Deep Learning for Modelling and Separating Gravitational Wave Signals, Glitches, and Noise

Tom Dooney

DOI: https://doi.org/10.71583/20260226td
Printed by: proefschriftmaken.nl

Deep Learning for Modelling and Separating Gravitational Wave Signals, Glitches, and Noise

Proefschrift

ter verkrijging van de graad van doctor
aan de Open Universiteit
op gezag van de rector magnificus
prof. dr. Theo Bastiaens
ten overstaan van een door het
College voor promoties ingestelde commissie
in het openbaar te verdedigen

op *26 Feb 2026* te Heerlen
om *16.00* uur precies

door

Tom Dooney, MSc
geboren op 18 Juli 1995 te Dublin, Ireland

Promotores

Prof. dr. Stefano Bromuri	Open Universiteit
Prof. dr. Chris van den Broeck	Utrecht University / Nikhef

Copromotores

Dr. Daniel Tan	Open Universiteit
Dr. R. Lyana Curier	Open Universiteit

Leden beoordelingscommissie

Prof. dr. Harald Vranken	Open Universiteit
Prof. dr. Natasha Alechina	Open Universiteit
Prof. dr. Ik Siong Heng	University of Glasgow
Prof. dr. Elena Cuoco	University of Bologna

Dedicated to

Bastiaan Heeren

(28/5/1978–22/11/2024)

We are all in the gutter, but some of us are looking at the stars.

— Oscar Wilde, *Lady Windermere's Fan* (1892)

Abstract

Since the first direct detection of gravitational waves (GWs) in 2015, the LIGO–Virgo –KAGRA collaboration has observed hundreds of events, marking a new era in astrophysics. As detector sensitivity improves, the rate of GW detections continues to rise, enhancing the scientific output of the collaboration. However, a key challenge is the presence of non-Gaussian noise transients in GW detectors, called glitches, which can mimic or obscure genuine signals, degrading the parameter estimation of GW sources and increasing false alarms. If glitch rates remain unchanged, the growing number of detections will naturally lead to more instances of glitch–signal overlap.

Traditional glitch mitigation methods are often computationally intensive and struggle to scale with the growing volume of GW data. This challenge becomes more pressing with the advent of next-generation detectors–such as the Einstein Telescope and Cosmic Explorer–which are expected to detect hundreds of events per day. As a result, the need for fast, accurate, and scalable analysis methods is becoming increasingly critical. This thesis leverages deep learning as a fast, data-driven approach to glitch mitigation and signal reconstruction, aiming to improve the accuracy and efficiency of GW data analysis.

This thesis presents two main contributions: (1) deep learning models for denoising and reconstructing GW signals and glitches, and (2) generative models for simulating realistic glitches and waveforms. We first introduce DeepExtractor, a framework that learns to subtract background noise and recover transient signals and glitches without templates. Extending the DeepExtractor approach across a multiple-detector network and incorporating signal models during training improves the estimation of GW source parameters in glitch-contaminated data by allowing for glitch removal even when they overlap with astrophysical events. To support data augmentation and simulation-based testing, we develop Derivative GAN (DVGAN) and its class-conditional variant cDVGAN, which incorporate a derivative-based discriminator to improve training stability and sample fidelity. We further advance cDVGAN to generate diverse, high-quality glitch samples spanning a wide range of realistic detector noise morphologies, all within a single user-controlled model. Together, these methods provide scalable and accurate alternatives to traditional pipelines, enhancing the robustness of GW data analysis as detectors become increasingly sensitive and their data more complex.

Samenvatting

Sinds de eerste directe detectie van zwaartekrachtsgolven (gravitational waves, GWs) in 2015 heeft de LIGO–Virgo–KAGRA-samenwerking honderden gebeurtenissen waargenomen, waarmee een nieuw tijdperk in de astrofysica is ingeluid. Naarmate de gevoeligheid van de detectoren toeneemt, blijft ook het aantal gedetecteerde zwaartekrachtsgolven groeien, wat leidt tot een steeds grotere wetenschappelijke opbrengst. Een belangrijke uitdaging hierbij is echter de aanwezigheid van niet-Gaussische, kortdurende ruisverstoringen in de detectoren, zogenoemde glitches. Deze verstoringen kunnen echte signalen nabootsen of verbergen, waardoor de schatting van eigenschappen van de bron verslechtert en het aantal valse alarmen toeneemt. Als het optreden van glitches niet afneemt, zal het groeiende aantal detecties onvermijdelijk leiden tot meer gevallen waarin glitches en zwaartekrachtsgolfsignalen elkaar overlappen.

Traditionele methoden voor glitch-mitigatie zijn vaak rekenintensief en schalen slecht met het toenemende datavolume. Dit probleem wordt nog urgenter met de komst van detectoren van de volgende generatie—zoals de Einstein Telescope en Cosmic Explorer—die naar verwachting honderden gebeurtenissen per dag zullen detecteren. Hierdoor wordt de behoefte aan snelle, nauwkeurige en schaalbare analysemethoden steeds groter. In dit proefschrift wordt deep learning ingezet als een snelle, datagedreven benadering voor glitch-mitigatie en signaalreconstructie, met als doel de nauwkeurigheid en efficiëntie van de analyse van zwaartekrachtsgolfdata te verbeteren.

Het proefschrift presenteert twee hoofdbijdragen: (1) deep-learningmodellen voor het onderdrukken van ruis en het reconstrueren van zwaartekrachtsgolfsignalen en glitches, en (2) generatieve modellen voor het simuleren van realistische glitches en golfvormen. Eerst introduceren we DeepExtractor, een raamwerk dat leert om achtergrondruis te modelleren en dit af te halen van het detectorsignaal, en zo tijdelijke signalen en glitches te reconstrueren zonder gebruik te maken van templates. Door deze aanpak uit te breiden naar een netwerk van meerdere detectoren en signaalmodellen op te nemen tijdens de training, wordt de schatting van eigenschappen van de bron verbeterd in data die door glitches is verstoord, zelfs wanneer glitches samenvallen met astrofysische gebeurtenissen. Ter ondersteuning van data-augmentatie en simulatiegebaseerde tests ontwikkelen we vervolgens de Derivative GAN (DVGAN) en de klasse-conditionele variant cDVGAN, waarin een discriminator op basis van afgeleiden wordt gebruikt om de trainingsstabiliteit en de kwaliteit van gegenereerde samples te verbeteren. Daarnaast breiden we cDVGAN verder uit om binnen één door de gebruiker controleerbaar model een grote diversiteit aan hoogwaardige glitches te genereren, die een breed scala aan realistische ruisvormen in detectoren bestrijken. Samen bieden deze methoden schaalbare en nauwkeurige alternatieven voor traditionele analysemethoden, en vergroten zij de robuustheid van zwaartekrachtsgolfanalyse naarmate detectoren gevoeliger worden en de data complexer.

Acknowledgments

This PhD represents four years of research, but it builds on a much longer journey shaped by the support and influence of many people. While it is impossible to thank everyone, I will do my best.

Firstly, I want to thank my parents for giving me every opportunity to succeed, both academically and personally; none of this would have been possible otherwise. I am especially grateful to my father, Ronan, for fostering my interest in physics from an early age, and to my mother, Anne-Marie, for guiding me towards data science after my bachelor's degree in theoretical physics. Together, your influence set me on the path that led to this PhD.

I would also like to thank my brothers, Jack and Jamie, and my sister, Alison, for always showing interest in and supporting my work.

I am deeply grateful to my promoter at Open Universiteit, Stefano, to whom I owe an enormous amount. He encouraged me to apply for the position, originally a two-year research role, and subsequently went to great lengths to transform it into a full four-year PhD. For this, I am deeply grateful, as well as for his patience whenever I needed someone to complain to.

Alongside Stefano, I would also like to thank my co-promoters at Open Universiteit, Daniel and Lyana, for their excellent supervision on the machine learning and AI aspects of my work. I also wish to thank the Dean of Open Universiteit, Petra, who was instrumental in securing the funding and support that made this PhD possible.

I would like to thank Bastiaan, who is no longer with us, for the warm welcome he gave me when I first moved to Utrecht. As Head of Computer Science at Open Universiteit at the time, he ensured a smooth transition and did his utmost to make me feel comfortable and supported. He was greatly appreciated by many, both personally and professionally. May he rest in peace.

My sincere thanks go to my promoter at Utrecht University's GRASP department, Chris, for agreeing to take me on as an external candidate and for providing invaluable guidance, as well as a productive environment at GRASP in which I could appropriately develop AI research in the context of gravitational-wave science. I am grateful to all of my supervisors across both institutions for granting me the freedom to explore my own ideas, while also offering critical direction when it mattered most.

I would like to thank Sarah, Professor at UMass Dartmouth, for her support of my research, particularly the generative modelling aspects of this thesis, as well as for supporting my travel to the United States for collaboration. That experience meant a great deal to me and is one I will not forget.

I am very grateful to Melissa and Antoni for taking me in when I first moved to Utrecht, for supporting me through multiple apartment moves, and for being there during some of the more difficult moments of the PhD. Your friendship and support mean a great deal to me. I also greatly enjoyed collaborating with Melissa on several

projects and value our work together. I would also like to thank Stefano S. (a different Stefano to the one mentioned above) for being such a good friend during my time in Utrecht and for teaching me so much about gravitational waves.

I want to thank Bart and Vivian for their friendship and support throughout my time in the Netherlands. I also want to thank Jiri for being a valued friend during the final year of the PhD and for being there through some challenging times towards the end.

I would like to thank my colleagues at GRASP: Tomek, Luca, Sumit, Mick, Fabian, Martin, Peter, Quirijn, Marc, Olaf, Johanna, Justus, Nick, Rik, Noor, Suzanne, Bas, Rene, Marcel, Alessandro, Marta, and Raimond for creating such a pleasant environment. It was a pleasure sharing many enjoyable lunch and coffee breaks with you all.

A special thanks goes to Thibeau for being such a good friend and travel buddy, and to Harsh, who taught me so much about gravitational waves and has been a great friend throughout. I have thoroughly enjoyed our collaboration and very much look forward to our future projects together.

