gravitational Wave Open Science Center (GWOSC) [7], do not contain any known GW signal detection. The initial data, sampled at 16 kHz, are downsampled to 2048 Hz, bandpass-filtered between 20 Hz and 1 kHz, and whitened using inverse amplitude spectral density (ASD) in the frequency domain. Subsequently, the data are divided into one-second non-overlapping segments and used to obtain three categories of segments labeled in the following way:

- noise: the data are compatible with stationary background noise, i.e., are free of transient instrumental artifacts (glitches) or known GW events or harware injections.
- glitch: the data include one or several transient instrumental artifacts (glitches). The times of glitches occurrence are obtained merging the list of glitches detected by the citizen science project `Gravity Spy` [8] with the loudest background triggers from the unmodeled transient search *coherent WaveBurst* (`cWB`) [9, 10].
- signal: the data include a (simulated) astrophysical signal, added to the stationary background noise. The signal injections consist in BBHs with component masses m_1 and m_2 chosen randomly, with $m_1 > m_2 \geq 10M_\odot$ and a total mass $M = m_1 + m_2$ uniformly distributed in $33M_\odot \leq M \leq 60M_\odot$. We consider non-spinning BH, so the dimensionless spin magnitudes χ_1 and χ_2 are set to 0. The phase at coalescence and polarization angle are uniformly drawn from $(0, 2\pi)$, and the inclination angle from $(0, \pi)$. Right ascension and declination are fixed to zero since focus lies on a single detector. The source's luminosity distance is scaled for a uniform distribution in optimal signal-to-noise ratio (SNR) between 8 and 20. The waveform model used is SEOBNRv4 [11]. Simulated signals are added randomly within a segment, ensuring the merger part is fully contained.

The training set comprises 250, 000 segments each for the noise and signal classes, and 70, 000 for the glitch class, with a 20% fraction reserved for validation. The testing set includes 500, 000 samples for both the noise and signal classes, and 80, 000 for the glitch class.

1.2.2 Classifier Architectures

We use three different types of neural networks architectures: Convolutional Neural Network (CNN) [12], Temporal Convolutional Network (TCN) [13] and Inception Time (IT) [14]. In all three cases, a coarse exploration of the parameter space is used to set the hyperparameters.

The structure of the CNN is detailed in Table 1.1 and consist in four convolutional layers and a fully connected layer.

The TCN architecture comprises a "TCN layer" [15] consisting of 6 dilated convolutional layers with 32 filters, a kernel size of 16, and default dilation factors of (1, 2, 4, 8, 16, 32), respectively. A dropout rate of 0.1 is applied. The output of the TCN layer feeds into a final dropout layer with a rate of 0.5, followed by a dense embedding layer to complete the model. For this case only it was necessary to downsample the data to 1024 Hz in order to have a receptive field [15] larger than the length of input sequence.

The hyperparameters used for the IT architecture are the same suggested by the authors of Ref. [14], i.e. an ensemble of five Inception Networks initialised randomly. Each Inception Networks contains 10 layers each one withe 32 filters of lengths 20, 40 and 80.

1.2.3 Comparison of the Classifiers Performances

With the hyperparameters of the three networks fixed, each classifier is trained 10 times with different (random) initializations of the model weights and dropouts.

The goal of each training is to minimize the categorical cross-entropy loss function, using a standard Adam optimizer and a batch size of 24. Throughout the training process, the area under the Receiver Operating Characteristics (ROC) curve [16] is

Table 1.1 Structure of the CNN considered in this study [1]. The type of the layer is either convolutional (Conv) or fully connected (Dense). The activation function is either the rectified linear unit (`relu`) or the `softmax` function [12]

Layer number	1	2	3	4	5
Type	Conv	Conv	Conv	Conv	Dense
Number of filters	256	128	64	64	–
Kernel size	16	8	8	4	–
Stride length	4	2	2	1	–
Activation function	`relu`	`relu`	`relu`	`relu`	`softmax`
Dropout rate	0.5	0.5	0.25	0.25	–
Max pooling	4	4	2	2	–

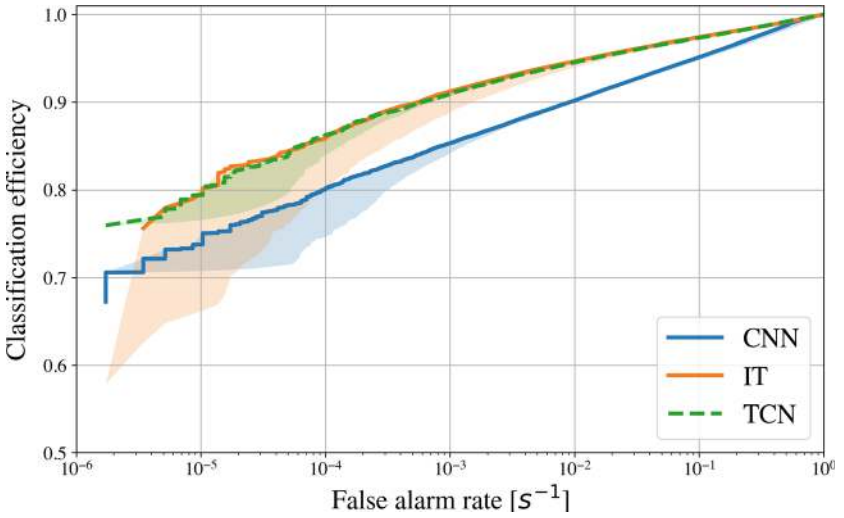


Fig. 1.1 ROC curves for the three considered classifiers, CNN (blue), IT (orange), and TCN (green), illustrating the classification efficiency versus the false alarm rate. Each classifier is trained 10 times. The solid (or dashed for TCN) line represents the result obtained for the best model, while the shaded area covers the range from the best to the worst model. The plot is a simplified version of Fig. 4 of Ref. [1]

evaluated on the validation data and stored for each epoch with the corresponding model. The model selected is the one with the highest area under the ROC curve.

After this training phase, the selected models are tested, and in Fig. 1.1, the ROC curves of the 10 models for each network are shown. To improve the readability of the plot, for each classifier type, the model with the best ROC is shown, and in each case, a shaded area covers the range from the best to the worst model.

The TCN and IT classifiers exhibit similar ROC curves, both demonstrating a notable improvement compared to CNN.

It's important to note that each classifier provides the probability that a given data segment belongs to any of the three classes. The ROC plots presented above were generated using a threshold based on the probability of being classified as a signal, denoted as P_s. For each threshold value, we calculated the fraction of correctly classified signals (classification efficiency) and the fraction of glitches or noise segments erroneously classified as signals (false alarm rate). This representation of the output allows, thus, to explore better what can be achieved with the networks with respect to, for example, a simple confusion matrix [16] which is usually built assuming that you would decide to which class your data belong according to the highest among the probabilities associated to each class.

For the subsequent analysis, we opt to concentrate on segments classified with $P_s = 1$, indicating the highest confidence in signal detection by the classifier. In the testing dataset, we identified 0, 1, and 2 segments classified as noise or glitch for

the CNN, TCN, and IT classifiers, respectively, meeting this selection criterion. This level of rejection power, with between 0 and 2 false alarms in 5.8×10^5 trials, aligns with the initially targeted false alarm rate.

1.3 Application to the Remaining O1 L1 Data

The classifiers are then applied to all the remaining L1 data in O1, excluding the month used for training and testing the classifiers (see Sect. 1.2) and intervals of $\pm$ 1 s around the merger time of the three known events in O1 (GW150914, GW151012 and GW151226). Following the same procedure described in Sect. 1.2, also in this case the data are downsampled, bandpass-filtered, whitened and divided into one-second non-overlapping segments.

In Table 1.2 the numbers of segments passing the selection cut $P_s = 1$ for the three classifiers are reported.

It's remarkable that only one segment meets the selection criteria for all three classifiers: GPS = 1135945474 (2016-01-04 12:24:17 UTC). We delve deeper into this segment in the following section.

A separate analysis of the 3 known events shows that only GW150914 would meet the requirement of $P_s = 1$ for all the classifiers. This is probably due to the fact that both GW151012 and GW151226 have single detector optimal SNRs for L1 lower than the minimum value of 8 used to train the network.

1.3.1 Detailed Analysis of the 2016-01-04 Event

To verify the possible astrophysical origin of the data in the segment starting at GPS = 1135945474 (2016-01-04 12:24:17 UTC), we perform various checks.

First, we visually inspect the segment using the time-frequency Q transform [17, 18]. Figure 1.2 illustrates the time-frequency representation of the entire segment. A transient appears approximately 0.37 s after the segment's start, with a frequency of around 150 Hz. Upon closer examination, the transient's shape strongly suggests a frequency-modulated chirp-like characteristic.

Table 1.2 Numbers of segments passing the selection cut $P_s = 1$ for the three classifiers

Classifier type	Tot. segments	Segments in single-detector time
CNN	4	2
TCN	105	14
IT	9	2

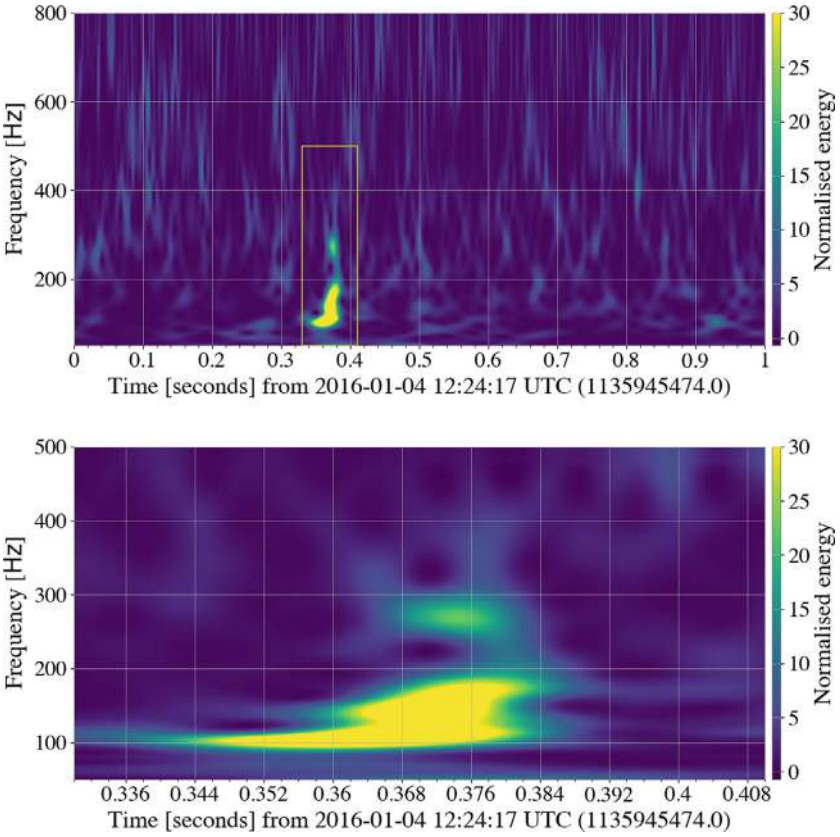


Fig. 1.2 Figure 8 from Ref. [1]. Time-frequency representation of the segment at 2016-01-04 12:24:17 UTC (GPS = 1135945474 s) recorded by the L1 detector. The top panel shows the entire segment. The bottom panel is a detailed view that focuses on the transient signal at $t \sim 0.37$ s. This representation is obtained through a Q transform [17] with quality factor $Q = 12$. To facilitate comparison, the dynamic range is fixed, following a similar approach as described in [6], and the colormap is saturated at a maximum value of 30 for the normalized energy

Second, we check if this GPS time exists in the `Gravity Spy` database [19], and we find that it is considered glitch of "Blip" type, a family of instrument glitches with an uncertain origin. We apply our classifiers to all the 600 Blip glitches identified by `Gravity Spy` in the part of the O1 dataset under analysis, and we find a compatibility with the overall background distribution (see Figs. 7 and A1 of Ref. [1]), except for the segment from January 4 which appears to be an outlier relative to the other Blip glitches.

Additionally, we attempt to fit the transient signal with a gravitational wave (GW) waveform model associated with a compact binary merger using the Bayesian inference library `Bilby` [20] and the `IMRPhenomXPHM` waveform model [21]. The analysis yield a signal-versus-noise log Bayes factor of 47. The estimated time of

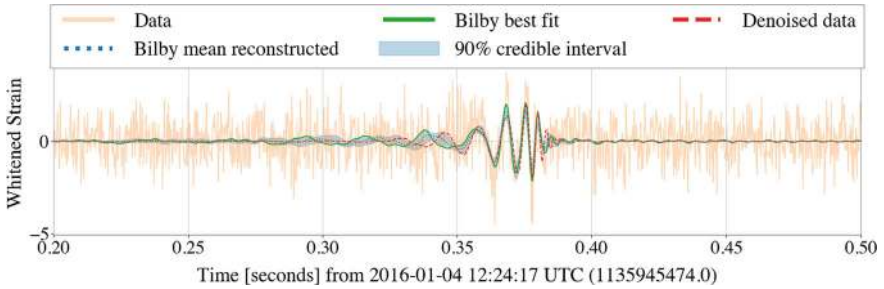


Fig. 1.3 Figure 9 from Ref. [1]. Comparison of the whitened L1 data (orange line) with the reconstructed waveform obtained from the posterior mean (dotted blue line), and from the maximum likelihood fit (solid green line) both computed using `Bilby`, and the 90% credible interval (blue) along with the ML denoising convolutional autoencoder neural network described in [22] (dashed red line)

arrival of the merger at the detector is GPS = $1135945474.373^{+0.076}_{-0.07}$. The fit results in the time domain are showed in Fig. 1.3 where the whitened data can be seen in orange, the maximum likelihood fit in green, and the posterior mean with the 90% credible belt in blue.

For an independent verification of the signal, Fig. 1.3 includes the waveform estimate from the denoising convolutional autoencoder (dashed red) described in Chap. 5 of this book (see [22] for more details). Besides the general agreement of the waveforms, it is interesting that the SNR of the denoised waveform is found to be 9.7 which, according to the discussion in [22], is sufficiently high to suggest an astrophysical origin.

Even more surprising is the absence of any significant residual after subtracting the best-fitting waveform (shown in green in Fig. 1.3), as depicted in Fig. 1.4.

All the aforementioned checks align with the event having an astrophysical origin. The 90% credible intervals for parameters found with the Bayesian analysis are: measured (redshifted) chirp mass $\mathcal{M} = 30.18^{+12.3}_{-7.3} M_\odot$, (redshifted) component masses $m_1 = 50.7^{+10.4}_{-8.9} M_\odot$ and $m_2 = 24.4^{+20.2}_{-9.3} M_\odot$, binary effective spin $\chi_{\text{eff}} = 0.06^{+0.4}_{-0.5}$ and luminosity distance $d_L = 564^{+812}_{-338}$ Mpc; see [23] for a definition of those physical parameters. These values align with the observed population of binary black holes up to this point.

1.4 Conclusion

This study showcases the effectiveness of training neural network classifiers on real data from ground-based gravitational wave detectors, particularly during single-detector observing periods. Our findings indicate that architectures tailored for time-series classification, such as IT or TCN, outperform conventional CNNs. Using one month of LIGO Livingston detector data from the O1 observing run for training and

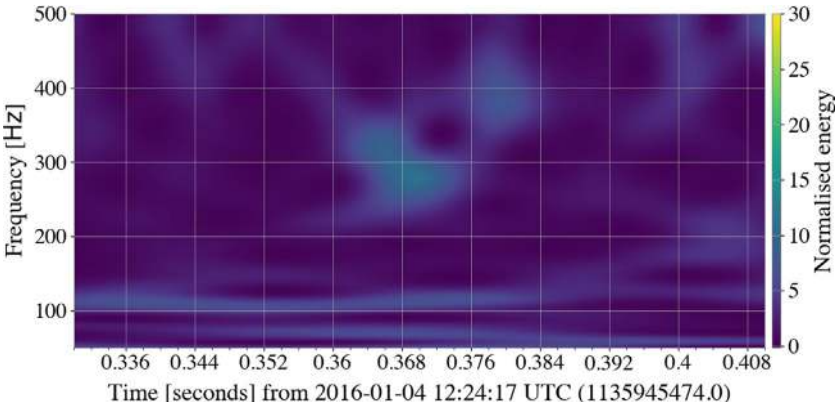


Fig. 1.4 Figure 10 from Ref. [1]. Time-frequency representation of the residual after the subtraction of the maximum likelihood fit waveform obtained with `Bilby` (green line in Fig. 1.3) from the data segment at 2016-01-04 12:24:17 UTC (GPS = 1135945474 s). This representation adheres to the same settings as in Fig. 1.2, utilizing a Q transform with a quality factor $Q = 12$ and the dynamic range is capped at a maximum of 30 for normalized energy, aligning with [6] to facilitate comparison. No excess power is visible in this plot

testing, these models accurately detect a potential gravitational wave signal of astrophysical origin on January 4, 2016, when applied to the remaining three months of O1 data. Various diagnostic tests support the plausibility of its astrophysical origin.

We propose an operational method where data from multiple detectors during the initial month of an observing run, labeled by standard matched filtering-based pipelines, are used to train neural network models. These classifiers can then be deployed on data collected during subsequent single-detector periods. Once trained, these classifiers offer low-latency triggers at a manageable computational cost.

References

1. Trovato, A., et al.: Neural network time-series classifiers for gravitational-wave searches in single-detector periods. Class. Quantum Grav. **41**, 125003 (2024)
2. Creighton, J., Anderson, W.: Gravitational-Wave Physics and Astronomy: An Introduction to Theory, Experiment and Data Analysis. Wiley, Hoboken, US (2011)
3. IGWN | Public Alerts User Guide, see Section "Alert Threshold Trials Factor". https://emfollow.docs.ligo.org/userguide/analysis/index.html
4. Sachdev, S., et al.: (2019). arXiv:1901.08580
5. Davies, G.S.C., Harry, I.W.: Class. Quantum Grav. **39**, 215012 (2022)
6. Nitz, A.H., et al.: A search for gravitational waves from binary mergers with a single observatory. APJ **897**, 169 (2020). https://doi.org/10.3847/1538-4357/ab96c7
7. Gravitational Wave Open Science Center. https://gwosc.org
8. Bahaadini, S., et al.: Machine learning for Gravity Spy: Glitch classification and dataset. Inf. Sci. **444**, 172–186 (2018). https://doi.org/10.1016/j.ins.2018.02.068

9. Drago, M., et al.: Coherent WaveBurst, a pipeline for unmodeled gravitational-wave data analysis. SoftwareX **14**, 100678 (2021). https://doi.org/10.1016/j.softx.2021.100678
10. Klimenko, S., et al.: cWB pipeline library: 6.4.0. https://doi.org/10.5281/zenodo.4419902
11. Bohé, A., et al.: Improved effective-one-body model of spinning, nonprecessing binary black holes for the era of gravitational-wave astrophysics with advanced detectors. Phys. Rev. D **95**, 044028 (2017)
12. Goodfellow, I.J., et al.: Deep Learning. MIT Press (2016). http://www.deeplearningbook.org
13. Bai, S., et al.: An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. https://doi.org/10.48550/arXiv.1803.01271
14. Ismail Fawaz, H. *et al.*: Inceptiontime: finding alexnet for time series classification. Data Min. Knowl. Disc. **34**, 1936–1962 (2020). https://doi.org/10.1007/s10618-020-00710-y
15. Remy, F.: Temporal Convolutional Networks for Keras. https://github.com/philipperemy/keras-tcn
16. Fawcett, T.: An introduction to ROC analysis. Pattern Recognit. Lett. **27**, 861–874 (2006). https://doi.org/10.1016/j.patrec.2005.10.010
17. Macleod, D.M., et al.: SoftwareX **13**, 100657 (2021)
18. Chatterji, S., et al.: Class. Quantum Grav. **21**, S1809–S1818 (2004)
19. Glanzer, J., et al.: 2021 Gravity Spy Machine Learning Classifications of LIGO Glitches from Observing Runs O1, O2, O3a, and O3b. https://doi.org/10.5281/zenodo.5649212
20. Ashton, G., et al.: APJSS **241**, 27 (2019)
21. Pratten, G., et al.: Phys. Rev. D **103**(10), 104056 (2021)
22. Bacon, P., Trovato, A., Bejger, M.: Mach. Learn. Sci. Technol. **4**, 035024 (2023)
23. Christensen, N., Meyer, R.: Rev. Mod. Phys. **94**, 025001 (2022)

Chapter 2
A Simple Self Similarity-Based Unsupervised Noise Monitor for Gravitational-Wave Detectors

Marco Cavaglià

Abstract The anticipated high volume of gravitational-wave observations in the near future will require the development of reliable, unsupervised techniques for data quality assessment and signal detection and interpretation. We present a simple noise monitoring pipeline for gravitational-wave detectors that uses self-similarity analysis and an unsupervised machine learning anomaly detection algorithm. The approach may be used in real time to detect non-astrophysical noise transients at different time scales, as well as to identify periods of noise non-stationarity. We demonstrate how it works with two examples of data collected by one of the LIGO interferometers during the third observation run of the LIGO, Virgo, and KAGRA collaborations.

2.1 Gravitational-Wave Astrophysics in the Next Decade

LIGO [1], Virgo [2], and KAGRA [3] (LVK) scientists are gathering a record number of GW signals from various astronomical compact binary coalescence (CBC) sources [4]. Improved detector sensitivities will add thousands more gravitational wave (GW) detections to current catalogs in the coming years, expanding our understanding of the universe [5]. Further down the road, proposed 3G detectors [6, 7] will reveal hundreds of thousands of CBC signals, many of which will originate from multi-messenger astronomical sources [8].

These discoveries will provide GW scientists with the long-sought data to unravel some of the yet unsolved mysteries surrounding the GW sky; the origin of the observed black hole population [9], the nature of "low-mass gap" compact objects [10], the equation of state of neutron-rich matter at high densities and low temperatures [11], and of course, the many "unknown unknowns."

M. Cavaglià (✉)
Institute of Multi-messenger Astrophysics and Cosmology, Missouri University of Science and Technology, Rolla, MO, USA
e-mail: cavagliam@mst.edu

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_2

At the same time, the sheer number of expected detections will lead to challenges in collecting, validating, and interpreting their signals. To extract the most physics from GW data, researchers will need to automate data quality techniques as well as search and parameter estimation pipelines to the greatest extent feasible in the coming years.

The demand for automation is not limited to GW science. Automation of analytical workflows is required in all (scientific and non-scientific) efforts that generate or rely on large amounts of data. This conclusion has resulted in the rapid development of machine learning (ML) algorithms [12] and artificial intelligence [13], which are now widely used in today's society. ML research has historically been driven by industry applications. However, the emergence of large-scale, complex experiments such as GW detectors, high-energy colliders [14, 15], cosmic ray detectors [16–18], and electromagnetic (EM) telescopes [19–21] has led to an explosion in the use of ML methods in science. GW astrophysics is not immune to this [22]. While ML cannot be the only answer to the future GW detection rate crunch (streamlining analysis workflows and improved use of human and infrastructure resources will also be critical), it is widely acknowledged that ML will play an increasingly important role in GW research.

2.2 Monitoring the Noise of GW Detectors in Real Time

The analysis of instrumental and environmental noise sources is a critical component of the GW data analysis workflows for both existing and proposed GW detectors [23, 24]. A GW detector's noise floor is typically non-stationary and non-Gaussian [25]. Its sensitivity is limited by fundamental and technical noise sources, as well as transient and persistent noise artifacts caused by physical disturbances and non-linear couplings between detector subsystems and their environments [23].

Monitoring data stationarity, identifying and flagging noisy data, and potentially leveraging this information to remove noise artifacts with other methods [26–29] are all necessary requirements for GW detection. Excess noise in connection with a GW from a CBC source can drastically affect the signal's parameter estimation and sky localization. Noise (non-)stationarity may also affect background estimation and the significance of unmodeled GW cadidates. To complicate the matter, due to the anticipated growth of detected signals, all detector characterization and data quality tasks will need to be done in low-latency in the not too distant future.

In recent years, there have been great advances in the automation of noise analysis tasks that minimize human involvement and shorten the latency of the process (see, e.g., [22] for a review). Some of the techniques that are currently in use include signal processing tools such as omicron [30], machine learning (ML) algorithms such as iDQ [31], automated checks of lock status and noise stationarity, and monitors to determine physical couplings between the detectors and their environments. However, many of these processes still require human intervention for final validation.

Typically, the ML algorithms used in the GW workflow, such as neural networks, are trained on pre-built data sets. Although these supervised algorithms have proven useful in a wide range of applications [22], their training and testing paradigms may be inadequate in non-stationary and real-time settings. As the detector's power spectral density varies over short and long timescales (either by design or due to accidental factors), reliable predictions from supervised ML algorithms may require frequent re-training.

In this context, developing new unsupervised methods to monitor noise variations in the detectors is especially relevant. Along these lines, there have been a few interesting studies of ML anomaly detection (ML-AD) algorithms [32–34]. ML-AD may be the key to developing automated, unsupervised systems for noise monitoring.

The ML-AD paradigm is based on the idea of allowing the algorithm to learn the noise as it evolves and report what is anomalous without any prior training. Generative Adversarial Network (GAN) models and Temporal Outlier Factor (TOF) algorithms appear to be viable machine learning approaches for developing an unsupervised noise monitor. Recently, there has been exploratory research into GAN models for anomaly detection on time-frequency maps, detection of unmodeled GW signals [33], and development of glitch template banks [35, 36]. TOF has been used to the detection of noise transients, with promising results [34].

Another important aspect of designing an effective ML algorithm is the ability to build effective, higher-level features from raw data [37]. Rapid feature generation is one of the challenges that ML algorithms may confront within the present GW detector software infrastructure. For example, ML image-based noise classification in low latency may be inefficient because to the time necessary to construct time-frequency maps, particularly when hundreds of auxiliary data channels must be processed. Self-similarity (SSA) techniques represent a potential new approach for rapid feature generation.

Over the years, SSA has been applied in a variety of fields [38–45]. SSA arises naturally in nonlinear dynamical systems and may be utilized to estimate the complexity of a data set [46]. This property makes it ideal for characterizing physical devices with non-linear couplings that generate a large amount of data, such as GW detectors.

GW detectors produce data in the form of a multidimensional series of real-valued, discretely sampled measurements. In this scenario, the value of the SSA (fractal) dimension is related to the data's frequency content [47]. Therefore, the fractal dimension varies as the GW detector's noise changes over time.

The fractal dimension of the detector data stream may be a useful metric for determining the instrument's state, measuring data stationarity, and detecting non-astrophysical excess noise in the low-latency software architecture [48].

Because SSA detects noise fluctuations on both short and long timescales, it may be used in a number of investigations. Its output may then be passed to traditional or ML-based methods to characterize transient noise, monitor detector stationarity, and identify stretches of low-quality data. In addition to traditional methodologies, the SSA measure may be utilized as input for coherence methods to reveal non-linear couplings in detector subsystems [49]. In the context of GW interferometry, SSA

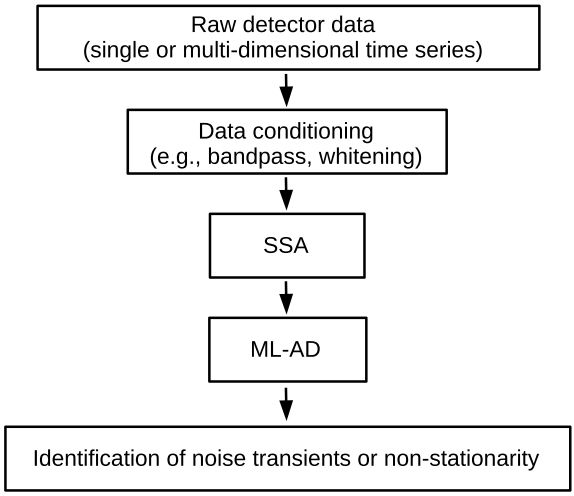


Fig. 2.1 Workflow of the proposed unsupervised SSA+ML approach for noise monitoring in a GW detector. Raw detector data is first conditioned before it is fed into the SSA algorithm, which generates a set of features that describe the variability of the noise. An unsupervised machine learning method is then used to identify anomalous fluctuations in the SSA features. The remaining part of this article describes a basic implementation of this method

has been used to characterize the time evolution of Virgo and KAGRA seismometer data [50–53]. A more recent example can be found in Ref. [54].

In this note, we propose integrating ML-AD with SSA to provide an unsupervised tool for monitoring noise stationarity in low-latency. Figure 2.1 depicts a process for the approach. We demonstrate this concept with a basic "out-of-the-box" implementation.

2.3 Self-Similarity as a (Rapid) Feature Generator

SSA sets, or fractals, are sets that have a non-integer dimension less than that of their Euclidean covering space [46]. In the case of one-parameter valued data sets, such as time series of amplitude $A(t)$, the covering space is $\mathrm{R}^2 \equiv \{t\} \otimes \{A(t)\}$ and the dimension of the set is $1 \leq D_F < 2$. For example, a smooth curve $C(t; t \in [0, T])$ has $D_F = 1$ whereas the dimension of an (infinitely dense) random series $R(t; t \in [0, T])$ is $D_F \to 2$ because it covers the whole two dimensional Euclidean area $T \times R$.

Discretely-sampled physical measurements do not strictly define a fractal set and allow only for an approximate measure of D_F. In such a case, different definitions of D_F lead to different values. As a consequence, no physical measurement allows for a unique definition of its fractal dimensionality. However, typically, what is interesting is the variation of D_F across the set. It is this variation that encodes all the relevant

information for the characterization of a physical data set, such as the noise of a GW detector. More details about the subtleties of defining D_F for a discretely-sampled data set can be found in Ref. [48].

The variation (VAR) algorithm [55–57] is an efficient method for estimating the SSA dimension of a one-parameter time series with N (equally spaced) samples A_i. The VAR estimator is calculated the functions

$$F_k \equiv \langle F_{k,j} \rangle_j = \left\langle \left| \max[A_{j-k} \dots A_{j+k}] - \min[A_{j-k} \dots A_{j+k}] \right| \right\rangle_j , \tag{2.1}$$

where $k = 1, 2, \dots N/2 - 1$, $j = [k, k+1, \dots N-k+1)$, and then calculating $D_F = 2 - S$, where S is the slope of the log-log $\ln(F_k)(\ln(k))$ curve.

Tests on white and Brownian noise, as well as known fractal sets, reveal that the VAR estimator is accurate enough to be used for time series with $\sim 10^3$ or more samples, with average errors of a few percent [48]. In our implementation of the algorithm, the fractal dimension of white noise is typically $D_F \sim 1.8$, which is roughly 10% lower than the theoretical value of $D_F = 2$.

Figure 2.2 shows the variation of the fractal dimension during two ten-minute spans of LIGO-Livingston public data collected during the LVK third observation run. The fractal dimension is calculated using a sped-up variant of the VAR approach, which allows D_F to be computed in real time (see Ref. [48] for details).

The variation in D_F encodes the change in detector noise over time. The left panel shows the fractal dimension calculated from the interferometer's raw strain output. For the first five minutes, the value of D_F remains roughly constant, with the majority of values falling between $D_F = 1$ (linear noise) and $D_F = 1.2$. The dimension then rapidly transitions to values around ~ 1.8, indicating that the detector's background

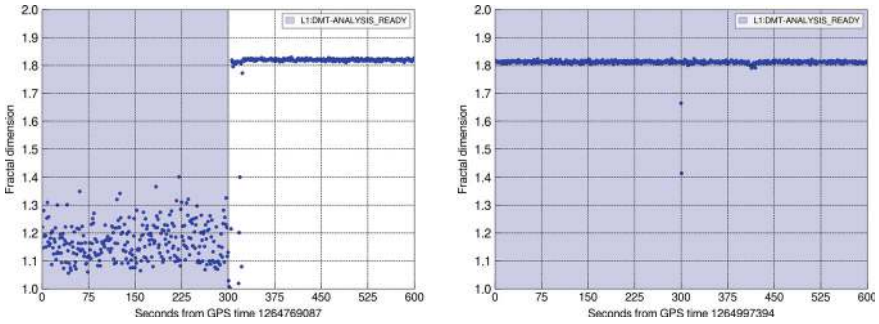


Fig. 2.2 Left panel: fractal dimension for a ten-minute period of interferometer raw output data. The interferometer is in observing mode (`L1:DMT-ANALYSIS_READY`) for the first five minutes before losing lock. Right panel: fractal dimension for a ten-minute period of whitened data. The interferometer is in observing mode throughout the period, however a 2-second glitch appears after 300 s into the plot. The data for both periods is from the LIGO-Livingston detector and was collected during the third LVK observing run. The data is sampled at 16,384 Hz. The fractal dimension is calculated using the VAR estimator mentioned in the text. Each point represents D_F for a second of data

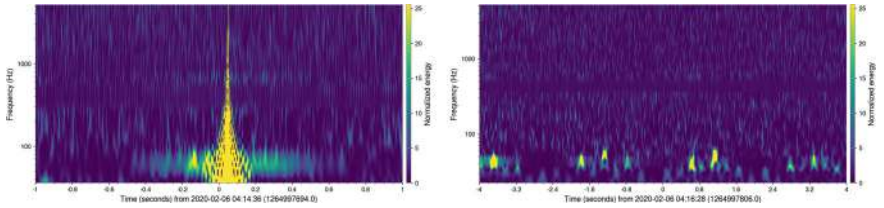


Fig. 2.3 Left panel: Q-transform [58] plot centered on the stronger glitch from the right plot of Fig. 2.2. The Q-scan indicates the presence of a loud overflow glitch in the interferometer. Right panel: Q-scan around the time of the weaker D_F variation, beginning at approximately 410 s into the data stretch. In this case, as well, the variation of the fractal dimension is caused by noise transients, specifically scattered light

noise has changed. A follow-up investigation reveals that at this point in time, the interferometer arm cavities lose resonance (lock loss), and the detector drops out of observing mode. The right panel depicts the variation in D_F calculated on whitened strain data. As predicted, the fractal dimension is stable around $\sim$1.8, which is the expected value for white noise in the VAR approximation. Five minutes into the stretch, the value of D_F drops between $\sim$1.66 and 1.41 for two seconds. A drop in the fractal dimension indicates the presence of a short-lived noise transient. A follow-up investigation reveals that, in this instance, the interferometer retains lock, but its binary neutron star inspiral range decreases by a factor of around 10 to roughly 10 Mpc. At around 410 s into the data stretch, there is a weaker noise transient that lasts a few seconds.

Figure 2.3 shows Q-transform [58] plots ("Q-scans") centered at around the times of the anomalous fractal dimension points from the right plot of Fig. 2.2. The Q-scan in the left panel corresponds to the time of the strong glitch with D_F ranging from 1.41 to 1.66. It indicates the presence of a significant noise transient, possibly caused by a sensor overflow, similar to the transient that overlapped with the GW170817 binary neutron star signal [59]. The Q-scan in the right panel corresponds to the time of the weaker glitch, which occurs $\sim$410 s after the start of the data stretch. Also in this case, the Q-scan reveals the occurrence of noise transients caused by scattered light in the interferometer optical systems. This noise transient is longer lived, lasting several tens of seconds. The excess noise in its loudest portion is strong enough to trigger a noticeable variation of the fractal dimension.

2.4 Machine-Based Anomaly Detection for Noise Monitoring

SSA can detect excess non-astrophysical noise and measure noise non-stationarity in real time. ML-AD can automate this procedure. Loud noise transients, like the one in the center of the data stretch in Fig. 2.2, are easily detectable. A simple

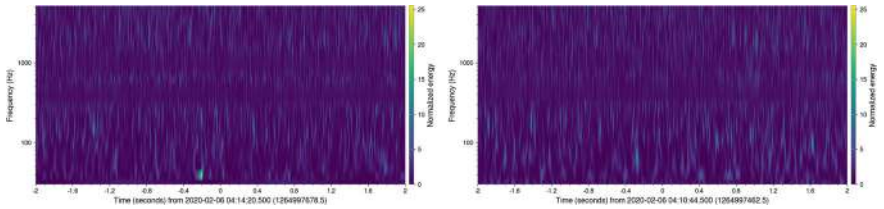


Fig. 2.4 Additional anomalous times identified by the LOF algorithm in the data stretch from the right panel of Fig. 2.2. The Q-scan in the left panel shows a glitch at around 40 Hz. The Q-scan in the right panel does not show any evident excess power. In this case, either the noise transient is at lower frequency or the identification is a false positive

cut-off of $D_F < 1.7$ can reveal loud glitches in whitened time series. However, weaker noise transients are harder to detect. Similarly, quantifying noise non-stationarity on unwhitened data needs a more sophisticated method for identifying anomalous periods in an unsupervised setting. Here, we will show how to develop a rudimentary ML-AD technique for detecting (some of the) weaker glitches in the D_F data stream.

In our basic application, we use the sklearn [60] implementation of the unsupervised Local Outlier Factor (LOF) detection method. The LOF approach identifies outliers in the fractal dimension series by comparing the local density of D_F values to the local densities of adjacent data points. This method is thought to be especially useful for fractal dimension series produced from whitened data, as D_F remains constant throughout clean periods.

We run the sklearn LOF algorithm on a rolling 60-second window. We first standardize D_F by subtracting its average and dividing by its standard deviation, and then pass the standardized data to the LOF algorithm. For the purpose of simplicity and to demonstrate the process, we do not tune any hyperparameters and merely set the number of neighbors to 20 and data contamination to 0.01. All other LOF parameters are set as the default values for the sklearn implementation. Even without hyperparameter tuning, we are able to identify the scattered light noise transient in the data, as well as two additional anomalous points. Figure 2.4 shows the Q-scans at the times of these anomalies. The first of these two anomalous times has a glitch. The second Q-scan does not reveal any obvious transient noise. This might be a false positive, the result of a bad hyperparameter choice. More research is clearly needed, but the fact that a basic, off-the-shelf ML-AD method already yields substantial results is quite promising.

2.5 Conclusions

The identification of noise transients and noise non-stationarity is critical for enhancing the quality of GW data and extracting astrophysical information from detectors. Over time, the application of machine learning has helped speed up and optimize

this process. Nevertheless, integrating supervised ML techniques into the current GW detector software framework can often be challenging. Furthermore, because to the improved sensitivity of future detectors, the development of unsupervised machine learning algorithms is particularly desirable. In this context, rapid feature creation for noise characterization, as well as ML-AD methods, might be quite useful. In this note, we have introduced a simple method for doing so that combines an SSA metric with a basic LOF ML-AD algorithm. While further investigation is definitely necessary, we hope that our work may start a discussion and pave the path for additional in-depth studies along this line of research.

Acknowledgements This work was partially supported by the U.S. National Science Foundation awards PHY-2011334 and PHY-2308693 is based upon work supported by NSF's LIGO Laboratory which is a major facility fully funded by the National Science Foundation. The author is grateful for computational resources provided by the LIGO Laboratory and supported by the U.S. National Science Foundation Grants PHY-0757058 and PHY-0823459, as well as resources from the Gravitational Wave Open Science Center, a service of the LIGO Laboratory, the LIGO Scientific Collaboration and the Virgo Collaboration. Part of this research has made use of data, software and web tools obtained from the Gravitational Wave Open Science Center [61] and publicly available at https://www.gw-openscience.org.

LIGO was constructed and is operated by the California Institute of Technology and Massachusetts Institute of Technology with funding from the U.S. National Science Foundation under grant PHY-0757058. Virgo is funded by the French Centre National de la Recherche Scientifique (CNRS), the Italian Istituto Nazionale di Fisica Nucleare (INFN) and the Dutch Nikhef, with contributions by Polish and Hungarian institutes. The author would like to thank the many colleagues of the LIGO Scientific Collaboration and the Virgo Collaboration who have provided invaluable help over the years and useful comments about this work.

Software for this analysis is written in Python 3.x [62] (https://www.python.org/) and uses standard Open Source libraries and community-contributed modules from the Python Package Index (PyPI) repository [63] including numpy, scipy, gwpy, h5py, pandas, matplotlib, nds2utils, numba [64], and sklearn [60]. Q-transform [58] plots have been generated with LIGO DV-Web [65].

This manuscript has been assigned LIGO Document Control Center number LIGO-P2400275.

References

1. Aasi, J., et al.: [LIGO Scientific Collaboration]. Advanced LIGO. Class. Quant. Grav. **32**, 074001 (2015). https://doi.org/10.1088/0264-9381/32/7/074001. [arXiv:1411.4547 [gr-qc]]
2. Acernese, F., et al.: [Virgo Collaboration]. The advanced Virgo detector. J. Phys. Conf. Ser. **610**(1), 012014 (2015). https://doi.org/10.1088/1742-6596/610/1/012014
3. Abe, H., et al.: [KAGRA]. Galaxies **10**(3), 63 (2022). https://doi.org/10.3390/galaxies10030063
4. LIGO, Virgo, and KAGRA Collaborations. Gravitational-Wave Candidate Event Database (GraceDB). https://gracedb.ligo.org/
5. Abbott, B.P., et al.: KAGRA, LIGO scientific and Virgo. Living Rev. Relativ. **19**, 1 (2016). https://doi.org/10.1007/s41114-020-00026-9. [arXiv:1304.0670 [gr-qc]
6. Cosmic Explorer. https://cosmicexplorer.org/index.html
7. Einstein Telescope. https://www.et-gw.eu/
8. Borhanian, S., Sathyaprakash, B.S.: [arXiv:2202.11048 [gr-qc]]

9. Abbott, R., et al.: [LIGO Scientific, VIRGO and KAGRA]. [arXiv:2111.03634 [astro-ph.HE]]
10. Abac, A.G., et al.: [LIGO Scientific, VIRGO and KAGRA]. [arXiv:2404.04248 [astro-ph.HE]]
11. Abbott, B.P., et al.: [LIGO Scientific and Virgo]. Phys. Rev. Lett. **121**(16), 161101 (2018). https://doi.org/10.1103/PhysRevLett.121.161101 [arXiv:1805.11581 [gr-qc]]
12. Mitchell, T.M.: "Machine Learning," McGraw-Hill International Editions Computer Science Series, McGraw-Hill; 1st edition (January 1, 1997). ISBN-10:0071154671; ISBN-13:978-0071154673
13. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach, Pearson, 3rd edition (January 1, 2015). ISBN-10:9789332543515; ISBN-13:978-9332543515
14. Wright, A., Webb, R.: Nature **448**, 269 (2007). https://doi.org/10.1038/448269a
15. Gray, H.M.: Rev. Phys. **6**, 100053 (2021). https://doi.org/10.1016/j.revip.2021.100053
16. Gaisser, T., Halzen, F.: Annu. Rev. Nucl. Part. Sci. **64**(1), 101–123m (2014). https://doi.org/10.1146/annurev-nucl-102313-025321
17. The Pierre Auger Collaboration, "Science Reviews - from the end of the world (Argentina)," Vol.1, No.4, pp. 8–33 (2020)
18. Zornoza, J.D., Zuniga, J.: [arXiv:1209.6480 [hep-ex]]
19. D. Scott Ltd for SKA Organisation, "Advancing Astrophysics with the Square Kilometre Array," Vol.1-2 (2015) ISBN: 978-1-909204-70-6, https://www.skatelescope.org/
20. Janssen, G., Hobbs, G., McLaughlin, M., Bassa, C., Deller, A.T., Kramer, M., Lee, K., Mingarelli, C., Rosado, P., Sanidas, S., Sesana, A., Shao, L., Stairs, I.H., Stappers, B.W., Verbiest, J.P.W.: PoS **AASKA14**, 037 (2015). https://doi.org/10.22323/1.215.0037. [arXiv:1501.00127 [astro-ph.IM]]
21. Gardner, J.P., Mather, J.C., Clampin, M., Doyon, R., Greenhouse, M.A., Hammel, H.B., Hutchings, J.B., Jakobsen, P., Lilly, S.J., Long, K.S., Lunine, J.I., McCaughrean, M.J., Mountain, M., Nella, J., Rieke, G.H., Rieke, M.J., Rix, H.-W., Smith, E.P., Sonneborn, G., Stiavelli, M., Stockman, H.S., Windhorst, R.A., Wright, G.S.: Space Sci. Rev. **123**, 485 (2006). https://doi.org/10.1007/s11214-006-8315-7. [arXiv:astro-ph/0606175 [astro-ph]]
22. Cuoco, E., Powell, J., Cavaglià, M., Ackley, K., Bejger, M., Chatterjee, C., Coughlin, M., Coughlin, S., Easter, P., Essick, R., Gabbard, H., Gebhard, T., Ghosh, S., Haegel, L., Iess, A., Keitel, D., Marka, Z., Marka, S., Morawski, F., Nguyen, T., Ormiston, R., Puerrer, M., Razzano, M., Staats, K., Vajente, G., Williams, D.: Mach. Learn. Sci. Tech. **2**(1), 011002 (2021). https://doi.org/10.1088/2632-2153/abb93a. [arXiv:2005.03745 [astro-ph.HE]]
23. Davis, D., et al.: [LIGO]. Class. Quant. Grav. **38**(13), 135014 (2021). https://doi.org/10.1088/1361-6382/abfd85. [arXiv:2101.11673 [astro-ph.IM]]
24. Abbott, B.P., et al.: [LIGO Scientific and Virgo Collaborations]. Characterization of transient noise in advanced LIGO relevant to gravitational wave signal GW150914. Class. Quant. Grav. **33**(13), 134001 (2016). https://doi.org/10.1088/0264-9381/33/13/134001. [arXiv:1602.03844 [gr-qc]]
25. Abbott, B.P., et al.: [LIGO Scientific and Virgo]. Class. Quant. Grav. **37**(5), 055002 (2020). https://doi.org/10.1088/1361-6382/ab685e. [arXiv:1908.11170 [gr-qc]]
26. Yu, H., Adhikari, R.X.: Front. Artif. Intell. **5**, 811563 (2022). https://doi.org/10.3389/frai.2022.811563. [arXiv:2111.03295 [astro-ph.IM]
27. Vajente, G., Huang, Y., Isi, M., Driggers, J.C., Kissel, J.C., Szczepanczyk, M.J., Vitale, S.: Phys. Rev. D **101**(4), 042003. https://doi.org/10.1103/PhysRevD.101.042003. [arXiv:1911.09083 [gr-qc]]
28. Ormiston, R., Nguyen, T., Coughlin, M., Adhikari, R.X., Katsavounidis, E.: Phys. Rev. Res. **2**(3), 033066 (2020). https://doi.org/10.1103/PhysRevResearch.2.033066. [arXiv:2005.06534 [astro-ph.IM]]
29. Davis, D., Littenberg, T.B., Romero-Shaw, I.M., Millhouse, M., McIver, J., Di Renzo, F., Ashton, G.: [arXiv:2207.03429 [astro-ph.IM]]
30. Robinet, F., Arnaud, N., Leroy, N., Lundgren, A., Macleod, D., McIver, J.: SoftwareX **12**, 100620 (2020). https://doi.org/10.1016/j.softx.2020.100620. [arXiv:2007.11374 [astro-ph.IM]]

31. Essick, R., Godwin, P., Hanna, C., Blackburn, L., Katsavounidis, E.: [arXiv:2005.12761 [astro-ph.IM]]
32. Morawski, F., Bejger, M., Cuoco, E., Petre, L.: Mach. Learn. Sci. Tech. **2**(4), 045014 (2021). https://doi.org/10.1088/2632-2153/abf3d0. [arXiv:2103.07688 [astro-ph.IM]]
33. Boudart, V.: A convolutional neural network to distinguish glitches from minute-long gravitational wave transients. [arXiv:2210.04588 [gr-qc]]
34. Ding, J., Ng, R., McIver, J.: Class. Quant. Grav. **39**(13), 135011 (2022). https://doi.org/10.1088/1361-6382/ac7278. [arXiv:2111.09465 [gr-qc]]
35. Lopez, M., Boudart, V., Schmidt, S., Caudill, S.: Simulating transient noise bursts in LIGO with $\mathtt{gengli}$. [arXiv:2205.09204 [astro-ph.IM]]
36. Powell, J., Sun, L., Gereb, K., Lasky, P.D., Dollmann, M.: [arXiv:2207.00207 [astro-ph.IM]]
37. Zheng, A., Casari, A.: Feature engineering for machine learning: principles and techniques for data scientists. O'Reilly Media; 1st edition (April 14, 2018). ISBN-10:1491953241; ISBN-13:978-1491953242
38. Brambilla, E. (ed.): Fractal Analysis—Applications in Physics, Engineering and Technology. InTechOpen (June 14, 2017). ISBN-10:9535131915; ISBN-13:978-9535131915
39. Dekking, M., Lévy-Véhel, J., Lutton, E., Tricot, C. (eds.): Fractals: Theory and Applications in Engineering, 1st edn. Springer (1999). ISBN-10:1852331631; ISBN-13:978-1852331634
40. Brambilla, E. (ed.): Fractal Analysis—Applications in Health Sciences and Social Sciences. InTechOpen (2017). ISBN-10:953513213X; ISBN-13:978-9535132134
41. Nonnenmacher, T.F., Losa, G.A., Weibel, E.R.: Fractals in Biology and Medicine. Mathematics and Biosciences in Interaction. Birkhäuser, Basel (2013). ISBN: 9783034885010. https://books.google.com/books?id=hVr2BwAAQBAJ
42. Lévy-Véhel, J., Lutton, E.: Fractals in Engineering: New Trends in Theory and Applications. Springer, London (2005)
43. Encarnacao, J.L., Peitgen, H.O., Sakas, G., Englert, G.: Fractal Geometry and Computer Graphics. Beiträge zur Graphischen Datenverarbeitung. Springer, Berlin, Heidelberg (2012)
44. Brown, C.T., Thurmond Witschey, W.R., Liebovitch, L.S.: The broken past: fractals in archaeology. J. Archaeol. Method Theory **12**, 37–78 (2005)
45. Peters, E.E.: Fractal Market Analysis: Applying Chaos Theory to Investment and Economics. Wiley Finance, (1994). ISBN: 9780471585244. https://books.google.com/books?id=_bkoySKyc_cC
46. Mandelbrot, B.B.: Fractal Geometry of Nature, 2nd edn. Times Books (1982). ISBN: 978-0716711865
47. Higuchi, T.: Phys. D Nonlinear Phenom. **46**, 254–264 (1990). https://doi.org/10.1016/0167-2789(90)90039-R
48. Cavaglià, M.: Class. Quant. Grav. **39**(13), 135012 (2022). https://doi.org/10.1088/1361-6382/ac7325. [arXiv:2201.09984 [gr-qc]]
49. Vajente, G.: Brute force coherence and non stationary noise analysis. LIGO Document G1500230
50. Bianchi, S.: J. Open Source Softw. **5**, 1828 (2020). https://doi.org/10.21105/joss.01828
51. Longo, A., Bianchi, S., Plastino, W., Idźkowski, B., Suchiński, M., Bulik, T.: Pure Appl. Geophys. **177**, 2597–2603 (2020). https://doi.org/10.1007/s00024-019-02395-x
52. Longo, A., Bianchi, S., Plastino, W., Fiori, I., Fiorucci, D., Harms, J., Paoletti, F., Barsuglia, M., Falxa, M.: Pure Appl. Geophys. **177**, 3395–3406 (2020). https://doi.org/10.1007/s00024-020-02428-w
53. Longo, A., Bianchi, S., Plastino, W., Miyo, K., Yokozawa, T., Washimi, T., Araya, A.: Pure Appl. Geophys. **178**, 3461–3470 (2021). https://doi.org/10.1007/s00024-021-02810-2
54. Laguarta, P., van der Laag, R., Lopez, M., Dooney, T., Miller, A.L., Schmidt, S., Cavaglià, M., Caudill, S., Driessens, K., Karel, J., et al.: Class. Quant. Grav. **41**(5), 055004 (2024). https://doi.org/10.1088/1361-6382/ad1f26. [arXiv:2310.03453 [astro-ph.IM]]
55. Tricot, C., Quiniou, J.F., Wehbi, D., Roques-Carmes, C., Dubuc, B.: Evaluation de la dimension fractale d'un graphe (1988)

56. Dubuc, B., Zucker, S.W., Tricot, C., Quiniou, J.F., Wehbi, D.: Proc. R. Soc. Lond. Ser. Math. Phys. Sci. **425**(1868), 113–127 (1989). The Royal Society. http://www.jstor.org/stable/2398496
57. Dubuc, B., Dubuc, S.: J. Numer. Anal. **33**(2), 602–626 (2006). Society for Industrial and Applied Mathematics. https://doi.org/10.1137/0733032. https://epubs.siam.org/doi/10.1137/0733032
58. Chatterji, S., Blackburn, L., Martin, G., Katsavounidis, E.: Class. Quant. Grav. **21**, S1809–S1818 (2004). https://doi.org/10.1088/0264-9381/21/20/024. [arXiv:gr-qc/0412119 [gr-qc]]
59. Abbott, B.P., et al.: [LIGO Scientific and Virgo]. Phys. Rev. Lett. **119**(16), 161101 (2017). https://doi.org/10.1103/PhysRevLett.119.161101. [arXiv:1710.05832 [gr-qc]]
60. Pedregosa, F., et al.: J. Mach. Learn. Res. **12**, 2825–2830 (2011)
61. Gravitational Wave Open Science Center. https://doi.org/10.7935/pr1e-j706. https://www.gw-openscience.org/
62. Van Rossum, G., Drake, F.L.: Python 3 Reference Manual. CreateSpace, Scotts Valley, CA (2009). ISBN: 1441412697
63. Python Package Index—PyPI. Python Software Foundation. https://pypi.org/
64. Numba, an open source JIT compiler, version 0.54.1 (7 October, 2021). Python Software Foundation, Copyright 2012–2020, Anaconda, Inc. and others. https://numba.pydata.org/
65. Areeda, J.S., Smith, J.R., Lundgren, A.P., Maros, E., Macleod, D.M., Zweizig, J.: Astron. Comput. **18**, 27–34 (2017). https://doi.org/10.1016/j.ascom.2017.01.003. [arXiv:1611.01089 [astro-ph.IM]]

Chapter 3
Simulation of Transient Noise Bursts in Gravitational Wave Interferometers

Melissa Lopez, Stefano Schmidt, and Francesco di Renzo

Abstract Gravitational-wave (GW) interferometers encounter significant challenges from transient noise artefacts, known as glitches. These glitches impair the sensitivity and data quality, complicating the detection of GW signals. To enhance detection capabilities, better modelling and inclusion of glitches in large-scale studies are essential. In this chapter, we explore the application of Generative Adversarial Networks (GANs), a cutting-edge deep learning algorithm, to learn the distribution of blip glitches and generate artificial populations. By reconstructing glitches in the time domain, we provide a smooth input for the GAN, enabling the creation of approximately 10^3 glitches from Hanford and Livingston detectors in less than one second. The performance and quality of the generated glitches are assessed using several metrics. We also introduce gengli, a user-friendly open-source software package that includes practical examples of the trained network's usage. Furthermore, we demonstrate a practical application to improve the understanding of Gravity Spy's performance, one of the current state-of-the-art glitch classifiers. Future work aims to extend this methodology to other glitch classes, ultimately creating an open-source interface for mock data generation. This will enhance stress testing of search pipelines and increase confidence in GW signal detection.

M. Lopez (✉) · S. Schmidt
Institute for Gravitational and Subatomic Physics (GRASP), Utrecht University, Princetonplein 1, 3584 CC Utrecht, The Netherlands
e-mail: m.lopez@uu.nl

S. Schmidt
e-mail: s.schmidt@uu.nl

Nikhef, Science Park 105, 1098 XG Amsterdam, The Netherlands

F. di Renzo
Université Lyon, Université Claude Bernard Lyon 1, CNRS, IP2I Lyon/IN2P3, UMR 5822, 69622 Villeurbanne, France
e-mail: f.di-renzo@ip2i.in2p3.fr

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_3

3.1 Introduction

As the amplitude of gravitational waves (GW) is minuscule, and thanks to the development of technology, large detectors were constructed to measure this space-time deformation accurately. Since the real world is full of imperfections and due to the sensitivity of GW detectors, the main strain of the detector $h(t)$ has undesired noise contributions. Indeed, many textbooks treat the detector noise as Gaussian and stationary, but it is a poor approximation of its data. The understanding and unbiased modelling of the different sources of noise is fundamental to infer the significance of GW signals and their astrophysical properties, so these efforts are a significant portion of LIGO-Virgo-KAGRA collaboration's work, known as *detector characterization* tasks [1].

GW interferometers are complex experiments with many sub-systems, which couple to the main detector strain $h(t)$ causing transient non-astrophysical bursts of non-Gaussian noise, colloquially known as *glitches* [2, 3]. Glitches may be caused by the environment (e.g., earthquakes, wind, anthropogenic noise) or couplings with instruments (e.g., control systems, electronic components [4]), though in many cases their causes remain unknown [5]. They come in a large variety of time-frequency morphologies, have a typical duration of between sub-seconds and seconds, and a high rate of occurrence ($\sim$1 per minute during the first half of the third observing run [6]).

Glitches are problematic due to their large abundance and capability of hampering GW data analysis. They can reduce the amount of analyzable data increasing the noise floor, produce false positives in GW data, affect the estimation of the detector power spectral density and reduce candidate significance in searches for short- and long-lived GW signals [7–11]. Glitches can also bias astrophysical parameter estimation, making it difficult to determine which part of the signal corresponds to a glitch and which part to the actual GW event [12–14]. Additionally, glitches can impact line-cleaning procedures in GW searches, which rely on replacing disturbed frequency bins with artificially generated data, consistent with their neighbours [11, 15, 16]. If the surrounding data contains elevated noise floors, the efficacy of mitigation methods will be reduced.

Because of everything previously mentioned, there is a need for better modelling and inclusion of glitches in large-scale studies, such as stress testing pipelines. In this chapter, we delve into an exploration of the current state-of-the-art methods employed to generate synthetic populations of glitches utilizing advanced Machine Learning (ML) techniques. Additionally, we will describe the tools developed within the g2net action and engage in a discussion regarding the future of this field.

3.2 Background

3.2.1 Glitch Characterization

Glitch identification and characterization is a crucial first step towards their mitigation, but due to their overwhelming amount, their characterization by hand is unfeasible [17–19]. A promising option is then to construct ML algorithms for their identification. Most of the current approaches to glitch characterization with ML utilize supervised classification algorithms, where models learn to identify glitches through labelled data representations of GW strain data $h(t)$ [20–26]. In practice, glitches are visualized in time-frequency representations (2-dimensional input) known as Q-transform, a modification of the standard short-time Fourier transform parameterized by a quality factor Q [27–29]. However, this procedure presents several limitations. Firstly, generating labelled data is an expensive task, since ML methods need a lot of examples for training, and experts must vet the labelling procedure. Secondly, glitch classes are highly unbalanced, biasing the models towards the most common classes. Moreover, supervised learning needs fixed class definitions that are not exhaustive nor representative of all glitch morphologies, as there could be many possible subclasses to discover [22]. Furthermore, as GW detectors are improved, novel glitch morphologies could arise [30].

Despite these challenges, these methods have been instrumental in GW detector characterization and data analysis. The most well-known glitch classifier is `GravitySpy`, which combines supervised ML and citizen science to characterize glitches present in LIGO data according to their morphologies in GW strain data h(t) [20, 23]. The trained algorithm assigns glitches to a pre-defined class and gives a confidence score that it belongs to this class.

3.2.2 Modeling Glitches

As discussed in the previous subsection, glitches are commonly represented as images where the glitch is in the centre surrounded by detector noise. For illustration in Fig. 3.1 we present a glitch *(left)* and a binary black hole merger *(right)* surrounded by noise. Although there exists an option to create synthetic time-frequency populations, this approach presents limitations in terms of flexibility, given that the noise enveloping the glitch is unique to each observing run, i.e. we could not utilize background noise from the second observing run for third-generation analysis. Furthermore, having a fixed surrounding background implies a fixed loudness or signal-to-noise ratio (SNR). To gain flexibility for future applications, we can extract the glitches from their surrounding detector noise using wavelet modelling. In this way, the input to the ML algorithm will be 1-dimensional instead of 2-dimensional, but except for the dimensionality of the input, the inner workings of the network remain unchanged.

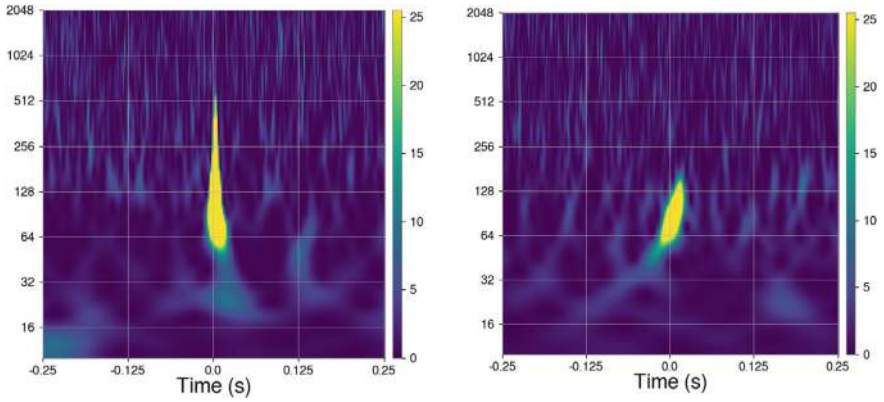


Fig. 3.1 *(Left)* Q-transform of a *Blip* glitch retrieved from `Gravity Spy` [20]. *(Right)* Q-transform of an event with total mass $106.6^{+13.5}_{-14.8} M_{\odot}$

3.3 Methodology

3.3.1 Generative Adversarial Networks

The current state-of-the-art algorithm to generate synthetic glitches is called Generative Adversarial Networks (GAN). GAN [31] are a class of generative algorithms in which two neural networks compete with each other to achieve realistic image generation. One network, known as the *generator*, generates new images from random noise, while the other, known as the *discriminator*, tries to discriminate the generated images from the real training data. The generator progressively learns which features of the real images should be mimicked to fool the discriminator and save them into the latent space, which can be understood as a compressed representation of the input data learnt by the generator. At the end of the training, new images are drawn by randomly taking a latent space vector and passing it to the generator, which has learned to translate it into a realistic image. Figure 3.2 shows an overview of the

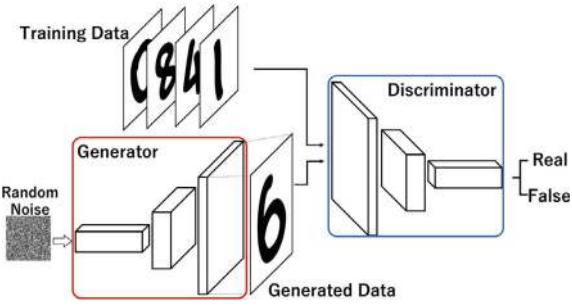


Fig. 3.2 Typical GAN architecture retrieved from [34]

original architecture of GAN for generating 2-dimensional data with convolutional layers, but depending on the application, the neurons used can be either fully connected or convolutional for 1- or 2-dimensional input. This early approach has been shown to work well under some hyperparameter configurations [32]. However, early GAN architecture [31] suffers from the significant problems of vanishing gradients and meaningless loss function [33].

3.3.2 Wasserstein Generative Adversarial Networks

Wasserstein GANs [35] (WGAN) was developed to address these issues previously mentioned by making use of the Earth's mover distance estimator, or Wasserstein-1 distance (W_1) [36], which computes the similarities between two distributions. W_1 is evaluated through the discriminator as the training progresses and increases monotonically while never saturating, providing a meaningful loss metric even for two disjoint distributions. Since W_1 is continuous and differentiable, it yields reliable gradients, allowing us to train the discriminator to optimality to obtain high-quality generations. This change of paradigm led Arjovsky et al. [35] to reformulate the optimization problem as:

$$\theta_{opt} = \arg\min_{\theta} W_1(P_x \| P_{\tilde{x}}), \tag{3.1}$$

where W_1 is evaluated between the real and generated distribution P_x and $P_{\tilde{x}}$, we rewrite Eq. 3.1 as,

$$\theta_{opt} = \arg\min_{\theta} \max_{\phi:\|D(x,\phi)\|_L \leq 1} L(\phi, \theta) \tag{3.2}$$

with the discriminator loss:

$$L(\phi, \theta) = -E_{x\sim P_x}\big[D(x, \phi)\big] + E_{\tilde{x}\sim P_{\tilde{x}}}\big[D(\tilde{x}, \phi)\big] \tag{3.3}$$

where D and G refer to the discriminator and the generator with parameters ϕ and θ, respectively. $E_{x\sim P_x}$ indicates that the expression has been averaged over a batch of real samples x, while $E_{\tilde{x}\sim P_{\tilde{x}}}$ has been averaged over a batch of generated samples $\tilde{x}$. The new condition over ϕ in expression Eq. 3.2 imposes a constraint on the discriminator D, which must be 1-Lipschitz continuous [35]. In practice, this can be achieved in two ways: clipping the weights of the discriminator beyond a specific value c [35], or adding a regularization term to the discriminator loss.

3.3.3 Gradienty Penalty and Consistency Term

Adding a regularization term to the discriminator loss (Eq. 3.3) to enforce the Lipschitz condition has been widely accepted. This regularization term is known as gradient penalty, $\mathcal{G}$, and its mathematical formulation is as follows:

$$L_{tot} = L(\phi, \theta) + \lambda\, \mathcal{G}(\phi) \quad \text{with } \mathcal{G}(\phi) = E_{\hat{x}\sim P_{\hat{x}}}\Big[\big(\|\nabla_x D(\hat{x}, \phi)\|_2 - 1\big)^2\Big], \quad (3.4)$$

where λ is the regularization parameter, $\|\cdot\|_2$ stands to the L2-norm and $\hat{x}$ is evaluated as $\hat{x} = \tilde{x}\, t + x\,(1-t)$, being t uniformly sampled $\sim [0, 1]$. This method has shown impressive applications such as [37], but it is not restricted to WGANs [38, 39].

Nonetheless, unlike weight clipping, gradient penalty cannot enforce the Lipschitz condition everywhere, particularly at the beginning of the training. This can prevent the generator from converging to the optimal solution. To overcome this obstacle, Wei et al. have proposed a second penalization term to add to the loss from Eq. 3.3, called consistency term, C, [40]. They applied their new constraint to two perturbed versions of the real samples x, introducing dropout layers into the discriminator architecture. This ultimately leads to two different estimates noted $D(x')$ and $D(x'')$. The consistency term is defined as,

$$\begin{aligned} C(\phi) = E_{x\sim P_x}\big[\max\big(0,\ & d(D(x', \phi), D(x'', \phi)) + \\ & 0.1\, d(D\_(x', \phi), D\_(x'', \phi)) - M'\big)\big], \end{aligned} \quad (3.5)$$

where d(.,.) is the L2 metric, $D_$stands for the second-to-last layer output of the discriminator, and M' is a constant value. Wei et al. found that controlling the second-to-last layer output helps improve the performance of the WGANs. Thus, the final discriminator loss is then [40]:

$$L_{\text{tot}} = L(\phi, \theta) + \lambda_1\, \mathcal{G}(\phi) + \lambda_2\, C(\phi)\,, \quad (3.6)$$

with λ_2 being the consistency parameter. This type of WGAN was called consistency term GAN (CT-GAN).

3.4 Synthetic Glitches in Time Series

As mentioned earlier, our focus lies in the generation of synthetic glitches within time series data. To achieve this objective, the approach presented in [41] involved adapting CT-GAN to train on glitch time series obtained through the wavelet modelling algorithm known as `BayesWave` [42]. Given the resource-intensive nature of glitch extraction, the authors opted to generate a time series for a specific class of glitches, specifically the *Blip* glitches identified by `Gravity Spy`, extracted

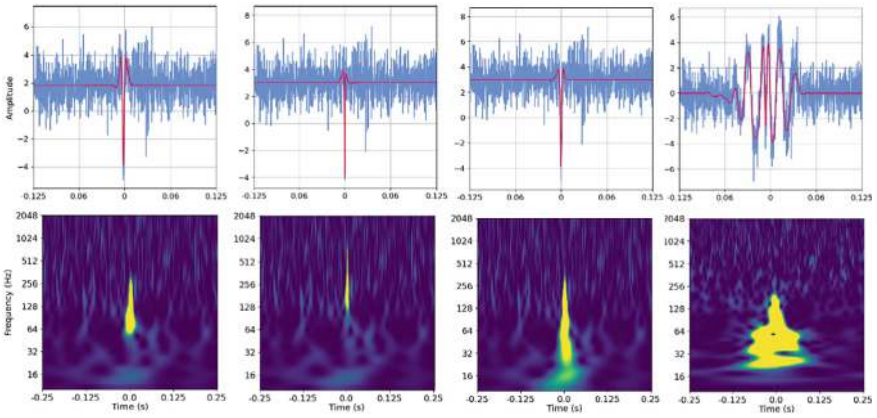


Fig. 3.3 Time series representation (top row) and Q-transform of synthetic glitches from Hanford

from Hanford and Livingston during the second observing run. These glitches are selected with a `Gravity Spy` confidence $\gtrsim$0.9 sampled at 4096 Hz, and whitened with `BayesLine` [43].

After the training of the CT-GAN, and given a 100-dimensional vector drawn from a normally distributed latent space (as it is common in other GAN-related works), and 10^3 *Blips* can be generated in $\approx$5 s for each detector. As an example, we present in Fig. 3.3 different artificial *Blips* from Livingston in the time domain, and for visualization, we also compute their Q-transform as in [20]. In the time-frequency representation, we can see that CT-GAN has been able to capture the distinct symmetric 'teardrop' of *Blips* in the expected frequency range [30, 250] Hz, but sometimes it generated glitches that have a completely different morphology. Indeed, as `Gravity Spy` has a certain degree of bias, it misclassifies some glitches that are present in the input data set. Therefore, even if most synthetic glitches have the characteristic 'teardrop' shape by visual inspection, it is necessary to perform a statistical test to assess the performance of CT-GAN (see [41] for details).

3.5 `Gengli`: Open-Source Glitch Generator

The authors in [41] extended their work in [44] and provided an open-source, user-friendly package for glitch generation known as `gengli`. Aside from generating whitened glitches, `gengli` provides further examples of applications, such as injecting glitches, i.e. adding glitches in the detector strain, and scaling glitches to a desired SNR, among others. Depending on our application, perhaps we would like to generate synthetic glitches similar to real ones. For this aim, to assess the degree of

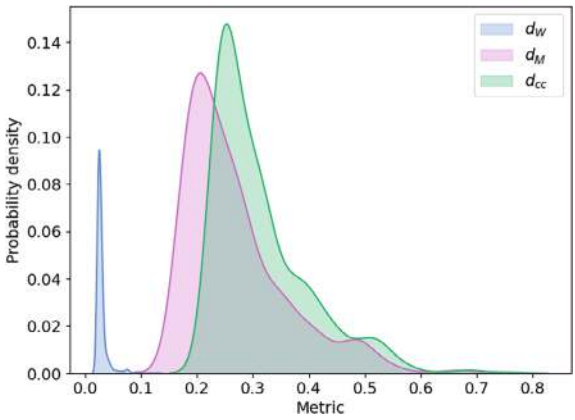


Fig. 3.4 Probability distribution of distances (d_W, d_M, d_{cc}) for a benchmark of 10^3

"similarity" with respect to the statistical distribution, `gengli` defines several distance metrics, namely Wasserstein distance (d_W), mismatch (d_M) and normalized cross-covariance (d_{cc}) [44].

To compare against a statistical distribution we can generate a benchmark set of N_b glitches (see Fig. 3.4 for $N_b = 1000$). For each of the $N_b(N_b - 1)/2$ pairs of glitches in the benchmark set, (d_W, d_M, d_{cc}) are computed. For each new glitch being generated, `gengli` also computes the set of average distances (d_W, d_M, d_{cc}) between the glitch and the benchmark set, and measures the set of percentiles (p_W, p_{mm}, d_{cc}) of each of the distances with respect to the benchmark distances. The triple score (p_W, p_{mm}, d_{cc}) allows us to filter glitches based on an anomaly score interval $[p_{min}, p_{max}]$. The code will output only glitches for which *all* the three anomaly scores lie within the given interval.

3.6 Applications: Studying `Gravity Spy`'s Performance

A possible application of `gengli` generations is to understand the confidence and classification labels of `Gravity Spy` for different SNRs. For this aim, we created a population of 10^3 glitches and we injected them in real whitened noise of the third observing run with SNR $\rho \in [1.0, 50]$. Note that this SNR is $> \rho_{Blips} \sim 20$, as we want to test the behaviour of `Gravity Spy` for louder signals. To avoid the influence of the background we use the same real whitened noise from Hanford in the GPS time range [1262540000, 1262540040] s[1] for every glitch realization, with sufficient data to avoid border effects in the whitening procedure.

In Fig. 3.5 we present the average `Gravity Spy` confidence c_{GS} for 10^3 artificial *Blip* glitches as a function of SNR ρ. We can observe that c_{GS} decreases as we increase ρ, with a minimum of $c_{GS} \approx 0.65$ at $\rho \sim 7.5$. As we will see in the next

[1] The data can be retrieved from https://gwosc.org/O3/.

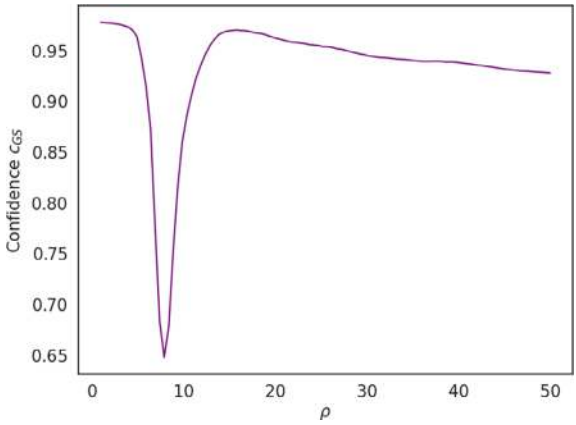


Fig. 3.5 Average `Gravity Spy` confidence c_{GS} for 10^3 artificial *Blip* glitches, generated with `gengli`, as a function of SNR ρ

graphics, this minimum is because `Gravity Spy` is confident that there is no glitch in the input data, known as *No_Glitch* class, until we increase the loudness of the glitches to $\rho \sim 7.5$. This behaviour is expected, since `Gravity Spy` is trained on Omicron's ≥ 7.5. Furthermore, `Gravity Spy` performance decreases slowly with the increase of ρ, which might be a symptom that `Gravity Spy` is biased towards loudness. Nonetheless, further investigation is needed.

In Fig. 3.6 we plot the classification labels, with maximum classification probability, for different ρ_{opt} of Hanford population. As we mentioned before, we can observe that at $\rho = 1.0$ every glitch is classified as *No_Glitch*, since the signal is below the bed of the noise. However, as we increase $\rho = 7.43$, `Gravity Spy` classifies some of the glitches as *Blips*, and some others as *Repeating_Blips*. At $\rho = 18.32$ most glitches are classified as *Blips*, with some misclassifications probably caused by anomalies in the data. Nevertheless, as we increase ρ, the number of *Blip* classifications degrades, increasing the number of glitches that are classified as *Koi_Fish* or *Blip_Low_Frequency*.

In Fig. 3.7 we present the joint and marginal distribution of c_{GS} as a function of the mismatch d_M (see [44] for details). Note that the marginal distributions are expressed as probability densities. For Fig. 3.8a, where $\rho = 1.0$, we can see how every glitch is classified as *No_Glitch* with $c_{GS} \approx 1.0$. As we increase $\rho = 7.43$ (see Fig. 3.8b) `Gravity Spy` starts to change its classification towards *Blip* class, which is dominant with high c_{GS} for $\rho = 18.43$ (see Fig. 3.8c). However, for $\rho = 45.55$ and as observed in Fig. 3.8d, while still most of the glitches are classified as *Blip*, we begin to see mis-classifications of *Blip_Low_Frequency* and *Koi_Fish* with $c_{GS} > 0.6$. As a final test, we selected a glitch that was anomalous according to `Gravity Spy` but not `gengli` (Fig. 3.8a), and a glitch that was anomalous for both, `Gravity Spy` and `gengli` (Fig. 3.8b).

On one hand, in Fig. 3.8a (middle panel) we can see that the glitch has a narrow peak, but according to `gengli`'s metrics, its morphology is close to the rest of the glitch population (low d_W, high d_M and d_k). In the top panel, we can see that

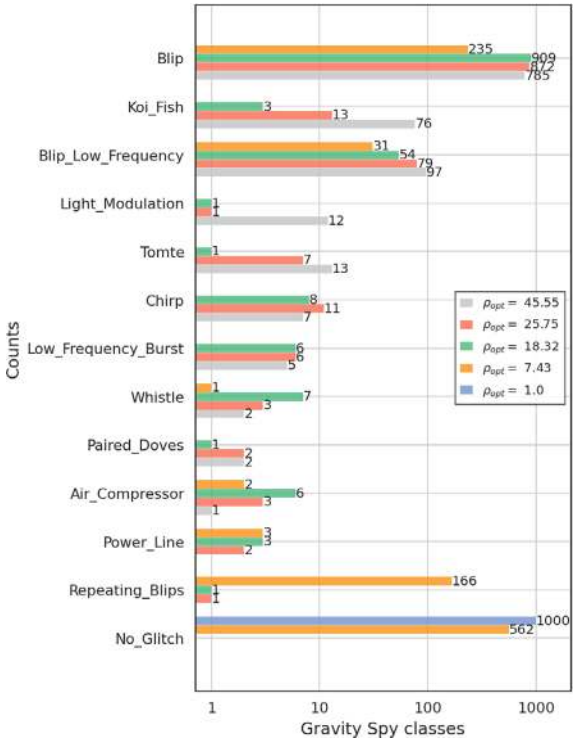


Fig. 3.6 `Gravity Spy` classes for 10^3 generated *Blips* from Hanford

`Gravity Spy` is unable to see this glitch until $\rho = 11.89$, wrongly classifying it as *Whistle* with $c_{GS} \approx 1.0$ for $15 \lesssim \rho \lesssim 40$. Afterwards, c_{GS} degrades with increasing ρ. In Fig. 3.8b (bottom panel) we can see that the glitch is faint even at $\rho = 18.32$.

On the other hand, in Fig. 3.8b (middle panel) we observe that the glitch does not have a standard *Blip* morphology (high d_W, low d_M and d_k). Since it is an anomalous glitch, `Gravity Spy` is confused about its class. In Fig. 3.8b (bottom panel) we can see that the glitch at $\rho = 18.32$ does not have the characteristic "tear-drop" shape of *Blip* glitches.

3.7 Discussion

As we have seen in the previous Sections, `gengli` is a powerful tool to further understand current glitch identification algorithms and help enhance searches of GW, as it was proposed in [45, 46]. Nonetheless, the main limitation of `gengli` is that it can produce a single class of glitches, and according to experts in the field, 23 classes exist. `Gravity Spy` data set is highly imbalance, having some classes $>10^3$ examples, and some others <100. This is an added challenge for GANs approaches,

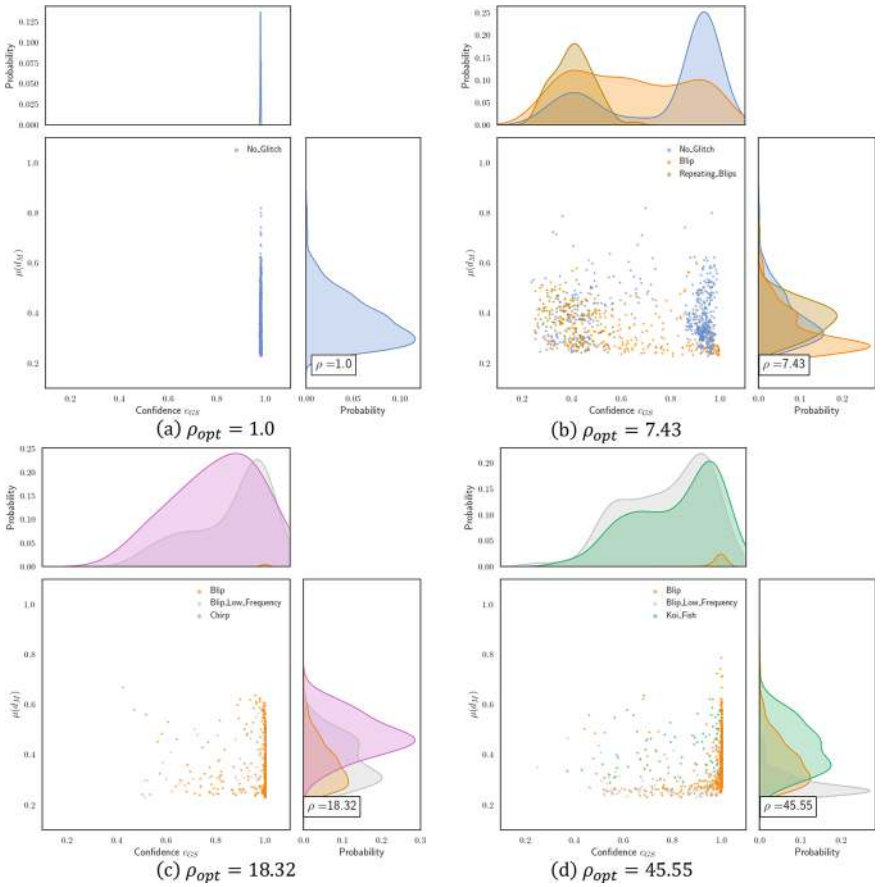
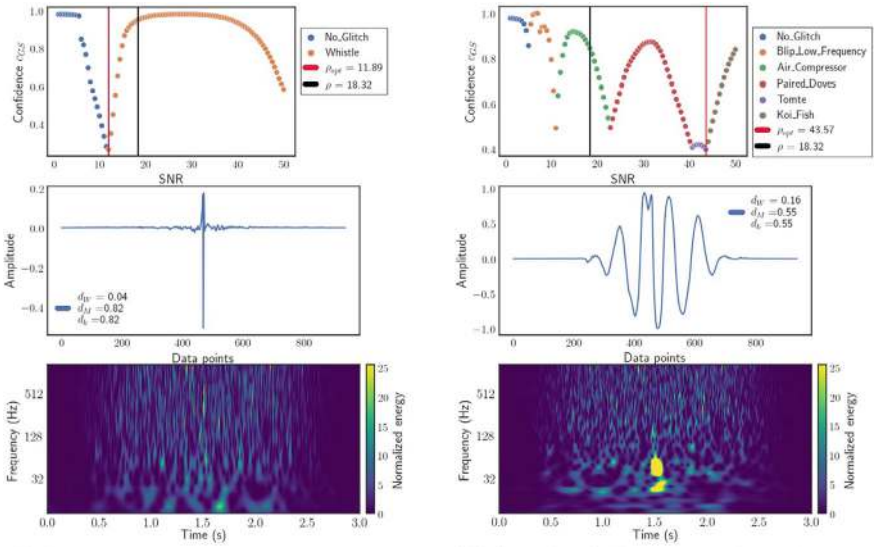


Fig. 3.7 Joint and marginal distribution of `Gravity Spy` confidence against mis-match d_M for different ρ_{opt}: *No_Glitch* (blue), *Blip* (orange), *Repeating_Blips* (brown), *Blip_Low_Frequency* (grey), *Chirp*(pink), *Koi_Fish* (green)

since they need a lot of data to learn the underlying distribution of the population. Furthermore, the pre-processing using BW is computationally intensive. Indeed, just to extract the populations of *Blip*, *Tomte* and *Koi_Fish* glitches for the extension of this work, we had to use ≈1.2 million CPU hours.

To overcome this issue, authors in [47, 48] propose to use TorchGAN and ProGAN, respectively, with Q-transforms instead of time series for all glitch classes. The main goal of these two approaches is to tackle the imbalanced problem of `Gravity Spy`. Thus, authors in [47] propose to over-sample glitch classes to 5, 000 examples, i.e. they sample randomly from the class distribution allowing examples to occur more than once. On the other hand, authors in [48] do not employ similar techniques as, due to the architecture of ProGAN, the underlying distribution of the data is learnt with a better generalization and less overfitting.



(a) Anomalous glitch according to Gravity Spy, but not gengli

(b) Anomalous glitch according to Gravity Spy, and gengli.

Fig. 3.8 Selected `gengli` glitches. *(Top panel):* confidence c_{GS} as a function of SNR. *(Middle panel):* Raw glitch signal output from `gengli`. *(Bottom panel):* spectrogram of glitch injected in real detector noise from Hanford at $\rho = 18.32$

While both algorithms proposed in [47, 48] show high performance when classifying artificially generated glitches with CNN, proving the added value of data-augmentation approaches, these methods are less flexible for mock data challenges as the glitch is not separated from the original background, a non-trivial task. Hence, authors in [49, 50] continue to think about novel GAN architectures in time series, also demonstrating that data-augmentation approaches will not only be relevant for current detectors but also for Einstein Telescope.

3.8 Conclusion and Future Prospects

In this chapter, we have discussed the generation of glitches with ML within the g2net action. In our work based on Refs. [41, 44], we have developed a methodology to generate artificial *Blip* glitches from real data using a ML algorithm known as GAN. To be able to generate these glitches, the input *Blips* need to be extracted from their surrounding detector noise, which is an expensive task.

Due to the instability of GAN algorithms, in this particular research, we trained a CT-GAN [40]. The network uses Wasserstein distance as a loss function, which allows it to train its discriminator to optimality. It is also heavily penalized to avoid training instabilities and to learn the underlying distribution of *Blips* accurately.

To assess the performance of CT-GAN, we have developed several statistical measurements to test the similarity between the input and the synthetic population. The results of these metrics indicate that the neural network was able to learn the underlying distribution of *Blip* glitches, despite the presence of some anomalous generations due to imperfections of the input data set.

In this proof-of-concept investigation, we have demonstrated that it is possible to isolate *Blip* glitches from their surrounding noise and learn their underlying distribution with an ML-based method in the time domain, providing several examples of its usage. This methodology allows us to generate better quality data, and it provides us with flexibility that would be challenging to achieve with Q-transforms. Furthermore, we also present our open-source package `gengli`: it provides an easy-to-use interface to the trained GAN output and has some additional features such as building a glitch population with or without anomalies, resampling, re-colouring and scaling, and further examples within the documentation. As a case study, we use `gengli` to understand the behaviour of `Gravity Spy`, showcasing its classification task as a function of the SNR with some possible bias towards loudness. However, further research is needed.

The long-term goal of this investigation is to learn other classes of glitches in the time domain. While extracting glitches from its background is an expensive task, it could be improved via specific wavelet design for glitches, among other possibilities.

References

1. Abbott, B.P., et al.: A guide to LIGO-Virgo detector noise and extraction of transient gravitational-wave signals. Class. Quantum Grav. (2020). https://doi.org/10.1088/1361-6382/ab685e
2. Blackburn, L., et al.: The LSC glitch group: monitoring noise transients during the fifth LIGO science run. Class. Quant. Grav. (2008). https://doi.org/10.1088/0264-9381/25/18/184004
3. Abbott, B.P., et al.: Characterization of transient noise in Advanced LIGO relevant to gravitational wave signal GW150914. Class. Quant. Grav. (2016). https://doi.org/10.1088/0264-9381/33/13/134001
4. Soni, S., et al.: Reducing scattered light in LIGO's third observing run. Class. Quant. Grav. (2020). https://doi.org/10.1088/1361-6382/abc906
5. Cabero, M., et al.: Blip glitches in advanced LIGO data. Class. Quant. Grav. (2019). https://doi.org/10.1088/1361-6382/ab2e14
6. Abbott, R., et al.: GWTC-2: compact binary coalescences observed by LIGO and Virgo during the first half of the third observing run. Phys. Rev. X (2021). https://doi.org/10.1103/PhysRevX.11.021053
7. Abbott, B.P., et al.: Effects of data quality vetoes on a search for compact binary coalescences in advanced LIGO's first observing run. Class. Quant. Grav. (2018). https://doi.org/10.1088/1361-6382/aaaafa
8. Abbott, R., et al.: All-sky search for continuous gravitational waves from isolated neutron stars using advanced LIGO and advanced Virgo O3 data. Phys. Rev. D (2022). https://doi.org/10.1103/PhysRevD.106.102008
9. Steltner, B., et al.: Deep Einstein@Home all-sky search for continuous gravitational waves in LIGO O3 public data. APJ (2023). https://doi.org/10.3847/1538-4357/acdad4

10. Abbott, R., et al.: Upper limits on the isotropic gravitational-wave background from advanced LIGO and advanced Virgo's third observing run. Phys. Rev. D (2021). https://doi.org/10.1103/PhysRevD.104.022004
11. Steltner, B., Papa, M.A., Eggenstein, H.B.: Identification and removal of non-Gaussian noise transients for gravitational-wave searches. Phys. Rev. D (2022). https://doi.org/10.1103/PhysRevD.105.022005
12. Pankow, C., et al.: Mitigation of the instrumental noise transient in gravitational-wave data surrounding GW170817. Phys. Rev. D (2018). https://doi.org/10.1103/PhysRevD.98.084016
13. Davis, D., et al.: Improving the sensitivity of advanced LIGO using noise subtraction. Class. Quantum Grav. (2019). https://doi.org/10.1088/1361-6382/ab2e14
14. Driggers, J.C., et al.: Improving astrophysical parameter estimation via offline noise subtraction for advanced LIGO (2019). https://doi.org/10.1103/PhysRevD.99.042001
15. Powell, J.: Parameter estimation and model selection of gravitational wave signals contaminated by transient detector noise glitches. Class. Quant. Grav. (2018). https://doi.org/10.1088/1361-6382/aacf18
16. Covas, B.P., et al.: Identification and mitigation of narrow spectral artifacts that degrade searches for persistent gravitational waves in the first two observing runs of advanced LIGO. Phys. Rev. D (2018). https://doi.org/10.1103/PhysRevD.97.082002
17. Davis, D., White, L.V., Saulson, P.R.: Utilizing aLIGO glitch classifications to validate gravitational-wave candidates. Class. Quant. Grav. (2020). https://doi.org/10.1088/1361-6382/ab91e6
18. Davis, D., et al.: LIGO detector characterization in the second and third observing runs. Class. Quant. Grav. (2021). https://doi.org/10.1088/1361-6382/abfd85
19. Powell, J., et al.: Classification methods for noise transients in advanced gravitational-wave detectors. Class. Quant. Grav. (2015). https://doi.org/10.1088/0264-9381/32/21/215012
20. Zevin, M., et al.: Gravity spy: integrating advanced LIGO detector characterization, machine learning, and citizen science. Class. Quant. Grav. (2017). https://doi.org/10.1088/1361-6382/aa5cea
21. George, D., Shen, H., Huerta, E.A.: Deep transfer learning: a new deep learning glitch classification method for advanced LIGO. Phys. Rev. D (2017). https://doi.org/10.1103/PhysRevD.97.101501
22. Bahaadini, S., et al.: Machine learning for gravity spy: glitch classification and dataset. Inf. Sci. (2018). https://doi.org/10.1016/j.ins.2018.02.068
23. Glanzer, J., et al.: Data quality up to the third observing run of advanced LIGO: gravity spy glitch classifications. Class. Quant. Grav. (2023). https://doi.org/10.1088/1361-6382/acb633
24. Ferreira, T.A., Costa, C.A.: Comparison between t-SNE and cosine similarity for LIGO glitches analysis. Class. Quant. Grav. (2022). https://doi.org/10.1088/1361-6382/ac813d
25. Razzano, M., et al.: GWitchHunters: machine learning and citizen science to improve the performance of gravitational wave detector. Nucl. Instrum. Methods Phys. Res. (2023). https://doi.org/10.1016/j.nima.2022.167959
26. Alvarez-Lopez, S., et al.: GSpyNetTree: a signal-vs-glitch classifier for gravitational-wave event candidates (2023). https://doi.org/10.48550/arXiv.2304.09977
27. Brown, J.C.: Calculation of a constant Q spectral transform. J. Acoust. Soc. Am. **89**, 425–434 (1991)
28. Robinet, F., et al.: Omicron: a tool to characterize transient noise in gravitational-wave detectors. SoftwareX (2020). https://doi.org/10.1016/j.softx.2020.100620
29. Jung, P.J., et al.: Sensing and vetoing loud transient noises for the gravitational-wave detection. J. Korean Phys. Soc. (2018). https://doi.org/10.3938/jkps.73.1197
30. Soni, S., et al.: Discovering features in gravitational-wave data through detector characterization, citizen science and machine learning. Class. Quant. Grav. (2021). https://doi.org/10.1088/1361-6382/ac1ccb
31. Goodfellow, I.J., et al.: Generative adversarial nets (2014)
32. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. In: 4th International Conference on Learning Representations, ICLR (2016). https://doi.org/10.48550/arXiv.1511.06434

33. Weng, L.: From GAN to WGAN (2019). https://doi.org/10.48550/arXiv.1904.08994
34. Harada, S., et al.: Biosignal generation and latent variable analysis with recurrent generative adversarial networks. IEEE Access (2019). https://doi.org/10.1109/ACCESS.2019.2934928
35. Arjovsky, M., et al.: Wasserstein generative adversarial networks. In: Proceedings of the 34th International Conference on Machine Learning (2017). https://doi.org/10.48550/arXiv.1701.07875
36. Kantorovich, L.V.: Mathematical methods of organizing and planning production. Manag. Sci. (1960). https://doi.org/10.1287/mnsc.6.4.366
37. Karras, T., et al.: Progressive growing of GANs for improved quality, stability, and variation. In: International Conference on Learning Representations (2018). https://doi.org/10.48550/arXiv.1710.10196
38. Salimans, T., et al.: Improved techniques for training GANs. In: Proceedings of the 30th International Conference on Neural Information Processing Systems (NIPS'16) (2016). https://doi.org/10.48550/arXiv.1606.03498
39. Mescheder, L., Geiger, A., Nowozin, S.: Which training methods for GANs do actually converge? In: Proceedings of the 35th International Conference on Machine Learning (ICML'18) (2018). https://doi.org/10.48550/arXiv.1801.04406.
40. Wei, X., et al.: Improving the improved training of Wasserstein GANs: a consistency term and its dual effect. In: International Conference on Learning Representations (2018). https://doi.org/10.48550/arXiv.1803.01541
41. Lopez, M., et al.: Simulating transient noise bursts in LIGO with generative adversarial networks. Phys. Rev. D (2022). https://doi.org/10.1103/PhysRevD.106.023027
42. Cornish, N.J., Littenberg, T.B.: BayesWave: Bayesian inference for gravitational wave bursts and instrument glitches. Class. Quant. Grav. (2015). https://doi.org/10.1088/0264-9381/32/13/135012
43. Littenberg, T.B., Cornish, N.J.: Bayesian inference for spectral estimation of gravitational wave detector noise. Phys. Rev. D (2015). https://doi.org/10.1103/PhysRevD.91.084034
44. Lopez, M., et al.: Simulating transient noise bursts in LIGO with gengli (2022). https://doi.org/10.48550/arXiv.2205.09204
45. Boudart, V., Fays, M.: Machine learning algorithm for minute-long burst searches. Phys. Rev. D (2022). https://doi.org/10.1103/PhysRevD.105.083007. eprint: arXiv:2201.08727
46. Meijer, Q., et al.: Gravitational-wave searches for cosmic string cusps in Einstein Telescope data using deep learning. Phys. Rev. D (2024). https://doi.org/10.1103/PhysRevD.109.022006
47. Powell, J., et al.: Generating transient noise artefacts in gravitational-wave detector data with generative adversarial networks. Class. Quant. Grav. (2023). https://doi.org/10.1088/1361-6382/acb038
48. Jianqi, Y., et al.: On improving the performance of glitch classification for gravitational wave detection by using Generative Adversarial Networks. Mon. Not. Roy. Astron. Soc. (2022). https://doi.org/10.1093/mnras/stac1996
49. Dooney, T., Bromuri, S., Curier, L.: DVGAN: Stabilize Wasserstein GAN training for time-domain Gravitational Wave physics. In: 2022 IEEE International Conference on Big Data (Big Data). IEEE Computer Society (2022). https://doi.org/10.1109/BigData55660.2022.10021080
50. Dooney, T., et al.: cDVGAN: one flexible model for multi-class gravitational wave signal and glitch generation (2024). https://doi.org/10.48550/arXiv.2401.16356

Chapter 4
Efficient ML Algorithms for Detecting Glitches and Data Patterns in LIGO Time Series

Elena-Simona Apostol and Ciprian-Octavian Truică

Abstract The field of Gravitational Wave research has exploded in recent years. Powerful laser interferometers like Advanced Laser Interferometer Gravitational Wave Observatory (LIGO) and Advanced Virgo are now listening for the universe's most violent events. While these technological marvels have significantly improved the precision of gravitational wave data collection, the data itself is not perfect. Noise can still creep in, potentially leading to misinterpretations. One problematic type of noise in Gravitational Wave data is the glitch. We define a glitch as a noise event that can either masquerade as a real signal from space or degrade the overall quality of the data. In this chapter, we present our current work for the task of detecting Gravitational Wave glitches using Machine Learning and Deep Learning models. We also establish a clear benchmark to compare their performance, highlighting the models that achieve the best results based on three key metrics: accuracy, precision, and recall.

4.1 Introduction

The detection and analysis of Gravitational Waves provided access to astrophysical discoveries that were not possible through other means. Several laser interferometers on Earth's surface (e.g., Advanced Laser Interferometer Gravitational Wave Observatory (LIGO) [1], Advanced Virgo [2]) are used to search gravitational signals in the frequency range from 10 Hz to 10 kHz. The gravitational signals that terrestrial interferometers search for are typically mixed with a lot of background noise that needs to be removed. Glitches, which are noises that mimic genuine astrophysical signals and/or affect the quality of the received signal data, are the most difficult to

E.-S. Apostol (✉) · C.-O. Truică
National University of Science and Technology Politehnica Bucharest,
060042 Bucharest, Romania
e-mail: elena.apostol@upb.ro

C.-O. Truică
e-mail: ciprian.truica@upb.ro

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_4

remove. Thus, the main issue with the received signals from the terrestrial interferometers is that the collected data may contain a high number of glitches. To address this, scientists have explored various detection algorithms. However, most current methods focus on specific types of signals as they are based on matched-filtering, making them susceptible to errors. On the other hand, Machine Learning and Deep Learning algorithms are more versatile and offer promising solutions for handling the complexity of gravitational wave data analysis.

In this chapter, we present our research on detecting Gravitational Wave glitches on an annotated corpus consisting of records from the first observing run of the Advanced LIGO detectors in Hanford and Livingston, collected between September and December 2015. This corpus is annotated using 22 classes for transient noises, defining our objective as a multi-class classification problem. To tackle this task, we designed and tested a variety of models, including classical Machine Learning, Deep Learning, and even combinations of these approaches (ensemble models). For each of these models, we first delve into the details of the algorithms used. Then, we analyze their performance on the LIGO dataset using several metrics like accuracy, precision, and recall. We also present two new architectures proposed by us, i.e., ShallowWaves and DeepWaves Ensembles [3]. In order to test all our 24 implemented Machine Learning/Deep Learning models, we divided the LIGO dataset into training and testing sets using the Stratified K-Folds technique. For the classical Machine Learning algorithms, we employed grid search to find the optimal settings that maximized their performance. In general, the accuracy of the classical models was good, while the performance of the deep learning models varied depending on the specific network architecture used. In our experiments, the DeepWaves Ensemble, which combines multiple deep learning models, achieved the highest overall accuracy, followed closely by the ShallowWaves Ensemble. The majority of these experiments were also presented in our original paper [3].

4.2 Problem Definition for the Detection Task

In this section, we outline the research problem by defining the Gravitation Wave glitch detection task. This task takes as input a stream of multi-variant time series data $t_i \in \mathbb{R}^n$ collected from a Gravitational Wave generator and annotated by experts. Each data point t_i in our time series input is characterized by a set of features, denoted by $t_i = \{\Sigma p^i_j\}$. Where p^i_j captures specific properties of the signal, like its frequency, how wide its range of frequencies is, the signal-to-noise ratio (SNR), and how long it lasts. This way of organizing the data allows us to analyze the properties of the signals and separate the true space signals from the noise.

For the annotated dataset, we consider a collection of categories $C = \{c_k\}$. These categories represent different types of glitches that might corrupt our data, like electrical interference from power lines or noise caused by ground movement. Each type of glitch has a distinct fingerprint in how its energy is distributed across different frequencies over time, as shown in this online research [4].

The main goal of the detection task is to build models that can classify data points into these categories. These models, called multi-class classifiers, will take a data point with various features (represented by a set of numbers in n-dimensional space) and assign it to the most likely glitch category ($c_k \in \mathbb{R}^n$ from set C). We want these models to be:

1. Efficient and easy to use on different systems (i.e., efficiently implemented and distributed);
2. Accurate even when we have a limited amount of data, especially when that data is not evenly distributed across the different glitch categories (i.e., good statistical performance on small to medium unbalanced datasets).

4.3 Detecting Glitches Using Classical Machine Learning Models

In this section, we present the proposed supervised classical Machine Learning algorithms used to detect anomalies, i.e., the glitches, in the annotated Gravitational Wave dataset. Classical Machine Learning models are simpler algorithms than Deep Learning models. They rely on statistical methods and mathematical models, such as linear regression, support vector machines, or decision trees.

K-Nearest Neighbor (KNN)

KNN is a lazy learning method as no model is learned from the training data. A learning process only occurs when a test example needs to be classified. Thus, it does not produce a model but is a simple method that determines the class of an example based on the labels of its neighbors. The algorithm uses a distance function that computes the distance from the test example to the examples in the training set. KNN works in three major steps:

1. **Distance Measure**. First, it uses a distance function to calculate the distance between each point in the dataset.
2. **Nearest Neighbors**. Then, it picks a predefined number of the closest data points (k-nearest neighbors).
3. **Majority Vote**. Finally, KNN looks at the categories of those nearest neighbors. Whichever category is most frequent among those neighbors becomes the predicted category for the new data point.

Gaussian Naïve Bayes (GNB)

GNB is a probabilistic classification method that uses the Bayes theorem to compute the probability for each new data point to belong to a category, i.e., $P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$. When dealing with numerical continuous data, GNB makes an assumption: it supposes that the values for each category follow a specific bell-shaped curve, i.e., a Gaussian distribution [5]. By analyzing these curves, GNB calculates the probability of a new data point belonging to each category.

Thus, given a continuous variable x, we employ the following steps:

1. Segment the data by class;
2. Compute the associated mean and variance of x for each class (e.g., for class c_k, we note μ_k and σ_k^2);
3. For some observation value v, we have $p\left(x = v \mid c_k\right) = \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{(v-\mu_k)^2}{2\sigma_k^2}}$.

Logistic Regression (LogReg)

Assuming a linear or nonlinear model of dependency, regression models are used to predict the value of a given continuous variable based on the values of other variables. Logistic regression analyzes the relationships between different factors (predictor variables) and uses those relationships to estimate the probability of an independent variable falling into a certain category given this set of predictors.

Given a set of independent variables $X = \{x_1, x_2, ..., x_n\}$ and a set of classes $C = \{c_1, c_2, ..., c_k\}$, the multinomial logistic regression model computes the probability of an element to belong to one of these classes as $p\left(C = c_j\right) = \frac{e^{X\beta^{(j)}}}{1+\sum_{j=1}^{k} e^{X\beta^{(j)}}}$.

Support Vector Machine (SVM)

SVM [6] is a supervised discriminative binary classifier that determines the best separation hyperplane to group the labeled points in a dataset. Thus, given a dataset with m-dimensional points $X = \{x_1, x_2, ..., x_n\}$ and two classes $C = \{-1, +1\}$, the algorithm tries to determine a function $f(x) =< w \cdot x > +b$ that separates the points into the two classes. The 'best' hyperplane is given $f(x) = 0$. The theory shows that the best plane is the one maximizing the so-called margin ($Margin = \frac{2}{||w||}$) (the minimum orthogonal distance between a positive and negative point from the training set), which means minimaxing $\frac{||w||^2}{2}$. This optimization problem is solvable by rewriting the above inequality using a Lagrangian formulation and then finding a solution using Karush-Kuhn-Tucker (KKT) conditions.

In many situations, there is no hyperplane for separation between positive and negative examples. In such cases, it is possible to map the training data points (examples) in another space, a higher-dimensional one. Here data points may be linearly separable. The mapping function gets examples (vectors) from the input space X and maps them in the so-called feature space F: $\Phi : X \rightarrow F$. In solving the optimization problem for finding the linear separation hyperplane in the new feature space F, all terms containing training examples are only of the form $\Phi(x_i) \cdot \Phi(x_j)$. By replacing this dot product with a function in both x_i and x_j the need for finding disappears. Such a function is called a kernel function: $K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$. For finding the separation hyperplane in F, we must only replace all dot products with the chosen kernel function and then proceed with the optimization problem like in the separable case.

SVM is a binary classifier, to solve multi-class problems there are two strategies: one versus one (ovo) and one versus the rest (ovr).

Decision Trees

A Decision Tree Classifier constructs a classification model in the form of a tree structure. It subdivides a dataset into smaller subsets while at the same time, a corresponding decision tree is incrementally developed. The decision tree will consist of decision nodes and leaf nodes. Decision nodes represent the attributes of the dataset, while the leaf nodes indicate the final classes assigned to the data. The branches of the tree represent conditions based on discrete values or intervals of the corresponding attributes, determining the path to the next decision nodes or leaf nodes.

ID3 [7] is an algorithm that constructs a decision tree classifier in a top-down manner choosing at each node the 'best' attribute for branching. The decision tree is built top-down from a root node and involves partitioning the data into subsets that contain instances with similar values (homogenous). ID3 utilizes entropy to compute the homogeneity of a sample.

Given a set of classes $C = \{c_1, c_2, ..., c_k\}$ and a labeled dataset $X = \{x_1, x_2, ..., x_n\}$ with each observation x_i containing m attributes $A = \{a_1, a, ..., a_m\}$, the entropy is $H(X) = \sum_{i=1}^{k} p(c_i) log_2 (p(c_i))$, with $p(c_i)$ the probability of class c_i in X dataset.

If attribute A_t having r distinct values is considered for branching, it will partition X in r disjoint sets, $X = \cup_{i=1}^{r} X_i$. The entropy of the split is given by $H(X|a_t) = \sum_{i=1}^{r} \frac{|X_i|}{|X|} H(X_i)$ with $\frac{|X_i|}{|X|}$ the proportion of elements in $|X_i|$ to the number of elements in the dataset $|X|$. Because the purity of the datasets is increasing, $H(X)$ is bigger than $H(X|a_t)$. The difference between them is called the information gain $IG(X, a_t) = H(X) - H(X|a_t)$. The 'best' attribute is determined by the highest gain.

C4.5 [8] is the improved version of ID3 that better handles numeric (continuous) attributes. It also performs post-pruning to deal with noisy data.

CART (Classification And Regression Tree) [9] works on the same principles as C4.5 and ID3. The main difference is in the function used for calculating the homogeneity of a sample (computing the split). CART uses the Gini Impurity: $GI(X) = 1 - \sum_{i=1}^{k} \left(\frac{p(c_i)}{|X|}\right)^2$. The Gini Impurity for a split is $GI(X|a_t) = \sum_{i=1}^{r} \frac{|X_i|}{|X|} GI(X_i)$. Unlike C4.5 where the maximum Information Gain is used to make the split, for CART the split is done for the minimum Gini Impurity.

4.4 Detecting Glitches Using Deep Learning Models

In this section, we present the Deep Learning models that we employed to solve the problem of detecting anomalies in the Gravitational Waves datasets. We present the different types of layers in our experiments, i.e., the simple perceptron, multilayer perceptron, recurrent, and convolutional neural network. In our experiments, we combined these layers to form more complex Deep Learning architectures.

Perceptron

The Perceptron is a basic neural network (NN) with just one layer – the output layer. It uses the logistic regression equations to process the input data and a threshold function, i.e., the perception activation function, is applied for classification. Depending on the desired outcome, the most used activation functions are:

- rectified linear unit (ReLU), leaky ReLU: appropriate for binary classification,
- hyperbolic tangent (tanh), sigmoid function: used in multi-classification when the output must be restricted to positive values.

Multi-layer Perceptron (MLP)

The MLP is a NN algorithm that solves classification problems by stacking many layers of perceptrons in a fully connected network. An MLP has an input layer (receives data), optional hidden layers (process information), and an output layer (gives the final answer). Each layer can have a different number of neurons and computes the outcome using the input from the layer before it, except for the input layer that has no layer before it. The number of hidden layers and neurons in each layer can be adjusted based on the problem that needs to be solved.

Each neuron follows the perceptron model and uses parameters, represented by a vector of weights and a bias. These are adjusted during computation and initialized at the beginning. One method to do this is to initialize the weights with random small values and the bias with zero. An activation function is used to compute the output. The function is the same for each neuron of a layer and, usually, non-linear in the middle layers and liner in the output layer.

Long Short-Term Memory Network (LSTM)

Recurrent neural networks (RNNs) are a special type of NN. Unlike traditional NNs that process individual pieces of information independently, RNNs can take into account the order or relationships between elements in a sequence. There are several variants of RNNs, e.g., Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Bidirectional RNN. In our experiments, we chose to implement the Long Short-Term Memory (LSTM) variant, as it manages to avoid two of the main issues of the other types of RNN: the vanishing and the exploding gradient.

LSTM [10] is an extension of the Recurrent Neural Network (RNN) that uses two state components for classification. The first component is a short-term memory that learns the short-term dependency between the prior and present states. The second component is a long-term memory, representing a cell's internal state that retains long-term dependency between its prior and current states.

LSTM uses three gates to preserve the long-term memory in the state:

1. input gate ($i \in \mathbb{R}^n$);
2. forget gate ($f \in \mathbb{R}^n$);
3. output gate ($o \in \mathbb{R}^n$).

For each element in the input sequence, each layer computes:

$$i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi})$$
$$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf})$$
$$g_t = \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg})$$
$$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho})$$
$$c_t = f_t * c_{t-1} + i_t * g_t$$
$$h_t = o_t * \tanh(c_t)$$

where

- h_t is the hidden state at time t,
- c_t is the cell state at time t,
- x_t is the input at time t,
- h_{t-1} is the hidden state of the layer at time $t-1$ or the initial hidden state at time 0,
- i_t, f_t, g_t, o_t are the input, forget, cell, and output gates,
- σ is the sigmoid function, and
- $*$ is the Hadamard product.

In a multilayer LSTM, the input $x_t^{(l)}$ of the l-th layer, with $l \geq 2$, is the hidden state $h_t^{(l-1)}$ of the previous layer multiplied by dropout $\delta_t^{(l-1)}$, where each $\delta_t^{(l-1)}$ is a Bernoulli random variable which is 0 with probability dropout, a given hyperparameter.

Convolutional Neural Network (CNN)

CNN is a type of NN mostly utilized in analyzing visual images and is modeled after the pattern of neuronal connections found in the visual cortex. One of the differences between regular NNs and CNNs is that a CNN explicitly assumes that the input is an image, which makes the preprocessing required much lower than for other classification algorithms [20]. A CNN consists of an input layer. an output layer, as well as several hidden layers, i.e., a convolutional, a pooling, and a fully connected layer.

The fully connected layer is identical to the hidden layer in regular NNs. The neurons in the fully connected layer have full connections to all activations in the previous layer, so their activations can be computed with matrix multiplication followed by a bias offset.

The convolution layer is the core building block in CNNs that does most of the computation. The parameters considered for this layer are the learnable filters. They extend through the whole depth of the input, even though they are spatially small in width and height. For text classification, the depth of the input is 1. A learnable filter can be seen as a window sliding across the input, and the only part of the input that is analyzed at a certain time is the one inside the window. The result of the analysis is the dot product between the entries in the filter and the input at the current position of the window.

The function of a pooling layer in a CNN is to progressively reduce the spatial size of the representation to lower the number of parameters and computation in

the network, contributing also in controlling overfitting. That is why it is common to insert a pooling layer between successive convolution layers. The pooling layer operates independently on every depth slice of the input and resizes it spatially, using the MAX operation.

Deep Belief Networks (DBN)

DBNs [11] are a class of feedforward deep neural networks that consist of multiple hidden layers with connections between the layers but not between units within each layer. DBNs are a sophisticated and powerful type of generative neural network that uses an unsupervised machine learning model to produce results. DBNs are essentially stacks of simpler structures called Restricted Boltzmann Machines (RBMs). Unlike traditional neural networks trained all at once, DBNs are trained in a step-by-step fashion. One RBM is trained at a time, uncovering elementary features in the data. Then, the next RBM builds upon the knowledge of the previous one, learning more intricate relationships.

4.5 Detecting Glitches Using Ensemble Models

In this section, we present the proposed Ensemble methods for detecting glitches in a stream of Gravitational Wave data.

Ensemble methods combine several classifiers to obtain a superior one. While combined classifiers share the same learning methodology, they differ in terms of training datasets and weight examples. Boosting and Bagging (Bootstrap Aggregation) are used in ensemble approaches.

Bootstrap uses statistical methods (e.g., mean and standard deviation) for estimating a quantity from a data sample. Given a sample of n values $X = \{x_1, x_2, ..., x_n\}$, it means $\underline{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$. If n is small, then the sample mean has an error. The estimate of the mean can be improved by using bootstrap. Bootstrap will create multiple random sub-samples of the initial sample with replacement (meaning the same value can be selected multiple times), calculate the mean for each subsample, and calculate the average of all the sub-samples means and use it as the estimated mean for the initial sample.

Bagging [12] is an ensemble solution designed to improve the stability and accuracy of machine learning. One major property of the Bagging approach is that it constructs a training set from the initial labeled data set by sampling with replacement. Thus, it reduces the variance and helps to avoid overfitting. Bagging consists of the following steps:

1. Starting with the original dataset, build n training datasets by sampling with replacement (bootstrap samples),
2. For each training dataset, build a classifier using the same learning algorithm,
3. The final classifier is obtained by combining the results of each classifier (by voting for example).

Boosting [13] works by building a sequence of weak classifiers and adding them to the structure of the final strong classifier. Weak learner (classifier) is a classification algorithm with a substantial error rate in which performance is not random. In other words, a weak learner has an accuracy only slightly better than using random guessing. The weak classifiers are weighted based on weak learners' accuracy. Also, data is reweighted after each weak classifier is built such as examples that are incorrectly classified to gain some extra weight. The result is that the next weak classifiers in the sequence focus more on the examples that previous weak classifiers missed.

Random Forests (RF)

RF [14, 15] is an ensemble classifier consisting of a set of decision trees. This technique combines multiple decision trees, where each tree makes a prediction, and the final output reflects the most frequent prediction from all the trees. RF offers an advantage over simply bagging decision trees. The issue with decision tree algorithms is their greedy nature. They choose the feature for splitting data points based on a strategy that minimizes error at that specific step. This can lead to highly similar structures across the trees in a bagged ensemble, resulting in correlated predictions. RF addresses this by modifying how the sub-trees are built. Instead of considering all features at each split point, the algorithm randomly selects a subset of features, reducing the correlation between the trees' predictions.

Extremely Randomized Trees (ERT)

ERTs take the randomization in Random Forests a step further when splitting nodes in the decision trees. They introduce randomness in two ways:

1. **Random Feature Selection**. At each split point, instead of considering all features, the ERT method randomly selects a subset of features to evaluate.
2. **Random Split Point Selection**. Even within the chosen feature, ERTs do not pick the single best-split point based on error minimization. Instead, they randomly select a possible split point from a range within the feature's values.

In extreme cases, ERT can create entirely random trees with structures that have no dependence on the actual data's values. Also, by the appropriate choice of a parameter, the strength of the randomization can be adjusted to problem specifics.

Adaptive Boosting (AdaBoost)

AdaBoost [16, 17] combines the output of weak learners into a weighted sum, which is the classifier's final output. It builds a linear combination of weak classifiers into a strong classifier. AdaBoost gives a "weight" to each training example, which establishes the likelihood that each example will be included in the training set. As such, examples with higher weights are more likely to be included in the training set, and vice versa.

AdaBoost increases the weight of the misclassified examples after a classifier has been trained. This way, the misclassified examples will comprise a bigger portion of the training set of the subsequent classifier, and ideally, the next trained classifier will perform better on them [18].

Gradient Boosting Decision Tree (GBT)

XGBoost is a machine learning technique similar to other boosting algorithms, i.e. creates a sequence of weak classifiers to solve a classification problem. Its main strategy is to minimize each weak classifier's prediction error based on the previous ones by using the gradient descent method [19]. The weak classifiers are usually decision trees of linear classifiers, e.g. linear regression. In our experiments, we employ three types of boosters in conjunction with CART as the weak classifier, as follows:

1. **Gbtree booster**: uses a gradient descent algorithm to minimise the loss function
2. **Dart booster**: is a variant of gbtree that prevents overfitting by utilizing dropout techniques
3. **Gblinear booster**: substitutes generalised linear regression for decision trees

We train three distinct Gradient Boosted Trees (XGBoost-gbtree, XGBoost-dart, and XGBoost-linear) using the CART algorithm and the three different boosters.

ShallowWaves Ensemble

In our original paper [3], we propose a new ensemble ML architecture, i.e., ShallowWaves. The proposed ShallowWaves Ensemble consists of three branches and a Hard Voting classifier (Fig. 4.1).

The first branch executes a Random Forests with CART classifier (RF CART), the second one executes an Extremely Randomized Trees with CART classifier (ERT CART), and the final one executes a Gradient Boosted Trees with Gbtree booster (XGBoost-gbtree). Thus, the first two algorithms grow CART trees using bagging, while the last uses gradient boosting techniques. The classifiers for the first two algorithms are implemented using the Sklearn library, as for the last one, we use the XGBClassifier from the xgboost library.

In the final step, the predictor gathers the results of each model and uses a hard voting method to make the final decision. In our experiments, we deduce that by using this method, ShallowWaves Ensemble reduces the number of False Positive and False Negative predictions, increasing the accuracy of anomaly identification.

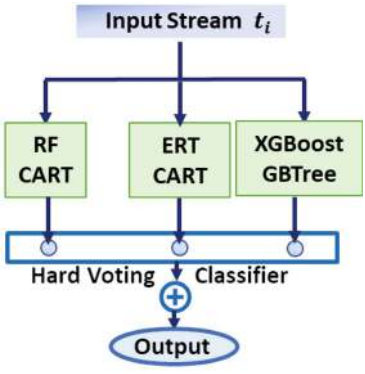


Fig. 4.1 ShallowWaves ensemble model

DeepWaves Ensemble

DeepWaves Ensemble is the second proposed architecture described in our original paper [3]. This ensemble consists of four branches, and each branch has several types of Deep Learning networks and layers, i.e., CNN, LSTM, MapPooling (MP), Flatten, and Dense. The output from the four branches goes into a Linear classifier which gives the final prediction (Fig. 4.2).

The first branch extracts and creates informative representations using CNN (Sect. 4.4) and MaxPolling (MP) layers. MaxPooling layers perform a downsampling operation on the data, reducing its dimensionality while preserving the most important features. A Flatten layer is used towards the end, in order to take the multi-dimensional output from the convolutional layer and transform it into a single-dimensional vector. Finally, the Dense layer combines the features and makes predictions about the category an input belongs to.

The next two branches are more suited in situations where the dataset size is small and employ various CNN and LSTM layer configurations to extract additional features and enhance the overall accuracy of the architecture. As final steps, we also add a Flatten and a Dense layer. The final branch represents the input data's features hierarchically by using stacked LSTM layers. A Dense layer combines the features and outputs the predictions.

The Deep Learning models are implemented using the Keras library.

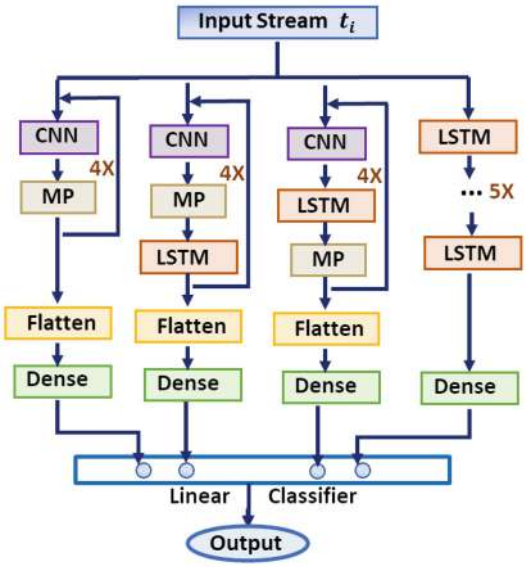


Fig. 4.2 DeepWaves ensemble model

4.6 Results and Discussions

This section includes the dataset description and an analysis of the results of the detection task, as well as final discussions. The experiments for the detection task are done using a 6-node cluster. Each node has the following specifications: Ubuntu 22.04 x 64 OS, Intel Core i7-4790S CPU with 8 cores at 3.20GHz, 16 GB RAM, and 500GB HDD.

4.6.1 LIGO Dataset

The dataset used for training and testing consists of a set of annotated Gravitational Waves time series, mostly labeled with different types of noises and glitches. The original data was collected from September 2015 to December 2015, during the first observing run (O1) [21] of the Advanced LIGO, which includes the Hanford (H), and Livingston (L) detectors.

The dataset we used contains 6,667 entries. These entries are labeled with 22 different categories of transient noise. This noise has properties that change over time (non-stationary) and does not follow a normal distribution (non-Gaussian). As can be seen in Table 4.1, the number of entries varies greatly between each noise category. For example, there are many more entries classified as "Blip" (1763 entries) compared to entries classified as "1080Lines" (only 4 entries). This uneven distribution across categories is something we need to consider when analyzing the data.

We use the dataset's eight features and one label for training and prediction. Each entry has the following features: the time at which the entry was detected (*GPStime*), the peak frequency of the gravitational wave spectrum (*peakFreq*), the signal-to-noise ratio (*snr*), the central frequency of the wavelet (*centralFreq*), *duration*, *bandwidth*, *id*, and the interferometer which captured the entry (*ifo*).

4.6.2 Classical Machine Learning Results

In this subsection, we present the results of the glitch-detection task when using classical Machine Learning models, i.e., KNN, GNB, LogReg, CART, C4.5, and SVM (for more details on these models see Sect. 4.3).

To get the most out of our classical Machine Learning models, we used a technique called grid search to fine-tune their hyperparameters. This helps us identify the configuration that works best for our data. Thus, we carefully adjusted the models' settings and then evaluated their performance using various metrics that consider both overall accuracy and how well they handle the uneven categories in our data. To measure how well each model performed, we considered the following metrics: Accuracy, Weighted Precision, Micro Precision, Weighted Recall, and Micro Recall.

Table 4.1 Glitch classes for the detection task

Label	#	Description
Scattered light	427	Low-frequency, long duration, humpy
Power line	450	Narrow in frequency, last for $\approx 0.2 - 0.5$ s
1080Lines	4	Steady stream, around 1080 Hz
1400Ripples	83	Short-duration, around 1400 Hz
Air compressor	57	Short-duration glitches—around 50 Hz
Blip	1763	Teardrop' shape, 30–500 HZ
Repeating blips	91	Blip glitches that repeat
Violin mode	137	Short and dot-like
Whistle	146	W or V shape
Scratchy	269	Short-duration repeating, intermediate freq.
Helix	270	Resemble a vortex, intermediate freq.
Light modulation	400	Several bright spikes in close succession
Low frequency burst	527	Loud, short-lived, low-freq.
Wandering line	21	Lines, long duration, meander in freq.
Koi fish	709	Similar to Blip, resemble a fish in shape
Low frequency lines	494	Horizontal lines at low freq.
Chirp	60	Sweeping upwards in freq. over time
Extremely loud	448	From a major disturbance in the detectors
Paired doves	26	Repeating glitches, alternate increasing-decreasing freq.
Tomte	93	Lower-freq., usually triangular in shape
No glitch	41	No apparent transient noise structure
None of the above	151	A catch-all for glitches that do not fit

Table 4.2 Classical machine learning models—results

Algorithm	Accuracy	Weighted precision	Micro precision	Weighted recall	Micro recall
KNN	0.70 ± 0.01	0.70 ± 0.01	0.70 ± 0.01	0.70 ± 0.01	0.70 ± 0.01
GNB	0.63 ± 0.01	0.66 ± 0.01	0.63 ± 0.01	0.63 ± 0.01	0.63 ± 0.01
LogReg	0.61 ± 0.01	0.52 ± 0.02	0.61 ± 0.01	0.61 ± 0.01	0.61 ± 0.01
CART	0.78 ± 0.01	0.78 ± 0.01	0.78 ± 0.01	0.78 ± 0.01	0.78 ± 0.01
C4.5	0.79 ± 0.01	0.79 ± 0.01	0.79 ± 0.01	0.79 ± 0.01	0.79 ± 0.01
SVM	0.58 ± 0.01	0.58 ± 0.01	0.58 ± 0.01	0.58 ± 0.01	0.58 ± 0.01

Table 4.2 shows the performance of our classical Machine Learning models. As a first point, the decision tree algorithms, i.e., CART and C4.5, achieved high accuracy for our glitch detection task. Secondly, Gaussian Naive Bayes (GNB), Logistic Regression (LogReg), and Support Vector Machine (SVM) had the worst performance.

4.6.3 Deep Learning Results

In this subsection, we analyze the results obtained by the employed Deep Learning models.

Deep Neural Networks have a large number of parameters which increase exponentially with the layer added to the architecture. Due to the size of such architectures, it is not feasible to use techniques such as hyperparameter tuning. Although fine-tuning the hyperparameters is not done, we employ other methods to better determine the performances of the proposed model. Firstly, we use Stratified K-Folds with a $k = 10$ to split the data into training and test sets for all our experiments. Secondly, we perform ablation testing, analyzing each individual model to determine the impact of each NN on the overall architecture.

Table 4.3 presents the results obtained by employing Stratified K-Folds with a $k = 10$ on each individual Deep Learning model. Furthermore, we conduct a comparative analysis of the performance of the Deep models to evaluate their effectiveness in detecting glitches. The Multi-Layer Perceptron (MLP) uses 5 layers and a single output layer. The first and fifth layers contain 8 neurons and use the ReLU activation function. The second and fourth layers have 16 neurons and use the softmax activation function. The middle layer, i.e., the third layer, uses 32 neurons and the ReLU activation function. The output layer has a neuron for each class and uses the softmax activation function. Based on the obtained results, we see no real difference between the Perceptron, which contains only one layer, and the MLP. For the Deep Belief Network (DBN), we use a hidden layer structure containing 256 by 256 units and the ReLU activation function for neurons. With this configuration, the DBN obtains the overall worse results in our experiments. Based on the results obtained by the DBN

Table 4.3 Deep learning models—results

Algorithm	Accuracy	Weighted precision	Micro precision	Weighted recall	Micro recall
Perceptron	0.74 ± 0.09	0.77 ± 0.04	0.74 ± 0.09	0.74 ± 0.09	0.74 ± 0.09
MLP	0.74 ± 0.03	0.63 ± 0.07	0.74 ± 0.03	0.74 ± 0.03	0.74 ± 0.03
DBN	0.60 ± 0.01	0.53 ± 0.02	0.60 ± 0.01	0.60 ± 0.01	0.60 ± 0.01
LSTM	0.81 ± 0.03	0.74 ± 0.07	0.81 ± 0.03	0.81 ± 0.03	0.81 ± 0.03
CNN	0.71 ± 0.05	0.63 ± 0.07	0.71 ± 0.05	0.71 ± 0.05	0.71 ± 0.05
$5 \times$ LSTM	0.83 ± 0.01	0.73 ± 0.02	0.83 ± 0.01	0.83 ± 0.01	0.83 ± 0.01
$4 \times$ (CNN + MP)	0.81 ± 0.02	0.75 ± 0.03	0.81 ± 0.02	0.81 ± 0.02	0.81 ± 0.02
$4 \times$ (CNN + LSTM + MP)	0.84 ± 0.03	0.80 ± 0.06	0.84 ± 0.03	0.84 ± 0.03	0.84 ± 0.03
$4 \times$ (CNN + MP + LSTM)	0.86 ± 0.03	0.81 ± 0.06	0.86 ± 0.03	0.86 ± 0.03	0.86 ± 0.03

architecture, we can conclude that such a complex Deep Learning architecture does not manage to find the best representation of the input data to classify the glitches accurately.

Out of all the Deep Neural Network architectures we tested, the best performer was one that combined a Convolutional Neural Network (CNN), a Max Pooling layer (MP), and an LSTM layer four times in a sequence, i.e., 4 × (CNN + MP + LSTM).

Interestingly, increasing the number of LSTM layers stacked together resulted in a slight decrease in performance. However, it still did better than using just a single LSTM layer.

The CNN-based architectures use a MaxPooling Layer (MP) to calculate the downsampled feature maps. From the experimental evaluation, we observed that the worst-performing architectures have a CNN layer followed by an MP one. As the number of layers increases, we also observe that the performance increases significantly, i.e., CNN versus 4 x (CNN + MP). The performance is further increased by adding LSTM layers to the CNN architecture, i.e., 4 × (CNN + LSTM + MP) and 4 × (CNN + MP + LSTM). Thus, we can conclude that by alternative layer CNN + MP layers with LSTM layers, the performance increases significantly.

4.6.4 Ensembles Results

Finally, we present our results for the ensemble models.

Table 4.4 dives into the performance of the tested ensemble models. Similar to the single models, we see some interesting trends. Ensemble methods that combine multiple decision trees, like AdaBoost, Random Forests, Extremely Randomized

Table 4.4 Ensembles models—results

Algorithm	Accuracy	Weighted precision	Micro precision	Weighted recall	Micro recall
AdaBoost CART	0.78 ± 0.01	0.79 ± 0.01	0.78 ± 0.01	0.78 ± 0.01	0.78 ± 0.01
AdaBoost C4.5	0.79 ± 0.01	0.79 ± 0.01	0.79 ± 0.01	0.79 ± 0.01	0.79 ± 0.01
RF CART	0.84 ± 0.01	0.83 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01
RF C4.5	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01
ERT CART	0.84 ± 0.01	0.83 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01
ERT C4.5	0.84 ± 0.01	0.83 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01
XGBoost-gbtrees	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01
XGBoost-gblinear	0.61 ± 0.01	0.53 ± 0.02	0.61 ± 0.01	0.61 ± 0.01	0.61 ± 0.01
XGBoost-dart	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01
ShallowWaves ensemble	0.89 ± 0.02	0.88 ± 0.03	0.89 ± 0.02	0.89 ± 0.02	0.89 ± 0.02
DeepWaves ensemble	0.91 ± 0.03	0.87 ± 0.05	0.91 ± 0.03	0.91 ± 0.03	0.91 ± 0.03

Trees, and XGBoost with gradient boosting trees (XGBoost-gbtree), again achieve high accuracy. Notably, XGBoost-gbtree and XGBoost with tree boosting (XGBoost-dart) perform identically, while the linear model (XGBoost-gblinear) falls behind.

The ShallowWaves ensemble, built using classical Machine Learning techniques, has the best accuracy among all the other classical ensembles we tested. However, the DeepWaves Ensemble surpasses all the other models, including ShallowWaves, demonstrating superior performance in glitch detection.

4.6.5 Discussions

As a final discussion point, we summarized the results of the best models for our glitch detection task (Table 4.5).

The DeepWaves Ensemble truly shines among the models we tested. Through the integration of various deep learning architectures, we have implemented a powerful ensemble that achieves several key improvements:

1. **Reduced Errors**: DeepWaves Ensemble minimizes both false positives (mistaking noise for glitches) and false negatives (missing actual glitches). This leads to better scores on all the evaluation metrics we used.
2. **Enhanced Feature Understanding**: The ensemble learns a more comprehensive picture of the data (feature space) by leveraging the strengths of different deep learning models.

Table 4.5 Best classification models—results summary

Algorithm	Accuracy	Weighted precision	Micro precision	Weighted recall	Micro recall
RF CART	0.84 ± 0.01	0.83 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01
RF C4.5	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01
ERT CART	0.84 ± 0.01	0.83 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01
ERT C4.5	0.84 ± 0.01	0.83 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01
XGBoost-gbtree	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01
XGBoost-dart	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01
$4 \times$ (CNN + LSTM + MP)	0.84 ± 0.03	0.80 ± 0.06	0.84 ± 0.03	0.84 ± 0.03	0.84 ± 0.03
$4 \times$ (CNN + MP + LSTM)	0.86 ± 0.03	0.81 ± 0.06	0.86 ± 0.03	0.86 ± 0.03	0.86 ± 0.03
ShallowWaves Ensemble	0.89 ± 0.02	0.88 ± 0.03	0.89 ± 0.02	0.89 ± 0.02	0.89 ± 0.02
DeepWaves Ensemble	0.91 ± 0.03	0.87 ± 0.05	0.91 ± 0.03	0.91 ± 0.03	0.91 ± 0.03

3. **Unveiling Hidden Patterns**: DeepWaves Ensemble can identify subtle characteristics within the data (hidden features) that help it distinguish between different glitch types more effectively.

As final thoughts, the most important points to take from our experiments are as follows:

1. **Not all complex models are best**: For datasets like ours with uneven categories (highly unbalanced), simpler Machine Learning models can sometimes outperform even Deep Learning models. This is because Deep Learning models might struggle to find the right settings, i.e., parameters, for the data.
2. **Tailor-made solutions are key**: The best approach depends on the specific data we are working with. We need a data-driven strategy to pick the most effective model and its hyperparameters.
3. **Strength in numbers**: In our experiments, combining multiple models into an ensemble proved to be the most successful strategy.

4.7 Conclusions

Gravitational wave hunters face a challenge: glitches. These bursts of noise can mimic real gravitational waves, potentially hiding the true signals from space. To tackle this problem, this chapter investigates the use of Machine Learning approaches to solve the problem of efficiently detecting glitches in Gravitational Wave data streams. In our research, we explored multiple classical Machine Learning, Deep Learning, and ensemble models to see if they can effectively distinguish glitches from real gravitational waves in the data.

Our experiments revealed that the champion glitch detector among all the models we tested is the DeepWaves Ensemble, achieving the highest overall accuracy. Following closely behind is the ShallowWaves Ensemble.

References

1. Aasi, J., Abbott, B.P., Abbott, R., Abbott, T., Abernathy, M.R., Ackley, K., Adams, C., Adams, T., Addesso, P., Adhikari, R.X., Adya, V.: Advanced ligo. Class. Quantum Grav. **32**(7), 074001 (2015)
2. Acernese, F.A., Agathos, M., Agatsuma, K., Aisa, D., Allemandou, N., Allocca, A., Amarni, J., Astone, P., Balestri, G., Ballardin, G., Barone, F.: Advanced Virgo: a second-generation interferometric gravitational wave detector. Class. Quantum Grav. **32**(2), 024001 (2014)
3. Apostol, E.S., Truică, C.O.: Efficient machine learning ensemble methods for detecting gravitational wave glitches in LIGO time series. In: 2023 IEEE 19th International Conference on Intelligent Computer Communication and Processing (ICCP), pp. 79–86. IEEE (2023)
4. LIGO: Virgo Data Characterization and Impact on Gravitational Wave Searches. https://www.ligo.org/science/Publication-VirgoDetchar/ (2024)
5. Strickland, J.: Data Analytics Using Open-source Tools. Lulu.com (2016)

6. Cortes, C., Vapnik, V.N.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
7. Quinlan, J.R.: Induction of decision trees. Mach. Learn. **1**(1), 81–106 (1986)
8. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers (1993)
9. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Wadsworth & Brooks/Cole Advanced Books & Software, Monterey, CA (1984)
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
11. Hua, Y., Guo, J., Zhao, H.: Deep belief networks and deep learning. In: Proceedings of 2015 International Conference on Intelligent Computing and Internet of Things, pp. 1–4. IEEE (2015)
12. Breiman, L.: Bagging predictors. Mach. Learn. **24**(2), 123–140 (1996)
13. Schapire, R.E.: The strength of weak learnability. Mach. Learn. **5**(2), 197–227 (1990)
14. Ho, T.K.: Random decision forests. In: International Conference on Document Analysis and Recognition, pp. 278–282 (1995)
15. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
16. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: International Conference on Machine Learning, pp. 148–156 (1996)
17. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. **55**(119), 23–37 (1997)
18. Akerkar, R., Sajja, P.S.: Intelligent Techniques for Data Science. Springer International Publishing (2016)
19. Mason, L., Baxter, J., Bartlett, P.L., Frean, M.: Boosting algorithms as gradient descent. In: Advances in Neural Information Processing Systems, vol. 12, pp. 512–518. MIT Press (1999)
20. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst. 1097–1105 (2012)
21. LIGO Scientific Collaboration: LIGO O1 Data Release. Gravitational Wave Open Science Center (2016)

Chapter 5
Denoising Gravitational-Wave Signals from Binary Black Holes with Dilated Convolutional Autoencoder

Michał Bejger, Philippe Bacon, and Agata Trovato

Abstract Broadband frequency output of gravitational-wave detectors is a non-stationary and non-Gaussian time series data stream populated by local disturbances and transient artifacts, which evolve on the same timescale as the gravitational-wave signals and may corrupt the astrophysical information. This contribution presents a denoising algorithm dedicated to expose the astrophysical signals by employing a convolutional neural network in the encoder-decoder configuration. The denoising procedure is applied to coalescing binary black hole signals in the publicly available LIGO O1 time series strain data. The denoising convolutional autoencoder neural network is trained on a dataset of simulated astrophysical signals injected into the real detector's noise and a dataset of detector noise artifacts ("glitches"), and its fidelity is tested on real gravitational-wave events from O1 and O2 LIGO-Virgo observing runs.

5.1 Introduction

Raw GW strain data are fundamentally noisy time series in which the GW signals are hidden; for the detailed description of the GW data, see [4, 6]. Classically, in order to confirm the existence of a signal in the data time-series, one has to apply a matched filtering methods (e.g., [19, 21, 32]). This is an optimal technique only when the background noise is Gaussian and stationary. It requires substantial computational resources and a fine grid of filter templates build from GW signals parameters to find the match.

M. Bejger (✉) · P. Bacon · A. Trovato
INFN Sezione di Ferrara, Ferrara, Italy
e-mail: bejger@fe.infn.it

Nicolaus Copernicus Astronomical Center, Polish Academy of Sciences, Warszawa, Poland

Université Paris Cité, CNRS, Paris, France

Dipartimento di Fisica, Università di Trieste, Trieste, Italy

INFN, Sezione di Trieste, Trieste, Italy

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_5

Here we are studying an enhancement to the established data-processing methods in order to facilitate a trigger generation process or simply be one of the first parts of a GW data-analysis pipeline. Denoising of the GW data was applied in the past using the total-variation method [25, 27], with the split Bergman regularization to obtain the total-variation regularization [26], and with the dictionary learning [28]. From the deep neural network (NN) point-of-view, denoising methods were applied in [31] with the `WaveNet` implementation [30], as well as using the auto-encoder (AE) architecture [23, 24] to perform the denoising task, i.e. as a denoising auto-encoder (DAE) [13]: instead of encoding and subsequently decoding an input sequence to itself, the training consists of feeding the noisy ("corrupted") input and expecting a noiseless ("clean") output. Recent works that apply this paradigm include [22] with the concept of Long Short-Term Memory/ recurrent neural networks (LSTM/RNN, see e.g. [14, 15, 20]), as well as [10], which used a combination of the convolutional NN (CNN, see e.g. [11, 12]) and LSTM; simple artificial NN [18] were also employed to denoise GW signals overlapping with instrumental glitches. An algorithm implemented in [17], based on the local polynomial approximation combined with the relative intersection of confidence intervals rule for the filter support selection is applied to denoise the GW burst signals emitted during core collapse supernovæ events.

In [8], on which this chapter is based, a purposefully simple version of the DAE, based on one-dimensional input CNN paradigm, was implemented and applied to the noisy ("corrupted") time series GW data containing astrophysical signals immersed in the real noise, in order to study limitations in recovering the noiseless ("clean") GW signals in this realistic setup. The CNN-DAE approach has advantages over implementations of DAE already existing in the literature, the primary being the fact that the CNN implementation is smaller and trains faster than recurrent NN. It is demonstrated that a relatively small CNN DAE with a few dilated decoder layers [33] is able to train on the GW signal waveforms injected to realistic LIGO data time series, and recover real GW events. This type of a lightweight algorithm can be considered a potentially useful trigger generator, performing a role of rapid initial classification of GW signals, and/or data characterization tasks.

5.2 Denoising AE Model

CNN is a type of NNs that applies a set of convolutions (by means of the kernel filters) to the network's input [11, 12]. In the context of denoising, training a CNN consists in finding the filters weights which optimally extract the preferable features in input data and are present in clean (noise-free) data. The CNN layers in this work adopt the AE architecture. In general, the AE NN represents an identity function by compressing the representation of input data and then decompressing it. During the training, the overall network will learn its own sparse representation of the input signals. AE are composed of two networks: an encoder g_ϕ, and a decoder f_θ. A denoising AE (DAE) procedure consists in training an AE with a corrupted version

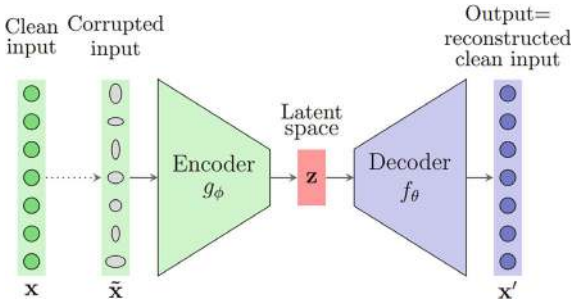


Fig. 5.1 Data flow diagram of a DAE architecture. The clean input data $\mathbf{x}$ is corrupted (denoted by $\tilde{\mathbf{x}}$) and fed to the encoder part; for the well-trained network, the decoded output (reconstructed clean input) $\mathbf{x}'$ should be a close analogue of the clean input $\mathbf{x}$, i.e. $\mathbf{x}' \approx \mathbf{x}$. The middle part of the DAE—the latent space—is denoted by $\mathbf{z}$. In case of $\mathbf{x} \equiv \tilde{\mathbf{x}}$ the DAE becomes a classical AE

of the input signal (i.e. clean signal immersed in the noisy time series), denoted by $\tilde{\mathbf{x}}$, and by demanding that the recovered output $\mathbf{x}'$ is as close to the original clean input $\mathbf{x}$ as possible. A schematic representation of a DAE is presented in Fig. 5.1.

For the DAE loss function, the following was chosen:

$$L_{DAE}(\theta, \phi) = \sum_{i=1}^{N} \left(x_i - f_\theta(g_\phi(\tilde{x}_i))\right)^2, \tag{5.1}$$

where $f_\theta(g_\phi(\tilde{\mathbf{x}})) = \mathbf{x}'$ is the DAE output, and $\mathbf{x}$ is the ground-truth (clean) signal waveform input. The network structure is described in detail in Sect. 2.2, Table 1 of [8].

5.3 Training and Testing Data

The input "clean" signals used in this project are simulated astrophysical GW signals from binary BHs (BBHs). In general, astrophysical GW signals from close binary compact systems exhibit a characteristic increase of GW amplitude and frequency during the inspiral (the "chirp"), followed by the merger of the binary components, and the ringdown GW emission from the remnant [2]. Approximately, the GW inspiral frequency evolves as [1]

$$f_{GW}^{-8/3}(t) = \frac{(8\pi)^{8/3}}{5} \left(\frac{G\mathcal{M}_c}{c^3}\right)^{5/3} (t_c - t) \; + \text{higher order corrections}, \tag{5.2}$$

where t_c is the time of coalescence, and the $\mathcal{M}_c$ is a function of component masses M_1 and M_2, called the *chirp mass*:

$$\mathcal{M}_c = \frac{(M_1 M_2)^{3/5}}{(M_1 + M_2)^{1/5}}. \tag{5.3}$$

For production runs, it was assumed that signal waveforms—the amplitude-frequency evolution $h(f)$ or, equivalently, $h(t)$—are well-modeled using general relativity. For the sake of this study the SEOBNRv4 waveform model [9] was employed, assuming non-spinning components with masses randomly chosen from a uniform distribution in the range $M_1, M_2 \in (10,\ 30)\, M_\odot$, compatible with the current state of observational knowledge on the binary BH population [3]. An optimal sky localization is assumed for a given time segment. Other parameters describing the source were selected randomly: coalescence phase and polarization angle from a range of $(0, 2\pi)$, and the inclination angle from a range of $(0, \pi)$. The "corrupted" input is obtained injecting these astrophysical GW signals in pre-selected only-noise time series segments from the science-quality data stream of the LIGO Livingston detector, collected during the O1 observing run (see the Gravitational Wave Open Science Center [6] for details).

Working under the assumption of the additive property of the noise (i.e., the time series d containing the GW signal h immersed in noise n is $d = n + h$), then the signal-to-noise ratio (SNR) ρ, corresponding to the best matching filter (template h equals the signal) is

$$\rho = \frac{(d|h)}{\sqrt{(h|h)}}, \quad \text{where} \quad (d|h) = 4\Re \int \frac{\tilde{d}(f)\tilde{h}^*(f)}{S_n(f)} \mathrm{d}f, \tag{5.4}$$

with $\tilde{d}(f)$ a Fourier transform of the time series $d(t)$, and $S_n(f)$ the power spectral density (PSD) of the detector; the asterisk denotes complex conjugation [19]. The detector's PSD $S_n(f)$ represents the frequency-dependent sensitivity in a broad range of frequencies, and is quantified by its sensitivity curve (for details on how the PSD is computed see, e.g., [4]). From the astrophysical perspective, the SNR is a function of the waveform amplitude and, since the waveform describes the evolution of the GW amplitude, is inversely proportional to the luminosity ("loudness") distance. The data set instances are labelled with their *optimal* matched-filter SNR

$$\rho_{opt} = \sqrt{(h|h)} = \sqrt{4 \int_0^\infty \frac{|\tilde{h}(f)|^2}{S_n(f)} \mathrm{d}f}, \tag{5.5}$$

which approximates ρ, assuming that the noise effect is negligible, $d \approx h$. The optimal matched-filter SNR ρ_{opt} is a good first order approximation to the actual matched-filter SNR ρ in e.g. stationary Gaussian noise [16].

Subsequently, a synthetic distribution of source luminosity distances was produced such that the ρ_{opt} distribution is uniform in the range of $(5,\ 20)$ in order to consider a wide range of signals during the training. Additionally, the dependence of signals' strength at different frequencies was normalized by performing the *whitening* of the time series data with added signals: the Fourier representation of the time

domain data was divided by an estimate of the amplitude spectral density of the noise $\sqrt{S_n(f)}$ (ASD, square root of the PSD) to ensure that the data has equal significance in each frequency bin [5]. The network is trained on 7000 data time series containing injected astrophysical signals. In addition, the training set contains 1000 time series from the O1 LIGO Livingston data when known instrumental artifacts ("glitches") are present in the data.

5.4 Selected Results

The training is evaluated using the figure of merit commonly used in the GW astronomy, the waveform overlap O, which compares the original ("clean", ground truth) waveform h and the denoised waveform h^d obtained at the output of the DAE:

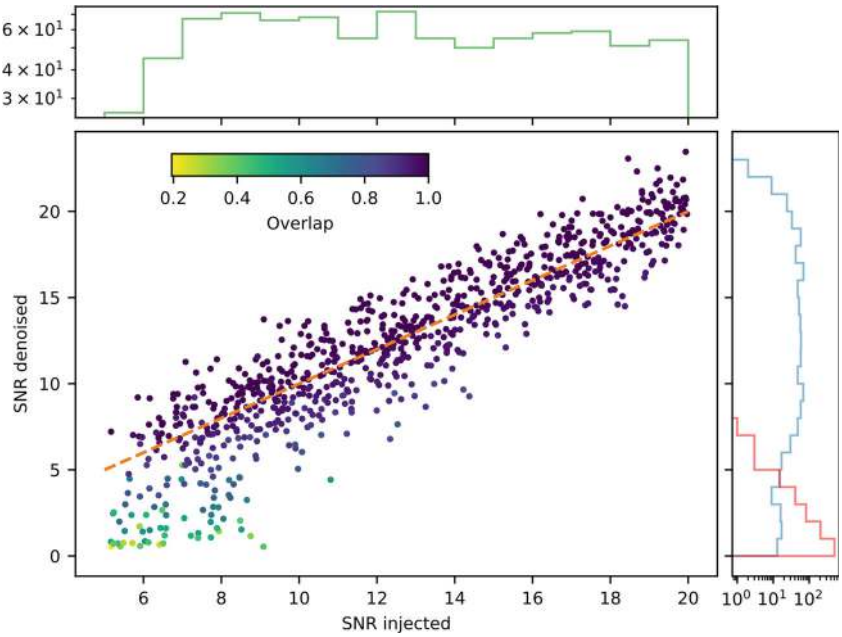


Fig. 5.2 Denoised SNR (calculated from the DAE output analogously to Eq. 5.5, but with denoised output waveform h^d instead of the originally-injected signal h, vertical axis) as a function of the injected SNR (horizontal axis) for a testing dataset of 1000 data instances with added astrophysical GW waveforms. Points are colored by their corresponding overlap values (Eq. 5.6). Orange dashed line denotes the denoised SNR equal to injected SNR. Side histograms (in logarithmic scale) show the distribution of the injected SNR (upper plot), and SNRs denoised from samples containing added GW waveforms (blue histogram), and—for comparison—not containing GW signals (i.e. pure noise, red histogram), respectively

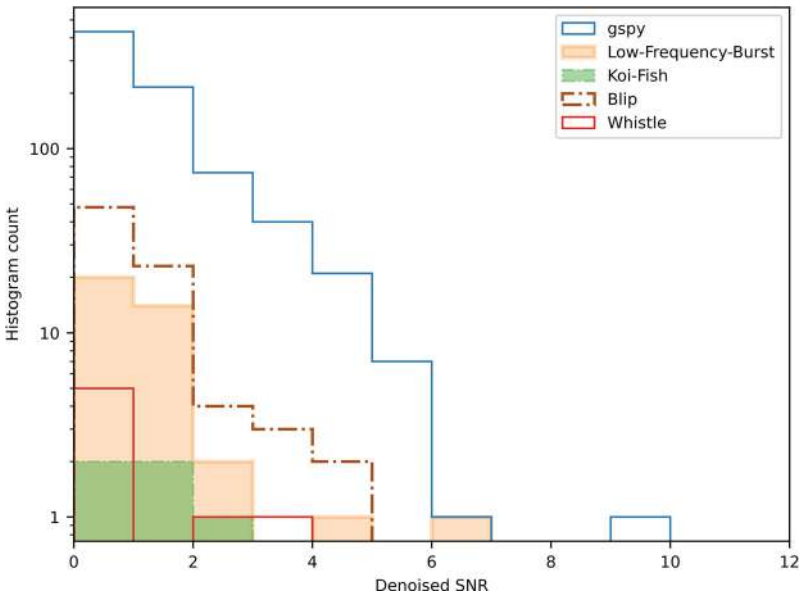


Fig. 5.3 Evaluation of the DAE on instrumental glitches. The logarithmic vertical scale plot shows histograms of denoised output SNR for a selection of 38 Low Frequency Burst glitches, 5 Koi Fish type glitches, 80 Blips and 7 Whistle glitches. Blue line marks the evaluation on 792 assorted various types of glitches. All the glitches data are obtained from the `Gravity Spy` database [34]. The glitches have their estimated intrinsic SNR > 10

$$O = \sqrt{\sum_{i=0}^{N} h_i h_i^d \left(\sum_{i=0}^{N} h_i h_i \right)^{-1}}, \tag{5.6}$$

where $N = 2048$ is the number of points in the time series. Figure 5.2 shows the comparison between the injected SNR ρ_{opt} and the recovered (denoised) output SNR $\rho_{out,d}$, both calculated using the optimal matched filter SNR formula (Eq. 5.5), for 1000 instances from the validation set. The color code indicates the waveforms overlap. As expected, in cases of high overlap the denoised SNR approximates quite well the injected one. The cases of lower overlap have a denoised SNR close to zero. The distribution follows the ideal $\rho_{out} \equiv \rho_{out,d}$ relation with a root-mean-square of residuals of 1.9 and variance of residuals of 3.8. The denoised SNR may be used as an approximate proxy for detection criterion: 8.2% of the output signals have $\rho_{opt,d} < 5$, whereas 1.3% of signals have both $\rho_{opt} > 8$ and $\rho_{opt,d} < 5$, i.e. are potentially strong enough to detect with the standard methods, but incorrectly recovered by the DAE. Additionally, the denoising procedure was performed on the same detector time series samples but *without* injected GW signals, to study the output signals. The distribution of output SNR in that case is depicted by the red histogram; only a few noise-only samples have $\rho_{out,d} > 5$.

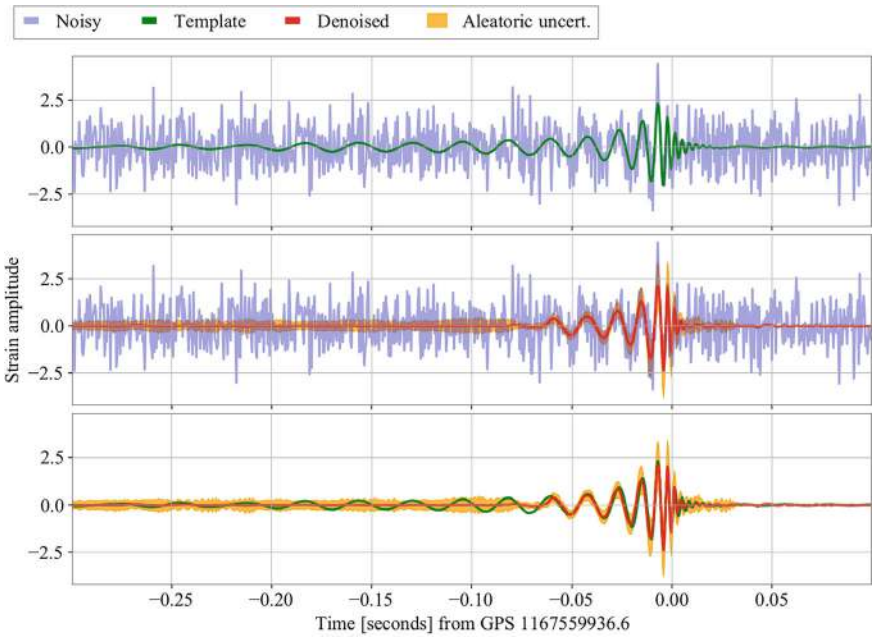


Fig. 5.4 Denoising applied to the O2 data GW170104 event for the H1 detector. The green curve is the best matched filter result of the parameter estimation, red curve is the DAE output, and orange region denotes aleatoric uncertainty estimate (see [8] for details). Component masses are $30.8^{+7.3}_{-5.6}$ $M_\odot$ and $20.0^{+4.9}_{-4.6}$ $M_\odot$, and the single-detector optimal SNRs is $9.5^{+1.3}_{-1.6}$

The DAE is also evaluated on known instrumental glitches, extracted from the `Gravity Spy` database [34]. The corresponding 1 s LIGO data segments centered at these GPS times have been downloaded via the Gravitational Wave Open Science Center [6]. Results are shown in Fig. 5.3. Last, but not least, the trained DAE network was validated on real events. Here, the LIGO-Virgo O2 significant event detected by the Hanford detector, GW170104, is presented. The result of denoising is shown in Fig. 5.4; see the published article [8] for more details, examples and discussion.

5.5 Conclusions

The DAE method presented here is potentially a versatile pre-processing tool prior to detection and/or source parameter estimation pipelines used to analyse data collected by ground based instruments. As an example of its usefulness, it was employed for denoising of a 1-detector trigger, found in O1 LIGO data with deep neural network algorithms developed in [29], see Chap. 1 in this book; the DAE produced an output consistent with the standard parameter estimation method, `bilby` [7].

References

1. Abbott, B.P., Abbott, R., Abbott, T.D., Abernathy, M.R., Acernese, F., Ackley, K., Adams, C., Adams, T., Addesso, P., et al.: The basic physics of the binary black hole merger gw150914. Annalen der Physik **529**(1–2), 1600209 (2016). https://doi.org/10.1002/andp.201600209
2. Abbott, B.P., et al.: The basic physics of the binary black hole merger GW150914. Annalen der Physik **529**(1–2), 1600209 (2017). https://doi.org/10.1002/andp.201600209
3. Abbott, B.P., et al.: GWTC-1: a gravitational-wave transient catalog of compact binary mergers observed by LIGO and Virgo during the first and second observing runs. Phys. Rev. X **9**(3), 031040 (2019). https://doi.org/10.1103/PhysRevX.9.031040
4. Abbott, B.P., et al.: A guide to LIGO-Virgo detector noise and extraction of transient gravitational-wave signals. Class. Quantum Grav. **37**(5), 055002 (2020). https://doi.org/10.1088/1361-6382/ab685e
5. Abbott, B.P., et al.: A guide to LIGO-virgo detector noise and extraction of transient gravitational-wave signals. Class. Quantum Grav. **37**(5), 055002 (2020). https://doi.org/10.1088/1361-6382/ab685e
6. Abbott, R., et al.: Open data from the first and second observing runs of Advanced LIGO and Advanced Virgo. SoftwareX **13**, 100658 (2021). https://doi.org/10.1016/j.softx.2021.100658
7. Ashton, G., et al.: BILBY: a user-friendly Bayesian inference library for gravitational-wave astronomy. Astrophys. J. Suppl. **241**(2), 27 (2019). https://doi.org/10.3847/1538-4365/ab06fc
8. Bacon, P., Trovato, A., Bejger, M.: Denoising gravitational-wave signals from binary black holes with a dilated convolutional autoencoder. Machine Learning: Science and Technology **4**(3), 035024 (2023). https://doi.org/10.1088/2632-2153/acd90f
9. Bohé, A., et al.: Improved effective-one-body model of spinning, nonprecessing binary black holes for the era of gravitational-wave astrophysics with advanced detectors. Phys. Rev. D **95**(4), 044028 (2017). https://doi.org/10.1103/PhysRevD.95.044028
10. Chatterjee, C., Wen, L., Diakogiannis, F., Vinsen, K.: Extraction of binary black hole gravitational wave signals from detector data using deep learning. Phys. Rev. D **104**(6), 064046 (2021). https://doi.org/10.1103/PhysRevD.104.064046
11. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. The MIT Press (2016)
12. Gu, J., et al.: Recent advances in convolutional neural networks. Pattern Recognition **77**, 354–377 (2018) DOI https://doi.org/10.1016/j.patcog.2017.10.013. URL http://www.sciencedirect.com/science/article/pii/S0031320317304120
13. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006) DOI https://doi.org/10.1126/science.1127647. URL http://www.ncbi.nlm.nih.gov/sites/entrez?db=pubmed&uid=16873662&cmd=showdetailview&indexed=google
14. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997) DOI https://doi.org/10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735
15. Jain, L.C., Medsker, L.R.: Recurrent Neural Networks: Design and Applications, 1st edn. CRC Press Inc, USA (1999)
16. Jaranowski, P., Królak, A.: Gravitational-Wave Data Analysis. Formalism and Sample Applications: The Gaussian Case. Living Reviews in Relativity **15**(1), 4 (2012). DOI https://doi.org/10.12942/lrr-2012-4
17. Lopac, N., Lerga, J., Cuoco, E.: Gravitational-Wave Burst Signals Denoising Based on the Adaptive Modification of the Intersection of Confidence Intervals Rule. Sensors **20**(23), 6920 (2020). https://doi.org/10.3390/s20236920
18. Mogushi, K., Quitzow-James, R., Cavaglià, M., Kulkarni, S., Hayes, F.: Nnetfix: an artificial neural network-based denoising engine for gravitational-wave signals. Machine Learning: Science and Technology **2**(3), 035018 (2021) DOI https://doi.org/10.1088/2632-2153/abea69. URL https://dx.doi.org/10.1088/2632-2153/abea69

19. Owen, B.J., Sathyaprakash, B.S.: Matched filtering of gravitational waves from inspiraling compact binaries: Computational cost and template placement. Phys. Rev. D **60**(2), 022002 (1999). https://doi.org/10.1103/PhysRevD.60.022002
20. Pascanu, R., Gulcehre, C., Cho, K., Bengio, Y.: How to Construct Deep Recurrent Neural Networks. arXiv e-prints arXiv:1312.6026 (2013)
21. Sathyaprakash, B.S., Dhurandhar, S.V.: Choice of filters for the detection of gravitational waves from coalescing binaries. Phys. Rev. D **44**, 3819–3834 (1991) DOI https://doi.org/10.1103/PhysRevD.44.3819. URL https://link.aps.org/doi/10.1103/PhysRevD.44.3819
22. Shen, H., George, D., Huerta, E.A., Zhao, Z.: Denoising Gravitational Waves using Deep Learning with Recurrent Denoising Autoencoders. arXiv e-prints arXiv:1711.09919 (2017)
23. Shen, H., George, D., Huerta, E.A., Zhao, Z.: Denoising Gravitational Waves with Enhanced Deep Recurrent Denoising Auto-Encoders. arXiv e-prints arXiv:1903.03105 (2019)
24. Shen, H., Zhao, Z., George, D., Huerta, E.: Denoising Gravitational Waves using Deep Learning with Recurrent Denoising Autoencoders. In: APS April Meeting Abstracts, *APS Meeting Abstracts*, vol. 2018, p. S14.008 (2018)
25. Torres, A., Marquina, A., Font, J.A., Ibáñez, J.M.: Total-variation-based methods for gravitational wave denoising. Phys. Rev. D **90**(8), 084029 (2014). https://doi.org/10.1103/PhysRevD.90.084029
26. Torres, A., Marquina, A., Font, J.A., Ibáñez, J.M.: Split Bregman Method for Gravitational Wave Denoising. In: Gravitational Wave Astrophysics, vol. 40, p. 289 (2015). DOI https://doi.org/10.1007/978-3-319-10488-1_25
27. Torres-Forné, A., Cuoco, E., Marquina, A., Font, J.A., Ibáñez, J.M.: Total-variation methods for gravitational-wave denoising: Performance tests on Advanced LIGO data. Phys. Rev. D **98**(8), 084013 (2018). https://doi.org/10.1103/PhysRevD.98.084013
28. Torres-Forné, A., Marquina, A., Font, J.A., Ibáñez, J.M.: Denoising of gravitational wave signals via dictionary learning algorithms. Phys. Rev. D **94**(12), 124040 (2016). https://doi.org/10.1103/PhysRevD.94.124040
29. Trovato, A., Chassande-Mottin, É., Bejger, M., Flamary, R., Courty, N.: Neural network time-series classifiers for gravitational-wave searches in single-detector periods. arXiv e-prints arXiv:2307.09268 (2023). DOI https://doi.org/10.48550/arXiv.2307.09268
30. van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., Kavukcuoglu, K.: WaveNet: A Generative Model for Raw Audio. arXiv e-prints arXiv:1609.03499 (2016)
31. Wei, W., Huerta, E.A.: Gravitational wave denoising of binary black hole mergers with deep learning. Physics Letters B **800**, 135081 (2020). https://doi.org/10.1016/j.physletb.2019.135081
32. Wiener, N.: Extrapolation, Interpolation, and Smoothing of Stationary Time Series. Wiley, New York (1949)
33. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions (2016)
34. Zevin, M., et al.: Gravity Spy: integrating advanced LIGO detector characterization, machine learning, and citizen science. Classical and Quantum Gravity **34**(6), 064003 (2017). https://doi.org/10.1088/1361-6382/aa5cea

Chapter 6
A Fast and Time-Efficient Glitch Classification Method: A Deep Learning-Based Visual Feature Extractor for Machine Learning Algorithms

Osman Tayfun Bişkin, İsmail Kirbaş, and Ali Çelik

Abstract Glitches, non-Gaussian transient waves which mimic gravitational-wave signals, are abundant in detectors and impact data quality. Therefore, identifying glitch type and eliminating them is crucial to unveil true astrophysical events. In this study, we evaluate the performance of logistic regression, extreme gradient boost, and support vector machines for glitch classification using features extracted via transfer learning on Inception-v3 and ResNet-50 models. We used two transfer learning strategies: fine-tuning pre-trained models with our dataset and using pre-trained models as feature extractors. Our results show that transfer learning significantly reduces training time compared to fine-tuning. The transfer learning method achieved a classification accuracy of 93.98% with the lowest training time of 37.6 s. Transfer learning resulted in 24 and 31 times faster training for ResNet-50 and Inception-v3, respectively, proving highly beneficial for glitch classification in the LIGO experiment.

This chapter is based on `Bişkin O.T, Kırbaş İ, Çelik A., 2023, In: Astronomy and Computing. 42: 100683`, https://doi.org/10.1016/j.ascom.2022.100683.

O. T. Bişkin (✉)
Department of Electrical-Electronics Engineering, Burdur Mehmet Akif Ersoy University, Burdur, Turkey
e-mail: tbiskin@mehmetakif.edu.tr

İ. Kirbaş
Department of Computer Engineering, Burdur Mehmet Akif Ersoy University, Burdur, Turkey
e-mail: ismailkirbas@mehmetakif.edu.tr

A. Çelik
Department of Physics, Burdur Mehmet Akif Ersoy University, Burdur, Turkey
e-mail: alicelik@mehmetakif.edu.tr

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_6

6.1 Introduction

In 1916, Albert Einstein predicted the existence of gravitational waves, disturbances in space-time caused by massive accelerating objects like black holes and neutron stars [1]. The first proof of gravitational waves came with the GW150914 event, reported by the Laser Interferometer Gravitational Wave Observatory (LIGO) collaboration in September 2015, which detected waves from two colliding black holes 1.3 billion light-years away [2]. Since then, LIGO has recorded over 90 gravitational waves, marking significant progress in the field [3].

LIGO's interferometers, consisting of two 4 km long perpendicular arms [4, 5], are designed to detect extremely small signals resulting from gravitational waves. However, gravitational waves are small due to gravity being a weak force, their effects on space-time are smaller than the size of an atomic nucleus [6]. Consequently, LIGO's detectors must be highly sensitive, not only to gravitational waves but also to various environmental noises, instrumental disturbances, and seismic activities that can interfere with the signals. Among these interferences, non-Gaussian transient waves known as glitches can mimic gravitational wave signals and degrade the data quality. Successfully identifying and eliminating glitches is crucial to improve the accuracy of gravitational wave observations.

Machine learning (ML) and deep learning techniques have become essential tools in image classification and time series analysis over the past decade [7–16]. Specifically, convolutional neural networks (CNNs) using Q-transform and wavelet methods have been preferred for analyzing gravitational wave signals to obtain meaningful information by analysing them in time-frequency domain [17, 18]. This study aims to classify the ten most common glitch types using deep learning methods applied to images generated through Q-transform. Traditional manual methods for glitch classification are inadequate, necessitating the development of rapid and accurate automated techniques. Previous works have introduced various algorithms for glitch classification, including those based on principal component analysis, machine learning, and wavelet detection [19], showing promising results in both simulated and real data [20].

While several machine learning models [19–23] have been promising in classifying specific glitch morphologies, covering the entire range of glitch types remains a significant challenge. Human eyes are still the best classifiers, leading to the development of the Gravity Spy project on Zooniverse.org, which enlists citizen scientists to help physicists and astronomers label data. This hand-labeled dataset is then used to train machine learning algorithms to identify new, unseen glitch categories. Various glitches from numerous sources have been identified, with over twenty morphologies recognized and named based on structure and shape, as shown in Fig. 6.1. For example, "whistle" glitches are caused by radio transmissions, "blip" glitches by unknown causes, and the "tomte" class is named for its resemblance to a tomte gnome hat. Unclassified glitches are tagged as "none of the above," and a new class is created if many similar glitches appear in this category. More details on common glitch types are in [24].

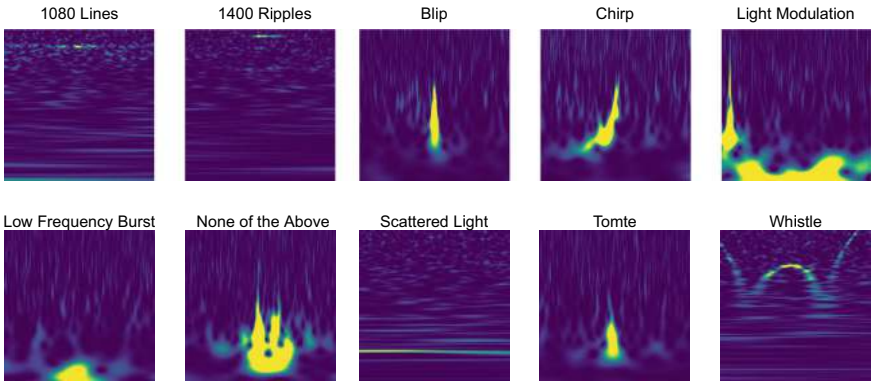


Fig. 6.1 Q-transformed image of each glitch class used in this study

Despite the success of these methods, the challenge of covering the entire range of glitch types remains. The Gravity Spy project addresses this by utilizing citizen scientists to label glitches, which are then used to train machine learning models for identifying new data categories [23]. The process requires periodic retraining of models with new data, especially after upgrades to the measurement system, which can introduce new types of glitches. Therefore, reducing training time without compromising accuracy is essential. This study focuses on using transfer learning with pre-trained deep models as feature extractors for machine learning algorithms. By comparing the classification results and training times of transfer learning and fine-tuning approaches, we demonstrate that transfer learning significantly reduces computational costs while maintaining high classification accuracy, making it a valuable approach for LIGO's glitch classification process.

6.2 Methods

In this work, we utilize deep learning models together with machine learning algorithms to classify glitches efficiently. One of the main factors that prolong training time in deep learning networks is the dense and non-freezing layers near the output layer. Therefore, we propose eliminating the training of non-freezing layers and instead using fast-trainable machine learning algorithms such as SVM, Extreme Gradient Boost (XGBoost), and logistic regression (LR).

Our strategy consists of two stages. In the first stage, pre-trained deep learning models extract features from Q-transformed images of glitches. In the second stage, we reduce the dimension of features by selecting the first 100 principal components with the highest variance using principal component analysis (PCA). Then, SVM, LR, and XGBoost algorithms are used to train the machine learning models.

6.2.1 Inception-V3 and ResNet-50 Pre-Trained Deep Learning Models

Inception-v3 is the third version of Google's Inception Convolutional Neural Network with 48 deep layers [25]. It was designed for image classification with a deeper network without increasing the number of parameters and is commonly used as a pre-trained deep neural network.

ResNet-50 is a 50-layer deep convolutional neural network proposed as a residual learning framework [26]. It addresses the problem of vanishing/exploding gradients in deeper networks by using residual connections, allowing for deeper networks without performance degradation.

Both models have been pre-trained on the ImageNet database [27], which contains over one million images and 1000 different object categories.

6.2.2 Transfer Learning and Fine-Tuning

In the literature, various transfer learning methods are employed for deep models, with definitions and taxonomies explored based on their strategies [28–30]. A commonly used method is fine-tuning, but not all transfer learning techniques utilize it. Feature extraction from pre-trained networks, as used in this study, is another approach [31]. These methods can exhibit different performance levels [32].

Transfer learning often involves using pre-trained models like Inception-v3 and ResNet-50 to extract feature vectors, which are then used to train machine learning classifiers such as SVM [33, 34]. On the other hand, the fine-tuning method, the most prevalent technique [31], involves freezing the earlier layers of pre-trained models and training the remaining layers for the new task. For a model with L layers, fine-tuning involves training the last k layers and freezing the first $L - k$ layers, with k determined by the dataset size and model depth [35]. Both methods replace the final layer of the model; however, fine-tuning re-trains the last k layers with new data, requiring more computational resources and training time, dependent on k [31].

In this study, Inception-v3 and ResNet-50 models pre-trained on the ImageNet dataset are used as feature extractors without further training. The extracted features are then used to classify glitches using machine learning algorithms, and the performance of fine-tuned networks is compared with the proposed approach.

6.2.3 PCA

PCA is a dimension reduction technique that transforms data into a lower-dimensional space [36]. Let M denote the data dimension. First, the $M \times M$ covariance matrix is calculated. Then, the eigenvalues of the covariance matrix are sorted in descending

order, and the eigenvectors corresponding to the k largest eigenvalues are selected as the principal components. The data is mapped onto the principal components using the transformation equation:

$$\bar{\mathbf{x}} = \mathbf{A}^T(\mathbf{x} - \mu)$$

Here, μ is the mean vector, $\mathbf{A}$ is the $k \times k$ transformation matrix, and T denotes the transpose operation. The columns of $\mathbf{A}$ consist of the largest k eigenvectors of the covariance matrix.

6.2.4 Constant Q-Transform (CQT)

The constant Q-transform (hereafter referred to as Q-transform), introduced by [37], is a signal transform method similar to the short-time Fourier transform (STFT). CQT transforms a signal from the time domain to the time-frequency domain. While both CQT and STFT are similar, the center frequency of the frequency bins in CQT is geometrically spaced.

Let the CQT of a signal be defined as follows [38]:

$$G(k,n) = \sum_{j=n-(N_k/2)}^{n+(N_k/2)} g(j)a_k^*(j - n + N_k/2), \tag{6.1}$$

where $g(j)$ denotes a discrete time-domain signal, and $k = 1, 2, \ldots, K$ correspond to frequency bins. In Eq. (6.1), $a_k^*(n)$ is referred to as a basis function, also known as a time-frequency atom. The symbol $(*)$ indicates the complex conjugate, meaning that $a_k^*(n)$ represents the complex conjugate of $a_k(n)$. The time-frequency atoms $a_k(n)$ are expressed as:

$$a_k(n) = \frac{1}{N_k}\omega\left(\frac{n}{N_k}\right)e^{-i2\pi n f_k/f_s}. \tag{6.2}$$

In this equation, f_k represents the center frequency of the k-th frequency bin, f_s is the sampling frequency, and $\omega(t)$ denotes a window function.

6.2.5 XGBoost

A scalable tree boosting system, XGBoost, introduced by [39], is an optimized version of the Gradient Boosting algorithm. Both of them use an ensemble of weak learners, but XGBoost utilizes parallelized boosted decision trees and is used for regression and classification problems. It quickly converges to a stationary point without over-fitting, improving the model at each iteration by correcting previous errors.

6.2.6 SVM

The statistical theory behind the support vector machine (SVM) was proposed by [40]. Let the training dataset be given as $D = \left\{(x_n, y_n) \mid x_n \in \mathbb{R}^{1\times M}, y_n \in \{-1, 1\}\right\}$ where $n = 1, \ldots, N$. Here, N is the total number of samples, and M is the feature size of each sample.

Let $x_n \in \mathbb{R}^{1\times M}$ be an individual sample, and y_n denote the label of the corresponding sample x_n. The SVM maximizes the hyperplane margin between the nearest samples of each group. The hyperplane in SVM, created by a kernel function, can be calculated by solving:

$$\begin{aligned} &\min && \|\mathbf{w}\|^2 \\ &\text{subject to} && \mathrm{y}_n\,(\mathbf{w}\cdot\mathbf{x}_n + b) - 1 \geq 0 \quad n = 1, \ldots, N, \end{aligned} \tag{6.3}$$

where b is a bias value. Minimizing $\mathbf{w}$ results in maximizing the margin between classes. Using the Lagrange multiplier, the above optimization problem can be rewritten as:

$$\begin{aligned} &\min && \frac{1}{2}\|\mathbf{w}\|^2 + \lambda \sum_{n=1}^{N} \psi_n \\ &\text{subject to} && \mathrm{y}_n\,(\mathbf{w}\cdot\mathbf{x}_n + b) \geq 1 - \psi_n \quad n = 1, \ldots, N. \end{aligned} \tag{6.4}$$

Here, ψ_n is a slack variable, and the objective function in (6.4) is minimized by reducing the sum of slack variables. λ is the regularization term adjusting the trade-off between the loss function and miss classification. The optimization problem given above can be solved using linear, sigmoid, polynomial, or radial basis function kernels.

6.2.7 LR

Logistic regression models are widely used in medical science, social sciences, and machine learning. Logistic models are based on the logistic function, initially used to describe population growth and some chemical reactions [41]. The standard logistic function is given as:

$$p(x) = \frac{1}{1 + e^{-x}}. \tag{6.5}$$

For a given sample x_n, $p_n = p(x_n)$ is the probability corresponding to the target y_n. In logistic regression, the following cost function is minimized at each iteration:

$$C = \sum_{n=1}^{N} (y_n \ln(y_n) + (1 - y_n)\ln(1 - p_n)). \tag{6.6}$$

Here, C represents the cost function.

6.2.8 *Proposed Method*

Our strategy consists of two stages, as shown schematically in Fig. 6.2. In the first stage, we extract features from images using Q-transform. Pre-trained Inception-v3 and ResNet-50 models serve as feature extractors. Feature maps are obtained from the layer before the final fully connected layer, with sizes $5 \times 5 \times 2048$ for Inception-v3 and $7 \times 7 \times 2048$ for ResNet-50. These feature maps are then flattened into vectors $f_I \in \mathbb{R}^{1\times 51200}$ and $f_R \in \mathbb{R}^{1\times 100352}$, respectively, representing the features extracted by the models.

In the next stage, we reduce the dimension of the features using Principal Component Analysis. The extracted features are transformed into the principal components of the feature space. Using the scikit-learn library [43], we reduce the dimensions to the first 100 components with the highest variance, resulting in feature vectors $f_{PCA} \in \mathbb{R}^{1\times 100}$. This selection helps decrease the feature dimension and consequently reduces the training time for the machine learning algorithms. Figure 6.3 shows the distribution of the training data on the first four principal components with the highest explained variance ratio (EVR).

At the final stage, the reduced feature set $f_{PCA} \in \mathbb{R}^{1\times 100}$ is fed into ML algorithms to build classification models. The hyperparameters of the ML models are optimized

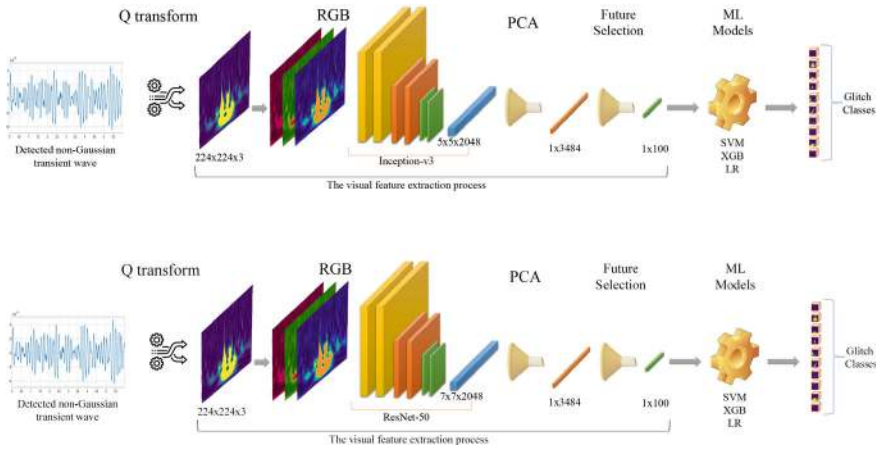


Fig. 6.2 Schematic diagram of deep learning-based glitch classification pipeline. Figure from [42]

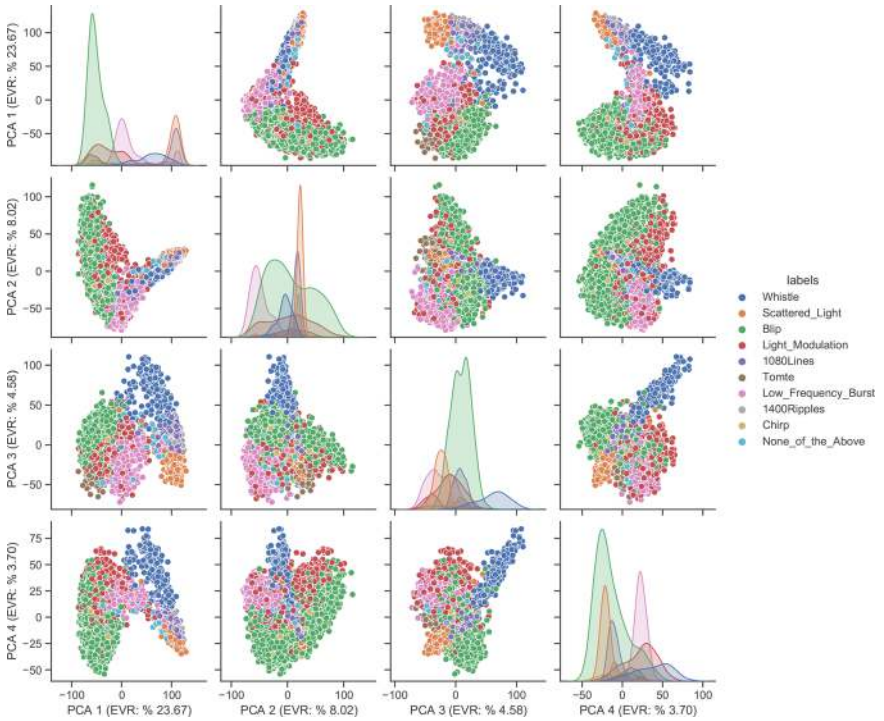


Fig. 6.3 Distribution of training data mapped on the principal components

using the Optuna library [44], an open-source hyperparameter optimization toolbox. The Optuna library combines efficient searching and pruning algorithms to improve the cost-effectiveness of optimization. In our experiments, ML models are optimized with Optuna to ensure they perform under optimal conditions. For this study, the kernel of the SVM and the number of estimators, maximum depth, and learning rate of the XGBoost algorithm are determined using the Optuna framework. We employed linear, sigmoid, and polynomial kernels for SVM, finding that the sigmoid kernel provided the best classification performance.

6.2.9 Dataset

The dataset is sourced from the Gravity Spy project on Zooniverse.org, containing ten classes of glitches. Formally, it is represented as $D = \{(x_n, y_n) \mid x_n \in \mathbb{R}^{1\times M}, y_n \in \{1, \ldots, 10\}\}$, with $M = 10$ classes and $N = 4318$ total samples. The dataset used in this study exhibits data imbalance, which is a significant factor that can substantially diminish a model's performance. Two common approaches to addressing imbalance are down-sampling, which reduces the sample size of each class to the

smallest class, and up-sampling, which generates synthetic data [45]. However, in this study, neither approach was applied because down-sampling would lead to a loss of data and up-sampling would necessitate generating new data.