I would also like to thank the staff at GRASP, Monique and Marie-Thérèse, for their warmth and their help with administrative matters and accommodation. I wish to thank Anuradha for her excellent advice throughout my time at GRASP, both regarding working within the collaboration and my research more broadly. I look forward to working together in the coming years.

I am very grateful to my friends in Portugal for making my time there so incredible. You know who you are, but in particular I would like to thank Diogo, Vasco, Florian, and Miguel. I cherish our time together, and our discussions on AI and data science which greatly shaped my perspective.

Although I have lived abroad for many years, I remain deeply grateful to the friends back home in Ireland who have always been there for me during my return visits. While there are too many to name, I would like to thank in particular Adam, Dan, David, Paul, Rory, Conor M., Conor O'L., Corey, Fitz, Ciarán, Emmanuele, and Nathan.

Finally, I express my deepest thanks to Sílvia for her love and support throughout this PhD journey. This work would not have been possible without you.

Contents

List of Figures

List of Tables

List of Abbreviations

ANN Artificial Neural Network
ASD Amplitude Spectral Density
AUC Area Under the Curve
BBH Binary Black Holes
BCE Binary Cross-Entropy
BNS Binary Neutron Star
CBC Compact Binary Coalesence
CCSN Core-Collapse Supernova
CE Cosmic Explorer
cGAN Conditional Generative Adversarial Network
CMB Cosmic Microwave Background
CNN Convolutional Neural Networks
cWB Coherent Wave Burst
DFT Discrete Fourier Transform
DiT Diffusion Transformer
DL Deep Learning
DNN Deep Neural Network
DWT Discrete Wavelet Transform
EOB Effective One Body
EM Electromagnetic
ET Einstein Telescope
FAP False Alarm Probability
FAR False Alarm Rate
FF Fitting Factor
FFT Fast Fourier Transform
FN False Negative
FP False Positive
FPR False Positive Rate
GAN Generative Adversarial Networks
GP Gradient Penalty
GPU Graphics Processing Unit
GRB Gamma-Ray Burst
GRU Gated Recurrent Unit
GW Gravitational Wave

GWOSC Gravitational-Wave Open Science Center
GWTC Gravitational-Wave Transient Catalog
H1 LIGO Hanford
iDQ Inferential Data Quality
IMBH Intermediate-Mass Black Hole
IMR Inspiral–Merger–Ringdown
JS Jensen–Shannon
KAGRA Kamioka Gravitational Wave Detector
KL Kullback–Leibler
L1 LIGO Livingston
LIGO Laser Interferometer Gravitational-wave Observatory
LISA Laser Interferometer Space Antenna
LSTM Long Short-Term Memory
LVK LIGO-Virgo-Kagra
MAE Mean Absolute Error
MCMC Markov Chain Monte Carlo
ML Machine Learning
MLP Multilayer Perceptron
MM Minimal Match
MSE Mean-Squared Error
NR Numerical Relativity
NS Neutron Star
NSBH Neutron star–Black hole
O1 First Observing Run
O2 Second Observing Run
O3 Third Observing Run
O4 Fourth Observing Run
Phenom Phenomenological
PN Post-Newtonian
PCA Principal Component Analysis
PSD Power Spectral Density
ReLU Rectified Linear Unit
RJMCMC Reversible-Jump Markov Chain Monte Carlo
ROC Receiver Operating Characteristic
RMSE Root-Mean-Squared Error
RNN Recurrent Neural Network
SGD Stochastic Gradient Descent
SNR Signal-to-Noise Ratio
STFT Short-Time Fourier Transform
SVM Support Vector Machine
TN True Negative

TP True Positive
TPR True Positive Rate
TT Transverse–Traceless
VAE Variational Autoencoder
WGAN Wasserstein Generative Adversarial Networks
WDM Wilson–Daubechies–Meyer

Introduction

In front of my apartment in Utrecht, a small inlet of the Vecht river is home to a flock of ducks. It rains often in the Netherlands, and whenever it does, the raindrops make ripples in the water and the ducks glide over them, not paying them any noticeable attention. The ripples can be described by properties such as their amplitude and frequency, which carry information about the raindrop that produced them such as the size, shape, mass and speed of the raindrop as it struck the water. If the ducks had an Einstein among them, capable of formulating a theory to predict the properties of those ripples from the properties of the raindrops, they might be able to infer something about the hidden world above the surface by measuring those ripples. But as far as we know, the ducks are still waiting for their Einstein.

Up until Einstein changed our perception of the universe in 1915 with his theory of General Relativity, we were like those ducks—unaware of the ripples constantly passing through us. General Relativity explained what Newton's theory of gravity could not—such as Mercury's strange orbit–and revealed that space and time, which Einstein unified as *spacetime*, are curved by mass. Even though we can't see this curvature, it's all around us: the Earth bends it. Even our bodies bend it, although our contribution to the curvature is infinitesimally small. Einstein also predicted that massive objects could send ripples through spacetime, which he called gravitational waves (GWs), though he thought they'd be far too small to ever detect. It took a century for technology to prove him wrong. On the 14^{th} September 2015, the Laser Interferometer Gravitational-wave Observatory (LIGO), America's twin gravitational-wave (GW) detectors, finally caught one.

The GW event known as GW150914 originated from the merger of two black holes; objects so massive and compact that not even light can escape their gravitational pull. LIGO detected the final few orbits of these black holes as they spiraled ever closer together, completing their inspiral and finally merging to form a single, larger black hole.

The energy released in this cataclysmic event propagated outward as GWs, eventually reaching Earth. From the detected signal, we were able to infer several key properties of the system. The wave travelled approximately 1.3 billion light-years before arriving at the detectors and came from a region in the southern hemisphere. The two black holes were estimated to have masses of 36 and 29 times the mass of the sun, and their merger produced a black hole of about 62 solar masses. The missing mass (about three suns' worth) was converted into GW energy.

LIGO measures GWs using a conceptually simple but extraordinarily precise method. If spacetime stretches and compresses as a gavitational wave passes, then the lengths of two perpendicular laser arms will change ever so slightly. By comparing the returning light from these two directions, the detector can reveal these minute distortions in the form of a one-dimensional length difference. By the time they reach

Earth, these distortions correspond to length changes on the order of 10^{-18} m, about a thousandth the diameter of a proton over LIGO's four-kilometer arms. Since September 2015, the detector network of LIGO's Hanford and Livingston detectors, along with Italy's Virgo detector in Pisa, has detected over 300 GW signals.

The first GW detections came from merging black holes, but on the 17$^{\text{th}}$ August 2017, something entirely new appeared: a signal from two neutron stars merging together, known as GW170817. With both LIGO detectors and Italy's Virgo detector observing, the source could be triangulated, and an alert went out worldwide. Within seconds, NASA's Fermi telescope detected a gamma-ray burst from the same region, and soon telescopes across the globe—including Hubble, VLT, Subaru, and SALT—captured the optical afterglow of a kilonova. For the first time, we were seeing and "hearing" the same cosmic event. Thus began the era of multi-messenger astronomy.

Unlike black holes, neutron stars are made of real matter; mostly neutrons, the same particles that make up much of the mass around us. Observing their merger lets us probe how matter behaves under the most extreme conditions in the universe. GWs give us a new way to explore such events; not by seeing light, but by "listening" to the vibrations of spacetime itself. They open a window onto physics unreachable by electromagnetic observations alone, revealing phenomena that would otherwise remain completely invisible.

The way LIGO measures GWs—by tracking expansions and contractions of space along two perpendicular directions—is surprisingly familiar. We encounter analogous waves every day: sound waves[1]. Like GWs, sound spreads through space in all directions and can be represented neatly in one dimension by recording it digitally. Anyone who's used music software has seen sound waves visualized this way. While GW detectors measure changes in length over time, digital audio systems record variations in voltage over time. Despite their very different origins, the visual representations of these signals look strikingly similar, as shown in Figure 1. And just like audio data, GW data can be distorted by background noise—like the static that ruins your favorite song on the radio.

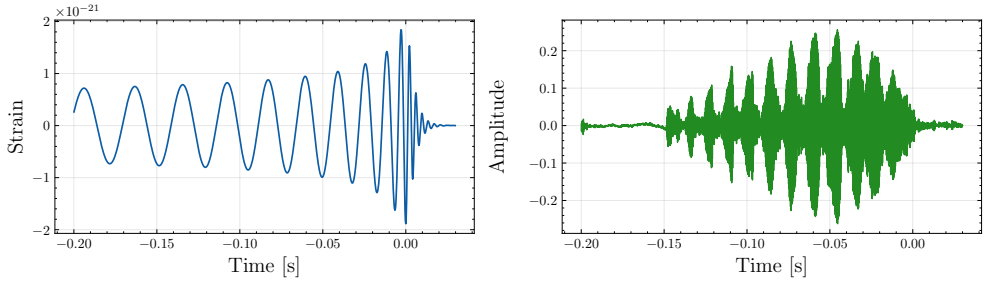


Figure 1: Comparison between a simulated GW signal from two colliding black holes (left) and a digital recording of a bird chirp (right), each showing a ∼0.2 s waveform segment. Despite arising from entirely different physical phenomena, both exhibit the characteristic frequency increase—or "chirp"—over time.

[1]While they may appear similar when visualized digitally, sound waves and GWs differ fundamentally in nature. Sound waves are longitudinal pressure waves, oscillating in the same direction as they propagate, whereas GWs are transverse, oscillating perpendicular to their direction of travel.

GW scientists are essentially trying to hear the purely cosmic sounds buried beneath the static of earthly noise. Since the detectors must be extraordinarily sensitive, they register a wide range of terrestrial disturbances; seismic rumbles, electrical hums, even traffic from nearby towns during rush hour. These earthly disturbances contaminate the signals we care about.