We split the dataset into 70% for training, 10% for validation, and 20% for testing, with samples randomly selected from each class. This process was repeated five times to average the performance results. The samples, in the form of 224 × 224 images, were created using Q-transform on time-series signals.

6.3 Simulation Results

Simulations were run on a PC with an i7-9750H CPU (2.6 GHz, 16 GB RAM) and an NVIDIA GTX 1660 Ti GPU (6 GB). The models were trained using NVIDIA CUDA 10.1 Toolkit with cuDNN v7.6.5 on Windows 10, employing TensorFlow v2.3.0 and Keras v2.4.3 for compatibility.

The first stage involved feature extraction using pre-trained models. PCA then reduced the feature dimensions. We evaluated ML algorithms based on training time and accuracy, selecting 100 principal components to balance training time and accuracy. Thus, the 3454 principal components were reduced to 100 using feature selection (FS).

Tables 6.1, 6.2, 6.3 and 6.4 summarize the results. Tables 6.1 and 6.2 show training time, test time, and accuracy for fine-tuning and transfer learning strategies using Inception-v3 and ResNet-50 models. Tables 6.3 and 6.4 provide precision, recall, and F1-scores. Initially, the models are fine-tuned using our dataset. The results of the ML algorithms in Tables 6.1 and 6.2 are derived from features extracted using the Inception-v3 and ResNet-50 models, respectively. These results are given in the section titled "fine-tuning". In the section titled "visual feature extracted using transfer learning," we demonstrate our proposed method of extracting features from the Inception-v3 and ResNet-50 models. The features obtained from these pre-trained models are then utilized by machine learning algorithms, with their classification performances detailed in Tables 6.3 and 6.4. This approach allows for a comparison of machine learning algorithm performance using features from fine-tuning and transfer learning strategies separately.

As illustrated in Table 6.1, the Inception-v3+PCA+FS+SVM, Inception-v3+ PCA + FS+XGB, and Inception-v3+PCA+FS+LR models outperform other models in terms of training time. Additionally, the LR model exhibits higher classification accuracy compared to the others. Reducing the number of principal components through feature selection significantly decreases training time. The same feature extraction method used with Inception-v3 is also applied to the ResNet-50 model for glitch classification. Table 6.2 presents the results of ML algorithms using features extracted by ResNet-50. Consistent with the results in Table 6.1, applying PCA followed by feature selection enhances ML algorithms' performance regarding training time. All three ML algorithms train faster with features extracted by the Inception-v3 model.

Table 6.1 Train time, test time and accuracy of models trained by image-based features extracted by utilizing Inception-v3 model

		Train Time (s)	Test Time (s)	Test ACC (%)	F1-Score
Fine-tuning	Inception-v3	1166.000	3.7040	95.72	0.9322
	Inception-v3+PCA+SVM	11231.3087	16.2374	95.49	0.9326
	Inception-v3+PCA+XGB	4123.4387	3.9452	90.05	0.8096
	Inception-v3+PCA+LR	1188.5208	3.7648	96.06	0.9329
	Inception-v3+PCA+FS+SVM	1188.7495	3.7933	93.75	0.9018
	Inception-v3+PCA+FS+XGB	1198.3370	3.7302	90.39	0.8252
	Inception-v3+PCA+FS+LR	1191.7931	3.7127	95.72	0.9344
Visual feature extracted using transfer learning	Inception-v3	–	3.4360	–	–
	Inception-v3+PCA+SVM	91.9230	21.3061	94.68	0.9132
	Inception-v3+PCA+XGB	142.1751	3.6534	85.07	0.7235
	Inception-v3+PCA+LR	61.4758	3.5759	95.72	0.9291
	Inception-v3+PCA+FS+SVM	37.7900	3.5206	92.25	0.8844
	Inception-v3+PCA+FS+XGB	37.8293	3.4470	85.53	0.7387
	Inception-v3+PCA+FS+LR	37.6145	3.4516	93.98	0.9111

Table 6.2 Train time, test time, and accuracy of models trained by image-based features extracted by utilizing ResNet-50 model

		Train Time (s)	Test Time (s)	Test ACC (%)	F1-Score
Fine-tuning	ResNet-50	989	5.1550	96.06	0.9250
	ResNet-50+PCA+SVM	1216.5736	54.0941	93.87	0.8861
	ResNet-50+PCA+XGB	1414.9887	5.5709	87.50	0.7903
	ResNet-50+PCA+LR	1054.3457	5.4353	94.79	0.9015
	ResNet-50+PCA+FS+SVM	1010.7211	5.2338	91.90	0.8627
	ResNet-50+PCA+FS+XGB	1012.7118	5.1660	87.50	0.7383
	ResNet-50+PCA+FS+LR	1010.6343	5.1620	93.40	0.8867
Visual feature extracted using transfer learning	ResNet-50	–	5.1550	–	–
	ResNet-50+PCA+SVM	250.0836	56.3033	92.36	0.8146
	ResNet-50+PCA+XGB	1213.4106	5.5609	86.46	0.7721
	ResNet-50+PCA+LR	4579.2933	5.2573	93.06	0.8625
	ResNet-50+PCA+FS+SVM	39.3270	5.2408	89.24	0.7837
	ResNet-50+PCA+FS+XGB	44.7894	5.1680	89.35	0.7950
	ResNet-50+PCA+FS+LR	42.3449	5.1600	92.13	0.8487

Table 6.3 Classification performances of ML algorithms on test set by using image based features extracted by employing pretrained Inception-v3 model

		1080 lines	1400 ripples	Blip	Chirp	Light mod.	Low freq. burst	None of the above	Scat. light	Tomte	Whistle
Inception-v3+ PCA+SVM	Prec.	92.50	87.50	95.25	100.00	92.41	090.40	100.00	100.00	93.75	100.00
	Rec.	96.10	100.00	99.45	78.57	82.95	95.76	58.82	98.81	93.75	86.36
	F1-Sc.	94.27	93.33	97.30	88.00	87.43	93.00	74.07	99.40	93.75	92.68
Inception-v3+ PCA+XGB	Prec.	92.54	65.38	90.77	100.00	53.98	87.07	40.00	92.13	90.91	96.23
	Rec.	80.52	80.95	94.77	35.71	69.32	85.59	11.76	97.62	62.50	77.27
	F1-Sc.	86.11	72.34	92.72	52.63	60.70	86.32	18.18	94.80	74.07	85.71
Inception-v3+ PCA+LR	Prec.	98.68	91.30	97.28	100.00	86.81	93.28	78.57	98.81	100.00	100.00
	Rec.	97.40	100.00	98.62	85.71	89.77	94.07	64.71	98.81	93.75	93.94
	F1-Sc.	98.04	95.45	97.95	92.31	88.27	93.67	70.97	98.81	96.77	96.88
Inception-v3+ PCA+FS+SVM	Prec.	88.89	90.00	94.95	100.00	79.35	87.90	90.00	100.00	93.75	98.11
	Rec.	93.51	85.71	98.35	85.71	82.95	92.37	52.94	95.24	93.75	78.79
	F1-Sc.	91.14	87.80	96.62	92.31	81.11	90.08	66.67	97.56	93.75	87.39
Inception-v3+ PCA+FS+XGB	Prec.	88.75	93.75	88.35	62.50	66.67	81.75	46.15	97.59	66.67	92.45
	Rec.	92.21	71.43	96.14	35.71	56.82	87.29	35.29	9643	62.50	74.24
	F1-Sc.	90.45	81.08	92.08	45.45	61.35	84.43	40.00	97.01	64.52	82.35
Inception-v3+ PCA+FS+LR	Prec.	98.63	87.50	96.47	100.00	83.33	91.53	78.57	98.78	88.24	93.85
	Rec.	93.51	100.00	97.80	92.86	85.23	91.53	64.71	96.43	93.75	92.42
	F1-Sc.	96.00	93.33	97.13	96.30	84.27	91.53	70.97	97.59	90.91	93.13

Table 6.4 Classification performances of ML algorithms on test set by using image based features extracted by employing pretrained ResNet-50 model

		1080 lines	1400 ripples	Blip	Chirp	Light mod.	Low freq. burst	None of the above	Scat. light	Tomte	Whistle
ResNet-50+ PCA+SVM	Prec.	81.11	85.71	94.69	100.00	93.67	90.24	66.67	96.47	85.71	98.31
	Rec.	94.81	57.14	98.35	78.57	84.09	94.07	47.06	97.62	75.00	87.88
	F1-Sc.	87.43	68.57	96.49	88.00	88.62	92.12	55.17	97.04	80.00	92.80
ResNet-50+ PCA+XGB	Prec.	90.14	78.95	94.05	100.00	58.56	83.33	37.50	93.02	83.33	90.16
	Rec.	83.12	71.43	95.87	85.71	73.86	80.51	17.65	95.24	62.50	83.33
	F1-Sc.	86.49	75.00	94.95	92.31	65.33	81.90	24.00	94.12	71.43	86.61
ResNet-50+ PCA+LR	Prec.	91.14	79.17	95.71	100.00	90.36	89.52	63.64	95.35	92.31	96.67
	Rec.	93.51	90.48	98.35	78.57	85.23	94.07	41.18	97.62	75.00	87.88
	F1-Sc.	92.31	84.44	97.01	88.00	87.72	91.74	50.00	96.47	82.76	92.06
ResNet-50+ PCA+FS+SVM	Prec.	72.73	66.67	92.99	100.00	79.12	91.53	66.67	98.78	92.31	95.74
	Rec.	93.51	28.57	98.62	78.57	81.82	91.53	35.29	96.43	75.00	68.18
	F1-Sc.	81.82	40.00	95.72	88.00	80.45	91.53	46.15	97.59	82.76	79.65
ResNet-50+ PCA+FS+XGB	Prec.	78.89	91.67	94.16	100.00	79.75	86.07	38.46	97.62	75.00	95.00
	Rec.	92.21	52.38	97.80	78.57	71.59	88.98	29.41	97.62	75.00	86.36
	F1-Sc.	85.03	66.67	95.95	88.00	75.45	87.50	33.33	97.62	75.00	90.48
ResNet-50+ PCA+FS+LR	Prec.	85.06	70.00	95.71	100.00	87.36	92.17	56.25	95.35	92.31	98.18
	Rec.	96.10	66.67	98.35	85.71	86.36	89.83	52.94	97.62	75.00	81.82
	F1-Sc.	90.24	68.29	97.01	92.31	86.86	90.99	54.55	96.47	82.76	89.26

When evaluating the test times of the models, it is noted that the SVM model runs relatively slower when the number of input parameters exceeds 3400. However, after the feature selection process reduces the input parameters to 100, the test times of the models converge and fall below 4 s.

Hyperparameters of the ResNet-50 and Inception-v3 models are separately calculated for fine-tuned and transfer learning strategies. In order to have a fair comparison, we ran all models up to 100 epochs. We also set the "early stopping" criterion if there is no improvement in the validation accuracy after ten epochs. We added a fully connected layer to the last layer for fine-tuning. Also, when fine-tuning, we froze the first 275 layers in Inception-v3 and the first 185 layers in ResNet. We adjusted the number of frozen layers by trial and picked the optimum number of frozen layers based on the highest accuracy score at the end of every trial. The learning rate of models is selected as 0.0001. We optimized the ML models using Optuna Library.

6.4 Conclusion

This study explores methodologies for classifying glitches across ten different classes, evaluating training time and accuracy for each. Machine learning models using LR, XGBoost, and SVM algorithms follow two feature extraction methods with pre-trained Inception-v3 and ResNet-50 models. The first method fine-tunes pre-trained models using our dataset, while the second uses them as feature extractors. Features are reduced from 51200 to 100 to accelerate training while preserving variance. Transfer learning significantly reduces training time compared to fine-tuning, offering 24 and 31 times faster training with ResNet-50 and Inception-v3, respectively. Despite dataset imbalance, logistic regression achieved a classification accuracy of 93.98% with minimal training time.

References

1. Einstein, A.: Approximative integration of the field equations of gravitation. Sitzungsber. Preuss. Akad. Wiss. Berlin (Math. Phys.) **1916**(688–696), 1 (1916)
2. Abbott, B.P., Abbott, R., Abbott, T.D., Abernathy, M.R., Acernese, F., et al.: Observation of gravitational waves from a binary black hole merger. Phys. Rev. Lett. **116**(6), 061102 (2016). https://doi.org/10.1103/PhysRevLett.116.061102
3. LSC News. https://www.ligo.org/news/index.php#GWTC3-TGRwebinar. Accessed 21 Apr 2022
4. Aasi, J., Abbott, B.P., Abbott, R., Abbott, T., Abernathy, M.R., et al.: Advanced LIGO. Class. Quantum Grav. **32**(7), 074001 (2015)
5. Harry, G.M.: LIGO Scientific Collaboration. Advanced LIGO: the next generation of gravitational wave detectors. Class. Quantum Grav. **27**(8), 084006 (2010)
6. Abbott, B.P., Abbott, R., Abbott, T.D., Abernathy, M.R., Acernese, F., et al.: GW150914: the advanced LIGO detectors in the era of first discoveries. Phys. Rev. Lett. **116**(13), 131103 (2016)
7. Kırbaş, İ, Kerem, A.: A new vibration-based hybrid anomaly detection model for preventing high-power generator failures in power plants. Energy Sources Part A Recover. Util. Environ. Eff. **43**(23), 3184–3202 (2021). https://doi.org/10.1080/15567036.2021.1960654

8. Kırbaş, İ, Özmen, Ö.: Classification of canine fibroma and fibrosarcoma histopathological images using convolutional neural networks. In: Kose, U., Alzubi, J. (eds.) Deep Learning for Cancer Diagnosis, pp. 67–77. Springer Singapore, Singapore (2021)
9. Morawski, F., Bejger, M., Cuoco, E., Petre, L.: Anomaly detection in gravitational waves data using convolutional autoencoders. Mach. Learn. Sci. Technol. **2**(4), 045014 (2021)
10. Gabbard, H., Messenger, C., Heng, I.S., Tonolini, F., Murray-Smith, R.: Bayesian parameter estimation using conditional variational autoencoders for gravitational-wave astronomy. Nat. Phys. **18**(1), 112–7 (2022)
11. Cuoco, E., Powell, J., Cavaglià, M., Ackley, K., Bejger, M., et al.: Enhancing gravitational-wave science with machine learning. Mach. Learn. Sci. Technol. **2**(1), 011002 (2020)
12. Gabbard, H., Messenger, C., Heng, I.S., Tonolini, F., Murray-Smith, R.: Bayesian parameter estimation using conditional variational autoencoders for gravitational-wave astronomy. Nat. Phys. **18**(1), 112–7 (2022)
13. Kırbaş, İ.: Investigation of predictive performance of LSTM artificial neural networks on Brownian time series. In: 5th International Conference On Engineering And Natural Science (ICENS) Book of Proceedings, Prague, Czech Republic, pp. 105–112 (2019)
14. Kırbaş, İ.: NAR based forecasting interface for time series analysis: T-seer. In: IV International Conference on Engineering and Natural Science (ICENS), Kiev, Ukraine, pp. 144–149 (2018)
15. Dükkancı, A., Kırbaş, İ.: Rolling bearing content failure classification using machine learning algorithms. In: Proceedings on 2nd International Conference on Technology and Science, Burdur, pp. 235–239 (2019)
16. Kırbaş, İ, Peker, M.: Signal detection based on empirical mode decomposition and Teager-Kaiser energy operator and its application to P and S wave arrival time detection in seismic signal analysis. Neural Comput. Appl. **28**(10), 3035–3045 (2017)
17. George, D., Shen, H., Huerta, E.A.: Deep transfer learning: a new deep learning glitch classification method for advanced LIGO. Phys. Rev. D **97**(10), 101501 (2018)
18. Cuoco, E., Razzano, M., Utina, A.: Wavelet-based classification of transient signals for gravitational wave detectors. In: 26th European Signal Processing Conference (EUSIPCO), Rome, pp. 2648–2652 (2018)
19. Powell, J., Trifirò, D., Cuoco, E., Heng, I.S., Cavaglià, M.: Classification methods for noise transients in advanced gravitational-wave detectors. Class. Quantum Grav. **32**(21), 215012 (2015)
20. Powell, J., Torres-Forné, A., Lynch, R., Trifirò, D., Cuoco, E., et al.: Classification methods for noise transients in advanced gravitational-wave detectors II: performance tests on advanced LIGO data. Class. Quantum Grav. **34**, 034002 (2017). https://doi.org/10.1088/1361-6382/34/3/034002
21. Mukherjee, S., Obaid, R., Matkarimov, B.: Classification of glitch waveforms in gravitational wave detector characterization. J. Phys. Conf. Ser. **243**, 012006 (2010)
22. Rampone, S., Pierro, V., Troiano, L., Pinto, I.M.: Neural network aided glitch-burst discrimination and glitch classification. Int. J. Mod. Phys. C **24**(11), 1350084 (2013)
23. Mukund, N., Abraham, S., Kandhasamy, S., Mitra, S., Philip, N.S.: Transient classification in LIGO data using difference boosting neural network. Phys. Rev. D **95**(10), 104059 (2017)
24. LIGO-G1500642-v23: aLIGO Glitch Classes Seen So Far. https://dcc.ligo.org/LIGO-G1500642-v23/public. Accessed 21 Apr 2022
25. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)
26. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
27. ImageNet. https://www.image-net.org/. Accessed 25 July 2022
28. Iman, M., Rasheed, K., Arabnia, H.R.: A review of deep transfer learning and recent advancements. arXiv:2022:2201.09679

29. Celik, A.: A fast and time-efficient machine learning approach to dark matter searches in compressed mass scenario. Eur. Phys. J. C **83**(12), 1–16 (2023)
30. Celik, A.: Exploring hidden signal: fine-tuning ResNet-50 for dark matter detection. Comput. Phys. Commun. **305**, 109348 (2024)
31. Falconi, L.G., Perez, M., Aguilar, W.G., Conci, A.: Transfer learning and fine tuning in breast mammogram abnormalities classification on CBIS-DDSM database. Adv. Sci. Technol. Eng. Syst. J. **5**(2), 154–165 (2020)
32. Guan, S., Loew, M.: Breast cancer detection using transfer learning in convolutional neural networks. In: 2017 IEEE Applied Imagery Pattern Recognition Workshop (AIPR), pp. 1–8 (2017)
33. Huynh, B.Q., Li, H., Giger, M.L.: Digital mammographic tumor classification using transfer learning from deep convolutional neural networks. J. Med. Imaging **3**(3), 034501 (2016)
34. Perre, A.C., Alexandre, L.A., Freire, L.C.: Lesion classification in mammograms using convolutional neural networks and transfer learning. Comput. Methods Biomech. Biomed. Eng. Imaging & Vis. **7**(5), 550–556 (2018)
35. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? Adv. Neural Inf. Process. Syst. **27**, 3320–3328 (2014)
36. Jolliffe, I.T.: Principal Component Analysis, 2nd edn. Springer Series in Statistics. Springer-Verlag, New York (2002)
37. Brown, J.C.: Calculation of a constant Q spectral transform. J. Acoust. Soc. Am. **89**(1), 425–34 (1991)
38. Schörkhuber, C., Klapuri, A.: Constant-Q transform toolbox for music processing. In: 7th Sound and Music Computing Conference, Barcelona, Spain, pp. 3–64 (2010)
39. Chen, T., Guestrin, C. Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794 (2016)
40. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–97 (1995)
41. Cramer, J.S.: The origins of logistic regression (December 2002). Tinbergen Institute Working Paper No. 2002-119/4, Available at SSRN: https://ssrn.com/abstract=360300 or https://doi.org/10.2139/ssrn.360300
42. Bişkin, O.T., Kırbaş, İ, Çelik, A.: A fast and time-efficient glitch classification method: a deep learning-based visual feature extractor for machine learning algorithms. Astron. Comput. **42**, 100683 (2023)
43. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., et al.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
44. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: a next-generation hyperparameter optimization framework. In: Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 2623–2631 (2019)
45. Guillaume, L., Nogueira, F., Christos, K.A.: Imbalanced-learn: a python toolbox to tackle the curse of imbalanced datasets in machine learning. J. Mach. Learn. Res. **18**(1), 559–563 (2017)

Part II
Machine Learning Application for Gravitational Wave Detector Study and Control Systems

The accurate detection and analysis of gravitational wave signals pose significant challenges, primarily due to the pervasive noise and the need for highly sensitive and precise control systems in gravitational wave detectors. Integrating machine learning techniques into the study and control of gravitational wave detectors could improve the sensitivity, reliability, and overall performance of these sophisticated instruments. Machine learning algorithms, with their ability to learn and adapt to complex data patterns, are ideally suited to address the many problems associated with gravitational wave detection. Gravitational wave detectors, such as LIGO and Virgo, are susceptible to various noise sources, including seismic activity, thermal fluctuations, and instrumental imperfections. Machine learning techniques can be employed to model and predict these noise sources, enabling more effective noise reduction strategies. Maintaining the optimal operation of gravitational wave detectors involves complex control systems that adjust various parameters in real time. Machine learning can improve these control systems by predicting necessary adjustments and automating responses to environmental changes and internal system fluctuations. This leads to more stable and sensitive detectors, capable of operating at peak performance for extended periods.

Significant effort and research are being dedicated to developing algorithms aimed at suppressing noise in real time, using data from external sensors. A notable example of this is the suppression of Newtonian noise, which arises from ground vibrations and environmental factors that can interfere with gravitational wave detectors. By integrating advanced sensor networks and real-time monitoring systems, these algorithms are designed to accurately identify and filter out unwanted noise, improving the sensitivity and precision of gravitational wave measurements. The ongoing study of such techniques holds great promise for enhancing the overall performance of gravitational wave observatories.

In this collection of chapters, we explore the latest advances and applications of machine learning in this field.

Chapter 7
AI-Powered Charge Monitoring in Interferometers

Federico Armato and Andrea Chincarini

Abstract The sensitivity of gravitational waves interferometers is limited by numerous sources of noise. One such source arises from the charge deposition on the test mass, which interacts with the surrounding electrical field, introducing an undesired non-gravitational force on the test mass. Unfortunately, very little is known about the charge deposition and its dynamics, so that this noise is among the least modeled in the detector. In this chapter, we illustrate a procedure for implementing an effective and non-invasive charge monitoring system capable of extracting the maximum amount of information with minimal disturbance. Specifically, we explain the proposed methodology and compare different possible criteria in the choice of the monitoring system.

7.1 Introduction

The sensitivity of gravitational waves (GWs) detectors is influenced by multiple sources of noise. One such source is associated with the deposition of charge on the test masses (TMs) [1].

This issue arises as the deposited charge interacts with the surrounding electrical fields, introducing an undesired non-gravitational force on the TM.

Fluctuations attributed to a charge on the TMs have been observed in both the Virgo interferometer and the LIGO [2] and GEO [3] detectors, where charge accumulation on the mirrors has been identified.

In particular, in Virgo, at the end of O3 preparation phase, it was discovered that TMs exhibited electrical charging, with surface density values on the order of several

F. Armato (✉)
UNIGE, Department of Physics, Via Dodecaneso 33, 16146 Genova, Italy
e-mail: federico.armato@edu.unige.it

A. Chincarini
INFN, Sezione Genova, Via Dodecaneso 33, 16146 Genova, Italy
e-mail: andrea.chincarini@ge.infn.it

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_7

tens of pC/cm^2. Furthermore, the charge distribution was non-uniform across the dielectric surface of the mirrors [4].

The charge accumulation has historically never been consistently monitored and the charge dynamics during interferometer operations is unknown. Hence we propose to address this problem with a minimally-invasive array of sensors, optimally configured to monitor some physical characteristics of the charge on the TMs [5].

7.2 Materials and Strategies

In our analysis we considered the Test Mass (TM) and the payload (PAY).

The payload is the last stage of the TM suspension, located in the lower part of the tower. This area houses essential components such as large baffles (LBs), ring heaters (RHs) and precision controls, specifically the coil actuators [6].

The significance of taking into account the payload stems from its proximity to the Test Mass: the PAY is the structure surrounding the TM, thus representing the element most likely to be closely coupled with the charge on the TM (Fig. 7.1).

The payload structure and the TM can be simulated with a finite-element (FE) software to get the electrical field distribution given an arbitrary charge configuration. Simulations enable us to determine the potentials and the forces generated by charge distributions on the test mass and the surrounding field.

Our objective is to solve the reverse process, that is to ascertain the charge distribution on the TM given the electrical fields measured in an arbitrary set of coordinates outside the TM, as illustrated in Fig. 7.2.

This task is intricately challenging, both because of the intricacy of the payload structure, and because in practice there are significant constraints on the sensing coordinates, where in the experimental realization one would place field sensors. Indeed, part of our research aim to understand how to optimize sensors locations.

The combination of these spatial limitations, coupled with the heightened sensitivity of the interferometer, underscores the imperative for adopting minimally invasive solutions.

Therefore the primary objective of our endeavor is twofold:

- Develop a methodology for deriving the charge configurations on the test mass based on the measured fields or their potentials.
- Establish criteria for selecting sensor locations, ensuring optimal placement within spatial constraints.

The simulations were conducted using COMSOL Multiphysics 6.0. Specifically, we utilized the COMSOL Multiphysics CAD Import Module to import the Virgo North End (NE) payload, provided by the Virgo Collaboration. The PAY was simplified to facilitate the simulations (Fig. 7.3).

Furthermore, the COMSOL Multiphysics AC/DC Module was employed to simulate charge distributions on the test mass and to measure the potential generated by these charge distributions.

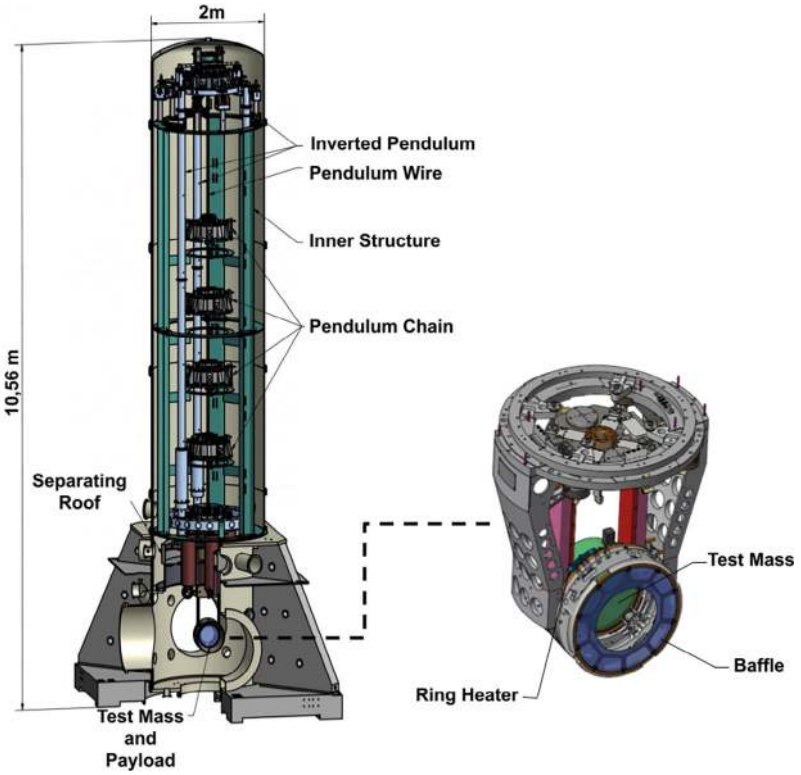


Fig. 7.1 CAD drawings of the Advanced Virgo payload integrated in the Super Attenuator suspension. Image provided by the European Gravitational Observatory (EGO)

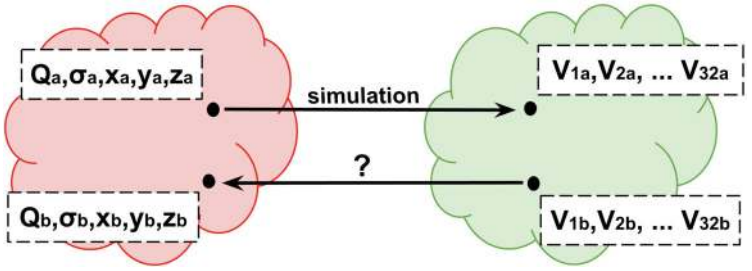


Fig. 7.2 Using simulation, we move from the charge configuration space to the measurement space, but we need a way to determine the charge configuration given the potential

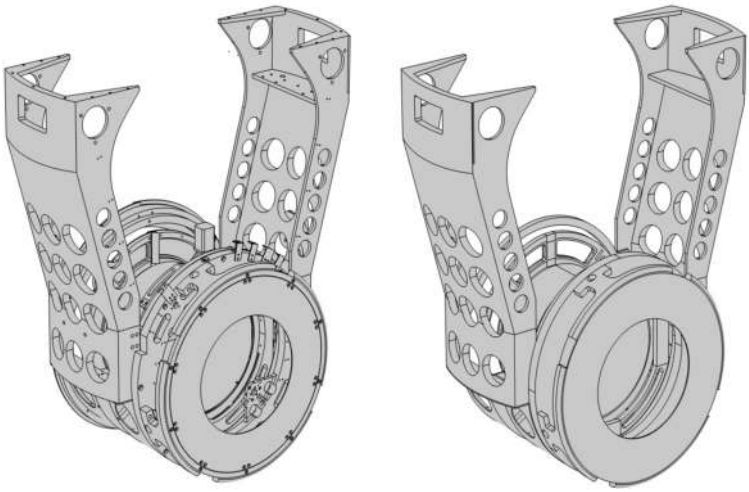


Fig. 7.3 On the left, the original NE payload model. On the right the simplified version

We employed MATLAB R2023a to analyze the simulated data and implement the problem solution.

7.3 Methods

We examined the impact of a single Gaussian-distributed charge.

Specifically we explored 32 sensor locations candidates on the PAY (Fig. 7.4) and we simulated more than 100,000 different Gaussian distributions on the TM (Fig. 7.5), where we had the flexibility to choose the parameters: intensity, standard deviation and position (Appendix 1).

We exploited Neural Networks (NNs) trained on the simulations to determine the charge value from the measured potentials (Fig. 7.6).

To maximize information while minimizing the number of sensors, we implemented sensor selection criteria.

Two methods exploit the Principal Component Analysis (PCA), which is a dimensionality reduction technique where the data are linearly transformed onto a new coordinate system, called Principal Components (PCs).

The features we are interested in are briefly summarized as follows: the PCs of a set of n variables are linear combinations of the original variables and they are ranked by variance explained on the data [7].

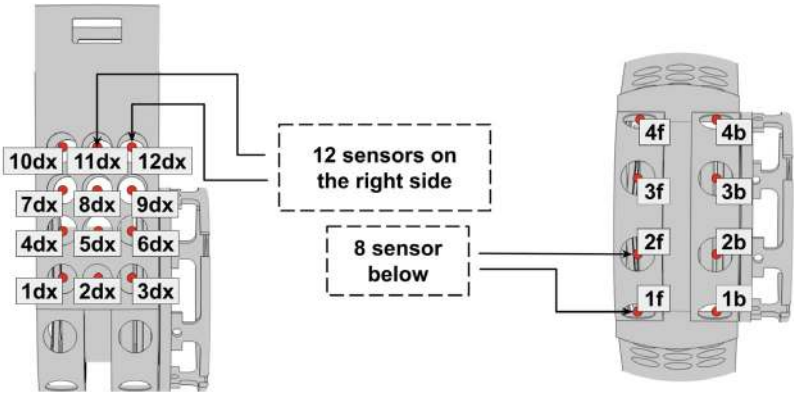


Fig. 7.4 In red the potential sensors locations in the NE PAY. The sensors on the left of the TM are mirrored counterparts of those on the right shown in picture

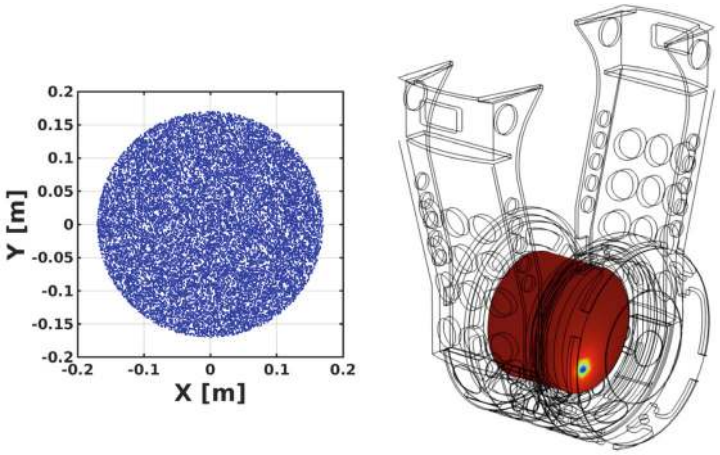


Fig. 7.5 On the left, the simulated Gaussian positions on the front test mass face. On the right, the simulated system with one of the possible charge location

- **Importance Method**

  The rationale behind the Importance criterion is to keep the most informative PCs. To reach this goal the sensors are ranked according to their contribution to the PCs weighted on their percentage of variance explanation.
- **Loadmax Method**

  The rationale behind the Loadmax Criterion is to preserve as many Principal Components as possible, to accomplish this we retain the most significant sensor for each PC.

  Especially, in selecting the first sensor, we prioritize the one that contributes the most to the first PC. Similarly, for the second sensor, priority is given to the one that contributes the most to the second PC, and so on.

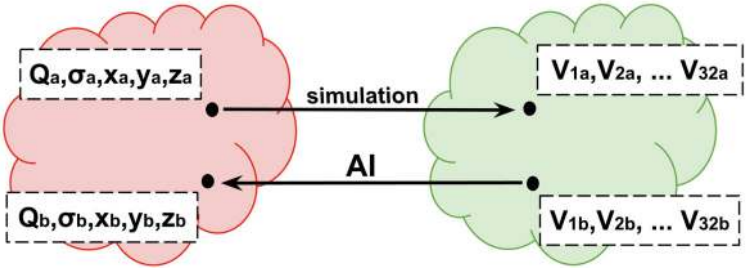


Fig. 7.6 Using simulation, we move from the charge configuration space to the measurement space; conversely, using neural networks, we can determine the charge configuration given the potential values on the sensors

A third criterion is the **Mincorr Method**, in which we choose the first sensor based on the Importance criterion. Subsequently, the second sensor chosen is the least correlated, followed by selecting the third sensor as the least correlated with the previous two, and so forth. The concept driving the Mincorr criterion is to limit information redundancy.

These criteria represent just a subset of possible methods. It is crucial to emphasize that the selection of sensors, and consequently the criterion employed, is intricately tied to the specific information we aim to acquire. For instance, certain sensors may excel in determining charge value but perform poorly when ascertaining charge position.

Furthermore, the existence of noise renders it challenging for the sensor system to discover the precise charge configuration. Additionally, even in the absence of noise, the constrained number of sensors precludes the possibility of achieving such a reconstruction.

As a result, we have opted for a simplified representation of positional information in binary form. Specifically, we aimed to determine whether the charge was situated on the front or back face, the upper or lower part, and the right or left side of the TM faces, as illustrated in Fig. 7.7.

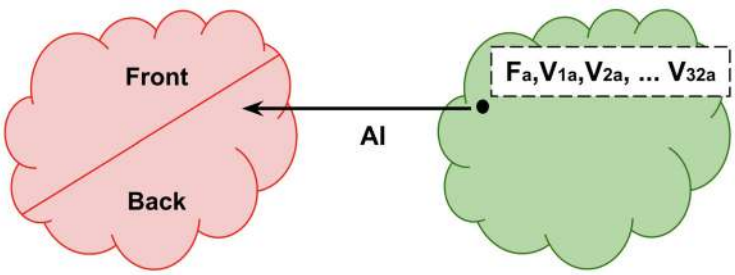


Fig. 7.7 The positional information of the charge is simplified into binary discriminations, such as whether the charge is located on the front or back face of the TM

7.4 Results

Different sensor rankings emerge for each method. In particular, Table 7.1 illustrates the comparison between the Importance, Loadmax and Mincorr criteria.

We used different neural networks (Appendix 3) to determine the charge and standard deviation value (Fig. 7.8). Specifically for each quantity we considered NNs at varying Signal to Noise Ratios (SNR) and we trained them solely on the potentials measured by the top four most informative sensors based on each criterion: Importance, Loadmax, and Mincorr.

Table 7.1 Sensor ranking from the best sensor to the worst according to Importance, Loadmax and Mincorr criteria. Complete table in Appendix 2

Sensor ranking		
Importance	Loadmax	Mincorr
3f	1b	3f
2f	4f	10dx
2dx	1f	10sx
2sx	3dx	11dx
5sx	2f	11sx
1dx	5sx	12dx
5dx	3b	12sx
1sx	12dx	7sx
1f	1dx	4b
4f	4dx	7dx
...	...	...

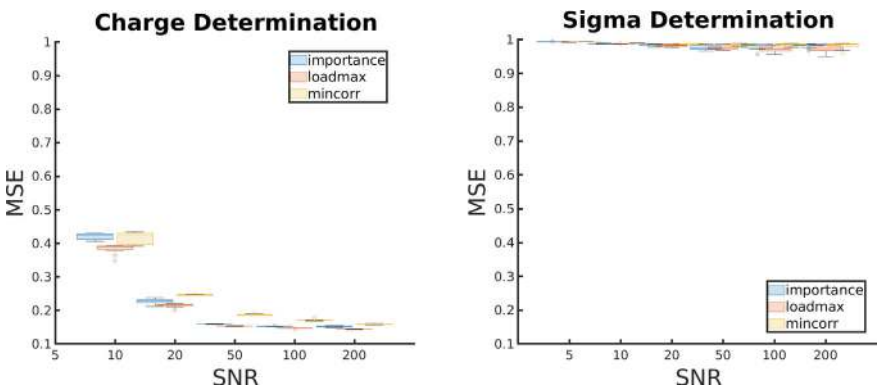


Fig. 7.8 MSE in charge value and standard deviation determination for different Neural Networks at varying SNR, where the input data have been z-score normalized (data in Appendix 4)

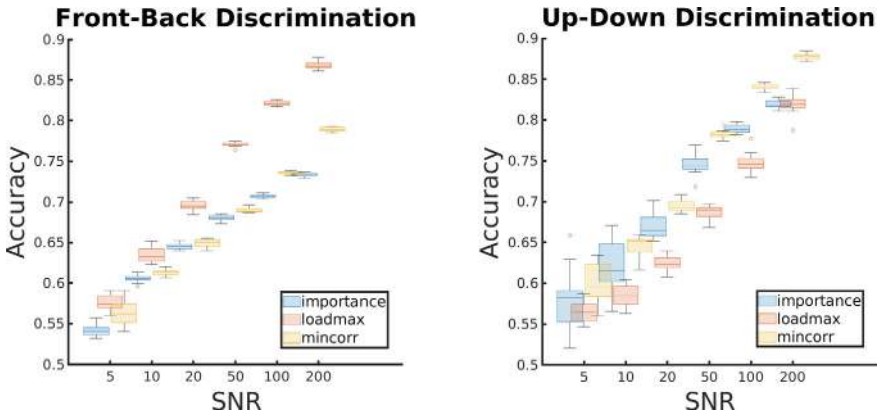


Fig. 7.9 Accuracy in front-back and up-down discrimination for different neural networks at varying SNR (data in Appendix 4)

Similarly, categorical neural networks (Appendix 3) were trained to discern three key aspects: the placement of the charge on either the front or back, the upper or lower part, the right or left part of the test mass face (Fig. 7.9).

The SNR is defined with respect to the highest potential measured by the most sensitive sensor among the 32 sensors.

We can observe that standard deviation is poorly determined, no matter which criterion is employed; while the location accuracy is strongly dependent on the specific neural network employed.

For instance, the NNs trained on the four sensors preferred by the Loadmax criterion, excel in front-back discrimination, but show reduced effectiveness in other tasks like up-down discrimination. Therefore, our criterion choice depends on which aspect is more crucial.

To clarify this point we compared the efficiency of neural networks trained using only one sensor. The idea was to understand which sensor brings alone more information and which type.

The results are shown in Fig. 7.10, where it becomes evident how different sensors provide diverse information. For example, sensors like 2dx and 3dx excel in right-left discrimination but perform poorly in up-down discrimination. In contrast, sensors like 12dx and 12sx exhibit excellence in up-down discrimination but lack proficiency in right-left discrimination. Lastly, sensors like 4sx and 4dx demonstrate an intermediate capability in both aspects.

As a practical example, let us consider the situation in which we have only four sensors and we want to understand where to locate them in order to get the best result.

To make this choice, first of all we have to understand which physical quantity is more significant, which is strongly linked to the discharging method we want to employ. Indeed, techniques like high-vacuum electron gun [8] need to know the

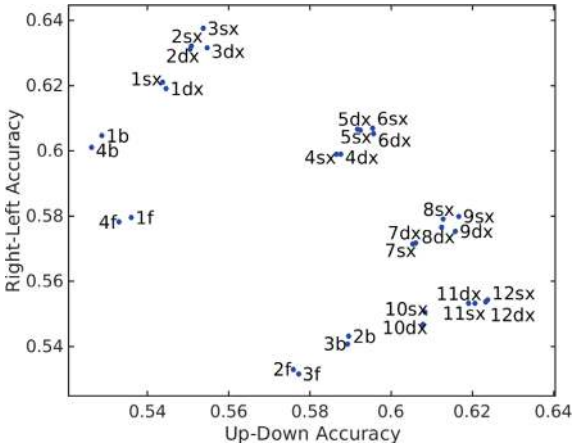


Fig. 7.10 The plot depicts sensor accuracy in discerning charge location on the test mass face, distinguishing between up-down and right-left positions

charge location while other techniques like the ion-gas cloud [9] in a higher pressure environment do not need this information. As a consequence, in the first case the most relevant information is the charge location; while, in the second scenario, its value.

Considering the first possibility, we choose the sensors in order to maximize the probability to discover the charge position. To reach this goal we take into account the neural network probability to pick the wrong face $P(\overline{FB})$ and we compare it with the probability to pick the correct face $P(FB)$ which, according to the Law of Total Probability, we splitted in the probability to pick the correct quarter $P(UD \cap RL \cap FB)$, the probability to pick the wrong adjacent quarters $P(UD \cap \overline{RL} \cap FB)$, $P(\overline{UD} \cap RL \cap FB)$ and the probability to pick the wrong far quarter $P(\overline{UD} \cap \overline{RL} \cap FB)$ (Fig. 7.11).

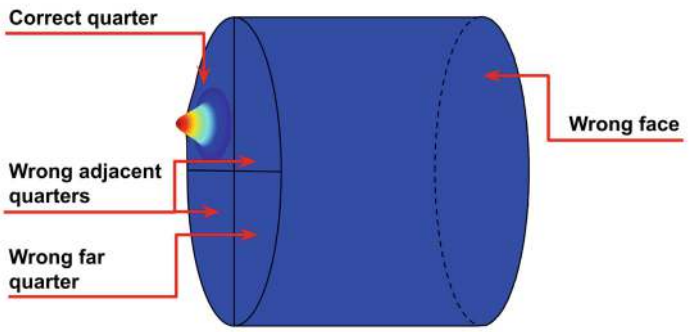


Fig. 7.11 Schematic division of the test mass according to the probabilities introduced

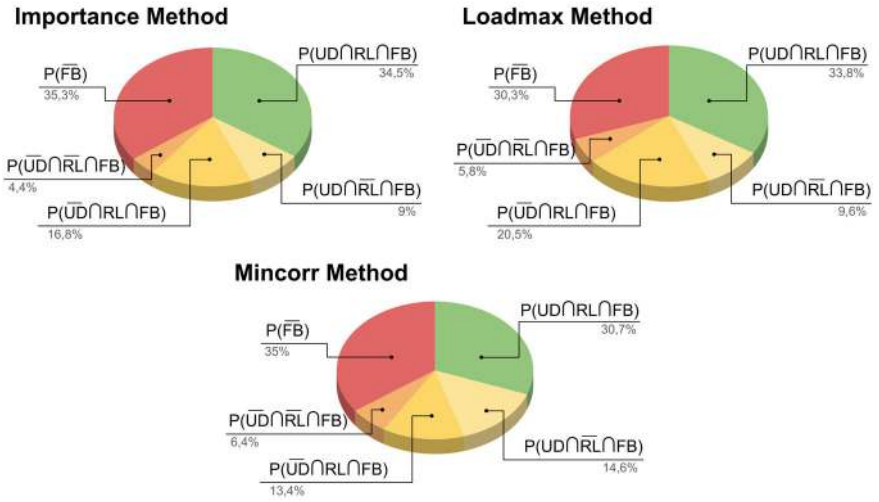


Fig. 7.12 Probability to gain correct or wrong informations utilizing different neural networks

Once we calculated these quantities we compare their value for all the criteria (Fig. 7.12). In doing so we mostly have to consider both the probability to pick the correct quarter and the probability to properly understand whether the charge is located on the front or back face. Indeed, discover the face is crucial, since we are assuming to adopt a directional discharging method.

Both Importance and Loadmax methods have a greater probability to pick the correct quarter with respect to the Mincorr method which has also a great probability to pick the wrong face. Therefore we can exclude this criterion.

Regarding Importance and Loadmax methods, they have a similar probability to pick the correct quarter (the difference between the two criteria is $< 1\%$), but Loadmax is more efficient in picking the correct face ($\sim 5\%$ better), therefore we will opt for the four sensors selected by the Loadmax criterion.

7.5 Discussion

We have devised a method which has the ability to determine the charge distribution of a system exploiting the potentials measured by a certain number of sensors (Fig. 7.13).

The remarkable feature of this operational approach lies in its flexibility since the same procedures can be employed in different geometries. Indeed, it is sufficient to import the model of the system of interest in a simulation, and then the same steps seen for the model under examination can be carried out.

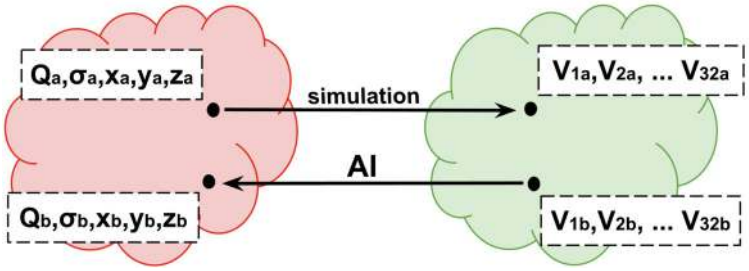


Fig. 7.13 Using simulation, we move from the charge configuration space to the measurement space; conversely, using neural networks, we can determine the charge configuration given the potential values on the sensors

Based on the charge measurements conducted at Virgo [4], the potential measured by the point sensors in the simulations falls below the minimum value detectable with current technologies. Consequently, there is a need to devise a method to surpass this limitation.

A possible solution currently under exploration involves replacing the sensors with potential generators and measuring the resultant force. This approach would have two virtues:

- It would allow us to transpose the same procedure by substituting measured potentials with the induced forces.
- It would have an extreme precision since it would use the interferometer itself to infer the force value.

On the other hand, in this framework the scenario becomes somewhat more intricate. The NN does not perform the inverse operation, since the input potentials are known. Consequently, the input values consist of the injected potentials and the measured force (Fig. 7.14). Furthermore, the presence of multiple potential generators enables the possibility for distinct charge distributions to result in the same force. Conversely, by modifying the potentials, it becomes evident that various forces can correspond to the same charge distribution (Fig. 7.15).

We are aware of certain limitations in our study. The simulation lacks experimental validation, which is indeed necessary for robust verifications since we should verify that our simulations accurately reproduces the real system.

Moreover, our charge analysis is limited, as we exclusively examined Gaussian-distributed charges and we positioned them solely on the front or back face of the test mass. However, assuming a Gaussian distribution is a reasonable approximation, considering our expectation of localized charge distributions.

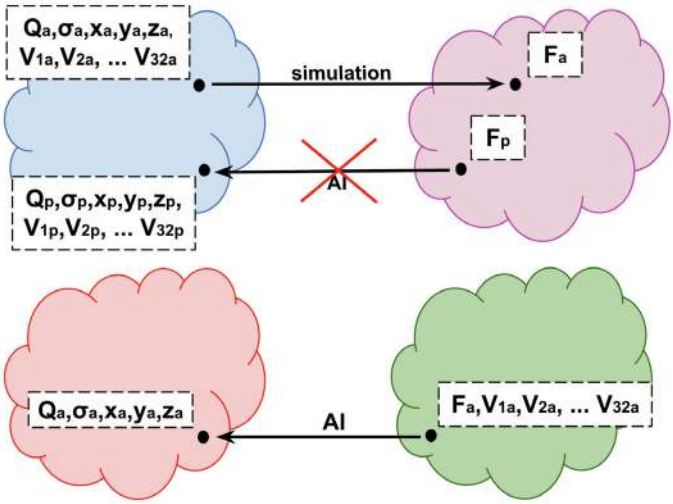


Fig. 7.14 By fixing the potentials and employing simulation, we navigate from the charge configuration space to the measurement space. Conversely, neural networks enable us to deduce the charge configuration given the fixed potentials and the measured force

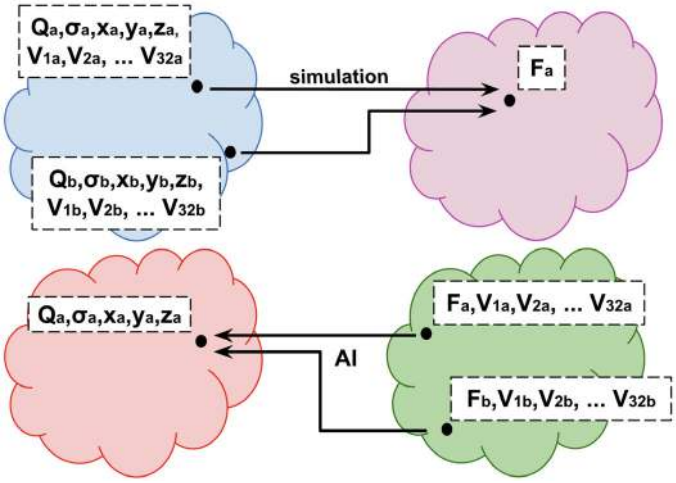


Fig. 7.15 The freedom to choose potentials adds degrees of freedom

Acknowledgements We express our gratitude to the Virgo Collaboration, particularly Ettore Majorana, for providing the CAD model of the NE payload. The COMSOL Multiphysics license was generously supplied by the Istituto Nazionale di Fisica Nucleare (INFN), and the MATLAB license was graciously provided by the University of Genova (UNIGE).

Appendix 1

Using COMSOL Multiphysics we simulated 124,000 symmetric Gaussian distributions. In particular, we simulated 62,000 charge distributions on the TM front face and the same number on the back face.

The charge values were chosen taking into account the measurement made in Virgo [4]. Indeed we simulated random charge values in a range between 1 and $800pC$ (Table 7.2).

Moreover we considered both positive and negative charge values. Specifically, we simulated 74,000 positive charge distributions and 50,000 negative ones.

We considered localized charge distribution. Indeed, the simulated standard deviation value fell within a range between 1 mm and 2 cm.
Regarding the Gaussian positions on the TM face, we randomly distributed them across the entire surface of radius 17 cm (Fig. 7.16).

Table 7.2 Maximum and minimum simulated value for charge and standard deviation

Simulated data		
	Charge modulus	Standard deviation
max	800 pC	2 cm
min	1 pC	1 mm

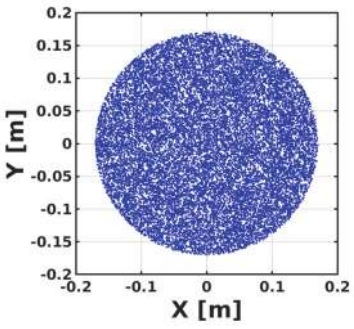


Fig. 7.16 The considered Gaussian locations on the front TM face

Appendix 2

In Table 7.3 you can find the complete sensor ranking according to Importance, Loadmax and Mincorr criteria.

Table 7.3 Sensor ranking from the best sensor to the worst according to Importance, Loadmax and Mincorr criteria

Sensor ranking		
Importance	Loadmax	Mincorr
3f	1b	3f
2f	4f	10dx
2dx	1f	10sx
2sx	3dx	11dx
5sx	2f	11sx
1dx	5sx	12dx
5dx	3b	12sx
1sx	12dx	7sx
1f	1dx	4b
4f	4dx	7dx
4sx	9sx	1b
4dx	3f	9dx
6sx	2b	8sx
6dx	3sx	8dx
3sx	7dx	3b
3dx	2dx	6sx
8dx	9dx	6dx
8sx	6dx	2b
7dx	11sx	4sx
7sx	10sx	4dx
9dx	8dx	3sx
9sx	12sx	3dx
11sx	8sx	2sx
11dx	5dx	2dx
10sx	7sx	1sx
10dx	1sx	5dx
12dx	4b	9sx
12sx	11dx	4f
2b	2sx	1f
1b	10dx	1dx
4b	6sx	2f
3b	4sx	5sx

Appendix 3

The neural networks used to determine charge and standard deviation value consist of 1 layer with 20 neurons. The activation function is the hyperbolic tangent sigmoid and the Levenberg-Marquardt backpropagation algorithm has been used to train the NNs.

The categorical NNs consist of 1 layer wih 20 neurons.The activation function is the rectified linear unit and the Broydon-Fletcher-Goldfarb-Shanno quasi-Newton algorithm has been used to train the NNs.

Appendix 4

In Table 7.4 you can find the accuracy in charge localization for neural networks trained using the four preferred sensors according to Importance, Loadmax and Mincorr criteria.

Table 7.4 The reported value and error correspond respectively to the mean and standard deviation obtained generating fifteen neural networks under the same conditions

Front-back accuracy			
SNR	Importance	Loadmax	Mincorr
200	0.7330 ± 0.0024	0.8683 ± 0.0042	0.7893 ± 0.0026
100	0.7070 ± 0.0019	0.8217 ± 0.0028	0.7358 ± 0.0017
50	0.6808 ± 0.0032	0.7708 ± 0.0026	0.6897 ± 0.0028
20	0.6455 ± 0.0036	0.6961 ± 0.0057	0.6492 ± 0.0048
10	0.6055 ± 0.0042	0.6360 ± 0.0101	0.6127 ± 0.0038
5	0.5408 ± 0.0070	0.5762 ± 0.0101	0.5640 ± 0.0149
Up-down accuracy			
SNR	Importance	Loadmax	Mincorr
200	0.8191 ± 0.0050	0.8188 ± 0.0117	0.8780 ± 0.0040
100	0.7892 ± 0.0051	0.7475 ± 0.0114	0.8406 ± 0.0036
50	0.7447 ± 0.0113	0.6863 ± 0.0086	0.7826 ± 0.0047
20	0.6695 ± 0.0147	0.6244 ± 0.0097	0.6947 ± 0.0069
10	0.6227 ± 0.0305	0.5862 ± 0.0128	0.6468 ± 0.0116
5	0.5794 ± 0.0356	0.5651 ± 0.0121	0.5979 ± 0.0246
Right-left accuracy			
SNR	Importance	Loadmax	Mincorr
200	0.9304 ± 0.0105	0.9054 ± 0.0081	0.8842 ± 0.0103
100	0.9156 ± 0.0150	0.8728 ± 0.0105	0.8393 ± 0.0103
50	0.8723 ± 0.0172	0.8432 ± 0.0131	0.7854 ± 0.0120
20	0.8021 ± 0.0197	0.7845 ± 0.0254	0.6763 ± 0.0141
10	0.7230 ± 0.0324	0.6990 ± 0.0283	0.5997 ± 0.0189
5	0.6625 ± 0.0448	0.6363 ± 0.0426	0.5952 ± 0.0175

Table 7.5 The reported value and error correspond respectively to the mean and standard deviation obtained generating fifteen neural networks under the same conditions. The input data have been z-score normalized

Charge MSE			
SNR	Importance	Loadmax	Mincorr
200	0.1513 ± 0.0021	0.1441 ± 0.0009	0.1587 ± 0.0013
100	0.1525 ± 0.0009	0.1466 ± 0.0008	0.1711 ± 0.0024
50	0.1594 ± 0.0005	0.1534 ± 0.0004	0.1872 ± 0.0024
20	0.2271 ± 0.0079	0.2157 ± 0.0046	0.2469 ± 0.0020
10	0.4202 ± 0.0089	0.3844 ± 0.0124	0.4158 ± 0.0169
5	0.9822 ± 0.0018	0.9780 ± 0.0020	0.9778 ± 0.0020
Standard deviation MSE			
SNR	Importance	Loadmax	Mincorr
200	0.9824 ± 0.0056	0.9726 ± 0.0118	0.9821 ± 0.0088
100	0.9808 ± 0.0082	0.9730 ± 0.0097	0.9843 ± 0.0049
50	0.9774 ± 0.0065	0.9770 ± 0.0063	0.9855 ± 0.0026
20	0.9851 ± 0.0027	0.9825 ± 0.0032	0.9876 ± 0.0015
10	0.9882 ± 0.0009	0.9872 ± 0.0009	0.9899 ± 0.0005
5	0.9941 ± 0.0007	0.9935 ± 0.0003	0.9940 ± 0.0003

In Table 7.5 you can find the mean square error (MSE) in charge value and standard deviation determination for analogous neural networks.

References

1. Ugolini, D., et al.: Charging issues in LIGO. In: 30th International Cosmic Ray Conference, vol. 3, pp. 1283–1288 (2007)
2. Prokhorov, L.G., Mitrofanov, V.P.: Space charge polarization in fused silica test masses of a gravitational wave detector associated with an electrostatic drive. Classical and Quantum Gravity (2010)
3. Hewitson, M., et al.: Charge measurement and mitigation for the main test masses of the geo 600 gravitational wave observatory (2007)
4. Fiori, I., et al.: The hunt for environmental noise in Virgo during the third observing run. Galaxies (2020)
5. Armato, F., Chincarini, A.: Charge monitoring of test masses in gravitational waves interferometers. Nucl. Instrum. Methods Phys. Res. **1069**
6. Accadia, T., et al.: Advanced Virgo technical design report. Technical Report VIR-0128A-12, The Virgo Collaboration (2012)
7. Jolliffe, I.T.: Principal Component Analysis. Springer (2011)
8. Spallino, L., et al.: Can electrons neutralize the electrostatic charge on test mass mirrors in gravitational wave detectors? Phys. Rev. D (2022)
9. Campsie, P., et al.: Charge mitigation techniques using glow and corona discharges for advanced gravitational wave detectors. Classical and Quantum Gravity (2011)

Chapter 8
Machine Learning to Optimize Newtonian Noise Cancellation in Third-Generation Gravitational Wave Detectors

Francesca Badaracco and Luca Naticchioni

Abstract Newtonian noise affects gravitational wave detectors in the low frequency band (below 20 Hz). It is generated by gravity fluctuations happening nearby the detector. Being it related to passing seismic waves, it can be predicted by monitoring the seismic field. The Einstein Telescope, a third-generation gravitational-wave detector, will be built underground and it will need a Newtonian noise cancellation system. In this paper we discuss how a cancellation system can be designed when the available seismic data will be scarce.

8.1 Introduction

The third generation (3G) of gravitational wave (GW) detectors will expand the observation band to the Low Frequency (LF) band (2–10 Hz). This will enable key astrophysical observations, such as the initial inspiral phase of merging neutron stars, intermediate mass black hole coalescences, and isolated neutron stars. In the LF band, the primary constraints on detector sensitivity are seismic noise and gravity fluctuations, also known as Newtonian noise (NN) [1]. NN is caused by gravity fluctuations resulting from density variations due to seismic waves passing near the most sensitive parts of a GW detector (e.g. the test masses). Pressure fluctuations (atmospheric and machinery induced) also generate a non-negligible component of NN at low frequencies. This noise directly couples to the test masses of the detector, therefore it is very difficult to physically reduce it with experimental techniques. Nevertheless, the NN signal can be estimated and subtracted from the detector output

F. Badaracco (✉)
Dipartimento di Fisica, via Dodecaneso 33, 16146 Genova, Italy
e-mail: francesca.badaracco@unige.it

L. Naticchioni
INFN sez. di Roma, P.le Aldo Moro 2, 00185 Rome, Italy
e-mail: luca.naticchioni@roma1.infn.it

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_8

by knowing the seismic field measured by a 3D seismometer array deployed around the test masses. Gaussian Process (GP) regression has proven to be effective in constructing a surrogate model of the seismic field for optimization purposes [2]. GPs integrate two key features: they are mathematically analogous to established models, yet they can learn from data to predict new values accurately [3]. Under certain conditions, GPs bear a strong resemblance to large neural networks. In the realm of machine learning, GPs are a potent tool, applicable to both regression and classification problems [4]. In the geophysical field, they are recognized as "kriging" [5, 6].

Designing an optimal array for NN cancellation in underground GW detectors presents several challenges. The scarcity of data may hinder meaningful GP regression, and the cost of deploying underground sensors will be substantial (see Sect. 8.4). Therefore, research is necessary to gain insights for optimizing the array and managing the associated costs. The subsequent sections will discuss the application of GP regression in current detectors (Sect. 8.2) and its potential use in third-generation GW detectors (Sect. 8.3). Lastly, Sect. 8.4 will detail the steps involved in borehole installation, which is essential for data collection and the deployment of seismic sensors post-optimization.

8.2 Gaussian Process Regression for Optimal Positioning

The initial strategy for identifying the optimal array for an underground detector like ET is assuming a homogeneous and isotropic seismic field [7, 8]. This approach is useful in order to inspect the properties of a NN cancellation system, such as the reduction factor versus the number of sensors, the average distance between sensors, how the optimal array changes with variations in the seismic field composition (shear- and compression-wave content) and its robustness to deployment in sub-optimal positions. Clearly, the final array configuration will have to be adapted to the real seismic field which will likely not be exactly isotropic and homogeneous. A similar work was conducted in Virgo to find the optimal array based on seismic data [2]. GP regression was applied starting from collected seismic data to create a surrogate model of the cost function used in the optimization process. This approach poses already some challenges regarding data scarcity, which was successfully solved for Virgo. However, for ET, this problem will be even more significant since the array will need to be optimized in a three-dimensional space, and data collection will be both expensive and challenging. Surface seismic data are relatively simpler to obtain, while underground seismic data need the excavation of boreholes and the deployment and maintenance of the seismic sensors (see Sect. 8.4). For this reason, we can already expect that the available data to perform the GP regression will be insufficient. Some improvements to the process have already been proposed, such as using the information of simulated cross-correlations to create priors for the GP hyperparameters [9]. Better hyperparameters derived from data will aid in reducing the estimation error, but if the data are too sparse we can assume that also

the simulations will be a poor approximation of the reality, therefore the uncertainty will remain high. The next section will outline potential future work to gain more insights into the optimal array for NN cancellation.

8.3 Future Perspective for Optimal Positioning

The high cost of collecting underground seismic data will inevitably lead to data scarcity, which will impact the design of the NN cancellation system in ET. However, the impact of this data scarcity can be mitigated by leveraging simulated cross-correlations [10] to enhance GP regression, as discussed in the previous section. While uncertainty will persist, it could be utilized to examine potential effects on the performance of the NN cancellation system. Once the GP model is trained, it is possible to draw from the GP new functions representing the seismic field. However, having scarce data will entail that the drawn seismic fields will have large errors. Usually, the average function (i.e. the estimated seismic field) should be used to identify the optimal array. With scarcity of data, the average function will not represent the reality. However, such a GP could still be exploited to test the impact of data scarcity on the optimal array and assist in determining the need for another measurement campaign. Indeed, after determining the optimal array based on the average function, additional seismic fields could be drawn from the GP. They could be used to evaluate how NN cancellation performance vary when the seismic field differs from the one assumed during the optimization. In such case, the seismic field would still be compatible with the seismic data, but it simply would differ where data are missing. Furthermore, this kind of GP could provide insights into the most effective locations for new measurements.

The first step for such a GP, would be that of assuming an infinite three-dimensional space. However, to find a balance between costs and performance, it could be beneficial to examine how incorporating a surface into the underground model influences the optimal array configuration. Indeed, there is likely to be some correlation between the surface seismic field and the underground one. Therefore, a feasible compromise might involve deploying a large number of sensors on the surface and in the caverns, while limiting the number of sensors installed in boreholes.

8.4 Borehole Installation

The process of reconstructing the seismic field for NN cancellation in an underground 3G gravitational wave detector necessitates the deployment of multiple triaxial broadband seismometers at varying depths around the test masses. These sensors can be placed in dedicated boreholes drilled in the vicinity of the main caverns that house the detector test masses. The cost of a borehole is proportional to its diameter, particularly due to the steel lining, making the use of compact sensors a

more cost-effective option. In traditional geophysical monitoring, only a single commercial high-quality broadband seismometer is installed per borehole. However, for the NN cancellation 3D array, considering the number of sensors required, multiple seismometers could be installed at different depths in each borehole. This would necessitate modifications to the commercial cabling and clamping systems. Furthermore, these compact seismometers typically have a limited tolerance to tilt relative to the local gravitational vertical (usually within a few degrees). Therefore, the verticality should be continuously monitored and verified during the borehole drilling process.

After the drilling of each borehole, a structural log is required to localize the depth of the main discontinuities in the drilled rocks. These discontinuities should be avoided in the sensor installation phase. In the following steel tube installation, it is crucial to achieve a good cementing of the tube to the surrounding rocks. This ensures a good mechanical transmission of seismic waves and reduces the resonances of the steel tube.

Finally, the borehole seismometers and their DAQ electronics must be able to observe the Earth background as represented by the New Low Noise Model (NLNM) [11]. This is particularly important at the lower frequencies of interest for a 3G GW detector. Therefore, they should at least posses a sensitivity of the order of 10^{-10} m/s/$\sqrt{\text{Hz}}$ between 2 and 10 Hz.

8.5 Conclusions

Underground GW detectors such as ET [12] will pose significant challenges. One of these will be represented by the NN: the final limitation to Earth-bound GW detectors at low frequencies. Current techniques aimed at designing the optimal array for an efficient NN cancellation are based on the use of GP regression applied on seismic data. In the future, collecting seismic data will be more challenging and expensive as underground seismometers will require the drilling and setting-up of boreholes. Hence, GP regression will provide a less precise optimal array. How the scarcity of data will impact the NN array design should be further studied as the NN reduction factor that the NN cancellation system will be able to provide will have an important impact on the final ET sensitivity at low frequencies.

References

1. Harms, J.: Terrestrial gravity fluctuations. Living Rev. Relativity **22**, 1 (2019)
2. Badaracco, F., Harms, J., Bertolini, A., Bulik, T., Fiori, I., Idzkowski, B., Kutynia, A., Nikliborc, K., Paoletti, F., Paoli, A., Rei, L., Suchinski, M.: Machine learning for gravitational-wave detection: surrogate Wiener filtering for the prediction and optimized cancellation of Newtonian noise at Virgo. Classical and Quantum Gravity **37**, 195016 (2020)

3. Williams, C.K.I., Rasmussen, C.E.: Gaussian Processes for Machine Learning 2. MIT press Cambridge, MA (2006)
4. Kim, H.-C., Lee, J.: Clustering based on Gaussian processes. Neural Comput. **19**, 3088 (2007)
5. Jones, D.R.: A taxonomy of global optimization methods based on response surfaces. J. Global Optim. **21**, 345 (2001)
6. Forrester, A., Sobester, A., Keane, A.: Engineering Design via Surrogate Modelling: A Practical Guide. Wiley (2008)
7. Badaracco, F., Harms, J.: Optimization of seismometer arrays for the cancellation of Newtonian noise from seismic body waves. Classical and Quantum Gravity **36**, 145006 (2019)
8. Badaracco, F., Harms, J., Rei, L.: Joint optimization of seismometer arrays for the cancellation of Newtonian noise from seismic body waves in the Einstein telescope. Classical and Quantum Gravity **41**, 025013 (2024)
9. Andric, T.: Providing priors to Bayesian array optimization for the Sardinian candidate site of the Einstein telescope. Il Nuovo Cimento C **45**, 6 (2022)
10. Andric, T., Harms, J.: Simulations of gravitoelastic correlations for the Sardinian candidate site of the Einstein telescope. J. Geophys. Res.: Solid Earth, **125**(10), e2020JB020401 (2020)
11. Peterson, J.R.: Observations and modeling of seismic background noise. U.S. Geological Survey Rept. **93**, 322 (1993)
12. ET Steering Committee Editorial Team, "Design Report Update 2020 for the Einstein Telescope"

Chapter 9
Sensor Placement Algorithms and Learning Methods for Gravitational Wave Interferometers

Conor Muldoon

Abstract This chapter explores combinatorial optimization techniques for sensor placement within the context of applications for gravitational wave detectors. In cases where an objective function, being optimised subject to a cardinality constraint, is submodular, or has a diminishing returns property, and is monotone, a simple greedy algorithm achieves near-optimal performance. More broadly, in the context of gravitational wave interferometry, placing sensors requires the optimisation of objective functions that are neither submodular nor supermodular. Such NP-hard optimisation problems can be addressed through the use of a priori hand-crafted heuristics and meta-heuristics inspired by natural, biological, and evolutionary processes. This chapter introduces a novel approach to address the problem by learning latent functions using pointer networks within an actor-critic framework, leveraging attention networks along with deep reinforcement learning. Preliminary implementation challenges and future directions are discussed, highlighting the potential for improved scalability and efficiency for complex optimization tasks.

9.1 Introduction

There are a variety of applications within gravitational wave interferometry where the optimal placement of sensors enables the removal of noise. Sources of noise, as discussed in the other chapters in Part 2, include the deposition of charge on the interferometer mirrors [1] and Newtonian noise [2–4], which is related to passing seismic waves and affects detectors in the low frequency band. The buildup and motion of surface charge on the optics of the detector is due to both abrasive contact with materials and interaction with cosmic rays, whereas Newtonian noise or gravity gradient noise is a result of fluctuations in the local gravitational field due to mass density variations.

C. Muldoon (✉)
Department of Computing and Mathematics, Manchester Metropolitan University, Chester St, Manchester M1 5GD, UK
e-mail: c.muldoon@mmu.ac.uk

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_9

Gravitational wave observatories, such as LIGO and Virgo, consist of systems of mirrors, which are referred to as test masses (TMs), suspended freely via multi-stage pendulum systems. The disposition of charge on the TMs can be mitigated using an optimised configuration of sensors to monitor physical characteristics of the charge. The motion of the TMs is minimised to ensure, to the extent possible, that the variations in the distance between mirrors is due to gravitational waves alone. This is achieved passively via the pendulum systems and actively via sensors and actuators. Newtonian noise, however, bypasses this isolation of the TMs in that there is a gravitational coupling between the TMs and the seismic field. Fluctuations in the field occur due to variations in atmospheric pressure, movement of the detector infrastructure, anthropogenic activities, and so forth. Thus, Newtonian noise will be present in the interferometer data. To address this issue, an estimate of the Newtonian noise can be subtracted from the interferometer data by placing seismometers around the TMs to monitor the noise source. An estimate is required in that the gradient noise induced at the TMs cannot be measured directly.

This paper proposes the use of submodular optimisation to address the problem of choosing sensor locations for charge monitoring and pointer networks, attention networks, and deep reinforcement learning to monitor the Newtonian noise source. Greedy algorithms for optimising the latter do not perform well in that the objective function is not submodular and the number of sensors deployed must be considered as a whole. For the former, in the case of a monotone submodular objective function, a greedy approach will perform close to optimal with a bound of $(1 - 1/e)$. This is the best bound that can be achieved for a polynomial-time algorithm assuming $P \neq NP$ [6].

The remainder of the chapter is organised as follows. Section 9.2 discusses related methods for sensor placement. Submodular optimisation is covered in Sect. 9.3. The use of sequence-to-sequence learning for sensor placement is discussed in Sect. 9.4. Section 9.5 provides an overview of the implementation, preliminary findings, and directions for future work. Section 9.6 concludes the chapter.

9.2 Related Research

Several approaches have been adopted in the literature that optimise sensor placements based on Principal Component Analysis (PCA) [1, 7]. One issue with the use of PCA in this context, however, is that the principal components represent a linear combination of sensor locations and will not be valid for a true sensor deployment; there cannot be a fraction of a sensor deployed for instance. Prior research on sensor placement [8, 9] has made use of submodular optimisation and selection criteria, such the mutual information and conditional entropy of Gaussian processes and the mean squared prediction error to achieve near-optimal performance in certain settings. This is the approach advocated here. Furthermore, submodular optimisation has been adopted in determining sensor placements for optimal Kalman filtering [10] and in maximising the Fisher information [11].

Metaheuristic algorithms [13] have previously been adopted to address the problem of determining the optimal sensor deployment locations for estimating the Newtonian noise induced on the TMs [5]. These algorithms include Particle Swarm Optimization (PSO) [14], Basin-Hopping [15], and Differential Evolution [21]. PSO simulates particle positions and velocities to optimize a given measure of quality. Basin-Hopping combines random perturbations with local optimization to escape local minima. Differential evolution iteratively improves candidate solutions through search space navigation. Greedy algorithms for submodular optimisation do not perform well for the problem of estimating Newtonian noise and the number of sensors deployed as a whole needs to be taken into account to achieve good performance. One benefit of these metaheuristic algorithms over the approach discussed in this chapter is that they do not require the space to be discretised, for instance into a grid, in contrast to combinatorial algorithms. The approach discussed here, however, if scaled up, will enable the learning of heuristics and metaheuristics for sensor placement, rather than hardcoding heuristics and metaheuristics, to improve performance in a similar manner to how reinforcement learning can surpass anthropogenic heuristics for games, such as Go and Chess [22, 23].

9.3 Submodular Sensor Placement

There are several alternate approaches to performing sensor placement. When there is a single cardinality constraint and the objective function is submodular, or approximately submodular, and is monotone, a greedy algorithm (see Algorithm 1 from [8]) performs near optimally. In such cases, the algorithm has a bound of $(1, -1/e)$, which is the best that can be achieved for a polynomial-time algorithm assuming $P \neq NP$ [6]. The greedy algorithm performs better than this bound in practice, even in cases whereby the data are highly correlated. Thus, there is a gap between theory and practice, but this is explained, to a certain extent, in [16].

Submodular sensor placement represents a classic subset selection problem where a finite set of locations is given, and the objective is to choose a subset that maximizes utility. Utility functions, such as the conditional entropy and mutual information of Gaussian processes and the mean squared prediction error, which is approximately submodular, are used to evaluate the selected and unselected sets of sensor locations. This is a special case of the NP-complete set cover problem.

A set function is submodular if it satisfies the diminishing returns property. That is, the incremental benefit of adding a sensor decreases the more sensors are added. Formally, a set function $f : 2^S \rightarrow \mathbb{R}$ is submodular if for $A \subset B \subset S$ and $x \in S$, $f(A \cup \{x\}) - f(A) \geq f(B \cup \{x\}) - f(B)$.

In practice, there will be locations where sensors cannot be placed. Additionally, sensors have a physical size and can only be placed to a certain accuracy. With the methods discussed in this section and Sect. 9.4, it is a requirement to discretise the space, for instance into a grid. This is not a requirement for metaheuristic algorithms. With a fine enough granularity or grid spacing, this will not be an issue in terms of

the objective function in that arbitrary precision cannot be achieved in any case and relatively small changes in position will not have a significant impact on utility. Having a fine granularity is an issue in terms of scalability, however, regarding sequence-to-sequence learning.

9.4 Sequence-To-Sequence Learning for Sensor Placement

The sensor placement problem, from a combinatorial perspective, can be considered in terms of sequence-to-sequence learning [17]. A representation can be created with labels for potential sensor locations and a mapping created to a subset of optimal sensor locations given a utility function, such as the set of locations that maximise the signal when the subtraction of an estimate of the Newtonian noise is considered.

One approach to sequence-to-sequence learning is through that of pointer networks (see Fig. 9.1). Pointer networks, introduced by Vinyals et al. [20], are used for problems where the output sequence length differs from the input sequence length. Pointer networks avail of attention in using context to determine the next token to follow in a chain in a stochastic manner.

Reinforcement learning can be adopted as tool in solving optimization problems. Traditional handcrafted methods or heuristics for such problems often fail to scale or adapt to new data efficiently. Deep Reinforcement Learning (DRL) enables systems

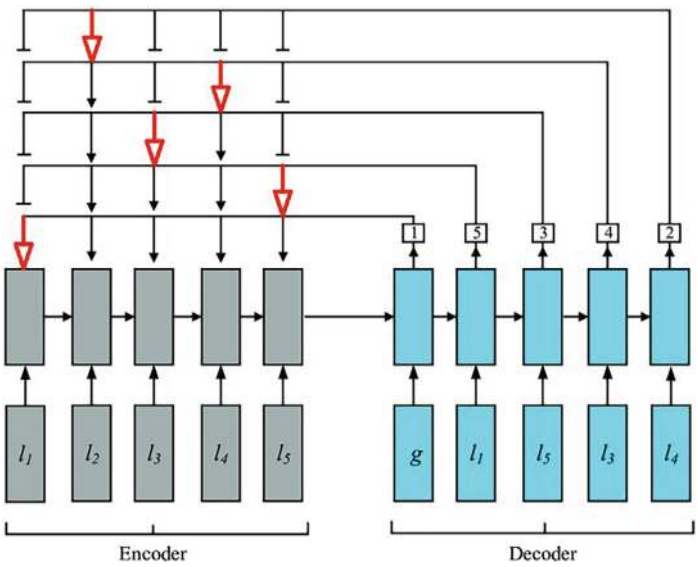


Fig. 9.1 Pointer network [18, 19]

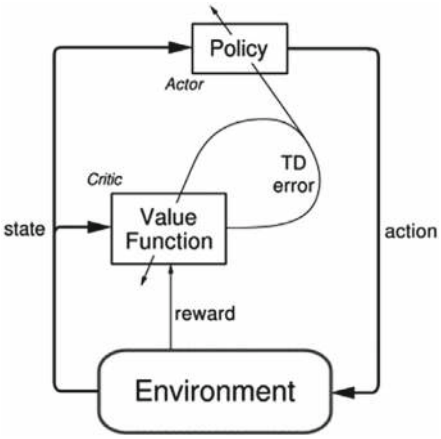


Fig. 9.2 Actor-critic framework [26, 27]

to learn and improve heuristics and latent functions through interaction with the environment.

The research discussed in this chapter draws from [25], where DRL was used to address the Traveling Salesperson Problem (TSP). This chapter adopts a similar approach through the use of an actor-critic architecture, but it is applied to the problem of sensor placement rather than the TSP. The actor-critic architecture (see Fig. 9.2), as described by Sutton and Barto [26], involves two main components:

- The actor chooses actions based on the current policy.
- The critic evaluates the actions taken by the actor and provides feedback.

DRL methods employed in actor-critic frameworks and pointer networks have shown promise in this type of setting using architectures such as recurrent neural networks with Long Short-Term Memory (LSTM) cells or transformers, along with optimisers, such as Adam [24]. In addition to the TSP, DRL has been used to solve the knapsack problem [12]. This is closer to the problem discussed here than the TSP in that a subset is chosen. Indeed, the same architecture for solving the knapsack problem could be adopted by giving all potential sensor locations the same weight in the knapsack and changing the objective function to be non-additive[1].

[1] It should be noted, however, that if using dynamic programming, rather than a pointer network, to solve the knapsack problem in this modified form, the results will be no better than using a greedy algorithm.

9.5 Implementation and Preliminary Findings

The initial implementation was carried out using PyTorch on an NVIDIA Tesla M10 GPU. The actor-critic framework was adopted along with a neural network architecture with LSTM cells. An 80 sensor location pointer network was used. The objective function adopted minimised the relative residual Newtonian noise spectral density spectrum left in the interferometer data by a Wiener filter (see Eq. 2 from [5]). The architecture worked in principle but faced challenges related to scalability and GPU memory limitations. Specifically, the need to discretise the space for the encoding for sequence-to-sequence learning reduced the efficacy of the proposal when using a Tesla M10 or a GPU with similar computational resources in comparison to state-of-the-art metaheuristic algorithms.

A far finer grid spacing and resolution can be adopted when a greedy algorithm is used. The performance of using the greedy algorithm when the objective function is monotone submodular has been demonstrated in prior research [8, 11]. In the context of determining sensor placements for estimating Newtonian noise, however, this approach did not perform well. Preliminary results indicated that using pointer networks worked to a certain extent with the hardware adopted but not well enough to make it a viable alternative to current methods.

There are several lines of investigation for future research. In general, in machine learning, overfitting is considered a problem. In the context of combinatorial optimisation, however, overfitting for a given problem instance is desirable in cases where there is no need for generalisation and the goal is to find the best solution to the given instance. This will be the case in determining optimal sensor placements for gravitational wave interferometers. Furthermore, algorithmic improvements could be made to the approach discussed. For instance, beam search with truncation when used with heuristic breadth-first search methods could be used to reduce the computational complexity on average. Additionally, the use of transformers instead of LSTM cells will likely lead to better performance for combinatorial sensor placement problems given their success in large language models and other application domains. The methods discussed in this chapter have a variety of applications in discrete optimization problems beyond sensor placement, highlighting the potential impact and versatility of sequence-to-sequence learning for problems of this type. With rapid advances semiconductor technology and with the availability of additional resources, scalability issues will be less pronounced.

9.6 Conclusion

Determining the optimal placement of sensors in the context of gravitational wave interferometry is key to enabling the subtraction of noise in scenarios where it cannot be mitigated through passive or active control. Depending on the objective function adopted, various algorithms are apt to perform this task. For certain classes of prob-

lem, where the objective function is monotone submodular, or has a diminishing returns property, the greedy algorithm will perform in a near-optimal manner. In other cases, however, alternative approaches must be adopted. This paper proposed an approach to learning good sensor placements using an actor-critic framework and pointer networks. There were issues in terms of scalability, but with rapid advances in hardware, these issues will be alleviated in the future.

References

1. Armato, F., Chincarini, A.: Charge monitoring of test masses in gravitational waves interferometers. Nucl. Instrum. Methods Phys. Res. Sect. A: Accelerators, Spectrometers, Detectors Associated Equipment 169833 (2024)
2. Jose, R., Kalaimani, R.: Optimization of sensor placement for broadband Newtonian noise cancellation in GW detectors. In: 2021 25th International Conference on System Theory, Control and Computing (ICSTCC), pp. 132–137 (2021)
3. Driggers, J., Harms, J., Adhikari, R.: Subtraction of Newtonian noise using optimized sensor arrays. Phys. Rev. D—Particles, Fields, Gravitation, Cosmol. **86**, 102001 (2012)
4. Badaracco, F., Harms, J., Rei, L.: Joint optimization of seismometer arrays for the cancellation of Newtonian noise from seismic body waves in the Einstein telescope. Classical and Quantum Gravity. **41**, 025013 (2024)
5. Badaracco, F., Harms, J.: Optimization of seismometer arrays for the cancellation of Newtonian noise from seismic body waves. Classical and Quantum Gravity. **36**, 145006 (2019)
6. Nemhauser, G., Wolsey, L., Fisher, M.: An analysis of approximations for maximizing submodular set functions-I. Math. Program. **14**, 265–294 (1978)
7. Brunton, B., Brunton, S., Proctor, J., Kutz, J.: Optimal sensor placement and enhanced sparsity for classification (2013). ArXiv Preprint. ArXiv:1310.4217
8. Krause, A., Singh, A., Guestrin, C.: Near-optimal sensor placements in Gaussian processes: theory, efficient algorithms and empirical studies. J. Machine Learning Res. **9** (2008)
9. Krause, A., McMahan, H., Guestrin, C., Gupta, A.: Robust submodular observation selection. J. Machine Learning Res. **9** (2008)
10. Tzoumas, V., Jadbabaie, A., Pappas, G.: Sensor placement for optimal Kalman filtering: fundamental limits, submodularity, and algorithms. In: 2016 American control conference (ACC), pp. 191–196 (2016)
11. Liu, L., Hua, C., Xu, J., Leus, G., Wang, Y.: Greedy sensor selection: leveraging submodularity based on volume ratio of information ellipsoid. IEEE Trans. Signal Process. **71**, 2391–2406 (2023)
12. Sur, G., Ryu, S., Kim, J., Lim, H.: A deep reinforcement learning-based scheme for solving multiple knapsack problems. Appl. Sci. **12**, 3068 (2022)
13. Abdel-Basset, M., Abdel-Fatah, L., Sangaiah, A.: Metaheuristic algorithms: a comprehensive review. In: Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications, pp. 185–231 (2018)
14. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings Of ICNN'95-international Conference On Neural Networks, vol. 4, pp. 1942–1948 (1995)
15. Wales, D., Doye, J.: Global optimization by basin-hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms. The J. Phys. Chem. A **101**, 5111–5116 (1997)
16. Das, A., Kempe, D.: Submodular meets spectral: greedy algorithms for subset selection, sparse approximation and dictionary selection. In: Proceedings of the 28th International Conference on International Conference on Machine Learning, pp. 1057–1064 (2011)

17. Sutskever, I.: Sequence to Sequence Learning with Neural Networks (2014). ArXiv Preprint. ArXiv:1409.3215
18. Fang, J., Rao, Y., Luo, Q., Xu, J.: Solving one-dimensional cutting stock problems with the deep reinforcement learning. Mathematics. **11**, 1028 (2023)
19. Creative Commons, Attribution 4.0 International. https://creativecommons.org/licenses/by/4.0/
20. Vinyals, O., Fortunato, M., Jaitly, N.: Pointer networks. Adv. Neural Inf. Process. Syst. **28** (2015)
21. Pant, M., Zaheer, H., Garcia-Hernandez, L., Abraham, A., and Others: Differential evolution: a review of more than two decades of research. Eng. Appl. Artif. Intell. **90**, 103479 (2020)
22. Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., and Others: Mastering the game of go without human knowledge. Nature **550**, 354–359 (2017)
23. Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., and Others: A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. Science **362**, 1140–1144 (2018)
24. Zhang, Z.: Improved Adam optimizer for deep neural networks. In: 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS), pp. 1–2 (2018)
25. Bello, I., Pham, H., Le, Q., Norouzi, M., Bengio, S.: Neural combinatorial optimization with reinforcement learning (2016). ArXiv Preprint. ArXiv:1611.09940
26. Sutton, R., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press (2018)
27. Creative Commons, Attribution-NonCommercial-NoDerivatives 4.0 International. https://creativecommons.org/licenses/by-nc-nd/4.0/

Part III
Machine Learning for Gravitational Wave Signal Analysis

Gravitational wave detectors are constantly bombarded by a variety of noise sources, ranging from seismic vibrations and thermal noise to instrumental errors. These noise sources can obscure or mimic the signals produced by astrophysical events, making detection an extremely challenging task. Traditional data analysis methods, while effective, often struggle to handle the volume and complexity of data generated. Machine learning offers a powerful alternative. Machine learning excels at pattern recognition, making it ideally suited for detecting and classifying gravitational wave signals. Once trained on simulated or previously observed signals, machine learning algorithms can quickly and accurately identify signals from various astrophysical sources, such as binary black hole mergers, neutron star collisions, and supernovae.

Machine learning algorithms can be used to estimate the parameters of gravitational wave sources with high accuracy. By training on a wide range of simulated waveforms, these models can infer the masses, spins, and orbital parameters of binary systems, as well as the distance and orientation of the source relative to Earth. This information is crucial for understanding the astrophysical properties and evolution of the sources.

The ability to process and analyze data in real time is a significant advantage of machine learning in gravitational wave detection. Traditional methods often require extensive computational resources and time-consuming post-processing. In contrast, machine learning algorithms can analyze incoming data streams in real time, providing immediate alerts and enabling rapid follow-up observations with other telescopes and detectors. This capability is particularly important for multi-messenger astronomy, where the simultaneous detection of gravitational waves and electromagnetic signals can yield rich scientific rewards.

Chapter 10
Selected Machine Learning Techniques for Gravitational Wave Bursts

Maxime Fays

Abstract Gravitational wave bursts, transient events lasting from milliseconds to minutes within the frequency band of current generation detectors, originate from a range of astrophysical phenomena, including core-collapse supernovae, neutron star glitches, and highly eccentric black hole mergers. Due to the complexity and diversity of these sources, their signal morphologies are often poorly modeled or completely unknown, making traditional matched-filter techniques ineffective for many target sources. More critically, detection methods must be sensitive to entirely unexpected phenomena, adopting an "eyes wide open" approach to enhance detection capabilities beyond known or predictable events. This chapter explores the integration of several machine learning techniques in the analysis of gravitational wave bursts, addressing the challenges posed by unmodeled and unknown signal morphologies and outlining the strategies developed to approach these signals with minimal assumptions.

Keywords Gravitational waves · Machine learning · Bursts · Data analysis · Convolutional neural network · Gaussian mixture model

10.1 Introduction

Gravitational wave astronomy has rapidly advanced due to recent detections. While these advances have been driven primarily by the detection of compact binary mergers, gravitational wave bursts present distinct challenges due to their transient and often unmodeled nature. These signals require advanced data analysis techniques that go beyond traditional approaches.

Gravitational wave bursts, unlike the well-modeled signals from binary black hole or neutron star mergers, often do not conform to pre-existing waveform templates. Bursts can arise from various sources, such as core-collapse supernovae, neutron star glitches, hyperbolic encounters, or more exotic events, such as mergers involv-

M. Fays (✉)
Universite de Liege, Liege, Belgium
e-mail: maxime.fays@uliege.be

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_10

ing previously unknown astrophysical objects [1, 2]. These signals are often complex, short-lived, and unpredictable, making their detection and analysis a significant challenge [3].

The primary difficulty in detecting gravitational wave bursts is due to the unpredictable nature of their sources and signal morphologies. Traditional methods, such as matched-filtering techniques, rely on well-modeled waveform templates, which are often unavailable for events like core-collapse supernovae and neutron star glitches [4]. While matched-filtering is optimal when theoretical models closely match observed data [5], it is less effective for unmodeled signals [3]. This limitation underscores the need for model-independent detection methods.

Transient noise, or "glitches," further complicates the detection of gravitational wave bursts. Gravitational wave detectors, such as those used by LIGO and Virgo, are susceptible to non-Gaussian noise artifacts that can mimic burst-like signals [6]. These glitches can result from various environmental or instrumental sources, making it challenging to distinguish true astrophysical signals from noise. Effective noise rejection techniques are essential to mitigate false positives and improve detection confidence.

In response to these challenges, machine learning techniques have been incorporated into gravitational wave detection frameworks. Deep learning, in particular, has emerged as a powerful tool due to its ability to learn directly from data without requiring predefined models [7]. This shift towards data-driven analysis allows algorithms to classify complex patterns and improve sensitivity to gravitational wave bursts [6]. Methods like Convolutional Neural Networks (CNNs) and Multi-Layer Perceptrons (MLPs) have been applied to handle the high-dimensional data from gravitational wave detectors, significantly enhancing detection accuracy while reducing false positives.

Machine learning is also important for multi-messenger astronomy, where the speed of detection is essential. Quick detection and classification of gravitational wave signals allow for rapid follow-up observations with telescopes and other instruments to capture complementary electromagnetic signals, neutrinos, or cosmic rays [7]. The near-instantaneous processing capabilities of machine learning models increase the likelihood of successful multi-messenger observations, thus enabling more comprehensive studies of the underlying astrophysical events [3].

10.2 Short-Duration Bursts

Short-duration gravitational wave bursts typically last less than a second [6], originating from some of the most cataclysmic events in the universe. They are emitted by a variety of astrophysical phenomena, ranging from mergers of compact objects to asymmetric supernova explosions and encounters involving cosmic strings [8]. The identification and analysis of such bursts are critical for advancing understanding of cosmic events involving extreme gravitational fields.

One primary source of short-duration bursts is the merger of compact binary systems [9], including black holes and neutron stars. Mergers produce gravitational waves that peak in intensity as the objects spiral closer and eventually coalesce [10]. The final moments before the merger may emit intense bursts of gravitational waves [11], especially in cases involving lower mass or highly asymmetric systems. Events involving compact binary systems help in understanding the properties of black holes and neutron stars but also provide insights into the dynamics of binary systems and the nature of their environments [3].

Another significant contributor of short-duration gravitational wave signals is the core-collapse of massive stars, leading to supernovae [4]. Stellar explosions are asymmetric in nature and are theorized to emit bursts of gravitational waves as their cores implode and rebound [12]. The precise mechanism and characteristics of gravitational emissions depend heavily on the internal structure of the collapsing star and the dynamics of the collapse itself [1].

Cosmic strings, hypothetical one-dimensional topological defects formed during phase transitions in the early universe, are also potential sources of short-duration bursts [13]. Interactions such as cusp formation [14] or reconnection events within a network of cosmic strings [15] can release bursts of gravitational waves. Signals from cosmic strings are particularly interesting for cosmology, providing a unique window into the conditions of the early universe and the physics of the very high energies involved in these phase transitions [6].

Moreover, encounters and interactions of neutron stars, either with each other or with black holes, can emit short bursts of gravitational waves [3]. Such encounters may not always lead to immediate mergers but can result in phenomena that emit gravitational waves detectable as short bursts [7]. Interactions involving neutron stars are important for understanding the population and distribution of neutron stars and black holes in galaxies, as they could offer critical data on the end stages of stellar evolution and the dynamics of dense stellar environments [16].

10.2.1 Coherent WaveBurst (cWB)

Coherent WaveBurst (cWB) [2] is a data analysis pipeline that analyzes data from multiple gravitational wave detectors simultaneously, leveraging the coherent nature of gravitational wave detection. cWB does not strictly rely on predefined templates to identify wave signatures, thus allowing for a broader detection capability.

One of the foundational techniques employed within this framework is the use of one-dimensional Wilson-Daubechies-Meyer wavelet transformation. The Wavelets are particularly effective because they can be easily scaled and shifted to match the local characteristics of a signal in the time-frequency domain [7].

Moreover, cWB employs a strategy that combines data from different detectors to enhance the signal-to-noise ratio (SNR). This approach aids in isolating potential gravitational wave events based on their energy distribution across both time and frequency domains.

The core of cWB functionality revolves around the reconstruction of waveforms using a constrained likelihood approach. The algorithm assesses the coherence of the detected signals across the network of detectors, enhancing its ability to pinpoint signal patterns that stand out from the typically non-Gaussian and non-stationary noise [3]. This involves creating a time-frequency representation of the detector data, which allows the algorithm to analyze the data in small segments, targeting specific signal characteristics.

Following the wavelet transform, cWB engages in pattern matching, where it looks for specific patterns in the time-frequency map that resemble expected gravitational wave signatures. These patterns include simplistic geometric shapes such as crosses and chirps (both ascending and descending), utilized to match the expected behaviors of gravitational waveforms without relying on precise waveform models [12].

By applying these methods, cWB can reconstruct the signal from the noisy data by enhancing the coherence of the waveform across the detector network. This process involves adjusting the methods to maximize the overlap with any real signal present, thereby improving the signal-to-noise ratio.

The selection and application of wavelets and pattern matching techniques are adaptive; the cWB algorithm adjusts based on the characteristics of the noise and the potential signal. This maximizes the coherent energy across the network of detectors, focusing on parts of the signal that are consistent across multiple detectors—a strong indicator of a true gravitational wave event as opposed to noise, which is often non-coherent [7].

The cWB algorithm then selects potential GW events, often referred to as *triggers*, and assigns each an individual statistical significance based on the time-shifting method. The technique is based on artificially inducing a non-physical temporal shift between the data streams from different detectors, thereby creating an ensemble of simulated non-coincident datasets. The procedure starts with the identification of a potential gravitational wave signal in the coincident dataset, which is then subjected to multiple iterations of time-shifting. In each iteration, the data from one detector is shifted by a predetermined interval, typically on the order of seconds or minutes, relative to the data from the other detectors. This process creates a new set of non-coincident datasets, each representing an alternative scenario where the observed signal is merely a coincidental fluctuation in the noise rather than an actual gravitational wave event [2]. By comparing this probability to a predetermined threshold, typically set at 1 or 0.1%, it is possible to estimate whether the observed event is statistically significant and therefore worthy of further investigation and potential classification as a confirmed gravitational wave detection.

One of the key strengths of cWB is its adaptability to different types of gravitational wave signals and noise environments. It dynamically adjusts its analysis parameters based on the quality and nature of the incoming data, making it extremely robust and versatile in real-world detection scenarios [7].

cWB has been instrumental in numerous gravitational wave discoveries, notably playing a crucial role in the detection of the first gravitational waves (GW150914) [11]. Since then, it has detected gravitational waves from a variety of sources, including binary black hole mergers [5] and neutron star collisions [6].

10.2.2 Gaussian Mixture Model (GMM)

The Gaussian Mixture Model (GMM) is a probabilistic model that assumes the underlying distribution of the data is a mixture of multiple Gaussian distributions. In this model, each Gaussian component represents a cluster or subgroup within the overall dataset. The GMM is defined as follows:

Let $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ be a set of N data points in a D-dimensional space, where each data point $\mathbf{x}_i$ is a D-dimensional vector. The GMM assumes that the underlying distribution of $\mathcal{X}$ can be modeled as a mixture of K Gaussian distributions, where K is a predetermined number of components.

The probability density function (PDF) of the GMM is defined as:

$$p(\mathbf{x}|\theta) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

where $\theta = (\pi_1, ..., \pi_K, \boldsymbol{\mu}_1, ..., \boldsymbol{\Sigma}_1, ..., \boldsymbol{\Sigma}_K)$ is the set of model parameters, and:

- π_k is the mixing coefficient or weight of the kth component, such that $0 < \pi_k < 1$ and $\sum_{k=1}^{K} \pi_k = 1$
- $\boldsymbol{\mu}_k$ is the mean vector of the kth Gaussian component
- $\boldsymbol{\Sigma}_k$ is the covariance matrix of the kth Gaussian component
- $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is the PDF of a multivariate Gaussian distribution with mean $\boldsymbol{\mu}_k$ and covariance $\boldsymbol{\Sigma}_k$

The log-likelihood function of the GMM can be written as:

$$\mathcal{L}(\theta|\mathcal{X}) = \sum_{i=1}^{N} \log p(\mathbf{x}_i|\theta) = \sum_{i=1}^{N} \log \left[\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right]$$

The model parameters θ can be estimated using the Expectation-Maximisation (EM) algorithm, which iteratively updates the model parameters to maximise the log-likelihood function. The EM algorithm consists of two stages: the Expectation step (E-step) and the Maximization step (M-step). Both steps are alternating and converge to a maximum likelihood estimate of the model parameters.

The Expectation step computes the expected value of the complete data log-likelihood with respect to the current estimate of the model parameters. Specifically, it computes:

- The probability of each observation belonging to each cluster given the current estimates of the model parameters.
- The expected value of the cluster assignments for each observation.

The maximization step updates the estimates of the model parameters to maximize the expected complete data log-likelihood computed in the E-step. It computes:

- The new estimate of the mixture weights (i.e., the probability of each cluster).
- The new estimate of the mean and covariance matrix for each cluster.

Both steps are needed because the GMM likelihood function is complex and difficult to optimize directly. The key challenge is that the cluster assignments are latent variables, which makes it hard to compute the likelihood of the observed data.

By alternating between the E-step and M-step, the EM algorithm can iteratively refine the estimates of the model parameters and the cluster assignments.

One major drawback of the EM algorithm is that it can get stuck in local optima, especially when the model is complex or the data is limited. This happens because the EM algorithm iteratively updates the parameters based on the current estimate of the latent variables, and if the initial values are not well-chosen, the algorithm may converge to a suboptimal solution. EM can also be computationally expensive for large datasets, as it requires iterating over the entire dataset multiple times.

To prevent these drawbacks, several techniques can be employed. One approach is to use different initial values or random restarts to avoid local optima. Another technique is to use a variant of EM, such as incremental EM or online EM, which can handle large datasets more efficiently. Regularization techniques, such as adding penalties to the likelihood function, can also help prevent overfitting and improve the robustness of the algorithm.

Model selection criteria such as the Bayesian Information Criterion (BIC) can be used to select the best model from a set of candidate models. It balances the fit of the model with its complexity, penalizing models with too many parameters, and can prevent overfitting by selecting the most appropriate model for the data.

Once optimal model parameters and the number of Gaussian components have been defined using the Bayesian Information Criterion (BIC), a log-likelihood-based detection statistic, $W = \ln(\hat{L})|_{\hat{K}}$, is constructed using Gaussian Mixture Models (GMM) to differentiate between signal and noise triggers.

The approach models the signal (s) and noise (g) as two separate classes, each represented by their respective GMMs. For each trigger, a detection statistic T is calculated as $T = W_s - W_g$, where W_s and W_g represent the maximum log-likelihood statistics for the signal and noise models, respectively [3].

The GMM-based detection strategy has been shown to extend the number of detected events while maintaining low false alarm rates, significantly improving detection efficiency for short-duration burst signals [16, 17].

10.2.3 Gradient Boosting

Gradient Boosting is an ensemble learning algorithm that combines multiple weak models to create a strong predictive model. The core idea is to iteratively train decision trees on the residuals of the previous tree, which are the differences between the predicted and actual values. This process is repeated multiple times, with each sub-

sequent tree attempting to correct the errors of the previous one. The final prediction is made by combining the predictions from all the individual trees [7].

Let the training dataset be $\{(x_i, y_i)\}_{i=1}^{N}$, where x_i represents the feature vector and y_i is the corresponding target value. Gradient Boosting initializes a prediction model with some arbitrary values, denoted as $F_0(x)$. Then, for each iteration $m = 1, 2, ..., M$, it performs the following steps.

Firstly, the pseudo-residuals are computed as:

$$r_{im} = -\left[\frac{\partial L(y_i, F_{m-1}(x_i))}{\partial F_{m-1}(x_i)}\right],$$

where L is a differentiable loss function. These residuals represent the "errors" of the previous prediction model.

Next, a decision tree $\phi_m(x)$ is trained on the pseudo-residuals using a squared error loss function. The goal is to minimize the sum of the squared pseudo-residuals, which can be expressed as:

$$\sum_{i=1}^{N}(r_{im} - \phi_m(x_i))^2.$$

Then, the prediction model $F_m(x)$ is updated by adding the new decision tree multiplied by a learning rate ν, which controls how quickly the algorithm learns:

$$F_m(x) = F_{m-1}(x) + \nu \cdot \phi_m(x).$$

Finally, the process repeats until a predetermined number of iterations M is reached. The final prediction model is represented as $F_M(x)$.

XGBoost [18] implements Gradient Boosting with several key modifications and enhancements. It uses a more accurate approximation of the loss function using second-order gradients, which helps to improve the convergence rate. Additionally, XGBoost introduces regularization techniques, such as L1 and L2 regularization on the weights of the decision trees, to reduce overfitting. Furthermore, it employs an approximate algorithm for splitting the nodes in the decision trees, called the "exact greedy algorithm", which reduces computational complexity while maintaining accuracy.

XGBoost also incorporates other enhancements, like support for sparse data, handling missing values, and parallel processing capabilities, making it a highly efficient and scalable implementation of Gradient Boosting [3].

10.3 Long-Duration Bursts

The detection of long-duration gravitational wave bursts poses several challenges, primarily due to the unpredictable nature of these signals and the complex noise environment. Long-duration bursts, which can last from several seconds to minutes,

are difficult to model accurately. In contrast to Compact Binary Coalescences (CBCs), where the waveform is well understood through general relativity, the waveforms of long-duration bursts lack precise models. This absence of templates complicates the search, making it necessary to develop more generalized detection algorithms, such as the excess-power method, which focuses on identifying excess energy in the time-frequency domain [19, 20].

Detection pipelines for long-duration bursts rely on methods capable of identifying a wide range of signal morphologies without being constrained by predefined templates. Approaches based on anomaly detection or unsupervised learning have been proposed to handle this variety of signal shapes [21, 22].

Another key challenge is the presence of noise and transient glitches in the data. Gravitational wave detectors like LIGO and Virgo are subject to various environmental and instrumental noise sources. These noise transients, known as glitches, can mimic or obscure true GW signals [23, 24]. Glitches are especially problematic for long-duration bursts, which have a higher probability of overlapping with noise transients due to their extended duration [2].

In long-duration burst searches, the extended signal duration increases the likelihood of coinciding with multiple noise transients, making it difficult to distinguish between true signals and glitches. Advanced machine learning techniques, such as the ALBUS model, described in the sext section, are used to mitigate the impact of these noise artifacts and improve detection efficiency.

Time-frequency (TF) analysis is currently used by all long-duration bursts search pipelines. TF maps provide a representation of the signal's frequency evolution over time, essential for identifying potential burst signals. However, achieving the necessary resolution in both time and frequency domains is difficult. High time resolution is required to capture rapid changes, while high frequency resolution is necessary to distinguish the signal from noise [25]. Improving these techniques is an active area of research to enhance detection sensitivity [26].

The computational demands of analyzing long-duration bursts are significant. Searching over a wide range of possible signal morphologies and durations increases the computational load, particularly when using high-resolution TF analysis [27]. In addition, real-time analysis is required to enable follow-up observations by electromagnetic observatories, placing further strain on computational resources.

The sensitivity of detectors to long-duration bursts is generally lower than for shorter-duration events, as long-duration bursts may have lower signal-to-noise ratios (SNRs) [28]. Glitches further degrade sensitivity, complicating the detection of true GW signals.

10.3.1 Anomaly Detection for Long-Duration Burst Searches (ALBUS)

ALBUS (Anomaly Detection for Long-Duration Burst Searches) is a machine learning model developed to address the challenges of detecting long-duration GW bursts. It employs a Convolutional Neural Network (CNN) architecture designed to process time-frequency maps [29].

The ALBUS architecture is inspired by the U-Net model, which is well-suited for image segmentation tasks [30]. The U-Net consists of an encoder-decoder structure: the encoder compresses the input data into a lower-dimensional representation, and the decoder reconstructs the data, focusing on relevant regions of interest.

The encoder reduces the dimensionality of the input TF maps while capturing essential features of the signal. This is achieved through a series of convolutional layers, each followed by a rectified linear unit (ReLU) activation function and pooling layers [31]. The convolutional layers extract features such as the frequency evolution of the signal, while the pooling layers reduce the spatial resolution, allowing the network to focus on key features.

The operation of each convolutional layer is represented as:

$$h_{l+1} = \sigma(W_l * h_l + b_l)$$

where h_l is the output of the l-th layer, W_l is the weight matrix, $*$ denotes the convolution operation, b_l is the bias term, and σ is the ReLU activation function [32].

At the deepest layer, known as the bottleneck, the network captures the most abstract features of the input data. This layer compresses the TF map into a low-dimensional representation that retains the most critical information.

The decoder mirrors the encoder, progressively increasing the spatial resolution while reducing the feature map depth. The goal of the decoder is to reconstruct the TF map, highlighting areas where a GW signal is likely present. The decoder uses transposed convolutions (also known as deconvolutions) to upsample the feature maps, generating two outputs: the Anomaly Map, which highlights potential signals, and the Glitch Map, which identifies regions affected by noise.

The upscaling operation is given by:

$$h_{l-1} = \sigma(W_l^T * h_l + b_l)$$

where W_l^T is the transposed weight matrix used to reverse the convolution operation [33]. A sigmoid activation function is applied to the final output to constrain the values between 0 and 1, representing the probability of each pixel belonging to a signal or glitch [34].

ALBUS incorporates skip connections between corresponding layers of the encoder and decoder to retain high-resolution information from the earlier encoding

stages [30]. These connections ensure accurate reconstruction of the signal, pixel by pixel. The skip connections are represented as:

$$h'_{l+1} = [h_{l+1}, h_l]$$

where h'_{l+1} is the concatenated feature map at layer $l + 1$, and h_l is the corresponding feature map from the encoder.

ALBUS uses supervised learning, and its training dataset consists of synthetic chirp signals injected into real LIGO noise data. These synthetic signals are designed to mimic the expected frequency evolution of long-duration bursts and are used to generate TF maps for training. The signals are generated using various models, including linear, quadratic, hyperbolic, and logarithmic chirps, with parameters chosen randomly to expose ALBUS to diverse signal shapes [29].

ALBUS is trained using the mean squared error (MSE) loss function, which quantifies the difference between the predicted and target output maps. The MSE is computed for each pixel, optimizing ALBUS to act as a noise-removal filter that highlights signals [35].

The Adam optimizer is used during training, adjusting the learning rate based on the gradients computed during backpropagation [34]. The model is trained iteratively, and the version with the lowest validation loss is selected for testing.

ALBUS has been tested on various datasets, including synthetic and real-world data, to evaluate its performance in detecting long-duration bursts. The key metrics for evaluation include detection accuracy, glitch discrimination, and robustness to different signal morphologies [29].

ALBUS has shown high accuracy in detecting weak signals, even in the presence of significant noise. It is particularly effective at identifying signals with complex frequency evolutions, such as those from eccentric compact binary coalescences (ECBCs) or magnetar remnants . One of ALBUS's strengths is its ability to differentiate between true GW signals and noise transients. The Glitch Map helps isolate noise artifacts, reducing false positives and improving overall detection sensitivity. When integrated into search pipelines, ALBUS has been shown to enhance the detection of long-duration bursts relative to traditional methods, while minimizing false positives [36].

References

1. Abbott, B.P., et al.: Observation of gravitational waves from a binary black hole merger. Phys. Rev. Lett. **116**, 061102 (2016)
2. Klimenko, S., et al.: Method for detection and reconstruction of gravitational wave transients with networks of advanced detectors. Phys. Rev. D **93**, 042004 (2016)
3. O'Brien, B. et al.: A data-driven machine learning approach to optimize gravitational wave burst detection (2021). arXiv, eprint: 2112.10956
4. Gossan, S.E., et al.: Core-collapse supernovae: gravitational waves from realistic 3D models and an optimal model-independent detection strategy. Phys. Rev. D **93**, 042002 (2016)

5. Abbott, B.P. et al.: GWTC-3: compact binary coalescences observed by LIGO and Virgo during the second part of the third observing run (2020). arXiv, eprint: 2111.03606
6. Abbott, B.P., et al.: All-sky search for short-duration gravitational wave bursts in the third Advanced LIGO and Advanced Virgo observing runs. Phys. Rev. D **101**, 102001 (2020)
7. Mishra, T. et al.: Optimization of model independent gravitational wave search using machine learning (2021). arXiv, eprint: 2105.04739
8. Abbott, B.P., et al.: Constraints on cosmic strings using data from the first Advanced LIGO observing run. Phys. Rev. D **97**, 102002 (2018). https://doi.org/10.1103/PhysRevD.97.102002
9. Abbott, R., et al.: Phys. Rev. X **11**, 021053 (2021)
10. Abbott, B.P., et al.: Observation of gravitational waves from a binary black hole merger. Phys. Rev. Lett. **116**, 061102 (2016). https://doi.org/10.1103/PhysRevLett.116.061102
11. Abbott, B.P., et al.: GW150914: the advanced LIGO detectors in the era of first discoveries. Phys. Rev. Lett. **116**, 131103 (2016)
12. Klimenko, S., et al.: A model-independent method for gravitational wave burst searches. Phys. Rev. D **83**, 102001 (2011)
13. Siemens, X., et al.: Gravitational wave bursts from cosmic (super)string cusps. Phys. Rev. D **76**, 104006 (2007)
14. Damour, T., Vilenkin, A.: Gravitational wave bursts from cusps and kinks on cosmic strings. Phys. Rev. D **64**, 064008 (2001)
15. Abbott, B.P. et al.: Search for gravitational wave bursts associated with gamma-ray bursts during the third observing run of LIGO-Virgo (2021). arXiv, eprint: 2010.14533
16. Lopez, D., et al.: Utilizing Gaussian mixture models in all-sky searches for short-duration gravitational wave bursts. Phys. Rev. D **105**, 063024 (2022). https://doi.org/10.1103/PhysRevD.105.063024
17. Smith, L., et al.: The enhancement of Gaussian mixture modelling as an application to the coherent WaveBurst algorithm in the search for short gravitational wave transients (2024). arXiv, eprint: 2407.16414
18. Chen, T., Guestrin, C.: XGBoost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794 (2016). https://doi.org/10.1145/2939672.2939785
19. Anderson, W.G., et al.: Excess power statistic for detection of burst sources of gravitational radiation. Phys. Rev. D **63**, 042003 (2001)
20. Klimenko, S., et al.: Localization of gravitational wave sources with networks of advanced detectors. Phys. Rev. D **83**, 102003 (2011)
21. Macquet, A., et al.: Long-duration transient gravitational-wave search pipeline. Phys. Rev. D **104**, 102005 (2021)
22. Abbott, R., et al.: All-sky search for long-duration gravitational-wave bursts in the third Advanced LIGO and Advanced Virgo run. Phys. Rev. D **104**, 102001 (2021). https://doi.org/10.1103/PhysRevD.104.102001
23. Abbott, B.P., et al.: A guide to LIGO-Virgo detector noise and extraction of transient gravitational-wave signals. Class. Quant. Grav. **37**, 055002 (2020)
24. Sutton, P. J.: Gravitational-Wave Burst Detection: Sources. Banach Center School of Gravitational Waves. Warsaw (2013)
25. Cuoco, E., et al.: Enhancing gravitational-wave science with machine learning. Machine Learning: Sci. Technol. **2**, 011002 (2021)
26. Cornish, N.J., Littenberg, T.B.: Bayeswave: Bayesian inference for gravitational wave bursts and instrument glitches. Class. Quant. Grav. **32**, 135012 (2015)
27. Lopez, M., et al.: Simulating transient noise bursts in LIGO with generative adversarial networks. Phys. Rev. D **106**, 023027 (2021)
28. Abbott, B.P., et al.: All-sky search for long-duration gravitational wave transients in the first Advanced LIGO observing run. Class. Quant. Grav. **35**, 065009 (2018). https://doi.org/10.1088/1361-6382/aaab76
29. Boudart, V., Fays, M.: Machine learning algorithm for minute-long burst searches. Phys. Rev. D **105**, 083007 (2022)

30. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional neworks for biomedical image segmentation (2015). arXiv preprint arXiv:1505.04597
31. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS), vol. 15, pp. 315–323 (2011)
32. LeCun, Y., et al.: Gradient-based learning applied to document recognition. Proc. IEEE **86**, 2278–2324 (1998)
33. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016)
34. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2014). arXiv preprint arXiv:1412.6980
35. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer (2009)
36. Peters, S.: Mémoire Université de Liège. Unpublished master's thesis (2024). https://matheo.uliege.be/handle/2268.2/20153
37. Abbott, B.P., et al.: All-sky search for long-duration gravitational-wave bursts in the third advanced LIGO and Advanced Virgo run. Phys. Rev. D **104**, 102001 (2021)
38. Klimenko, S., et al.: A model-independent method for gravitational wave burst searches. Class. Quant. Grav. **31**, 035022 (2014)
39. Szczepanczyk, M. et al.: Coherent WaveBurst: a pipeline for unmodeled gravitational-wave data analysis (2021). arXiv, eprint: 2105.04739

Chapter 11
Sparse Dictionary Learning for Gravitational-Wave Signal Denoising, Reconstruction and Classification

Miquel Llorens-Monteagudo, Alejandro Torres-Forné, José A. Font, and Antonio Marquina

Abstract This chapter presents recent advancements in the application of Sparse Dictionary Learning (SDL) to gravitational wave (GW) signal processing, specifically in denoising, glitch removal and signal classification. We outline the mathematical framework of SDL and its role in improving GW data analysis by efficiently representing signals with sparse dictionaries. The method's effectiveness is demonstrated in several use cases, including reducing noise in signals from core-collapse supernovae and binary black hole mergers, as well as mitigating transient noise (e.g., blip glitches) in LIGO data. Additionally, we explore the use of Low-Rank Shared Dictionary Learning for classifying GW signals with high morphological similarity, particularly those from different supernova explosion mechanisms. The results underscore the potential of SDL for refining signal recovery and classification, offering new possibilities for enhancing the precision and reliability of GW detections.

M. Llorens-Monteagudo (✉) · A. Torres-Forné · J. A. Font
Departament d'Astronomia i Astrofísica, Universitat de València, Dr. Moliner 50, 46100 Burjassot (València), Spain
e-mail: miquel.llorens@uv.es

A. Torres-Forné
e-mail: alejandro.torres@uv.es

J. A. Font
e-mail: j.antonio.font@uv.es

A. Torres-Forné · J. A. Font
Observatori Astronòmic, Universitat de València, Catedrático José Beltrán 2, 46980 Paterna (València), Spain

A. Marquina
Departament de Matemàtiques, Universitat de València, Dr. Moliner 50, 46100 Burjassot (València), Spain
e-mail: antonio.marquina@uv.es

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_11

11.1 Introduction

Since the beginning of the observational campaigns of the advanced, ground-based GW detectors (Advanced LIGO [1], Advanced Virgo [4], and KAGRA [49]) the amount of data these instruments are collecting is increasing rapidly, along with its complexity. As in many other scientific areas, the challenge of handling the complexity and dimensionality of this data can be tackled through their sparse representation, on the condition of losing as little information as possible.

One technique to achieve such a goal is sparse dictionary learning (SDL) in which the sparse representation of the data is attained through the linear combination of basic elements, atoms, composing a dictionary. The development of algorithms for SDL has been a subject of great interest in the last decades [13, 17, 30, 61, 64]. SDL constitutes an alternative approach to traditional signal representation procedures based on Fourier decomposition or modern representations based on wavelets, chirplets, or warplets.

In this chapter we present an overview of the current status of SDL efforts for GW signal denoising, reconstruction and classification. Generally, SDL-based methods involve two key steps: learning and reconstruction. Initially, a dictionary is learned from a training dataset composed of noise-free signals that are split into patches. This dataset is assumed to approximate the targeted population of signals we aim to recover. The training process modifies the initial dictionary to better represent the training data, according to a proposed objective function, which balances reconstruction accuracy and sparsity of the representation. Both the learning and reconstruction steps involve optimizing a plethora of hyperparameters, each contributing differently to the effectiveness of the process. The reconstruction optimization step then applies the learned dictionary to the training signals injected into detector noise, emulating real input data. Reconstruction hyperparameters are optimized to ensure that the dictionary's reconstructions are as close as possible to the original, noise-free training signals. This is again achieved by enforcing sparsity on the reconstruction vectors. This two-stage process enhances signal denoising and reconstruction by leveraging the patterns captured in the learned dictionary.

11.2 Mathematical Framework

In this section, we review the mathematical foundation upon which SDL-based methods are built. We begin by introducing signal reconstruction, formally referred to as basis pursuit, as it serves as the basis for the learning method. The learning method is a modified and more complex version, where the dictionary itself is introduced as a second variable in the minimization problem.

11.2.1 Basis Pursuit (LASSO)

Mallat and Zhang [31] defined a dictionary as a collection of signals called atoms such that $\mathbf{u} = \mathbf{D}\boldsymbol{\alpha}$, where $\mathbf{u}$ is the signal to be reconstructed, $\mathbf{D}$ is the dictionary, i.e. a matrix composed of a atoms of length l, and $\boldsymbol{\alpha}$ is a sparse a-dimensional vector which contains the coefficients of the representation.

In the context of GW data analysis, the detector strain $\mathbf{h}(t) \in \mathbb{R}^l$ is typically modeled as a superposition of the actual GW signal $\mathbf{u}(t)$[1] plus some additive noise $\mathbf{n}(t)$ following a linear degradation model

$$\mathbf{h}(t) = \mathbf{u}(t) + \mathbf{n}(t)\,. \tag{11.1}$$

Given an overcomplete dictionary $\mathbf{D} \in \mathbb{R}^{l\times a}$, where the number of atoms a is greater than their length l, there is a sparse vector $\boldsymbol{\alpha} \in \mathbb{R}^a$ for which $\mathbf{D}\boldsymbol{\alpha} \sim \mathbf{u}$. Attention must be drawn to the similarity symbol, for the reconstruction $\mathbf{D}\boldsymbol{\alpha}$ ought to be closer to the original signal $\mathbf{u}$ than to the strain $\mathbf{h}$. Because the original signal is usually unknown, finding $\boldsymbol{\alpha}$ involves imposing a limitation to its similarity to the detector's strain. This problem can be written as the minimization of an objective function composed by two terms,

$$\boldsymbol{\alpha} = \arg\min_{\boldsymbol{\alpha}} \left\{ \|\mathbf{h} - \mathbf{D}\boldsymbol{\alpha}\|_2^2 + \lambda_{\text{rec}} \|\boldsymbol{\alpha}\|_1 \right\}\,. \tag{11.2}$$

In this equation the first term is the *error term*, measuring how well the solution fits the data through the L^2-norm. The second term, weighted by a Lagrangian multiplier λ_{rec}, is the *regularization term*. The multiplier λ_{rec} is subscripted to make its specific purpose explicit; however, it can also be used for other purposes, as will be shown in Sect. 11.2.2. The use of the L^1-norm provides a constraint on the number of dictionary atoms used, and the minimization becomes a convex variational problem. This approach is known as *basis pursuit* [13] or *LASSO* (least absolute shrinkage and selection operator) [53]. The regularization term in the L^1-norm promotes zeros in the components of the vector coefficient $\boldsymbol{\alpha}$ and, thus, the solution of this variational problem is typically the sparsest one.

11.2.2 Sparse Dictionary Learning

The prototype signals of a dictionary can be chosen as a predefined set of functions, such as a Fourier basis (frequency dictionaries), wavelet functions (wavelet dictionaries) or Gabor wavelet decomposition (time-frequency dictionaries). However, the idea of using a dictionary learned from data has led to significant improvement in signal denoising and reconstruction [17]. Nowadays, SDL algorithms are being

[1] For simplicity we use *GW signal* to refer to any kind of targeted waveform observable in a GW detector, including non-astrophysical transients such as glitches.

developed along this direction [30], and very efficient methods have been devised to solve the optimization problem inherent to learning dictionaries.

Consider a training dataset from which p patches of length l are extracted, $\boldsymbol{U} = [u_1, \ldots, u_p] \in \mathbb{R}^{l \times p}$, using a random windowing function. In general the number of training patches is large compared with the number of atoms and their length, $p \gg a > l$, because each signal only uses a few elements in $\boldsymbol{D}$ for the representation. The trained dictionary is then obtained by adding the dictionary matrix $\boldsymbol{D}$ as a variable in the minimization problem,

$$\boldsymbol{D} = \underset{\boldsymbol{\alpha}, \boldsymbol{D}}{\arg\min} \frac{1}{l} \sum_{i=1}^{p} \left\{ \|\boldsymbol{u}_i - \boldsymbol{D}\boldsymbol{\alpha}_i\|_2^2 + \lambda_{\text{learn}} \|\boldsymbol{\alpha}_i\|_1 \right\} , \tag{11.3}$$

where the summation index i indicates the i-th row of the matrix $\boldsymbol{\alpha} \in \mathbb{R}^{p \times a}$, which contains the coefficients of the sparse representation of each atom. The value of the regularization parameter λ_{learn} in this context only affects the sparsity of the learned atoms. Said atoms, $\{\boldsymbol{d}_i\}_{i=1}^{l}$, are constrained to have an L^2-norm less or equal to one, $\boldsymbol{d}_i^T \boldsymbol{d}_i \leq 1$, to prevent $\boldsymbol{D}$ from being arbitrarily large.

This minimization problem is solved by the algorithm proposed by Mairal et al. in [29] with the mini-batch optimization. This is a block-coordinate descend method which minimizes $\boldsymbol{D}$ and $\boldsymbol{\alpha}_i$ separately for each iteration t,

$$\boldsymbol{\alpha}^t = \underset{\boldsymbol{\alpha}}{\arg\min} \left\{ \frac{1}{2} \|\boldsymbol{u}_t - \boldsymbol{D}^{t-1}\boldsymbol{\alpha}\|_2^2 + \lambda_{\text{learn}} \|\boldsymbol{\alpha}\|_1 \right\} , \tag{11.4}$$

$$\boldsymbol{D}^t = \underset{\boldsymbol{D}}{\arg\min} \frac{1}{t} \sum_{i=1}^{t} \left\{ \frac{1}{2} \|\boldsymbol{D}\boldsymbol{\alpha}_i^t - \boldsymbol{u}_i\|_2^2 + \lambda_{\text{learn}} \|\boldsymbol{\alpha}_i\|_1 \right\} , \tag{11.5}$$

with the advantage of being parameter-free and not requiring any learning rate. Additionally, the minimization can be sped up by setting the initial value of the atoms in $\boldsymbol{D}^0$ to random segments of the training data.[2]

The λ_{learn} from the learning step is a separate hyperparameter, and its optimal value is therefore not related to the λ used in the sparse reconstruction, referred to as λ_{rec}.

[2] Given enough iterations, the initial conditions of both $\boldsymbol{\alpha}$ and $\boldsymbol{D}$ do not have a significant impact in the final performance of the method due to the convex nature of the problem.

11.3 Overview of SDL Applications in GW Data Analysis

11.3.1 Denoising of GW Signals

SDL was first used for GW data analysis by Torres-Forné et al. in [55], where the capabilities of learned dictionaries to denoise GW signals was assessed. The datasets employed comprised two catalogs of GW signals, namely a catalog of 128 waveforms from rotational core collapse supernova (CCSN) simulations [16] and a second one from BBH simulations [33] containing 174 waveforms, from which only the first 100 were used.

Dictionaries were trained and optimized in the same way for both catalogs: each waveform catalog was split into 3 subsets; 80% of the data was used for initializing and training the dictionary, 15% for validation, and the remaining 5% for testing. Signals were resampled to 16384 Hz and injected in non-white Gaussian noise with the power spectral density (PSD) of the Advanced LIGO proposed broadband configuration [1], with frequencies ranging from 10 Hz to 8192 Hz. Validation and test signals were injected at a constant signal-to-noise ratio (SNR), defined as

$$\mathrm{SNR} = \sqrt{4\Delta t^2 \Delta f \sum_{k=1}^{N_f} \frac{|\tilde{h}(f_k)|^2}{S(f_k)}}\,, \tag{11.6}$$

where $\tilde{h}$ indicates the Fourier transform of signal strain h, S is the sensitivity curve of the detector expressed as a PSD, f_k is each of the components of the frequency vector, N_f is the number of positive frequencies, and Δt and Δf are the time step and frequency step, respectively. Test signals were injected 20 times each at 20 SNR, with different noise realizations in order to study the variability introduced by the background noise. The dictionary was trained selecting 30000 random patches uniformly distributed from all the learning subset. The hyperparameters to be optimized were the sample length l and the number of atoms a. We define the optimal regularization parameter λ_{opt} as the value that yields the best reconstructions, determined by comparing the reconstructed signals to the original waveforms according to two loss functions; the mean squared error (MSE) and the structural similarity index measure (SSIM) [59]. In [55] it was found that in general the optimum size of the dictionary was larger for the BBH dataset than for the CCSN catalog, due to BBH waveforms being longer. Moreover, when atoms were too short w.r.t. the size of the original signal, the reconstruction was more oscillatory due to the background noise. On the contrary, if the window was too big, it became more difficult to reconstruct the smallest oscillations from the original signal.

In spite of using the mean value of the regularization parameter for all signals (of the two catalogs), and despite each one being injected into a different noise realization, the dictionaries were able to reconstruct them relatively well. The error estimates are summarized in Table 11.1. A visual example of the denoising is shown in Fig. 11.1, where the best (left) and worst (right) reconstructions from the CCSN catalog are

Table 11.1 Summary of best and worst reconstructions of CCSN and BBH signals, in terms of MSE and SSIM as error estimators, for two SNR values. Since each signal at a given SNR was injected with 20 different noise realizations, only the best and worst values are shown in brackets

Signal		SNR 20		SNR 10	
		MSE ($\times 10^{-3}$)	SSIM	MSE ($\times 10^{-3}$)	SSIM
CCSN	Worst	[0.210−0.084]	[0.83−0.72]	[0.861−0.205]	[0.72−0.51]
	Best	[0.033−0.015]	[0.97−0.93]	[1.389−0.021]	[0.96−0.74]
BBH	Worst	[0.060−0.027]	[0.86−0.79]	[0.104−0.039]	[0.83−0.66]
	Best	[0.025−0.019]	[0.89−0.86]	[0.084−0.027]	[0.87−0.76]

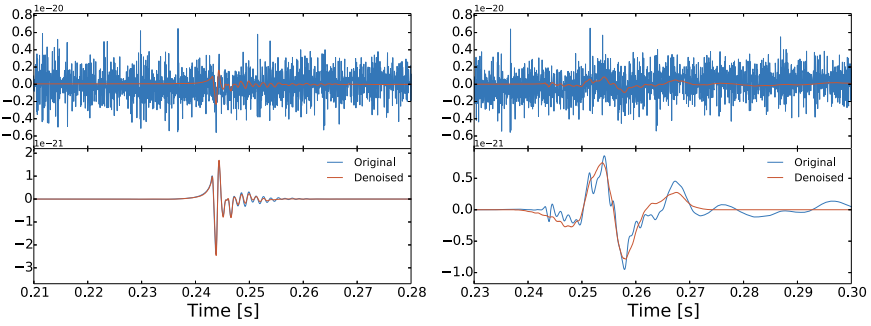


Fig. 11.1 Denoising example of two signals from the CCSN test subset. Both are injected at SNR 20, and reconstructed with their respective dictionary at the mean optimal hyperparameters. Upper panels: noisy signals (blue) superimposed with the original waveform (red). Lower panels: comparison between denoised signals (red) with the originals (blue). Their MSE and SSIM values are 0.018×10^{-3} and 0.98 for the signal on the left panel, and 0.271×10^{-3} and 0.67 for the signal on the right panel. *Reproduced with permission from* [55]. *Copyright 2016 by the American Physical Society*

compared to the noisy injections. In the best case, the positive and negative peaks associated with the hydrodynamical bounce following core collapse were accurately recovered, although when the signal amplitude decreased becoming weaker than the noise, the dictionary yielded a null reconstruction. In contrast, in the worst case the small oscillations were missed, yet the overall morphology was properly captured. This happened for signals containing morphological traits notably different from the common features of the CCSN dictionary. The individual results could be improved by lowering the value of λ_{rec}, effectively decreasing the sparsity of the reconstruction and thus increasing the number of atoms used by the dictionary.

About the same conclusions were drawn for the BBH datasets; signals were well recovered during the three distinctive parts, namely the inspiral, merger and ringdown. In particular, the worst case depicted in Fig. 11.2 shows that the phase was well captured and the main differences w.r.t. the original waveform appeared only in the amplitude. This was partially due to the original signal being longer towards the left side of the inspiral than what was intended to be captured by the dictionary.

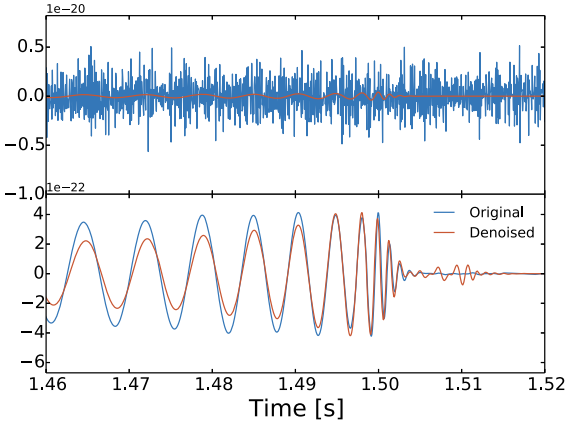


Fig. 11.2 Denoising example of a test signal from the BBH catalog, injected at SNR 20. The upper panel shows the noisy signal (blue) superimposed with the original waveform (red), and the lower panel compares the denoised signal with the original. Their MSE and SSIM values are 0.031×10^{-3} and 0.86 respectively. *Reproduced with permission from* [55]. *Copyright 2016 by the American Physical Society*

The spurious oscillations visible after the ringdown indicate that the optimum value of λ_{rec} is probably greater than the used.

It is also worth mentioning that the results can be improved if the reconstruction is done in more than a single step. This was shown in [55] by reconstructing a CCSN signal at SNR 6 with a two-step process. First a lower λ_{rec} than the optimum was chosen in order to recover the signal and part of the noise. Then, the arrival time of the signal was estimated using a spectrogram, and in the final step the denoising was performed again through iterative reconstructions to minimize (maximize) the MSE (SSIM). In the example shown in [55] this resulted in an accurate retrieval of the GW signal comparable to that obtained from a injection at SNR 10.

A denoising test of SDL using real data was performed in [55] using signal GW150914 [26]. The data was preprocessed as little as possible, performing a single highpass above 30 Hz to remove seismic noise and applying a specific filter to remove all spectral lines characteristic of the Advanced LIGO detector at the time of the detection. The reconstructed signal is depicted in Fig. 11.3, setting $\lambda = 0.004$ for the BBH dictionary. While visually the denoised signal seems in remarkable good agreement with the numerical relativity waveform in the last cycles of the inspiral, the merger, and the ringdown, the estimators (computed at ± 0.15 s from the minimum of the NR signal) indicate only a modest accuracy, MSE = 0.0075 and SSIM = 0.4901 .

Finally, we note that more recent results on the application of SDL for GW denoising have been reported in [7]. This paper shows that SDL is an excellent approach to detect and reconstruct GW signals from massive black hole binaries (in the mHz frequency range accessible to LISA) buried in the Galactic foreground noise associated with mergers of white-dwarf binaries.

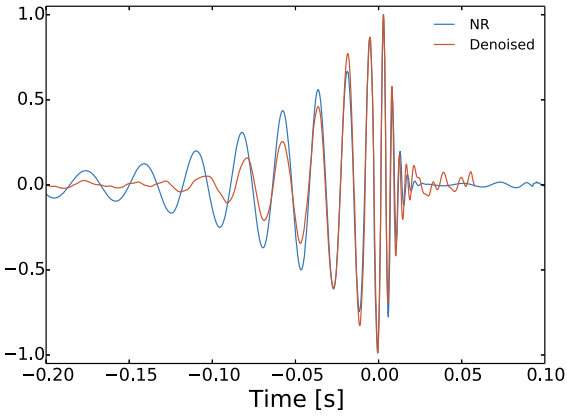


Fig. 11.3 Denoising of signal GW150914 detected by Advanced LIGO Hanford interferometer using the BBH dictionary. The blue line indicates the NR template and the red curve corresponds to the reconstructed signal. The amplitude of both signals has been rescaled to lie in the interval $[-1, 1]$. *Reproduced with permission from* [55]. *Copyright 2016 by the American Physical Society*

11.3.2 Removal of Blip Glitches

Apart from the main noise contributions making difficult the detection in ground-based GW detectors, such as seismic noise, thermal noise and quantum noise, which are relevant at low ($\mathcal{O}$(Hz)), medium ($\mathcal{O}$(100 Hz)) and high ($\mathcal{O}$(kHz)) frequencies, respectively, short transient noise signals, typically referred to as *glitches*, also abound. One of the main noise characterization efforts in the LVK detector network is focused on these transient sources of noise. While their origin is instrumental and environmental, they often mimic potential astrophysical sources with a wide variety of morphologies [2, 3]. Even further, they can pollute actual GW signals as happened in the case of the BNS signal GW170817 [19].

Following [55] Torres-Forné et al. [56] showed how learned dictionaries can be applied to mitigate the impact of glitches in real data from the Advanced LIGO detectors. The study was focused on blip glitches, a type of noise transient with a typical duration of 10 ms and a frequency bandwidth of 100 Hz. Blip glitches are one of the worst noise contributors in the Advanced LIGO detectors which significantly reduce the sensitivity of searches for high-mass CBC [12]. In [56] 100 blips were selected from the data stream of Advanced LIGO's first observing run (O1). Each glitch was contained in a one-second window centered at the GPS time of the glitch as recorded in Gravity Spy [63]. Given that detector's data is populated by a wide variety of noise, as stated before, all data was whitened (using the autoregressive model of [14, 15]) to remove all systematic sources of noise and flatten the data in frequency. The study employed 85% of the data for training the dictionaries and 15% for testing. A single denoising dictionary was sufficient for this task, initialized from a single glitch split up into windows of 1024 samples, and trained over 30000 random

patches from all the training signals. Dictionary hyperparameters were optimized following the same procedure of Sect. 11.3.1, with the added difficulty of not knowing how "original" blips are supposed to look like. To address this issue two different estimators were introduced as loss functions; the first one defines an alternate SNR by comparing the spectrum of the glitch segment to the average spectrum of the data,

$$\mathrm{SNR} = \frac{\max(S(t))}{\overline{S(t)}}, \quad S(t) = \int_{20}^{\frac{f_s}{2}} S(t, f)\, df, \tag{11.7}$$

and the second one measures the spectral flatness of the given sample again w.r.t. the glitch-free background,

$$W = \frac{\exp\left(\frac{1}{f_s}\int_{-f_s/2}^{f_s/2} \ln(P(f))\, df\right)}{\frac{1}{f_s}\int_{-f_s/2}^{f_s/2} P(f)\, df}\,. \tag{11.8}$$

The best results were obtained with 192 atoms of 128 samples each, with an added step consisting on smoothing the signals in order to improve the convergence of the learning step and reduce spurious oscillations inside atoms, achieved through the use of the rROF method [47, 54]. The optimum value of the regularization parameter λ_{rec} was defined w.r.t. both estimators using the iterative reconstruction procedure introduced in the previous section, with the goal of extracting the glitch while keeping the background unaltered. For this, however, a variation was needed; instead of waiting for the i-th reconstruction to be lower than a given threshold, the stop condition was set to guarantee zeros at the extremes of the signal where only background noise was present while still retrieving components of the glitch at the center of the window.

Figure 11.4 shows the time-frequency diagram of three illustrative examples of blip glitches from the test set reconstructed with the optimum parameters. In all three (and for the whole test set) the background was kept almost untouched when no glitch was present, with only the glitches themselves being partially recovered. Their impact was significantly reduced, although some parts often remained (see middle and right panels). We note that the recovery was not homogeneous in all frequencies; the power was reduced for low and middle frequencies up to $\sim$ 500 Hz, while higher frequencies proved to be challenging for the dictionary. Furthermore, in the right column of Fig. 11.4 the morphology of the spectral line around $\sim$ 1300 Hz, which is clearly not part of the glitch, was left unaffected. This is a good feature of the dictionary in situations were a blip is overlapped with an actual GW, as the method would be capable of leaving the latter untouched. In this example, however, the spectral line is another source of noise and thus requires an additional technique to mitigate its impact, which motivated the next procedure.

To improve the aforementioned shortcomings an additional method was proposed in [56] by combining the iterative reconstruction of the initial dictionary with a second dictionary applied to the initial residuals. Being the target high-frequency

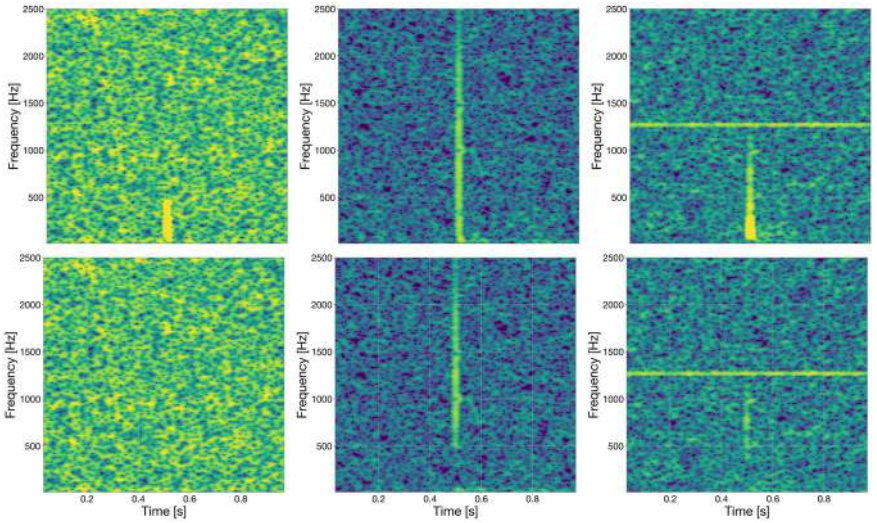


Fig. 11.4 Time-frequency diagram (spectrogram) of three illustrative examples of blip glitches, with data previously whitened to flatten the detector's intrinsic background noise and spectral lines in order to visually highlight glitches. The original data and the residuals are represented in the upper and bottom panels, respectively. *Reproduced with permission from* [56]. *Copyright 2020 by the American Physical Society*

components of the glitch, the second dictionary was built with smaller atoms of 16 samples, with 40 atoms sufficing for the overcomplete condition, and a lower value of λ_{rec}. Since these settings would inevitably increase the partial recovery of spurious oscillations from the background noise, the reconstruction window for the second dictionary was restricted to the most significant part of the blip within the first reconstruction. This resulted in an effective reduction of the glitch remains up to ~ 1 kHz, leaving the background noise relatively unaltered. Numerical results from both estimators (see Table 11.2) revealed that when the first denoising did not remove a glitch completely, the second dictionary was able to improve the results. Therefore, glitch denoising with multiple dictionaries is a convenient strategy and had a negligible increment in computational cost.

Finally, [56] applied their deglitching method to the BNS signal GW170817 [19]. In this detection, a short instrumental noise transient appeared in the LIGO-Livingston detector data about 1.1 s before merger, partially blocking the inspiral segment. This unfortunate event provided a good opportunity to asses the capabilities of SDL to reconstruct this glitch and to analyze the impact of the algorithm on the actual astrophysical signal. The results are shown in Fig. 11.5. The spectrograms were computed using the Q-transform [20] in order to facilitate the comparison with the LVK results [19]. We note that the shape of this glitch is notoriously different from those included in the training set used in [56]. Thus, the denoising dictionary was not specifically tailored to deglitch this particular noise transient. Furthermore,

Table 11.2 Quantitative assessment of the deglitching procedures. The columns report the values of the estimators, SNR and W, for the data containing the original noise transients (subindex 'o') and for the residuals after deglitching, and both for a single dictionary (subindex 'single') and for multiple dictionaries (subindex 'multi'). *Reproduced with permission from* [56]. *Copyright 2020 by the American Physical Society*

Test #	SNR_o	SNR_{single}	SNR_{multi}	W_o	W_{single}	W_{multi}
1	5.3	1.3	1.3	0.99	0.99	0.99
2	5.2	2.4	1.3	0.94	0.94	0.93
3	9.2	1.3	1.3	0.99	0.99	0.99
4	37.1	10.2	1.5	0.84	0.92	0.97
5	18.1	2.3	1.7	0.99	0.98	0.98
6	13.5	3.8	1.7	0.98	0.98	0.98
7	4.1	1.3	1.3	0.98	0.98	0.98
8	6.3	1.3	1.3	0.96	0.97	0.97
9	13.4	9.8	2.9	0.98	0.98	0.98
10	8.7	3.0	1.7	1.00	0.99	0.99
11	7.3	1.3	1.3	0.99	0.99	0.99
12	5.8	1.3	1.2	0.99	1.00	1.00
13	4.5	1.8	1.3	0.99	0.99	0.99
14	14.2	1.2	1.2	0.99	0.99	0.99
15	15.2	2.2	1.3	0.88	0.88	0.89
16	6.2	1.3	1.3	0.99	0.99	0.99

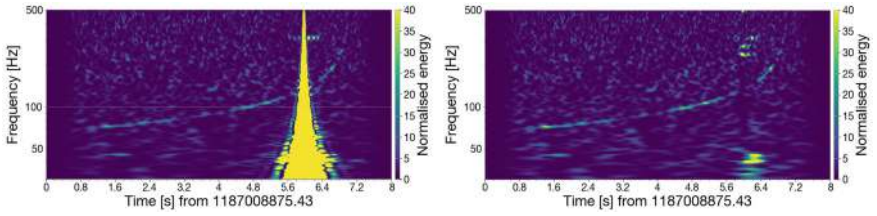


Fig. 11.5 Time-frequency diagrams of 8 s of data corresponding to the GW170817 signal. The left panel shows the original data from LIGO-Livingston. The right panel displays the data after subtracting the glitch using a single blip-trained dictionary with 256 samples. *Reproduced with permission from* [56]. *Copyright 2020 by the American Physical Society*

the duration of the glitch was significantly longer. Hence, a dictionary with longer atoms (256 samples) was employed while keeping the other hyperparameters the same, which in turn increased the uncertainty about the dictionary used being optimal. Nevertheless, as the right panel of Fig. 11.5 shows, the glitch was removed from the data for the most part while recovering almost the totality of the inspiral signal. The remaining bits of the glitch around frequencies $\sim$400 Hz and below $\sim$50 Hz

were not removed after applying a second smaller dictionary. This could be potentially improved if the dictionaries were trained with a larger set of glitches from different morphologies.

11.3.3 Classification of Glitches in Simulated Noise

SDL has been used for classification tasks in image processing applications [28, 45] such as face recognition [60] or texture classification [37]—but also in signal classification [18, 22]. On those fields, more complex approaches presented in recent years have shown improved results in comparison [27, 51, 52]. In the realm of GW data analysis, SDL algorithms have just begun to be used for classification tasks [25, 43, 48].

A simple SDL classification algorithm was presented by Llorens-Monteagudo et al. [25] to reconstruct simulated GW glitches injected in Gaussian noise. Based on the well-known performance of the dictionaries and the quality of their reconstructions, a first denoising step was introduced to highlight the morphological traits of a given input signal over the background noise of the detector. This step was repeated using as many dictionaries $\{\boldsymbol{D}_i\}_{i=1}^{c}$ as signal categories (c), hence obtaining slightly different reconstructions $\{\boldsymbol{p}_i\}_{i=1}^{c}$ for each input signal. These are referred to as *parent* reconstructions. The second step is where the actual discrimination takes place. During the training phase, multiple pools of example signals were built in advance, each corresponding to a denoising dictionary / category and all of them padded to the same length. These collections served as generic references, illustrating the expected reconstructions of each dictionary. For each reconstruction $\boldsymbol{p}_i$ the closest n_p examples were selected from each pool of sample signals, and linearly combined into a single one per category. After this step the number of signals is c times the number of reconstructions, leaving a tree-like structure of signals per input, as shown in Fig. 11.6. This second batch of signals was referred to as *children* reconstructions.

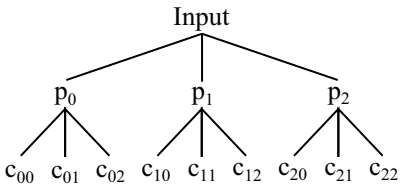


Fig. 11.6 Example of a tree diagram of all reconstructions $\boldsymbol{p}_i$ and sample signals $\boldsymbol{c}_{i,j}$ which would be generated after the second step of a given input signal if only three different categories are considered ($c = 3$). Figure from [25]

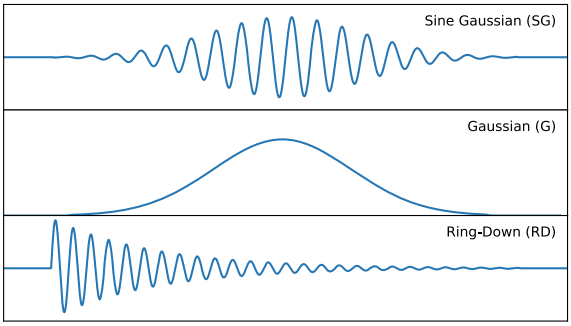


Fig. 11.7 Examples of each waveform morphology included in the data set, in the time domain. Figure from [25]

The predicted category was chosen according to

$$i_{\text{dict}} = \arg\min_i \prod_j \frac{1 - \text{SSIM}(p_j, c_{ji})}{2}, \tag{11.9}$$

where the SSIM acts as a loss function. This allows to find the dictionary that yields the most "verisimilar" reconstructions during step 1, under the assumption that the dictionary whose category coincides with the input signal will produce reconstructions closer to the corresponding reference pool, while not deviating much from the rest of pools than the rest of the dictionaries with their respective pools.

The study of [25] employed the synthetic noise transients (glitches) of [38]. The dataset comprised three different type of glitches, namely Sine Gaussian (SG), Gaussian (G), and Ring-Down (RD), with frequencies ranging from 40 to 1500 Hz. Figure 11.7 depicts an example of each category in the time domain. Gaussian glitches were introduced as a simple and well differentiated morphology, while SG and RD are relatively more complex and similar between them. The training dataset was built by fixing the amount of training patches p to 20000 instead of the number of complete signals. This was done because Gaussian signals are almost 2 orders of magnitude shorter than the other two categories. Hence, if the number of signals were set equal for all categories then Gaussian signals would be under-represented in terms of samples. On the other hand, for the validation and test subsets a total of 300 glitches (100 per category) were generated for parameter optimization, and 3000 more for the final test, guaranteeing that the statistical work related to classification was performed with balanced data sets.

Validation and test signals were injected into non-white Gaussian noise at SNR $=$ 20, using the definition in Eq. (11.6). However, training signals were kept noise-free because it was found that training the dictionaries with noisy patches produced reconstructions bloated with spurious oscillations in the case of the Gaussian signals. This was attributed to their short duration, which translated to being represented by so few samples that makes them less discernible from the background noise.

Table 11.3 Optimum values of all hyperparameters used for the final classification test in [25]. Each row corresponds to a single type of glitch

Category	λ_{learn}	a	l	λ_{rec}	n_p	a_p
Sine Gaussian	0.02	512	256	0.09	1	256
Gaussian	0.006	256	128	0.09	1	256
Ring-Down	0.01	512	256	0.09	1	256

The optimization of the hyperparameters was done in a greedy fashion. Part of those parameters (the learning parameter λ_{learn}, the length of atoms l, and the number of atoms a) were optimized to produce the best reconstructions according to the MSE estimator. However, [25] found that classification dramatically improved when optimizing the parameter of the reconstruction λ_{rec} w.r.t. the classification itself instead of focusing on the accuracy of the denoising. In general, the optimum value for classification turned out to be lower than the optimum for reconstruction. The remaining hyperparameters related to the *children* reconstructions, namely the number of sample signals in each pool a_p and the number of sample signals used per child n_p, were trivially chosen as those which maximized the accuracy of the classification at a reasonable computational cost. The values are reported in Table 11.3. It is worth noting that the optimal value for n_p was the lowest, which is consistent with the goal of the sample pools, i.e. providing a strict discriminating selection.

Test glitches were injected in noise at random SNR values between 1 and 400 to add more variety. From all 3000 glitches 96% were classified correctly. As shown in the confusion matrix of Fig. 11.8, the Gaussian dictionary was the most successful with only one lost glitch. This was attributed to the morphological simplicity of Gaussian glitches which, added to their comparatively short duration, make them hard to replicate by the other more complex morphologies, and conversely easier to use for replicating those. Misclassifications between sine-Gaussian and Ring-Down glitches were expected due to their similarity, although the asymmetry in results is worth a mention. SG glitches of high frequency have shorter durations which, combined to spurious oscillations of the background noise, can make them more

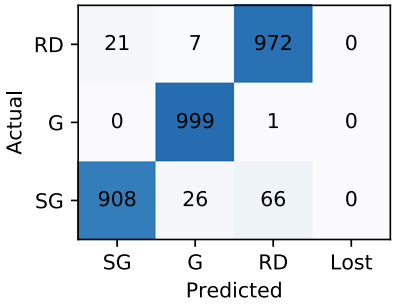


Fig. 11.8 Confusion matrix showing the classification results of the test set. Rows correspond to the actual morphology of test glitches, and columns to the morphology predicted by classification dictionaries. Figure from [25]

similar to RD glitches when injected at low SNR. This, however, is more confusing for the RD dictionary than for the SG dictionary because the latter will need more atoms to replicate a sudden increase in amplitude due to its designed morphology.

11.3.4 Classification of the CCSN Explosion Mechanism

A recent application of SDL for the classification of the CCSN explosion mechanism has been reported by Powell et al. [43]. A learned dictionary dubbed LRSDL (Low-Rank Shared Dictionary Learning) [57], specifically designed for classification, was used in this investigation. This algorithm extends the Fisher Discrimination Dictionary Learning (FDDL) [62] by capturing shared components among diverse waveform classes. It has demonstrated strong performance in scenarios with limited training samples, surpassing previous dictionary-based algorithms in computational efficiency. The classification dictionary $\overline{\boldsymbol{D}} \equiv [\boldsymbol{D} \quad \boldsymbol{D}_0]$ and reconstruction vector $\overline{\boldsymbol{X}} \equiv [\boldsymbol{X}^T, (\boldsymbol{X}^0)^T]^T$ exhibit a more complex structure compared to the denoising dictionary. The dictionary comprises two "type" of matrices; the first type represents the class-specific features and thus there are as many as classes, $\{\boldsymbol{D}_i\}_{i=1}^{C}$, while the second type captures the shared parts among two or more classes, hence being only one, $\boldsymbol{D}_0$.

During the learning process, for a given input waveform $\boldsymbol{Y}$, the method tries to reconstruct it by linearly combining atoms from both types of dictionaries such that

$$\boldsymbol{Y} \approx \boldsymbol{D}\boldsymbol{X} + \boldsymbol{D}_0\boldsymbol{X}^0\,, \tag{11.10}$$

imposing different restrictions to each part of the dictionary. For the class-specific part FDDL constraints are imposed, while the shared part $\boldsymbol{D}_0$ is enforced to be low-rank in order to avoid the assimilation of class-specific features, and the components of the shared code vector $\boldsymbol{X}^0$ to be close to each other so that its contribution to each class is as homogeneous as possible.

This leads to a total of six hyperparameters, with half of them representing the physical size of dictionaries $\boldsymbol{D}_c \in \mathbb{R}^{l \times a_C}$ and $\boldsymbol{D}_0 \in \mathbb{R}^{l \times a_0}$, and the remaining half being the regularization parameters λ_1 (sparsity of $\boldsymbol{X}$), λ_2 (sparsity of $\boldsymbol{X}$ and homogeneous contribution of $\boldsymbol{X}_0$), and η (low-rank regularization of $\boldsymbol{D}_0$).

The classification of an unknown input signal $\boldsymbol{Y}$ is performed first by reconstructing it as in (11.10), finding the coefficient vector $\overline{\boldsymbol{X}}$ with the sparsity constraint and homogeneity contribution of $\boldsymbol{X}_0$. Afterwards, the contribution of the shared dictionary is extracted from the input signal, $\overline{\boldsymbol{Y}} = \boldsymbol{Y} - \boldsymbol{D}_0\boldsymbol{X}^0$, and finally the predicted index of the class is determined by the class-specific dictionary which "adapted" better to $\overline{\boldsymbol{Y}}$,

$$\arg\min_{1 \le c \le C} \left(w||\overline{\boldsymbol{Y}} - \boldsymbol{D}_c\boldsymbol{X}^c||_2^2 + (1-w)||\boldsymbol{X} - \boldsymbol{m}_c||_2^2 \right)\,, \tag{11.11}$$

where $\boldsymbol{m}_c$ is the mean vector of $\boldsymbol{X}_c$, and w is a weight to balance the contribution between the similarity term and the homogeneity contribution term. It is worth noting that because of how the classification is performed, the input signal cannot be split into smaller windows. Hence, all atoms in the classification dictionary must be of the same length of the signals to be classified, as opposed to the denoising dictionary whose atoms' length can be optimized.

Alongside two other algorithms (based on Bayesian model selection and convolutional neural networks), the LRSDL dictionary was used in [43] to identify the explosion mechanism of simulated CCSN by their GW morphology alone. Several mechanisms have been proposed and two of those were included in the datasets of [43]. The most usual explosion mechanism is thought to be neutrino-driven [23]. GWs from massive stars that explode by this mechanism contain most of their energy in f- or g-modes with frequencies above $\sim$500 Hz [32, 35, 36, 39, 44], and may also contain low-frequency modes due to the standing accretion shock instability (SASI) [9, 58], which typically fade away before shock revival. An example is shown in the top right spectrogram of Fig. 11.9. The second mechanism, named magneto-rotational, is expected to be an additional contribution to the neutrino-driven mechanism in rapidly rotating CCSN explosions with powerful magnetic fields [8, 34, 42]. GWs from these sources may present large broadband spikes at the time of the core bounce (bottom left spectrogram in Fig. 11.9), and change the relationship between the f-/g-modes and the mass and radius of the proto-neutron star. A third category was included in [43] for the case of non exploding collapse, which corresponds to a situation in which the shock wave does not gain enough energy to be revived. Nevertheless, GWs are still emitted in this case before black hole formation, with

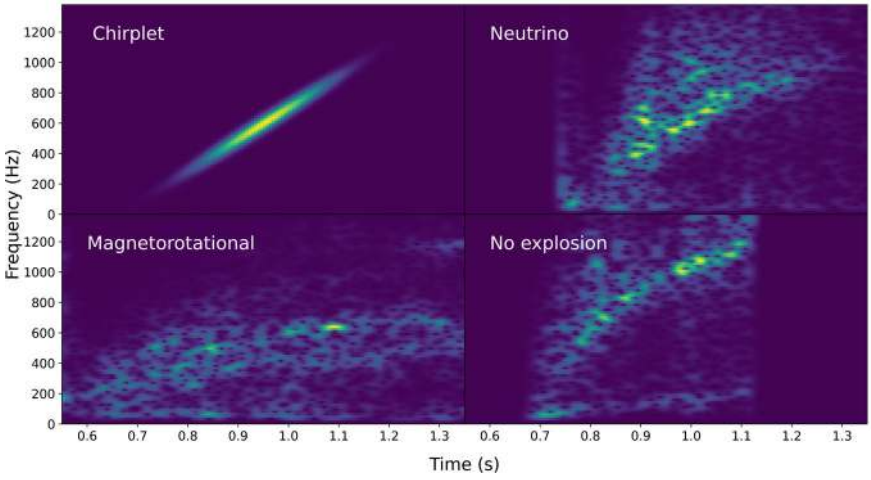


Fig. 11.9 An time-frequency representation of each of the four types of CCSN waveforms used in the dataset. *Reproduced with permission from* [43]. *Copyright 2024 by the American Physical Society*

Table 11.4 Summary of the original (noiseless) signals included in the training and test dataset. The following abbreviations are used: Neu (neutrino-driven), Mag (magneto-rotational), Nox (nonexploding), and Chi (chirplet). A detailed description of each model can be found in [43]

Category	Training		Test	
	#	Models	#	Models
Neu	8	`s40`$_{\text{FR}}$ [36] $10M_{\odot}$, $11M_{\odot}$, $19M_{\odot}$, $60M_{\odot}$ [44] `y20` [40] `z85_SFHo`, `z85_SFHx` [41]	3	$18M_{\odot}$ [39] $12M_{\odot}$ [44] `m39` [40]
Mag	8	`m39_B10` [42] `O`, `P`, `W` [34] `l1_90d`, `l2_gB`[11] `A26`, `A39` [6]	2	`m39_B12` [42] `A13` [6]
Nox	7	`s40NR` [36] `mesa20_gw`, `mesa20_pert_gw`, `mesa20_LR_gw` [35] $13M_{\odot}$ [44] `z100 (SFHo, SFHx)` [41]	2	`s18np` [40] `C15` [32]
Chi	10	2×(600,650,750,800,850) Hz	2	700 Hz

the most distinctive feature being the low-frequency SASI component lasting longer. Finally, to study the response of the classification algorithm when confronted with a waveform whose morphology did not match the previous classes, a fourth category was included, namely synthetic chirplet glitches (top left spectrogram in Fig. 11.9).

Table 11.4 summarizes the train and test datasets used in [43]. In order to study different scenarios, test signals were injected in noise from three different GW detectors: Advanced LIGO, Einstein Telescope and the proposed Australian high-frequency detector NEMO. The first two were generated by recoloring real data from the LIGO Livingston detector (during the third observing run, a segment starting at GPS time 1238179840 and spanning for 4096 s) to the respective detector's design sensitivities (ref. Fig. 11.10). On the other hand, the NEMO detector's noise was simulated from Gaussian noise because of the disparity between its frequency sensitivity and that of the other two detectors. The signals were injected in 10 s intervals at SNR values of 25, 30, 35, 40, and 45. In total, 368 signals were injected in each detector, 92 nonexploding (Nox), 138 neutrino-driven (Neu), 92 magneto-rotational (Mag) and 46 chirplets (Chi).

Before the LRSDL dictionary was used for classification a previous pre-processing step was applied consisting in whitening the data (as in Sect. 11.3.2) and denoising it with a single learned dictionary. The dictionary was trained with all training signals except chirplets (the unknown/foreign category), making no distinctions between categories since the goal of this dictionary was only to recognize and reconstruct any of them from within detector's data. Furthermore, the classification dictionary was trained by injecting the train signals into Advanced LIGO noise (also from the

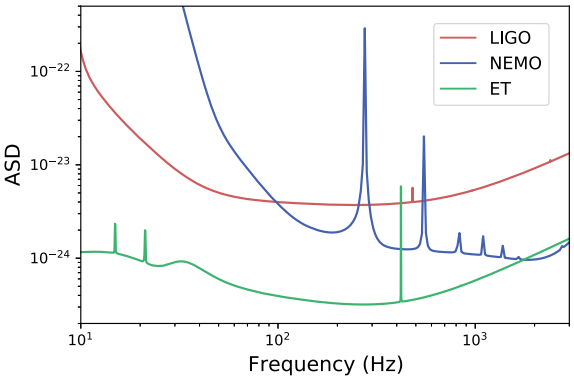


Fig. 11.10 The amplitude spectral density (ASD) curves for the Advanced LIGO design sensitivity [1], Einstein Telescope [21] and NEMO [5] gravitational-wave detectors. *Reproduced with permission from* [43]. *Copyright 2024 by the American Physical Society*

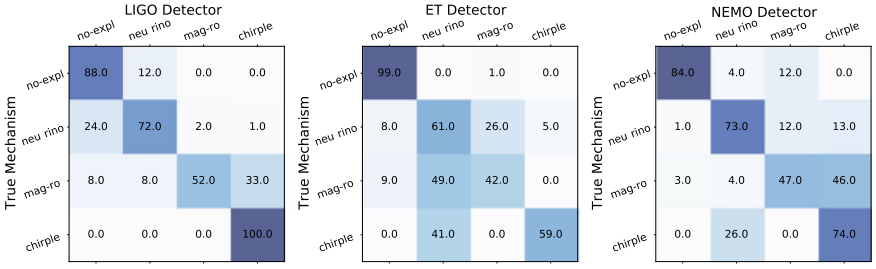


Fig. 11.11 SDL classification results for all detectors. The corresponding accuracy is 78% for Advanced LIGO, 65.3% for the Einstein Telescope, and 69.5% for the NEMO detector. *Reproduced with permission from* [43]. *Copyright 2024 by the American Physical Society*

third observing run but at only two different GPS times not used for testing and with no recoloring whatsoever) and denoising them with the aforementioned denoising dictionary, this time with the data properly labeled. As a final note, the LRSDL dictionary by default was not able to manage an unknown type of signal (it would always classify any input), therefore its code was modified by adding a confidence threshold in its loss. A signal which yielded a value surpassing this threshold would be considered not recognized as any of the three main categories (Nox, Neu, Mag) and thus classified (in this case) as a chirplet.

Results from the final classification are shown in the confusion matrices of Fig. 11.11. The best performance was found with the Advanced LIGO detector, for which the algorithm was able to tell apart chirplets from CCSN GWs. On the opposite extreme were magneto-rotational signals, with half of them misclassified (mostly mixed into the chirplet category). This could potentially be explained by the perhaps too simplistic criteria used to discard signals as chirplets; however, it

must be stressed that few original test signals were available, with the total number of injections achieved by copying them at several SNR values and GPS times (equivalent to noise realizations). The denoising dictionary proved to be effective at recovering signals in previous sections, which in this case reduces the effectiveness of this kind of *data augmentation* by providing the LRSDL dictionary with almost exact copies. Hence, if by chance one of the original signals were confused by the classification algorithm, most of its injected copies would be expected to meet the same outcome. Neutrino and nonexploding signals were reasonably well differentiated, with the misclassifications coinciding with injections at the lowest SNR where it is more challenging to detect the SASI components.

In the case of the Einstein Telescope almost all nonexploding waveforms were correctly classified, which is consistent with the increased sensitivity of the detector especially at low frequencies. Results degraded in the case of neutrino-driven and magneto-rotational signals, however. This was partially due to the optimization of hyper-parameters performed with only LIGO detector's noise (for computational reasons), outweighing the increased sensitivity of the Einstein Telescope.

Finally, results for the NEMO detector were overall similar to the ET detector. The accuracy for nonexploding sources decreased in all likelihood due to the reduced sensitivity of NEMO at low frequencies. Magneto-rotational signals were again largely misclassified, although this time because of the confidence threshold marking almost half of them as chirplets. This was compensated by a slight increase in the accuracy of neutrino-driven and chirplets, the latter ones better discriminated because of their higher frequency.

11.4 Conclusions

In this chapter, we have presented the recent application of Sparse Dictionary Learning (SDL) techniques to gravitational wave (GW) signal processing, focused on denoising, glitch removal, and signal classification.

Our work shows how SDL is effective in both denoising astrophysical signals and glitch removal. Tailored dictionaries were able to reduce noise while preserving the integrity of GW signals, particularly in core-collapse supernovae (CCSN) and binary black hole (BBH) cases. Additionally, the removal of blip glitches in Advanced LIGO data highlighted SDL's potential to cleanly eliminate transients without affecting the underlying signals. Moreover, the application of the state-of-the-art LRSDL algorithm to signal classification showed that learned dictionaries could distinguish between morphologically similar signals, such as those from neutrino-driven and magneto-rotational supernovae, even in noisy environments.

Overall, we believe that SDL shows promise as a tool in GW data analysis. The ability to construct and optimize dictionaries for specific tasks, whether it be denoising, deglitching, or classification, opens up new possibilities for improving the accuracy and reliability of GW detections. We hope that the continued advancement

of SDL techniques will contribute to the future of gravitational wave astronomy, enhancing our ability to *hear* our universe.

Acknowledgements Work supported by the Spanish Agencia Estatal de Investigación (grant PID2021-125485NB-C21) funded by MCIN/AEI/10.13039/501100011033 and ERDF A way of making Europe, by the Generalitat Valenciana (grant CIPROM/2022/49), and by the European Horizon Europe staff exchange (SE) programme HORIZON-MSCA-2021-SE-01 (NewFunFiCo-101086251).

References

1. Aasi, J., et al.: Advanced LIGO. Class. Quant. Grav. **32**, 074001 (2015)
2. LIGO Scientific Collaboration, Virgo Collaboration, et al.: Effects of data quality vetoes on a search for compact binary coalescences in Advanced LIGO's first observing run. Class. Quant. Grav. **35**, 065010 (2018)
3. LIGO Scientific Collaboration, Virgo Collaboration, et al.: A guide to LIGO–Virgo detector noise and extraction of transient gravitational-wave signals. Class. Quant. Grav. **37**, 055002 (2020)
4. Acernese, F., et al.: Advanced Virgo: a second-generation interferometric gravitational wave detector. Class. Quant. Grav. **32**, 024001 (2015)
5. Ackley K, et al. Neutron Star Extreme Matter Observatory: A kilohertz-band gravitational-wave detector in the global network. Publications of the Astronomical Society of Australia (2020)
6. Aguilera-Dena, D., Langer, N., Moriya, T., Schootemeijer, A.: Related progenitor models for long-duration Gamma-ray bursts and type Ic superluminous supernovae. The Astrophys. J. **858**, 115 (2018)
7. Badger, C., Martinovic, K., Torres-Forné, A., Sakellariadou, M., Font, J.: Dictionary learning: a novel approach to detecting binary black holes in the presence of galactic noise with LISA. Phys. Rev. Lett. **130**, 091401 (2023)
8. Bisnovatyi-Kogan, G., Moiseenko, S., Ardelyan, N.: Magnetorotational mechanism of the explosion of core-collapse supernovae. Phys. Atomic Nuclei. **81**, 266–278 (2018)
9. Blondin, John M., Gipson, Emily, Harris, Sawyer, Mezzacappa, Anthony: The standing accretion shock instability: enhanced growth in rotating progenitors. The Astrophys. J. **835**, 170 (2017)
10. Bugli, M., Guilet, J., Obergaulinger, M., Cerdá-Durán, P., Aloy, M.: The impact of non-dipolar magnetic fields in core-collapse supernovae. Month. Notices of the R. Astron. Soc. **492**, 58–71 (2019)
11. Bugli, M., Guilet, J., Obergaulinger, M., Cerdá-Durán, P., Aloy, M.: The impact of non-dipolar magnetic fields in core-collapse supernovae. Month. Notices of the R. Astron. Soc. **492**, 58–71 (2020)
12. Cabero, M., Lundgren, A., Nitz, A., Dent, T., Barker, D., Goetz, E., Kissel, J., Nuttall, L., Schale, P., Schofield, R., Davis, D.: Blip glitches in Advanced LIGO data. Class. Quant. Grav. **36**, 155010 (2019)
13. Chen, S., Donoho, D., Saunders, M.: Atomic decomposition by basis pursuit. SIAM Rev. **43**, 129–159 (2001)
14. Cuoco, E., Calamai, G., Fabbroni, L., Losurdo, G., Mazzoni, M., Stanga, R., Vetrano, F.: On-line power spectra identification and whitening for the noise in interferometric gravitational wave detectors. Class. Quant. Grav. **18**, 1727 (2001)
15. Cuoco, E., Losurdo, G., Calamai, G., Fabbroni, L., Mazzoni, M., Stanga, R., Guidi, G., Vetrano, F.: Noise parametric identification and whitening for LIGO 40-m interferometer data. Phys. Rev. D **64**, 122002 (2001)

16. Dimmelmeier, H., Ott, C., Marek, A., Janka, H.: Gravitational wave burst signal from core collapse of rotating stars. Phys. Rev. D **78**, 064056 (2008)
17. Elad, M., Aharon, M.: Image denoising via sparse and redundant representations over learned dictionaries. IEEE Trans. Image Process. **15**, 3736–3745 (2006)
18. Grosse, R.B., Raina, R., Kwong, H., Ng, A.Y.: Shift-invariance sparse coding for audio classification. CoRR (2012). abs/1206.5241
19. LIGO Scientific Collaboration, Virgo Collaboration, et al.: GW170817: observation of gravitational waves from a binary neutron star inspiral. Phys. Rev. Lett. **119**, 161101 (2017)
20. Macleod, D., Urban, A., Coughlin, S., Pitkin, M., Altin, P., Quintero, E., Leinweber, K.: GWpy: Python package for studying data from gravitational-wave detectors (2019). ascl:1912.016
21. Hild, S., et al.: Sensitivity studies for third-generation gravitational wave observatories. Class. Quant. Grav. **28**, 094013 (2011)
22. Huang, K., Aviyente, S.: Sparse representation for signal classification **19** (2006)
23. Janka, H.: Neutrino-driven explosions, p. 1095 (2017)
24. Li, S., Fu, Y.: Learning robust and discriminative subspace with low-rank constraints. IEEE Trans. Neural Netw. Learn. Syst. **27**, 2160–2173 (2016)
25. Llorens-Monteagudo, M., Torres-Forné, A., Font, J., Marquina, A.: Classification of gravitational-wave glitches via dictionary learning. Class. Quant. Grav. **36**, 075005 (2019)
26. LIGO Scientific Collaboration, Virgo Collaboration, et al.: Observation of gravitational waves from a binary black hole merger. Phys. Rev. Lett. **116**, 61102 (2016)
27. Mahdizadehaghdam, S., Panahi, A., Krim, H., Dai, L.: Deep dictionary learning: a PARametric NETwork approach. IEEE Trans. Image Process. **28**, 4790–4802 (2019)
28. Mairal, J., Bach, F., Ponce, J., Sapiro, G., Zisserman, A.: Discriminative learned dictionaries for local image analysis, pp. 1–8 (2008)
29. Mairal, J., Bach, F., Ponce, J., Sapiro, G.: Online dictionary learning for sparse coding. In: Proceedings of the 26th International Conference on Machine Learning, pp. 1–8 (2009)
30. Mairal, Julien, Bach, Francis R., Ponce, Jean: Task-driven dictionary learning. IEEE Trans. Pattern Anal. Mach. Intell. **34**, 791–804 (2012)
31. Mallat, S., Zhang, Z.: Matching pursuits with time-frequency dictionaries. IEEE Trans. Signal Process. **41**, 3397–3415 (1993)
32. Mezzacappa, A., Marronetti, P., Landfield, R., Lentz, E., Yakunin, K., Bruenn, S., Hix, W., Messer, O., Endeve, E., Blondin, J., Harris, J.: Gravitational-wave signal of a core-collapse supernova explosion of a $15M_{\odot}$ star. Phys. Rev. D **102**, 023027 (2020)
33. Mroué, A., Scheel, M., Szilágyi, B., Pfeiffer, H., Boyle, M., Hemberger, D., Kidder, L., Lovelace, G., Ossokine, S., Taylor, N., Zenginoğlu, A., Buchman, L., Chu, T., Foley, E., Giesler, M., Owen, R., Teukolsky, S.: Catalog of 174 binary black hole simulations for gravitational wave astronomy. Phys. Rev. Lett. **111**, 241104 (2013)
34. Obergaulinger, M., Aloy, M.: Magnetorotational core collapse of possible GRB progenitors—I. Explosion mechanisms. Month. Not. R. Astron. Soc. **492**, 4613–4634 (2020)
35. O'Connor, E., Couch, S.: Exploring fundamentally three-dimensional phenomena in high-fidelity simulations of core-collapse supernovae. The Astrophys. J. **865**, 81 (2018)
36. Pan, K., Liebendörfer, M., Couch, S., Thielemann, F.: Equation of state dependent dynamics and multi-messenger signals from stellar-mass black hole formation. The Astrophys. J. **857**, 13 (2018)
37. Peyré, G.: Sparse modeling of textures. J. Math. Imaging Vis. **34**, 17–31 (2009)
38. Powell, J., Trifirò, D., Cuoco, E., Heng, I., Cavaglià, M.: Classification methods for noise transients in advanced gravitational-wave detectors. Class. Quant. Grav **32**, 215012 (2015)
39. Powell, J., Müller, B.: Gravitational wave emission from 3D explosion models of core-collapse supernovae with low and normal explosion energies. Month. Not. R. Astron. Soc. **487**, 1178–1190 (2019)
40. Powell, J., Müller, B.: Three-dimensional core-collapse supernova simulations of massive and rotating progenitors. Month. Not. R. Astron. Soc. **494**, 4665–4675 (2020)
41. Powell, J., Müller, B., Heger, A.: The final core collapse of pulsational pair instability supernovae. Month. Not. R. Astron. Soc. **503**, 2108–2122 (2021)

42. Powell, J., Müller, B., Aguilera-Dena, D., Langer, N.: Three dimensional magnetorotational core-collapse supernova explosions of a 39 solar mass progenitor star. Month. Not. R. Astron. Soc. **522**, 6070–6086 (2023)
43. Powell, J., Iess, A., Llorens-Monteagudo, M., Obergaulinger, M., Müller, B., Torres-Forné, A., Cuoco, E., Font, J.: Determining the core-collapse supernova explosion mechanism with current and future gravitational-wave observatories. Phys. Rev. D **109**, 063019 (2024)
44. Radice, D., Morozova, V., Burrows, A., Vartanyan, D., Nagakura, H.: Characterizing the gravitational wave signal from core-collapse supernovae. The Astrophys. J. Lett. **876**, L9 (2019)
45. Ramirez, I., Sprechmann, P., Sapiro, G.: Classification and clustering via dictionary learning with structured incoherence and shared features. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 3501–3508 (2010)
46. Recht, B., Fazel, M., Parrilo, P.: Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. SIAM Rev. **52**, 471–501 (2010)
47. Rudin, L., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D: Nonlinear Phenomena **60**, 259–268 (1992)
48. Saiz-Pérez, A., Torres-Forné, A., Font, J.: Classification of core-collapse supernova explosions with learned dictionaries. Mon. Not. R. Astron. Soc. **512**, 3815–3827 (2022)
49. Somiya, K., Collaboration.: Detector configuration of KAGRA–the Japanese cryogenic gravitational-wave detector. Class. Quant. Grav. **29**, 124007 (2012)
50. Tang, H., Wei, H., Xiao, W., Wang, W., Xu, D., Yan, Y., Sebe, N.: Deep micro-dictionary learning and coding network (2018). arXiv e-prints. arXiv:1809.04185
51. Tang, H., Liu, H., Xiao, W., Sebe, N.: When dictionary learning meets deep learning: deep dictionary learning and coding network for image recognition with limited data. IEEE Trans. Neural Netw. Learn. Syst. **32**, 2129–2141 (2021)
52. Tariyal, S., Majumdar, A., Singh, R., Vatsa, M.: Deep dictionary learning. IEEE Access **4**, 10096–10109 (2016)
53. Tibshirani, R.: Regression shrinkage and selection via the Lasso. J. R. Stat. Soc. (Series B) **58**, 267–288 (1996)
54. Torres, A., Marquina, A., Font, J., Ibáñez, J.: Total-variation-based methods for gravitational wave denoising. Phys. Rev. D **90**, 084029 (2014)
55. Torres-Forné, A., Marquina, A., Font, J., Ibáñez, J.: Denoising of gravitational wave signals via dictionary learning algorithms. Phys. Rev. D. **94**, 124040 (2016)
56. Torres-Forné, A., Cuoco, E., Font, J., Marquina, A.: Application of dictionary learning to denoise LIGO's blip noise transients. Phys. Rev. D. **102**, 023011 (2020)
57. Vu, T., Monga, V.: Fast low-rank shared dictionary learning for image classification. IEEE Trans. Image Process. **26**, 5160–5175 (2017)
58. Walk, L., Foglizzo, T., Tamborra, I.: Standing accretion shock instability in the collapse of a rotating stellar core. Phys. Rev. D. **107**, 063014 (2023)
59. Wang, Z., Bovik, A., Sheikh, H., Simoncelli, E.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**, 600–612 (2004)
60. Wright, J., Yang, A., Ganesh, A., Sastry, S., Ma, Y.: Robust face recognition via sparse representation. IEEE Trans. Pattern Anal. Machine Intell. **31**, 210–227 (2009)
61. Xu, Y., Li, Z., Yang, J., Zhang, D.: A survey of dictionary learning algorithms for face recognition. IEEE Access **5**, 8502–8514 (2017)
62. Yang, M., Zhang, L., Feng, X., Zhang, D.: Fisher discrimination dictionary learning for sparse representation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 543–550 (2011)
63. Zevin, M., Coughlin, S., Bahaadini, S., Besler, E., Rohani, N., Allen, S., Cabero, M., Crowston, K., Katsaggelos, A., Larson, S., Lee, T., Lintott, C., Littenberg, T., Lundgren, A., Østerlund, C., Smith, J., Trouille, L., Kalogera, V.: Gravity spy: integrating advanced LIGO detector characterization, machine learning, and citizen science. Class. Quant. Grav. **34**, 064003 (2017)
64. Zhang, Z., Zhang, Y., Xu, M., Zhang, L., Yang, Y., Yan, S.: A survey on concept factorization: from shallow to deep representation learning. Inf. Process. Manage. **58**, 102534 (2021)

Chapter 12
Searching for Long-Duration Transient Gravitational Waves: Convolutional Neural Networks Applied to Glitching Pulsars

David Keitel, Luana M. Modafferi, and Rodrigo Tenorio

Abstract Besides compact binary mergers and other sources, long-duration quasi-monochromatic signals from spinning deformed neutron stars have long been one of the prime targets of ground-based gravitational-wave detectors. Glitching pulsars in particular can be a source of such signals that are not quite as persistent as those from more quiescent neutron stars, but could be detectable on timescales of hours to months. Within the framework of the g2net COST action, at the University of the Balearic Islands a project has been pursued to develop a hybrid search approach for such signals that combines intermediate products from a matched-filter search over a bank of frequency-evolution templates with neural networks trained to identify signals of varying start time, duration and amplitude evolution. Here we summarise the motivation for this project and the results originally presented in Modafferi, Keitel & Tenorio 2023, Physical Review D 108, 023005 [1].

Keywords Gravitational waves · Neutron stars · Pulsars · Pulsar glitches · Convolutional neural networks

12.1 Introduction

While the LIGO–Virgo–KAGRA network [2–4] has already detected a large variety of gravitational-wave signals from compact binary coalescences [5], the search for the first examples of many other types of astrophysical sources is still ongoing. One long-standing target are individual spinning deformed neutron stars, which can emit long-duration quasi-monochromatic signals in the sensitive band of current or future ground-based gravitational-wave detectors. The primary class of such signals are the so-called *continuous waves* (CWs) [6], where the emission is powered by the gentle

D. Keitel (✉) · L. M. Modafferi · R. Tenorio
Departament de Física, Universitat de les Illes Balears, IAC3–IEEC, Crta. Valldemossa km 7.5, 07122 Palma, Spain
e-mail: david.keitel@ligo.org

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_12

spin-down of the neutron star and will typically remain persistent for longer than an observing run. On the other hand, newborn neutron stars or those perturbed by a strong transient event could emit long-duration transient signals on timescales between those from binary mergers and CWs. One class of such signals has been labeled "transient continuous waves" (tCWs), defined as sharing the main characteristics of true persistent CWs: at any given time, the signals are quasi-monochromatic (limited to a narrow band in frequency), and their frequency and amplitude vary only slowly over time.

We summarise here the motivation and recent efforts to search for tCWs from one particular class of sources, namely glitching pulsars, over timescales of hours to months. We focus on how neural networks can help in their detection, summarising work that was done within the framework of the g2net COST action at the University of the Balearic Islands and originally presented in Modafferi, Keitel & Tenorio 2023, Physical Review D 108, 023005 [1].

12.2 Astrophysical Motivation

Pulsars are the most numerous observational manifestation of neutron stars detected so far, with over 3500 included in the ATNF catalogue[1] as of 2024. They are ideal targets for CW searches since the precise electromagnetic timing solutions allow for highly sensitive fully-coherent single-template [7] or narrow-band [8] searches at acceptable computational cost and low statistical trials factors. But as first suggested by Prix, Giampanis & Messenger in 2011 [9], the subset of pulsars that experience sudden spin-up events called glitches could also be tCW emitters. In a glitch, energy is not created out of nowhere but likely sourced either from quakes that release mechanical stresses in the neutron star's crust [10, 11] or, in the preferred family of models for most of the glitching population, from the superfluid interior of the neutron star [12, 13]. The superfluid is expected to lag the slow-down of the observable outer layers and can then transfer some of its excess rotational energy to those in a glitch event.