Occasionally, short and distinct bursts of noise appear in the data that sometimes look remarkably similar to real GWs. We call these artifacts "glitches", and they occur frequently, about one per minute. By coincidence, some glitches have even occurred at the exact moment a true GW passed through Earth, making it difficult to separate the signal from the noise. This was exactly what happened during the first binary neutron star event that we discussed above, GW170817: a glitch appeared in one of the LIGO detectors almost coincident with the merger, a crucial part of the signal for extracting meaningful physics. The glitch threatened to obscure one of the most important astrophysical discoveries of the decade and had to be carefully modelled and subtracted before any meaningful physics could be extracted from the data.

GW astronomy has long relied on traditional statistical methods to produce measurements precise enough to reveal the physics of our universe. These techniques have proven highly effective at extracting GW signals from noisy data, and they have been crucial in handling glitches, including the one that affected the 2017 neutron star event. However, this reliability comes at a cost: the analyses are computationally intensive and can take significant time to produce results. As detectors become more sensitive and data grows exponentially—especially with next-generation observatories like the Einstein Telescope and Cosmic Explorer—we will need new tools to keep up. This is where deep learning, or artificial intelligence, comes in.

Despite the hype around "AI", the models in this thesis aren't intelligent—at least, not in any human sense. They're sophisticated pattern recognizers, flexible enough to fit the complex structure of GW data, once you show ("train") them exactly what the patterns are they have to recognize. For example, in the case of music, a neural network can be trained to isolate and remove the vocal track from songs, regardless of the accompanying instruments. When trained on enough songs, it can then generalize this task to new songs it has not encountered before, enabling applications such as karaoke systems. In this sense, these models function like adaptable meshes that mold themselves to the patterns inherent in the data. When trained appropriately, they become remarkably powerful within their domain, and are typically far faster than the traditional techniques currently in use.

In essence, this thesis is about building deep learning algorithms and frameworks that can complement—and perhaps, with further development, eventually replace—many of the traditional methods used to analyze GW data. My primary focus is on leveraging the power of deep learning to model and mitigate glitches. To this end, the thesis addresses two broader goals.

First, we design and train specialized neural networks to separate GW signals, detector background noise, and glitches, accurately modelling each component, like separating instruments or vocals in a song. Achieving this could enable scientists to analyze data far more quickly than current methods allow, especially when signals and glitches occur simultaneously. We will also see how these approaches open up exciting possibilities for rapid GW detection, as well as for modelling more exotic types of signals expected from next-generation observatories. Second, we develop

new generative models capable of producing realistic GW signal and glitch samples that can be used in simulations to test and validate analysis pipelines. These models are conceptually similar to generative networks that create music from text prompts, which have become increasingly prevalent in the public domain. Such approaches offer promising new avenues for simulation-based studies and detector characterization in GW physics.

Ultimately, this thesis aims to demonstrate the potential of deep learning to advance the precision science of GW astrophysics, particularly by modelling and mitigating the effects of glitches on the measurements and inference we can extract from GW data.

Thesis contributions

This thesis advances GW data analysis by introducing deep learning frameworks that address two major challenges in current and future detector networks:

1. Model-agnostic reconstruction and mitigation of non-Gaussian transient noise artifacts (glitches) in real time.

2. Scalable and realistic generative modelling of GW data for simulation, validation, and machine learning–based pipelines.

Deep Learning for Signal and Glitch Reconstruction

The first major contribution is the development of DeepExtractor, a fast, model-agnostic reconstruction framework capable of isolating excess transient power from background noise in GW detector data. This contribution is based on Publication 2 in the list of selected publications below. Unlike traditional template-based methods, DeepExtractor learns to model and subtract the detector background noise, enabling the recovery of both astrophysical signals and non-Gaussian transients in a fully data-driven way.

Key innovations include:

- A Power Spectral Density (PSD)-informed U-Net architecture that operates on the magnitude and phase components of short-time Fourier transform (STFT) spectrograms, preserving both amplitude and phase information critical for time-domain reconstruction.

- A residual learning strategy that trains the network to estimate the noise component rather than the clean waveform, leading to improved generalization and robustness across unseen glitch morphologies.

- Real-time performance: in simulated experiments, DeepExtractor achieves superior reconstruction accuracy to the state-of-the-art BayesWave algorithm while delivering over 10,000× speedup, reconstructing a 2 s segment of data in just 0.1 s. This enables true online use during observing runs.

Applied to real LIGO data from its *third obvsering run* (O3), DeepExtractor successfully reconstructs a wide range of glitch classes with high fidelity, as validated by

the Gravity Spy classifier (90% accuracy) and unsupervised clustering analyses. It also reconstructs three real GW events during O3, showing good agreement with the maximum-likelihood templates for those events.

Building on this, the thesis extends DeepExtractor to jointly reconstruct signal, noise, and glitch components in multi-detector networks. By exploiting cross-detector coherence, DeepExtractor performs deterministic glitch–signal separation and represents an early step toward holistic glitch mitigation using deep learning; a task traditionally performed only by Bayesian techniques such as BayesWave.

Generative Modelling for Realistic Glitch Simulation

The second major contribution is the introduction of new generative deep learning frameworks capable of synthesizing realistic glitches and signal waveforms for simulation-based testing, data augmentation, and pipeline validation. This contribution is based on Publications 3 and 5 in the list of selected publications.

The thesis introduces the Derivative GAN (DVGAN), which integrates derivative-based feedback to stabilize adversarial learning and improve the physical realism of generated samples. DVGAN consistently outperforms baseline Wasserstein GANs (WGANs) in producing smooth, morphologically accurate glitches and waveforms.

Building on this, the Conditional Derivative GAN (cDVGAN) extends the architecture to a multi-class conditional setting, allowing explicit control over glitch type and enabling the generation of hybrid samples that interpolate between known classes. This flexibility enhances the diversity of training datasets for machine learning–based search pipelines. A series of ablation experiments demonstrate that derivative-informed adversarial feedback improves the quality and utility of generated data for downstream CNN-based detection tasks.

Evaluations using UMAP visualizations and Gravity Spy classification confirm that cDVGAN-generated glitches reproduce the morphological diversity of real detector data across some of the most prominent glitch classes observed in LIGO's detectors.

Finally, we demonstrate that cDVGAN can serve as a holistic time-domain glitch generator, capable of producing diverse distributions of glitch morphologies by learning DeepExtractor glitch reconstructions within a single, user-controlled model.

Summary of Impact

Collectively, the contributions of this thesis:

- Demonstrate that deep learning can achieve accurate, and real-time reconstruction of arbitrary GW signals and glitches from detector noise.
- Provide a framework for glitch mitigation in realistic detector noise.
- Introduce derivative-informed generative models capable of synthesizing morphologically diverse and physically consistent glitches.

Through these developments, the thesis presents practical, data-driven pathways toward faster and more resilient GW data analysis.

Thesis outline and author contribution

This thesis presents original research conducted by the author on the application of deep learning to GW data analysis, focusing on signal reconstruction, glitch mitigation, and realistic data generation for simulation-based studies. The work is structured into four parts:

Part I provides the theoretical and methodological background on GWs, detectors, data analysis and machine learning foundations. Part II introduces the DeepExtractor framework for signal and glitch reconstruction and presents initial results on glitch–signal-noise separation.

Part III presents the novel DVGAN and cDVGAN architectures for generative modelling of realistic glitch and signal data. Part IV presents the conclusions, future work and broader impact of this thesis.

Author's contributions. The author developed all of the core deep learning models presented in this thesis, including the DeepExtractor, DVGAN, and cDVGAN methods. This encompassed the full pipeline design, implementation, training, evaluation, and visualization of results. The author also curated, preprocessed, and generated all training and testing datasets used for model development, including real LIGO data, simulated Gaussian noise, and injected GW signals. The experiments comparing the proposed methods to traditional approaches were designed, executed, and interpreted by the author. In addition to the publications on which this thesis is primarily based, the author made minor contributions to Publications 1 and 4 in the list of selected publications, supporting the design and implementation of these methods.

A PhD student at Utrecht University, Harsh Narola, implemented BayesWave for the comparison in Chapter 4 against DeepExtractor. Parameter estimation results in Chapter 5 for selected events were also measured by Harsh. The author performed the comparative analysis between the parameter estimation outputs and the results obtained from the deep learning framework.

A Bachelor student at Utrecht University, Mees De Boer, constructed the training dataset used for the cDVGAN experiments in Chapter 8 by extracting them from real LIGO data using DeepExtractor. Supervised by the author, he also developed the diffusion (DiT) model, and conducted the Gravity Spy classification of DeepExtractor reconstructions in Chapter 4 and of the DiT-generated glitches in Chapter 8.

Selected Publications

1. H. Narola, T. Wouters, L. Negri, M. Lopez, T. Dooney, F. Cireddu, M. Wils, I. C. F. Wong, P. T. H. Pang, J. Janquart, A. Samajdar, C. Van Den Broeck, and T. G. F. Li, "Null-stream-based third-generation-ready glitch mitigation for gravitational wave measurements," *Physical Review D* **112**, 024079 (2025). doi:10.1103/PhysRevD.112.024079

2. T. Dooney, H. Narola, S. Bromuri, R. L. Curier, C. Van Den Broeck, S. Caudill, and D. S. Tan, "Time-domain reconstruction of signals and glitches in gravitational wave data with deep learning," *Physical Review D* **112**, 044022 (2025). doi:10.1103/PhysRevD.112.044022

3. T. Dooney, R. L. Curier, D. S. Tan, M. Lopez, C. Van Den Broeck, and S. Bro-

muri, "One flexible model for multiclass gravitational wave signal and glitch generation," *Physical Review D* **110**, 022004 (2024). doi:10.1103/PhysRevD.110.022004

4. P. Laguarta *et al.*, "Detection of anomalies amongst LIGO's glitch populations with autoencoders," *Classical and Quantum Gravity* (2024). doi:10.1088/1361-6382/ad1f26

5. T. Dooney, S. Bromuri, and L. Curier, "DVGAN: Stabilize Wasserstein GAN training for time-domain gravitational wave physics," in *Proceedings of the 2022 IEEE International Conference on Big Data (Big Data)*, pp. 5468–5477 (2022). doi:10.1109/BigData55660.2022.10021080

Part I

Background

Chapter 1

Gravitational Waves, their Sources, and Detectors

1.1 From Einstein's Curved Spacetime to Gravitational Waves

Although this thesis focuses on the development of applied deep learning methods for gravitational-wave (GW) data analysis, it is useful to briefly review the underlying physical theory. Einstein's *General Relativity* provides the foundation for our understanding of GWs, and a short introduction helps place the subsequent data-driven work in its proper scientific context. This section therefore offers a concise overview of Einstein's theory and shows how the prediction of GWs naturally emerges from it.