Such tCWs from glitching pulsars are not a guaranteed source, since the emission details, even up to orders of magnitude in amplitude, are model-dependent and have not been predicted in detail yet. However, an indirect upper limit was derived by [9] based on the maximum energy that can be liberated in a two-fluid interaction, and they argued that a similar limit with only a factor of two difference holds for the crustal quake scenario. Over an emission time τ, this energy budget translates to a dimensionless root-mean-square amplitude upper limit

$$\hat{h}_0 \leq \frac{1}{d}\sqrt{\frac{5G}{2c^3}\frac{\mathcal{I}}{\tau}\frac{|\Delta f_{\mathrm{gl}}|}{f_{\mathrm{rot}}}}. \tag{12.1}$$

[1] https://www.atnf.csiro.au/research/pulsar/psrcat/.

where d is the distance to the pulsar, G is Newton's constant, c is the speed of light, $\mathcal{I}$ is the moment of inertia of the neutron star, f_{rot} is its rotational frequency, and Δf_{gl} is the observed step change in frequency at the glitch. For the simple assumption of a "rectangular" tCW signal with constant amplitude over a duration τ, the upper limit can be directly expressed in terms of the nominal amplitude parameter h_0 as used in the CW literature [6]:

$$h_0 \leq \frac{1}{d}\sqrt{\frac{5G}{2c^3}\frac{\mathcal{I}}{\tau}\frac{|\Delta f_{\mathrm{gl}}|}{f_{\mathrm{rot}}}}. \tag{12.2}$$

One key observation about this result is that it scales with the same factor of $1/\sqrt{\tau}$ as the typical strain amplitude sensitivity of (t)CW searches. This means that at the same energy budget, tCWs will be approximately equally detectable independently of their actual length, as long as the search method can adapt efficiently to that length scale.

This upper limit from [9] was used by Moragues et al. [14] to study future detection prospects of tCWs from glitching pulsars, computing it for a set of 726 glitches from 217 pulsars observed up to that point. Those were compared with the sensitivity curves of current and future GW detectors, which correspond to their noise spectral densities scaled by a sensitivity depth factor for realistic narrow-band search setups. The main result is reproduced here in Fig. 12.1 and indicates that, if any glitching pulsars emit tCW-like radiation near the maximal allowed energy budget, there are nonvanishing detection chances for some nearby and strongly glitching pulsars, like Vela, already with the LIGO detectors from the current O4 observing run on. Next-generation detectors can provide an even deeper reach into the known population of glitching pulsars. A more detailed analysis of the differences for rectangular and exponentially-decaying tCWs (corresponding to the two vertical axes of the figure) can also be found in [14].

12.3 Searches so Far and Their Limitations

The first practical searches for tCWs after glitches from known pulsars were done using the *transient* $\mathcal{F}$-statistic*method* introduced by [9]. This is a matched-filter method derived from the classic $\mathcal{F}$-statistic for CWs [15]. It starts from a likelihood ratio between a signal hypothesis for a CW embedded in Gaussian noise compared with a pure Gaussian noise hypothesis, then maximises it over the four amplitude parameters of the signal model to obtain a detection statistic that "only" remains to be evaluated over a bank of possible frequency evolution parameters (the frequency at a reference time, one or more spindown parameters, and the pulsar's sky position). While the standard $\mathcal{F}$-statistic for CWs is usually evaluated over a fixed duration, e.g. a full observing run, the transient $\mathcal{F}$-statistic adds on top a brute-force evaluation of all possible start times and durations within the observation time.

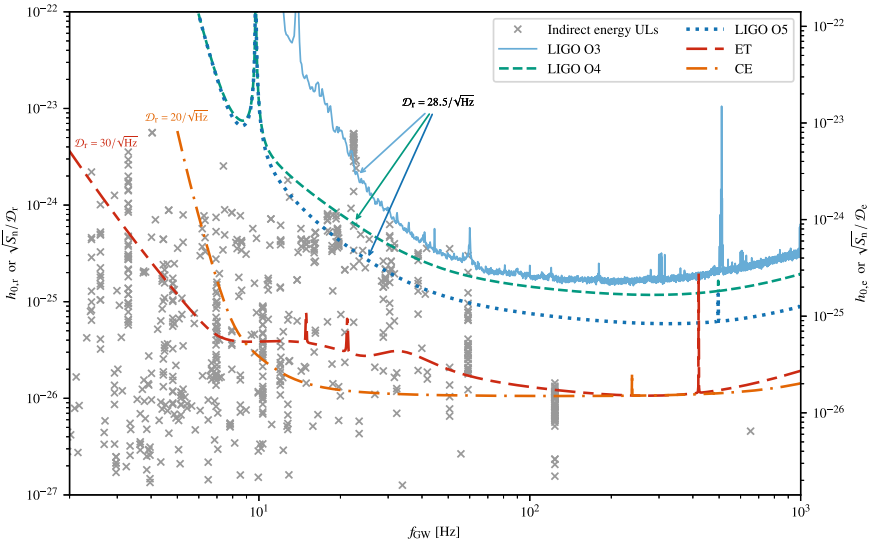


Fig. 12.1 Detection prospects for tCWs from glitching pulsars, based on the previously observed population, from [14]. The gravitational-wave frequency is assumed to be $f_{GW} = 2f_{rot}$. Crosses correspond to the upper limits from Eq. (12.2) and the sensitivity curves S_n for each detector are scaled by realistic sensitivity depth factors $\mathcal{D}$ for matched-filter searches. Reproduced from Fig. 1 of "Prospects for detecting transient quasi-monochromatic gravitational waves from glitching pulsars with current and future detectors", J. Moragues et al., MNRAS 519, 5161–5176 (2023), (c) 2022 the authors, published by Oxford University Press on behalf of Royal Astronomical Society. See that paper for additional details

The method was first applied in practice to open LIGO data from the O2 observing run [16] in [17], to search for tCWs of up to 4 months in length after one glitch each from the Crab and Vela pulsars. Searches for nine glitches from six pulsars during the O3 observing run, using LIGO and in some cases Virgo data, were performed as well [8, 18]. In no case was a promising detection candidate found, and the indirect energy upper limit has not yet been reached for any target.

Searches with the transient $\mathcal{F}$-statistic are mainly computationally limited, since the cost of evaluating all partial sums for different signal start times and durations exceeds the pure $\mathcal{F}$-statistic matched filter cost by orders of magnitude [9, 19]. Hence, these practical searches so far have been limited to narrow-band searches over less than a Hertz around the $f_{GW} = 2f_{rot}$ frequency of known glitching pulsars. All-sky all-frequency all-time searches for unknown glitching pulsars would require radically different approaches. In addition, the method is mostly only computationally feasible when limiting to the simple rectangular amplitude evolution, while an astrophysically more intuitive exponential signal decay makes the evaluation much more costly again [9, 19]. These two limitations have motivated the exploration of machine-learning approaches to tCW detection.

12.4 Convolutional Neural Network Setup

We now focus on the approach taken in Modafferi, Keitel & Tenorio 2023 [1]. This is a hybrid matched filter—machine learning setup, in which the aim was not to completely replace the $\mathcal{F}$-statistic based pipeline, but only to improve over its computationally limited part: the evaluation of partial sums over different possible start times and durations for possible tCW signals in a given data set. Keeping the initial matched-filter stage over a bank of frequency evolution templates makes it easier to stay close to the near-optimal sensitivity of the pure transient $\mathcal{F}$-statistic searches (for tCW signals exactly matching the assumed signal model), while for pure machine-learning approaches, such as those based on pattern recognition in spectrograms, it is often a much greater challenge to get into such sensitivity regimes far below the detector noise curves. On the other hand, a central goal was to keep flexibility in amplitude evolution while increasing computational efficiency.

The basic inputs to the neural networks are hence not the gravitational-wave strain series, nor Fourier transforms, nor spectrograms. Instead intermediate outputs of the $\mathcal{F}$-statistic matched filter code in LALSuite [20] are used, the so-called $\mathcal{F}$-statistic *atoms* as described in [9, 21]. These are evaluated for each Short Fourier Transform (SFT) into which the data is typically divided for (t)CW searches—1800 s each in this example. They are a set of seven quantities at each SFT timestamp: three that describe the sky-position dependent detector sensitivity in terms of antenna pattern matrix components and four that are the real and imaginary parts of two projections of the data. See appendix A of [1] as well as [9, 21] for details.

The network architecture chosen for this study is a convolutional neural network (CNN) including three one-dimensional convolutional layers with rectified linear unit (ReLU) activation functions, a flattening + dropout layer, as well as four fully connected layers and the final output layer, all also with ReLU activations. This is illustrated in Fig. 12.2. The output quantity of the network is trained to be a proxy for the signal-to-noise ratio (SNR) of tCW signals in the data.

A key part of the project was to develop a training strategy that is well suited to a realistic detection problem at hand and allowed for achieving a sensitivity close to the full transient $\mathcal{F}$-statistic method. Training is done with a mean-squared error

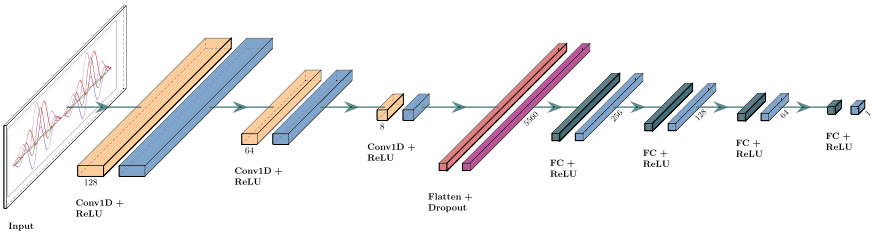


Fig. 12.2 CNN architecture used in [1]. Reprinted with permission from [1]. Copyright 2023 by the American Physical Society. With special thanks to Vincent Boudart

loss function and with a two-stage curriculum learning [22] strategy. The training set was constructed to emulate the O2 Vela post-glitch search from [17]. In the first stage, the signal part of the training set have large SNRs uniformly distributed from 6 to 40, while in the second stage a larger set of weaker signals with SNRs uniformly distributed between 4 and 10 are added. On the first set the Adam optimiser [23] is used, which is know for rapid convergence, and the standard stochastic gradient descent optimiser [24] for the second stage, which was found to produce less overfitting in the final network state after 200 + 1000 training epochs in the two stages. Training is done first on synthetic data ($\mathcal{F}$-statistic atoms directly simulated under a Gaussian noise assumption, without a detour via fully simulated time-series or SFT data) and then on real O2 data [16].

Additionally, a *two-stage filtering* method was developed, where a subset of candidates with high CNN ranking are re-ranked with the most sensitive $B_{\mathrm{tS/G}}$ statistic, an improved version of the transient $\mathcal{F}$-statistic that instead of maximising over start time and duration performs Bayesian marginalisation over these two parameters [9]. This is illustrated in a flowchart in Fig. 12.3.

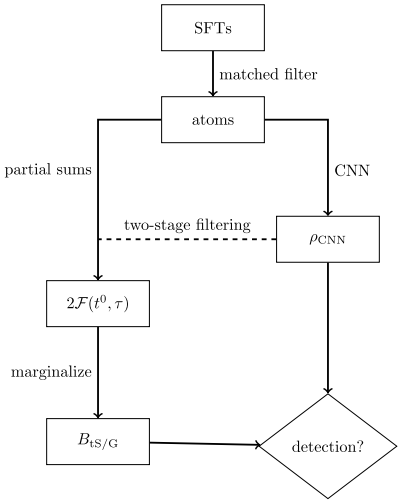


Fig. 12.3 Comparison, as a flowchart, of the pure transient $\mathcal{F}$-statistic method, the pure CNN detection statistic (based on $\mathcal{F}$-statistic atom inputs), and the two-stage filtering where a subset of candidates with high CNN ranking are re-ranked with the most sensitive $B_{\mathrm{tS/G}}$ statistic (an improved version of the transient $\mathcal{F}$-statistic using Bayesian marginalisation over start time and duration rather than maximisation). Reprinted with permission from [1]. Copyright 2023 by the American Physical Society

12.5 Results

The main attraction of the CNN approach is the higher computational efficiency. In [1] a training time of 1.8 h per data set on an Nvidia Tesla V100 was reported. For a full search setup with $\approx 1.15 \times 10^7$ templates (corresponding to the O2 Vela glitch search from [17]), the two-stage filtering pipeline took only 1.5 h total, compared to the over 4×10^4 h the CPU version of the traditional pipeline would have taken and 95 h for its GPU version [19]. Speedups were more modest for rectangular signals, but still more than a factor of 10 per template.

Detailed evaluations of the CNN performance in [1] were first presented in terms of SNR recovery as well as receiver-operator characteristic curves (ROCs), i.e. the detection probability as a function of false-alarm probability over a fixed test set. These tests were done on both synthetic and real data, with both rectangular and exponentially decaying signals. To meaningfully compare performance with a realistic narrow-band tCW search with hundreds of thousands to millions of frequency evolution templates, it is necessary to cover false-alarm probabilities down to 10^{-7} with a correspondingly large test set. The results are very promising in terms of reaching within 10% of the detection performance of the transient $\mathcal{F}$-statistic at such low false-alarm probabilities with the CNN-estimated SNR as the only detection statistic, and with the two-stage filtering approaching it to within a few percent (for rectangular signals) or even within measurement uncertainties (for exponential signals).

As the most realistic end-to-end test, the full search from [17] on O2 data for tCWs after the 2016 glitch of the Vela pulsar [25] was then reproduced with the CNN-based pipeline, and also extended for the first time to explicitly cover exponentially-decaying signals. The resulting upper limits based on the two-stage filtering method, a threshold corresponding to the highest $B_{\mathrm{tS/G}}$ value from the final candidate list, and signal injections into the real data, are reproduced here in Fig. 12.4. They are competitive with the much more costly full transient $\mathcal{F}$-statistic based pipeline within 6–7%.

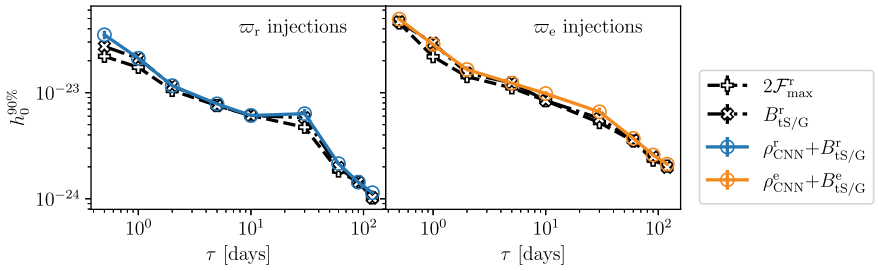


Fig. 12.4 Upper limits from [1] for tCWs after the Vela glitch in O2, comparing the two-stage filtering ($\rho_{\mathrm{CNN}} + B_{\mathrm{tS/G}}$) against the traditional ranking statistics. The left panel is for rectangular signal amplitude windows (ϖ_r) and the right panel for exponentially decaying ones (ϖ_e) Reprinted with permission from [1]. Copyright 2023 by the American Physical Society

It was also investigated how the networks can be extended to a two-parameter output, estimating the signal duration τ in addition to SNR. This way, beyond a pure detection pipeline, a first level of parameter estimation by CNNs is also included. This was only intended as a proof of principle, without full attempts to optimise the network architecture to this new goal. Results are however already promising. First, the SNR estimates do not suffer noticeably from the new feature. Second, after limiting to signals with sufficient SNR to be at least marginally detectable, only a small fraction of signals has severely misestimated durations (3% for rectangular signals and 7% for exponentially decaying signals) and the rest have a root-mean-square duration error of 27% for rectangular and 29% for exponential signals. These numbers correspond to a CNN trained and evaluated on synthetic data only. Testing the same network on real data makes recovery for rectangular signals only marginally worse while showing more impact on exponential signals. This parameter-estimation network was not retrained on real data, so such a step and/or further work on the network architecture could certainly improve this performance significantly.

12.6 Discussion and Future Outlook

The work in [1] was a first step towards production-level machine-learning based searches for long-duration quasi-monochromatic signals (tCWs) from glitching pulsars, demonstrating that a hybrid approach starting from intermediate matched-filter outputs and using a final follow-up stage with a traditional Bayes factor statistic on a small subset of the candidates that were top-ranked by the CNN can achieve sensitivities approaching that of the traditional pipeline at much lower computational cost and greater flexibility in signal amplitude evolution.

Similar ideas have been pursued for other long-duration transient signals, e.g. for rapid-spindown signals from neutron star remnants of binary mergers in [26]. In general, the field of long-duration gravitational-wave transients between the regimes of explosive transients and persistent CWs is less well-explored both observationally and in terms of algorithm development, and machine-learning approaches have a great opportunity to have observational impact in this type of searches for novel detections.

Acknowledgements We thank the ULìege group from the STAR Institute for hosting L.M.M. during a g2net (CA17137) short term scientific mission working on this project, and the members of g2net, the GRAVITY group at UIB, as well as the Continuous Waves working group for fruitful discussions. This work was supported by the Universitat de les Illes Balears (UIB); the Spanish Agencia Estatal de Investigación grants CNS2022-135440, PID2022-138626NB-I00, RED2022-134204-E, RED2022-134411-T, funded by MICIU/AEI/10.13039/501100011033, the European Union NextGenerationEU/PRTR, and the ERDF/EU; and the Comunitat Autònoma de les Illes Balears through the Direcció General de Recerca, Innovació I Transformació Digital with funds from the Tourist Stay Tax Law (PDR2020/11 - ITS2017-006) as well as through the Conselleria d'Economia, Hisenda i Innovació with grant numbers SINCO2022/6719 (European Union NextGenerationEU/PRTR-C17.I1) and SINCO2022/18146 (co-financed by the European Union and FEDER Operational Program 2021–2027 of the Balearic Islands); and EU COST Actions

CA18108 and CA17137. During the original work on the study published in [1], in addition DK was supported by the Spanish Ministerio de Ciencia, Innovación y Universidades (ref. BEAGAL 18/00148) and cofinanced by the Universitat de les Illes Balears, LMM was supported by the Universitat de les Illes Balears, and RT was supported by the Spanish Ministerio de Ciencia, Innovación y Universidades (ref. FPU 18/00694). This material is based upon work supported by NSF's LIGO Laboratory which is a major facility fully funded by the National Science Foundation. The authors gratefully acknowledge the computer resources at Artemisa, funded by the European Union ERDF and Comunitat Valenciana as well as the technical support provided by the Instituto de Física Corpuscular, IFIC (CSIC-UV), and computational resources provided by the LIGO Laboratory and supported by National Science Foundation Grants PHY-0757058 and PHY-0823459.

Competing Interests The authors have no conflicts of interest to declare that are relevant to the content of this chapter.

References

1. Modafferi, L.M., Tenorio, R., Keitel, D.: Phys. Rev. D **108**(2), 023005 (2023). https://doi.org/10.1103/PhysRevD.108.023005
2. Aasi, J., et al.: Class. Quant. Grav. **32**, 074001 (2015). https://doi.org/10.1088/0264-9381/32/7/074001
3. Acernese, F., et al.: Class. Quant. Grav. **32**(2), 024001 (2015). https://doi.org/10.1088/0264-9381/32/2/024001
4. Akutsu, T., et al.: Nature Astron. **3**(1), 35 (2019). https://doi.org/10.1038/s41550-018-0658-y
5. Abbott, R., et al.: Phys. Rev. X **13**(4), 041039 (2023). https://doi.org/10.1103/PhysRevX.13.041039
6. Riles, K.: Living Rev. Rel. **26**(1), 3 (2023). https://doi.org/10.1007/s41114-023-00044-3
7. Abbott, R., et al.: Astrophys. J. **935**(1), 1 (2022). https://doi.org/10.3847/1538-4357/ac6acf
8. Abbott, R., et al.: Astrophys. J. **932**(2), 133 (2022). https://doi.org/10.3847/1538-4357/ac6ad0
9. Prix, R., Giampanis, S., Messenger, C.: Phys. Rev. D **84**, 023007 (2011). https://doi.org/10.1103/PhysRevD.84.023007
10. Ruderman, M.: Astrophys. J. **382**, 587 (1991). https://doi.org/10.1086/170745
11. Middleditch, J., Marshall, F.E., Wang, Q.D., Gotthelf, E.V., Zhang, W.: Astrophys. J. **652**, 1531 (2006). https://doi.org/10.1086/508736
12. Andersson, N., Comer, G.L., Prix, R.: Phys. Rev. Lett. **90**, 091101 (2003). https://doi.org/10.1103/PhysRevLett.90.091101
13. Andersson, N., Comer, G.L., Prix, R.: Mon. Not. Roy. Astron. Soc. **354**(1), 101 (2004). https://doi.org/10.1111/j.1365-2966.2004.08166.x
14. Moragues, J., Modafferi, L.M., Tenorio, R., Keitel, D.: Mon. Not. Roy. Astron. Soc. **519**(4), 5161 (2023). https://doi.org/10.1093/mnras/stac3665
15. Jaranowski, P., Krolak, A., Schutz, B.F.: Phys. Rev. D **58**, 063001 (1998). https://doi.org/10.1103/PhysRevD.58.063001
16. Abbott, R., et al.: SoftwareX **13**, 100658 (2021). https://doi.org/10.1016/j.softx.2021.100658
17. Keitel, D., Woan, G., Pitkin, M., Schumacher, C., Pearlstone, B., Riles, K., Lyne, A.G., Palfreyman, J., Stappers, B., Weltevrede, P.: Phys. Rev. D **100**(6), 064058 (2019). https://doi.org/10.1103/PhysRevD.100.064058
18. Modafferi, L.M., Moragues, J., Keitel, D.: J. Phys. Conf. Ser. **2156**(1), 012079 (2021). https://doi.org/10.1088/1742-6596/2156/1/012079
19. Keitel, D., Ashton, G.: Class. Quant. Grav. **35**(20), 205003 (2018). https://doi.org/10.1088/1361-6382/aade34
20. LIGO Scientific Collaboration, Virgo Collaboration, KAGRA Collaboration.: LVK Algorithm Library - LALSuite. Free software (GPL) (2018). https://doi.org/10.7935/GT1W-FZ16

21. Prix, R.: The F-statistic and its implementation in ComputeFstatistic_v2 (2018). https://dcc.ligo.org/T0900149/public
22. Bengio, Y., Louradour, J., Collobert, R., Weston, J.: In: Proceedings of the 26th Annual International Conference on Machine Learning (Association for Computing Machinery, New York, NY, USA, 2009), ICML '09, pp. 41–48. https://doi.org/10.1145/1553374.1553380
23. Kingma, D.P., Ba, J.: In: 3rd International Conference on Learning Representations (ICLR 2015) (2014)
24. Ruder, S.: arXiv e-prints (2016). https://doi.org/10.48550/arXiv.1609.04747. arXiv:1609.04747
25. Palfreyman, J., Dickey, J.M., Hotan, A., Ellingsen, S., van Straten, W.: Nature **556**(7700), 219 (2018). https://doi.org/10.1038/s41586-018-0001-x
26. Miller, A.L., et al.: Phys. Rev. D **100**(6), 062005 (2019). https://doi.org/10.1103/PhysRevD.100.062005

Chapter 13
Core-Collapse Supernova Waveforms Classification

Alberto Iess, Elena Cuoco, Jade Powell, and Filip Morawski

Abstract Core-collapse supernova (CCSN) are one of the sources to emit GWs that are yet to be detected. Furthermore, they represent an interesting candidate for multi-messenger observation, due to their neutrino and electromagnetic emissions. In this chapter we describe and evaluate search and classification methods for gravitational waves from CCSN explosions based on convolutional and recurrent neural network architectures. The proposed approaches use whitened time-series and whitened time-frequency representations as inputs to the deep learning models. We validate the classification accuracy of the models on CCSN waveforms, derived from state-of-the-art hydrodynamical simulations, and instrumental glitches, using both simulated and real interferometer background noise from Virgo, LIGO and Einstein Telescope. We show the robustness of the described methods in distinguishing CCSN and noise transients, and derive accuracies in multi-class waveform model classification in both a single and multi-detector setup.

This chapter is based on `Iess, F., Cuoco, E., Morawski, F. and Powell, J., 2021, In: Machine Learning: Science and Technology 1 025014,` DOI 10.1088/2632-2153/ab7d31, https://iopscience.iop.org/article/10.1088/2632-2153/ab7d31 and `Iess, F., Cuoco, E., Morawski, F., Nicolaou, C. and Lahav, O. 2023 In: A & A 669, A42` DOI 10.1051/0004-6361/202142525, https://www.aanda.org/articles/aa/full_html/2023/01/aa42525-21/aa42525-21.html.

A. Iess (✉)
Scuola Normale Superiore, Piazza dei Cavalieri, 7, 56126 Pisa, Italy
e-mail: alberto.iess@sns.it

E. Cuoco
Department of Physics and Astronomy, University of Bologna and INFN Bologna, Viale Berti Pichat 6/2, 40127 Bologna, Italy
e-mail: elena.cuoco@unibo.it

J. Powell
OzGrav, Centre for Astrophysics and Supercomputing, Swinburne University of Technology, Hawthorn, Melbourne 3122, Australia

F. Morawski
Nicolaus Copernicus Astronomical Center, Polish Academy of Sciences, Bartycka 18, 00-716 Warsaw, Poland
e-mail: fmorawski@camk.edu.pl

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_13

13.1 Introduction

CCSN are catastrophic stellar explosions that occur at the end of the life cycle of massive stars, typically those exceeding $8M\odot$. These events are triggered when the star exhausts its nuclear fuel, causing the core to collapse under its own gravity and resulting in the formation of a neutron star or black hole. A CCSN is accompanied by the release of an enormous amount of energy in the form of multi-messenger emission of neutrinos, gravitational waves and electromagnetic radiation [1].

CCSN are crucial for understanding stellar evolution and the dynamics of massive stars. They play a significant role in heavy element nucleosynthesis, driving galactic chemical evolution, and influencing star formation. The study of GWs emitted by CCSN offers valuable insight into the core-collapse mechanism, providing details about the properties of the progenitor star, the dynamics of the collapse, and the behaviour of matter under extreme conditions.

Recent advancements in hydrodynamical simulations have improved our understanding of CCSN emission mechanisms and GW detectors like Advanced LIGO [2, 3], Virgo [4], and the planned Einstein Telescope [5] will enhance the possibility to detect and analyse these signals in the near future. To detect this type of GW signal, however, we cannot rely on the matched filter method typically used in the context of binary mergers. This arises from the fact that it is not possible to model the exact emitted waveform, which is characterised by a high degree of stochasticity due to the inherently chaotic processes involved in the collapse, such as turbulent convection and standing accretion shock instabilities (SASI). Pipelines based on wavelet the wavelet transform and excess power are used to detect unmodelled or poorly modelled signals. Examples of such pipelines are cWB [6] and WDF [7], which rely on the wavelet transform to capture the characteristics of a signal in the time-frequency domain. ML techniques, such as convolutional neural networks, are increasingly being proposed as part of CCSN detection and classification pipelines, due to their ability to find and model non-linear patterns in complex GW data from CCSN.

13.2 Deep Learning Algorithms for CCSN Classification

ML methods can bolster analysis of GWs from CCSN, specifically tackling detection, classification, reconstruction, and parameter estimation problems. We put our effort and focus on classification, assuming a power excess pipeline will find triggers for the data. While parameter estimation for CCSN signals is fairly new, some studies have been done with the aim of defining universal relations [8]. While inspired by recent advancements in the field of ML, specifically in computer vision and object identification, we show how two classes of algorithms, based on convolution and recurrent architectures, can be exploited for CCSN and transient classification.

13.2.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) [9] are deep learning models primarily used for image and spatial data processing. The central component of CNNs is the convolutional layer, which applies filters to input data to produce feature maps, capturing spatial relationships and detecting patterns like edges and textures. This process involves convolving filters across the data, preserving spatial hierarchies.

Pooling layers, such as max pooling, reduce the dimensionality of feature maps by down-sampling, making the network more manageable and less prone to overfitting. Fully connected layers at the end of the network aggregate features to make final predictions.

CNNs are well suited in tasks like image classification, object detection, and segmentation due to their ability to learn hierarchical feature representations from data with minimal pre-processing, making them a cornerstone in computer vision and deep learning.

13.2.2 Long Short-Term Memory Networks

Long short-term memory networks (LSTMs) are a type of recurrent neural network designed to handle long-term dependencies in sequential data [10]. LSTMs use a gating mechanism, comprising input, forget, and output gates, to control the flow of information and maintain long-term memory, effectively mitigating issues like vanishing and exploding gradients. LSTMs process data one step at a time, updating hidden and cell states dynamically based on new inputs and previous states. This allows them to capture complex temporal patterns and long-range dependencies. Due to their robust learning capabilities, LSTMs are widely used in applications such as time series prediction, natural language processing, and speech recognition, making them a crucial tool for handling sequential data in deep learning.

The use of LSTM architectures is suited to track correlations in complex data such as that output at gravitational wave interferometers. There are, however, a number of possible disadvantages of LSTMs compared to CNNs. LSTMs require a fine hyperparameter tuning and, in their original formulation, have issues tracking longer dependencies of the order of ~ 1000 time steps. This can be problematic when dealing with data sampled at various kHz. A number of modifications can be introduced to the original architecture to tackle this disadvantages, such as peephole or working memory connections [11], or the introduction of convolutional and pooling layers in a hybrid network design to reduce the dimensionality of the data fed to the LSTM layer.

13.3 Datasets

The datasets were generated by adding CCSN waveforms to interferometer noise. We used simulated gaussian noise in Ref. [12] and real noise in the follow-up study [13] to take into account the varying non-stationary power spectral density, which is affected by environmental conditions that produce glitches and disturbances. For the follow-up analysis on real data 44 segments from the O2 public science run [14] were chosen based on data quality flags. The segments cover the GPS range from $t_{GPS} = 1185669120$ to $t_{GPS} = 1186070528$. We leave the first 300 s without injections to allow PSD estimate with WDF and compute the whitening parameters [15]. The CCSN model waveforms used in the two studies (s11, s13, he3.5, s18p, s25, m39, y20, s18np) are described in [13] and references therein. A standard downsampling and low-pass filtering procedure is applied before injecting the CCSN into 4096 Hz gravitational wave detector data with the preferred sky distribution [16]:

$$h(t) = F_+(\alpha, \delta, \lambda, \beta, \chi, \eta)\, h_+(t) + F_\times(\alpha, \delta, \lambda, \beta, \chi, \eta)\, h_\times(t), \tag{13.1}$$

where α is the right ascension and δ the declination, λ, β the longitude and latitude of the interferometer, χ an angle that defines the orientation the detector and η the angle between its arms. In Ref. [13] we fixed the source distance to 1 kpc with a uniform sky distribution, while in Ref. [12] we allowed for variable distances. As mentioned, short duration noise transient, referred to as glitches, are also present in the datasets to test the ability of ML models to distinguish them from the astrophysical counterparts. The segments with injections are whitened with WDF and event trigger GPS times are produced based on a threshold chosen on the normalized wavelet coefficients squared coefficients sum. WDF and other power excess pipelines will not produce triggers for every signal present in the data, especially at low SNRs. This has to be taken into account in order to build a balanced dataset for training and testing. A window of the order of 2 s is chosen around the GPS times to produce whitened timeseries or whitened time-frequency images which are the final samples in training, validation and testing, which amount to 60, 10, and 30% of the full dataset. Associated SNRs are computed as described in [17]:

$$\left(\frac{S}{N}\right)^2 = 4 \int_0^{f_{max}} \frac{\tilde{s}(f) \left[\tilde{h}(f)^*\right]_{t_0=0}}{S_n(f)}\, e^{2\pi i f t_0}\, df, \tag{13.2}$$

where $S_n(f)$ is the one-sided noise PSD, f_{max} is the maximum cutoff frequency and $\tilde{s}(f)$ and $\tilde{h}(f)^*$ are the fourier and conjugate fourier transforms of the data and the CCSN waveform template. We show examples of real noise from O2 with a glitch and a CCSN injection in Fig. 13.1.

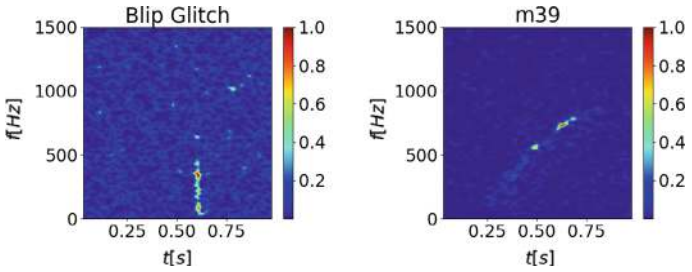


Fig. 13.1 Example of a blip glitch (*left*) and a CCSN m39 waveform (*right*) added to real interferometer noise from the O2. The spectrograms are rescaled to the [0, 1] range for a 2D CNN classifier. Image taken from [13]

13.4 CCSN Classification

The samples produced in time and time-frequency domain carry important information about the signal phase and amplitude evolution. The spectrogram representation is particularly useful to find patterns in the distributed power of a signal at different frequencies. The image samples are fed to 2D CNN architectures, while the whitened timeseries are used to train the 1D CNN and the LSTM network. Typical choices for the network hyperparameters, such as the learning rate, number of LSTMs and convolutional layers, dimensions of the filters, and batch sizes are chosen. We use the Adam optimizer with a learning rate $\alpha = 0.001$ and binary or categorical cross-entropy loss function if we want to distinguish between glitches and CCSN or between different CCSN models.

The architecture for the 1D CNN includes 4 convolutional layers with kernels of size 3. The number of convolutional filters are (120, 80, 80, 40). Max pooling (2, 2) and spatial dropout on 40% of nodes is applied after each convolutional layer. The output from the CNN architecture is fed into two Fully Connected (FC) layers of sizes 200 and 100 respectively. The activation function used in these layers is the rectified linear unit (ReLU). The final layer is FC, with the number of nodes given by the number of class labels and a softmax activation function to yield probabilities for each class. The 2D CNN takes as inputs spectrograms centered on the samples using multiple FFTs of 0.125 s with overlap and a kaiser window. The network is composed of 3 convolutional layers with kernels of sizes (4, 4), (3, 3), (2, 2) followed by 2 by 2 Max pooling. The output from the CNN architecture is fed into a FC layer and onto the output layer that yields classification probabilities. The recurrent network used in [13] is composed of 2 bidirectional LSTM layers with dimensions of 64 and 32 and spatial dropout on 10% of the output nodes from the first LSTM layer. The outputs are processed by 4 FC layers of decreasing size, with *tanh* activations. The output layer uses a softmax function to yield probabilities for each class. We present the classification results for simulated gaussian noise background and real O2 noise, separately.

13.4.1 Simulated Gaussian Background

To evaluate the CNN's ability to generalize to different CCSN models, the networks are trained on a custom set of signal models and tested on a fifth model not included in the training phase, along with scattered light and sine gaussian glitch models for binary classification. Single detector noise background data is simulated using Advanced Virgo and ET noise budget curves. The 2-D CNN exhibits the best classification sensitivity for both Virgo O3 and ET design noise background, as shown in Table 13.1.

Both 1-D and 2-D CNNs classify are able to correctly distinguish unseen waveforms from glitches, with slightly worse results for the only model which exhibits strong SASI at low frequencies, s25. ET has slightly worse results for this waveform, due to the fact that the high frequency configuration sensitivity was selected to produce the background noise. A full analysis which includes asymmetric neutrino and mass flows will strongly benefit from a the low frequency ET sensitivity curve configuration. The networks can also be efficiently trained to classify the signals into the exact CCSN model class. The results improve with increasing SNRs, as shown in Fig. 13.2.

Table 13.1 Binary classification on unknown CCSN and glitch models not included in training. For the 1-D and 2-D CNN implementations and both detector datasets, we show the sensitivity for the *Signal* class, reported as a percentage. We also report the *Total* accuracy, ratio of the number of correctly classified samples in one of the two classes and the total number of test samples. Table taken from [12]

VO3	1-D CNN accuracy		2-D CNN accuracy	
Test set	Signal	Total	Signal	Total
s11	93.9	93.7	98.0	94.3
he3.5	96.2	95.5	95.2	97.6
s18	97.5	96.7	98.4	97.9
s13	94.5	94.4	94.4	96.9
s25	95.1	95.1	92.2	95.9
ET	1-D CNN accuracy		2-D CNN accuracy	
Test set	Signal	Total	Signal	Total
s11	94.5	96.7	95.5	97.2
he3.5	98.0	97.8	98.5	97.6
s18	92.1	94.2	92.4	96.2
s13	95.9	96.6	84.5	94.1
s25	73.3	83.2	89.6	95.5

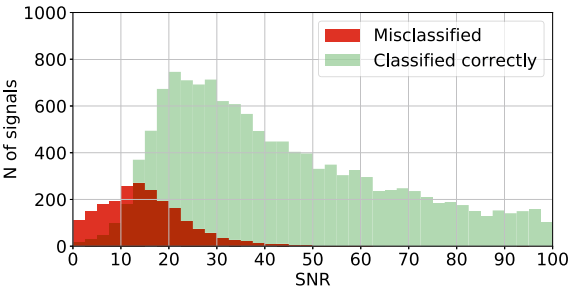


Fig. 13.2 The SNR distribution of the classified CCSN signals from the VO3 and ET test sets in the multi-class classification task in the 0 to 100 SNR range. Image taken from [12]

13.4.2 O2 Noise Background

While a dataset built on a simulated gaussian noise background is the first step to assess the efficiency of a GW data analysis pipeline, processing real detector data poses additional challenges, due to the non-stationarity of the detector PSD and the presence of non-linear noise transients. In [13] we used real O2 noise as a background for the injected CCSN signals. Different types of glitches naturally occur in this dataset, so we did not simulate additional noise transients. We used three-detector data from Advanced LIGO and Advanced Virgo to increase the robustness of our analysis. We trained and tested our 1-D, 2-D and LSTM models for multiclass classification of the CCSN waveforms. This time, however, we did not label the individual glitch types and used a single class for all noise transients. As in the case of simulated noise described in Sect. 13.4.1, we initially set ourselves the goal of classifying test samples into the individual CCSN model class using single interferometer data. Since we injected all waveforms at 1kpc for this study, the SNRs are then defined by: the CCSN waveform amplitudes at such distance, the antenna pattern, the variability of the noise background, and the segment whitening procedure. The last two points impact Virgo data specifically for the O2 science run, since the noise PSD changes significantly from the initial seconds used to estimate the whitening coefficients. The first important takeaway from this classification procedure lies in the comparison of the three chosen ML algorithms. As shown in Fig. 13.3, all three algorithms converge to low values of the cost function computed on the validation set, but the LSTM network is considerably slower to train. The 2-D CNN on the other hand converges after less than 10 training epochs.

Results for the three networks are comparable, with good sensitivities for the individual classes for the Advanded LIGO datasets. The Advanced Virgo dataset contains less CCSN signals due to the detector's lower sensitivity at 1kpc. For this reason the better classified waveforms are those that represent more energetic CCSN models, like y20 and m39. Apart from being the fastest model to train, the 2-D CNN is also the most accurate in correctly classifying the samples, as clearly shown by the individidual sensitivities reported in Table 13.2.

In a real case scenario there are stretches of time where multi-detector data is available. This can greatly enhance our ability to distinguish CCSN signals, since we can

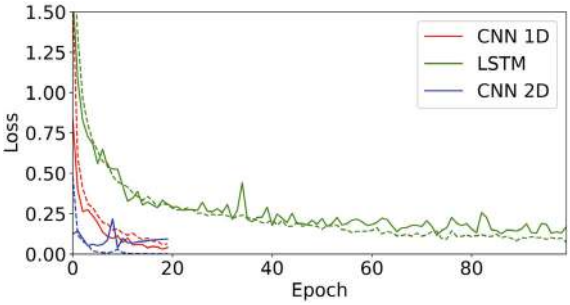


Fig. 13.3 Evolution of the categorical cross-entropy cost function for the 1-D, 2-D and LSTM models. The dashed lines are training losses, while the solid lines are validation losses. Image taken from [12]

Table 13.2 True positive rates for the noise and individual CCSN signal classes, along with the total sensitivity, for the Advanced Virgo (V1) and Advanced LIGO Livingston (L1), and Hanford (H1) on real data from the O2 science run. Due to its low energy the s11 model could not be detected by WDF in Advanced Virgo data

		Waveform									
ITF	Model	Noise	s11	s18p	s18np	He3.5	m39	y20	s13	s25	Total
V1	LSTM	49.2	*	3.6	58.4	0.0	89.5	69.9	0.0	89.8	73.7
	CNN 1D	44.6	*	8.4	10.9	0.0	84.3	73.1	0.0	87.4	68.3
	CNN 2D	48.6	*	9.6	39.4	3.6	92.3	72.5	0.0	94.6	75.2
L1	LSTM	90.1	0.0	98.2	92.8	85.4	98.7	96.0	87.1	94.8	93.6
	CNN 1D	99.4	0.0	89.5	95.3	82.2	99.2	98.2	75.5	98.8	95.9
	CNN 2D	99.8	0.0	99.1	99.3	97.4	100.0	99.7	91.6	99.8	99.3
H1	LSTM	96.2	0.0	95.5	96.8	89.1	99.7	95.9	75.1	97.6	95.4
	CNN 1D	99.0	0.0	90.1	99.3	91.6	98.4	100.0	80.6	97.4	96.5
	CNN 2D	99.7	0.0	99.6	99.8	96.8	99.7	99.8	96.8	99.2	99.1

leverage the full network SNR obtained by the individual SNRs in the three detectors. Contrarily, glitches are not expected to exhibit correlation in different detectors, since they are often due to local environmental noise sources. Figure 13.4 shows the confusion matrix for multi-class classification using multi-detector data. The ML model used merges the output probabilities obtained by the 1-D, 2-D and LSTM networks rather than the individual architectures.

The dataset used for training, validation and testing is built on the triggers that fall in a custom coincidence time window among the different detectors, distributed as follows: 675 noise, 1940 m39, 1557 s25, 1139 y20, 491 s18np, 329 s18p, 115 he3.5, 76 s13, 0 s11. Note that this dataset is small because we set the coincidence requirement. Despite the limited number of samples available, the true positive rates

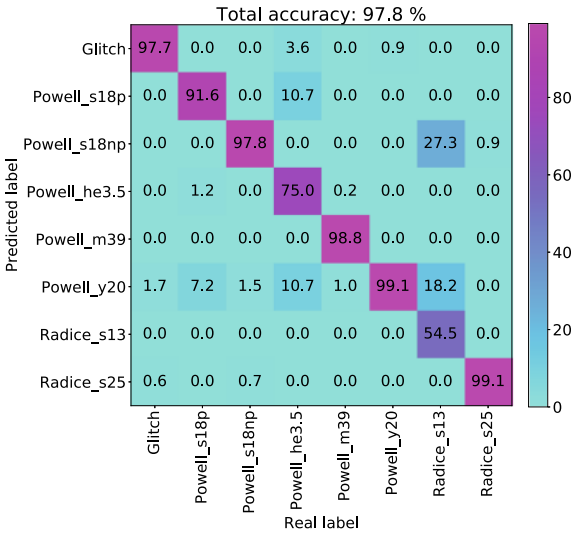


Fig. 13.4 Test set classification confusion matrix obtained by merging the outputs of LSTM, 1D CNN, and 2D CNN models trained on samples from a three interferometer network with LIGO (L1, H1) and Virgo (V1) data. Image taken from [13]

for the different CCSN model classes are still greater than 90% for waveforms with more than 300 samples. Only the poorly represented he3.5 and s13 fall below such threshold, while the s11 model is not present at all. The merged three detector model is also efficient in classifying glitches.

13.5 Discussion

We investigated the possibility of applying specialized NN architectures to correctly classify transient signals in GW data. While this is specifically intriguing when dealing with unmodeled or partially modeled waveforms, such as those produced by CCSN, the same approach can be used for other types of burst signals, such as those associated to different types of binary mergers (stellar BBH, IMBH, NS-BH, BNS). The speed and sensitivity of 2D-CNN models that leverage time-frequency signal morphology is unmatched, but 1-D CNN is also a fairly good choice, as shown in previous studies [18]. The main drawbacks for LSTMs are the long convergence time and strong dependency on the hyperparameter choice. It should be noted that at the time of writing many additional architectures, which present added layers of complexity and novel elements, have been used in computer vision and time-series classification studies. Convolutional layers, however, are still among the preferred choices in most image classification tasks. Another important result is that the ML algorithms are efficient and accurate also on real interferometer data, with only a slight loss of sensitivity in multi-class classification. In conclusion, recent studies show that the availability of large training sets will allow to effectively leverage ML

algorithms to reduce the noise background triggers due to glitches, while also providing a useful tool to classify individual CCSN signal models. Further effort should now be directed towards signal reconstruction and source parameter estimation.

References

1. Kuroda, T., et al.: Correlated signatures of gravitational-wave and neutrino emission in three-dimensional general-relativistic core-collapse supernova simulations. Astrophys. J. **851**, 1 (2017). https://doi.org/10.3847/1538-4357/aa988d
2. Harry, G.M.: LIGO scientific collaboration: advanced LIGO: the next generation of gravitational wave detectors. Class. Quant. Grav. **27**, 8 (2010). https://doi.org/10.1088/0264-9381/27/8/084006
3. Aasi, J., Abbott, B.P., Abbott, R., et al.: Advanced LIGO. Class. Quant. Grav. **32**, 7 (2015). https://doi.org/10.1088/0264-9381/32/7/074001
4. Acernese, F., Agathos, M., Agatsuma, K., et al.: Advanced Virgo: a second-generation interferometric gravitational wave detector. Class. Quant. Grav. **32**, 2 (2015). https://doi.org/10.1088/0264-9381/32/2/024001
5. Punturo, M., Abernathy, M., Acernese, F., et al.: The Einstein telescope: a third-generation gravitational wave observatory. Class. Quantum Grav. **27**, 19 (2010). https://doi.org/10.1088/0264-9381/27/19/194002
6. Klimenko, S., Vedovato, G., Drago, M., et al.: Method for detection and reconstruction of gravitational wave transients with networks of advanced detectors. Phys. Rev. D **93**, 4 (2016). https://doi.org/10.1103/PhysRevD.93.042004
7. Cuoco, E., Razzano, M., Utina, A.: Wavelet-based classification of gravitational wave transients with advanced LIGO and Virgo using XGBoost. In: 26th European Signal Processing Conference (EUSIPCO), pp. 2648–2652 (2018). https://doi.org/10.23919/EUSIPCO.2018.8553393
8. Torres-Forné, A., Cerdá-Durán, P., Obergaulinger, M., Müller, B., Font, J.A.: Universal relations for gravitational-wave asteroseismology of protoneutron stars. Phys. Rev. Lett. **123**, 051102 (2019). https://doi.org/10.1103/PhysRevLett.123.051102; Erratum: Phys. Rev. Lett. **127**, 239901 (2021) https://doi.org/10.1103/PhysRevLett.127.239901
9. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**, 11 (1998). https://doi.org/10.1109/5.726791
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**, 8 (1997). https://doi.org/10.1162/neco.1997.9.8.1735
11. Landi, F., Baraldi, L., Cornia, M., Cucchiara, R.: Working memory connections for LSTM. Neural Netw. **144**, 334–341 (2021). https://doi.org/10.1016/j.neunet.2021.08.030
12. Iess, A., et al.: Core-Collapse supernova gravitational-wave search and deep learning classification. Mach. Learn.: Sci. Technol. **1**, 025014 (2020). https://doi.org/10.1088/2632-2153/ab7d31
13. Iess, A., et al.: LSTM and CNN application for core-collapse supernova search in gravitational wave real data. A&A **669**, A42 (2023). https://doi.org/10.1051/0004-6361/202142525
14. Abbott, R., Abbott, T.D., Abraham, S., et al.: Open data from the first and second observing runs of Advanced LIGO and Advanced Virgo. SoftwareX **13**, 100658 (2021). https://doi.org/10.1016/j.softx.2021.100658
15. Cuoco, E., Calamai, G., Fabbroni, L., Losurdo, G., Mazzoni, M., Stanga, R., Vetrano, F.: Online power spectra identification and whitening for the noise in interferometric gravitational wave detectors. Class. Quantum Grav. **18**, 9 (2001). https://doi.org/10.1088/0264-9381/18/9/309
16. Schutz, B.F.: Networks of gravitational wave detectors and three figures of merit. Class. Quantum Grav. **28**, 12 (2011). https://doi.org/10.1088/0264-9381/28/12/125023

17. Allen, B., Anderson, W.G., Brady, P.R., Brown, D.A., Creighton, J.D.E.: FINDCHIRP: An algorithm for detection of gravitational waves from inspiraling compact binaries. Phys. Rev. D **85**, 12 (2012). https://doi.org/10.1103/PhysRevD.85.122006
18. Chan, M.L., Heng, I.S., Messenger, C.: Detection and classification of supernova gravitational wave signals: a deep learning approach. Phys. Rev. D **102**, 4 (2020). https://doi.org/10.1103/PhysRevD.102.043022

Chapter 14
Detecting Gravitational Waves as Anomalies with Convolutional Autoencoders

Filip Morawski, Michał Bejger, Elena Cuoco, and Luigia Petre

Abstract In this chapter, we summarize a generic approach to the analysis of gravitational wave (GW) data: we consider the GW to be the anomaly in the signal received by the detector and focus on detecting such anomalies. This approach works because detectable gravitational waves are rare (at the moment), transient signals, distinct from the slow-varying, non-stationary noise typically present in the detectors. The anomalies under investigation arise mainly from gravitational waves produced by the merger of binary black hole systems. However, our analysis extends beyond GW signals to encompass glitches identified within the real LIGO/Virgo dataset, accessible through the Gravitational Waves Open Science Center. Our anomaly detection process is based on deep learning techniques, specifically convolutional autoencoders trained on both simulated and real detector data. We demonstrate the efficacy of our method by reconstructing injected GW signals and explore how the detection of anomalies is influenced by the strength of the gravitational wave, quantified via the matched filter Signal-to-Noise Ratio (SNR). Furthermore, we apply our methodology to localize anomalies within the temporal domain of the time-series data that models the gravitational wave. The validity of our approach is confirmed by applying

This chapter is based on Morawski, F., Bejger, M., Cuoco, E. and Petre, L., 2021, In: Machine Learning: Science and Technology. 2, 4, 29 p., 045014, DOI 10.1088/2632-2153/abf3d0, https://iopscience.iop.org/article/10.1088/2632-2153/abf3d0.

F. Morawski · M. Bejger
Nicolaus Copernicus Astronomical Center, Polish Academy of Sciences,
Bartycka 18, 00-716 Warsaw, Poland
e-mail: bejger@fe.infn.it

M. Bejger
INFN Sezione di Ferrara, Via Saragat 1, 44122 Ferrara, Italy

E. Cuoco
Department of Physics and Astronomy, University of Bologna and INFN Bologna, Viale Berti Pichat 6/2, 40127 Bologna, Italy
e-mail: elena.cuoco@unibo.it

L. Petre (✉)
Faculty of Science and Technology, Åbo Akademi University, 20500 Turku, Finland
e-mail: luigia.petre@abo.fi

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_14

it to real-world data containing verified gravitational wave detections: our method is able to generalize and identify GW events not included in the training dataset.

14.1 Introduction

The first gravitational wave (GW) detection on September 14, 2015 [1] marked a pivotal moment in astrophysics. Since then, the collaborative efforts of the LIGO and Virgo teams, leveraging the Advanced LIGO [2] and Virgo [3] interferometers, have yielded 50 GW candidate signals by the end of the O3 observing run in 2020 [4–7]. These detections have provided crucial data for verifying theoretical models of gravitational wave sources, including binary systems of black holes (BH) and neutron stars (NS), while also probing the fundamental nature of gravity [8].

With further sensitivity enhancements in the interferometers, it is expected that a significantly larger volume of space will be explored, increasing the frequency of BH and NS merger detections [9]. This growing dataset offers unprecedented opportunities to study the nature of these compact objects and space-time itself.

GW data analysis is inherently challenging due to its noise-dominated nature, where astrophysical signals are often buried within various types of detector noise [8, 9]. Traditional approaches like matched filtering, employed in pipelines such as PyCBC [12] and GstLAL [13], rely on known GW templates, but are computationally intensive and sensitive to glitches—instrumental artifacts that can mimic or obscure GW signals. Alternative methods, for instance unmodeled GW burst searches like the coherent Wave Burst (cWB) pipeline [14, 15], offer flexibility in detecting GWs with unknown or partially modeled waveforms, though they remain susceptible to short-duration glitches.

Deep learning (DL) [16] has emerged as a powerful tool in GW analysis, particularly for its ability to learn from large datasets efficiently, without explicit programming [17]. In this work, we propose a model-independent DL-based method to detect anomalies-defined as extra-ordinary, rare transient features distinct from normal detector noise-in GW data. These anomalies may represent either GWs or instrumental glitches, and our method is tested using simulated and real BBH signal injections.

DL techniques are well-suited for anomaly detection due to their ability to model non-linearities in the data. The method's effectiveness is validated using confirmed GW detections, demonstrating its potential as an efficient Event-Trigger-Generator (ETG), especially when implemented on GPUs for real-time data processing. This work contributes to the growing field of DL applications in GW astrophysics, where rapid developments are enhancing the field's analytical capabilities [18, 19].

The outline of this work is as follows: Sect. 14.2 introduces the underlying deep learning components, Sect. 14.3 describes the data generation procedures, Sect. 14.4 presents our results, and Sect. 14.5 offers concluding remarks.

14.2 Deep Learning Algorithms

In this section, we overview how a combination of two deep learning techniques—convolutional neural networks (CNNs) and autoencoders (AEs)—can help us differentiate between gravitational wave (GW) signals and noise. We first shortly describe these methods and then present their application to the problem at hand.

14.2.1 CNN and AE

A *convolutional neural network* (CNN) [16] is a deep, feed-forward artificial neural network designed for processing structured data arrays, such as images. Unlike standard neural networks that use matrix multiplication, CNNs utilize a convolution operator as their core feature. This operator processes the input array (e.g., of size $m \times n$) by sliding a smaller kernel (or filter) of dimension $p \times p$, where typically $p < min(m, n)$. The convolution operation inner-multiplies overlapping $p \times p$ regions of the input with the kernel, replacing each region with the result of the inner-multiplication (a number), thereby reducing dimensionality. A simplified illustration of this convolution process is shown in Fig. 14.1.

Multiple layers are typically stacked sequentially in CNNs, each employing a varying number and type of kernels. These kernels are selected to enable the network to learn distinct features at each layer (e.g., edges, corners). Convolution layers are interspersed with other specialized layers, such as pooling layers, to progressively reduce the dimensionality of the data. This process continues until the final activation function maps the last layer to a vector, where each element represents a class probability for classifying the input. The class with the highest probability is considered the predicted class.

The sequential convolution operations, combined with the overlapping of input array elements and the sliding kernels, mimic the hierarchical structure of the human visual cortex, which processes visual information in stages, identifying increasingly

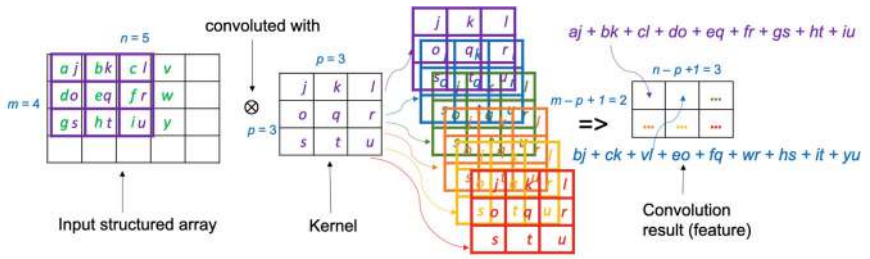


Fig. 14.1 The 2-dimensional 4×5 array on the left is seen as 6 overlapping 3×3 arrays; each of these smaller arrays is inner-multiplied with the 3×3 kernel, resulting in the 2×3 array on the right

complex features. CNNs excel at recognizing patterns in structured data, such as images, and have thus become the state-of-the-art approach in image classification and computer vision.

In our approach, we leverage the benefits of CNNs by embedding them within an *autoencoder* (AE) architecture [32]. An AE [16] is a specialized deep neural network that progressively encodes and compresses the input data before reconstructing an output based solely on the most compressed representation, known as the hidden layer, latent representation, or bottleneck. The core assumption of an AE is that the input data contains an underlying structure, such as correlations among input features, which the AE learns and exploits by enforcing the passage of information through the latent layer. When input features are independent, compression and reconstruction become significantly more challenging.

An optimal AE achieves a balance between sensitivity to the inputs—enabling accurate reconstruction—and insensitivity to noise or redundancies, preventing overfitting. This ensures that the latent representation retains only the essential variations in the data necessary for reconstruction. In our work, we employ an undercomplete AE, where the latent space dimension is smaller than the input dimension. This design helps to avoid overfitting by ensuring that the model cannot trivially replicate the input at the output.

The simple autoencoder (AE) depicted in Fig. 14.2, consisting of a single hidden layer, functions by encoding the input $y \in \mathbb{R}^d$ into a latent representation $h \in \mathbb{R}^p$, where $p < d$. Assuming the encoder activation function is $f : \mathbb{R}^d \to \mathbb{R}^p$, the encoding can be expressed as $h = f(Wy + b)$, where f can be a sigmoid function, `ReLU`, or something else, W is a weight matrix, and b represents the bias term, both initialized randomly and updated during training.

The decoder activation function, denoted by $g : \mathbb{R}^p \to \mathbb{R}^d$, maps the latent variable h back to the reconstructed output $\tilde{y} = g(\tilde{W}h + \tilde{b})$. In this case, the activation function g, weight matrix $\tilde{W}$, and bias term $\tilde{b}$ are not necessarily related to their encoding counterparts, f, W, and b. When only `ReLU` is used as the activation function and a single hidden layer is present, the AE behaves as a linear autoencoder. However, with additional hidden layers or non-linear activations, the AE becomes non-linear, which enhances its ability to capture more abstract features. Hence, the encoder and decoder generally constitute fully functional neural networks, not merely individual activation functions.

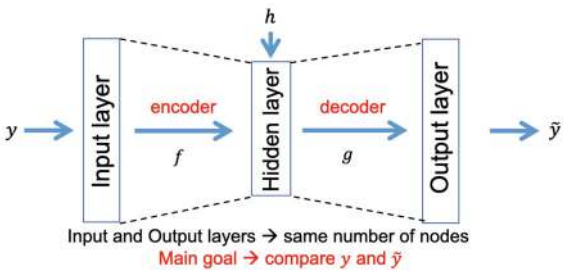


Fig. 14.2 An AE with one hidden layer

The training of an autoencoder (AE) is based on minimizing the reconstruction loss, commonly measured using the mean squared error (MSE) function [16]:

$$\mathcal{L}(y, \tilde{y}) = \|y - \tilde{y}\|^2 = \left\| y - g(\tilde{W} f(Wy + b) + \tilde{b}) \right\|^2 . \tag{14.1}$$

During training, the network parameters W, b, $\tilde{W}$, and $\tilde{b}$ are iteratively updated until the loss $\mathcal{L}(y, \tilde{y})$ reaches a sufficiently low value and further training no longer reduces it, indicating convergence. Various algorithms can be used to update these parameters, and in this work, we employ the ADAM optimizer (adaptive moment estimation) [33], which adjusts the learning rate dynamically during training. ADAM has demonstrated superior performance for large datasets, high-dimensional parameter spaces, and non-stationary input data.

The hyperparameters of the AE include the learning rate (which controls the size of each parameter update), batch size (number of training examples used in each update step), the optimization method (or optimizer), the architecture (number and size of layers), and the choice of loss function. These hyperparameters critically influence the performance of the AE.

Since the autoencoder (AE) reduces dimensionality during encoding, it is frequently used for feature extraction. However, compared to traditional dimensionality reduction techniques such as Principal Component Analysis (PCA), the AE offers a more powerful generalization, as it can capture non-linear relationships in the input data. While PCA seeks to identify a lower-dimensional hyperplane that approximates the original data, the AE can learn non-linear manifolds of minimal dimensionality, as illustrated in Fig. 14.3. Essentially, the AE models a vector field that maps the input data onto lower-dimensional manifolds that represent regions of high data density. When the learned manifold effectively characterizes the input data, the AE has successfully captured the structure of the data.

14.2.2 Applying the CNN AE

Although convolutional neural networks (CNNs) were originally designed for analyzing 2D data such as images [16], we apply them here in a simplified 1D form, as the signals we analyze are time series. The CNN-based autoencoder (AE) is designed to perform two tasks. For input instances containing only detector background noise (i.e., no anomalies), the AE is trained to reconstruct the noise as accurately as possible. However, for instances containing an anomalous signal—such as a gravitational wave (GW) or glitch in real data—the AE is trained to ignore the anomaly and reconstruct the data as though the signal were absent.

By comparing the input with the output reconstructed by the AE, the anomaly within the time-series data can be effectively isolated and studied further. The use of the AE is thus crucial, as it only reconstructs the patterns it was trained on, in this case

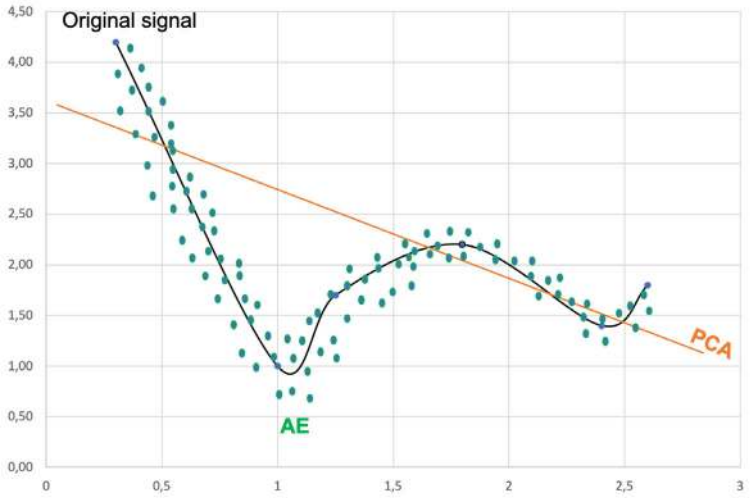


Fig. 14.3 Example of the manifold concept. Non linear versus linear dimension reduction

the detector noise, while excluding any other components from the reconstruction, such as GWs or glitches, which it treats as noise.

The loss value $\mathcal{L}(y, \tilde{y})$ is expected to vary depending on the presence of anomalies in the input data. For an anomalous input, $\mathcal{L}(y, \tilde{y})$ will be higher compared to an anomaly-free input, as the difference between y (containing noise or noise with anomaly) and $\tilde{y}$ (reconstructed noise) is greater. This difference correlates with the amplitude of the anomaly, leading to the expectation that the autoencoder (AE) trained on data with stronger anomalies will converge to a higher $\mathcal{L}(y, \tilde{y})$ during training.

The final architecture[1] employed in Sect. 14.4 was selected based on empirical tests. We evaluated architectures with 1 to 8 hidden layers, and the final configuration, depicted in Fig. 14.4, includes 3 hidden convolutional layers: encoding, latent representation, and decoding, with 256, 128, and 256 neurons, respectively. The kernel size for all layers was set to 3×1. All layers except the final output layer used `ReLU` as the activation function, while the final layer, responsible for reconstructing the input signal, used a sigmoid activation function. Additional hyperparameters included the ADAM optimizer [33] with a learning rate of 0.0005 and a batch size of 32.

The following section provides details on the datasets used for training, validation, and testing.

[1] The algorithm is implemented in `Python` [34] using the `Keras/TensorFlow` library [35, 36], with GPU support. Development was carried out on an NVidia Quadro P6000 (sponsored via the NVidia GPU seeding grant), and production runs were executed on the Prometheus cluster (Academic Computer Centre CYFRONET AGH), equipped with Tesla K40 GPU nodes, running CUDA 10.0 [37] and cuDNN 7.3.0 [38].

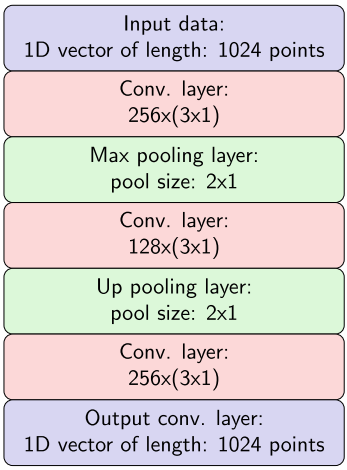


Fig. 14.4 Diagram shows the networks' layer structure and architecture. The size below convolutional layers correspond to the *Number of filters* times *Kernels size* of every layer. The second dimension of all layers is equal to unity since the initial input data were 1D time-series vectors

14.3 Training Data Sets and Data Flow

We generated two types of training datasets: a simplified dataset, consisting of simulated detector strain time series based on a colored normal noise distribution, designated as DataSet 1 (DS1), and a realistic dataset derived from actual data from the LIGO-Virgo O2 observing run [39], which is publicly accessible through the Gravitational Waves Open Science Center (GWOSC) [40], referred to as DataSet 2 (DS2). Both datasets adhere to the same general data processing flow as depicted in Fig. 14.5.

The whitening process referenced in the workflow diagram serves to eliminate contributions from stationary detector noise and adjusts the sensitivity across different frequencies [41]. This results in a uniform amplitude spectral density, making it easier to identify and compare gravitational wave (GW) signals embedded in the data. The whitening filter was independently recalculated for DataSet 1 (DS1) and DataSet 2 (DS2), as well as for each interferometer, to account for variations in sensitivity. The whitening procedure applied to both datasets was performed in the frequency domain using modules from the pyCBC Python library [42].

To simulate an astrophysical gravitational wave (GW) signal emitted by a binary black hole (BBH) system, we employed the IMRPhenomv4 waveform model [43], which captures the inspiral, merger, and final black hole (BH) ringdown phases. The component BH masses m_1 and m_2 were selected to match those estimated for the first detected event, GW150914 [1]. Specifically, the masses m_1 and m_2 were sampled based on the Initial Mass Function (IMF) within the ranges consistent with

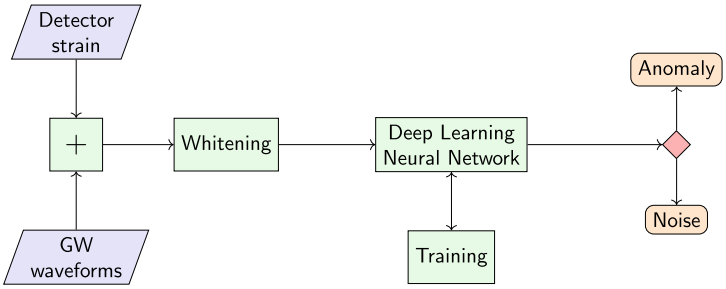


Fig. 14.5 Data flow of the project. The first step consists of data generation. GW waveforms were injected into the detector strain, either simulated or real (e.g. public data from the GWOSC platform). Since the raw time-series signal varied with frequency, the whitening procedure was applied to simplify data for further training. The training of the DL algorithms aimed to recover as many injected anomalies as possible, while limiting the false positive rate of the pipeline (noise samples incorrectly classified as anomaly)

the uncertainties in GW150914's mass estimation: m_1 in the range of $32.5-40.3$ $M_\odot$ and m_2 within $26.2-33.6$ $M_\odot$. The IMF power law index was chosen as $\alpha = -2.35$ [44].

Luminosity distances were sampled uniformly between 200 and 800 Mpc to span a realistic range of matched filter Signal-to-Noise Ratios (SNR), ranging from 4 to 40 across different interferometers, as depicted in the bottom-right plot of Fig. 14.6. The sky position of the source was optimized for each detector at a specific observation time. Examples of simulated GW signals are illustrated in the top-right plot of Fig. 14.6.

The DS1 dataset was constructed based on two assumed sensitivity curves for gravitational wave (GW) detectors. Each curve characterized the detector's sensitivity as a function of frequency, simulating realistic strain time series output. In our analysis, we employed the designed sensitivity for the advanced Virgo (aVirgo) detector from the O3 run (version without squeezing) [45, 46], as well as for the advanced LIGO (aLIGO) interferometers [47]. The term 'designed' refers to the sensitivity level these interferometers were expected to achieve following their planned upgrades.

To prepare the data, band-pass filtering was applied to remove noise components at high frequencies (above 1 kHz) and low frequencies (below 30 Hz, corresponding to seismic noise), as current interferometers lack sensitivity to detect GWs outside this range. The data was then resampled from 4096 to 1024 Hz. An example of the DS1 output time series is shown in the bottom-left plot of Fig. 14.6. Pre-generated GW signals were injected into the simulated strain and subsequently underwent whitening.

The DS2 (realistic) dataset was created using publicly available data from the LIGO-Virgo O2 observing run, accessed via the GWOSC platform [40]. The data was sourced from three interferometers: LIGO Hanford, LIGO Livingston, and Virgo,

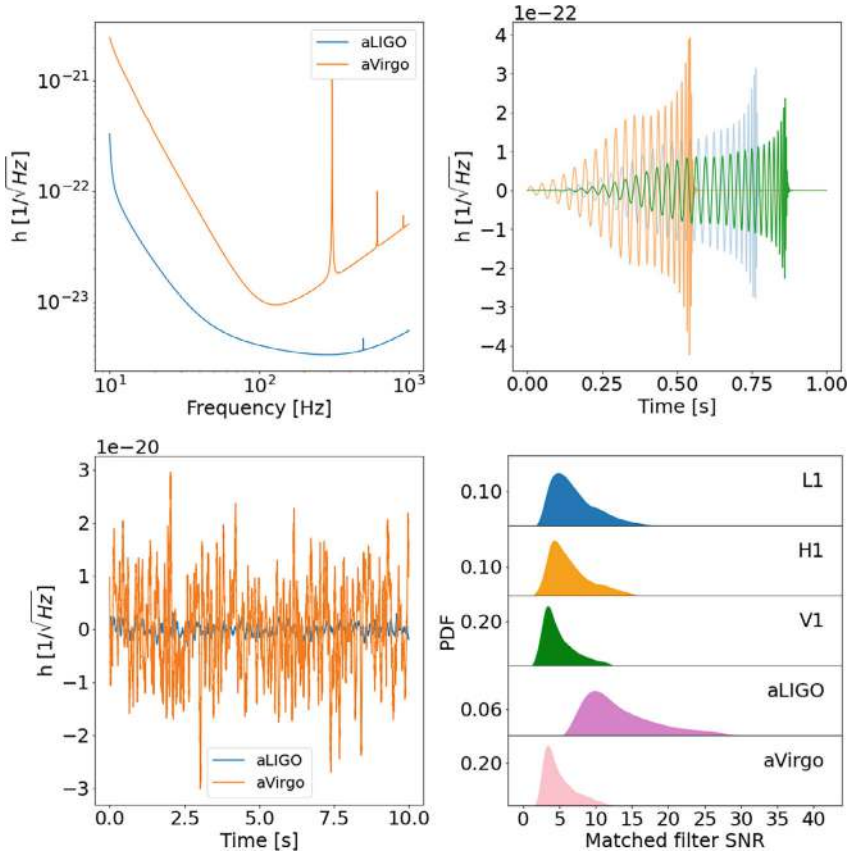


Fig. 14.6 *Top left*: Designed sensitivity of aVirgo and aLIGO interferometers. The detectors were expected to reach this level of sensitivity over broad range of frequencies after all planned upgrades. *Top right*: Examples of generated BBH GW waveforms injected into the strain data as anomalies. The GW were generated for the following parameters: blue signal—$m1 = 29\ M\odot, m2 = 24\ M\odot$, $distance = 440\ Mpc$; orange signal—$m1 = 33\ M\odot, m2 = 27\ M\odot, distance = 380\ Mpc$; green signal—$m1 = 28\ M\odot, m2 = 23\ M\odot, distance = 600\ Mpc$. *Bottom left*: Examples of the simulated data using the above sensitivity curves. *Bottom right*: Distributions of matched filter SNR of simulated GW injected into the real data for detectors: LIGO Hanford (blue), LIGO Livingston (orange) and Virgo (green) as well simulated data for: aLIGO (violet) and aVirgo (pink)

denoted as $H1$, $L1$, and $V1$, respectively. For each detector, we selected six hours of data to train the deep learning models. Specifically, we used data between GPS times 1187270656 and 1187295232 for $L1$, between 1174958080 and 1174982656 for $H1$, and between 1187672064 and 1187696640 for $V1$. The same gravitational wave signals used in the simulated dataset were injected into the real strain data.

This resulted in three distinct distributions of matched filter signal-to-noise ratio (SNR) for the same set of injections, as each detector exhibited different sensitivity levels. These distributions are illustrated in the bottom-right plot of Fig. 14.6. For

comparison, we also included SNR distributions from the simulated datasets. The real strain data, with injected anomalies, was subjected to whitening and resampled from 4096 to 1024 Hz.

We generated five datasets in total: two simulated datasets for aVirgo and aLIGO, and three based on real LIGO Livingston, LIGO Hanford, and Virgo O2 data. These datasets were segmented into one-second intervals and divided into training, validation, and testing subsets, with proportions of 65, 10, and 25%, respectively. An additional test set was created using one hour of data centered around the GPS times of confirmed gravitational wave (GW) detections. The data was whitened and resampled as described earlier. For this test set, we selected three binary black hole (BBH) detections from the O2 run—GW150914, GW170608, and GW170814—chosen for their high network signal-to-noise ratio (SNR) [48].

14.4 Results

The results are organized into three subsections. The first subsection presents the findings from anomaly searches conducted on the simulated dataset, which includes injected gravitational wave (GW) signals. The second subsection focuses on the results of anomaly searches in the real data from the Virgo and LIGO interferometers. The final subsection demonstrates the effectiveness of anomaly detection in real data containing confirmed GW detections.

14.4.1 Anomaly Searches on Simulated Data

The CNN-AE described in Sect. 14.2 was initially trained on the whitened, simulated data containing gravitational waves (GWs). Model convergence was observed after 100 epochs, with the mean squared error (MSE) loss function reaching approximately $6 \cdot 10^{-5}$ for the aVirgo data and 10^{-4} for the aLIGO data. Training was extended for an additional 100 epochs to monitor for overfitting; however, no overfitting occurred, and the MSE remained stable at the aforementioned values. The learning history of the autoencoder (AE), trained on both simulated datasets, showing the results for both the training and validation sets, is presented in the left plot of Fig. 14.7.

For aVirgo, the same set of GW signals covered a lower signal-to-noise ratio (SNR) range compared to aLIGO, due to aVirgo's reduced detector sensitivity (as shown in the top-left plot of Fig. 14.6). Consequently, this led to the model converging to a lower MSE for the aVirgo dataset, as the differences between the 'anomalous' input and the 'anomaly-free' reconstruction were smaller compared to the aLIGO data (see Sect. 14.2.2 for further details).

As detailed in Sect. 14.2.2, the properly trained CNN-AE successfully reconstructed the pure detector noise, irrespective of the presence of anomalies in the input data. By subtracting the input from the reconstructed output, the underlying

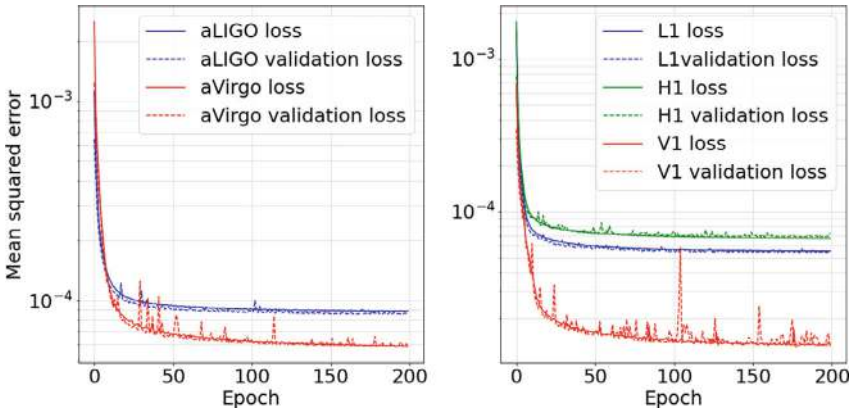


Fig. 14.7 The learning history of the AE trained on the whitened, simulated aVirgo and aLIGO datasets (*left plot*) as well as on the whitened, real data for $L1$, $H1$ and $V1$ datasets (*right plot*). The convergence was achieved after 100 epochs. We trained for 200 epochs, to investigate the onset of overfitting: it did not appear

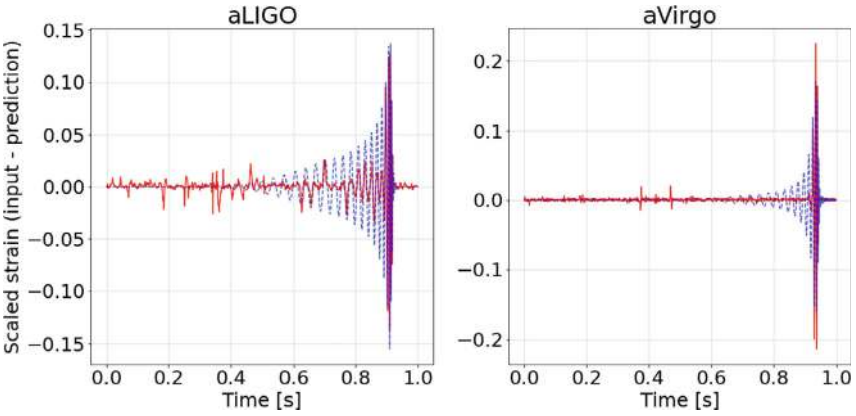


Fig. 14.8 The examples of signal reconstruction using the CNN-AE on aLIGO (*left plot*) and aVirgo data (*right plot*). The plots (red curves) were generated after subtracting the input signal from the AE predictions. For comparison, we added the expected signal using blue, dashed curve (difference between the input signal and the ground-truth output signal)

signal was effectively recovered. The differences between the initial input data and the autoencoder's output reconstruction were calculated. Examples of these results are illustrated in Fig. 14.8, where the dashed blue lines represent the expected signal, while the red lines correspond to the values obtained from the AE reconstruction.

For the aLIGO data, the gravitational waveforms were accurately reconstructed in most cases, with the anomalies distinctly differing from the surrounding noise. The recovered portion predominantly corresponded to the merger phase of the gravitational wave, with a smaller contribution from the inspiral phase. However, for the

aVirgo dataset, the reconstruction quality was notably poorer and often dominated by surrounding noise. In rare instances, such as the example shown in the right plot of Fig. 14.8, the merger phase was reconstructed. A detailed summary of the match between the injected and reconstructed waveforms, calculated using the $< x_1|x_2 >$ metric in the time domain, is provided in Appendix 14.6.1.

Given that the autoencoder (AE) successfully reconstructed the majority of anomalies within the data (particularly for the aLIGO dataset), we next sought to establish a metric for automatic anomaly detection. We selected the mean squared error (MSE) as this metric, calculated between the input data and the AE output, as described in Sect. 14.2.2. Figure 14.9 presents histograms of MSE values for two signal types within the dataset: noise and injected GW signals. As expected, MSE values for noise were significantly lower and approached zero, whereas the MSE values for GW signals were larger. A region of overlap between the histograms of both signal types (marked in burgundy in Fig. 14.9) was observed. Nevertheless, most of the instances containing injected GW signals in the aLIGO dataset, and nearly half in the aVirgo dataset, had MSE values greater than those for noise. Furthermore, we added a detection threshold to these histograms, indicating the number of injected GW signals that were correctly identified as anomalies (hatched area in Fig. 14.9).

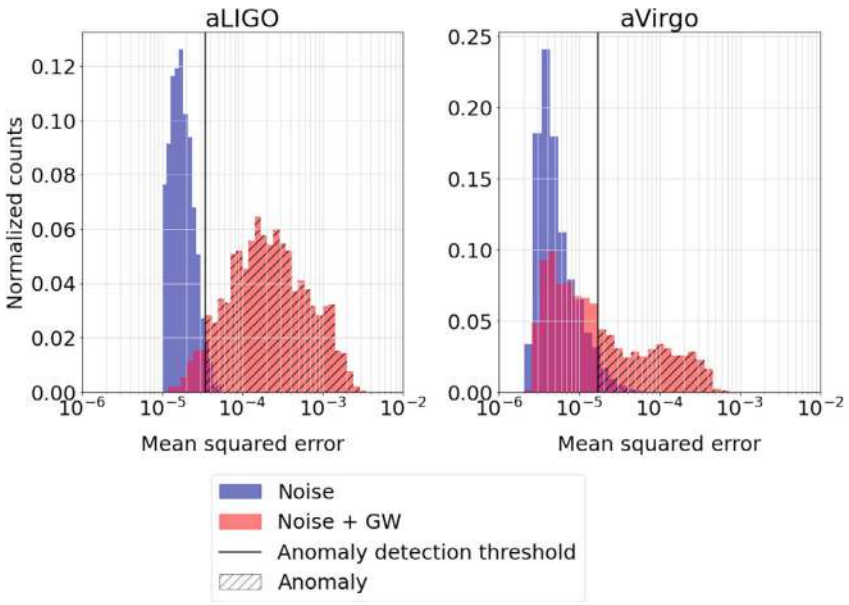


Fig. 14.9 Distribution of MSE between the AE predictions and the input strain for two types of studied signals: noise (blue histogram) and injected GW (red histogram). In the broad range of MSE for aLIGO dataset the injected GW had significantly higher MSE than noise, allowing a definite distinction between these signal types. Additional black vertical line representing anomaly DT was added to emphasize amount of detected anomalies (hatched areas)

The detection threshold for anomaly detection was determined by examining the relationship between the false positive rate (FPR) and MSE. By fixing the FPR at a predefined level, we set the detection threshold (DT) corresponding to the MSE. In this analysis, we fixed the FPR at 5%, resulting in the following thresholds: $DT_{simV} = 1.6 \cdot 10^{-5}$ for aVirgo and $DT_{simL} = 3.1 \cdot 10^{-5}$ for aLIGO. The results of anomaly searches at FPR = 5% are summarized in Table 14.1 as a confusion matrix. Additionally, the comparison of anomaly detection efficiency for both interferometers is illustrated in the left plot of Fig. 14.10 through Receiver Operating Characteristic (ROC) curves. Across all FPR ranges, the aLIGO detector demonstrated a significantly higher detection efficiency, or True Positive Rate (TPR).

The values presented in Table 14.1 quantitatively represent the findings from both panels in Fig. 14.9. The 'Anomaly' row corresponds to the hatched regions shown in the panels. For the aLIGO dataset, 96% of detected anomalies correctly corresponded to injected gravitational waves (GW), with minimal contamination from noise samples. In contrast, for the aVirgo dataset, 59% of samples—mostly those

Table 14.1 Results of anomaly detection of CNN-AE at FPR = 5% for aLIGO and aVirgo dataset in the form of confusion matrix. Columns relate to the ground-truth values whereas rows to the predictions. For aLIGO dataset significant majority of detected anomalies corresponded to the data samples with injected GW. However in case of aVirgo more than a half of data samples with injected GW did not exceed the DT_{simV} as a result of low SNR (see Fig. 14.6 for comparison between aLIGO and aVirgo GW SNR distributions)

	aLIGO		aVirgo	
	Injected GW (%)	Noise (%)	Injected GW (%)	Noise (%)
Anomaly	96	5	41	5
Non-anomaly	4	95	59	95

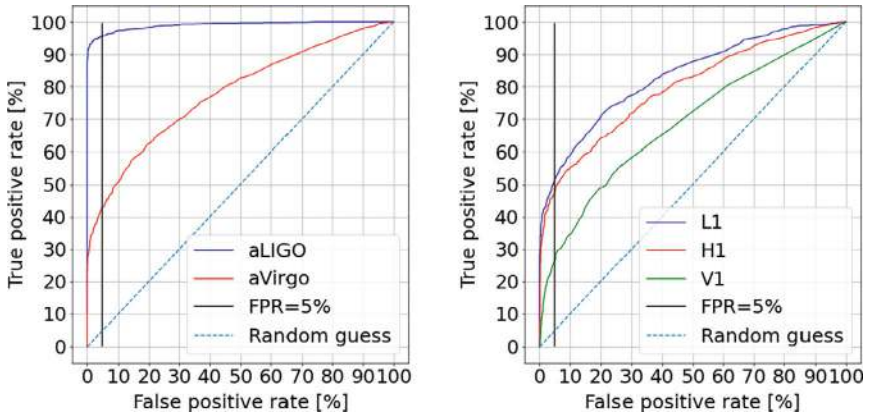


Fig. 14.10 Receiver operating characteristic curves for simulated data (*left plot*) and real data (*right plot*). Black vertical line corresponds to the $FPR = 5\%$ chosen as the criterion for the anomaly detection threshold

with low SNR (around 10 or less)—did not exceed the detection threshold (DT) and were thus classified as non-anomalous. Further details illustrating the relationship between the SNR of the injected GW and the MSE of the reconstructed waveform are provided in Appendix 14.6.2.

14.4.2 Anomaly Searches on Real Data

Subsequently, the autoencoders (AEs) were trained on whitened, real data from the O2 observational run, which was collected from three interferometers: $V1$, $L1$, and $H1$, with injected binary black hole (BBH) gravitational waveforms. The right plot in Fig. 14.7 illustrates the learning history of the AE for each detector's dataset. Similar to the results obtained from the simulated data, the AE reached convergence after approximately 100 epochs. The training was extended to assess potential overfitting, but none was observed. Since the difference between the 'anomalous' input and the 'anomalous-free' reconstruction for $V1$ was the smallest among the datasets, due to the lowest SNR range, the AE converged towards the lowest mean squared error (MSE). Correspondingly, the smaller SNR range for the $H1$ dataset, compared to the $L1$ dataset, resulted in lower MSE values for $H1$ than for $L1$.

As with the simulated data, the AE's ability to reconstruct the detector noise was evaluated. The differences between the input strain and AE reconstructions were compared against the expected values. Examples of these comparisons are shown in Fig. 14.11. A summary of the match between the injected and reconstructed waveforms is provided in Appendix 14.6.1. The AE trained on $L1$ data achieved the

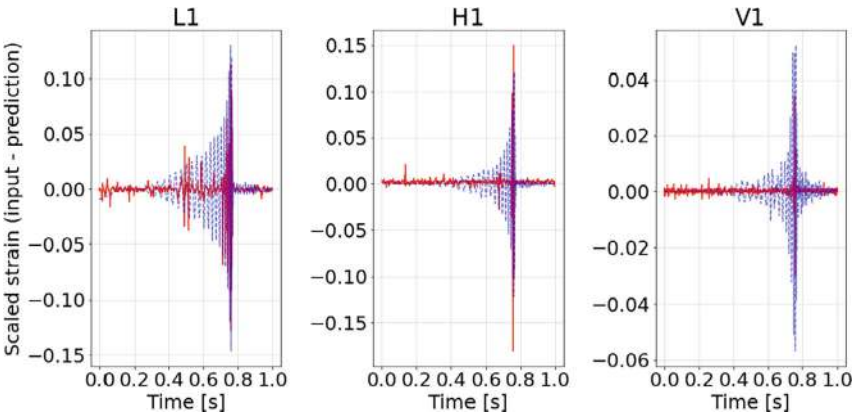


Fig. 14.11 Three examples of GW reconstruction using AE. Presented plots were generated after subtracting from the AE predictions, input signal (red curve). For the comparison we added the expected signal using blue, dashed curve (difference between the input signal and the ground-truth output signal). From left to right: results for $L1$, $H1$ and $V1$ detectors

best results, with the injected gravitational waves extracted predominantly during the merger phase, and partially during the inspiral phase. Conversely, the AEs trained on the other datasets primarily reconstructed the merger phase. In general, the AE struggled to reconstruct the lower amplitude and frequency components of the GW signal, such as the inspiral and ringdown phases.

To compute the anomaly detection threshold, we generated histograms of the MSE for each detector's dataset and compared the MSE values with the false positive rate (FPR). The results are displayed in Fig. 14.12. The anomalies spanned a broader range of MSE values than the noise, with an overlapping region that varied across datasets (highlighted in burgundy in Fig. 14.12). This overlap was smallest for the $L1$ dataset and largest for the $V1$ dataset.

As with the simulated data, the anomaly detection threshold was defined by assuming FPR = 5%, resulting in the following thresholds: $DT_{L1} = 1.3 \cdot 10^{-5}$ for $L1$, $DT_{H1} = 2.2 \cdot 10^{-5}$ for $H1$, and $DT_{V1} = 4.3 \cdot 10^{-6}$ for $V1$. The results of the anomaly searches on the real datasets are presented in Table 14.2 as a confusion matrix. The 'Anomaly' row corresponds to the hatched areas shown in the panels of Fig. 14.12.

For the $L1$ and $H1$ datasets, approximately half of the detected anomalies were correctly associated with injected GW signals. The samples that did not exceed their respective detection thresholds (DT) typically had low SNRs. This was also the case

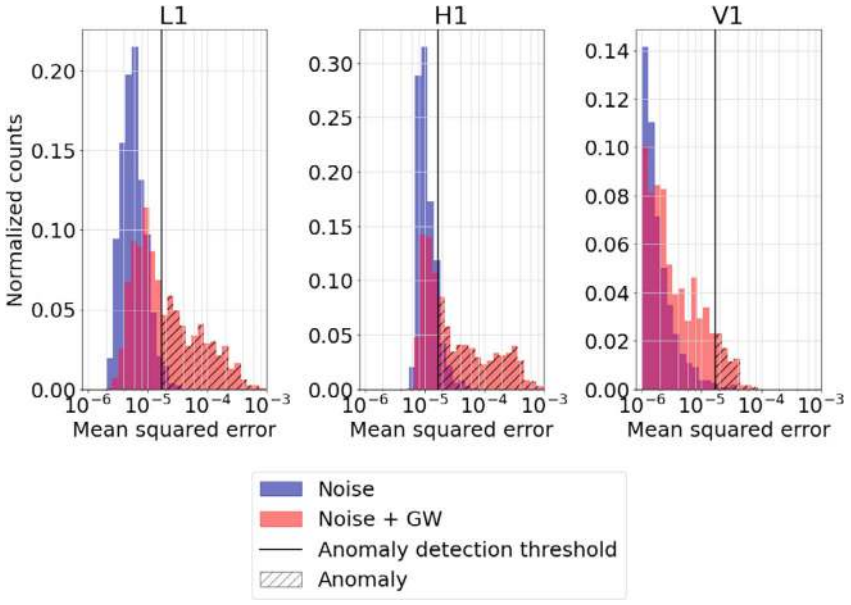


Fig. 14.12 Distribution of MSE between the AE predictions and the input strain for two types of studied signals: injected GW (red histograms) and noise (blue histograms). Plots correspond to the $L1$ (*left plot*), $H1$ (*middle plot*) and $V1$ (*right plot*) datasets. Additional black vertical line representing anomaly DT was added to emphasize amount of detected anomalies (hatched areas)

Table 14.2 Results of anomaly detection of CNN-AE at FPR=5% for $L1$, $H1$ and $V1$ datasets in the form of confusion matrix. Columns relate to the ground-truth values whereas rows to the predictions. For all datasets significant majority of detected anomalies correctly corresponded to the data instances with injected GW. However, more than a third part of non-anomalous class (samples that did not exceed DT for a given detector) related to the low SNR injected GW (see Fig. 14.6 for comparison between $L1$, $H1$ and $V1$ GW SNR distributions)

	$L1$		$H1$		$V1$	
	Inj. GW (%)	Noise (%)	Inj. GW (%)	Noise (%)	Inj. GW (%)	Noise (%)
Anomaly	52	5	50	5	27	5
Non-anomaly	48	95	50	95	73	95

for the $V1$ dataset, where only 27% of the samples exceeded DT_{V1}. Additional details regarding the relationship between the SNR of the injected GW signals and the MSE of the reconstructed waveforms for the real datasets can be found in Appendix 14.6.2.