Readers already familiar with General Relativity—or those content to take Einstein's word for it (as most of us do)—may wish to skip ahead to the next section.

1.1.1 Essentials of General Relativity

Gravity, though the weakest of the four fundamental forces, is the one that shapes the universe on its largest scales. Einstein's theory of *General Relativity*, published in 1915, replaced Newton's concept of gravity as a force with a deeper idea: massive objects curve the geometry of spacetime, and this curvature tells matter how to move. Mathematically, this relationship is expressed by Einstein's field equations:

$$G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}. \tag{1.1}$$

Here, c is the speed of light, G is the Newtonian gravitational constant, $G_{\mu\nu}$ is the Einstein tensor, which describes the curvature of spacetime, and $T_{\mu\nu}$ is the stress-energy tensor, which represents the energy and momentum of matter and radiation. This deceptively compact equation encapsulates the essence of General Relativity: the geometry of spacetime and the distribution of matter and energy are inseparably linked. In the physicist John Archibald Wheeler's memorable words: *"Matter tells spacetime how to curve, and spacetime tells matter how to move."*

In curved spacetime, objects move not because they are pulled by a force, but because they follow the straightest possible paths within that curved geometry. These paths are called *geodesics*. In flat spacetime, a geodesic is simply a straight line; in curved spacetime, it appears as a curved path.[1] This interplay between geometry

[1] A familiar example is the trajectory of an airplane flying between two cities on Earth: although

and energy determines the structure and evolution of the universe, from the motion of planets to the dynamics of galaxies and the expansion of the cosmos itself.

To interpret this equation, it helps to unpack what these symbols represent. Both $G_{\mu\nu}$ and $T_{\mu\nu}$ are *rank-2 tensors*—mathematical objects with two indices that describe relationships among the four dimensions of spacetime (one of time and three of space). The indices μ and ν can each take values $0, 1, 2, 3$, corresponding to the spacetime coordinates

$$x^\mu = (x^0, x^1, x^2, x^3) = (t, x, y, z).$$

Each tensor can be represented as a symmetric 4×4 array of components, for example:

$$G_{\mu\nu} = \begin{pmatrix} G_{00} & G_{01} & G_{02} & G_{03} \\ G_{10} & G_{11} & G_{12} & G_{13} \\ G_{20} & G_{21} & G_{22} & G_{23} \\ G_{30} & G_{31} & G_{32} & G_{33} \end{pmatrix}, \qquad T_{\mu\nu} = \begin{pmatrix} T_{00} & T_{01} & T_{02} & T_{03} \\ T_{10} & T_{11} & T_{12} & T_{13} \\ T_{20} & T_{21} & T_{22} & T_{23} \\ T_{30} & T_{31} & T_{32} & T_{33} \end{pmatrix}.$$

Each component expresses how two spacetime dimensions interact. For instance, T_{00} gives the energy density, $T_{0\mu}$ describes the flux of momentum (or equivalently, the flow of energy) in the μ^{th} spatial direction, and $T_{\mu\nu}$ represents stresses or pressures acting along spatial directions. Likewise, $G_{\mu\nu}$ summarizes how spacetime itself curves in response to those energy and momentum distributions. We will see below how $G_{\mu\nu}$ is constructed from the underlying geometry of spacetime.

The components of a tensor change in a precise, rule-governed way when the coordinate system changes. This transformation property ensures that the physical relationships the tensor represents (such as curvature or energy flow) remain the same for all observers, no matter what coordinate system they are in.

In other words, the individual numerical components of a tensor depend on the chosen coordinates, but the *geometric object* they represent does not. The tensor itself is an intrinsic feature of spacetime; a coordinate-independent relationship between directions, lengths, and flows. Changing coordinates only alters how we describe this object, not what it fundamentally is. This property, known as *covariance*, is what makes tensors the natural language of general relativity: they express physical laws in a form that remains valid regardless of how spacetime is curved or how coordinates are defined.

The most fundamental tensor in general relativity is the *metric tensor* $g_{\mu\nu}$. It defines the geometric structure of spacetime: how distances, durations, and angles are measured. The Einstein tensor $G_{\mu\nu}$ itself is built from derivatives of the metric tensor $g_{\mu\nu}$ and therefore encapsulates how the geometry of spacetime responds to matter and energy.

Mathematically, $g_{\mu\nu}$ is also a symmetric 4×4 tensor whose components quantify how pairs of spacetime dimensions (time or one of the three spatial axes) relate to each other:

$$g_{\mu\nu} = \begin{pmatrix} g_{00} & g_{01} & g_{02} & g_{03} \\ g_{10} & g_{11} & g_{12} & g_{13} \\ g_{20} & g_{21} & g_{22} & g_{23} \\ g_{30} & g_{31} & g_{32} & g_{33} \end{pmatrix}, \qquad g_{\mu\nu} = g_{\nu\mu}.$$

the route appears curved on a flat map, it actually follows a straight path—the shortest distance—on the curved surface of the planet.

In the presence of mass or energy, the components $g_{\mu\nu}(x)$ vary from point to point, encoding how gravity curves the underlying geometry and thereby alters the measurement of distances, durations, and angles. In the absence of such mass or energy, spacetime is flat, and the metric reduces to the Minkowski form:

$$\eta_{\mu\nu} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

In this case, the infinitesimal spacetime interval between two nearby events is

$$ds^2 = -c^2dt^2 + dx^2 + dy^2 + dz^2. \tag{1.2}$$

Here the time difference dt is multiplied by the speed of light c to express temporal separations in the same units as spatial ones, allowing time and space to be treated on equal footing within a single geometric framework.

This expression is the spacetime analogue of the Pythagorean theorem. In Euclidean geometry, the distance between two nearby points is given by

$$dl^2 = dx^2 + dy^2 + dz^2,$$

which measures purely spatial separation. Relativity generalizes this idea by including time as a fourth dimension—but with an opposite sign to reflect its distinct physical role. The metric defines how distances and angles are measured in this four-dimensional spacetime: the interval ds^2 combines changes in both space and time into a single geometric quantity that remains the same for all observers.

The negative sign in the time component encodes the fundamental difference between temporal and spatial directions, ensuring that spacetime distinguishes between *timelike*, *spacelike*, and *lightlike* separations[2]. A separation is timelike ($ds^2 < 0$) if two events can be connected by a signal traveling slower than light—such as the motion of a massive particle. It is spacelike ($ds^2 > 0$) if the events are too far apart in space for any causal influence to pass between them. Finally, it is lightlike ($ds^2 = 0$) when the separation corresponds to something moving exactly at the speed of light, such as a photon.

We saw how the indices μ and ν each take values $0, 1, 2, 3$, corresponding to the four spacetime coordinates. Quantities with indices written upstairs, such as A^μ, are called *contravariant* components and describe how a vector points or changes in spacetime. Those with indices written downstairs, such as A_μ, are *covariant* components, which represent how a quantity projects onto the local geometry. The metric tensor $g_{\mu\nu}$ acts as a bridge between the two, allowing indices to be raised or lowered:

$$A_\mu = g_{\mu\nu}A^\nu, \qquad A^\mu = g^{\mu\nu}A_\nu,$$

where $g^{\mu\nu}$ is the inverse of $g_{\mu\nu}$.

Raising or lowering an index is not merely a mathematical convenience, it ensures that physical quantities transform correctly between coordinate systems or reference

[2]On a spacetime diagram, these three cases correspond respectively to points *inside*, *outside*, and *on* the light cone, which defines the causal structure of spacetime.

frames. Contravariant components (A^μ) describe how something moves or points in spacetime, while covariant components (A_μ) describe how the geometry itself acts on that quantity. In curved spacetime, these two representations differ because the "grid" used to measure distances and directions is distorted. The metric tensor provides the translation between them, guaranteeing that inner products and physical laws remain consistent for all observers.

At first glance, all this discussion of tensors, indices, and coordinate transformations may seem rather abstract—especially for a lowly data scientist like myself. If any of the above feels overwhelming, the key takeaway is simple: because curvature depends on energy, and energy depends on curvature, Einstein's equations are profoundly non-linear, making them extraordinarily difficult to solve in full generality.

1.1.2 Linearizing Einstein's Field Equations

To make progress toward solving these non-linear equations, Einstein turned to a simpler regime known as the weak-field limit. He considered small disturbances of spacetime around a flat background, far away from any matter or energy that would otherwise curve spacetime. This approach, called *linearization*, assumes the gravitational field is weak and expands the metric as

$$g_{\mu\nu} = \eta_{\mu\nu} + h_{\mu\nu}, \quad |h_{\mu\nu}| \ll 1, \tag{1.3}$$

where $\eta_{\mu\nu} = \text{diag}(-1, 1, 1, 1)$ represents flat spacetime as described above, and $h_{\mu\nu}$ is a small perturbation describing "ripples" in the geometry.

While the Einstein tensor $G_{\mu\nu}$ normally involves products of derivatives of the metric, these expressions can be greatly simplified in the weak-field limit. Since $G_{\mu\nu}$ is constructed from the metric tensor $g_{\mu\nu}$ and its derivatives, substituting $g_{\mu\nu} = \eta_{\mu\nu} + h_{\mu\nu}$ and keeping only first-order (linear) terms in $h_{\mu\nu}$ eliminates the non-linear parts. In this approximation, the curvature depends linearly on the second derivatives of $h_{\mu\nu}$, yielding an equation with the same mathematical form as the standard wave equation:

$$\Box \bar{h}_{\mu\nu} = \frac{-16\pi G}{c^4} T_{\mu\nu}, \tag{1.4}$$

where $\bar{h}_{\mu\nu}$ is the *trace-reversed* perturbation, a rearranged form of $h_{\mu\nu}$ defined to simplify the algebra and remove redundant terms. The $\Box$ operator[3] acts on the components of $\bar{h}_{\mu\nu}$ and gives rise to equations that are mathematically equivalent to standard wave equations for each component of the metric perturbation. It combines the time and spatial second derivatives into a single spacetime operator, describing how disturbances propagate both in time and through space.

In empty space, where no matter or energy is present ($T_{\mu\nu} = 0$), the equations reduce to

$$\Box \bar{h}_{\mu\nu} = 0. \tag{1.5}$$

[3]The symbol $\Box = -\partial_t^2 + \nabla^2$ denotes the *d'Alembertian operator*, the four-dimensional analogue of the Laplacian. In Cartesian coordinates, it takes the form $\Box = -\frac{\partial^2}{c\partial t^2} + \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$,

This has the same mathematical form as the classical wave equation familiar from electromagnetism, implying that oscillatory, wave-like solutions traveling at the speed of light can satisfy it. A general plane-wave solution can be written as

$$\bar{h}_{\mu\nu}(t, \mathbf{x}) = A_{\mu\nu} \cos(\mathbf{k} \cdot \mathbf{x} - \omega t), \tag{1.6}$$

where $A_{\mu\nu}$ is a constant amplitude tensor, $\mathbf{k}$ is the wave vector, and $\omega = |\mathbf{k}|$ ensures propagation at the speed of light. These wave solutions describe small, propagating disturbances in spacetime curvature. But what actually produces them? Just as accelerating electric charges generate electromagnetic waves, accelerating masses, or more precisely, time-varying mass distributions with asymmetries, generate GWs. Because mass and energy are conserved, spacetime cannot radiate through monopole (total mass, constant) or dipole (center-of-mass motion, conservation of linear and angular momentum) terms. The lowest possible radiative contribution therefore arises from the *quadrupole moment*, which characterizes how the shape of a mass distribution changes over time—for example, two bodies orbiting each other in a binary system. In such systems, the evolving, asymmetric curvature of spacetime produces oscillations in the quadrupole moment, continuously emitting ripples that propagate outward at the speed of light.

1.1.3 Simplifying Further: Gauges and the TT Frame

While it may seem that we have already uncovered the GWs hidden within this formulation, not all of the wave-like solutions to Equation 1.5 correspond to physical phenomena. Some of them arise purely from our freedom to choose coordinates and therefore contain redundant, non-physical information. Even in this linearized form, the metric perturbations $h_{\mu\nu}$ still reflect this coordinate freedom i.e. the fact that the underlying physics remains unchanged under small shifts in the spacetime coordinates. To isolate the physically meaningful degrees of freedom and remove these redundancies, we impose a *gauge*: a convenient choice of coordinates that simplifies the equations and leaves only the true dynamical components - the ones that represent real GWs.

One particularly useful choice is the *transverse–traceless (TT) gauge*. In this gauge, the metric perturbation satisfies:

$$h^{0\mu} = 0, \qquad h^{\mu}{}_{\mu} = 0, \qquad \partial^{\mu} h_{\mu\nu} = 0, \tag{1.7}$$

meaning that the perturbation has no time components, is trace-free, and is transverse (divergence-free). Here, raising and lowering indices is done with the flat metric $\eta_{\mu\nu}$, and in this linearized context only affects signs for time components.

The first condition, $h^{0\mu} = 0$, ensures the wave produces no time distortions, only spatial ones[4]. The second, $h^{\mu}{}_{\mu} = 0$, removes any uniform expansion or contraction (like inflating or deflating space equally in all directions), leaving only shape distortions (stretching one direction while compressing another). The last, $\partial^{\mu} h_{\mu\nu} = 0$, enforces transversality, meaning the wave's effects occur only perpendicular to its direction of propagation. Together, these conditions eliminate all coordinate redundancies, leaving only two independent, physical components—corresponding to the

[4]This does not mean that time plays no role; rather, temporal distortions can be transformed away by an appropriate choice of coordinates, leaving only the measurable changes in spatial separation.

two polarization states of GWs. This is why the TT gauge captures the true physical content of a passing GW: it isolates the measurable distortions of spacetime itself.

To understand why only two independent components of a GW carry physical information, we begin with the full metric perturbation $h_{\mu\nu}$, a 4×4 matrix with 16 components. Since the perturbation is symmetric ($h_{\mu\nu} = h_{\nu\mu}$), this reduces the number of independent components from 16 to 10. Imposing the Lorentz gauge condition,

$$\partial^{\mu} \bar{h}_{\mu\nu} = 0,$$

removes four further degrees of freedom, leaving six. Finally, in the TT gauge, we set the temporal components of the perturbation to zero ($h_{\mu 0} = 0$) and impose tracelessness ($h^{\mu}{}_{\mu} = 0$). These conditions eliminate four more degrees of freedom, leaving only two. These two surviving components correspond to the measurable polarizations of a GW.

For a wave propagating along the z-axis, the remaining nonzero components of the metric perturbation take the form

$$h_{ij}^{TT}(t,z) = \begin{pmatrix} h_+(t,z) & h_\times(t,z) & 0 \\ h_\times(t,z) & -h_+(t,z) & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

where h_+ and $h_\times$ denote the two polarization states of the GW. Each of these components takes the plane-wave form introduced in Equation 1.6,

$$h_+(t,z) = A_+ \cos(kz - \omega t), \qquad h_\times(t,z) = A_\times \cos(kz - \omega t), \tag{1.8}$$

with amplitudes A_+ and $A_\times$ corresponding to the strengths of the two polarizations.

These are *transverse* waves because their effects occur only in the plane perpendicular to the direction of propagation (the x–y plane), as indicated by the vanishing z-components in the matrix above. Together, these two components fully describe the measurable distortions of spacetime produced by a passing GW.

1.1.4 Visualizing the Effect: How Gravitational Waves Stretch Space

We can now visualize how GWs distort space. Imagine a ring of free-floating test masses, like small particles in space, arranged in a circle in the x–y plane, as shown in Figure 1.1. As a GW passes along the z-axis, it alternately stretches and squeezes space, causing the distances between the particles to oscillate in the two characteristic polarization patterns, even though their coordinates remain fixed.

Substituting the waveforms from Equation 1.8 into the perturbed form of the Minkowski metric in Equation 1.2, and evaluating the resulting effect on freely floating test particles in the transverse (x–y) plane, yields the characteristic polarization patterns of a passing GW. For the + polarization, space expands along one axis while contracting along the perpendicular one:

$$\delta x = \frac{A_+}{2} x_0 \sin \omega t, \qquad \delta y = -\frac{A_+}{2} y_0 \sin \omega t.$$

For the × polarization, the stretching occurs along axes rotated by 45°:

$$\delta x = \frac{A_\times}{2} y_0 \sin \omega t, \qquad \delta y = \frac{A_\times}{2} x_0 \sin \omega t.$$

The result is a rhythmic distortion of space itself, alternately squashing and stretching any object within the wave's path. This deformation is what LIGO and Virgo measure using laser interferometry: the tiny change in distance (on the order of 10^{-19} meters) between mirrors several kilometers apart.

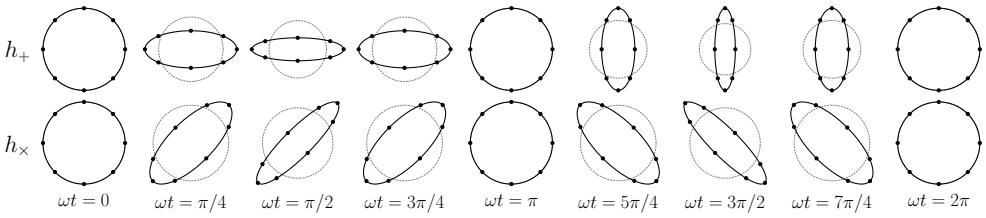


Figure 1.1: Deformation of a ring of test particles due to the + (top) and × (bottom) polarization modes of a GW propagating along the z-axis. The particles' coordinates remain fixed, but their physical separations oscillate.

1.1.5 Summary: Why Linearization and the TT Gauge Matter

The full Einstein equations describe how energy and momentum warp spacetime, but because they are non-linear, exact wave-like solutions are rare. Linearizing them transforms the problem into a wave equation, revealing that spacetime itself can carry energy and information across the universe.

The TT gauge isolates the physical degrees of freedom, showing that GWs are transverse, have two polarization states, and propagate at the speed of light. Together, these simplifications bridge the abstract mathematics of general relativity with observable physics: the tiny rhythmic distortions detected by modern interferometers are direct evidence of the spacetime ripples that Einstein first predicted more than a century ago.

1.2 Gravitational-Wave Sources

As discussed above, gravitational radiation originates from time-varying, asymmetric mass distributions i.e. systems with a changing quadrupole moment. The most powerful of these systems are compact binaries composed of the densest known objects in the Universe: pairs of black holes, neutron stars, or one of each, orbiting each other. Such systems produce the strongest curvature variations in spacetime and are therefore the most powerful sources of GWs known. Collectively, these systems are known as *compact binary coalescences* (CBCs), encompassing *binary black holes* (BBHs), *binary neutron stars* (BNSs), and mixed *neutron star-black hole binaries* (NS-BHs).

To date, all confirmed GW detections have originated from CBCs [1–3]. Each coalescence proceeds through three main phases: the *inspiral*, where the orbit shrinks as energy is radiated away through GWs; the *merger*, when the two compact objects collide and form a single remnant; and the *ringdown*, as that remnant settles into equilibrium through damped oscillations. These regimes are modelled respectively using post-Newtonian expansions, numerical relativity, and black-hole perturbation theory, as illustrated in Figure. 1.2. A more detailed discussion of CBC waveform modelling is provided later in Section 2.4.2.