Additionally, we conducted tests to evaluate the temporal localization of anomalies detected by our method. We compared the known times of the GW injections into the data with the times of the reconstructed signals. Specifically, we calculated the time difference between the maximum amplitude peaks of both signals and plotted histograms of these differences for all anomalies exceeding the previously computed detection thresholds (DT). For comparison, this procedure was also applied to the simulated datasets. The results are shown in Fig. 14.13.

In all cases examined, approximately 95% of detected anomalies were localized within 0.05 s intervals around the injection times. These findings suggest that this feature may not only be useful for detection and reconstruction but also holds poten-

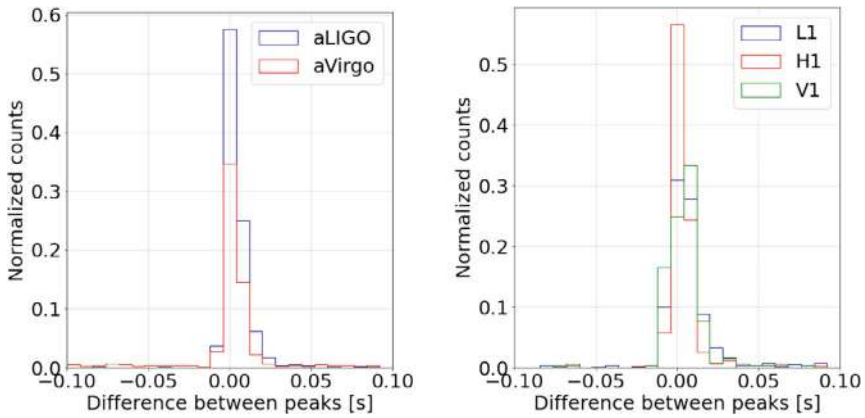


Fig. 14.13 Anomaly localisation based on the difference in peak positions between reconstructed and injected signals. *Left plot*: The results for the simulated datasets. *Right plot*: The results for the real datasets

tial for other applications, such as the temporal localization of signals across multiple detectors, and consequently, the sky localization of their sources.

14.4.3 Anomaly Searches on Confirmed GW Detections

Using a selection of real gravitational wave (GW) detections provided by the LIGO-Virgo collaboration via the GWOSC platform [40], we tested the autoencoder (AE) on three strong signals from the GWTC-1 O1-O2 catalog [39]: GW150914 [1], GW170608 [49], and GW170814 [50]. The reported network signal-to-noise ratio (SNR), $\rho_{net} = \sqrt{\sum_i \rho_i^2}$, was approximately 24 for GW150914, 15 for GW170608, and 16 for GW170814 [40]. Assuming two equally sensitive detectors, each would measure an SNR of approximately 17 for GW150914, 10 for GW170608, and 11 for GW170814. Notably, GW170814 was a three-detector event, with single-detector SNRs for $H1$, $L1$, and $V1$ of 7.3, 13.7, and 4.4, respectively [50]. Due to differences in detector sensitivity, these signals were registered with varying SNRs, though they remained near the single-detector SNR detection threshold, defined by the FPR=5% condition.

After whitening, the test data were input into the AE, and the reconstructed values were subtracted from the input data. The results of this subtraction for GW150914 are shown in the two upper-right plots of Fig. 14.14 for both LIGO detectors. The computed mean squared errors (MSE) were $MSE_{L1} = 3.2 \cdot 10^{-4}$ and $MSE_{H1} = 1.0 \cdot 10^{-3}$. For both detectors, the MSE values were significantly higher than the respective detection thresholds set at FPR=5%.

In the case of GW170608, the AE successfully detected the event, though the reconstructed signal was primarily limited to the merger phase for both LIGO detectors, as shown in the two middle-right plots in Fig. 14.14. This weaker reconstruction may be attributed to the different mass ranges of GW170608 compared to the GWs used in AE training. Specifically, the binary black hole (BBH) component masses for GW170608 were approximately $m_1 = 11.0$ and $m_2 = 7.6\ M_\odot$, which are notably smaller than the masses used in the training dataset (see Sect. 14.3 for more details). It is important to note that the H1 detector was nominally not in observing mode during this event, but the data were included using a modified segment list, as was done in the published analysis [40, 48, 49].

Despite this, the AE successfully detected GW170608, demonstrating its capacity to generalize and recognize gravitational waveforms it was not explicitly trained on, offering a significant advantage over the matched filtering method. Furthermore, the MSE values for both detectors were slightly above the detection thresholds at FPR=5%, with $MSE_{L1} = 5.3 \cdot 10^{-5}$ and $MSE_{H1} = 4.1 \cdot 10^{-5}$.

In the final test case, GW170814 was detected in both LIGO detectors, with a significant portion of the waveform successfully recovered, as illustrated in the two bottom-right plots of Fig. 14.14. However, for the $V1$ dataset, the AE failed to detect the event, likely due to the low reported SNR of $\simeq 4$ [50]. The MSE values for the

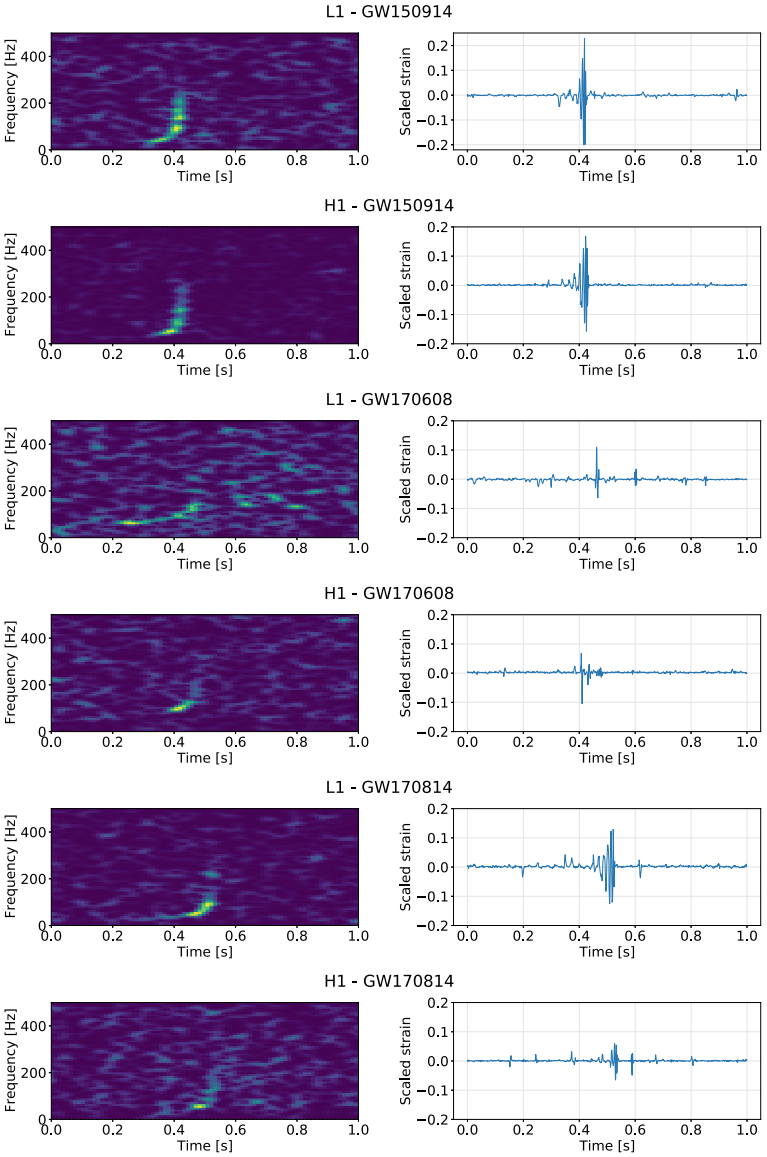


Fig. 14.14 Test of the CNN-AE on the dataset containing confirmed detections using $L1$ and $H1$ datasets for GW150914 (four upper plots), GW170608 (four middle plots) and GW170814 (four bottom plots). *Left plots*: spectrograms presenting the relation between frequency and time for the segment of real data containing GW. *Right plots*: the difference between the CNN-AE predictions and the input data

$L1$ and $H1$ datasets were above the detection threshold at 5%, with $MSE_{L1} = 2.2 \cdot 10^{-4}$ and $MSE_{H1} = 2.2 \cdot 10^{-4}$, whereas for $V1$, the MSE was below the threshold: $MSE_{V1} = 1.8 \cdot 10^{-6}$.

These findings confirm that the proposed CNN-AE method for anomaly detection is effective in identifying real gravitational wave events, even when the deep learning model is trained on relatively simple datasets. These datasets were constructed using information from specific GW waveform models, with limited variation in the range of masses and distances.

14.5 Summary

In this chapter, we demonstrated that autoencoders (AEs) are a promising method for anomaly detection in gravitational wave (GW) data. A simple AE architecture with three hidden layers was able to identify anomalies, including transient binary black hole (BBH) GWs and glitches in real data, whether trained on simulated or real datasets. Additionally, the method detected all three confirmed GW events tested, and even partially reconstructed their waveforms.

We introduced the mean squared error (MSE) as the metric for automatic anomaly detection, with a detection threshold defined by associating MSE with a false positive rate (FPR). At FPR=5%, nearly all injected GWs in LIGO's simulated dataset were detected, and around 50% of real detector data anomalies were identified. For Virgo, half of the injected signals in the simulated data were detected, while only a quarter were identified in real data. The poorer results on real data were due to the lower sensitivity of detectors during the O1-O2 observational runs compared to the designed sensitivity. However, improvements are expected with the O3 data after detector upgrades.

Our method also demonstrated high accuracy in localizing anomalies temporally, with all detected anomalies localized within 0.05 s of the injection time, which could be useful for sky localization of GW sources in multi-detector analyses. However, further study is required for such applications.

The successful detection of GW170608 showcased the generalization capability of the AE, detecting GWs with parameters different from those used for training, even in data nominally outside observing mode. Future projects include exploring recurrent neural networks (RNNs) for time-series data anomaly searches and investigating other GW types, such as core-collapse supernova signals.

Acknowledgements The work presented in this chapter was partially supported by the Polish NCN grants no. 2016/22/E/ST9/00037, 2017/26/M/ST9/00978 and 2020/37/N/ST9/02151, as well as the European Cooperation in Science and Technology COST action G2Net (CA17137). The Quadro P6000 used in this research was donated by the NVIDIA Corporation. The production computations were supported in part by PL-Grid Infrastructure and by the MNiSW grant for expansion of the strategic IT infrastructure for science. This research has made use of data, software and/or web tools obtained from the Gravitational Wave Open Science Center (https://www.gw-openscience.org/), a service of LIGO Laboratory, the LIGO Scientific Collaboration and the Virgo Collabo-

ration. LIGO Laboratory and Advanced LIGO are funded by the United States National Science Foundation (NSF) as well as the Science and Technology Facilities Council (STFC) of the United Kingdom, the Max-Planck-Society (MPS), and the State of Niedersachsen/Germany for support of the construction of Advanced LIGO and construction and operation of the GEO600 detector. Additional support for Advanced LIGO was provided by the Australian Research Council. Virgo is funded, through the European Gravitational Observatory (EGO), by the French Centre National de Recherche Scientifique (CNRS), the Italian Istituto Nazionale di Fisica Nucleare (INFN) and the Dutch Nikhef, with contributions by institutions from Belgium, Germany, Greece, Hungary, Ireland, Japan, Monaco, Poland, Portugal, Spain.

14.6 Appendices

14.6.1 Match/Overlap Between the Injected and Reconstructed Waveforms

14.6.1.1 Simulated Dataset

To measure the match between the injected and the reconstructed waveforms we used the normalized scalar product $< x_1|x_2 >$ in the time domain resulting in values in a range $(0, 1)$. Zero related to no match, whereas one to full match between the waveforms. Presented results in Fig. 14.15 corresponds to the whole studied dataset

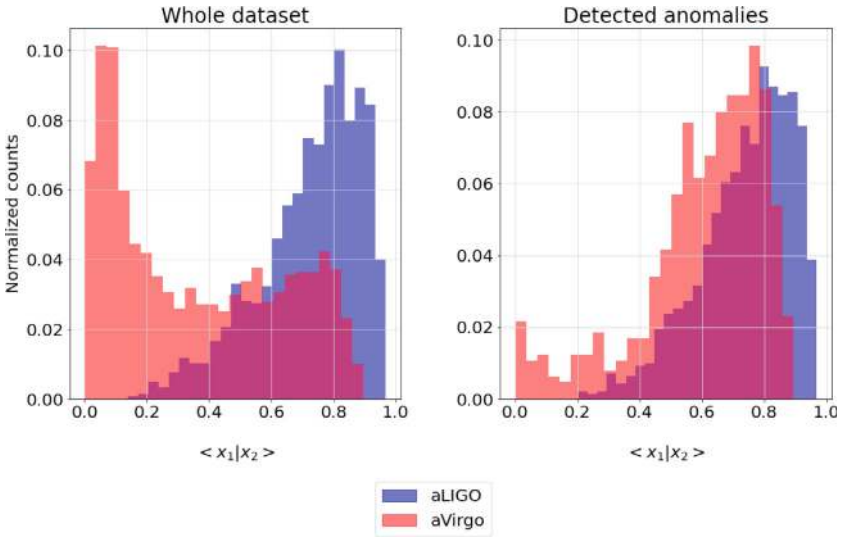


Fig. 14.15 Distribution of a match between the injected (x_1) and the reconstructed (x_2) waveforms as a function of normalized scalar product $< x_1|x_2 >$. Scalar product equals to one relates to the full match between waveforms whereas value of zero—no match. *Left plot*: results for the whole studied dataset of particular detector; *right plot*: results of a match after applying the anomaly detection threshold at FPR = 5%

for aLIGO and aVirgo detectors (left panel) as well as samples exceeding the anomaly detection thresholds (right panel). Applying respective DT allowed to substantially reduce number of samples with no match between the waveforms as a result of low SNR of injected GW. Samples exceeding DT were reconstructed to a greater extent which resulted in $< x_1|x_2 >$ closer to one.

14.6.1.2 Real Dataset

The same metric as in case of simulated dataset was used to study the match between the injected and the reconstructed waveforms for the real datasets. Presented in Fig. 14.16 results show the similarities in the match between consecutive datasets ($L1$, $H1$ and $V1$) as well as the effect of applying the anomaly detection threshold. Samples with $< x_1|x_2 >$ close to zero had low SNR. As a result they were poorly reconstructed which in turn translated into low value of MSE. Samples exceeding respective DT were reconstructed to a greater extent as in the case of simulated data. However, overall match was worse—the mean values of $< x_1|x_2 >$ for real datasets were around 0.6, whereas for simulated data 0.8.

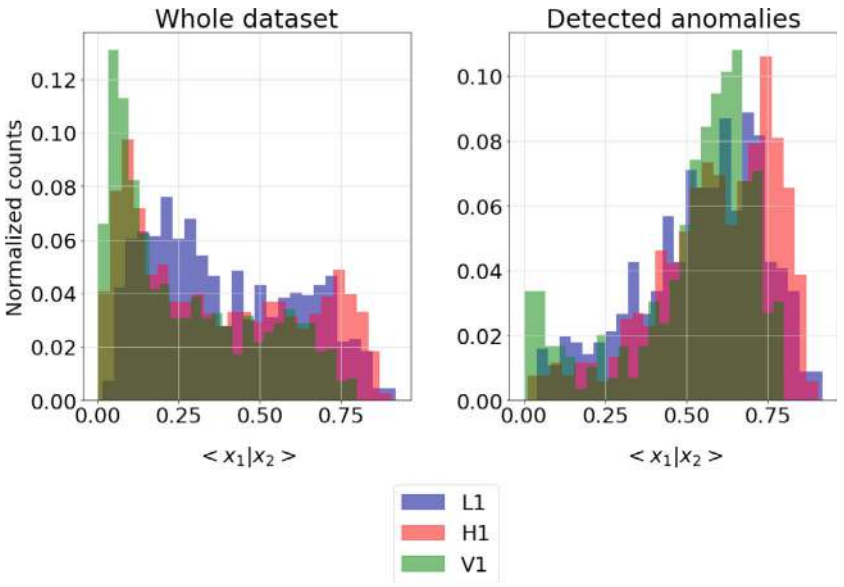


Fig. 14.16 Distribution of a match between the injected (x_1) and the reconstructed (x_2) waveforms as a function of normalized scalar product $< x_1|x_2 >$. Scalar product equals to one relates to the full match between waveforms whereas value of 0—no match. *Left plot*: results for the whole studied dataset of particular detector; *right plot*: results of a match after applying the anomaly detection threshold at FPR = 5%

14.6.2 Signal-to-Noise Ratio Versus Mean Squared Error

14.6.2.1 Simulated Dataset

For the aLIGO dataset above SNR = 20, the MSE-SNR relation was almost linear, with a small spread of individual data instances along MSE. Whereas with the decline of SNR, the spread of MSE significantly increased, characterized by the non-linearity in the MSE-SNR relation. Anomalies around the same SNR for values below 10 varied up to an order of magnitude in MSE. Manual inspection of data samples containing anomalies of low SNR provided an explanation of this behaviour. In the analysed samples, only the merger part of the gravitational waveform was recovered. For lower SNRs (below 10) the recovery was partial and dependent on the variability of the noise. If the amplitude of the noise in a given data segment was small comparing to the injected GW signal, the resulting MSE was higher than in the case of noise samples with larger amplitudes. Overall, for the aLIGO dataset, the susceptibility of AE to the local variability of the noise was inversely proportional to the SNR of the injected anomaly.

In case of the aVirgo dataset, the mentioned susceptibility was more significant. Matched filter SNRs for all injected GWs covered a smaller range of values than for aLIGO (compare SNR ranges on the bottom right plot in Fig. 14.6 for aLIGO and aVirgo datasets). In the majority of studied cases, the recovery of the anomaly was partial and limited to the merger part (Fig. 14.17).

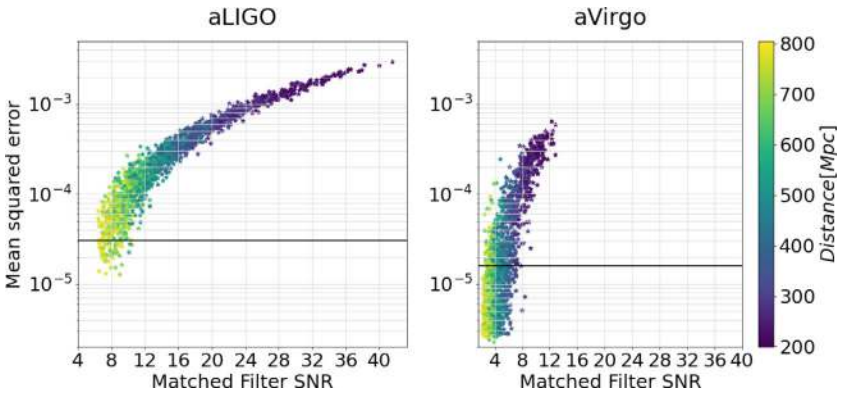


Fig. 14.17 Relation between MSE and matched filter SNR for aVirgo and aLIGO datasets colored with a distance to the GW source. Black horizontal line corresponds to the detection threshold of anomalies at FPR = 5%

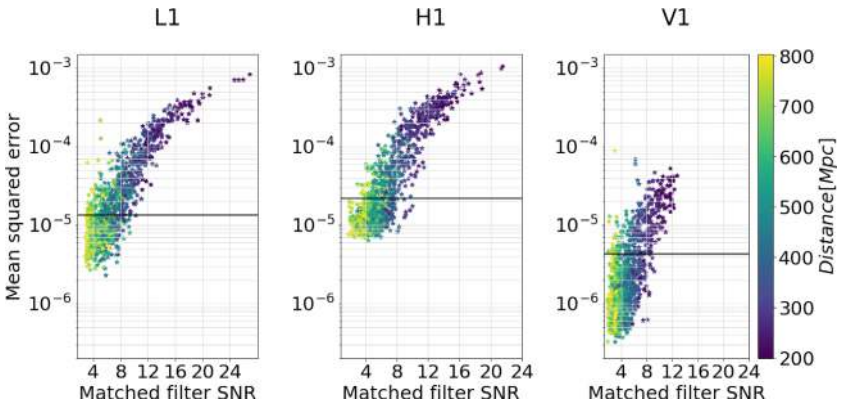


Fig. 14.18 Relation between the MSE and the matched filter SNR for $L1$ (*left plot*), $H1$ (*middle plot*) and $V1$ (*right plot*) datasets colored with a distance to the GW source. Black horizontal line corresponds to the detection threshold of anomalies at FPR $= 5\%$

14.6.2.2 Real Dataset

The relation between SNR and MSE presented similar features as for the simulated data discussed above. The variability of MSE for samples of similar SNR was largest among the weakest anomalies. The increase of the matched filter SNR led to the decrease of the spread in MSE, and thus their relation became more linear. This was the case for the AEs trained on the $L1$ and $H1$ datasets. In contrast, the AE trained on the $V1$ dataset was more susceptible to the local variability of the noise. Since the injected anomalies had a small SNR for $V1$ (majority of injected GW had SNR below 10), their amplitude was significantly lower than the detectors noise. As a result, the anomaly detection depended on the variability of the noise, which had a random character (Fig. 14.18).

References

1. Abbott, B.P., Abbott, R., Abbott, T.D., Abernathy, M.R., et al.: Phys. Rev. Lett. **116** 061102 (2016). arXiv:1602.03837
2. Aasi, J., Abbott, B.P., Abbott, R., Abbott, T., et al.: Class. Quantum Grav. **32**, 074001 (2015). arXiv:1411.4547
3. Acernese, F., Agathos, M., Agatsuma, K., Aisa, D., et al.: Class. Quantum Gravity **32**, 024001 (2015). arXiv:1408.3978
4. O3 summary, see more: https://www.ligo.caltech.edu/WA/news/ligo20200326
5. Virgo status, see more: https://www.virgo-gw.eu/status.html
6. Gravitational-Wave Candidate Event Database, see more: https://gracedb.ligo.org/superevents/public/O3/
7. Abbott, R., Abbott, T.D., Abraham, S., et al.: GWTC-2: compact binary coalescences observed by LIGO and Virgo during the first half of the third observing run (2020). arXiv:2010.14527

8. Abbott, B.P., Abbott, R., Abbott, T.D., Abraham, S., Acernese, F., Ackley, K., Adams, C., Adhikari, R.X., Adya, V.B., Affeldt, C., et al.: Astron. J. **882** (2019) L24 ISSN 2041-8213. https://doi.org/10.3847/2041-8213/ab3800
9. Acernese, F., Agathos, M., Agatsuma, K., Aisa, D., et al.: Phys. Rev. D **93**, 122003 (2016). arXiv:1602.03839
10. Kumar, A., Saminadayar, L., Glattli, D.C., Jin, Y., Etienne, B.: Phys. Rev. Lett. **76**(15), 2778–2781 (1996). https://link.aps.org/doi/10.1103/PhysRevLett.76.2778
11. Owen, B., Sathyaprakash, B.: Phys. Rev. D **60**(2), 022002 (1999). https://link.aps.org/doi/10.1103/PhysRevD.60.022002
12. Usman, S.A., Nitz, A.H., Harry, I.W., Biwer, C.M., Brown, D.A., Cabero, M., Capano, C.D., Dal Canton, T., Dent, T., Fairhurst, S., Kehl, M.S., Keppel, D., Krishnan, B., Lenon, A., Lundgren, A., Nielsen, A.B., Pekowsky, L.P., Pfeiffer, H.P., Saulson, P.R., West, M., Willis, J.L.: Class. Quantum Gravity **33**, 215004 (2016) arXiv:1508.02357
13. Sachdev, S., et al.: (2019). arXiv:1901.08580
14. Klimenko, S., et al.: Phys. Rev. D **93**, 042004 (2016). arXiv:5110.5999
15. Klimenko, S., et al.: Class. Quantum Gravity **25**, 114029 (2008). http://stacks.iop.org/0264-9381/25/i=11/a=114029
16. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. The MIT Press (2016). ISBN 0262035618, 9780262035613
17. Samuel, A.L.: IBM J. Res. Dev. **3**, 210–229 (1959)
18. Robinet, F., Arnaud, N., Leroy, N., Lundgren, A., Macleod, D., McIver, J.: SoftwareX **12**, 100620 (2020). ISSN 2352-7110, https://www.sciencedirect.com/science/article/pii/S2352711020303332
19. Chatterji, S., Blackburn, L., Martin, G., Katsavounidis, E.: Class. Quantum Gravity **21**, S1809–S1818 (2004). https://doi.org/10.1088/0264-9381/21/20/024
20. George, D., Huerta, E.: Phys. Lett. B **778**, 64–70 (2018). ISSN 0370-2693, http://www.sciencedirect.com/science/article/pii/S0370269317310390
21. Shen, H., George, D., Huerta, E.A., Zhao, Z.: Denoising gravitational waves with enhanced deep recurrent denoising auto-encoders. In: ICASSP 2019—2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3237–3241 (2019)
22. Dreissigacker, C., Sharma, R., Messenger, C., Zhao, R., Prix, R.: Phys. Rev. D **100**(4), 044009 (2019). https://link.aps.org/doi/10.1103/PhysRevD.100.044009
23. Morawski, F., Bejger, M., Ciecielag, P.: Mach. Learn.: Sci. Technol. (2020). http://iopscience.iop.org/10.1088/2632-2153/ab86c7
24. Beheshtipour, B., Papa, M.A.: Phys. Rev. D **101**(6), 064009 (2020). https://link.aps.org/doi/10.1103/PhysRevD.101.064009
25. Razzano, M., Cuoco, E.: Class. Quantum Gravity **35**, 095016 (2018). https://doi.org/10.1088/1361-6382/aab793
26. Iess, A., Cuoco, E., Morawski, F., Powell, J.: Mach. Learn.: Sci. Technol. **1**, 025014 (2020). https://doi.org/10.1088/2632-2153/ab7d31
27. Corizzo, R., Ceci, M., Zdravevski, E., Japkowicz, N.: Scalable auto-encoders for gravitational waves detection from time series data. Expert Syst. Appl. **151**, 113378 (2020). ISSN 0957-4174, http://www.sciencedirect.com/science/article/pii/S0957417420301986
28. Giles, D., Walkowicz, L.: Mon. Not. Roy. Astron. Soc. **484**, 834–849 (2019). arXiv:1812.07156
29. D'Addona, M., Riccio, G., Cavuoti, S., Tortora, C., Brescia, M.: Anomaly detection in astrophysics: a comparison between unsupervised deep and machine learning on kids data (2020). arXiv:2006.08235
30. Baron, D.: (2019). arXiv:1904.07248
31. Farina, M., Nakai, Y., Shih, D.: Phys. Rev. D **101**(7), 075021 (2020). https://link.aps.org/doi/10.1103/PhysRevD.101.075021
32. Baldi, P.: Autoencoders, unsupervised learning and deep architectures. In: Proceedings of the 2011 International Conference on Unsupervised and Transfer Learning Workshop—UTLW'11 (JMLR.org), vol. 27, pp 37–50 (2011)
33. Kingma, D.P., Ba, J.: (2014). arXiv:1412.6980

34. Van Rossum, G., Drake, F.L.: Python 3 Reference Manual (Scotts Valley, CA: CreateSpace) (2009). ISBN 1441412697
35. Chollet, F., et al.: (2015) Keras. https://keras.io
36. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., et al.: TensorFlow: large-scale machine learning on heterogeneous systems software available from tensorflow.org (2015). https://www.tensorflow.org
37. Nickolls, J., Buck, I., Garland, M., Skadron, K.: Queue **6**, 40–53 (2008). ISSN 1542-7730. https://doi.org/10.1145/1365490.1365500
38. Chetlur, S., Woolley, C., Vandermersch, P., Cohen, J., Tran, J., Catanzaro, B., Shelhamer, E.: (2014). arXiv:1410.0759
39. Abbott, B.P., Abbott, R., Abbott, T.D., Abraham, S., Acernese, F., Ackley, K., Adams, C., et al.: Phys. Rev. X **9**, 031040 (2019). arXiv:1811.12907
40. Collaboration, T.L.S., The Virgo Collaboration, Abbott, R., Abbott, T.D., et al.: (2019) arXiv:1912.11716
41. Cuoco, E., Calamai, G., Fabbroni, L., Losurdo, G., Mazzoni, M., Stanga, R., Vetrano, F.: Class. Quantum Gravity **18**, 1727–1751 (2001). https://doi.org/10.1088/0264-9381/18/9/309
42. Nitz, A., Harry, I., Brown, D., Biwer, C.M., Willis, J., Canton, T.D., Capano, C., Pekowsky, L., Dent, T., Williamson, A.R., Davies, G.S., De, S., Cabero, M., Machenschalk, B., Kumar, P., Reyes, S., Macleod, D., Finstad, D., Pannarale, F., Massinger, T., Tápai, M., Singer, L., Kumar, S., Khan, S., Fairhurst, S., Nielsen, A., Singh, S., Gadre, B.U.V., Dorrington, I.: gwastro/pycbc: Pycbc release v1.16.10 (2020). https://doi.org/10.5281/zenodo.4063644
43. Hannam, M., Schmidt, P., Bohé, A., Haegel, L., Husa, S., Ohme, F., Pratten, G., Pürrer, M.: Phys. Rev. Lett. **113**(15), 151101 (2014). https://link.aps.org/doi/10.1103/PhysRevLett.113.151101
44. Salpeter, E.E.: Astrophys. J. **121**, 161 (1955)
45. The Virgo Collaboration.: Advanced Virgo Baseline Design, note VIR-027A-09, May 16 2009 (2009). https://tds.virgo-gw.eu/?call_file=VIR-0027A-09.pdf
46. Virgo Interferometer Monitoring webpage. https://vim-online.virgo-gw.eu
47. Updated Advanced LIGO sensitivity design curve. https://dcc.ligo.org/LIGO-T1800044/public
48. Abbott, B.P., Abbott, R., Abbott, T.D., Abraham S.E.A.: (LIGO Scientific Collaboration and Virgo Collaboration). Phys. Rev. X **9**(3), 031040 (2019). https://link.aps.org/doi/10.1103/PhysRevX.9.031040
49. Abbott, B.P., Abbott, R., Abbott, T.D., Acernese, F., Ackley, K., Adams, C., Adams, T., Addesso, P., Adhikari, R.X., Adya, V.B., Affeldt, C., Afrough, M., Agarwal, B., Agathos, M., Agatsuma, K., Aggarwal, N., Aguiar, O.D., Aiello, L., Ain, A., Ajith, P., Allen, B., Allen, G., Allocca, A., et al.: Astrophys. J. Lett. **851**, L35 (2017). arXiv:1711.05578
50. Abbott, B.P., Abbott, R., Abbott, T.D., Acernese, F., Ackley, K., Adams, C., Adams, T., Addesso, P., Adhikari, R.X., Adya, V.B., Affeldt, C., Afrough, M., Agarwal, B., Agathos, M., Agatsuma, K., Aggarwal, N., Aguiar, O.D., Aiello, L., Ain, A., Ajith, P., Allen, B., Allen, G., Allocca, A., Altin, P.A., Amato, A., Ananyeva, A., Anderson, S.B., Anderson, W.G., Angelova, S.V., Antier, S., Appert, S., Arai, K., Araya, M.C., Areeda, J.S., Arnaud, N., et al.: (LIGO Scientific Collaboration and Virgo Collaboration) Phys. Rev. Lett. **119**(14), 141101 (2017). https://link.aps.org/doi/10.1103/PhysRevLett.119.141101

Chapter 15
One-Class Learning for Gravitational Waves Detection

Roberto Corizzo and Eftim Zdravevski

Abstract Gravitational waves are an exceptional opportunity for studying and interpreting phenomena from the universe. Automatic signal processing and machine learning techniques can provide significant support for the efficient detection and analysis of gravitational waves from large-scale data continuously collected by interferometers. This chapter discusses two approaches involving deep auto-encoder models to analyze and classify raw time series data into noise or gravitational waves. The goal is to provide astrophysicists with a tool that can quickly discard noisy time series and identify time series that potentially contain an actual astrophysical phenomenon. Experiments carried out on three datasets show that the discussed approaches implemented in a scalable manner using Apache Spark represent a valuable machine learning approach for astrophysical analysis, offering competitive accuracy compared to state-of-the-art methods.

15.1 Introduction

Gravitational Waves (GWs) represent a fascinating new frontier, offering us the chance to explore and interpret the mysteries of the universe. However, their analysis presents a multitude of intricate challenges, demanding specific expertise due to the complex nature of the data collected by detectors (interferometers), which are time series affected by environmental and instrumental noise.

Additional knowledge is required to perform other recurrent tasks, such as spectrogram analysis, filtering, and whitening, as outlined by [2, 3]. Furthermore, the

R. Corizzo (✉)
Department of Computer Science, American University, 4400 Massachusetts Avenue NW, 20016 Washington DC, USA
e-mail: rcorizzo@american.edu

E. Zdravevski
Faculty of Computer Science and Engineering, University Ss. Cyril and Methodius, Rugjer Boshkovikj 16, Skopje 1000, North Macedonia
e-mail: eftim.zdravevski@finki.ukim.mk

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_15

sheer volume of data collected by detectors, reaching petabytes per day, underscores the pressing need for new, automatic, and scalable methods. These methods must be capable of analyzing data in a timely manner and performing pre-processing and detection tasks that were traditionally executed manually.

For this reason, the adoption of automatic signal processing and machine learning techniques has been recognized as beneficial for their detection and analysis and reduces the effort and the cost of standard approaches such as template matching.

Machine-learning approaches for gravitational waves analysis have gained traction in recent years. However, most of the existing approaches require knowledge of gravitational waves signals, which is possible only in a supervised learning setting [4].

One-class learning models have widespread adoption in unsupervised and semi-supervised learning settings, particularly in cybersecurity and industrial process monitoring. Their usefulness lies in the fact that they do not require knowledge of anomalous patterns for the model training phase.

This consideration is particularly relevant in gravitational waves analysis since the majority of data collected by detectors is noise, resulting in a large amount of imbalanced data. In this context, the noise can be considered as the background data (abundant) used for model training, whereas the anomalous data (scarce) can be seen as the gravitational waves we aim to detect.

This chapter addresses gravitational wave detection from a one-class learning perspective. Specifically, we discuss two approaches involving deep auto-encoder models to analyze time series collected from detectors and provide a classification label (noise or an actual signal). The purpose is to accurately discard noisy time series and identify potentially intriguing ones that could be manually inspected. Our approaches leverage the abundance of noise time series data, which, in principle, has the potential to empower models with a greater degree of flexibility. This would allow them to detect gravitational waves with different morphologies without any explicit or pre-existing knowledge of such phenomena. We show that such an approach is effective since models trained exclusively with noise time series can detect gravitational waves as out-of-distribution data points, leveraging anomaly scores extracted from newly observed data. Moreover, our results show that these approaches are scalable, which is essential considering the large amount of data continuously collected by detectors.

15.2 Related Works

Astrophysical data analysis often involves handling noisy time series data. Techniques like data whitening in the time and frequency domains help remove stationary noise to reveal weak signals [2, 5]. Machine learning methods are crucial for tasks such as noise removal, anomaly detection, and classification of gravitational waves (GWs). These methods include supervised convolutional neural networks (CNNs) for glitch classification, as seen in Gravity Spy [4, 6]. Other approaches include

using CNNs for feature extraction and unsupervised clustering to identify new glitch classes [7, 8]. Despite the high accuracy of these methods, they rely on pre-processed spectrograms, which assume feasible human-intensive pre-processing. In contrast, this chapter focuses on real-time classification directly on strain data, assessing model performance under various conditions.

Time series classification methods are supervised and fall into three categories: feature-based, distance-based, and model-based [9]. Feature-based methods convert time series into feature vectors for classification. Distance-based methods, such as k-Nearest Neighbor, use similarity measures for classification. Model-based methods employ generative models like Naive Bayes and Hidden Markov Models. Supervised methods are useful with labeled datasets, while unsupervised methods are vital for real-time data streams, where labels are not available. An unsupervised anomaly detection approach can differentiate between normal and anomalous behaviors over time [10].

Anomaly detection techniques, as outlined by [11], include point, contextual, and collective anomalies. This chapter focuses on collective anomalies in time series data. Auto-encoders have shown high performance in this task due to their ability to learn low reconstruction error representations, as demonstrated by [12–14].

Auto-encoders are effective for data denoising, anomaly detection, and feature extraction in noisy time series data. This chapter presents two approaches (unsupervised and supervised) using auto-encoders combined with classification models to analyze and classify time series as noise or real signals. The preference for classification over denoising allows for automatic analysis of continuous data, enabling researchers to focus on potentially intriguing time series. Unlike previous approaches requiring pre-processing steps, our methods work directly on raw data collected by detectors.

Our methods aim to: detect GWs automatically from time series data; recognize various noise types; and efficiently scale with data volume using a distributed implementation on the Apache Spark framework. Compared to similar works [4, 6–8, 15, 16], our approach does not require pre-processing, and we analyze real annotated GWs events rather than simulated waveforms [17].

While models based on CNNs and RNNs are typically more complex and time-consuming to train, we focus on simpler auto-encoders for a balance between accuracy and scalability, facilitating efficient distributed training on computational clusters.

15.3 Methods

In this section, we describe two approaches for GWs detection in time series. The first approach exploits auto-encoder models to classify strain time series data with an unsupervised anomaly detection strategy, whereas the second one uses auto-encoders for feature extraction and performs supervised classification.

15.3.1 Time Series Classification via Anomaly Detection (AE)

In this chapter, we utilize auto-encoders as outlined in [18], due to their proven effectiveness in feature extraction and anomaly detection [13]. Our method involves training the auto-encoder with one-class data. During prediction, a high reconstruction error indicates an anomaly, potentially signifying an actual GW signal.

The idea is to train the auto-encoder with noise data (negative or normal class), which is abundant compared to the few GW signals. Unseen time series are classified based on their reconstruction error. Generally, noise is continuously present, whereas GW signals are rare and sporadic.

Each auto-encoder consists of an encoding function γ and a decoding function δ, aiming to minimize reconstruction loss:

$$\begin{aligned} \gamma : \mathcal{X} \to \mathcal{F}, \qquad \delta : \mathcal{F} \to \mathcal{X}, \\ \gamma, \delta = \arg\min_{\gamma,\delta} \|X - \delta(\gamma(X))\|^2, \end{aligned} \tag{15.1}$$

where $\mathcal{X}$ is the input space (time series) and $\mathcal{F}$ is the feature space.

The parametric, differentiable functions γ and δ are optimized to minimize reconstruction loss through backpropagation, extracting reduced-dimensionality vectors from raw data. The auto-encoder can have multiple hidden layers, with the final layer reconstructing the input. For classification, typically only the encoding part is used, as depicted in Fig. 15.1.

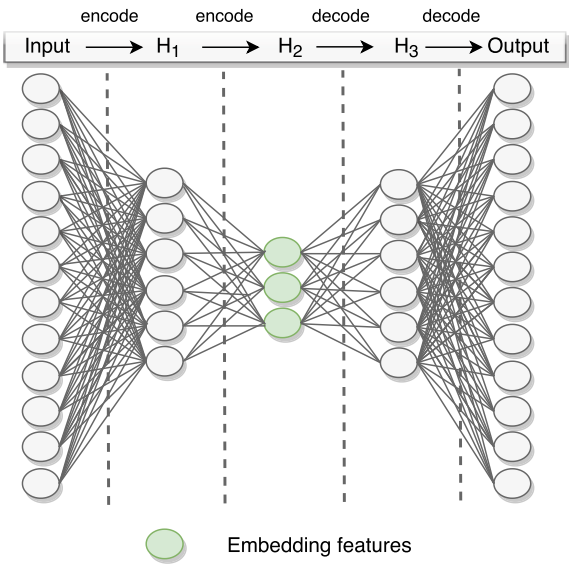


Fig. 15.1 Feature extraction process performed using the encoding function of the trained auto-encoder [19]

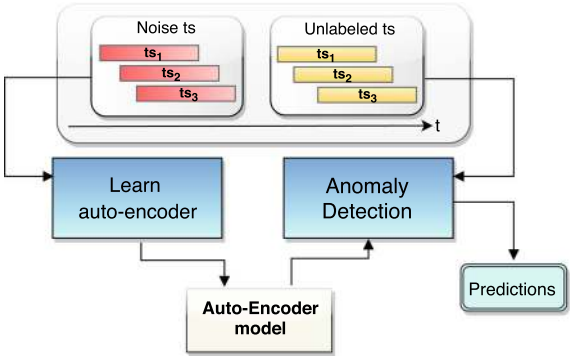


Fig. 15.2 Workflow of the AE method. Negative class time series (red) are exploited to train an auto-encoder model that accurately reconstructs noisy time series. At prediction time, a new time series of unknown class (yellow) is provided as input to the model, and the prediction class is returned, depending on the reconstruction error observed [19]

In the AE approach, the final layer mirrors the input layer for time series reconstruction. Conversely, the AE-FE approach uses only the encoding stage for feature extraction, utilized by classification models.

With one hidden layer, the encoding stage of an auto-encoder maps input $\mathbf{x} \in \mathbb{R}^d$ to a hidden representation $\mathbf{z} \in \mathbb{R}^p$ using σ, a sigmoid or rectified linear unit, weight matrix W, and bias vector b: $\mathbf{z} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b})$. The decoding stage reconstructs $\mathbf{x}$ from $\mathbf{z}$ as $\mathbf{x}' = \sigma'(\mathbf{W}'\mathbf{z} + \mathbf{b})$, minimizing the loss $\mathcal{L}_2(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|^2 = \|\mathbf{x} - \sigma'(\mathbf{W}'(\sigma(\mathbf{W}\mathbf{x} + \mathbf{b})) + \mathbf{b}')\|^2$.

During training, a threshold for the maximum allowed reconstruction error is calculated to distinguish between noise and real data. Instances with errors above this threshold are classified as actual GW time series, otherwise as noise. This process is illustrated in Fig. 15.2.

The threshold must account for the data distribution of reconstruction errors and possible changes over time (concept drift). It is recalculated during each training session as $\overline{e} + f \cdot \sigma_e$, following a one-tailed sigma rule from [20], where f is the factor for standard deviation, $\overline{e}$ is the average error, and σ_e is the standard deviation.

This automatic threshold calculation avoids manual definition, preventing performance degradation due to data distribution changes. Defining the threshold by standard deviations from the mean allows automatic adjustment based on training data error distribution. Threshold selection significantly impacts anomaly detection performance, as noted by [21–24].

While reconstruction error-based anomaly detection is known, it has not been applied to GWs before. One major drawback is the need for accurate threshold estimation, which is resolved by our automatic method, accounting for changes in the learned distribution.

15.3.2 *Feature Extraction with Supervised Classification (AE-FE)*

In deep neural networks, hidden layers capture increasingly abstract features, forming a hierarchy of complexity. Studies by [25, 26] demonstrate that deep auto-encoders maintain the feature extraction capabilities of traditional auto-encoders, enabling the creation of low-dimensional embeddings as shown in [27, 28].

This approach exclusively utilizes auto-encoders for feature extraction by defining hidden layers with fewer neurons, creating a reduced-dimensionality feature space known as bottleneck features. The encoding function of a trained auto-encoder transforms the input feature space F, with $|F|$ features, into a new feature space H_1 with $|H1| \ll |F|$. A subsequent encoding stage further reduces the dimensionality to H_2 with $|H2| \ll |H1|$.

By adopting auto-encoders, we extract a high-level, low-dimensional feature space, mitigating collinearity among features as discussed by [14], thereby enhancing the reliability of learning tasks. Extracted features are then used by a prediction model for classification.

We employ various prediction models, identified by a suffix in the method name. For instance, using Gradient-Boosted Trees (GBTs) results in AE-GBT. GBTs, as explained by [29], are ensemble-based and iteratively improve by minimizing a loss function. They model non-linear interactions effectively and have shown high performance in predictive tasks [30–33]. Other classifiers used include Logistic Regression (AE-LR), ExtraTrees (AE-ERT), Random Forest (AE-RF), XGBoost (AE-XGB), and Support Vector Machines (AE-SVM). In experiments, the AE-FE method is denoted by the chosen classifier (e.g., AE-LR, AE-ERT, AE-GBT, AE-XGB, AE-SVM).

Though commonly a final classification layer is added to the auto-encoder model, this requires a two-step optimization process. Our focus on accuracy and scalability leads us to prefer traditional machine learning methods for classification, which achieve high predictive accuracy with lower training time.

The feature extraction process is illustrated in Fig. 15.1, and the AE-FE approach in Fig. 15.3. This method trains supervised models on features extracted by auto-encoders, even across different classes. Even if the auto-encoder is trained on the negative class (noise), it can extract features for the positive class, assuming labeled data is available. Thus, the supervised model discriminates classes based on differences in encoded vector representations.

15.4 Experiments

In this section, we describe the datasets and experimental results obtained from two approaches: time series classification via anomaly detection (AE) and feature extraction with supervised classification (AE-FE).

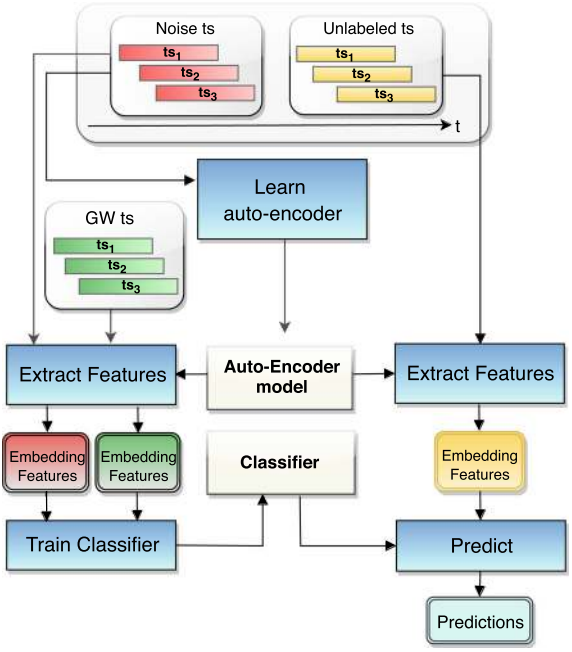


Fig. 15.3 Workflow of the AE-FE method. Negative class time series (red) are exploited to train an auto-encoder model that accurately reconstructs noisy time series. The auto-encoder encodes both negative class (red) and positive class (green) time series and the resulting representation is used to train a classifier. At prediction time, a new time series of unknown class (yellow) is provided to the classification model, which returns its prediction [19]

15.4.1 Data Description

We analyze time series data representing gravitational strain collected by detectors, which measure fractional changes in the lengths along two axes (x-arm and y-arm cavities) $h(t) = \frac{\Delta L_y - \Delta L_x}{L}$. The raw data, available in hdf5 format, includes three files: **meta** (metadata), **quality** (data quality), and **strain** (time series data).

Our dataset comprises positive class time series from four published gravitational wave discoveries (GW150914, LVT151012, GW151226, and GW170104) detected by LIGO interferometers, as described in [34, 35]. These short-duration BBH (Binary Black Hole) events are sampled at 4096 Hz, resulting in 1 and 3-second time series (4096 and 12288 elements, respectively). Each event includes four time series: data strain from H1 (Hanford) and L1 (Livingston) interferometers and their whitened versions after preprocessing.

Inspired by [36, 37], we perform data augmentation by shifting time series by offsets $o \in \{-0.25, -0.125, 0.125, 0.25\}$ s, generating 20 time series per event, totaling 80 time series.

To generalize over different types of noise, our dataset includes negative class time series from 70 different noise types generated by the PyCBC Python library, which is widely used for GWs data generation [17, 38, 39]. We also extract real gravitational noise near real GWs events (3–15 s). Time series are normalized using min-max normalization.

Table 15.1 Overview of the datasets used in the experiments. All variants of GW2 with varying noise rates present the same characteristics [19]

Dataset	Class	Number of time series	Length of time series (1 s)	Length of time series (3 s)
GW1/GW2	Signal	80	4096	12288
GW1/GW2	Noise	70	4096	12288
GW3	Signal	104	4096	12288
GW3	Noise	92	4096	12288
GW4	Noise	50,000	4096	NA

We create three datasets:

- **GW1**: Contains positive and negative class time series as described.
- **GW2**: Negative class time series from GW1, while positive class time series P are blended from whitened signals W and noisy series N at different noise rates ($\alpha = 0.1, 0.25, 0.50$). For each element i, the formula is:

$$\mathbf{p}[i] = (1 - \alpha) \cdot \mathbf{w}[i] + \alpha \cdot \mathbf{n}[i]. \tag{15.2}$$

- **GW3**: Positive class time series from GW1; negative class time series sampled from gravitational noise near events GW170729, GW170809, GW170817, GW170608, GW170814, GW170818, and GW170823, ensuring no traces of the GW event are included in the noise data.
- **GW4**: For scalability experiments, this dataset replicates negative class time series of 1 s. length (4096 feature values) from GW1, resulting in up to 50,000 time series, simulating the same data distribution as the original series.

An overview of the datasets is presented in Table 15.1. Gravitational waves events considered in the GW1, GW2 and GW3 datasets, and their Signal-to-Noise Ratio (SNR) observed at the interferometers are GW150914 with a SNR of 24; LVT151012 with a SNR of 9.7; GW151226 with a SNR 13; and GW170104 with a SNR of 13 [19]. Depending on the learning approach, time series are used for one-class learning and anomaly detection (AE approach) or for feature extraction and supervised classification (AE-FE approach). The datasets (GW1, GW2, GW3) are ordered by increasing recognition difficulty.

15.4.2 Experimental Setup

We experimented with different configurations of auto-encoder architectures, featuring 1 or 2 hidden layers for both encoding and decoding. For a single hidden layer, we used 512 or 1024 units. For two hidden layers, the first had 512 or 1024 units and the second had 256 or 512 units. For context, considering that the time series

analyzed are collected at a 4096 Hz sample rate, 512 hidden units correspond to $\frac{1}{8}$ of the 1-second time series length. A 5-fold cross-validation scheme was used for the GW1 and GW2 datasets. For the GW3 dataset, we used a leave-one-subject-out cross-validation scheme, resulting in 4 folds, where each fold had all representations of a specific signal (GW event) in the test set. The auto-encoders were trained using the LBFGS optimizer, minimizing reconstruction error on non-anomalous training data. Training stopped after 500 iterations or when the error reduction was below 10^{-5}.

For detection, we set the standard deviation factor to $f = 1.5$, based on a sensitivity study [19] that showed optimal classification performance in terms of F-score at this value, except for the GW2 dataset with noise $N = 0.5$, where it achieved the best recall.

We evaluated features extracted by the auto-encoders using various classifiers: Logistic Regression (AE-LR), Gradient-Boosted Trees (AE-GBT), ExtraTrees (AE-ERT), Random Forest (AE-RF), XGBoost (AE-XGB), and Support Vector Machines (AE-SVM). Logistic Regression used L2 norm penalization and $maxIter = 100$. Gradient-Boosted Trees had $maxIter = 10$ and other default Apache Spark MLlib parameters. For ExtraTrees and Random Forest, we tested $numEstimators \in \{10, 100, 1000\}$. XGBoost used $numEstimators = 1000$. SVM used an RBF kernel, with regularization parameter $C \in \{1, 10, 100\}$ and kernel coefficient $gamma \in \{0.1, 0.01, 0.001, 0.0001\}$.

We compared our method with one-dimensional convolutional neural networks (Conv1D). The first architecture had two convolutional layers (40 and 20 filters, length 3), max-pooling of size 2, and a dense layer (sizes 512 or 1024). The second architecture had two convolutional layers (40 and 20 filters, length 6), max-pooling of size 2, and a three-layer MLP with dense layers (sizes 512, 256 or 1024, 512) and a softmax classification layer.

Hyperparameters for competitor methods were selected via grid search: learning rate $lr \in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$, dropout $d \in \{0.1, 0.3\}$, batch size $bs \in \{8, 16, 32\}$, and dense units $du \in \{512, 1024\}$, following heuristics from [40, 41]. Optimization was done using nested cross-validation.

Additionally, we tested Deep Filtering [42], a state-of-the-art method for gravitational wave analysis using one-dimensional CNNs, featuring 4 dilated convolution layers (filter sizes 64, 128, 256, 512) and 2 fully connected layers (sizes 128, 64).

Our methods were implemented in Scala using Apache Spark, while competitors used Python and Keras [1]. The best results, in terms of F-Score, for each method and dataset are shown in Table 15.2.

Table 15.2 Summary of experimental results in terms of F-Score obtained with the optimal configuration for each method and dataset. The best F-Score obtained for each dataset is marked in bold. In cases where more than one configuration obtained the same optimal performance, we report the simplest configuration in terms of model architecture (least number of hidden layers and neurons). The number of hidden layers and neurons for Conv1D, Conv1D (2) and Deep Filtering refer to their final dense layers [19]

Dataset	Optimal configuration	Conv1D	Conv1D (2)	Deep filtering	AE	AE-LR	AE-GBT	AE-ERT	AE-RF	AE-XGB	AE-SVM
GW1	Time series length (s)	1	1	3	1	3	3	3	3	3	3
	Hidden layers	1	2	2	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	Neurons	512	512-256	128-64	1024	512	512	512	512	512	512
	F-Score	0.9933	0.9932	0.9932	0.9934	0.9933	**1**	**1**	**1**	**1**	**1**
GW2 (N = 10%)	Time series length (s)	1	1	3	1 or 3	3	1	3	1 or 3	1	3
	Hidden layers	1	2	2	1	1	1	1	1	1	1
	Neurons	512	512-256	128-64	512	512	512	512	512	512	512
	F-Score	0.9797	0.9796	0.7205	0.9735	**0.9933**	**0.9933**	**0.9933**	**0.9933**	**0.9933**	**0.9933**
GW2 (N = 25%)	Time series length (s)	1	1	3	1	1	1	3	3	3	3
	Hidden layers	1	2	2	2	1	1	1	1	1	1
	Neurons	512	1024-512	128-64	512-256	512	1024	1024	512	1024	512
	F-Score	0.939	0.9796	0.85	0.7469	0.973	**0.9933**	0.9798	0.9821	0.9709	0.9888
GW2 (N = 50%)	Time series length (s)	1	1 or 3	3	3	3	3	3	3	3	3
	Hidden layers	1	2	2	1	1	2	1	1	1	1
	Neurons	1024	512-256	128-64	512	1024	512-256	1024	512	1024	512
	F-Score	0.847	0.9659	0.6782	0.6021	0.946	**0.9866**	0.9595	0.9597	0.9463	0.9798
GW3	Time series length	1 s	3	3	1	3	3	3	3	3	3
	Hidden layers	1	2	2	1	1	2	1	1	1	1
	Neurons	1024	512-256	128-64	512	1024	512-256	1024	1024	512	1024
	F-Score	0.5889	0.5522	0.4155	0.6388	0.7797	0.6559	0.7947	0.7877	**0.8194**	0.7934

15.4.3 Accuracy Results

Table 15.2 presents Precision, Recall, and F-score results for AE, AE-FE, two Conv1D implementations, and Deep Filtering.

In astrophysical data analysis, Recall is often prioritized over Precision, aiming to detect more real gravitational waves (GW) at the expense of more False Positives, which can be manually filtered. A model with perfect Recall would predict all positives, but this is not ideal. Since all gravitational data is stored, historical data can be reprocessed to find missed events, making non-perfect Recall acceptable. However, false positives can be costly, especially when they influence decisions like telescope positioning. Therefore, Precision is also critical, necessitating a balance between Precision and Recall. Therefore, in our study to compare the methods we utilize the F-Score, which balances both metrics.

Results indicate that both AE and AE-FE achieve high accuracy in classifying noise and real GWs. The AE method, trained on noise time series, effectively discriminates between noise and GWs using reconstruction error with automatic thresholding. Predictive accuracy is similar across different auto-encoder architectures, AE, AE-FE, and Conv1D, with AE-FE achieving the best F-Score.

The AE approach does not assume prior knowledge of GW data distribution and classifies based on reconstruction error, which is advantageous given the limited observed and validated GW time series. This unsupervised method can accurately detect new phenomena without relying on a predefined class distribution.

The results indicate that AE and AE-FE perform comparably with a 10% noise rate, while AE and Conv1D degrade. The F-Score difference between AE-FE and other methods is more pronounced in this dataset compared to GW1, reinforcing AE-FE as the best approach.

AE degradation is particularly noticeable in Recall, while Precision remains relatively high, suggesting AE correctly identifies the positive class when it makes predictions but misses many positive instances due to increased false negatives. This outcome is expected since the GW2 dataset's positive class time series are contaminated with the same noise distribution learned by the auto-encoder. The strong perturbation makes real GW signals and noise distributions more similar, reducing reconstruction error differences among classes. Consequently, the AE approach's reconstruction error threshold becomes inaccurate for distinguishing classes. In contrast, AE-FE maintains optimal performance even with increased noise, accurately mapping feature variations to classes using positively labeled time series for training gradient-boosted trees.

For the AE approach, performance worsens with 3-second time series, especially at high noise rates. Here, noise contributes significantly to reconstruction error, making the signal's contribution negligible, thus complicating classification. Conversely, Deep Filtering's convolutional filters perform better with longer time series, though overall classification remains inferior to AE and AE-FE. AE performance also degrades with increased network size, suggesting larger networks propagate

noise more, reducing classification accuracy, whereas smaller networks effectively reduce noise and produce more accurate classifications.

With the GW3 dataset, all methods show performance degradation due to increased task complexity, but AE-FE remains the best, with satisfactory F-Score.

Table 15.2 highlights AE-FE's favorable performance. However, AE-FE assumes known positive class distribution, requiring data from both classes during training, similar to Conv1D and Deep Filtering. This assumption may be unrealistic in GW analysis, given limited knowledge of expected phenomena. Therefore, AE could be a viable alternative, as the context-dependent results favoring AE-FE might vary. Additional nonparametric statistical tests in [19] show that AE-SVM has the lowest ranking.

To evaluate the execution performance of our distributed implementation, scalability experiments were conducted.

To this end, the GW4 dataset was adopted, by replicating all 1-second negative class time series (4096 feature values) from the GW1 dataset up to 50,000 time series.

Two Spark cluster configurations on a Microsoft Azure HDInsight cloud infrastructure were utilized. For the local setting, we used 2 D12 head nodes and one D12 worker node. A D12 instance has four cores and 28 GB of RAM, a D13 instance has eight cores and 56 GB of RAM, and a D14 instance has 16 cores and 112 GB of RAM. The first cluster had 2 D12 head nodes and 4 D13 worker nodes, providing an 8x scale factor compared to the local setup. The second cluster had 2 D12 head nodes and 8 D14 worker nodes, providing a 16x scale factor compared to the local setup and a 2x scale factor compared to the first cluster.

Figure 15.4 shows the results in terms of the speedup factor (right) observed with the best parameter values on the two clusters. The speedup graph does not contain the last point, related to 50.000 time series, since the local execution could not be completed successfully due to an excessive memory and CPU overhead. Figure 15.5 presents a different perspective in terms of execution time with a varying number of worker CPUs and different amounts of processed data.

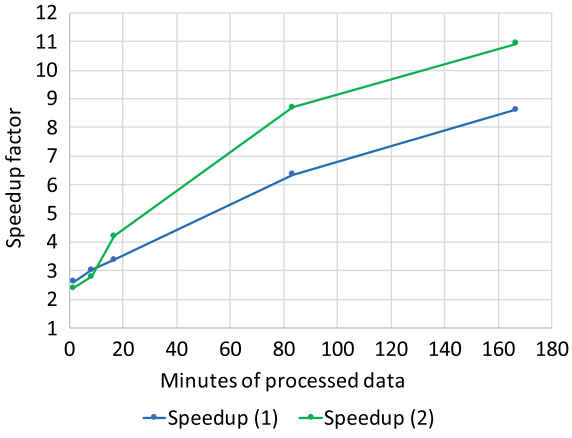


Fig. 15.4 Scalability results in terms of speedup for the AE approach, obtained with two cluster configurations. The two curves (1) and (2) refer to the different Spark cluster configurations on Azure HDInsight [19]

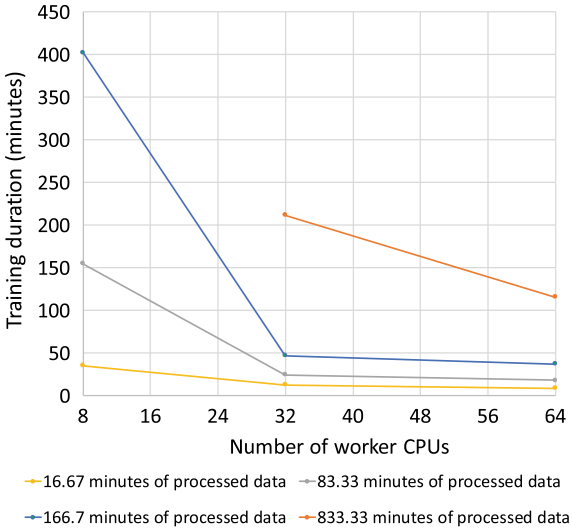


Fig. 15.5 Scalability results in terms of execution time with a varying number of CPUs: 8—Local, 32—Distributed (1), 64—Distributed (2). The missing point of the last series in the graph (833.33 min of processed data) denotes that the Local configuration with 8 worker CPUs could not complete the training process successfully due to an excessive memory and CPU overhead [19]

Overall, it is possible to observe that the discussed implementation is capable of scaling linearly as data increases. Moreover, the speedup factor obtained with the distributed execution is consistently high, and it still increases with the addition of worker cores, considering the improvement obtained by cluster configuration. This performance confirms that the discussed approach benefits from a cluster environment and scales well with large datasets.

The distributed implementation of the proposed approaches and the datasets are available to replicate the experiments at the following link: http://www.di.uniba.it/~corizzo/gw/.

15.5 Final Remarks and Future Outlook

In this chapter, we discussed unsupervised and supervised approaches for GW detection from strain data time series, eliminating the need for manual pre-processing and noise removal.

Our results showed that both AE and AE-FE outperform one-dimensional convolutional neural networks and the Deep Filtering method. AE is ideal when no prior knowledge of GW data distribution is available, which is realistic given the scarcity of labeled time series of verified phenomena. AE models, trained on various noise types, are sensitive to deviations and can detect interesting signals in noisy data, potentially indicating real phenomena.

When knowledge of real GW phenomena exists, the AE-FE approach, using features from an auto-encoder for classification models, outperforms AE. This method excels in detecting previously observed phenomena in new time series. Experiments

with varying noise levels revealed that AE performance degrades significantly at 25% noise or higher, while AE-FE remains more robust.

However, AE-FE relies on positive time series for training a supervised classifier, and its near-perfect predictive performance may degrade if GW morphology changes significantly in new, unseen time series.

Our methods, implemented using Apache Spark, show good scalability with large-scale data, making them well-suited for high-volume GW time series analysis.

Future work could focus on glitch classification tasks with time series data and exploring alternative neural network architectures optimized for detection.

References

1. Chollet, F., et al.: Keras (2015). https://keras.io
2. Cuoco, E., Calamai, G., Fabbroni, L., Losurdo, G., Mazzoni, M., Stanga, R., Vetrano, F.: Class. Quantum Gravity **18**, 1727 (2001)
3. Cuoco, E., Losurdo, G., Calamai, G., Fabbroni, L., Mazzoni, M., Stanga, R., Guidi, G., Vetrano, F.: Phys. Rev. D **64**, 122002 (2001)
4. Zevin, M., Coughlin, S., Bahaadini, S., Besler, E., Rohani, N., Allen, S., Cabero, M., Crowston, K., Katsaggelos, A.K., Larson, S.L., et al.: Class. Quantum Gravity **34**, 064003 (2017)
5. Biwer, C., Capano, C.D., De, S., Cabero, M., Brown, D.A., Nitz, A.H., Raymond, V.: Publ. Astron. Soc. Pac. **131**, 024503 (2019)
6. Bahaadini, S., Noroozi, V., Rohani, N., Coughlin, S., Zevin, M., Smith, J., Kalogera, V., Katsaggelos, A.: Inf. Sci. **444**, 172 (2018)
7. Razzano, M., Cuoco, E.: Class. Quantum Gravity **35**, 095016 (2018)
8. Gabbard, H., Williams, M., Hayes, F., Messenger, C.: Phys. Rev. Lett. **120**, 141103 (2018)
9. Xing, Z., Pei, J., Keogh, E.: SIGKDD Explor. Newsl. **12**, 40 (2010). ISSN issn1931-0145
10. Japkowicz, N.: Machine learning **42**, 97 (2001)
11. Chandola, V., Banerjee, A., Kumar, V.: ACM computing surveys (CSUR) **41**, 15 (2009)
12. Najafabadi, M., Villanustre, F., Khoshgoftaar, T., Seliya, N., Wald, R., Muharemagic, E.: J. Big Data **2**, 1 (2015)
13. Zhou, C., Paffenroth, R.: In:Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 665–674. organizationACM (2017)
14. Corizzo, R., Ceci, M., Japkowicz, N.: Big Data Res. (2019)
15. Bahaadini, S., Rohani, N., Coughlin, S., Zevin, M., Kalogera, V., Katsaggelos, A.K.: In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2931–2935. organizationIEEE (2017)
16. Bahaadini, S., Rohani, N., Katsaggelos, A.K., Noroozi, V., Coughlin, S., Zevin, M.: In: 2018 25th IEEE International Conference on Image Processing (ICIP), pp. 748–752. organization-IEEE (2018b)
17. Gebhard, T., Kilbertus, N., Parascandolo, G., Harry, I., Schölkopf, B.: In: Workshop on Deep Learning for Physical Sciences (DLPS) at the 31st Conference on Neural Information Processing Systems (NIPS), pp. 1–6 (2017)
18. Bengio, Y.Y., et al.: Foundations and trends® in machine learning **2**, 1 (2009)
19. Corizzo, R., Ceci, M., Zdravevski, E., Japkowicz, N.: Expert Syst. Appl. **151**, 113378 (2020)
20. Pukelsheim, F.: Am. Stat. **48**, 88 (1994)
21. An, J., Cho, S.: Special lecture on IE **2**, 1 (2015)
22. Khan, S.S., Taati, B.: Expert Syst. Appl. **87**, 280 (2017)
23. Bontemps, L., McDermott, J., Le-Khac, N.-A., et al.: In: International Conference on Future Data and Security Engineering, pp. 141–152. organizationSpringer (2016)

24. Clark, J.I.W., Liu, Z., Japkowicz, N.: In: 2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA), pp. 41–49 (2018)
25. Gehring, J., Miao, Y., Metze, F., Waibel, A.: In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3377–3381. organizationIEEE (2013)
26. Bertsekas, D.P.: IEEE/CAA J. Autom. Sin. **6**, 1 (2019)
27. Hinton, G., Salakhutdinov, R.: Science **313**, 504 (2006)
28. Japkowicz, N., Hanson, S.J., Gluck, M.A.: Neural Comput. **12**, 531 (2000)
29. Li, L., Lin, Y., Zheng, N., Wang, F.-Y.: IEEE/CAA J. Autom. Sin. **4**, 389 (2017)
30. Ganjisaffar, Y., Caruana, R., Lopes, C.V.: In: Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval, pp. 85–94. organizationACM (2011)
31. Chen, T., Guestrin, C.: In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794. organizationACM (2016)
32. Zieba, M., Tomczak, S.K., Tomczak, J.M.: Expert Syst. Appl. **58**, 93 (2016)
33. Ceci, M., Corizzo, R., Fumarola, F., Malerba, D., Rashkovska, A.: IEEE Trans. Ind. Inform. **13**, 956 (2017)
34. Acernese, F., Adams, T., Agathos, M., Agatsuma, K., Allocca, A., Astone, P., Ballardin, G., Barone, F., Barsuglia, M., Basti, A., et al.: J. Phys.: Conf. Ser. **610-1**, 012014 (2015) (organizationIOP Publishing
35. Aasi, J., Abbott, B., Abbott, R., Abbott, T., Abernathy, M., Ackley, K., Adams, C., Adams, T., Addesso, P., Adhikari, R., et al.: Class. Quantum Gravity **32**, 074001 (2015)
36. Shen, H., George, D., Huerta, E., Zhao Z.: (2017). arXiv:1711.09919
37. George, D., Shen, H., Huerta, E.: (2017). arXiv:1706.07446
38. Nielsen, A.B., Nitz, A.H., Capano, C.D., Brown, D.A.: J. Cosmol. Astropart. Phys. **2019**, 019 (2019)
39. Usman, S.A., Nitz, A.H., Harry, I.W., Biwer, C.M., Brown, D.A., Cabero, M., Capano, C.D., Dal Canton, T., Dent, T., Fairhurst, S., et al.: Class. Quantum Gravity **33**, 215004 (2016)
40. Bengio, Y.: In: Neural networks: Tricks of the Trade, pp. 437–478. publisherSpringer (2012)
41. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: J. Mach. Learn. Res. **15**, 1929 (2014)
42. George, D., Huerta, E.: Phys. Lett. B **778**, 64 (2018)

Chapter 16
Using Convolutional Neural Networks to Search Gravitational Wave Events in LIGO-Virgo-KAGRA Data

Marc Andrés-Carcasona, Alexis Menéndez-Vázquez, Mario Martínez, and Lluïsa-Maria Mir

Abstract The detection of gravitational waves events in the data taken by the experiments LIGO, Virgo and KAGRA requires an extensive computing process to extract the signals from the noise. The traditional technique to perform this task is matched filtering. Although powerful, it is computationally very expensive and new techniques are being studied. In this chapter, the use of convolutional neural networks to detect signals of compact binary coalescences using two-dimensional spectrograms as input is explained. Different neural networks are used for different mass ranges. The results of the searches in O3 data are shown.

16.1 Introduction

Gravitational waves (GW) can be produced by a variety of violent astrophysical events, among which are the mergers of compact binary systems. Typically consisting of either black holes or neutron stars, these compact binary coalescences (CBC) generate GW strong enough to be detected by the LIGO, Virgo and KAGRA interferometers [1–3]. These systems are particularly interesting because they provide not only direct evidence of the existence of these exotic objects, but also rich information about their properties and the nature of gravity under extreme conditions. However,

M. Andrés-Carcasona (✉) · A. Menéndez-Vázquez · M. Martínez · L.-M. Mir
Institut de Física d'Altes Energies (IFAE), The Barcelona Institute of Science and Technology, Campus UAB, E-08193 Bellaterra (Barcelona), Spain
e-mail: mandres@ifae.es

A. Menéndez-Vázquez
e-mail: amenendez@ifae.es

M. Martínez
e-mail: mmp@ifae.es

L.-M. Mir
e-mail: mir@ifae.es

M. Martínez
Catalan Institution for Research and Advanced Studies (ICREA), Barcelona, Spain

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_16

these experiments are subject to noise and the extraction of the GW signals requires a dedicated set of techniques that pose a computational challenge [4].

Traditionally, to detect these signals buried in detector noise, the so-called matched filtering technique has been used. This applies the optimal filter between the raw strain measured and a signal waveform template. Although very effective, this procedure requires a lot of computing resources, as an extensive set of signal templates (called template bank) has to be used to cover a wide set of possible parameters of the signal.

In response to these challenges, new techniques are being explored. Among them, the ones based in Machine Learning are gaining a lot of popularity [5–11] (see Ref. [12] for a comprehensive review). While various approaches are being studied, this chapter introduces in particular the application of Convolutional Neural Networks (CNN) as a method to detect CBC signals. The data preparation stages, the architecture of the CNNs and the learning procedure are explained and the results for real data from the third observing run (O3) are shown.

The chapter is organized as follows. Section 16.2 introduces the data preparation steps that are followed to generate the training images for the neural networks, and in Sect. 16.3 the architecture and the training procedure used is explained. In Sect. 16.4 the results of an injection campaign are presented together with an estimation of the sensitivity of this method. Finally, in Sect. 16.5 the results of a search in O3 data are displayed, and Sect. 16.6 is devoted to conclusions.

16.2 Data Preparation

The data taken by LIGO, Virgo and KAGRA are one-dimensional time series, typically sampled at 16 kHz. To reduce the computing power and ensure optimal performance of the CNNs, these data are usually pre-processed.

For the detection algorithm presented here, the first step is to down-sample the data to 4096 Hz. Afterwards, the time series is cut to contain only 5 s of data and whitened to remove the noise bias and to enhance the detection capabilities [9–11]. Following this step, the Q-transform is applied to obtain a spectrogram [13, 14]. This consists of a two-dimensional image in frequency and time that provides a visual representation of the signals. The Q-transform is particularly successful at revealing the signature chirp patterns of compact binary coalescence signals, which appear as bright, sweeping arcs in the spectrogram. By converting the original time series into a visual format, CNNs specialized in image and pattern recognition can be employed. In our case, 400 bins in time and 100 bins in frequency are used as they are sufficient to capture the signal in the spectrogram [9–11].

To train the neural network, a set of background-only and background plus signal images are used. For the signal, the PyCBC package [15–17] is used to produce the waveforms using the approximants *IMRPhenomPv2* [18] or *IMRPhenomD* [19, 20]. Then, these signals are injected in real noise data from the interferometers, after taking into account the orientations of the detectors, the antenna factors and the different times of arrival.

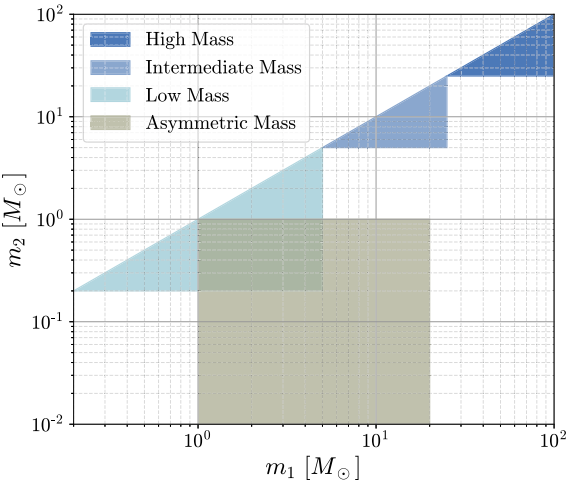


Fig. 16.1 Various regions of the masses covered for the training of each neural network

Table 16.1 Parameter space of the training set. All the distributions are assumed uniform in these ranges. We additionally assume that $m_1 \geq m_2$

Parameter	Symbol	HM	IM	LM	AM	Units
Mass 1	m_1	[25, 100]	[5, 25]	[0.2, 5]	[1, 20]	[$M_\odot$]
Mass 2	m_2	[25, 100]	[5, 25]	[0.2, 5]	[0.01, 1]	[$M_\odot$]
Distance	d	[100, 1400]	[1, 1000]	[1, 100]	[1, 100]	[Mpc]
Right ascension	α	$[0, 2\pi]$				[rad]
Declination*	δ	$[0, \pi]$				[rad]
Polarization angle	ψ	$[0, \pi]$				[rad]
Inclination	θ_{JN}	$[0, \pi/2]$				[rad]
Time of coalescence	t_c	[0.5, 0.9]				[s]

*For the AM, the cosine of the declination is uniformly sampled between $[-1, 1]$ instead

Since the parameter space is very broad, four different neural networks are considered according to the mass regions. The low-mass region covers the mass range between 0.2 and 5 $M_\odot$. The intermediate-mass comprises the masses between 5 and 25 $M_\odot$. The high-mass range covers the events between 25 and 100 $M_\odot$. Additionally, we consider a highly-asymmetric mass configuration, with the primary mass between 1 and 20 $M_\odot$ and the secondary one between 0.01 and 1 $M_\odot$. In this way, sub-solar mass events can potentially be detected (Fig. 16.1). The approximant used for the asymmetric-mass range is the *IMRPhenomD*, with a minimum frequency of 45 Hz, as otherwise the signal generation takes too long. The rest of CNNs use the *IMRPhenomPv2* waveform with a cut at a minimum frequency of 80 Hz for the low- and intermediate-mass ranges and of 25 Hz for the high-mass.

All the parameters used to train the neural networks are summarized in Table 16.1.

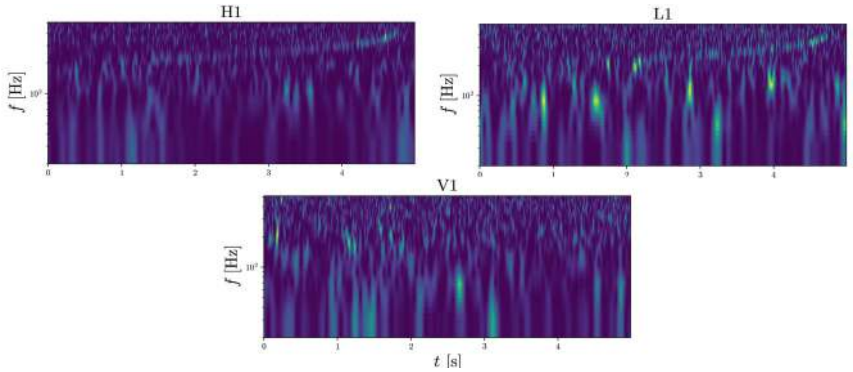


Fig. 16.2 Spectrogram of an injected signal with $m_1 = 14.4\ M_\odot, m_2 = 0.25\ M_\odot$ and $d = 21$ Mpc. Extracted from Ref. [10]

An example of a spectrogram used for the training containing a signal with parameters $m_1 = 14.4\ M_\odot$, $m_2 = 0.25\ M_\odot$ and $d = 21$ Mpc, injected into noise is shown in Fig. 16.2.

The decision to segment the data into 5 s windows is driven by the need to balance computational efficiency with the temporal resolution necessary to capture the dynamics of the GW signals. Each segment is ensured to be sufficiently long to include the inspiral, merger, and ringdown phases of the binary coalescence, yet kept short enough to ensure a manageable computational load for the neural network.

This careful tuning of data preparation parameters is essential to optimize the detection capabilities of the CNNs and to ensure that they can effectively learn from and generalize across the different noise and types of signals.

16.3 Neural Network Definition and Training

The architecture employed for the detection of these CBC events is a ResNet50 model, a residual deep convolutional neural network known for its efficacy in large-scale image recognition [21, 22]. The choice of ResNet50 for the architecture of the CNNs in this study is motivated by its proven success in various image recognition tasks across multiple fields. ResNet50's ability to mitigate the vanishing gradient problem through the use of skip connections allows it to learn effectively even from very deep networks, which is a crucial feature when dealing with complex and noisy data such as gravitational wave spectrograms.

The typical ResNet50 architecture includes a sequence of convolutional layers, each followed by a batch normalization and pooling layers and a densely connected output layer with a sigmoid activation function to perform the binary classification. Using the Adam optimizer [23], known for its efficiency in handling sparse gradients, and the binary cross-entropy for the loss function, the network iteratively adjusts its

weights across multiple epochs to minimize prediction errors. A total of the order of 10^5 images are used for the training. Some thousands of images are kept for the testing and validation.

Different CNNs are trained for each mass range, using information from the various interferometers. They cover the possible combinations of detectors: H1-L1, H1-V1, L1-V1 and H1-L1-V1. In this case, the information coming from a single interferometer is not considered, as it has been proven to perform considerably worse than for the double and triple combinations [10, 11].

Each of the sixteen neural networks is trained for 12 epochs and the one having a smaller validation loss is chosen. The model is implemented using Keras and TensorFlow's backend for GPUs [24]. It takes between two and three hours for each CNN to train.

To determine whether an image contains an event, the false alarm rate (FAR) is employed. The FAR can be computed as $\mathrm{FAR}(\eta) = N(\eta)/T$, where $\eta \in [0, 1]$ denotes the CNN output, $N(\eta)$ the number of background images with a CNN output greater than or equal to η and T the period of time analyzed. By using the time sliding technique [25], many images containing only noise can be analyzed reaching lower values of FAR. In this case, about 10^9 images were used, allowing to assign FAR values as low as $1/153$ years^{-1}.

As explained in Refs. [10, 11], the outputs of the four CNNs are combined to construct a single detection statistic by taking the mean. This is done to reduce the number of expected false positives, as without this step, the FAR when $\eta = 1$ is still higher than 1 year^{-1}, which is a reasonable number to claim a detection confidently enough. By requiring the various CNNs to be sure that the signal is present in the data, the background is less prone to confuse it, and a higher significance to the signals assigned.

As an example of the training evolution, in Fig. 16.3 the accuracy, loss and validation accuracy for the asymmetric mass (AM) trained with H1L1V1 data are displayed. The behaviour described is the one expected from a proper training; no under- or overfitting is present.

16.4 Injection Test

The injection test is a crucial part of validating the effectiveness of the trained CNNs in real-world scenarios. By simulating signals with the same parameters as those described in Table 16.1, but sampled uniformly in co-moving volume, and injecting them into real detector noise, we can assess the CNN's detection capabilities.

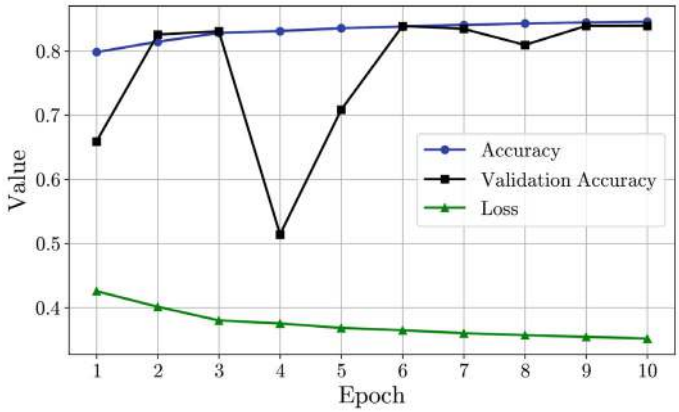


Fig. 16.3 Evolution of training statistics for the asymmetric-mass CNN with H1L1V1 information. Extracted from Ref. [10]

For each signal, the corresponding signal-to-noise ratio (ρ) can be estimated as

$$\rho^2 = 4 \int_{f_{\min}}^{f_{\max}} \frac{|\tilde{h}(f)|^2}{S_n(f)} \mathrm{d}f \,, \tag{16.1}$$

where $\tilde{h}(f)$ denotes the strain of the signal in the frequency domain and $S_n(f)$ the power spectral density of the noise. Then, the network signal-to-noise ratio can be obtained as

$$\rho_{\mathrm{net}}^2 = \sum_i \rho_i^2 \,, \tag{16.2}$$

where the index i runs over the different interferometers.

With this set of injections and setting the minimum FAR for considering a detection to 1 year^{-1}, the network signal-to-noise ratio for which a 50, 80 and 99% of signals are recovered can be computed. This is displayed in Table 16.2.

Table 16.2 Network signal-to-noise ratio for which a 50%, 80% and 99% of signals are recovered

CNN	$\rho_{\mathrm{net}(50\%)}$	$\rho_{\mathrm{net}(80\%)}$	$\rho_{\mathrm{net}(99\%)}$
High mass	12	15	28
Intermediate mass	14	16	28
Low mass	21	25	41
Asymmetric mass	16	22	41

These results show that a 99% of events with a signal-to-noise ratio higher or equal to 28 (for the high- and intermediate-mass) or 41 (for the low- and asymmetric-mass) can be recovered. The efficiency drops for the systems with smaller masses due to the longer, fainter and higher-frequency signals that they produce.

16.5 Results

The CNNs can be used to perform a search over O3 data. This is done by using only the data where the three interferometers were online, making a total of about 155 days of data, separated into images of 5 s of duration with an overlap of 2.5 s. This is, two consecutive images have an overlap of 50% to ensure that no event is missed. For each image the output of the four CNNs is computed and then combined. If this is smaller than or equal to a FAR of 1 year^{-1}, the image is considered to contain an event. Similarly, the inverse FAR (IFAR), in units of year, can be computed and compared to the expected distribution of noise events, which is assumed to follow a Poisson distribution. This is shown in Fig. 16.4.

The only detections reported are in the high-mass range. The rest of the results are compatible with a Poisson background and, therefore, there is no sufficient statistical significance to claim any other detection. The events detected can be compared to those reported in the latest catalog: GWTC-3. There are only 50 events in the catalog detected during O3 with the three interferometers online. Only 31 of those have masses that fall within the mass range of the trained CNNs and 16 of these are detected (with a FAR of 1 per year). This implies a detection efficiency of about a

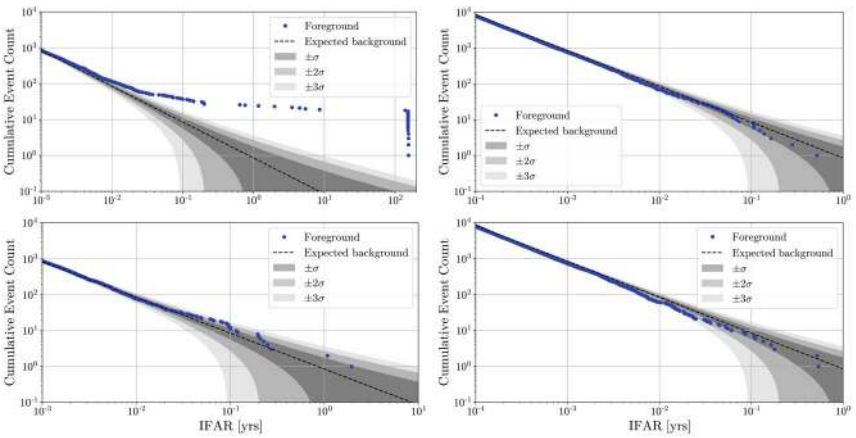


Fig. 16.4 Cumulative event count as a function of the inverse false alarm rate for the high-mass (top-left), low-mass (top-right), intermediate-mass (bottom-left) and asymmetric-mass (bottom-right) ranges. Extracted from Refs. [10, 11]

50%. Setting the FAR criterion to 1 per week, the number of events detected rises to 22, implying an efficiency of 70%. This comes at the expense of having more false positives.

In the case of the asymmetric-mass range CNN, the results are compatible with more sensitive searches that specifically target sub-solar mass events with matched filtering, as no signal has yet been detected [26, 27]. Despite this, the sensitivity of the search with the CNN is lower and the constraints to the population of such objects less tight [10].

In total, around 2k CPU hours have been used for each mass range running on Intel®Xeon®CPU E5-2680 v4 @ 2.40GHz. This reduces drastically the computing cost that is usually needed for match filtering analyses.

16.6 Conclusions

The results have shown that it is possible to detect CBC events using CNNs of a ResNet50 kind on spectrograms. This approach has proven to be less sensitive than traditional matched filtering, but faster. The O3 scan that has been carried out demonstrates that a third of the events of the catalog can be recovered when a cut of a FAR ≤ 1 years^{-1} is applied to select the candidates. This number increases to a 50% if the parameter space for which they have been trained is accounted for. Overall the conclusions are that CNNs are much faster than the traditional methods but still less sensitive.

Acknowledgements This research has made use of data or software obtained from the Gravitational Wave Open Science Center (gwosc.org), a service of the LIGO Scientific Collaboration, the Virgo Collaboration, and KAGRA. This material is based upon work supported by NSF's LIGO Laboratory which is a major facility fully funded by the National Science Foundation, as well as the Science and Technology Facilities Council (STFC) of the United Kingdom, the Max-Planck-Society (MPS), and the State of Niedersachsen/Germany for support of the construction of Advanced LIGO and construction and operation of the GEO600 detector. Additional support for Advanced LIGO was provided by the Australian Research Council. Virgo is funded, through the European Gravitational Observatory (EGO), by the French Centre National de Recherche Scientifique (CNRS), the Italian Istituto Nazionale di Fisica Nucleare (INFN) and the Dutch Nikhef, with contributions by institutions from Belgium, Germany, Greece, Hungary, Ireland, Japan, Monaco, Poland, Portugal, Spain. KAGRA is supported by Ministry of Education, Culture, Sports, Science and Technology (MEXT), Japan Society for the Promotion of Science (JSPS) in Japan; National Research Foundation (NRF) and Ministry of Science and ICT (MSIT) in Korea; Academia Sinica (AS) and National Science and Technology Council (NSTC) in Taiwan. The authors thankfully acknowledge the computational resources provided by PIC and MinoTauro, and the technical support provided by Barcelona Supercomputing Center (RES-FI-2021- 3-0020). This work is partially supported by the Spanish MCIN/AEI/ 10.13039/501100011033 under the grants SEV-2016-0588, PGC2018-101858-B-I00, and PID2020-113701GB-I00 some of which include ERDF funds from the European Union. IFAE is partially funded by the CERCA program of the Generalitat de Catalunya. M. A-C. is supported by the 2022 FI-00335 grant. This work was carried out within the framework of the EU COST action No. CA17137.