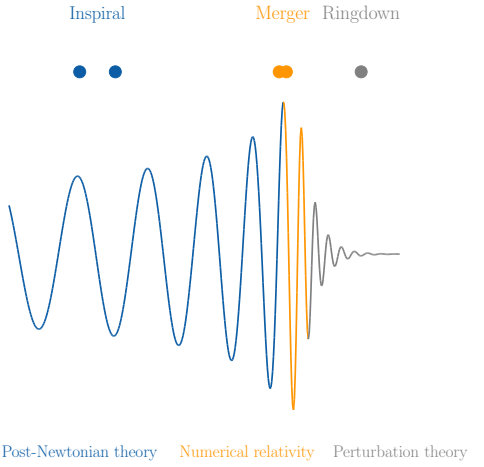


Figure 1.2: Time evolution of a binary system with $m_1 = 36M_\odot$ and $m_2 = 29M_\odot$, showing inspiral (blue), merger (orange), and ringdown (gray).

Beyond compact binaries, several other classes of astrophysical and cosmological systems that induce a changing quadrupole moment are expected to emit GWs: short-duration, poorly modelled *bursts* from core-collapse supernovae, gamma-ray bursts, or cosmic strings [4–6]; long-lived, nearly periodic *continuous waves* from rotating neutron stars with small surface asymmetries [7]; and the *stochastic background*, a random superposition of many unresolved signals of astrophysical or primordial origin [8–10]. However, the amplitudes expected from these sources are significantly weaker than those of CBCs and may only become observable with future generations of GW detectors.

1.3 Gravitational-Wave Detectors

1.3.1 Principles of Gravitational-Wave Detectors

GWs are extraordinarily faint by the time they reach Earth, causing extremely small changes in the separation between freely suspended test masses. Detecting them requires exquisitely sensitive instruments and a global effort that has taken over a century to realize since Einstein's prediction. Ground-based detectors such as LIGO [12], Virgo [13], and KAGRA [14] measure these displacements using highly sensitive Michelson interferometers, capable of detecting arm length variations of order 10^{-19} m.

A monochromatic laser beam is split into two perpendicular arms of equal length L. The beams reflect off mirrors (test masses) and recombine at the beam splitter, producing an interference pattern at the photodetector that depends on the relative phase accumulated along each arm. When a GW passes through the detector, it differentially stretches and squeezes the two arms, modulating the phase difference and thus the photodiode power.

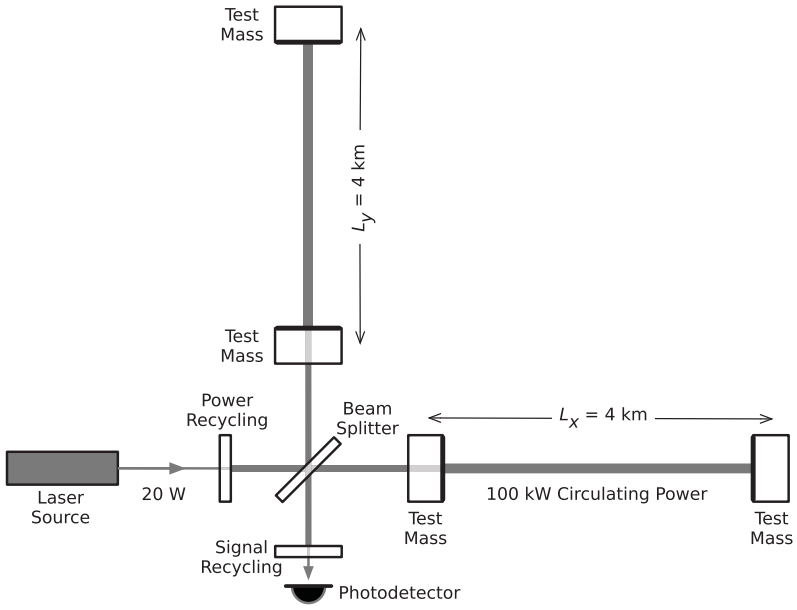


Figure 1.3: Schematic of a Michelson interferometer used in GW detectors such as LIGO and Virgo. A passing GW induces a differential stretch and squeeze of the interferometer arms, modulating the interference pattern measured by the photodetector. Adapted from [11].

The light intensity at the photodetector can be written schematically as

$$P_{\rm det} \propto P_0 \sin^2\big[\Delta\phi(t)/2\big], \tag{1.9}$$

where $\Delta\phi(t)$ is the phase difference between the returning beams. Small changes in the optical path length difference $\Delta L(t) = L_x - L_y$ lead to corresponding changes in phase,

$$\Delta\phi(t) = \frac{4\pi}{\lambda_L}\,\Delta L(t), \tag{1.10}$$

with λ_L the laser wavelength. Thus, a passing GW modulates the detected light power by perturbing $\Delta L(t)$.

In the long-wavelength limit ($\lambda_{\rm GW} \gg L$), the GW strain $h(t)$ produces [15] a fractional differential length change

$$\frac{\Delta L(t)}{L} = h(t), \tag{1.11}$$

which defines the fundamental detector observable.

For a general GW incident from an arbitrary sky direction, the measured strain is a linear combination of the two GW polarizations:

$$h(t) = F_+(\vartheta,\varphi,\psi)\,h_+(t) + F_\times(\vartheta,\varphi,\psi)\,h_\times(t), \tag{1.12}$$

where F_+ and $F_\times$ are the antenna pattern functions describing the directional sensitivity of the detector, and (ϑ,φ,ψ) denote the sky location and polarization angle.

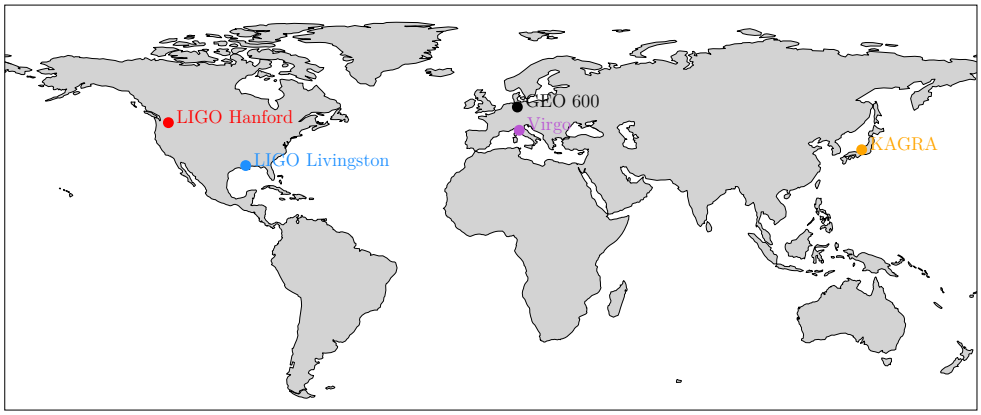


Figure 1.4: Global network of second-generation interferometric GW detectors.

The antenna pattern functions are given by [16]

$$F_+(\theta,\phi,\psi) = \tfrac{1}{2}(1+\cos^2\theta)\cos 2\phi\cos 2\psi - \cos\theta\sin 2\phi\sin 2\psi, \tag{1.13}$$

$$F_\times(\theta,\phi,\psi) = \tfrac{1}{2}(1+\cos^2\theta)\cos 2\phi\sin 2\psi - \cos\theta\sin 2\phi\cos 2\psi. \tag{1.14}$$

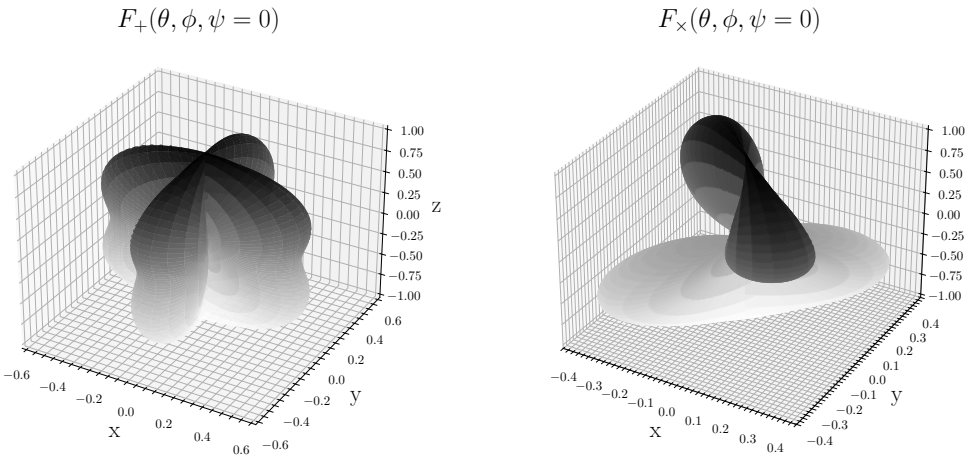


Figure 1.5: Antenna pattern functions F_+ and $F_\times$ for a ground-based interferometer, evaluated at $\psi = 0$.

Ground-based interferometers are operated at the "dark fringe," where the interference at the photodiode is nearly destructive in the absence of a GW. This configuration minimizes the steady laser power incident on the photodetector, thereby reducing Poisson (shot) noise from photon counting and allowing tiny phase modulations from passing GWs to produce measurable variations in light intensity.

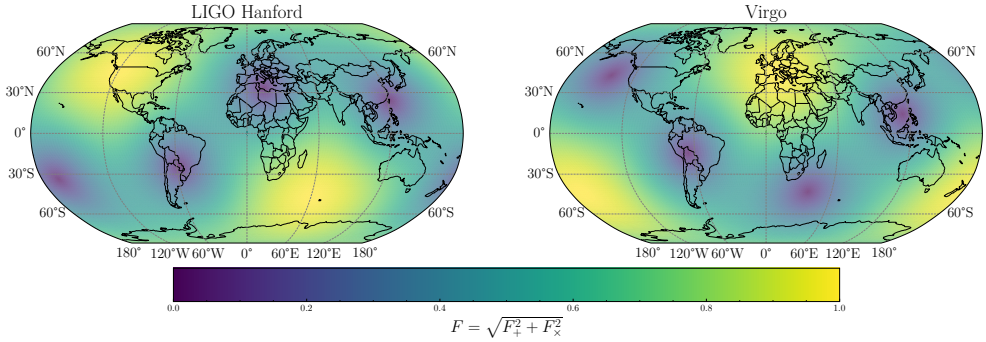


Figure 1.6: Overall detector response $F = \sqrt{F_+^2 + F_\times^2}$ for LIGO Hanford *(left)* and Virgo *(right)*.