References

1. Abbott, R., et al., (LIGO Scientiffc Collaboration, Virgo Collaboration): GWTC-1: a gravitational-wave transient catalog of compact binary mergers observed by LIGO and virgo during the first and second observing runs. Phys. Rev. X **9**, 031040 (2019)
2. Abbott, R., et al., (LIGO Scientiffc Collaboration, Virgo Collaboration): GWTC-2: compact binary coalescences observed by LIGO and virgo during the first half of the third observing run. Phys. Rev. X **11**, 021053 (2019)
3. R. Abbott et al. (LIGO Scientiffc Collaboration, Virgo Collaboration): GWTC-3: compact binary coalescences observed by LIGO and virgo during the second part of the third observing run (2021). arXiv:2111.03606
4. Abbott, B.P., et al., (LIGO Scientific Collaboration and Virgo Collaboration): A guide to LIGO-Virgo detector noise and extraction of transient gravitational-wave signals. Class. Quantum Gravity **37**, 055002 (2020)
5. Gabbard, H., Williams, M., Hayes, F., Messenger, C.: Matching matched filtering with deep networks for gravitational-wave astronomy. Phys. Rev. Lett. **120**, 141103 (2018)
6. George, D., Huerta, E.A.: Deep neural networks to enable real-time multimessenger astrophysics. Phys. Rev. D **97**(4), 044039 (2018)
7. Gebhard, T.D., Kilbertus, N., Harry, I., Schölkopf, B.: Convolutional neural networks: A magic bullet for gravitational-wave detection? Phys. Rev. D **100**(6) (2019)
8. George, D., Huerta, E.A.: Deep learning for real-time gravitational wave detection and parameter estimation: Results with advanced ligo data. Phys. Lett. B **778**, 64–70 (2018)
9. Menéndez-Vázquez, A., Kolstein, M., Martínez, M., Mir, L.M.: Searches for compact binary coalescence events using neural networks in the ligo/virgo second observation period. Phys. Rev. D **103**(6), 062004 (2021)
10. Andrés-Carcasona, M., Menéndez-Vázquez, A., Martínez, M., Mir, L.M.: Searches for mass asymmetric compact binary coalescence events using neural networks in the ligo/virgo third observation period. Phys. Rev. D **107**, 082003 (2023)
11. Menéndez-Vázquez, A., Andrés-Carcasona, M., Martínez, M., Mir, L.M.: Searches for compact binary coalescence events using neural networks in the ligo/virgo second observation period (2024). arXiv:2401.12912
12. Cuoco, E., Powell, J., Cavaglià, M., Ackley, K., Bejger, M., Chatterjee, C., Coughlin, M., Coughlin, S., Easter, P., Essick, R., et al.: Enhancing gravitational-wave science with machine learning. Mach. Learn. **2**, 011002 (2021)
13. Brown, J.C.: Calculation of a constant Q spectral transform. J. Acoust. Soc. Am. **89**, 425 (1991)
14. Brown, J.C., Puckette, M.S.: An efficient algorithm for the calculation of a constant Q transform. J. Acoust. Soc. Am. **92**, 2698 (1992)
15. Usman, S.A., et al.: The pycbc search for gravitational waves from compact binary coalescence. Class. Quantum Grav. **33**(21), 215004 (2016)
16. Nitz, A.H., Dent, T., Dal Canton, T., Fairhurst, S., Brown, D.A.: Detecting binary compact object mergers with gravitational waves: Understanding and improving the sensitivity of the pycbc search. Astrophys. J. **849**(2), 118 (2017)
17. Nitz, A.H., et al.: gwastro/pycbc: Pycbc release v1. 16.11. Zenodo (2020)
18. Khan, S., Chatziioannou, K., Hannam, M., Ohme, F.: Phenomenological model for the gravitational-wave signal from precessing binary black holes with two-spin effects. Phys. Rev. D **100**(2) (2019)
19. Husa, S., Khan, S., Hannam, M., Pürrer, M., Ohme, F., Forteza, X.J., Bohé, A.: Frequency-domain gravitational waves from nonprecessing black-hole binaries. I. New numerical waveforms and anatomy of the signal. Phys. Rev. D **93**, 044006 (2016)
20. Khan, S., Husa, S., Hannam, M., Ohme, F., Pürrer, M., Forteza, X.J., Bohé, A.: Frequency-domain gravitational waves from nonprecessing black-hole binaries. II. A phenomenological model for the advanced detector era. Phys. Rev. D **93**, 044007 (2016)

21. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (IEEE, 2016), pp. 770–778
22. He, K., Zhang, X., Ren, S., Sun, J.: Identity Mappings in Deep Residual Networks (2016). arXiv:1603.05027
23. Kingma, D.P., Ba, J.L.: Adam: a method for stochastic optimization. In: 3rd international conference on learning representations, ICLR 2015 - Conference Track Proceedings (2014)
24. Abadi, M., et al.: TensorFlow: a system for Large-Scale machine learning. In 12th USENIX symposium on operating systems design and implementation (OSDI 16), p. 265 (2016)
25. Abbott, R., et al., (LIGO Scientific Collaboration and Virgo Collaboration): Search for gravitational waves from galactic and extra-galactic binary neutron stars. Phys. Rev. D **72**, 082001 (2005)
26. Nitz, A.H., Wang, Y.-F.: Broad search for gravitational waves from subsolar-mass binaries through LIGO and Virgo's third observing run. Phys. Rev. D **106**, 023024 (2022)
27. Abbott, B., et al., (LIGO Scientific Collaboration, KAGRA Collaboration and Virgo Collaboration): Search for subsolar-mass black hole binaries in the second part of Advanced LIGO's and Advanced Virgo's third observing run. MNRAS **524**, 4 (2023)

Chapter 17
Detecting Gravitational Waves From Binary Black Hole Mergers Using Deep Convolutional Neural Networks and Quadratic Time-Frequency Distributions

Nikola Lopac, Jonatan Lerga, and Franko Hržić

Abstract Detecting gravitational waves (GWs) in measured data is a challenging task that demands advanced techniques for effective analysis due to the presence of intensive noise and the non-stationary nature of these signals. Quadratic time-frequency distributions (TFDs) from Cohen's class provide valuable tools for analyzing various non-stationary signals simultaneously in the time and frequency domain. This chapter reviews a method that integrates deep convolutional neural networks (CNNs) with these quadratic TFDs to enhance the detection of GWs from binary black hole (BBH) mergers. The approach was validated on a comprehensive dataset of 100.000 signals combining actual Laser Interferometer Gravitational-Wave Observatory (LIGO) data with synthetically simulated GW injections. Twelve different two-dimensional (2D) TFD representations were calculated (resulting in 1.2 million TFDs) and used as inputs to three high-performance CNN models: ResNet-101, Xception, and EfficientNet. The proposed approach demonstrated superior detection performance, achieving high values across various classification metrics. Furthermore, it outperformed a CNN model using original time-series data, proving to be a viable solution for detecting GWs in low signal-to-noise ratio (SNR) environments.

N. Lopac
Faculty of Maritime Studies, University of Rijeka, Rijeka, Croatia

N. Lopac (✉) · J. Lerga · F. Hržić
Center for Artificial Intelligence and Cybersecurity, University of Rijeka, Rijeka, Croatia
e-mail: nikola.lopac@pfri.uniri.hr

J. Lerga
e-mail: jonatan.lerga@riteh.uniri.hr

F. Hržić
e-mail: franko.hrzic@riteh.uniri.hr

J. Lerga · F. Hržić
Faculty of Engineering, University of Rijeka, Rijeka, Croatia

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_17

17.1 Introduction

Gravitational waves (GWs) require highly sensitive measurements due to their low amplitudes, making GW observations susceptible to various instrumental and environmental noise sources, even with cutting-edge equipment [1]. Consequently, GW measurements are corrupted by non-stationary, non-white, and non-Gaussian noise, presenting a significant challenge for detecting these signals amidst high noise levels.

GW detection methods have evolved to address these challenges, with specific algorithms developed for different signal types. Binary black hole (BBH) merger signals are primarily detected using matched filtering, a technique that correlates noisy measurements with a bank of waveform templates (see, e.g., [2]). However, this method is computationally demanding and optimal only for Gaussian noise. Denoising techniques, which do not require astrophysical signal information, have shown promise in noise reduction but must be coupled with other detection algorithms [3–5].

Machine learning (ML) has gained traction in GW astronomy, with applications in data denoising, parameter estimation, detector glitch classification, and GW detection [6]. While some studies have focused on detecting various types of GW signals using time-series data [7] or spectrograms [8], the primary focus has been on BBH signals due to their successful observation [9–11] by the Laser Interferometer Gravitational-Wave Observatory (LIGO) [12] and Virgo [13] detectors.

ML-based BBH signal detection has mainly utilized time-series data classification [14]. A significant advancement in this area was the introduction of one-dimensional (1D) deep convolutional neural networks (CNNs) for detecting BBH signals in noise [15, 16]. Subsequent studies have further explored using CNNs for classifying time-series BBH signals [17–19].

Despite the progress in ML-based GW detection, applying deep learning (DL) to two-dimensional (2D) transformations of BBH signals remains underexplored. Additionally, there has been a notable lack of approaches leveraging Cohen's class of time-frequency distributions (TFDs) for GW detection. These TFDs represent signal energy distribution in the joint time-frequency domain, and have been utilized as a powerful tool for analysis of various non-stationary signals, such as EEG, ECG, radar, and seismic signals [20].

This chapter explores the potential of quadratic TFDs from Cohen's class to enhance DL detection of BBH GW signals, reviewing a method developed by Lopac et al. [21–23]. The method involves training deep CNNs on 2D TFD representations of GW time-series signals. The DL models utilize 12 different TFDs to transform input data and three CNN architectures for classification. These models have been experimentally validated on a dataset comprising real-life noise data from LIGO detectors and simulated BBH GW signals. Additionally, the classification results are compared to those obtained from a 1D CNN model trained on the original time-series data. The analysis demonstrates that employing Cohen's class TFDs achieves high classification accuracy and significantly improves BBH GW detection performance compared to the time-series model.

The remainder of this chapter is organized as follows. Section 17.2 outlines the proposed TFD-CNN method, detailing data preparation and preprocessing, TFD calculation, and the implementation of CNNs. Section 17.3 presents and analyzes the results obtained from the experimental validation of the method. Finally, Sect. 17.4 summarizes the key conclusions.

17.2 TFD-CNN Method for Detecting BBH GWs

The method for detecting BBH GWs in intensive noise, proposed and developed in prior studies by Lopac et al. [21–23], combines quadratic TFDs from Cohen's class with 2D deep CNN architectures. This approach involves data preparation and preprocessing, TFD calculation, and the implementation and training of DL models used as classifiers, as illustrated in Fig. 17.1.

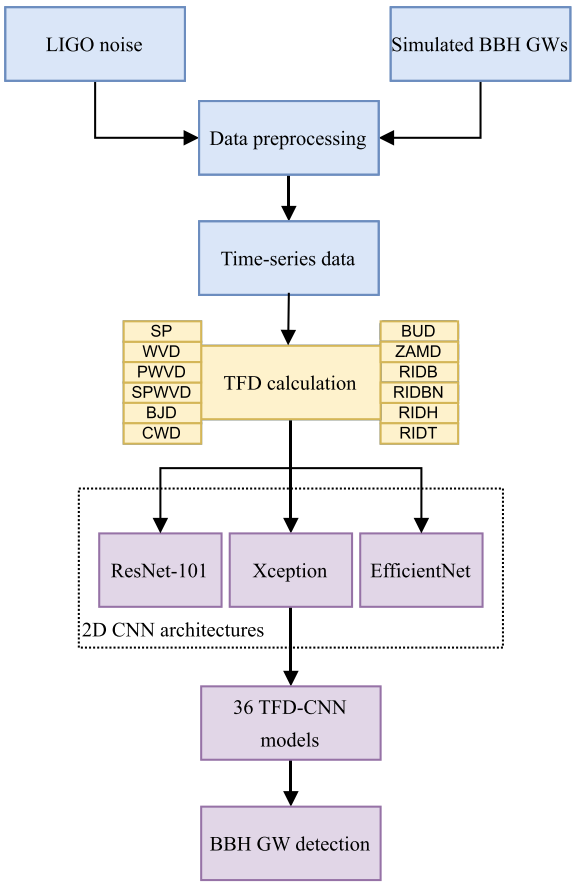


Fig. 17.1 Workflow of the TFD-CNN-based BBH GW detection method

17.2.1 Data Preparation and Preprocessing

The process of generating a comprehensive dataset includes acquiring real-life noise data from the LIGO detector, generating synthetic BBH GW signals, and preprocessing the data [19].

Real-life noise data were retrieved from the Gravitational Wave Open Science Center (GWOSC) repositories [24], specifically from LIGO's second observing run (O2) [25]. Data selection criteria included the operational status of LIGO detectors, meeting minimum quality requirements for compact binary coalescence (CBC) searches, and excluding segments with confirmed GW events or hardware injections. The data were downsampled from 4096 to 2048 Hz to reduce computational costs while maintaining sufficient frequency resolution.

Extensive simulations of BBH merger waveforms were performed using the LIGO Algorithm Library (LALSuite) [26] and PyCBC software packages [27]. The simulations employed the effective-one-body (EOB) waveform model SEOBNRv4 [28], which models spinning, non-precessing BBH mergers, with parameters randomly sampled from defined distributions. These synthetic waveforms, after applying a one-sided Tukey window, were projected onto the LIGO detectors' antenna patterns to produce noise-free GW signals. The synthetic GW signals were then injected into the background noise at specified network optimal matched-filter signal-to-noise ratios (NOMF-SNRs), ranging from 8 to 30.

The generated data underwent a whitening procedure based on a local estimate of the detector noise amplitude spectral density (ASD) calculated via the Welch method [29]. The data were then high-pass filtered at 20 Hz to remove low-frequency artifacts and cropped to 0.5-second segments, capturing the characteristic BBH chirp waveforms while reducing computational load. The final dataset consisted of 100,000 time-series examples, each 1024 samples long, with half containing injected BBH GW signals embedded in noise and half containing only noise.

17.2.2 TFD Calculation

The time-series data obtained in the previous stage were transformed into the time-frequency (t, f) domain using 12 TFDs from Cohen's class. TFDs are valuable tools for analyzing non-stationary signals, which often consist of multiple components and are affected by noise [30, 31]. These 2D signal representations enable simultaneous examination of both time and frequency domains [20, 32]. Below, the 12 TFDs utilized in the proposed approach are briefly described.

The spectrogram (SP) of the signal is obtained as the squared modulus of the short-time Fourier transform (STFT). The STFT is a linear TFD that performs the Fourier transform on segments of the signal $s(t)$ within a sliding time window $h(t)$ [33]:

$$SP(t, f) = \left| \int_{-\infty}^{\infty} s(\tau)\, h(\tau - t)\, e^{-j2\pi f \tau} d\tau \right|^2 . \tag{17.1}$$

While the SP provides a signal representation with low interference terms, it faces limitations in achieving high resolution in both time and frequency domains simultaneously. This challenge arises from the use of a fixed-size window: shorter windows offer better time resolution but poorer frequency resolution, while longer windows improve frequency resolution at the expense of time resolution.

Quadratic TFDs from Cohen's class are developed to address these limitations. These TFDs utilize the concept of the analytic signal, which is a complex signal obtained by applying the Hilbert transform to eliminate the negative frequency components from the real-valued signal to reduce cross-terms in quadratic distributions [20].

The Wigner–Ville distribution (WVD) is a fundamental TFD within Cohen's class. It is defined as the Fourier transform of the instantaneous autocorrelation function of the signal [34]:

$$WVD(t, f) = \int_{-\infty}^{\infty} s\left(t + \frac{\tau}{2}\right) s^*\left(t - \frac{\tau}{2}\right) e^{-j2\pi f \tau} d\tau . \tag{17.2}$$

The WVD offers high time-frequency resolution of the true signal components (auto-terms) but suffers from interference terms (cross-terms) when dealing with multi-component signals due to its quadratic nature. These interference terms can complicate the visual interpretation of the TFD.

To mitigate these unwanted terms, the WVD can be smoothed using appropriate kernels, resulting in TFDs known as reduced-interference distributions (RIDs) [20]. These kernels are designed in the 2D Doppler-lag (ν, τ) domain, also known as the ambiguity domain [20]. The Doppler (ν) variable, obtained by the Fourier transform of the time (t) variable, represents a frequency shift, whereas the lag (τ) variable represents a time shift [20]. Specifically, the highly oscillatory cross-terms are positioned away from the origin in the Doppler-lag domain, allowing them to be eliminated by applying a low-pass filter [20]. Applying the low-pass filter involves balancing the trade-off between the time-frequency resolution of the auto-terms and the extent of cross-term attenuation [20].

The pseudo Wigner–Ville distribution (PWVD) modifies the WVD by applying a time windowing function $h(t)$ to the autocorrelation function of the signal [35]. This time windowing acts as frequency smoothing, which attenuates the cross-terms oscillating in the frequency direction. However, this smoothing also decreases the frequency resolution of the signal's auto-terms. The PWVD is defined as [35]:

$$PWVD(t, f) = \int_{-\infty}^{\infty} h(\tau)\, s\left(t + \frac{\tau}{2}\right) s^*\left(t - \frac{\tau}{2}\right) e^{-j2\pi f \tau} d\tau . \tag{17.3}$$

The PWVD faces the challenge of cross-terms oscillating in the time direction, which are not attenuated. This limitation is addressed by the smoothed pseudo

Wigner–Ville distribution (SPWVD), which applies an additional smoothing window $g(t)$ in the time direction [36]. The SPWVD enables independent control of smoothing in both time and frequency domains by selecting appropriate lengths for the windows $h(t)$ and $g(t)$. This flexibility allows for better management of interference terms, although there is a trade-off between the level of interferences and time-frequency resolution. The SPWVD is defined as [36]:

$$SPWVD(t, f) = \int_{-\infty}^{\infty} h(\tau) \int_{-\infty}^{\infty} g(u-t)\, s\left(u+\frac{\tau}{2}\right) s^*\left(u-\frac{\tau}{2}\right) du\, e^{-j2\pi f\tau} d\tau\ . \tag{17.4}$$

The Choi–Williams distribution (CWD) employs an exponential kernel of width σ to suppress cross-terms while balancing resolution [37]. The parameter σ allows control over this trade-off: smaller values reduce cross-terms, while larger values enhance the resolution of auto-terms. Despite its effectiveness, the CWD does not offer independent control over time and frequency smoothing. The CWD is calculated as follows [33]:

$$\begin{aligned} CWD(t, f) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{\sqrt{\sigma}}{2\sqrt{\pi}\,|\tau|}\, e^{-\frac{\sigma u^2}{16\tau^2}}\, s\left(t+u+\frac{\tau}{2}\right) \\ \cdot s^*\left(t+u-\frac{\tau}{2}\right) du\, e^{-j2\pi f\tau} d\tau\ . \end{aligned} \tag{17.5}$$

The Butterworth distribution (BUD) enhances the CWD by offering improved preservation of auto-term resolution and better suppression of low-frequency cross-terms [38]. This is accomplished through a kernel that acts as a 2D low-pass filter in the ambiguity domain, with adjustable pass-band and transition region parameters [38]. The BUD is defined as follows [39]:

$$\begin{aligned} BUD(t, f) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{\sqrt{\sigma}}{2\,|\tau|}\, e^{-\frac{\sqrt{\sigma}|u|}{|\tau|}}\, s\left(t+u+\frac{\tau}{2}\right) \\ \cdot s^*\left(t+u-\frac{\tau}{2}\right) du\, e^{-j2\pi f\tau} d\tau\ . \end{aligned} \tag{17.6}$$

The Born-Jordan distribution (BJD) is an RID that preserves the time and frequency supports of a signal [40]. It utilizes a narrowband *sinc* kernel in the ambiguity domain, effectively suppressing cross-terms but at the expense of reduced auto-term resolution [41]. The BJD is defined as follows [42]:

$$BJD(t, f) = \int_{-\infty}^{\infty} \frac{1}{|\tau|} \int_{t-\frac{|\tau|}{2}}^{t+\frac{|\tau|}{2}} s\left(u+\frac{\tau}{2}\right) s^*\left(u-\frac{\tau}{2}\right) du\, e^{-j2\pi f\tau} d\tau\ . \tag{17.7}$$

The Zhao–Atlas–Marks distribution (ZAMD) employs a cone-shaped kernel to achieve a balance between time and frequency resolution while effectively reducing cross-terms. This distribution is derived by applying frequency smoothing to the BJD [43]:

$$ZAMD(t,f) = \int_{-\infty}^{\infty} h(\tau) \int_{t-\frac{|\tau|}{2}}^{t+\frac{|\tau|}{2}} s\left(u+\frac{\tau}{2}\right) s^*\left(u-\frac{\tau}{2}\right) du\, e^{-j2\pi f\tau} d\tau\ . \quad (17.8)$$

The RID with a kernel based on the first kind Bessel function of order one (RIDB) effectively suppresses cross-terms while preserving high time-frequency resolution. The RIDB achieves this balance by leveraging the low-pass filtering characteristics of the Bessel kernel [44, 45]:

$$RIDB(t,f) = \int_{-\infty}^{\infty} h(\tau) \int_{t-|\tau|}^{t+|\tau|} \frac{2g(u)}{\pi\,|\tau|} \sqrt{1-\left(\frac{u-t}{\tau}\right)^2} \cdot s\left(u+\frac{\tau}{2}\right) s^*\left(u-\frac{\tau}{2}\right) du\, e^{-j2\pi f\tau} d\tau\ . \quad (17.9)$$

The RID with a kernel based on the binomial coefficients (RIDBN) is defined as [45, 46]:

$$RIDBN(t,f) = \sum_{\tau=-\infty}^{\infty} \sum_{u=-|\tau|}^{|\tau|} \frac{1}{2^{2|\tau|+1}} \binom{2|\tau|+1}{|\tau|+u+1} \cdot s\left[t+u+\tau\right] s^*\left[t+u-\tau\right]\ e^{-j4\pi f\tau}\ . \quad (17.10)$$

The RID with a kernel based on the Hanning window (RIDH) is obtained as [42, 45]:

$$RIDH(t,f) = \int_{-\infty}^{\infty} h(\tau) \int_{-\frac{|\tau|}{2}}^{\frac{|\tau|}{2}} \frac{g(u)}{|\tau|} \left(1+\cos\left(\frac{2\pi u}{\tau}\right)\right) \cdot s\left(t+u+\frac{\tau}{2}\right) s^*\left(t+u-\frac{\tau}{2}\right) du\, e^{-j2\pi f\tau} d\tau\ . \quad (17.11)$$

Finally, the RID with a kernel based on the triangular window (RIDT) is calculated as [42, 45]:

$$RIDT(t,f) = \int_{-\infty}^{\infty} h(\tau) \int_{-\frac{|\tau|}{2}}^{\frac{|\tau|}{2}} \frac{2g(u)}{|\tau|} \left(1-\frac{2|u|}{|\tau|}\right) \cdot s\left(t+u+\frac{\tau}{2}\right) s^*\left(t+u-\frac{\tau}{2}\right) du\, e^{-j2\pi f\tau} d\tau. \quad (17.12)$$

Figures 17.2 and 17.3 illustrate the application of the described TFDs to time-series GW data, presenting 12 TFDs of a sample BBH GW signal embedded in the noise with NOMF-SNR of 20.

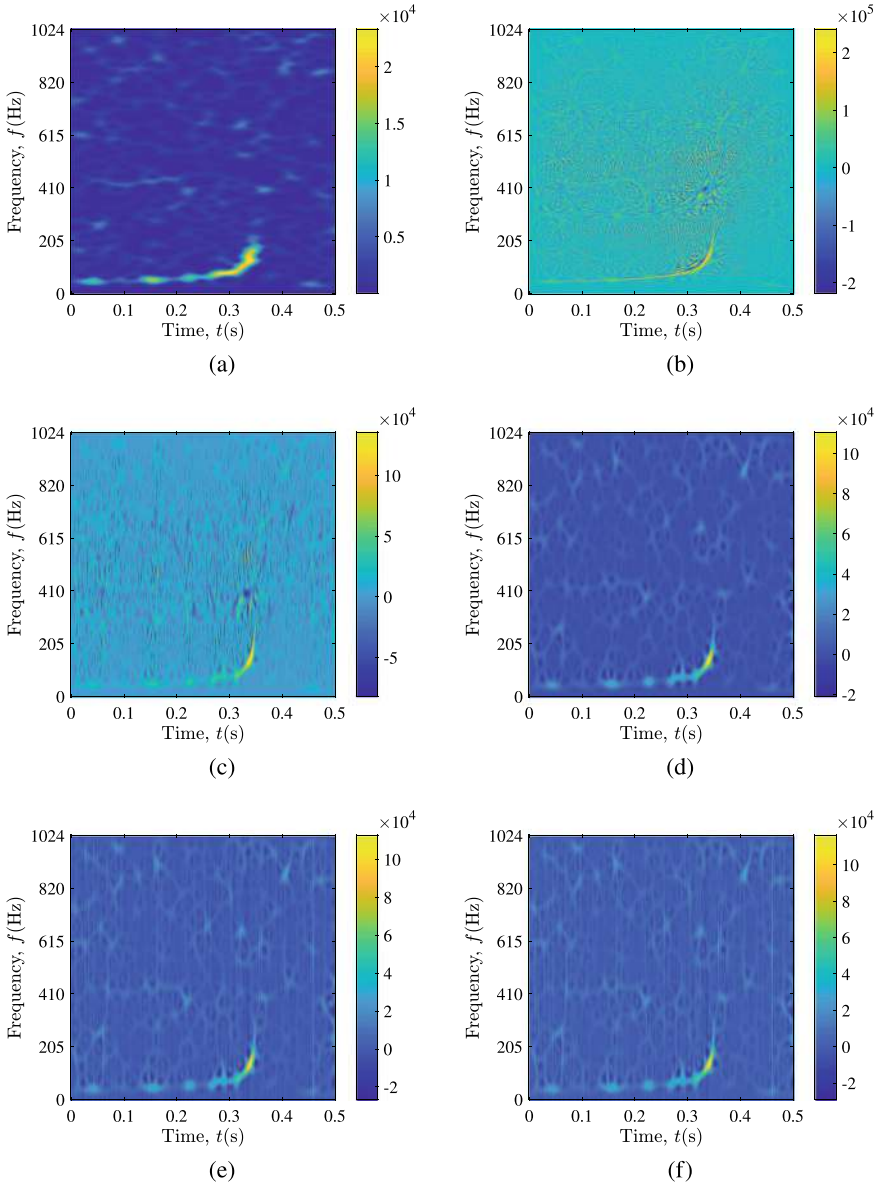


Fig. 17.2 TFDs of a sample BBH GW signal embedded in the noise (NOMF-SNR = 20): **a** SP; **b** WVD; **c** PWVD; **d** SPWVD; **e** CWD; **f** BUD

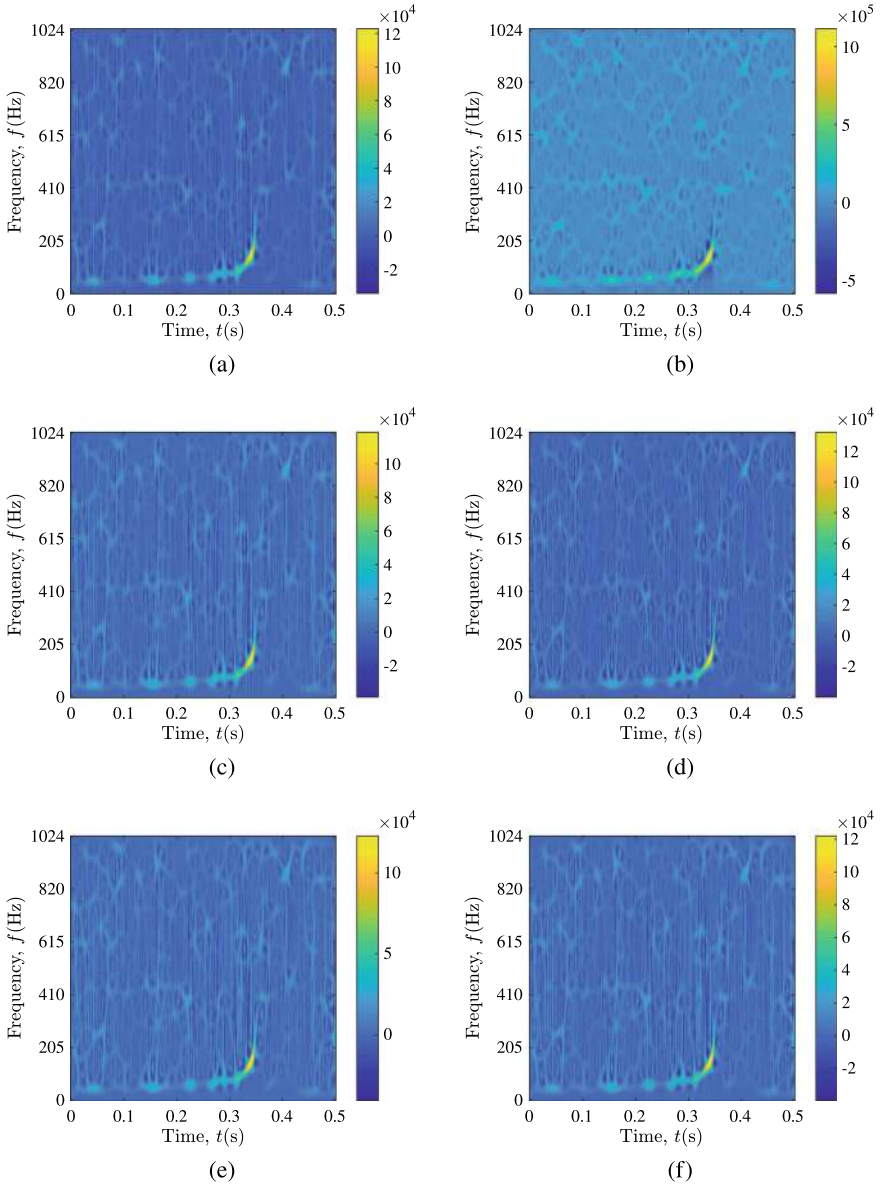


Fig. 17.3 TFDs of a sample BBH GW signal embedded in the noise (NOMF-SNR = 20): **a** BJD; **b** ZAMD; **c** RIDB; **d** RIDBN; **e** RIDH; **f** RIDT

17.2.3 TFD-CNN Models

After applying 12 different TFDs to the time-series dataset, 12 distinct datasets of 100,000 TFD images of size 256×256 were created (total of 1.2 million TFD images). These datasets were normalized and divided into training, validation, and test subsets with a 70-15-15 split. Each TFD dataset was used with three advanced CNN architectures: ResNet-101, Xception, and EfficientNet.

The ResNet-101 architecture addresses the vanishing gradient problem by incorporating skip connections, enabling the construction of deeper networks with higher accuracy [47, 48]. It begins with a 7×7 convolutional layer with 64 channels, followed by a 3×3 max-pooling layer. The network then includes four groups of bottleneck residual blocks with 256, 512, 1024, and 2048 output channels, respectively. The model ends with a global average pooling layer and a fully connected layer. In this study, the Adam optimizer with a learning rate of 1×10^{-6} and a batch size of 16 was used for training.

The Xception architecture extends the Inception module concept by utilizing depthwise separable convolutions, which improve accuracy and convergence speed while reducing computational costs [49]. The Xception network comprises 36 convolutional layers divided into 14 modules with linear residual connections, except for the first and last modules. The initial two layers perform standard convolutions, while the remaining layers use depthwise separable 3×3 convolutions followed by batch normalization. Additionally, 3×3 max-pooling layers with a stride of 2 are utilized. The training was performed using the Adam optimizer with a learning rate of 1×10^{-4} and a batch size of 32.

EfficientNet employs compound scaling to balance network width, depth, and resolution [50]. This architecture includes mobile inverted bottleneck convolution (MBConv) [51] and squeeze-and-excitation modules [52]. The EfficientNet-B2 variant was used in this study, with the RMSProp optimizer with a learning rate of 1×10^{-4} and a batch size of 32.

For all models, the final prediction layer was modified to output a single probability value using a sigmoid activation function, facilitating binary classification to distinguish between noise and GW signal examples. The binary cross-entropy loss function was employed, and all hyperparameters for model training were experimentally selected from a range of values defined in the original literature where the corresponding CNN architectures were first proposed. Figure 17.4 shows an overview of the CNN-based binary classification approach, where the output represents the probability that the input data example contains a BBH GW event.

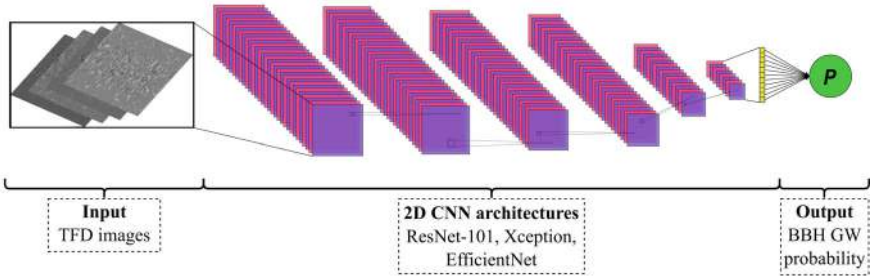


Fig. 17.4 TFD images processed by 2D CNN architectures for detecting BBH GW events

17.3 Results and Discussion

After training, the performance of each TFD-CNN model was evaluated on the test dataset. The models demonstrated exceptional classification performance across multiple evaluation metrics, as reported in studies by Lopac et al. [21–23]. The proposed method was also compared to a 1D CNN model adapted from [16] for classifying original time-series GW data. This adaptation involved resizing the input layer to match the length of the input data examples, necessitating smaller convolution kernels while keeping other parameters unchanged. The final output layer was modified to a single neuron with a sigmoid activation function to indicate the probability of a BBH GW signal. The training utilized the binary cross-entropy loss function and the Adam optimizer with an optimal learning rate of 1×10^{-5} and a batch size of 32.

The obtained classification accuracy and area under the receiver operating characteristic (ROC) curve (ROC AUC) values, as reported in [21–23], are shown in Tables 17.1 and 17.2, respectively. The TFD-CNN models achieved classification accuracy between 96.54 and 97.10%, and ROC AUC values ranged from 0.9850 to 0.9885. Compared to the 1D CNN model utilizing time-series data, the proposed approach showed substantial improvements, with gains of 3.39–3.95% in classification accuracy and 1.71–2.06% in ROC AUC.

Furthermore, precision values were reported between 97.55 and 99.51%, recall rates from 94.15 to 95.87%, F1 scores between 96.46 and 97.03%, and the area under the precision-recall curve (PR AUC) values ranging from 0.9899 to 0.9920 [21–23]. Additionally, the TFD-CNN models offered improvements of 0.35–2.31% in precision, 5.30–7.02% in recall, 3.62–4.19% in F1 score, and 1.27–1.48% in PR AUC over the time-series CNN model.

The robustness of the proposed method was further supported by additional metrics such as confusion matrices, ROC curves, and precision-recall curves, which consistently confirmed the method's high performance [21–23]. The statistical significance of the results was validated using McNemar's test, underscoring the reliability and effectiveness of the developed approach.

Table 17.1 Classification accuracy values of the evaluated TFD-CNN models

TFD	CNN		
	ResNet-101 (%)	Xception (%)	EfficientNet (%)
SP	96.91	96.95	**97.10**
WVD	96.54	**97.04**	96.82
PWVD	96.89	96.97	96.93
SPWVD	96.76	96.99	96.91
CWD	**96.95**	96.84	96.98
BUD	96.83	96.83	96.89
BJD	96.83	96.80	96.99
ZAMD	96.81	96.82	96.57
RIDB	96.85	96.77	96.83
RIDBN	96.93	96.91	96.80
RIDH	96.79	96.87	97.00
RIDT	96.80	96.86	96.85
Time-series model	93.15		

Table 17.2 ROC AUC values values of the evaluated TFD-CNN models

TFD	CNN		
	ResNet-101	Xception	EfficientNet
SP	0.9873	**0.9881**	0.9882
WVD	0.9854	0.9871	0.9857
PWVD	0.9865	0.9873	0.9869
SPWVD	0.9868	0.9880	0.9877
CWD	0.9870	0.9873	**0.9885**
BUD	0.9880	0.9867	0.9869
BJD	0.9880	0.9871	0.9882
ZAMD	0.9871	0.9876	0.9875
RIDB	**0.9881**	0.9873	0.9864
RIDBN	0.9880	0.9878	0.9875
RIDH	0.9863	0.9875	0.9880
RIDT	0.9871	0.9862	0.9850
Time-series model	0.9679		

17.4 Conclusions

This chapter reviewed a method for detecting BBH GWs in noisy measurements using deep CNNs as classifiers and quadratic TFDs as an alternative signal representation. The approach involved transforming time-series data into time-frequency representations using 12 different quadratic TFDs from Cohen's class and using these 2D representations as inputs for three advanced CNN models: ResNet-101, Xception, and EfficientNet. The proposed method was validated on a dataset combining real-life LIGO data with synthetic GW signals, achieving high detection performance with classification accuracy up to 97.10% and ROC AUC up to 0.9885. Compared to a 1D model using original time-series data, the TFD-CNN approach demonstrated significant improvements, with statistically significant gains across all metrics. These results underscore the effectiveness of using TFDs from Cohen's class to enhance DL-based detection of BBH GW events. Future research could explore the use of various TFDs for data augmentation, employ ensemble learning, and apply adaptive filtering to noisy data to further refine and enhance this approach.

Competing Interests The authors have no conflicts of interest to declare that are relevant to the content of this chapter.

Acknowledgements This work was carried out within the framework of the EU COST action CA17137. This work has been partially supported by the University of Rijeka project uniri-mladi-tehnic-23-15 and the project line ZIP UNIRI of the University of Rijeka, for the project UNIRI-ZIP-2103-4-22.

This research has made use of data, software and/or web tools obtained from the Gravitational Wave Open Science Center (https://www.gw-openscience.org/), a service of LIGO Laboratory, the LIGO Scientific Collaboration, and the Virgo Collaboration. LIGO Laboratory and Advanced LIGO are funded by the United States National Science Foundation (NSF) as well as the Science and Technology Facilities Council (STFC) of the United Kingdom, the Max-Planck-Society (MPS), and the State of Niedersachsen/Germany for support of the construction of Advanced LIGO and construction and operation of the GEO600 detector. Additional support for Advanced LIGO was provided by the Australian Research Council. Virgo is funded, through the European Gravitational Observatory (EGO), by the French Centre National de Recherche Scientifique (CNRS), the Italian Istituto Nazionale di Fisica Nucleare (INFN), and the Dutch Nikhef, with contributions by institutions from Belgium, Germany, Greece, Hungary, Ireland, Japan, Monaco, Poland, Portugal, Spain.

References

1. Abbott, B.P., et al.: Characterization of transient noise in Advanced LIGO relevant to gravitational wave signal GW150914. Class. Q. Gravity **33**(13) (2016). https://doi.org/10.1088/0264-9381/33/13/134001. Art. no. 134001
2. Nitz, A.H., Dal Canton, T., Davis, D., Reyes, S.: Rapid detection of gravitational waves from compact binary mergers with PyCBC Live. Phys. Rev. D **98** (2018). https://doi.org/10.1103/PhysRevD.98.024050. Art. no. 024050

3. Torres-Forné, A., Cuoco, E., Marquina, A., Font, J.A., Ibáñez, J.M.: Total-variation methods for gravitational-wave denoising: performance tests on Advanced LIGO data. Phys. Rev. D **98**(8) (2018). https://doi.org/10.1103/PhysRevD.98.084013. Art. no. 084013
4. Lopac, N., Lerga, J., Cuoco, E.: Gravitational-wave burst signals denoising based on the adaptive modification of the intersection of confidence intervals rule. Sensors **20**(23) (2020). https://doi.org/10.3390/s20236920. Art. no. 6920
5. Lopac, N., Lerga, J., Saulig, N., Stanković, L., Daković, M.: On optimal parameters for ICI-based adaptive filtering applied to the GWs in high noise. In: 2021 6th International Conference on Smart and Sustainable Technologies (SpliTech 2021), pp. 1–6. University of Split, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, IEEE, Bol and Split, Croatia (2021). https://doi.org/10.23919/SpliTech52315.2021.9566364
6. Cuoco, E., et al.: Enhancing gravitational-wave science with machine learning. Mach. Learn.: Sci. Technol. **2**(1) (2020). https://doi.org/10.1088/2632-2153/abb93a. Art. no. 011002
7. Krastev, P.G.: Real-time detection of gravitational waves from binary neutron stars using artificial neural networks. Phys. Lett. B **803** (2020). https://doi.org/10.1016/j.physletb.2020.135330. Art. no. 135330
8. Iess, A., Cuoco, E., Morawski, F., Powell, J.: Core-collapse supernova gravitational-wave search and deep learning classification. Mach. Learn.: Sci. Technol. **1**(2) (2020). https://doi.org/10.1088/2632-2153/ab7d31. Art. no. 025014
9. Abbott, B. P., et al.: GWTC-1: A gravitational-wave transient catalog of compact binary mergers observed by LIGO and Virgo during the first and second observing runs. Phys. Rev. X **9**(3) (2019). https://doi.org/10.1103/PhysRevX.9.031040. Art. no. 031040
10. Abbott, R., et al.: GWTC-2: Compact binary coalescences observed by LIGO and Virgo during the first half of the third observing run. Phys. Rev. X **11**(2) (2021). https://doi.org/10.1103/PhysRevX.11.021053. Art. no. 021053
11. Abbott, R., et al.: GWTC-3: Compact Binary Coalescences Observed by LIGO and Virgo During the Second Part of the Third Observing Run. Physical Review X **13**(4) (2023). https://doi.org/10.1103/PhysRevX.13.041039. Art. no. 041039
12. Aasi, J., et al.: Advanced LIGO. Class. Q. Gravity **32**(7) (2015). https://doi.org/10.1088/0264-9381/32/7/074001. Art. no. 074001
13. Acernese, F., et al.: Advanced Virgo: a second-generation interferometric gravitational wave detector. Class. Q. Gravity **32**(2) (2014). https://doi.org/10.1088/0264-9381/32/2/024001. Art. no. 024001
14. Wang, H., Wu, S., Cao, Z., Liu, X., Zhu, J.Y.: Gravitational-wave signal recognition of LIGO data by deep learning. Phys. Rev. D **101**(10) (2020). https://doi.org/10.1103/PhysRevD.101.104003. Art. no. 104003
15. George, D., Huerta, E.A.: Deep neural networks to enable real-time multimessenger astrophysics. Phys. Rev. D **97**(4) (2018). https://doi.org/10.1103/PhysRevD.97.044039. Art. no. 044039
16. George, D., Huerta, E.: Deep learning for real-time gravitational wave detection and parameter estimation: Results with Advanced LIGO data. Phys. Lett. B **778**, 64–70 (2018). https://doi.org/10.1016/j.physletb.2017.12.053
17. Fan, X., Li, J., Li, X., Zhong, Y., Cao, J.: Applying deep neural networks to the detection and space parameter estimation of compact binary coalescence with a network of gravitational wave detectors. Sci. China Phys. Mech. & Astron. **62**(6), 1–8 (2019). https://doi.org/10.1007/s11433-018-9321-7
18. Gabbard, H., Williams, M., Hayes, F., Messenger, C.: Matching matched filtering with deep networks for gravitational-wave astronomy. Phys. Rev. Lett. **120**(14) (2018). https://doi.org/10.1103/PhysRevLett.120.141103. Art. no. 141103
19. Gebhard, T.D., Kilbertus, N., Harry, I., Schölkopf, B.: Convolutional neural networks: A magic bullet for gravitational-wave detection? Phys. Rev. D **100**(6) (2019). https://doi.org/10.1103/PhysRevD.100.063015. Art. no. 063015
20. Boashash, B. (ed.): Time-Frequency Signal Analysis and Processing: A Comprehensive Reference, 2 edn. EURASIP and Academic Press Series in Signal and Image Processing. Academic Press, London, UK (2016)

21. Lopac, N., Hržić, F., Petrijevčanin Vuksanović, I., Lerga, J.: Detection of non-stationary GW signals in high noise from Cohen's class of time-frequency representations using deep learning. IEEE Access **10**, 2408–2428 (2022). https://doi.org/10.1109/ACCESS.2021.3139850
22. Lopac, N.: Detection of gravitational-wave signals from time-frequency distributions using deep learning. Ph.D. thesis, University of Rijeka, Faculty of Engineering (2022)
23. Lerga, J., Lopac, N., Bačnar, D., Hržić, F.: Combining time-frequency signal analysis and machine learning with an example in gravitational-wave detection. Rad Hrvatske akademije znanosti i umjetnosti. Tehničke znanosti **554**(22), 99–130 (2023). https://doi.org/10.21857/yq32ohx1v9
24. Abbott, R., et al. (LIGO Scientific Collaboration and Virgo Collaboration): Open data from the first and second observing runs of Advanced LIGO and Advanced Virgo. SoftwareX **13** (2021). https://doi.org/10.1016/j.softx.2021.100658. Art. no. 100658
25. LIGO Scientific Collaboration: The O2 Data Release (2019). https://doi.org/10.7935/CA75-FM95
26. LIGO Scientific Collaboration: LIGO Algorithm Library - LALSuite. free software (2018). URL https://doi.org/10.7935/GT1W-FZ16
27. Nitz, A., et al.: gwastro/pycbc: PyCBC Release v1.13.6 (2019). https://doi.org/10.5281/zenodo.2643618
28. Bohé, A., et al.: Improved effective-one-body model of spinning, nonprecessing binary black holes for the era of gravitational-wave astrophysics with advanced detectors. Phys. Rev. D **95** (2017). https://doi.org/10.1103/PhysRevD.95.044028. Art. no. 044028
29. Abbott, B.P., et al.: A guide to LIGO–Virgo detector noise and extraction of transient gravitational-wave signals. Class. Q. Gravity **37**(5) (2020). https://doi.org/10.1088/1361-6382/ab685e. Art. no. 055002
30. Jurdana, V., Lopac, N., Vrankic, M.: Sparse time-frequency distribution reconstruction using the adaptive compressed sensed area optimized with the multi-objective approach. Sensors **23**(8) (2023). https://doi.org/10.3390/s23084148. Art. no. 4148
31. Jurdana, V., Vrankic, M., Lopac, N., Jadav, G. M.: Method for automatic estimation of instantaneous frequency and group delay in time–frequency distributions with application in EEG seizure signals analysis. Sensors **23**(10) (2023). https://doi.org/10.3390/s23104680. Art. no. 4680
32. Cohen, L.: Time-Frequency Analysis. Prentice-Hall PTR, Upper Saddle River, NJ, USA (1995)
33. Hlawatsch, F., Auger, F. (eds.): Time-Frequency Analysis: Concepts and Methods. Digital Signal and Image Processing Series. ISTE/Wiley, London, UK / Hoboken, NJ, USA (2013)
34. Boashash, B.: Note on the use of the Wigner distribution for time-frequency signal analysis. IEEE Trans. Acoust. Speech Signal Proc. **36**(9), 1518–1521 (1988). https://doi.org/10.1109/29.90380
35. Claasen, T., Mecklenbräuker, W.: The Wigner distribution - A tool for time-frequency signal analysis, Parts I–III. Philips J. Res. **35**, 217–250, 276–300, 372–389 (1980)
36. Flandrin, P.: Some features of time-frequency representations of multicomponent signals. In: 1984 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 1984), vol. 9, pp. 266–269. IEEE, San Diego, CA, USA (1984). https://doi.org/10.1109/ICASSP.1984.1172741
37. Choi, H., Williams, W.J.: Improved time-frequency representation of multicomponent signals using exponential kernels. IEEE Trans. Acoust. Speech Signal Proc. **37**(6), 862–871 (1989). https://doi.org/10.1109/ASSP.1989.28057
38. Papandreou, A., Boudreaux-Bertels, G.F.: Generalization of the Choi-Williams distribution and the Butterworth distribution for time-frequency analysis. IEEE Trans. Signal Proc. **41**(1), 463–472 (1993). https://doi.org/10.1109/TSP.1993.193179
39. Sejdic, E., Djurovic, I., Jiang, J.: Time-frequency feature representation using energy concentration: An overview of recent advances. Digital Signal Proc. **19**(1), 153–183 (2009). https://doi.org/10.1016/j.dsp.2007.12.004
40. Cohen, L.: Generalized phase-space distribution functions. J. Math. Phys. **7**(5), 781–786 (1966). https://doi.org/10.1063/1.1931206

41. Cordero, E., de Gosson, M., Nicola, F.: On the reduction of the interferences in the Born-Jordan distribution. Appl. Comput. Harmon. Anal. **44**(2), 230–245 (2018). https://doi.org/10.1016/j.acha.2016.04.007
42. Jeong, J., Williams, W.J.: Kernel design for reduced interference distributions. IEEE Trans. Signal Proc. **40**(2), 402–412 (1992). https://doi.org/10.1109/78.124950
43. Zhao, Y., Atlas, L.E., Marks, R.J.: The use of cone-shaped kernels for generalized time-frequency representations of nonstationary signals. IEEE Trans. Acoust. Speech Signal Proc. **38**(7), 1084–1091 (1990). https://doi.org/10.1109/29.57537
44. Guo, Z., Durand, L., Lee, H.C.: The time-frequency distributions of nonstationary signals based on a Bessel kernel. IEEE Trans. Signal Proc. **42**(7), 1700–1707 (1994). https://doi.org/10.1109/78.298277
45. Auger, F., Flandrin, P., Gonçalvès, P., Lemoine, O.: Time-Frequency Toolbox: Reference Guide. CNRS/Rice University, France/USA (1996)
46. Williams, W.J., Jeong, J.: Reduced interference time-frequency distributions. In: B. Boashash (ed.) Time-Frequency Signal Analysis: Methods and Applications, chap. 3, pp. 74–97. Longman-Cheshire/Wiley, Melbourne/New York (1992)
47. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016), pp. 770–778. IEEE, Las Vegas, NV, USA (2016). https://doi.org/10.1109/CVPR.2016.90
48. Wu, Z., Shen, C., van den Hengel, A.: Wider or deeper: revisiting the ResNet model for visual recognition. Pattern Recognit. **90**, 119–133 (2019). https://doi.org/10.1016/j.patcog.2019.01.006
49. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017), pp. 1800–1807. IEEE, Honolulu, HI, USA (2017). https://doi.org/10.1109/CVPR.2017.195
50. Tan, M., Le,: Q.: EfficientNet: Rethinking model scaling for convolutional neural networks. In: 36th International Conference on Machine Learning (ICML 2019). Proceedings of Machine Learning Research, vol. 97, pp. 6105–6114. PMLR, Long Beach, CA, USA (2019)
51. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: MobileNetV2: inverted residuals and linear bottlenecks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2018), pp. 4510–4520. IEEE, Salt Lake City, UT, USA (2018). https://doi.org/10.1109/CVPR.2018.00474
52. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2018), pp. 7132–7141. IEEE, Salt Lake City, UT, USA (2018). https://doi.org/10.1109/TPAMI.2019.2913372

Chapter 18
Deep Residual Networks for Gravitational Wave Astronomy

Paraskevi Nousi, Alexandra Eleni Koloniari, Nikolaos Stergioulas, and Anastasios Tefas

Abstract Gravitational wave astronomy has emerged as a new branch of observational astronomy, since the first detection of gravitational waves in 2015. The current number of $O(100)$ detections is expected to grow by several orders of magnitude over the next two decades. As a result, current computationally expensive detection algorithms will become impractical. A solution to this problem, which has been explored in the last years, is the application of machine learning techniques to accelerate the detection of gravitational wave sources. In this chapter, the application of deep residual networks in achieving rapid detections with high sensitivity is presented. In particular, the AresGW algorithm, implemented using a 54-layer deep residual network, has demonstrated a remarkable ability to achieve a high detection rate in real noise. The results of the first Machine Learning Gravitation Wave Mock Data Challenge (MLGWSC-1) have underscored the effectiveness of the AresGW algorithm, highlighting its higher sensitivity over traditional detection algorithms, such as matched filtering or wavelet-based approaches. The introduction of Deep Adaptive Input Normalization (DAIN) and curriculum learning strategies has allowed substantial improvements in the training efficiency and robustness of AresGW. This progress is poised to bring about a new era in gravitation-wave astronomy, where machine learning will pave new paths for exploration and discovery.

P. Nousi
Swiss Data Science Center, ETH, Zürich, Switzerland

A. E. Koloniari · N. Stergioulas (✉)
Department of Physics, Aristotle University of Thessaloniki, Thessaloniki, Greece
e-mail: niksterg@auth.gr

A. E. Koloniari
e-mail: akolonia@auth.gr

A. Tefas
Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece
e-mail: tefas@csd.auth.gr

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_18

18.1 Introduction

Ever since 2015, when the initial detection of gravitational waves (GWs) from a binary black hole (BBH) system took place [1], the frequency of GW detections has been on the rise, nearing a point where they could be considered commonplace. Following the third observation run (O3), the latest catalog (GWTC-3, [2]) from the collaboration of Advanced LIGO [3], Advanced Virgo [4] and KAGRA [5, 6] listed 90 GW events, the majority of which were BBH mergers. The fourth observation run (O4) is in progress and alerts for more BBH detections are issued [7]. The integration of a fifth interferometer, LIGO-India [8], is projected to considerably improve both the sensitivity and the sky localization of the network. In addition, the development of third-generation ground-based detectors, such as the Einstein Telescope [9, 10] and the Cosmic Explorer [11, 12], is underway, and they are expected to significantly broaden our knowledge of astrophysical processes in the Universe [13–15].

Progress in GW astronomy, as previously outlined, has been the result of combined efforts across various domains. Accurate descriptions of the complete coalescence process, covering the entire inspiral, merger, and ringdown phases, can be achieved through different approaches, with `IMRPhenomXPHM` [16] and `SEOBNRv5PHM` [17] serving as two examples of advanced waveform models. The latest versions of these models consider the spin-induced precession of the binary orbit and the input from both the dominant and subdominant multipole moments of the emitted gravitational waves. However, increasing the complexity of a waveform model also results in significantly higher computational costs.

The Equation of State (EoS) of Neutron Stars (NSs) has been the focus of numerous investigations facilitated by astronomical observations. These investigations encompass the NICER mass and radius measurements [18–20], the evaluation of tidal deformability via gravitational waves [21–24], and also combined constraints, for instance, [25–28]. The discovery of the binary NS merger GW170817 [29, 30] has particularly stimulated additional studies in this field.

The use of machine learning techniques in the analysis of gravitational wave data has seen a significant rise in recent years (refer to [31–33] for reviews). This chapter offers an overview of various applications of machine learning in the field of gravitational-wave astronomy as discussed in [34–37].

The use of conventional matched filtering methods to achieve rapid detection is becoming progressively expensive and potentially unfeasible [38], due to both computational efficiency and precision. This is particularly the case for near-threshold systems with arbitrary spin orientations, which necessitate a significantly larger parameter space than in the aligned-spin scenario. The situation is likely to become even more challenging if template banks that deviate from general relativity (GR) are incorporated. However, unmodeled search algorithms exhibit restricted sensitivity, dependent on the specific GW source.

Lately, the application of machine-learning (ML) techniques, for instance, convolutional neural networks (CNN) or auto-encoders, has been explored as a promising approach to the challenge of identifying gravitational waves (GWs), see for example

[39–73] and [31–33] for reviews. Assessing the effectiveness of these efforts in a practical context has presented challenges. The inaugural Machine Learning Gravitational Wave Mock Data Challenge (MLGWSC-1) was successfully completed [74], offering an objective platform to gauge the sensitivity and performance of ML algorithms on modeled BBH injections in Gaussian and O3a detector noise, relative to conventional algorithms (see also [75] for updated results). In [36] a comprehensive presentation of the leading ML algorithm (AresGW) in the context of real O3a noise was provided and it was demonstrated that it could exceed the results achieved by standard setups of traditional algorithms in this particular scenario. This was achieved within a component mass range of $7-50M_{\odot}$ (equivalent to 70% of the events announced in the cumulative GWTC catalog [2]) and a relatively low false alarm rate (FAR) as small as one per month.

The AresGW algorithm, as described in [36], incorporates various elements that enhance the sensitive distance. It uses a 54-layer one-dimensional deep residual network (ResNet) [76], with greater capability than a standard CNN and Deep Adaptive Input Normalization (DAIN) [77] to tackle the non-stationary characteristics of O3a noise. In addition, the data set was augmented during training. The execution speed was improved with the introduction of a module-based whitening layer specific to the framework, which calculates the power spectral density (PSD) in a batched tensor format. Lastly, curriculum learning was utilized, enabling the network to initially learn waveforms with the highest signal-to-noise ratio (SNR). The network was constructed using PyTorch [78] and underwent training (inclusive of validation) on 12 days of data in 31 h on an A6000 GPU (spanning 14 epochs). Assessment of a month of test data on identical hardware was completed in less than 2 h. The key conclusions of [36] are summarized below.

18.1.1 Training and Test Datasets

The data set described in [36] covered a 12-day interval, incorporating real noise from the O3a LIGO operation along with nonaligned binary black hole waveform injections as specified in dataset 4 of [74]. This noise originated from O3a segments accessible through the Gravitational Wave Open Science Center (GWOSC) [79]. The selection included only segments that were at least 2 h long, featured high-quality data from both LIGO detectors, and omitted any 10-second periods surrounding events recorded in GWTC-2 (see [74] for additional details). Following these selection criteria, the dataset featured noise from the two aLIGO detectors, Hanford (H1) and Livingston (L1), spanning a total of 11 weeks and recorded at a sampling rate of 2048 Hz.

The training set waveforms were produced using the IMRPhenomXPHM model [16], with an initial frequency limit of 20 Hz. The component masses, m_1 and m_2, ranged from 7 to 50 solar masses, leading to a maximum signal duration of about 20 s. The distribution of the signals was uniform throughout the coalescence phase, polarization, inclination, declination, and right ascension (see [74] for further details).

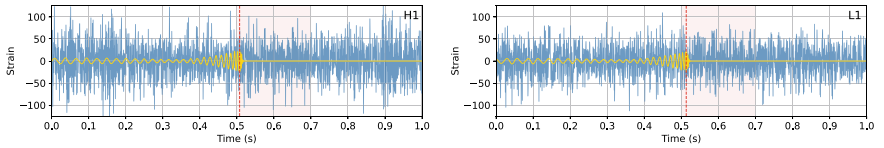


Fig. 18.1 A 2-channel segment of data from the training set used in [36] is displayed in the left and right panels, respectively, for the Hanford (H1) and Livingston (L1) detectors. The whitened strain of a 1 s segment is shown around the time of coalescence. The coalescence times in the detector frames are within the 0.5–0.7 s range (indicated by the shaded area). The injected waveform is scaled to match the difference between the whitened foreground and background segments. Figure from [36]

Unlike volume, which was not uniformly sampled, the chirp distance d_c [74] was specifically chosen to sample the luminosity distance d, enhancing the detection of smaller mass systems. The component spins were isotropically oriented, ranging from 0 to 0.99. The model included all available higher-order modes up to $(4, -4)$. Figure 18.1 displays a typical segment of the training set data.

The initial 12 days of the 11-week dataset constituted the training data set, yielding 740 k segments of background noise. Each of 38 k distinct waveforms was randomly integrated into approximately 19 different background segments, producing 740 k foreground segments with injections, thus forming a balanced training set. A validation set was also established, derived from weeks 4 to 7 of the 11-week dataset using a unique random seed for injections. The test dataset included noise from weeks 8 to 11 and featured injections with merger times randomly distributed between 24 and 30 s. The same random seed and offset as cited in [74] were applied to the test dataset.

Initially, the training data underwent a pre-processing step involving whitening, as referenced in [39, 80], followed by normalization through the DAIN algorithm [77, 81]. The DAIN model is trained by back-propagating gradients to update its parameters. Additionally, DAIN has the capability to modify the normalization approach used on the input *during inference*, enabling it to manage data that is non-stationary.

18.1.2 Deep Residual Networks

Residual neural networks [76] utilize skip connections to enhance training effectiveness, facilitating the flow of gradients to the initial layers of the architecture, thereby addressing the issue of vanishing gradients [82]. Coupled with sophisticated training techniques [83], this approach results in improved training outcomes as layers are added, enabling the development of significantly deeper networks compared to traditional CNNs.

The deep residual network described in [36] used 1D convolutions to classify 1-second long segments (2×2048-dimensional) as positive (with injection) or negative (solely noise). This network consisted of 54 layers organized into 27 blocks, each containing two convolutional layers with different numbers of filters. Stride-2 convolutions were implemented in blocks 5, 8, 11, 14, and 17, effectively reducing the dimensionality by half and incorporating an extra layer in the residual connection. Following each convolutional layer, batch normalization and ReLU activation were applied. The final two convolutional layers narrowed the output to a binary result (either noise with an injected waveform or just noise). The backpropagation was optimized using the Adam algorithm [84], and the loss function used was the regularized binary cross entropy, a modification of the finite cross-entropy loss function [56]. Dynamic augmentation was applied during the training phase. The network design is depicted in Fig. 18.2.

A learning strategy was formulated where the network first learns from the strongest injections before progressing to weaker ones. This method utilizes the ideal signal-to-noise ratio (SNR) of the injected signal, as obtained by the following equation

$$\mathrm{SNR} = 2\sqrt{\int_0^\infty df \frac{\tilde{h}(f)^2}{S_n(f)}}, , \tag{18.1}$$

where $\tilde{h}(f)$ is the amplitude of the Fourier transform of the injected signal and $S_n(f)$ is the power spectral density of the detector noise. Rather than employing the true optimal SNR, an empirical formula was established based solely on the chirp mass, the distance, and the inclination angle ι

$$\mathrm{SNR} = \frac{1261\ \mathrm{Mpc}}{D}\left(\frac{\mathcal{M}_\mathrm{c}}{M_\odot}\right)^{5/7}[0.7 + 0.3\cos(2\,\iota)]\,. \tag{18.2}$$

Figure 18.3 illustrates a comparison between the optimal SNR (computed using Eq. (18.1) and the PSD of the Hanford detector) for a set of 10^4 randomly selected injections and the estimated SNR as calculated by Eq. (18.2). Although the two distributions appear similar, the real SNR is influenced by various additional factors (such as the sky position and spin orientations).

Training initially started with signals that were clearly distinguishable and possessed a high estimated SNR during the initial four epochs. The network was then

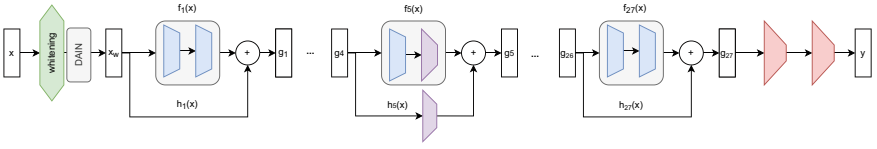


Fig. 18.2 Description of the residual network architecture in [36]. The input $\mathbf{x}$ is 2×2048-dimensional (see the text for details). Figure from [36]

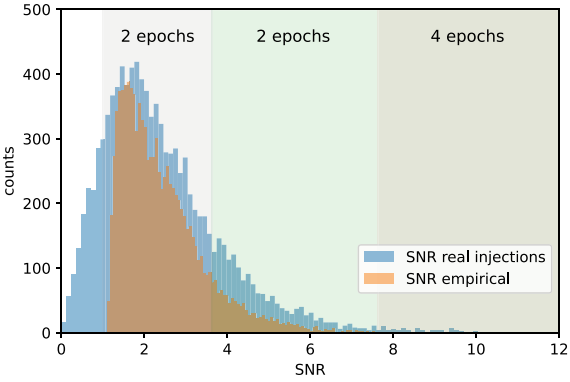


Fig. 18.3 Comparison of the SNR histogram, calculated with the empirical relation in Eq. (18.2), to the optimal SNR of 10^4 randomly chosen injections. The shaded areas represent the limited SNR values used in the first eight epochs of the learning strategy. Figure from [36]

incrementally trained on weaker signals, mastering all signals from the training set by the tenth epoch. Figure 18.3 illustrates the comparison of the first eight epochs with the SNR distribution. Due to the adopted training approach, the initial losses were low. Once the network had learned all the signals by the tenth epoch, the loss of the training set matched the loss of the validation set.

18.1.3 Detection of BBH Injections in Real Noise

The trained network analyzed test dataset segments, yielding a binary output indicative of either the presence of an injection or merely noise. This initial output served as a ranking statistic $\mathcal{R}$ (ranging from 0 to 1). A positive result was noted when $\mathcal{R} > 0.5$. Positives detected within a 0.3 s interval were clustered and reported as a single event (refer to Fig. 18.4 for an illustration). Post-deployment, the output was assessed every 0.1 s against the test dataset's known injection times. A positive output within 0.3 s of a specific injection's nominal merger time was deemed a true positive; otherwise, it was considered a false positive.

To evaluate the performance of the search algorithm, the false alarm rate is initially computed as a function of the ranking statistic, denoted as FAR($\mathcal{R}$). Following this, the search's sensitivity is calculated based on the ranking statistic, establishing a correlation between sensitivity and FAR (refer to [74] for detailed definitions and methodology).

Figure 18.5 illustrates the sensitive distance as a function of FAR for the best model (ResNet54d + SNR), compared to a basic model (ResNet54) and two leading GW detection algorithms, Coherent WaveBurst (cWB) and PyCBC. cWB, a waveform model-independent search method for GW signals, utilizes the constrained likelihood approach [85–87]. On the other hand, PyCBC [88] is based on a standard archival search configuration for compact-binary mergers [89]. The results for cWB and PyCBC shown in Fig. 18.5 are derived from [74], where they were tested on the same dataset. cWB employs wavelets, limiting its ability to achieve optimal

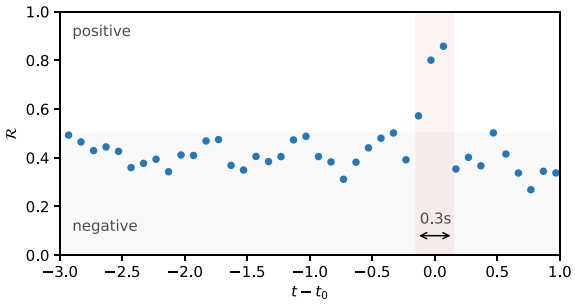


Fig. 18.4 Representative example of the ranking statistic for overlapping segments (with a stride of 0.1s). Segments with $\mathcal{R} < 0.5$ are classified as negatives. Segments with $\mathcal{R} > 0.5$ that cluster within 0.3s of a known injection at time t_0 are reported as a single true positive. Figure from [36]

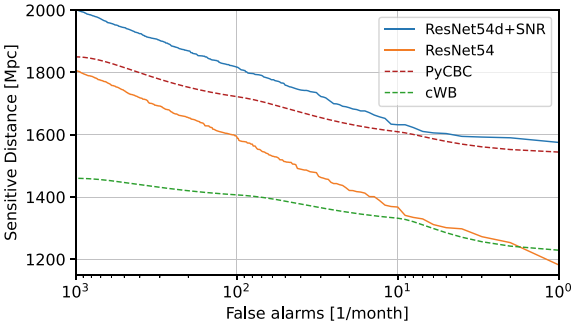


Fig. 18.5 Comparison of the performance of the best model (ResNet54d+SNR) in [36] with the simpler setup (ResNet54) used in [74] and two leading algorithms for GW detection, Coherent WaveBurst (cWB) and PyCBC (which only used aligned-spin templates, see text). All codes were tested on the same dataset established in [74]. The ResNet54d+SNR model outperformed the other algorithms at all false alarm rates in this setting. Figure from [36]

fitting factors, but has been enhanced with machine learning strategies recently [90]. PyCBC uses matched filtering of waveform templates, but the studies [74, 89] only used aligned spin templates, which limits its effectiveness for more complex waveforms due to computational constraints. The test data set in [36] used more complex waveforms, and hence this specific PyCBC search did not achieve optimal fitting factors. As depicted in Fig. 18.5, the best model in [36], which incorporated SNR-based curriculum learning, outperformed the specific PyCBC results at all FAR levels and also significantly exceeded the sensitivity of the unmodeled cWB search.

18.2 Conclusions

In this chapter, we have explored significant advancements in the field of gravitational wave astronomy through the application of machine learning techniques, specifically focusing on deep residual networks. The AresGW algorithm, implemented using a 54-layer deep residual network, has demonstrated remarkable ability to enhance the detection of gravitational waves in real noise. The results of the Machine Learning Gravitational Wave Mock Data Challenge (MLGWSC-1) have underscored the effectiveness of the AresGW model, highlighting its higher sensitivity over traditional detection algorithms, such as matched filtering or wavelet-based approaches. This version of AresGW identified gravitational wave signals across a mass range of 7–50 solar masses, which constitute approximately 70% of the events cataloged in GWTC. The introduction of Deep Adaptive Input Normalization (DAIN) and curriculum learning strategies has allowed substantial improvements in the training efficiency and robustness of AresGW.

As we look towards the future, the integration of more advanced machine learning techniques and the upgrades of gravitational wave detector networks promise to deepen our understanding of the universe's astrophysical processes. This progress is poised to bring about a new era in gravitational wave astronomy, where machine learning not only augments existing detection capabilities, but also opens up new avenues for exploration and discovery.

Acknowledgements We are grateful to our collaborators, Panagiotis Iosif, Nikolaos Passalis, for their contributions that led to the main publication summarized in this review. This work was carried out within the framework of the EU COST action No. CA17137. AK and NS acknowledge funding by the European Union through the Horizon Project 101131928 (ACME). This research has made use of data or software obtained from the Gravitational Wave Open Science Center (gwosc.org), a service of the LIGO Scientific Collaboration, the Virgo Collaboration, and KAGRA. This material is based upon work supported by NSF's LIGO Laboratory which is a major facility fully funded by the National Science Foundation, as well as the Science and Technology Facilities Council (STFC) of the United Kingdom, the Max-Planck-Society (MPS), and the State of Niedersachsen/Germany for support of the construction of Advanced LIGO and construction and operation of the GEO600 detector. Additional support for Advanced LIGO was provided by the Australian Research Council. Virgo is funded, through the European Gravitational Observatory (EGO), by the French Centre National de Recherche Scientifique (CNRS), the Italian Istituto Nazionale di Fisica Nucleare (INFN) and the Dutch Nikhef, with contributions by institutions from Belgium, Germany, Greece, Hungary, Ireland, Japan, Monaco, Poland, Portugal, Spain. KAGRA is supported by Ministry of Education, Culture, Sports, Science and Technology (MEXT), Japan Society for the Promotion of Science (JSPS) in Japan; National Research Foundation (NRF) and Ministry of Science and ICT (MSIT) in Korea; Academia Sinica (AS) and National Science and Technology Council (NSTC) in Taiwan.

References

1. Abbott, B.P., et al.: Observation of gravitational waves from a binary black hole merger. Phys. Rev. Lett. **116**(6), 061102 (2016)
2. Abbott, R., et al.: GWTC-3: compact binary coalescences observed by LIGO and Virgo during the second part of the third observing run. Phys. Rev. X **13**(4), 041039 (2023)
3. Aasi, J., et al.: Advanced LIGO. Class. Quantum Gravity **32**, 074001 (2015)
4. Acernese, F., et al.: Advanced Virgo: a second-generation interferometric gravitational wave detector. Class. Quantum Gravity **32**(2), 024001 (2014)
5. Akutsu, T., et al.: KAGRA: 2.5 generation interferometric gravitational wave detector. Nat. Astron. **3**(1), 35–40 (2019)
6. Akutsu, T., et al.: Overview of KAGRA: Detector design and construction history. Progr. Theor. Exp. Physi. **2021**(5), 08 (2020). 05A101
7. Abbott, B.P., et al.: Prospects for observing and localizing gravitational-wave transients with Advanced LIGO, Advanced Virgo and KAGRA. Living Rev. Relat. **23**(1), 3 (2020)
8. Saleem, M., et al.: The science case for LIGO-India. Class. Quantum Gravity **39**(2), 025004 (2022)
9. Punturo, M., et al.: The Einstein Telescope: a third-generation gravitational wave observatory. Class. Quantum Gravity **27**(19), 194002 (2010)
10. Maggiore, M., et al.: Science case for the Einstein telescope. J. Cosmol. Astropart. Phys. **2020**(3), 050 (2020)
11. Reitze, D., et al.: Cosmic Explorer: The U.S. contribution to gravitational-wave astronomy beyond LIGO. Bull. Am. Astron. Soc. **51**, 35 (2019)
12. Evans, M., et al.: A Horizon Study for Cosmic Explorer: Science, Observatories, and Community (2021). arXiv:2109.09882
13. Abbott, B.P., et al.: Exploring the sensitivity of next generation gravitational wave detectors. Class. Quantum Gravity **34**(4), 044001 (2017)
14. Reitze, D., et al. Expanding the Reach of Gravitational Wave Astronomy to the Edge of the Universe: The Gravitational-Wave International Committee Study Reports on Next Generation Ground-based Gravitational-Wave Observatories (2021). arXiv:2111.06986
15. Kalogera, V., et al.: The Next Generation Global Gravitational Wave Observatory: The Science Book (2021). arXiv:2111.06990
16. Pratten, G., et al.: Computationally efficient models for the dominant and subdominant harmonic modes of precessing binary black holes. Phys. Rev. D **103**(10), 104056 (2021)
17. Ramos-Buades, A., et al.: SEOBNRv5PHM: Next generation of accurate and efficient multipolar precessing-spin effective-one-body waveforms for binary black holes (2023). arXiv:2303.18046
18. Riley, T.E., et al.: A NICER view of PSR j0030+0451: Millisecond pulsar parameter estimation. Astrophys. J. **887**(1), L21 (2019)
19. Miller, M.C., et al.: PSR j0030+0451 mass and radius from nicer data and implications for the properties of neutron star matter. Astrophys. J. Lett. **887**(1), L24 (2019)
20. Miller, M.C., et al.: The radius of psr j0740+6620 from NICER and XMM-Newton data. Astrophys. J. Lett. **918**(2), L28 (2021)
21. Van Oeveren, E.D., Friedman, J.L.: Upper limit set by causality on the tidal deformability of a neutron star. Phys. Rev. D **95**(8) (2017)
22. Hinderer, T.: Tidal Love numbers of neutron stars. Astrophys. J. **677**, 1216–1220 (2008)
23. Chatziioannou, K.: Neutron-star tidal deformability and equation-of-state constraints. Gen. Relativ. Gravit. **52**(11), 109 (2020)
24. Dietrich, T., Hinderer, T., Samajdar, A.: Interpreting binary neutron star mergers: describing the binary neutron star dynamics, modelling gravitational waveforms, and analyzing detections. Gen. Relativ. Gravit. **53**(3), 27 (2021)
25. Biswas, B.: Bayesian model selection of neutron star equations of state using multi-messenger observations. Astrophys. J. **926**(1), 75 (2022)

26. Dietrich, T., Coughlin, M.W., Pang, P.T.H., Bulla, M., Heinzel, J., Issa, L., Tews, I., Antier, S.: Multimessenger constraints on the neutron-star equation of state and the hubble constant. Science **370**(6523), 1450–1453 (2020)
27. Landry, P., Essick, R., Chatziioannou, K.: Nonparametric constraints on neutron star matter with existing and upcoming gravitational wave and pulsar observations. Phys. Rev. D **101**(12), 123007 (2020)
28. Raaijmakers, G., Greif, S.K., Hebeler, K., Hinderer, T., Nissanke, S., Schwenk, A., Riley, T.E., Watts, A.L., Lattimer, J.M., Ho, W.C.G.: Constraints on the dense matter equation of state and neutron star properties from NICER's mass-radius estimate of PSR J0740+6620 and multimessenger observations. Astrophys. J. Lett. **918**(2), L29 (2021)
29. Abbott, B.P., et al.: GW170817: observation of gravitational waves from a binary neutron star inspiral. Phys. Rev. Lett. **119**(16), 161101 (2017)
30. Abbott, B.P., et al.: Properties of the binary neutron star merger GW170817. Phys. Rev. X **9**(1), 011001 (2019)
31. Cuoco, E., et al.: Enhancing gravitational-wave science with machine learning. Mach. Learn.: Sci. Technol. **2**(1), 011002 (2020)
32. Benedetto, V., Gissi, F., Ciaparrone, G., Troiano, L.: Ai in gravitational wave analysis, an overview. Appl. Sci. **13**(17) (2023)
33. Zhao, T., Shi, R., Zhou, Y., Cao, Z., Ren, Z.: Dawning of a New Era in Gravitational Wave Data Analysis: Unveiling Cosmic Mysteries via Artificial Intelligence – A Systematic Review (2023). arXiv:2311.15585
34. Fragkouli, S.-C., Nousi, P., Passalis, N., Iosif, P., Stergioulas, N., Tefas, A.: Deep residual error and bag-of-tricks learning for gravitational wave surrogate modeling. Appl. Soft Comput. **147**, 110746 (2023)
35. Nousi, P., Fragkouli, S.-C., Passalis, N., Iosif, P., Apostolatos, T., Pappas, G., Stergioulas, N., Tefas, A.: Autoencoder-driven spiral representation learning for gravitational wave surrogate modelling. Neurocomputing **491**, 67–77 (2022)
36. Nousi, P., Koloniari, A.E., Passalis, N., Iosif, P., Stergioulas, N., Tefas, A.: Deep residual networks for gravitational wave detection. Phys. Rev. D **108**(2), 024022 (2023)
37. Liodis, I., Smirniotis, E., Stergioulas, N.: Neural-network-based surrogate model for the properties of neutron stars in 4D Einstein-Gauss-Bonnet gravity. Phys. Rev. D **109**(10), 104008 (2024)
38. Couvares, P., et al.: Gravitational Wave Data Analysis: Computing Challenges in the 3G Era (2021). arXiv:2111.06987
39. Gabbard, H., Williams, M., Hayes, F., Messenger, C.: Matching matched filtering with deep networks for gravitational-wave astronomy. Phys. Rev. Lett. **120**, 141103 (2018)
40. George, D., Huerta, E.A.: Deep neural networks to enable real-time multimessenger astrophysics. Phys. Rev. D **97**, 044039 (2018)
41. Gebhard, T.D., Kilbertus, N., Harry, I., Schölkopf, B.: Convolutional neural networks: a magic bullet for gravitational-wave detection? Phys. Rev. D **100**(6), 063015 (2019)
42. Corizzo, R., Ceci, M., Zdravevski, E., Japkowicz, N.: Scalable auto-encoders for gravitational waves detection from time series data. Expert Syst. Appl. **151**, 113378 (2020)
43. Schäfer, M.B., Ohme, F., Nitz, A.H.: Detection of gravitational-wave signals from binary neutron star mergers using machine learning. Phys. Rev. D **102**, 063015 (2020)
44. Wang, H., Shichao, W., Cao, Z., Liu, X., Zhu, J.-Y.: Gravitational-wave signal recognition of LIGO data by deep learning. Phys. Rev. D **101**(10), 104003 (2020)
45. Krastev, P.G.: Real-time detection of gravitational waves from binary neutron stars using artificial neural networks. Phys. Lett. B **803**, 135330 (2020)
46. Skliris, V., Norman, M.R.K., Sutton, P.J.: Real-Time Detection of Unmodelled Gravitational-Wave Transients Using Convolutional Neural Networks (2020). arXiv:2009.14611
47. Lin, Y.-C., Wu, J.-H.P.: Detection of gravitational waves using Bayesian neural networks. Phys. Rev. D **103**(6), 063034 (2021)
48. Huerta, E.A., Khan, A., Huang, X., Tian, M., Levental, M., Chard, R., Wei, W., Heflin, M., Katz, D.S., Kindratenko, V., Mu, D., Blaiszik, B., Foster, I.: Accelerated, scalable and reproducible AI-driven gravitational wave detection. Nat. Astron. **5**, 1062–1068 (2021)

49. Marianer, T., Poznanski, D., Prochaska, J.X.: A semisupervised machine learning search for never-seen gravitational-wave sources. Mon. Not. R. Astr. Soc. **500**(4), 5408–5419 (2021)
50. Wei, W., Khan, A., Huerta, E.A., Huang, X., Tian, M.: Deep learning ensemble for real-time gravitational wave detection of spinning binary black hole mergers. Phys. Lett. B **812**, 136029 (2021)
51. Jadhav, S., Mukund, N., Gadre, B., Mitra, S., Abraham, S.: Improving significance of binary black hole mergers in Advanced LIGO data using deep learning: Confirmation of GW151216. Phys. Rev. D **104**(6), 064051 (2021)
52. Chaturvedi, P., Khan, A., Tian, M., Huerta, E.A., Zheng, H.: Inference-optimized ai and high performance computing for gravitational wave detection at scale. Front. Artif. Intell. **5** (2022)
53. Choudhary, S., More, A., Suyamprakasam, S., Bose, S.: Deep learning network to distinguish binary black hole signals from short-duration noise transients. Phys. Rev. D **107**(2), 024030 (2023)
54. Schäfer, M.B., Nitz, A.H.: From one to many: a deep learning coincident gravitational-wave search. Phys. Rev. D **105**, 043003 (2022)
55. Barone, F.P., Dell'Aquila, D., Russo, M.: A novel multi-layer modular approach for real-time fuzzy-identification of gravitational-wave signals. Mach. Learn. Sci. Tech. **4**(4), 045054 (2023)
56. Schäfer, M.B., Zelenka, O., Nitz, A.H., Ohme, F., Brügmann, B.: Training strategies for deep learning gravitational-wave searches. Phys. Rev. D **105**(4), 043002 (2022)
57. Baltus, G., Janquart, J., Lopez, M., Narola, H., Cudell, J.-R.: Convolutional neural network for gravitational-wave early alert: going down in frequency. Phys. Rev. D **106**, 042002 (2022)
58. Andrews, M., Paulini, M., Sellers, L., Bobrick, A., Martire, G., Vestal, H.: DeepSNR: A deep learning foundation for offline gravitational wave detection (2022). arXiv:2207.04749
59. Verma, C., Reza, A., Gaur, G., Krishnaswamy, D., Caudill, S.: Can Convolution Neural Networks Be Used for Detection of Gravitational Waves from Precessing Black Hole Systems? (2022). arXiv:2206.12673
60. Aveiro, J., Freitas, F.F., Ferreira, M., Onofre, A., Providência, C., Gonçalves, G., Font, J.A.: Identification of binary neutron star mergers in gravitational-wave data using object-detection machine learning models. Phys. Rev. D **106**(8), 084059 (2022)
61. Guo, W., Williams, D., Heng, I.S., Gabbard, H., Bae, Y.-B., Kang, G., Zhu, Z.-H.: Mimicking mergers: mistaking black hole captures as mergers. MNRAS **516**(3), 3847–3860 (2022)
62. Andrés-Carcasona, M., Menéndez-Vázquez, A., Martínez, M., Mir, L.M.: Searches for mass-asymmetric compact binary coalescence events using neural networks in the LIGO/Virgo third observation period. Phys. Rev. D **107**(8), 082003 (2023)
63. Langendorff, J., Kolmus, A., Janquart, J., Van Den Broeck, C.: Normalizing flows as an avenue to studying overlapping gravitational wave signals. Phys. Rev. Lett. **130**(17), 171402 (2023)
64. Dax, M., Green, S.R., Gair, J., Pürrer, M., Wildberger, J., Macke, J.H., Buonanno, A., Schölkopf, B.: Neural importance sampling for rapid and reliable gravitational-wave inference. Phys. Rev. Lett. **130**(17), 171403 (2023)
65. Bini, S., Vedovato, G., Drago, M., Salemi, F., Prodi, G.A.: An autoencoder neural network integrated into gravitational-wave burst searches to improve the rejection of noise transients. Class. Quantum Gravity **40**(13), 135008 (2023)
66. Tian, M., Huerta, E.A., Zheng, H.: Physics-inspired spatiotemporal-graph AI ensemble for gravitational wave detection (2023). arXiv:2306.15728
67. Murali, C., Lumley, D.: Detecting and denoising gravitational wave signals from binary black holes using deep learning. Phys. Rev. D **108**(4), 043024 (2023)
68. Bacon, P., Trovato, A., Bejger, M.: Denoising gravitational-wave signals from binary black holes with a dilated convolutional autoencoder. Mach. Learn.: Sci. Technol. **4**(3), 035024 (2023)
69. McLeod, A., Jacobs, D., Chatterjee, C., Wen, L., Panther, F.: Rapid Mass Parameter Estimation of Binary Black Hole Coalescences Using Deep Learning (2022). arXiv:2201.11126
70. Qiu, R., Krastev, P.G., Gill, K., Berger, E.: Deep learning detection and classification of gravitational waves from neutron star-black hole mergers. Phys. Lett. B **840**, 137850 (2023)

71. Jin, S.-J., Wang, Y.-X., Sun, T.-Y., Zhang, J.-F., Zhang, X.: Rapid identification of time-frequency domain gravitational wave signals from binary black holes using deep learning (2023). arXiv:2305.19003
72. Fernandes, T., Vieira, S., Onofre, A., Bustillo, J.C., Torres-Forné, A., Font, J.A.: Convolutional neural networks for the classification of glitches in gravitational-wave data streams. Class. Quantum Gravity **40**(19), 195018 (2023)
73. Freitas, O.G., Calderón Bustillo, J., Font, J.A., Nunes, S., Onofre, A., Torres-Forné, A.: Comparison of neural network architectures for feature extraction from binary black hole merger waveforms. Mach. Learn. Sci. Tech. **5**(1), 015036 (2024)
74. Schäfer, M.B., Zelenka, O., Nitz, A.H., Wang, H., Wu, S., Guo, Z.-K., Cao, Z., Ren, Z., Nousi, P., Stergioulas, N., Iosif, P., Koloniari, A.E., Tefas, A., Passalis, N., Salemi, F., Vedovato, G., Klimenko, S., Mishra, T., Brügmann, B., Cuoco, E., Huerta, E.A., Messenger, C., Ohme, F.: First machine learning gravitational-wave search mock data challenge. Phys. Rev. D **107**(2), 023021 (2023)
75. Zelenka, O., Brügmann, B., Ohme, F.: Convolutional Neural Networks for signal detection in real LIGO data (2024). arXiv:2402.07492
76. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
77. Passalis, N., Tefas, A., Kanniainen, J., Gabbouj, M., Iosifidis, A.: Deep adaptive input normalization for price forecasting using limit order book data. In: IEEE Transactions on Neural Networks and Learning Systems (2019)
78. Paszke, A., et al.: Pytorch: an imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019)
79. Abbott, R., et al.: Open data from the third observing run of LIGO, Virgo, KAGRA, and GEO. Astrophys. J. Suppl. **267**(2), 29 (2023)
80. Usman, S.A., Nitz, A.H., Harry, I.W., Biwer, C.M., Brown, D.A., Cabero, M., Capano, C.D., Dal Canton, T., Dent, T., Fairhurst, S., et al.: The pycbc search for gravitational waves from compact binary coalescence. Class. Quantum Gravity **33**(21), 215004 (2016)
81. Passalis, N., Kanniainen, J., Gabbouj, M., Iosifidis, A., Tefas, A.: Forecasting financial time series using robust deep adaptive input normalization. J. Signal Proc. Syst. **93**(10), 1235–1251 (2021)
82. Hanin, B.: Which neural net architectures give rise to exploding and vanishing gradients? Advances in Neural Information Processing Systems, 31 (2018)
83. Wightman, R., Touvron, H., Jégou, H.: Resnet strikes back: an improved training procedure in timm (2021). arXiv:2110.00476
84. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2014). arXiv:1412.6980
85. Klimenko, S., Vedovato, G., Drago, M., Salemi, F., Tiwari, V., Prodi, G.A., Lazzaro, C., Ackley, K., Tiwari, S., Da Silva, C.F., Mitselmakher, G.: Method for detection and reconstruction of gravitational wave transients with networks of advanced detectors. Phys. Rev. D **93**, 042004 (2016)
86. Drago, M., Klimenko, S., Lazzaro, C., Milotti, E., Mitselmakher, G., Necula, V., O'Brian, B., Prodi, G., Salemi, F., Szczepanczyk, M., Tiwari, S., Tiwari, V., Gayathri, V., Vedovato, G., Yakushin, I.: Coherent WaveBurst, a pipeline for unmodeled gravitational-wave data analysis. SoftwareX **14**, 100678 (2021)
87. Klimenko, S., Vedovato, G., Necula, V., Salemi, F., Drago, M., Poulton, R., Chassande-Mottin, E., Tiwari, V., Lazzaro, C., O'Brian, B., Szczepanczyk, M., Tiwari, S., Gayathri, V.: cWB pipeline library: 6.4.1 (2021)
88. Alex, N., et al.: gwastro/pycbc: v2.0.5 release of PyCBC (2022)

89. Nitz, A.H., Kumar, S., Wang, Y.-F., Kastha, S., Shichao, W., Schäfer, M., Dhurkunde, R., Capano, C.D.: 4-OGC: catalog of gravitational waves from compact binary mergers. Astrophys. J. **946**(2), 59 (2023)
90. Mishra, T., O'Brien, B., Szczepańczyk, M., Vedovato, G., Bhaumik, S., Gayathri, V., Prodi, G., Salemi, F., Milotti, E., Bartos, I., Klimenko, S.: Search for binary black hole mergers in the third observing run of advanced ligo-virgo using coherent waveburst enhanced with machine learning. Phys. Rev. D **105**, 083018 (2022)

Chapter 19
Convolutional Neural Networks for Signal Detection in Real LIGO Data

Ondřej Zelenka, Bernd Brügmann, and Frank Ohme

Abstract Results of recent publications on machine-learning based gravitational-wave searches vary greatly due to differences in evaluation procedures. The Machine Learning Gravitational-Wave Search Challenge [1] was organized to resolve these issues and produce a unified framework for machine-learning search evaluation. Six teams submitted contributions, four of which are based on machine learning methods and two are state-of-the-art production analyses. This chapter is a modified version of [2], which describes the submission from our team titled TPI FSU Jena and its updated variant. We also apply this algorithm to real O3b data and recover the relevant events of the GWTC-3 catalog. Reprinted with permission from [2]. Copyright (2024) by the American Physical Society.

19.1 Introduction

One of the most powerful known sources of gravitational waves (GWs) is a compact binary coalescence: the final stage of a binary system of compact objects, such as black holes or neutron stars. Analyzing the signal from such an event allows us to constrain the source parameters, such as component masses, which are relevant to the study of the black hole population in the universe, and the mechanism by which

O. Zelenka (✉)
Astronomical Institute of the Czech Academy of Sciences, 14100 Prague, Czech Republic
e-mail: zelenka@asu.cas.cz

Friedrich-Schiller-Universität Jena, 07743 Jena, Germany

Michael Stifel Center Jena, 07743 Jena, Germany

B. Brügmann
Friedrich-Schiller-Universität Jena, 07743 Jena, Germany

Michael Stifel Center Jena, 07743 Jena, Germany

F. Ohme
Max-Planck-Institut für Gravitationsphysik, Albert-Einstein-Institut, 30167 Hannover, Germany

Leibniz Universität Hannover, 30167 Hannover, Germany

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_19

supermassive black holes are formed [3, 4]. For this reason, GW observations are crucial in expanding our understanding of the universe.

Most contemporary detection pipelines are based on *matched filtering* [5] and use a *template bank* of expected waveforms. These pipelines are highly sensitive to signals covered by the template bank, but less sensitive to others. *Loosely modeled searches* are a complementary approach: they do not require the advanced knowledge of waveforms to be searched for, but they are less sensitive to compact-binary mergers than matched-filter searches [6–8].

With the broadening of the sensitive frequency range of detectors, it becomes necessary to increase the density of template banks. In addition, expanding the parameter-space of interest typically requires more templates to cover the signal manifold. This causes a steep rise in the size of template banks and therefore computational time of matched-filtering based algorithms. In particular, this is an issue when incorporating effects such as eccentricity [9], precession [10, 11], or higher-order modes [11, 12].

Moreover, matched-filter searches are optimal for an idealized Gaussian noise distribution. However, actual detector data deviate from this assumption [13]. While measures are taken to reduce the effect of this deviation, there are still optimizations to be made. These are some of the driving forces behind the search for new, more efficient methods to complement the matched-filter based analyses.

A rather new development is to use machine learning (ML) methods in GW astronomy. This was started by two pioneering papers on the topic of GW detection [14, 15]. Their approach consisted of applying convolutional neural networks to recognize whether individual 1 s-long whitened samples of Gaussian noise contain a binary black hole (BBH) GW signal. Additionally, applications in parameter estimation [16–18], denoising [19, 20], fast waveform generation [21], and more [22] have also been published; we, however, remain focused on the detection problem in this article.