1.3.2 The Importance of a Detector Network

Figure 1.4 shows the global network of ground-based detectors while Figure 1.5 illustrates the antenna pattern of a hypothetical detector. Figure 1.6 then shows the actual antenna patterns for LIGO Hanford and Virgo, highlighting how each instrument's sensitivity varies across the sky. The overall response, $F = \sqrt{F_+^2 + F_\times^2}$, varies with the source direction and polarization, making no single detector uniformly sensitive across the sky. This anisotropy underscores the importance of a global network of detectors: by combining signals from differently oriented interferometers, the network achieves accurate source localization and polarization reconstruction. The first binary neutron star event, GW170817, exemplified this advantage. Detected simultaneously by LIGO–Hanford, LIGO–Livingston, and Virgo, the source sky position was localized within a $28\,\mathrm{deg}^2$ area (90% confidence) at a distance of 40^{+8}_{-14} Mpc. This precise localization enabled the identification of electromagnetic (EM) counterparts, marking the first multimessenger (GW+EM) observation in history. Such achievements highlight the critical role of maintaining a full network of operational detectors.

1.3.3 Beyond the Idealized Interferometer

Achieving the sensitivity required for GW detection–fractional length changes of order 10^{-21}, or differential displacements of only $\sim 10^{-19}\,\mathrm{m}$ in a 4 km interferometer–demands a far more advanced design than the idealized Michelson configuration. Modern detectors employ Fabry–Pérot arm cavities [17], where partially reflective mirrors cause the laser light to bounce many times, effectively increasing the optical path length and enhancing the detector's response [18]. Maintaining resonance within these cavities requires active feedback control: magnetic actuators apply continuous corrections to the mirror positions, keeping the interferometer in a stable *lock* state. Precise calibration of the instrument response, accounting for actuator effects and feedback gain, is essential to recover the true GW strain [19–22]. Beyond the arm cavities and control systems, state-of-the-art detectors incorporate power-recycling optics, multistage mirror suspensions, ultra-high vacuum systems, squeezed-light sources, and advanced low-loss coatings to suppress technical noise and achieve quantum-limited

sensitivity [23].

1.4 Understanding Real Detector Noise

1.4.1 Overview and Sources of Noise

In practice, the strain data $s(t)$ recorded by GW detectors is a noisy estimate of the true signal $h(t)$,

$$s(t) = n(t) + h(t), \tag{1.15}$$

where $n(t)$ represents all background noise contributions. In many cases, $h(t) = 0$, corresponding to noise-only data.

A central assumption in GW data analysis is that $n(t)$ is approximately stationary and Gaussian over short timescales, although real detector noise often deviates from this idealization. Understanding and modelling these deviations is the focus of *detector characterization* [24–27], which ensures the quality and reliability of GW data.

Noise sources are broadly grouped into three categories:

- *Fundamental noise* — intrinsic and irreducible contributions such as quantum and thermal noise.
- *Technical noise* — instrumental or electronic effects that can often be mitigated.
- *Environmental noise* — external disturbances such as seismic, acoustic, or magnetic couplings [28].

1.4.2 Detector Sensitivity and the Power Spectral Density

The performance of a GW detector is quantified by its *power spectral density* (PSD), $S_n(f)$, which measures the average noise power per unit frequency. The square root of the PSD, the *amplitude spectral density* (ASD), gives the noise amplitude per unit frequency and provides a direct measure of strain sensitivity in $\mathrm{Hz}^{-1/2}$.

Figure 1.7 shows a representative ASD for LIGO Hanford during the third observing run, together with projected sensitivity curves for future upgrades and next-generation detectors such as the Einstein Telescope and Cosmic Explorer. The characteristic "U" in the curve marks the most sensitive frequency band, around 30–300 Hz for Advanced LIGO, where compact binary coalescence signals are most detectable. At lower frequencies ($\lesssim 20\,\mathrm{Hz}$), seismic and suspension noise dominate; thermal noise limits the mid-band, while quantum shot noise [29] sets the high-frequency floor above several hundred hertz. The PSD is not only a measure of detector performance but also a key input to data analysis (see Section 2.2.2), defining the noise weighting in matched filtering (see Section 2.4.2) and parameter estimation (see Section 2.5).

1.4.3 Noise Lines and Narrow-band Disturbances

Many instrumental subsystems couple into the strain channel $s(t)$, producing narrow spectral features or *noise lines* visible as peaks in the ASD [26]. These may appear

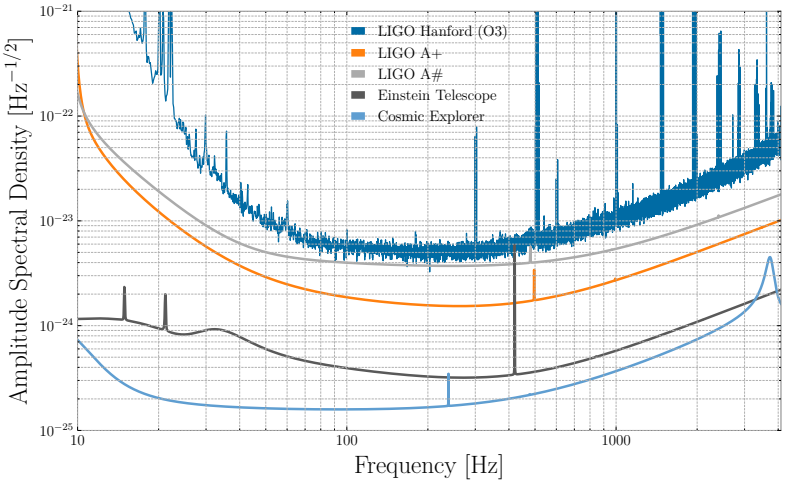


Figure 1.7: ASDs of LIGO Livingston during O3 (blue), and projected sensitivities for LIGO A$^\sharp$ (grey), A$^+$ (orange), Einstein Telescope (black), and Cosmic Explorer (light blue).

individually or as combs with spacing δf:

$$f_n = f_0 + n \cdot \delta f, \tag{1.16}$$

and arise from periodic disturbances such as power line harmonics (50/60 Hz), calibration lines, and suspension resonances.

Because unresolved lines can mimic or obscure certain GW signals, their identification and mitigation are continuous tasks within detector characterization [30]. This is achieved by correlating spectral features in $s(t)$ with hundreds of thousands of auxiliary channels that monitor the state of the detector and its environment. Auxiliary channels are classified as *safe* (insensitive to GWs) or *unsafe* (potentially sensitive). Times when excess coupling is detected in safe channels are flagged and excluded from astrophysical analyses [31]. In Part II of this thesis, we will see that accounting for such spectral line features is crucial for separating true GW signals and transient glitches (see next section) from the real detector background.

1.4.4 Glitches and Data Quality

In addition to stationary noise, detectors exhibit transient, non-Gaussian noise bursts known as *glitches* [32, 33]. These arise from complex instrumental or environmental couplings [34], often with unknown origins [35], and occur roughly once per minute during observing runs [2]. Glitches elevate the noise floor, bias parameter estimation, and can mimic or obscure astrophysical signals [36–39].

Periods of poor data quality are flagged automatically based on correlations between $s(t)$ and auxiliary sensors [31]. These *data quality vetoes* are categorized by severity: critical hardware or calibration issues (Category 1) lead to exclusion from all analyses, while milder but correlated disturbances (Categories 2–3) are selectively re-

moved from specific search pipelines [26, 27]. Together, these vetoes typically exclude less than 2% of the total observing time but significantly improve search sensitivity.

Given the rate and diversity of glitches, manual inspection is infeasible. Model-free algorithms, Omicron [40] and Gravity Spy [41], are utilized for glitch detection and characterization, respectively. These tools visualize and identify glitches by analyzing excess power in Q-scans (spectrograms) [42], shown in Figure 1.8, which are time-frequency representations (see Section 2.4.4 for a detailed description). The Gravity Spy project [41, 43] integrates machine learning with citizen science to classify glitches in their Q-scan representation. It identifies over twenty distinct morphologies, including well-known classes such as Blip, Scattered Light, and Tomte, shown in Figure 1.8.

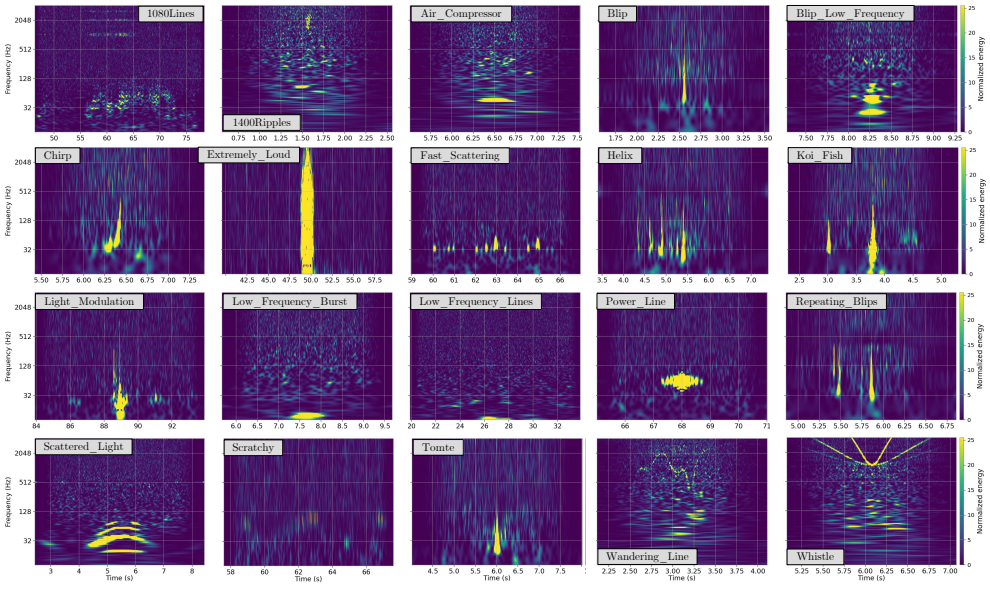


Figure 1.8: Q-scans (time–frequency representations) of representative glitch classes observed in LIGO during O3 [44]. The colour scale indicates the *normalized energy* in each time–frequency tile, defined as the squared magnitude of the whitened Q-transform coefficient and normalized by the expected Gaussian noise level. Brighter colours correspond to larger excess power relative to the background noise.