In recent years, a multitude of new results have been achieved on this topic [22–25]. However, due to differing choices in generation of test data, results in the literature are difficult to compare to each other. To resolve this issue, the First Machine Learning Gravitational-Wave Search Challenge (MLGWSC-1) [1, 26] has been organized. From 12 October 2021 until 14 April 2022, multiple teams developed ML based algorithms for detection of GW signals originating in BBH mergers in month-long streams of data from the two US-based Laser Interferometer Gravitational Observatory (LIGO) detectors [27]. The final test data were unknown to participants but followed a known distribution of both noise and sources, and no scoreboard was kept during the challenge. Eventually, 4 ML based submissions were received, as well as 2 conventional algorithms to provide a baseline. Their performance has been evaluated in detail and effects responsible for differing performance of submissions have been isolated.

We have authored one of the challenge submissions, titled “TPI FSU Jena”. On test datasets following a simplified Gaussian noise distribution, our search was the top ML submission and performed close to the matched-filter baseline, a similar submission being a close second. In addition, it had a comparatively short runtime.

However, on test data generated using LIGO open data [28], non-Gaussian noise artifacts polluted the search results to a large degree.

This section is a modified version of the work [2], which is published under the CC BY 4.0 license. In this work, we first briefly describe the MLGWSC-1, our submission, and choices made during its development. Following that, we describe the steps taken to further optimize the contribution after the end of the challenge, which greatly improve its performance when non-Gaussian noise transients are present in the data. Finally, we demonstrate the power of the developed searches by applying them to open data from the second half of the third observing run and recovering the GWTC-3 catalog events lying in the relevant portion of the source parameter space.

19.2 MLGWSC-1

19.2.1 Test Data

The test data consist of 2 strains from the LIGO Hanford and Livingston detectors. The script used to generate them was available to participants of the challenge with the option to specify its seed. For the final evaluation, a challenge dataset in the length of one month was generated after the challenge deadline using a previously unknown seed [1].

The test data exist in 4 levels named *datasets* of progressively increasing difficulty. The first three use background noise generated by a colored Gaussian model, while the fourth uses real noise from the O3a observing run [28]. The injection complexity is also increasing, from non-spinning, dominant mode only, to precessing waveforms with generic misaligned spins and multiple higher-order modes. For the sake of brevity, we only cover datasets 3 and 4, for details see the challenge paper [1].

Test data are generated using the script `generate_data.py` supplied by the MLGWSC-1 [26], which creates the background noise, generates waveforms and injects them into the noise, forming the foreground. Both the background and the foreground are stored in HDF5 files [29], each containing groups titled L1 and H1 for the Livingston and Hanford detectors, with the full length of the strain split into multiple segments labeled by their GPS start time. These segments are generated independently of each other. All time series are sampled at a rate of 2048 Hz. A low frequency cutoff of 15 Hz is applied to the background noise to allow for reduction in data size of the real detector noise to be downloaded.

The injection parameters are generated by the astrophysical distribution for all angular parameters and the distance is specified by generating the chirp distance, defined as [30]

$$d_c = d \cdot \left(\frac{\mathcal{M}_{c,0}}{\mathcal{M}_c} \right)^{5/6} , \tag{19.1}$$

where d is the luminosity distance, $\mathcal{M}_c = (m_1 m_2)^{3/5} / (m_1 + m_2)^{1/5}$ is the chirp mass, and $\mathcal{M}_{c,0} = 1.4/2^{1/5} M_\odot$ is a fiducial chirp mass. The squared chirp distance is drawn from a uniform distribution over the interval $d_c^2 \in \left[130^2\ \mathrm{Mpc}^2,\ 350^2\ \mathrm{Mpc}^2\right]$. Component masses are drawn in the detector frame from uniform distributions over different intervals depending on the dataset, following the primary/secondary mass constraint $m_1 \geq m_2$.

The events are placed at random intervals between 24 s and 30 s between merger times. The waveforms are generated using the `IMRPhenomXPHM` [31] phenomenological model, capable of accurate modeling of precession and higher-order modes. They are then projected on the corresponding detectors and injected into the background data to produce the foreground.

In the third dataset, Gaussian noise is generated using an unknown power spectral density (PSD). From a set of 20 PSDs derived from the O3a observing run data [28], for each detector and each individual segment, one is randomly chosen and used to generate the noise (see Sect. 19.3.1). Component masses $m_1,\ m_2 \in \left[7M_\odot,\ 50M_\odot\right]$ are drawn from a uniform distribution. The magnitudes of component spins are uniform from 0 to 0.99, and their directions are isotropically distributed. All higher-order modes available to the `IMRPhenomXPHM` [31] approximant are used, and the low frequency cutoff is chosen to be 20 Hz.

In the fourth dataset, real LIGO noise is used. A real noise file in the extent of approximately 3 months has been prepared by the MLGWSC-1 team, the data generation script randomly chooses segments from it to comprise the dataset background, and the L1 stream is time-shifted with respect to H1 by a random amount in order to introduce different noise realizations. The injections are generated in a manner identical to the third dataset.

19.2.2 Evaluation Procedure

The evaluation is done in a similar manner to [23, 24]. The submitted algorithms are applied to background data without any injections as well as to data with BBH injections to determine the relationship between their false-alarm rate (FAR) and sensitive distance.

Each submitted algorithm is required to take a file in the format described in Sect. 19.2.1 as input and produce a file containing identified candidate events as output. It must be an HDF5 file containing 3 datasets referring to the GPS time of the events, the ranking statistics and the tolerance for error in time.

The evaluation is performed by the `evaluate.py` script supplied by the MLGWSC-1 [26]. It requires the outputs of the submission algorithm on both the foreground and the background files as input, identifies true positives and determines the FAR at varying detection thresholds. The relationship between the FAR and the sensitive distance is in principle similar to the receiver operating characteristic, which is the relationship between the percentage of false positives and true positives as one varies the threshold for identification of a positive.

To obtain the sensitivity curve of an algorithm based on the identified background and foreground events, we first count the number of background events with a ranking statistic greater than the threshold. Dividing by the total duration of the background data analyzed (in this case 30 days = 2592000 s), we find the FAR. The sensitive volume of the search at FAR = $\mathcal{F}$ can be calculated by [32]

$$V(\mathcal{F}) = \int\int \epsilon(\mathcal{F};\ \mathbf{x},\ \Lambda)\,\phi(\mathbf{x},\ \Lambda)\ \mathrm{d}\mathbf{x}\mathrm{d}\Lambda\ , \tag{19.2}$$

where $\mathbf{x}$ are an injection's spatial coordinates, Λ the other injection parameters, $\epsilon(\mathcal{F};\ \mathbf{x},\ \Lambda)$ is the efficiency of the search, and $\phi(\mathbf{x},\ \Lambda)$ is the injection parameter distribution. If we denote $N_{I,\mathcal{F}}$ the number of found injections at a FAR = $\mathcal{F}$ and $\mathcal{M}_{c,i}$, $i = 1, \ldots, N_{I,\mathcal{F}}$ the chirp masses of the found injections, the expression simplifies to [1]

$$V(\mathcal{F}) \approx \frac{V(d_{\max})}{N_I} \sum_{i=1}^{N_{I,\mathcal{F}}} \left(\frac{\mathcal{M}_{c,i}}{\mathcal{M}_{c,\max}} \right)^{5/2} , \tag{19.3}$$

where N_I is the total number of injections and $\mathcal{M}_{c,\max}$ is the upper limit of injected chirp masses.

We then call the graph of the sensitive volume $V(\mathcal{F})$ as a function of the FAR the algorithm's *sensitivity curve* and these are the main criterion for the challenge evaluation.

The runtimes of submitted algorithms were also measured and are available in the challenge paper [1]. All submitted algorithms are evaluated on standardized hardware, provided by the challenge organizers. The hardware consists of a total of 8 Intel Xeon Silver 4215 cores at 2.5 GHz, 192 GB of RAM, and 8 nVidia RTX 2070 GPUs with CUDA support, 8 GB of VRAM each.

19.3 Experimental Setup

19.3.1 Data Processing

As described in [13], the standard noise model in LIGO detectors is correlated in the time domain. However, using the Fourier transform, in the frequency domain the noise is uncorrelated and described by a Gaussian distribution with zero mean and a frequency-dependent variance called the PSD and denoted $S_n(f)$.

Let us use $\tilde{\mathbf{d}}$ to denote the Fourier transform of a time series $\mathbf{d}$. Following [13], the transformation

$$\mathbf{d} \mapsto \mathbf{d}_w, \quad \tilde{d}_w(f) = \frac{\tilde{d}(f)}{\sqrt{S_n(f)}} \tag{19.4}$$

yields a time series with a flat PSD, corresponding to white noise. This process is called *whitening* and is a common method in GW data analysis. The PSDs in GW detectors rise steeply towards both low and high frequencies and the signals are dominated by strong noise at frequencies outside the most sensitive band of the detectors.

Following [23], we feed whitened data to the ML model. When applying to test data, the algorithm first estimates the PSD of the time series in question using Welch's method [33] with a segment duration of 0.5 s, then symmetrically truncates the time-domain response of the $S_n\,(f)^{-1/2}$ whitening filter to a width of 0.25 s, and uses this PSD to whiten the entire time series. This is done for each segment in the input data separately, as well as for each detector.

As the noise in LIGO detectors is not stationary over timescales on the order of days, one must account for the PSD drift. This is addressed by slicing the data into chunks shorter than the PSD-drift timescale in the test data generation process [26].

19.3.2 Training and Validation Data

In the training and validation datasets, the noise is taken from the real noise file provided by the MLGWSC-1. A segment from the file is chosen at random, its PSD is estimated and used to whiten the entire segment, and the whitened segment is sliced into 1-second samples. While the noise generation loop is running, these slices are used sequentially, and once all have been used, a new segment is whitened and sliced in the same manner. The PSD is retained through the processing of the entire segment for whitening of waveform injections.

To generate the waveform injections, we apply the Python package PyCBC [34]. The distributions of individual parameters are summarized in Table 19.1, they follow the distributions used in test datasets 3 and 4 (see Sect. 19.2.1) with exceptions, which we describe in the following paragraphs. A limited number of noise samples (given for each experiment in Sect. 19.4) are injected with a waveform and assigned the label $(1,\ 0)$, the remaining ones remain pure noise and are assigned the label $(0,\ 1)$. However, the waveforms are normalized to a network optimal signal-to-noise ratio (SNR) $\rho_{\mathrm{net}} = 1$ during the data generation procedure and only injected at a randomly generated $\rho_{\mathrm{net}} \in [7,\ 20]$ at each training epoch. Due to the SNR normalization, the luminosity distance is irrelevant and a fiducial 1 Mpc value (the PyCBC default) is passed to the approximant.

For consistency with the experiments of [23], we set the lower mass limit to $10M_\odot$ instead of $7M_\odot$ used to generate test datasets 2–4. An additional training run confirms that including the range $\left[7M_\odot,\ 10M_\odot\right]$ in the training data does not improve the performance of the search. We suspect this is due to the increased length of waveforms in this region of the parameter space [1], due to which a part of the waveform's SNR is outside the network's input window when the merger is aligned.

Furthermore, due to an oversight on our part, the inclination angle does not follow the astrophysical distribution $\cos\iota \in [-1,\ 1]$. However, this is not expected to pose

Table 19.1 Distributions from which waveform injection parameters are drawn. Intervals refer to a uniform distribution

Parameter	Uniform distribution
Approximant	`IMRPhenomXPHM`
Component masses	$m_1 \geq m_2 \in \left[10M_\odot,\ 50M_\odot\right]$
Spin magnitudes	$\left\|\boldsymbol{\chi}_1\right\|,\ \left\|\boldsymbol{\chi}_2\right\| \in [0,\ 0.99]$
Spin directions	Isotropic
Coalescence phase	$\Phi_0 \in [0,\ 2\pi)$
Inclination angle	$\iota \in [0,\ 2\pi]$
Declination	$\sin\theta \in [-1,\ 1]$
Right ascension	$\varphi \in [-\pi,\ \pi)$
Polarization angle	$\Psi \in [0,\ 2\pi)$
Sampling rate	2048 Hz
Low frequency cutoff	20 Hz

an issue, as the dominant effect of the inclination angle on the waveforms is a constant rescaling [35], which is lost as we normalize the waveforms to a fixed network SNR. A rerun of the code for the MLGWSC-1 submission with the astrophysical distribution confirms that the results are indistinguishable.

Both the training and validation data are generated by following the steps below:

1. Get noise:
   a. get next slice from current segment
   b. if segment finished, choose a new one at random, whiten it, slice it, and take its first slice
2. If applicable, generate waveform:
   a. set up parameters (see Table 19.1)
   b. generate waveform
   c. crop so that merger is within the given interval, append zeros
   d. whiten using the PSD of the corresponding noise segment
   e. normalize to optimal $\rho_{\text{net}} = 1$
3. Store noise and waveform separately. At each training epoch, inject at a newly generated optimal SNR.

For the original submission, we choose the training dataset to contain 500000 pure noise samples and 500000 noise+waveform samples, the validation dataset to contain 100000 pure noise samples and 100000 noise+waveform samples, and the sliced real noise is used. The updated submission instead uses 750000 pure noise samples and 250000 noise+waveform samples in the training dataset, and 150000 pure noise samples and 50000 noise+waveform samples in the validation dataset.

19.3.3 Test Data

Test data meant for evaluation before submitting are generated using the program `generate_data.py` supplied by the MLGWSC-1 [26]. For final testing, all 4 datasets are generated with the length of 2592000 s = 30 days, and the seed is set to 4261537. Dataset 4 with this seed is used to select the optimal epoch during training for the MLGWSC-1 submission (see Sect. 19.4.1) and to optimize the updated submission in Sect. 19.4.3.

The seeds used for generating the challenge datasets to evaluate the submitted algorithms to the MLGWSC-1 and to compute the final sensitivity curves are given in Sect. 19.2.1.

19.3.4 Machine Learning

The MLGWSC-1 is aimed at evaluating the performance of ML algorithms. In its simplest form, this corresponds to a model with an arbitrary number of free parameters whose error is being optimized over a large dataset. This is frequently done using gradient-descent based optimizers and their stochastic varieties, which approximate the gradients on small batches of the dataset in their successive iterations. The error function being optimized here is a modification of the binary cross entropy loss [23]

$$C\left(\bar{\mathbf{Y}},\ \mathbf{Y}\right) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{n} Y_{ij}\log\left(\left(1-\varepsilon\right)\bar{Y}_{ij}+\varepsilon\right)\ , \tag{19.5}$$

designed to remove divergences when an element of $\bar{\mathbf{Y}}$ is zero using the regularization parameter $0 < \varepsilon \ll 1$.

Neural networks are a class of ML models built of artificial neurons, these are functions defined as

$$f:\ \mathbb{R}^n \to \mathbb{R}\,, \tag{19.6a}$$

$$\mathbf{x} \mapsto \sigma\left(\sum_{i=1}^{n} w_i x_i + b\right)\ . \tag{19.6b}$$

The parameters w_i and b are called the weights and bias, respectively, and are optimized through the training process. The function σ is called an *activation function*, a popular choice we use here is the Exponential Linear Unit [36] with $\alpha = 1$

$$\mathrm{ELU}\left(z\right) = \begin{cases} \alpha\left(\exp\left(z\right)-1\right) & \text{if } z < 0\,, \\ z & \text{if } z \geq 0\,. \end{cases} \tag{19.7}$$

A feed-forward neural network is organized in layers of independent neurons, each of which feeds its output into neurons of the following layer. They can be fully connected, i.e. the input of each neuron consists of the outputs of all neurons in the previous layer, also called dense layers. In this paper, we also make use of convolutional layers, whose structure corresponds to a set of filters sliding over a multichannel input [37]. This reduces the number of independent connections and thus weights in the network.

Further components are max pooling layers, which act as a downsampling operation [38], and dropout layers, which improve the training convergence through a type of noise injection [39, 40]. For an introduction to ML and neural networks, we refer the reader to [41, 42].

19.3.5 Model Architecture

In this work, we use a simple convolutional neural network (CNN) design, which is an extension of the architecture used in [23]. For a simple implementation of the method used there, we first define a CNN called the *base network*, which does not have a final activation. Its architecture is shown in Table 19.2.

Unlike coincident searches such as the CNN-Coinc submission [24] to the MLGWSC-1, wherein the streams from each detector are analyzed separately and combined using a probability-based formula, we employ a coherent approach. The network accepts a two-channel input to carry data from two detector streams.

The base network produces 2 outputs, which we denote x_0, x_1. Following the method of [23], for training we append a Softmax layer

$$y_i = \text{Softmax}\,(\mathbf{x})_i = \frac{\exp(x_i)}{\sum_j \exp\left(x_j\right)}\,, \tag{19.8}$$

which maps its inputs to a set of positive numbers which sum up to one. The purpose of this activation is to represent uncalibrated probabilities [43] of different classes in classification problems, and in this case we wish the output y_0 to represent the probability that the input sample contains an astrophysical GW signal.

The networks are trained using the stochastic Adam optimizer [44] with a learning rate of $\gamma = 4 \cdot 10^{-6}$, and the other parameters set to their defaults in PyTorch [45] ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$), for a total of 250 epochs. The training dataset is split into batches of 32 samples, and each epoch consists of one optimizer step per batch.

When testing in the same manner, however, a numerical issue arises. In single-precision floating point arithmetic using PyTorch, y_0, which we would like to use as the ranking statistic of the resulting search, rounds up to 1 when $x_0 - x_1 \gtrsim 16$, which is well in the range of values encountered by the search. To resolve this, we rewrite Eq. (19.8) for $i = 0$ as

Table 19.2 Architecture of the base network. It accepts an input with 2 channels corresponding to 2 detector streams and possesses 635318 trainable weights. "KS" refers to kernel size, and "shape" is the output shape of the corresponding layer. The batch normalization layer is only used in the original submission to the MLGWSC-1 but not in the improved searches

Layer	KS	Shape	Activation
Input		2×2048	
(batch norm)		2×2048	
Convolution	33	16×2016	ELU
Convolution	32	16×1985	ELU
Convolution	17	16×1969	ELU
Convolution	16	16×1954	ELU
Max pooling	4	16×488	
Convolution	17	16×472	ELU
Convolution	16	32×457	ELU
Convolution	9	32×449	ELU
Convolution	8	32×442	ELU
Max pooling	3	32×147	
Convolution	9	32×139	ELU
Convolution	8	64×132	ELU
Convolution	9	64×124	ELU
Convolution	8	64×117	ELU
Max pooling	2	64×58	
Flatten		3712	
Dense		128	ELU
Dropout		128	
Dense		128	ELU
Dropout		128	
Dense		2	

$$y_0 = \frac{1}{1 + \exp(x_1 - x_0)} = \frac{1}{1 + \exp(-\Delta x)} \,. \tag{19.9}$$

We see that y_0 is a purely growing function of $\Delta x = x_0 - x_1$, which is therefore an equivalent ranking statistic, without suffering from the same numerical issue. Therefore, Δx is used as the ranking statistic in the search. This technique is called the unbounded softmax replacement. For more detailed information see [23].

The CNN is only part of the detection algorithm following [23], as it only accepts simple 1-second-long slices. The full algorithm consists of feeding overlapping slices of the test data to the network, applying a threshold, and clustering the results into candidate detections. First, the entire segment is whitened using the method described in Sect. 19.3.1.

Then, the segment is sliced into 1-s long samples with an offset of 0.1 s, which are fed to the network and the Δx outputs recorded. Because the networks are trained on injections with merger time 0.6–0.8 s after the sample start time, each slice is associated with the time 0.6 s after the start time of the slice, in order to compensate for this alignment.

A threshold is applied to the network outputs and those which exceed it are clustered by time, with a minimal separation of 0.35 s between clusters. Each of these clusters is then considered a candidate event to be saved in the output file. As the ranking statistic, the maximum of the network outputs in the cluster is used, and the time corresponding to the maximum is used as the time of the candidate event. For all output events, the value of 0.2 s is chosen as the time uncertainty in the search output (see Sect. 19.2.2), to match the size of the merger alignment interval in the training data.

The value of this first threshold is largely irrelevant in the context of a gravitational-wave search, certain ranges allowing for more events at higher FARs without compromising the low FAR end. The value of -8 is chosen as it maximizes the benefit in this case. This is demonstrated in the original publication [2].

19.4 Results

19.4.1 MLGWSC-1 Submission

The training and validation losses are monitored during the training, and their evolution is shown in Fig. 19.1. Out of the local minima of the validation loss, the global minimum as well as two earlier local minima are chosen and further tested by

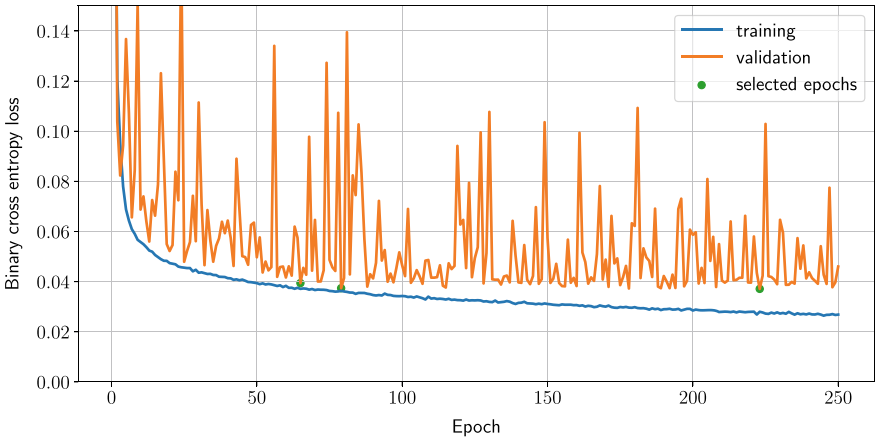


Fig. 19.1 Evolution of the training and validation loss values throughout the training of the MLGWSC-1 submission

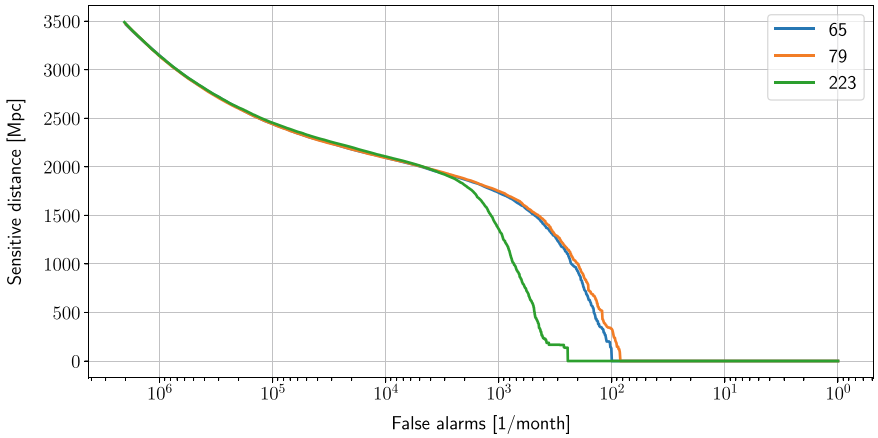


Fig. 19.2 Sensitivity curves (sensitive distance versus false alarm rate, see Sect. 19.2.2) of the network at 3 minima of the validation loss highlighted in Fig. 19.1 used to select the final network state for the submission. The test data used is dataset 4, generated in the length of 30 days with the seed set to 4261537

applying to the test datasets 3 and 4, the result for dataset 4 is shown in Fig. 19.2. The results on dataset 3 were virtually indistinguishable, for better performance on dataset 4 we choose the network state at epoch 79 for the submission to the MLGWSC-1.

19.4.2 MLGWSC-1 Results

The MLGWSC-1 received a total of six contributions, four of which are ML based. The remaining two are conventional analyses to provide a baseline; the first is the matched-filtering based PyCBC [32], the other is the loosely modeled search cWB [7, 46].

Rather than an extensive coverage of the MLGWSC-1 results, which are described in great detail in [1], this section focuses on a particular issue which occurs when real noise is presented to our algorithm. We would like to specifically bring to the reader's attention the performance of our algorithm (labeled D: TPI FSU Jena) and the algorithm labeled E: Virgo-AUTh, whose sensitivity curves on datasets 3 and 4 are shown in Fig. 19.3. Both ML submissions are plotted as dashed lines, in addition the PyCBC submission is shown.

Both submissions use a very similar approach. In the final testing, their performances are close to each other with D operating at a slightly higher sensitivity at all FARs, this gap widening as we approach $\mathcal{F} = 1\ \text{month}^{-1}$, on test dataset 3 (in fact, this holds on all datasets which use Gaussian noise [1]). However, the Virgo-AUTh algorithm retains $\geq 90\%$ of the sensitive distance of the TPI FSU Jena search

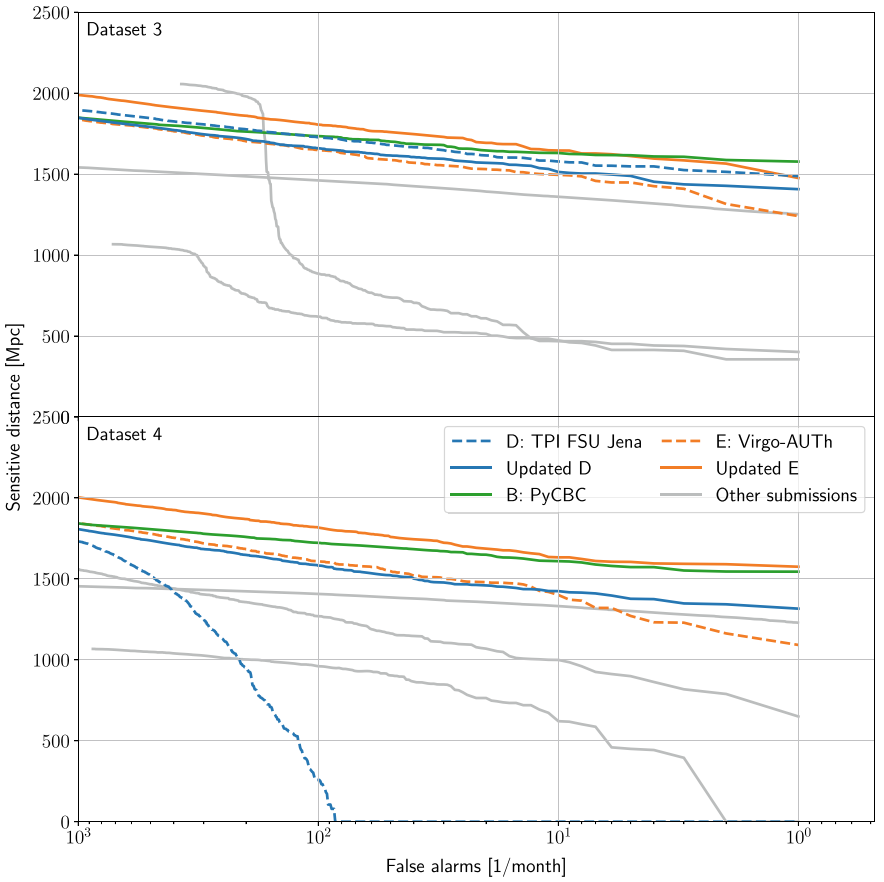


Fig. 19.3 Sensitivity curves of 3 selected submissions, along with updated versions of 2 of them, on datasets 3 and 4 of the MLGWSC-1. Each panel contains the performance of the submissions on one test dataset. Dashed lines mark conventional analyses and solid lines mark ML-based search algorithms. In case of the TPI FSU Jena and Virgo-AUTh teams, the dotted lines mark the original submissions, while the solid lines mark the updated algorithms. The remaining submissions are shown in gray for illustration of overall challenge results

at $\mathcal{F} \geq 2$ month^{-1}, and at $\mathcal{F} = 1000$ month^{-1} this gap narrows to a separation of roughly 4%.

Moving to dataset 4, the performance of the Virgo-AUTh algorithm degrades only mildly. In contrast, the performance of our submission deteriorates much more, losing all sensitivity at $\mathcal{F} < 10^2$ month^{-1}. This is due to omnipresent noise transients in real detector data, which produce triggers louder than the injected waveforms.

Finally, the runtimes of our algorithm are consistently lower than those of the other submissions. On average, the Virgo-AUTh search takes ∼50% longer to run on the challenge hardware on all 4 test datasets due to the higher complexity of its

network architecture. On datasets 2–4 the estimated runtimes of PyCBC are ∼40 times as large. We note that the given PyCBC runtimes are estimations as a different hardware setup is used to run the search.

19.4.3 Updated Submission

The updated submission uses the same optimization procedure as the original, with two modifications: a) the batch normalization layer is removed, b) the ratio of the two classes in the training and validation data is shifted.

Six training runs are performed. At each epoch, the network's sensitivity is evaluated on test data with real noise, and from each run the state with the highest sensitive distance at $\mathcal{F} = 1\,\text{month}^{-1}$ is chosen and labeled following the format `R<runnumber1-6>/<4-digitepochnumber>`. Of the resulting 6 states, we choose `R1/0021` for the final search algorithm as it has the highest sensitivity. Its sensitivity curves are shown in Fig. 19.3 alongside the curves of all submissions as well as the updated Virgo-AUTh search, called AResGW [47, 48].

The sensitivity on datasets using Gaussian noise deteriorates slightly; this is to be expected as one optimizes for a different noise distribution, rejecting potential glitches in data containing none. At $\mathcal{F} = 1\,\text{month}^{-1}$, the sensitive distance is reduced by 5.4%. In the overall ranking, ours remains the most sensitive of all ML submissions on Gaussian noise.

On real noise, the updated submission reaches the highest sensitivity of all ML submissions at $\mathcal{F} \lesssim 10\,\text{month}^{-1}$ and is narrowly outperformed by Virgo-AUTh at higher FARs. At $\mathcal{F} = 1\,\text{month}^{-1}$, our updated submission has a sensitive distance of 1316 Mpc, and Virgo-AUTh operates at 87% of this value. At the same time, the updated version of the Virgo-AUTh algorithm outperforms ours in both cases.

19.4.4 Application to O3b Data

The O3 LIGO observing run was split by a commissioning break into two phases, O3a and O3b [28, 49]. The first part is used to train the CNNs above to recognize BBH waveform injections in real LIGO noise. In this section, we apply the searches developed above to real data recorded by LIGO through the O3b phase and cross-reference the output with the transients recorded in the GWTC-3 catalog [5].

To query O3b data, we require a minimum segment length of one minute and the same data quality requirements as the real noise file used in the MLGWSC-1, known injections are not removed. This leaves us with a total of 8 228 706 s of data in a total of 2377 segments, amounting approximately to 95 days and 6 h. In comparison, the full O3b observing run was 147 days and 2 h in length.

We apply all 6 searches trained in Sect. 19.4.3 to these data. The GWTC-3 catalog [5] consists of 35 confident detections and 7 marginal ones. Events lying outside

Table 19.3 List of O3b events omitted from Table 19.4. Events listed in the first column are omitted due to insufficient data quality in either detector, and events listed in the second column are omitted due to being missed completely by all searches. The exception is GW191105_143521, which is recovered at $\mathcal{F} = 658$ month^{-1} by the `R5/0193` search and missed by the others

Data quality	Missed
GW200302_015811	GW191129_134029
GW200129_065458	GW200115_042309
GW200112_155838	GW200202_154313
GW191216_213338	GW200316_215756
	GW191105_143521
	GW191219_163120
	GW191103_012549
	GW200210_092254
	GW200220_061928

the segments of available data are excluded, leaving us with 31 confident and 4 marginal events to be found. These excluded events are listed in the left column of Table 19.3. In addition, we confirm that none of the events contained in available segments take place closer to either end of their respective segments than 46 s.

A catalog event is marked as found, if the search output contains an event within 0.2 s of the time given in the catalog, and it is assigned its corresponding ranking statistic t. The remaining catalog events are considered missed, and the remaining events reported by the search are considered false alarms. The catalog event is then considered detected at a FAR of

$$\mathcal{F} = \frac{N_{f>t}}{T} , \tag{19.10}$$

where $N_{f>t}$ is the number of false alarms louder than t, and T is the total length of the analyzed segments. In addition, if the FAR of an event is at least 1000 month^{-1}, it is also considered missed.

None of the events marked marginal in the GWTC-3 catalog are found by either of the searches. The resulting FARs of confident events in the analyzed segments are shown in Table 19.4. The table is split into three sections: in the first, the 90% credible intervals on both component masses lie fully in the $\left[10M_\odot,\ 50M_\odot\right]$ range used for training the networks, while in the third, at least one of them lies fully outside $\left[10M_\odot,\ 50M_\odot\right]$. The remaining cases are contained in the second section. The credible intervals and accompanying SNR values come from the catalog's parameter estimation pipeline based on Bilby [50, 51] and are supplied by GWOSC [52].

Let us comment shortly on the results of Table 19.4. Most importantly, all events in the first section, where the search algorithms are expected to operate at a high sensitivity, are found by all 6 tested networks at a FAR lower than 40 month^{-1},

Table 19.4 List of O3b events from the `GWTC-3-confident` catalog [5] and their FARs as identified by the 6 final searches. Events which are not recovered by the given search are marked by a hyphen. 13 events are omitted (see Table 19.3). The events are grouped into three sections based on their estimated component masses (see text for details)

		R1/0021	R2/0150	R3/0036	R4/0038	R5/0193	R6/0026
Event name	ρ_{MF}	$\mathcal{F}$ [month^{-1}]					
GW200224_222234	20.0	0.0	0.9	0.0	0.0	0.0	0.0
GW200311_115853	17.8	0.0	0.9	0.6	1.3	0.0	6.0
GW200225_060421	12.5	0.0	1.6	0.0	0.0	1.9	0.3
GW191215_223052	11.2	0.0	1.9	1.3	1.3	0.0	1.3
GW200208_130117	10.8	19.2	2.2	3.5	3.1	1.6	10.1
GW200219_094415	10.7	5.0	4.7	8.8	38.7	12.6	19.5
GW200209_085452	9.6	1.3	2.8	0.9	2.8	2.2	0.3
GW191204_110529	8.8	1.6	3.1	0.0	3.1	5.0	3.1
GW200308_173609	7.1	–	–	–	–	–	–
GW191222_033537	12.5	0.0	4.1	2.5	2.5	0.3	0.3
GW200128_022011	10.6	25.5	3.1	0.0	11.7	10.4	2.2
GW191230_180458	10.4	6.6	149	19.5	98.9	36.9	5.0
GW191127_050227	9.2	38.7	2.8	4.4	18.6	3.1	6.6
GW200220_124850	8.5	215	517	956	96.1	695	375
GW191126_115259	8.3	–	–	–	–	–	–
GW200216_220804	8.1	–	189	–	–	841	–
GW191113_071753	7.9	–	634	391	–	713	647
GW200306_093714	7.8	485	407	720	–	69.3	–
GW200208_222617	7.4	38.1	6.0	19.5	55.1	159	187
GW200322_091133	6.0	810	898	–	–	–	–
GW191204_171526	17.5	3.5	8.8	4.1	4.4	7.6	6.0
GW191109_010717	17.3	0.0	1.9	0.9	0.6	1.3	0.9

with the exception of GW200308_173609, which is the second weakest event in the catalog at $\rho_{\mathrm{MF}} = 7.1$. In the vast majority, the events are detected at $\mathcal{F} < 4\,\mathrm{month}^{-1}$.

In the second and third sections the searches are expected to operate at a reduced sensitivity as the corresponding parameter space is not fully covered in the training dataset. This is confirmed in Table 19.4, however, louder events at $\rho_{\mathrm{MF}} \gtrsim 9$ and $\rho_{\mathrm{MF}} \gtrsim 17$ in the second and third section, respectively, are also mostly detected at $\mathcal{F} < 10\,\mathrm{month}^{-1}$ by the ML-based searches.

As a final comment, Q-scan spectrograms of the loudest false alarms in the analyzed data seem to be consistent with them being known types of glitches. Full outputs of all 6 search algorithms as well as spectrograms of the 128 loudest events of each are publicly available in the data release [53].

19.5 Conclusion

We have presented a convolutional neural network-based gravitational-wave detection algorithm capable of performing comparably to conventional algorithms in specific settings, and its implementation, submitted to the MLGWSC-1. While the submission performs well on test data using Gaussian noise, the noise transients present in the data with real noise prove to be too much of a challenge and reduce its sensitivity to zero at relevant FARs. In the present work, we resolve this issue by a careful optimization of the training parameters and demonstrate that the updated search outperforms all other original challenge submissions besides the PyCBC matched-filter search.

At the same time, while each independent run of the updated algorithm converges to a state with high sensitivity of the resulting search, a detailed analysis reveals that the sensitivity is highly non-monotonic during the training [54]. In addition, Fig. 19.1 also shows unexpected oscillations in the validation loss. This phenomenon is not yet fully understood and warrants further investigation.

As a final application of the updated search, we analyze open data from the O3b observing run [28] of the LIGO-Virgo collaboration and cross-reference the results with the corresponding catalog GWTC-3 [5]. We demonstrate that in the intended regime of BBHs with component masses between $10M_\odot$ and $50M_\odot$, our searches can confidently detect events with a network SNR above 8. This is in line with contemporary matched-filter based searches, as the value 8 roughly corresponds to 1 false alarm per month [23].

Code

The code provided by the MLGWSC-1 organizers is available in [26]. The code used in the analyses is contained in [53]. The submission to the MLGWSC-1 as described in Sects. 19.3, 19.4.1 is stored in the directory `mlgwsc-1`. For the experiments detailed in Sects. 19.4.3, 19.4.4, the code is available in the subdirectory `correction` along with additional materials and results.

Acknowledgements O.Z. thanks the Carl Zeiss Foundation for the financial support within the scope of the program line "Breakthroughs" and is supported by the fellowship Lumina Quaeruntur No. LQ100032102 of the Czech Academy of Sciences. Further support has been provided by the COST network CA17137 "G2net".

The computational experiments were performed on the ARA cluster at the Friedrich-Schiller-Universität Jena and the Atlas cluster financed by the Gottfried Wilhelm Leibniz Universität Hannover and the Max-Planck-Gesellschaft through the Albert-Einstein-Institut Hannover. We thank the Observational Relativity and Cosmology division for access. F.O. acknowledges support by the Max Planck Independent Research Group Program.

Special thanks go to Marlin Schäfer for his great contribution through his work on organizing the MLGWSC-1 as well as discussions, and to all contributors to the challenge.

Competing Interests The authors have no conflicts of interest to declare that are relevant to the content of this chapter.

References

1. Schäfer, M.B., Zelenka, O., Nitz, A.H., Wang, H., Wu, S., Guo, Z.K., Cao, Z., Ren, Z., Nousi, P., Stergioulas, N., Iosif, P., Koloniari, A.E., Tefas, A., Passalis, N., Salemi, F., et al.: Phys. Rev. D **107**, 023021 (2023)
2. Zelenka, O., Brügmann, B., Ohme, F.: Phys. Rev. D **110**, 024024 (2024)
3. Abbott, B.P., Abbott, R., Abbott, T.D., Abraham, S., Acernese, F., Ackley, K., Adams, C., Adhikari, R.X., Adya, V.B., Affeldt, C., Agathos, M., Agatsuma, K., Aggarwal, N., Aguiar, O.D., Aiello, L., et al.: The Astrophysical Journal Letters **882**(2), L24 (2019)
4. Abbott, R., Abbott, T.D., Acernese, F., Ackley, K., Adams, C., Adhikari, N., Adhikari, R.X., Adya, V.B., Affeldt, C., Agarwal, D., Agathos, M., Agatsuma, K., Aggarwal, N., Aguiar, O.D., Aiello, L., et al.: Phys. Rev. X **13**, 011048 (2023)
5. Abbott, R., Abbott, T.D., Acernese, F., Ackley, K., Adams, C., Adhikari, N., Adhikari, R.X., Adya, V.B., Affeldt, C., Agarwal, D., Agathos, M., Agatsuma, K., Aggarwal, N., Aguiar, O.D., Aiello, L., et al.: Phys. Rev. X **13**, 041039 (2023)
6. Klimenko, S., Mohanty, S., Rakhmanov, M., Mitselmakher, G.: Phys. Rev. D **72**, 122002 (2005)
7. Klimenko, S., Vedovato, G., Drago, M., Salemi, F., Tiwari, V., Prodi, G.A., Lazzaro, C., Ackley, K., Tiwari, S., Da Silva, C.F., Mitselmakher, G.: Phys. Rev. D **93**, 042004 (2016)
8. Lynch, R., Vitale, S., Essick, R., Katsavounidis, E., Robinet, F.: Phys. Rev. D **95**, 104046 (2017)
9. Nitz, A.H., Lenon, A., Brown, D.A.: Astrophys J **890**(1), 1 (2020)
10. Harry, I., Privitera, S., Bohé, A., Buonanno, A.: Phys. Rev. D **94**, 024012 (2016)
11. Schmidt, S., Gadre, B., Caudill, S.: (2023)
12. Harry, I., Bustillo, J.C., Nitz, A.H.: Phys. Rev. D **97**, 023004 (2018)
13. Abbott, B.P., Abbott, R., Abbott, T.D., Abraham, S., Acernese, F., Ackley, K., Adams, C., Adya, V.B., Affeldt, C., Agathos, M., Agatsuma, K., Aggarwal, N., Aguiar, O.D., Aiello, L., Ain, A., et al.: Class. Quantum Gravity **37**(5), 055002 (2020)
14. George, D., Huerta, E.A.: Phys. Rev. D **97**, 044039 (2018)
15. Gabbard, H., Williams, M., Hayes, F., Messenger, C.: Phys. Rev. Lett. **120**, 141103 (2018)
16. Chua, A.J.K., Vallisneri, M.: Phys. Rev. Lett. **124**, 041102 (2020)
17. Green, S.R., Simpson, C., Gair, J.: Phys. Rev. D **102**, 104057 (2020)
18. Gabbard, H., Messenger, C., Heng, I.S., Tonolini, F., Murray-Smith, R.: Nat. Phys. **18**(1), 112–117 (2021)
19. Bacon, P., Trovato, A., Bejger, M.: Machine Learning: Science and Technology **4**(3), 035024 (2023)
20. Shen, H., Huerta, E.A., O'Shea, E., Kumar, P., Zhao, Z.: Machine Learning: Science and Technology **3**(1), 015007 (2021)
21. Schmidt, S., Breschi, M., Gamba, R., Pagano, G., Rettegno, P., Riemenschneider, G., Bernuzzi, S., Nagar, A., Del Pozzo, W.: Phys. Rev. D **103**, 043020 (2021)
22. Cuoco, E., Powell, J., Cavaglià, M., Ackley, K., Bejger, M., Chatterjee, C., Coughlin, M., Coughlin, S., Easter, P., Essick, R., Gabbard, H., Gebhard, T., Ghosh, S., Haegel, L., Iess, A., et al.: Machine Learning: Science and Technology **2**(1), 011002 (2020)
23. Schäfer, M.B., Zelenka, O., Nitz, A.H., Ohme, F., Brügmann, B.: Phys. Rev. D **105**, 043002 (2022)
24. Schäfer, M.B., Nitz, A.H.: Phys. Rev. D **105**, 043003 (2022)
25. Schäfer, M.B., Ohme, F., Nitz, A.H.: Phys. Rev. D **102**, 063015 (2020)
26. Schäfer, M., Zelenka, O., Müller, P., Nitz, A.H.: MLGWSC-1 Release v1.4 (2022). https://doi.org/10.5281/zenodo.7107410

27. Aasi, J., Abbott, B.P., Abbott, R., Abbott, T., Abernathy, M.R., Ackley, K., Adams, C., Adams, T., Addesso, P., Adhikari, R.X., Adya, V., Affeldt, C., Aggarwal, N., Aguiar, O.D., Ain, A., et al.: Class. Quantum Gravity **32**(7), 074001 (2015)
28. Abbott, R., Abe, H., Acernese, F., Ackley, K., Adhicary, S., Adhikari, N., Adhikari, R.X., Adkins, V.K., Adya, V.B., Affeldt, C., Agarwal, D., Agathos, M., Aguiar, O.D., Aiello, L., Ain, A., et al.: Astrophys. J. Suppl. Ser. **267**(2), 29 (2023)
29. The HDF Group. Hierarchical Data Format, version 5 (1997-2023). https://www.hdfgroup.org/HDF5/
30. Abbott, B., Abbott, R., Adhikari, R., Agresti, J., Ajith, P., Allen, B., Amin, R., Anderson, S.B., Anderson, W.G., Arain, M., Araya, M., Armandula, H., Ashley, M., Aston, S., Aufmuth, P., et al.: Phys. Rev. D **77**, 062002 (2008)
31. Pratten, G., García-Quirós, C., Colleoni, M., Ramos-Buades, A., Estellés, H., Mateu-Lucena, M., Jaume, R., Haney, M., Keitel, D., Thompson, J.E., Husa, S.: Phys. Rev. D **103**, 104056 (2021)
32. Usman, S.A., Nitz, A.H., Harry, I.W., Biwer, C.M., Brown, D.A., Cabero, M., Capano, C.D., Dal Canton, T., Dent, T., Fairhurst, S., Kehl, M.S., Keppel, D., Krishnan, B., Lenon, A., Lundgren, A., et al.: Class. Quantum Gravity **33**(21), 215004 (2016)
33. Welch, P.: IEEE Trans. Audio Electroacoust. **15**(2), 70 (1967)
34. Nitz, A.H., Harry, I., Brown, D., Biwer, C.M., Willis, J.: gwastro/pycbc: v2.4.0 release of PyCBC (2023). https://doi.org/10.5281/zenodo.10013996
35. Usman, S.A., Mills, J.C., Fairhurst, S.: Astrophys J **877**(2), 82 (2019)
36. Clevert, D.A., Unterthiner, T., Hochreiter, S.: (2015)
37. LeCun, Y., Bengio, Y.: Convolutional networks for images, speech, and time series, pp. 255–258. MIT Press, Cambridge, MA, USA (1998)
38. Zafar, A., Aamir, M., Mohd Nawi, N., Arshad, A., Riaz, S., Alruban, A., Dutta, A.K., Almotairi, S.: Appl. Sci. **12**(17) (2022). https://www.mdpi.com/2076-3417/12/17/8643
39. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: arXiv:1207.0580 (2012). https://doi.org/10.48550/arXiv.1207.0580
40. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: J. Mach. Learn. Res. **15**(56), 1929 (2014)
41. Mehta, P., Bukov, M., Wang, C.H., Day, A.G.R., Richardson, C., Fisher, C.K., Schwab, D.J.: Phys. Rep. **810**, 1 (2019)
42. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016). http://www.deeplearningbook.org
43. Guo, C., Pleiss, G., Sun, Y., Weinberger, K.Q.: In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning, Proceedings of Machine Learning Research, vol 70, (PMLR, 2017), Proceedings of Machine Learning Research, vol. 70, pp. 1321–1330
44. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2017)
45. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M. et al.: In: Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019)
46. Klimenko, S., Vedovato, G., Necula, V., Salemi, F., Drago, M., Poulton, R., Chassande-Mottin, E., Tiwari, V., Lazzaro, C., O'Brian, B., Szczepanczyk, M., Tiwari, S., Gayathri, V.: cwb pipeline library: 6.4.1 (2021). https://doi.org/10.5281/zenodo.5798976. https://doi.org/10.5281/zenodo.5798976
47. Nousi, P., Koloniari, A.E., Passalis, N., Iosif, P., Stergioulas, N., Tefas, A.: Phys. Rev. D **108**, 024022 (2023)
48. Nousi, P., Koloniari, A.E., Passalis, N., Iosif, P., Stergioulas, N., Tefas, A.: AResGW: Augmentation and RESidual networks for Gravitational Wave detection (2022). https://github.com/vivinousi/gw-detection-deep-learning
49. Abbott, B.P., Abbott, R., Abbott, T.D., Abraham, S., Acernese, F., Ackley, K., Adams, C., Adya, V.B., Affeldt, C., Agathos, M., Agatsuma, K., Aggarwal, N., Aguiar, O.D., Aiello, L., Ain, A., et al.: Living Rev. Relativ. **23**, 3 (2020)

50. Ashton, G., Hübner, M., Lasky, P.D., Talbot, C., Ackley, K., Biscoveanu, S., Chu, Q., Divakarla, A., Easter, P.J., Goncharov, B., Vivanco, F.H., Harms, J., Lower, M.E., Meadors, G.D., Melchor, D., et al.: Astrophys. J. Suppl. Ser. **241**(2), 27 (2019)
51. Romero-Shaw, I.M., Talbot, C., Biscoveanu, S., D'Emilio, V., Ashton, G., Berry, C.P.L., Coughlin, S., Galaudage, S., Hoy, C., Hübner, M., Phukon, K.S., Pitkin, M., Rizzo, M., Sarin, N., Smith, R., et al.: Mon. Not. R. Astron. Soc. **499**(3), 3295 (2020)
52. LIGO Scientific Collaboration, Virgo Collaboration: LIGO-Virgo strain data from GWTC-3 Catalog (2021). https://doi.org/10.7935/B024-1886. https://www.gw-openscience.org/GWTC-3
53. Zelenka, O.: MLGWSC-1 Submission (2022). https://github.com/ondrzel/ml-gw-search
54. Zelenka, O.: Applications of machine learning to gravitational waves. Ph.D. thesis, Friedrich-Schiller-Universität Jena (2023). https://www.db-thueringen.de/receive/dbt_mods_00059058

Chapter 20
Deep Learning Methods for Accelerating Gravitational Wave Surrogate Modeling

Paraskevi Nousi, Styliani-Christina Fragkouli, Nikolaos Stergioulas, and Anastasios Tefas

Abstract We explore the application of deep learning techniques to accelerate gravitational wave surrogate modeling. We focus on two recent approaches, using artificial neural networks (ANNs) with residual error modeling and autoencoder-driven spiral representation learning. For the ANN method, we demonstrate that adding a second network to learn residual errors significantly improves surrogate model accuracy. The autoencoder approach reveals an inherent spiral structure in the latent space representation of empirical interpolation coefficients. We take advantage of this insight to develop a neural spiral module that can be integrated into network architectures to accelerate training and improve performance. Comprehensive evaluations show that these methods achieve state-of-the-art accuracy while enabling faster waveform generation. The techniques presented have the potential to substantially accelerate gravitational wave data analysis as detector sensitivity improves and event rates increase.

P. Nousi (✉)
Swiss Data Science Center, ETH, Zürich, Switzerland
e-mail: pnousi@ethz.ch

S.-C. Fragkouli
Institute Of Applied Biosciences, Centre for Research and Technology Hellas, Thessaloniki, Greece
e-mail: sfragkoul@certh.gr

N. Stergioulas
Department of Physics, Aristotle University of Thessaloniki, Thessaloniki, Greece
e-mail: niksterg@auth.gr

A. Tefas
Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece
e-mail: tefas@csd.auth.gr

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1_20

20.1 Introduction

Since 2015, when the first gravitational waves (GWs) from a binary black hole (BBH) system were detected [1], GW detections have become increasingly common, moving closer to the point of being a regular occurrence. After the third observing run (O3), the most recent catalog (GWTC-3, [2]) from the Advanced LIGO [3], Advanced Virgo [4] KAGRA [5, 6] collaboration contained 90 GW events, almost all of which were BBH mergers. The 4th observing run (O4) is currently underway and a larger number of BBH detections are expected [7]. The addition of a fifth interferometer, LIGO-India [8], is expected to significantly enhance both the sensitivity and the sky localization of the network. Moreover, third-generation ground-based detectors such as the Einstein Telescope [9, 10] and Cosmic Explorer [11, 12] are currently being developed and are anticipated to greatly expand our understanding of the astrophysical processes in the Universe [13–15].

The advances in GW astronomy described above were made possible by collaborative efforts in multiple areas. Accurate descriptions of the entire coalescence, including the full inspiral, merger, and ringdown, can be obtained in different ways, with `IMRPhenomXPHM` [16] and `SEOBNRv5PHM` [17] being two examples of waveform models. However, the increased complexity of the waveforms increases their computational cost.

In recent years, there has been an increase in the utilization of machine learning approaches for the analysis of gravitational wave data (see [18–20] for reviews). This chapter provides a summary of deep learning applications [21, 22] to accelerate the construction of surrogate models for gravitational-wave astronomy.

20.2 ANN-Accelerated Surrogate Models

Surrogate modeling has been provided to reduce the considerable computational cost of evaluating waveform models [23, 24], which can significantly speed up EOB waveforms (e.g. [23, 25–28]) while still providing high accuracy within its valid parameter range. The SEOBNRv4 model has a three-dimensional parameter space $\boldsymbol{\lambda}$; the mass ratio q between the two black holes and their spins χ_1 and χ_2, assuming that they are aligned with the orbital angular momentum. A surrogate model for this waveform family was presented in [29]. Several machine learning techniques can be used to interpolate or fit the projection coefficients of a reduced basis representation of time-domain waveforms, and the most suitable method depends on the desired accuracy and dimensionality. For low-dimensional parameter spaces, interpolation is a viable option. However, as the dimensionality increases, interpolation becomes difficult due to the large number of data points usually needed. Artificial Neural Networks (ANNs) are proposed as a solution to estimate these coefficients since this approach allows for efficient execution on either a CPU or GPU.

In [21], it was observed that the residual errors after training an ANN to evaluate the coefficients of the surrogate model for the SEOBNRv4 model had a pattern with respect to the input parameters. It was then demonstrated that a second neural network could be trained to model these errors, leading to an improved method, in which the maximum mismatch between SEOBNRv4 waveforms and waveforms generated by the new surrogate model was more than one order of magnitude smaller than the baseline method. Here, we will provide a summary of the steps taken to create the surrogate model and the residual ANN network to accelerate the evaluation of its coefficients, as described in [21].

20.2.1 Constructing a Surrogate Model

We express the complex gravitational wave strain as $h(t;\boldsymbol{\lambda}) = h_+(t;\boldsymbol{\lambda}) - ih_\times(t;\boldsymbol{\lambda})$, where h_+ and $h_\times$ are the two independent polarizations [30], t is the time, and $\boldsymbol{\lambda}$ is a vector of intrinsic parameters. The SEOBNRv4 model [31] has a three-dimensional parameter space, with each waveform characterized by the mass ratio q (the ratio of the masses of the two black holes) and the dimensionless spins χ_1, χ_2 of the two black holes. Surrogate modeling is a process of approximating given signals using a reduced model, denoted $h_s(t;\boldsymbol{\lambda})$, such that the approximation given by the surrogate model, $h_s(t;\boldsymbol{\lambda})$, accurately reconstructs the actual waveform $h(t;\boldsymbol{\lambda})$ within a preset threshold of error. When considering only the dominant, quadrupole ($l = m = 2$) mode [30], the target becomes $h_s(t;\boldsymbol{\lambda}) \approx h_{2,2}(t;\boldsymbol{\lambda})$ where l, m are the spherical harmonics. To begin the surrogate modeling process, a training set of N waveforms $\{h_i(t;\boldsymbol{\lambda}_i)\}_{i=1}^N$ is created, where $\boldsymbol{\lambda}_i = (q, \chi_1, \chi_2)_i$. The mass ratio is limited to a predetermined interval, such as $1 \le q \le 8$, within which the surrogate model is designed to be accurate. The two spins can have values in the range $-0.99 \le \chi_{1,2} \le 0.99$.

A *Reduced Order Method* (ROM) basis is constructed from a training set using a greedy algorithm [23]. This is an iterative process that selects $n < N$ waveforms (and their corresponding $\{\boldsymbol{\lambda}_j\}_{j=1}^n$ values, the greedy points) that, after orthonormalization, form the reduced basis $\{e_j\}_{j=1}^n$. Each $\boldsymbol{\lambda}_i$ waveform in the training set is then expressed as a linear combination

$$h(t;\boldsymbol{\lambda}_i) \approx \sum_{j=1}^{n} c_j(\boldsymbol{\lambda}_i) e_j(t), \tag{20.1}$$

within a given error tolerance, where $\{c_j(\boldsymbol{\lambda}_i)\}_{j=1}^n = \langle h(t;\boldsymbol{\lambda}_i), e_j(t)\rangle$ are the orthogonal projection coefficients.

Next, a new *Empirical Interpolation Method* (EIM) basis $B_k(t)$ is obtained such that a waveform $h(t;\boldsymbol{\lambda}_j)$ can be expressed as a linear combination of the basis, i.e. $h\left(t;\boldsymbol{\lambda}_j\right) = \sum_{k=1}^n \alpha_k(\boldsymbol{\lambda}_j) B_k(t)$. The coefficients $\alpha_k(\boldsymbol{\lambda}_j)$ are equal to the waveform at particular times, $\{T_k\}_{k=1}^n$, known as the empirical time nodes, i.e. $\alpha_k(\boldsymbol{\lambda}_j) = h\left(T_k;\boldsymbol{\lambda}_j\right)$. For any other waveform $h\left(t;\boldsymbol{\lambda}_i\right)$ in the training set, the coefficients of the EIM representation are $\alpha_k(\boldsymbol{\lambda}_i) = h\left(T_k;\boldsymbol{\lambda}_i\right)$. This does not require the basis $B_k(t)$, so the

coefficients can be computed much faster than the projection coefficients in the ROM basis (which require the projection of the whole waveform).

In the end, a surrogate model is created by interpolating the coefficient matrix $\alpha_k(\boldsymbol{\lambda}_i)$ of the training set to find the coefficients $\hat{\alpha}_k(\boldsymbol{\lambda})$ for any $\boldsymbol{\lambda}$, such that

$$h\,(t;\boldsymbol{\lambda}) \approx \sum_{k=1}^{m} \hat{\alpha}_k(\boldsymbol{\lambda}) B_k(t). \tag{20.2}$$

The complexity of this process increases with the number of parameters in $\boldsymbol{\lambda}$. *Neural networks can be used to speed up this part of the process, as demonstrated in* [29].

In practice, the complex waveform can be expressed in terms of its *amplitude* A and *phase* ϕ, defined through

$$h_+(t;\boldsymbol{\lambda}) - h_\times(t;\boldsymbol{\lambda}) = A(t;\boldsymbol{\lambda}) e^{-i\phi(t;\boldsymbol{\lambda})}, \tag{20.3}$$

which leads to a more compact EIM basis. To construct the ROM and EIM bases, a training set of $N = 2 \times 10^5$ waveforms was randomly sampled in the parameter space of $1 \leq q \leq 8$, $-0.99 \leq \chi_{1,2} \leq 0.99$. The waveforms were aligned in amplitude and initial phase, the phase was unwrapped, and the time series was truncated to a common starting time of $-20000M$, with a total mass of $M = 60M_\odot$. This ensured that all waveforms began with a minimum frequency no larger than 15 Hz, and $100M$ of post-peak ringdown data was kept. The ROM and EIM bases were created using `RomPy` [23, 32]. To evaluate the accuracy of the reconstructed waveforms (after the training is completed), a *validation set* of 3×10^4 SEOBNRv4 waveforms (not included in the training set) was used.

For two waveforms with parameters $\boldsymbol{\lambda}_1$ and $\boldsymbol{\lambda}_2$, the inner product can be defined [33]

$$\langle h(\cdot;\boldsymbol{\lambda}_1), h(\cdot;\boldsymbol{\lambda}_2)\rangle = 4\Re \int_{f_{min}}^{f_{max}} \frac{\tilde{h}(f;\boldsymbol{\lambda}_1)\tilde{h}^*(f;\boldsymbol{\lambda}_2)}{S_n(f)} df, \tag{20.4}$$

where $\tilde{h}(f;\boldsymbol{\lambda})$ is the Fourier transform of $h(t;\boldsymbol{\lambda})$, $S_n(f)$ denotes the noise power spectral density (PSD) of the GW detector and the star notation stands for the complex conjugate. The inner product can be employed to normalize the Fourier transform of a waveform in the following manner:

$$\hat{h}(f;\boldsymbol{\lambda}) = \frac{\tilde{h}(f;\boldsymbol{\lambda})}{\langle h(\cdot;\boldsymbol{\lambda}), h(\cdot;\boldsymbol{\lambda})\rangle}, \tag{20.5}$$

Then, the *overlap* between two waveforms is defined as the inner product between normalised waveforms $\hat{h}(\cdot;\boldsymbol{\lambda}_1)$, $\hat{h}(\cdot;\boldsymbol{\lambda}_2)$, maximised over a relative time (t_0) and phase (ϕ_0) shift between the two waveforms:

$$O(\hat{h}(\cdot;\boldsymbol{\lambda}_1), \hat{h}(\cdot;\boldsymbol{\lambda}_2)) = \max_{t_0,\phi_0} \langle h(\cdot;\boldsymbol{\lambda}_1), h(\cdot;\boldsymbol{\lambda}_2)\rangle, \tag{20.6}$$

and, finally the *mismatch* is given by

$$\mathcal{M}(\hat{h}(\cdot;\boldsymbol{\lambda}_1),\hat{h}(\cdot;\boldsymbol{\lambda}_2)) = 1 - \mathcal{O}(\hat{h}(\cdot;\boldsymbol{\lambda}_1),\hat{h}(\cdot;\boldsymbol{\lambda}_2)). \tag{20.7}$$

The performance of the surrogate model can be evaluated by comparing the waveforms generated by the SEOBNRv4 model with the predictions of the surrogate, using the mismatch defined above.

20.2.2 Accelerating the Surrogate Model Using ANNs

To construct the surrogate model, an ANN was employed to interpolate the coefficients $\alpha_k(\boldsymbol{\lambda}_i)$ of the training set to find the coefficients $\hat{\alpha}_k(\boldsymbol{\lambda})$ for an arbitrary $\boldsymbol{\lambda}$. The improved model was compared with a baseline model that followed the architecture of [29]. The ANN had four hidden layers with 320 neurons in each. The batch size was 10^3 and the training lasted for 10^3 epochs. The Adam optimizer [34] with a learning rate of 10^{-3} and the ReLU activation function [35] were used for the amplitude network. For the phase network, the Adamax [34] optimizer with a learning rate of 10^{-2} and the softplus activation function [36] were employed. Preprocessing involved using $\log(q)$ as input instead of q, which was then scaled using the `StandardScaler` from `Scikit-Learn` [37]. At the output, the coefficients were used raw for the amplitude network and were scaled using `Scikit-Learn`'s `MinMaxScaler` for the phase network.

The ANN prediction of the EIM coefficients of the training set waveforms will be referred to as $\hat{\boldsymbol{y}}_i \equiv \{\hat{\alpha}_k(\boldsymbol{\lambda}_i)\}_{k=1}^n$. During training, the standard mean square error

$$MSE = \frac{1}{N}\sum_{i=1}^{N} \|\hat{\boldsymbol{y}}_i - \boldsymbol{y}_i\|_2^2 \tag{20.8}$$

was measured and minimized, where the $\|\cdot\|_2$ notation represents the Euclidean norm of a vector. The MSEs were in the range $\sim 10^{-8} - 10^{-7}$.

A second ANN was created to predict the residual errors after establishing the baseline ANN surrogate model. This was done due to the presence of structure in the residuals for some EIM coefficients, as seen in Fig. 20.1. The final predictions are the sum of the outputs of the two models. For all $\{\boldsymbol{\lambda}_i\}_{i=1}^N$ in the training set, one can obtain the corresponding predictions $\{\hat{\boldsymbol{y}}(\boldsymbol{\lambda}_i)\}_{i=1}^N$ and calculate the *residual*

$$\boldsymbol{e}_i \equiv \boldsymbol{y}(\boldsymbol{\lambda}_i) - \hat{\boldsymbol{y}}(\boldsymbol{\lambda}_i), \tag{20.9}$$

where, as already defined, $\boldsymbol{y}$ is the ground truth. The second network was created with the *same input and architecture as the first network*, but this time it was trained on the residuals $\boldsymbol{e}_i$ (which were first scaled using the "MinMaxScaler" from `scikit-learn` [37]) to make predictions for the residual $\hat{\boldsymbol{e}}(\boldsymbol{\lambda})$ at any $\boldsymbol{\lambda}$. When

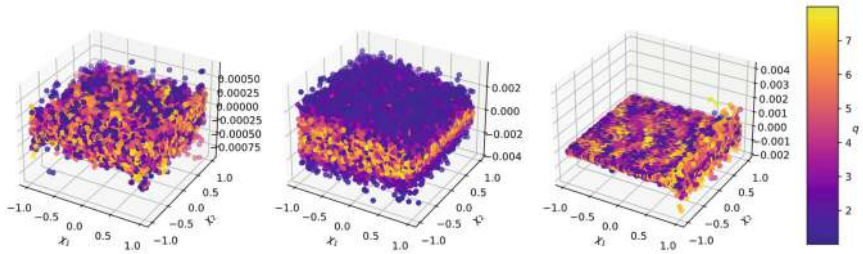


Fig. 20.1 The residual error for three chosen EIM coefficients for the amplitude is dependent on the input parameters $\boldsymbol{\lambda} = \{\chi_1, \chi_2, q\}$ (the dependence on q is illustrated with a colormap). The example in the left panel shows an unstructured distribution of residuals. In contrast, the example in the center panel reveals a strong dependence on the mass ratio q, while the example on the right displays a large residual error at the highest value of χ_1. Figure from [21]

the prediction $\hat{\boldsymbol{e}}$ for the residual is added to the prediction $\hat{\boldsymbol{y}}$ of the first network, an improved prediction is obtained.

$$\tilde{\boldsymbol{y}} \equiv \hat{\boldsymbol{y}} + \hat{\boldsymbol{e}}. \tag{20.10}$$

Figure 20.2 illustrates the difference in mismatches (for the validation set) between the baseline network and the case where a second network is added that models the residual error, as a violin plot. The median is marked by the middle horizontal line, whereas the minimum and maximum values are shown by the extent of the lines. The envelope of each panel is proportional to the density of points. The results in Fig. 20.2 demonstrate that adding a second network to learn the residual errors is beneficial for constructing surrogate models for gravitational waves from BBH inspiral. This strategy is likely to be advantageous for other types of GW template banks, such as binary neutron star inspiral waveforms.

20.3 Efficient Surrogate Models Using Autoencoders

Autoencoders (AEs) are a type of unsupervised neural network that is trained to reproduce its input by first transforming it into a lower-dimensional representation [38]. Generally, an autoencoder consists of an encoding component that maps the input to a compressed representation and a decoding component that reconstructs the input. Encoding and decoding functions can have symmetrical or asymmetrical architectures and usually comprise multiple layers of fully connected layers, convolutional layers, or recurrent modules. AEs have been studied for a variety of tasks, such as clustering [39, 40], classification [41, 42], and image retrieval [43, 44], due to their ability to extract semantically meaningful representations without labels. A typical AE architecture is shown in Fig. 20.3, with the input and output layers having the same number of neurons.

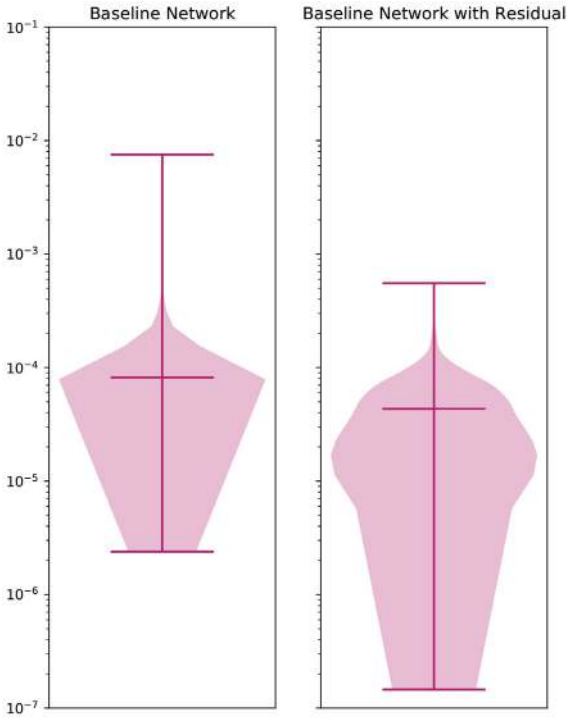


Fig. 20.2 A comparison of the mismatches (for the validation set) between the baseline network and the case when a second network that models the residual error is added is shown in the violin plots. The median is marked by the middle horizontal line, while the minimum and maximum values are indicated by the extent of the lines. The envelope of each panel is proportional to the density of points, and it is clear that a significant reduction of the mismatch is achieved when the network for the residual error is added. Figure from [21]

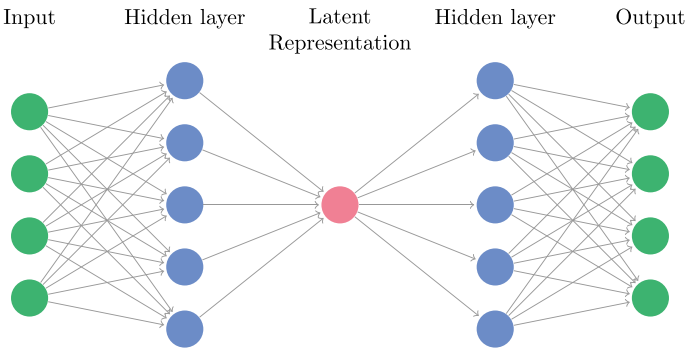


Fig. 20.3 Single hidden layer architecture of a fully connected Autoencoder. Figure from [22]

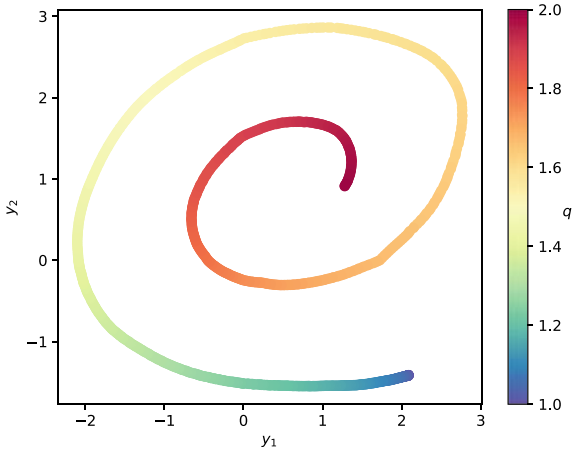


Fig. 20.4 Hidden representation uncovered by the AE for the empirical interpolation coefficients of a surrogate model of EOBNRv2 waveforms that is valid for $1 \leq q \leq 2$ (the color bar describes different values of the mass ratio q). When the values at the two neurons of the hidden representation are plotted against each other, a spiral structure emerges, along which the mass ratio appears to vary linearly with angle. Figure from [22]

In [22] a dataset comprising pairs of mass ratio q_i and corresponding EIM coefficient $\mathbf{a}_i$ ($i = 1, ..., N$), created with the EOBNRv2 nonspinning waveform model [45], was used to train an AE, with only the coefficients as input. This unsupervised process revealed a hidden relationship between each mass ratio q_i and the corresponding coefficients, as the mass ratios were unknown to the AE. Specifically, when choosing a two-dimensional intermediate representation, a spiral pattern emerged when visualizing this representation as a function of the mass ratio q, see Fig. 20.4. Below, we summarize the main steps presented in [22] to add a learnable spiral module to the ANN.

Following [23], a dataset of $N = 1000$ waveforms with mass ratios in the range $1 \leq q \leq 2$ was generated and a surrogate model was built, with a tolerance of 10^{-10}, resulting in a reduced basis of size $n = 11$. Next, a simple symmetric encoder-decoder AE architecture was used, with a two-dimensional hidden representation and two hidden fully-connected layers of 128 neurons on either side. The PReLU non-linearity [46] was used in all layers. The model was built using the PyTorch Deep Learning framework [47]. The EIM coefficients were used as input and output for this network. The AE was trained for 100 epochs with an initial learning rate of 0.001 and a batch size of 32. A multi-step multiplicative schedule was used with a gamma value of 0.9 and a step size of 15. The visual representation of the hidden layer is shown in Fig. 20.4, with the colors indicating the q values for each input coefficient. The spiral manifold in the hidden layer appears to describe a linear relationship between q and the angle θ of the spiral. The mean squared error of the reconstruction is 6.82×10^{-5}.

Based on the spiral pattern that emerged in Fig. 20.4, a neural spiral module was proposed in [22], which first transforms the input q into an angle θ, defined as

$$\theta := w \cdot q + b, \tag{20.11}$$

and subsequently maps θ onto a spiral structure of the form

$$\begin{aligned} s_x &:= (\alpha + \beta \cdot \theta) \cdot \cos\theta, \\ s_y &:= (\alpha + \beta \cdot \theta) \cdot \sin\theta, \end{aligned} \tag{20.12}$$

where w, b, α and β are parameters. These parameters are learnable, since the output is differentiable with respect to each of them. The spiral is fed to multiple, successive fully-connected layers, each with a nonlinear activation function, before reaching the final linear layer. An example of this architecture with two hidden layers is illustrated in Fig. 20.5. The inclusion of this module into an ANN accelerates the training process, leading to a significant reduction of the lowest achieved MSE.

The performance of various neural network architectures with fully-connected layers was assessed with and without the spiral module. The metrics used for evaluation were the waveform mismatch, inference speed, and memory requirements, with the maximum batch size that can be processed in a single forward pass on an NVIDIA RTX 2080 Ti GPU. All networks were trained for 2500 epochs with a batch size of 16, using the Adam optimizer [48] and an initial learning rate of 0.001, which was reduced by 0.95 every 150 epochs.

The inclusion of the spiral module significantly improved the mismatch achieved. When only one hidden layer was used, the baseline network with 128 neurons produced waveforms with a very poor mismatch (1.03×10^{-1} median mismatch). However, with the addition of the spiral module, even with only 32 hidden neurons, the median mismatch decreased by about 6 orders of magnitude. The best median and 95th percentile mismatch (9.41×10^{-9} and 3.48×10^{-8}) was achieved by the $\mathcal{S}$-32-64-128-64 network, which was able to generate up to 3.4 million coefficients in a single forward pass on the aforementioned GPU.

Finally, a spiral module was added to a neural network that was trained on a larger dataset of $N = 56000$ waveforms with $1 \leq q \leq 8$, where the q values were

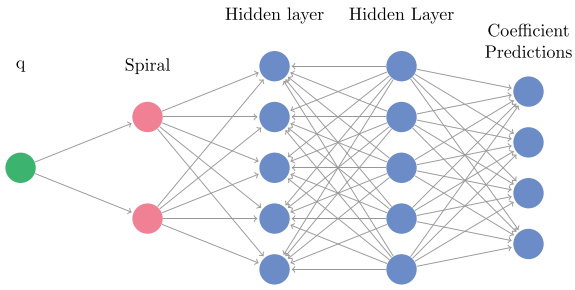


Fig. 20.5 Fully connected neural network with two hidden layers and the included spiral module. Figure from [22]

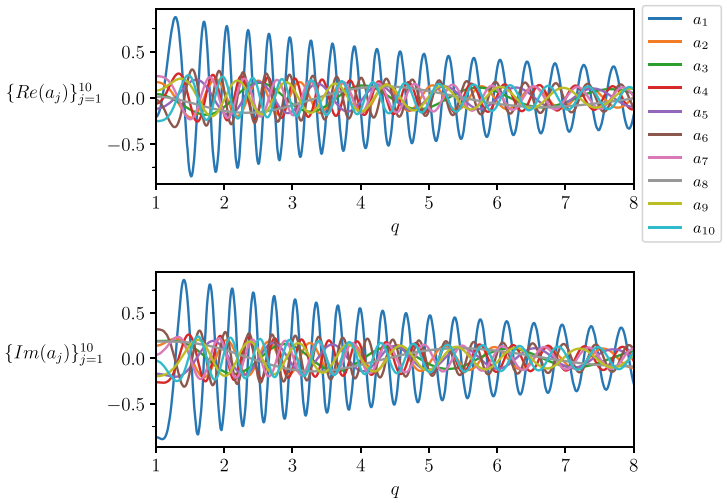


Fig. 20.6 Real (top) and imaginary parts (bottom) of the empirical interpolation coefficients $a_j(q)$ for a surrogate model of EOBNRv2 waveforms that is valid for $1 \leq q \leq 8$. Figure from [22]

equidistant. A validation and a test set were also created, each with 14000 waveforms, and the q values were randomly chosen in the range of $1 \leq q \leq 8$. Figure 20.6 shows the real and imaginary parts of the first ten coefficients of the EIM basis, $\{a_j(q)\}_{j=1}^{10}$. Despite some modulation of amplitude, each coefficient has a sinusoidal dependence with q (except near $q = 1$, where $dq/da_j = 0$ for all j).

Several neural networks were trained and tested on the dataset. All networks were trained for 5000 epochs, with a batch size of 32, and the Adam optimizer [48] with an initial learning rate of 0.001, which was reduced by 0.9 every 30 epochs. The training and validation loss per epoch for the $32 - 64 - 128 - 64$ network and the corresponding architecture with the addition of the spiral is shown in Fig. 20.7. The spiral addition resulted in a lower mean squared error, allowing smaller networks to achieve the same accuracy as larger networks, leading to a larger batch size that can be processed in a single forward pass when using a specific GPU card. For a different application of (variational) auto-encoders in GW astronomy, see [49].

20.4 Conclusions

We reviewed the efficacy of novel deep learning approaches for accelerating gravitational wave surrogate modeling. The residual error modeling technique using a second neural network provides a substantial improvement in the accuracy of surrogate models, as evidenced by the significant reduction in waveform mismatches. The autoencoder-driven spiral representation learning reveals intrinsic structure in the gravitational wave data that can be exploited to enhance model performance.

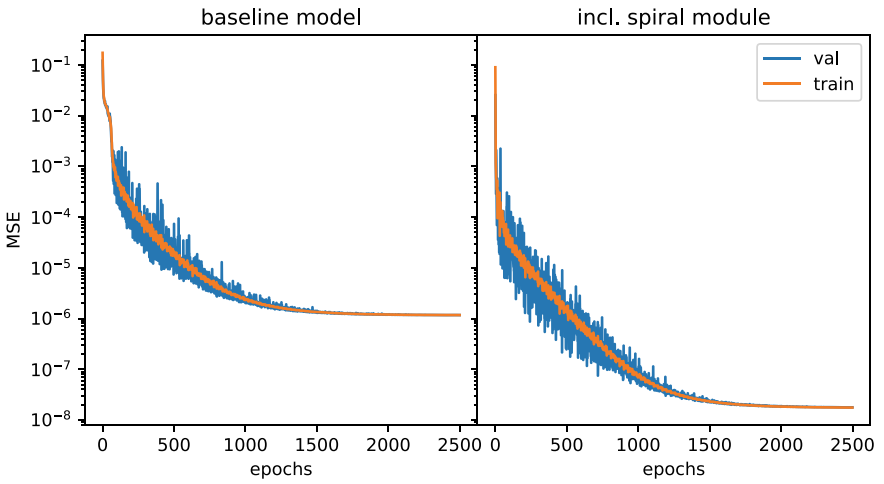


Fig. 20.7 Training and validation loss per epoch for the 32-64-128-64 network and the corresponding architecture with the addition of the spiral. Figure from [22]

The proposed neural spiral module leverages this structure to accelerate training and improve accuracy. Our comprehensive evaluations show that these methods not only achieve high accuracy, but also enable rapid waveform generation, critical for future gravitational wave data analysis. As gravitational wave astronomy advances with more sensitive detectors and higher event rates, these machine learning techniques offer a promising avenue for managing the increasing computational demands of waveform modeling and data analysis. Future work should focus on extending these methods to higher-dimensional parameter spaces and more complex waveform models, as well as exploring their application to real-time gravitational wave detection and parameter estimation.

Acknowledgements We are grateful to our collaborators, Panagiotis Iosif and Nikolaos Passalis, for their contributions that led to the main publication summarized in this review. This work was carried out within the framework of the EU COST action No. CA17137 (G2Net). NS acknowledges funding by the European Union through the Horizon Project 101131928 (ACME). This research has made use of data or software obtained from the Gravitational Wave Open Science Center (gwosc.org), a service of the LIGO Scientific Collaboration, the Virgo Collaboration, and KAGRA. This material is based upon work supported by NSF's LIGO Laboratory which is a major facility fully funded by the National Science Foundation, as well as the Science and Technology Facilities Council (STFC) of the United Kingdom, the Max-Planck-Society (MPS), and the State of Niedersachsen/Germany for support of the construction of Advanced LIGO and construction and operation of the GEO600 detector. Additional support for Advanced LIGO was provided by the Australian Research Council. Virgo is funded, through the European Gravitational Observatory (EGO), by the French Centre National de Recherche Scientifique (CNRS), the Italian Istituto Nazionale di Fisica Nucleare (INFN) and the Dutch Nikhef, with contributions by institutions from Belgium, Germany, Greece, Hungary, Ireland, Japan, Monaco, Poland, Portugal, Spain. KAGRA

is supported by Ministry of Education, Culture, Sports, Science and Technology (MEXT), Japan Society for the Promotion of Science (JSPS) in Japan; National Research Foundation (NRF) and Ministry of Science and ICT (MSIT) in Korea; Academia Sinica (AS) and National Science and Technology Council (NSTC) in Taiwan.

References

1. Abbott, B.P., et al.: Observation of gravitational waves from a binary black hole merger. Phys. Rev. Lett. **116**(6), 061102 (2016)
2. Abbott, R. et al.: GWTC-3: Compact binary coalescences observed by LIGO and Virgo during the second part of the third observing run. arXiv:2111.03606, November 2021
3. Aasi, J., et al.: Advanced LIGO. Class. Quant. Grav. **32**, 074001 (2015)
4. Acernese, F., et al.: Advanced Virgo: a second-generation interferometric gravitational wave detector. Class. Quantum Gravity **32**(2), 024001 (2014)
5. Akutsu, T. et al.: KAGRA: 2.5 generation interferometric gravitational wave detector. Nat. Astron. **3**(1), 35–40 (2019)
6. Akutsu, T. et al.: Overview of KAGRA: Detector design and construction history. Prog. Theor. Exp. Phys. (5), 08 2020. 05A101 (2021)
7. Abbott, B.P., et al.: Prospects for observing and localizing gravitational-wave transients with Advanced LIGO. Advanced Virgo and KAGRA. Liv. Rev. Relativity **23**(1), 3 (2020)
8. Saleem, M., et al.: The science case for LIGO-India. Class. Quantum Gravity **39**(2), 025004 (2022)
9. Punturo, M., et al.: The Einstein telescope: a third-generation gravitational wave observatory. Class. Quantum Gravity **27**(19), 194002 (2010)
10. Maggiore, M. et al.: Science case for the Einstein telescope. J. Cosmol. Astroparticle Phys. **2020**(3), 050 (2020)
11. Reitze, D. et al.: Cosmic explorer: the U.S. contribution to gravitational-wave astronomy beyond LIGO. Bull. Am. Astron. Soc. **51**, 35 (2019)
12. Evans, M. et al.: A Horizon Study for Cosmic Explorer: Science, Observatories, and Community (2021). arXiv:2109.09882
13. Abbott, B.P., et al.: Exploring the sensitivity of next generation gravitational wave detectors. Class. Quantum Gravity **34**(4), 044001 (2017)
14. Reitze, D., Punturo, M., Couvares, P., Katsanevas, S., Kajita, T., Kalogera, V., Lueck, H., McClelland, D., Rowan, S., Sanders, G., Sathyaprakash, B.S., Shoemaker, D., van den Brand, J.: Expanding the Reach of Gravitational Wave Astronomy to the Edge of the Universe: The Gravitational-Wave International Committee Study Reports on Next Generation Ground-based Gravitational-Wave Observatories (2021). arXiv:2111.06986
15. Kalogera, V., Sathyaprakash, B.S., Bailes, M., Bizouard, M-A. et al.: The Next Generation Global Gravitational Wave Observatory: The Science Book (2021). arXiv:2111.06990
16. Pratten, G., García-Quirós, C., Colleoni, M., Ramos-Buades, A., Estellés, H., Mateu-Lucena, M., Jaume, R., Haney, M., Keitel, D., Thompson, J.E., Husa, S.: Computationally efficient models for the dominant and subdominant harmonic modes of processing binary black holes. Phys. Rev. D **103**, 104056 (2021)
17. Ramos-Buades, A., Buonanno, A., Estellés, H., Khalil, M., Mihaylov, D.P., Ossokine, S., Pompili, L., Shiferaw, M.: SEOBNRv5PHM: Next generation of accurate and efficient multipolar precessing-spin effective-one-body waveforms for binary black holes (2023). arXiv:2303.18046
18. Cuoco, E., Powell, J., Cavaglià, M., Ackley, K., Bejger, M., Chatterjee, C., Coughlin, M., Coughlin, S., Easter, P., Essick, R., Gabbard, H., Gebhard, T. Ghosh, S., Haegel, L., Iess, A., Keitel, D., Márka, Z., Márka, S., Morawski, F., Nguyen, T., Ormiston, R., Pürrer, M., Razzano,

M., Staats, K., Vajente, G., Williams, D.: Enhancing gravitational-wave science with machine learning. Mach. Learn.: Sci. Technol. **2**(1), 011002 (2020)
19. Benedetto, V., Gissi, F., Ciaparrone, G., Troiano, L.: Ai in gravitational wave analysis, an overview. Appl. Sci. **13**(17) (2023)
20. Zhao, T., Shi, R., Zhou, Y., Cao, Z., Ren, Z.: Dawning of a New Era in Gravitational Wave Data Analysis: Unveiling Cosmic Mysteries via Artificial Intelligence – A Systematic Review (2023). arXiv:2311.15585
21. Fragkouli, Styliani-Christina., Nousi, Paraskevi, Passalis, Nikolaos, Iosif, Panagiotis, Stergioulas, Nikolaos, Tefas, Anastasios: Deep residual error and bag-of-tricks learning for gravitational wave surrogate modeling. Appl. Soft Comput. **147**, 110746 (2023)
22. Nousi, P., Fragkouli, S-C., Passalis, N., Iosif, P., Apostolatos, T., Pappas, G., Stergioulas, N., Tefas, A.: Autoencoder-driven spiral representation learning for gravitational wave surrogate modelling. Neurocomputing **491**, 67–77 (2022)
23. Field, S.E., Galley, C.R., Hesthaven, J.S., Kaye, J., Tiglio, M.: Fast prediction and evaluation of gravitational waveforms using surrogate models. Phys. Rev. X **4**(3), 031006 (2014)
24. Tiglio, M., Villanueva, A.: Reduced Order and Surrogate Models for Gravitational Waves (2021). arXiv:2101.11608
25. Pürrer, M.: Frequency domain reduced order model of aligned-spin effective-one-body waveforms with generic mass ratios and spins. Phys. Rev. D **93**(6), 064041 (2016)
26. Lackey, B.D., Bernuzzi, S., Galley, C.R., Meidam, J., Van Den Broeck, C.: Effective-one-body waveforms for binary neutron stars using surrogate models. Phys. Rev. D **95**(10), 104036 (2017)
27. Lackey, B.D., Pürrer, M., Taracchini, A., Marsat, S.: Surrogate model for an aligned-spin effective-one-body waveform model of binary neutron star inspirals using Gaussian process regression. Phys. Rev. D **100**(2), 024002 (2019)
28. Yun, Q., Han, W-B., Zhong, X., Benavides-Gallego, C.A.: Surrogate model for gravitational waveforms of spin-aligned binary black holes with eccentricities. Phys. Rev. D **103**(12), 124053 (2021)
29. Khan, Sebastian, Green, Rhys: Gravitational-wave surrogate models powered by artificial neural networks. Phys. Rev. D **103**(6), 064015 (2021)
30. Maggiore, M.: Gravitational Waves Volume 1: Theory and Experiments. Oxford University Press (2008)
31. Bohé, A., Shao, L., Taracchini, A., Buonanno, A., Babak, S., Harry, I.W., Hinder, I., Ossokine, S., Pürrer, M., Raymond, V., Chu, T., Fong, H., Kumar, P., Pfeiffer, H.P., Boyle, M., Hemberger, D.A., Kidder, L.E., Lovelace, G., Scheel, M.A., Szilágyi, B.: Improved effective-one-body model of spinning, nonprecessing binary black holes for the era of gravitational-wave astrophysics with advanced detectors. Phys. Rev. D **95**(4), 044028 (2017)
32. Galley, C.R.: RomPy package (2020). https://bitbucket.org/chadgalley/rompy/
33. Cutler, C., Flanagan, E.E.: Gravitational waves from merging compact binaries: how accurately can one extract the binary's parameters from the inspiral waveform? Phys. Rev. D **49**(6), 2658–2697 (1994)
34. Diederik, P.: Kingma and Jimmy Ba. A method for stochastic optimization, Adam (2017)
35. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: ICML, pp. 807–814 (2010)
36. Zheng, H., Yang, Z., Liu, W., Liang, J., Li, Y.: Improving deep neural networks using softplus units. In: 2015 International Joint Conference on Neural Networks (IJCNN), pp. 1–4 (2015)
37. scikit learn. 6.3. preprocessing data. https://scikit-learn.org/stable/modules/preprocessing.html
38. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P-A.: Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th International Conference on Machine Learning, pp. 1096–1103 (2008)
39. Xie, J., Girshick, R., Farhadi, A.: Unsupervised deep embedding for clustering analysis. In: International Conference on Machine Learning, pp. 478–487 (2016)

40. Nousi, P., Tefas, A.: Self-supervised autoencoders for clustering and classification. Evolving Systems, pp. 1–14 (2018)
41. Nousi, Paraskevi, Tefas, Anastasios: Deep learning algorithms for discriminant autoencoding. Neurocomputing **266**, 325–335 (2017)
42. Nousi, P., Tefas, A.: Discriminatively trained autoencoders for fast and accurate face recognition. In: International Conference on Engineering Applications of Neural Networks, pp. 205–215. Springer (2017)
43. Wu, P., Hoi, S.C.H., Xia, H., Zhao, P., Wang, D., Miao, C.: Online multimodal deep similarity learning with application to image retrieval. In: Proceedings of the 21st ACM international conference on Multimedia, pp. 153–162 (2013)
44. Carreira-Perpinán, M.A., Raziperchikolaei, R.: Hashing with binary autoencoders. In: Proceedings of the IEEE Conference On Computer Vision And Pattern Recognition, pp. 557–566 (2015)
45. Pan, Y., Buonanno, A., Boyle, M., Buchman, L.T., Kidder, L.E., Pfeiffer, H.P., Scheel, M.A.: Inspiral-merger-ringdown multipolar waveforms of nonspinning black-hole binaries using the effective-one-body formalism. Phys. Rev. D **84**(12), 124052 (2011)
46. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference On Computer Vision, pp. 1026–1034 (2015)
47. Paszke, Adam, Gross, Sam, Chintala, Soumith, Chanan, Gregory, Yang, Edward, DeVito, Zachary: Zeming Lin. Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In NIPS-W, Alban Desmaison (2017)
48. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2014). arXiv:1412.6980
49. Guo, W., Williams, D., Heng, I.S., Gabbard, H., Bae, Y-B., Kang, G., Zhu, Z-H.: Mimicking mergers: mistaking black hole captures as mergers. MNRAS **516**(3), 3847–3860 (2022)

Appendix: Code and Software Repositories

We have tried to gather references to code and tutorials used in the various chapters of this book. Below you will find a list of code repositories containing code or tutorials

- CA17137 g2net github repo
- gengli documentation
- gengli repository
- Scalable Auto-Encoders for Gravitational Waves Detection from Time Series Data (Corizzo, Zdravevski).
- gengli tutorial
- AresGW repository

E. Cuoco (ed.), *Gravitational Wave Science with Machine Learning*, Springer Series in Astrophysics and Cosmology, https://doi.org/10.1007/978-981-96-1737-1