Despite these advances, current glitch classification pipelines face limitations: human labelling remains time-intensive, datasets are class-imbalanced, and fixed taxonomies cannot fully capture evolving noise morphologies [45, 46]. Furthermore, they discard phase information by only considering magnitude Q-scans, which may be important for characterizing glitches. This is something we demonstrate in Chapter 8.

Glitch Mitigation

Mitigating glitches remains an active challenge, particularly when the overlap in time and frequency with gravitataional wave signals. Parametric and machine-learning

approaches have been explored [47–49]. However, up to the end of O3, BayesWave [36, 50–52] and gwsubtract [53–55] have been the only algorithms used for modelling and subtracting glitches in LIGO-Virgo-KAGRA data analyses [55]. While gwsubtract relies on strongly correlated auxiliary witness sensors to linearly subtract glitches from the strain data, BayesWave instead models the strain data directly as a combination of Gaussian noise, a CBC signal, and a glitch component.

Three distinct model assumptions guide the reconstruction of transient features in the data:

1. Gaussian noise with an astrophysical signal,
2. Gaussian noise with an instrumental glitch,
3. Gaussian noise with both a signal and a glitch.

The signal is represented using physically motivated GW templates (see Section 2.4.2), whereas glitches are modelled as a sum of sine–Gaussian (Gabor–Morlet) wavelets,

$$g(t;\boldsymbol{\theta}) = \sum_{j=1}^{N} A_j \exp\left[-\frac{(t-t_{0,j})^2}{\tau_j^2}\right] \sin(2\pi f_{0,j}(t-t_{0,j}) + \phi_{0,j})\,, \tag{1.17}$$

with amplitude A_j, central time $t_{0,j}$, decay timescale τ_j, central frequency $f_{0,j}$, and phase offset $\phi_{0,j}$ parameterizing the j^{th} wavelet.

BayesWave performs full Bayesian inference (see Section 2.5) to determine both the number of wavelets N and their parameters. Under the second model assumption (glitch-only), only the glitch wavelets are inferred, whereas under the third (signal+glitch), the signal and glitch wavelets are inferred simultaneously. This is achieved using a Reversible-Jump Markov Chain Monte Carlo (RJMCMC) sampler [56], which allows transitions between models of different dimensionality. For the signal model, realistic GW templates are projected coherently onto each detector [51], whereas for the glitch model, wavelet parameters are independent across detectors. In parallel, the noise power spectral density (PSD) is estimated directly from the data using a cubic spline combined with Lorentzian components [57].

In this thesis, we develop a deep-learning framework called DeepExtractor that provides a deterministic, rather than probabilistic, alternative to BayesWave's objectives. Chapter 4 addresses the first two model assumptions, and Chapter 5 targets the third assumption involving simultaneous signal and glitch reconstruction and separation.

Chapter 2

Gravitational-Wave Data Analysis

The goal of GW data analysis is to extract precise astrophysical information from a large amount of raw noisy detector data recorded by ground-based interferometers. This chapter introduces the core analysis tasks that underpin modern GW astronomy, with a particular emphasis on those most relevant to the methods developed in this thesis, namely; noise characterization, searches and parameter estimation. Before diving into the technical details of the data analysis tasks relevant to this research, we first highlight some of the major analysis goals of the LIGO-Virgo-KAGRA (LVK) collaboration.

2.1 Overview of Gravitational-Wave Data Analysis

This section provides a concise overview of the core components of modern GW data analysis pipelines and highlights several key scientific questions addressed by the field. Other major LVK science goals not covered here include searches for continuous waves [58–61], GWs from core-collapse supernovae [4, 62–64], and the stochastic GW background [6, 9, 65–67].

Noise Characterization and Data Conditioning. As discussed in the previous chapter, GW detectors operate at the limits of instrumental sensitivity, and the raw strain data $s(t)$ therefore contain a mixture of true astrophysical signals, instrumental noise, and glitches. The first step in any analysis is to statistically characterize this noise, typically through its PSD, which models the stationary component of the detector background (see Section 2.2). The data are then conditioned through filtering and whitening to suppress non-astrophysical features and to standardize the noise properties (see Section 2.3).

Waveforms. Modelled CBC analyses require fast and accurate predictions of GW signals emitted by BBH, BNS, and NSBH systems (see Section 2.4.2). State-of-the-art waveform models combine perturbative descriptions of binary dynamics with high-fidelity numerical-relativity simulations, leading to two main families: effective-one-body (EOB) models [68–73] and phenomenological models [56, 74–76].

Searches. Once the data have been conditioned, search methods (see Section 2.4) are used to identify candidate signals. Searches are broadly divided into *modelled* CBC searches—based on matched filtering [77] and using the waveform families

above—and *unmodelled* searches, which identify coherent excess power across detectors from poorly modelled or unexpected sources [78–80]. Unmodelled searches are less sensitive to CBC signals but are capable of detecting a wider variety of transient phenomena.

Parameter estimation. Once a candidate event is identified, Bayesian parameter estimation (see Section 2.5) is used to infer the physical properties of the source and quantify associated uncertainties [81–83]. This step transforms the output of the search pipelines into astrophysically meaningful measurements.

Rates and population. Combining many detections enables constraints on the populations of compact binaries, including merger rates and the statistical distributions of component masses, spins, and redshifts. Several methods leverage search results to obtain merger-rate estimates [84, 85]. With hundreds of detections now available, sophisticated Bayesian population analyses have emerged, using both parametric [86–88] and non-parametric [89–93] models, relying on event-level posteriors (see Section 2.5) to infer the underlying astrophysical distributions [1–3].

Tests of General Relativity. Comparing observed signals with theoretical predictions allows for stringent tests of General Relativity across the inspiral, merger, and ringdown regimes. Such tests constrain the dynamics of binary evolution [94, 95], the speed of GW propagation, and the number of spacetime polarizations [96–99]. To date, no statistically significant deviation from General Relativity has been observed.

Extreme matter and multi-messenger astronomy. Binary neutron star (BNS) mergers [100] allows us to study matter in cases where it is squeezed to the most extreme densisties (i.e. $\rho \simeq 10^{14} g/cm^3$). In such conditions, the macroscopic behavior of matter is still poorly understood. GW observations constrain neutron-star tidal deformability [101]. These mergers also produce energetic gamma ray bursts [102, 103] and the ejection of highly radioactive matter in what are called kilonovae [104]. Both the gamma ray burst and kilonova can be observed in the electromagnetic band with existing telescopes. In particular, the multi-messenger observation of GW170817 [105–107] yielded groundbreaking constraints on nuclear physics, jet dynamics, and heavy-element production [108–114]. Future BNS detections are expected to further expand these insights.

Cosmology. CBC mergers serve as "standard sirens": their GW signals provide a direct, calibration-free measurement of distance. Coupled with redshift information, this enables constraints on the expansion history of the Universe through measurements of the Hubble constant H_0 [115–119]. For multi-messenger events such as GW170817, the redshift is obtained from the host galaxy identified through electromagnetic counterparts [120, 121]. Although current uncertainties remain large, additional BNS standard-siren detections may help resolve the ongoing "Hubble tension" [122] between cosmic microwave background measurements [123] and estimates from "standard candles" [124, 125].

2.2 Characterizing Detector Noise

Noise is the dominant output of any ground-based detector. Currently, GWs from BBHs are recorded every few days, and even then their amplitude is typically one or two magnitudes smaller than the noise itself. Thus, characterizing the statistical properties of noise is essential for every data analysis application, particularly those involved with the detection and estimation of astrophysical parameters.

2.2.1 Definitions for Time-series Analysis

This section covers mathematical notations and definitions to supplement the analysis of GW data. In practice, the strain $s(t)$ from Equation 1.15 is recorded at discrete times t_n, separated by a constant spacing Δt. We define the sampling frequency (rate) as $f_s = 1/\Delta t$. Advanced LIGO's output is sampled at $f_s = 16,384\,\text{Hz}$, but for typical applications, it is often sufficient to analyze the data at $f_s = 4,096\,\text{Hz}$ or $f_s = 2,048\,\text{Hz}$. We can separate a time series $s(t)$ of length T by a real vector when it is sampled a a given sample rate such that

$$s_n = s(t_n), \qquad n \in \{0, \dots, N-1\} \tag{2.1}$$

where $N = T/\Delta t = T f_s$.

In certain analyses, it is convenient to represent a time series in the frequency domain, requiring a Fourier transform $\mathcal{F}$, which yields a complex frequency series $\tilde{s}(f) = \mathcal{F}s(t)$, written as

$$\tilde{s}(f) = \int_{-\infty}^{+\infty} dt\ s(t) e^{-i2\pi ft} \tag{2.2}$$

with its inverse

$$s(t) = \int_{-\infty}^{+\infty} df\ \tilde{s}(f) e^{i2\pi ft}. \tag{2.3}$$

It is observed that if $s(t)$ is real, then $\tilde{s}(f) = \tilde{s}^*(-f)$, where $(\,\cdot\,)^*$ denotes complex conjugation, meaning that there is a symmetry between the positive and negative frequencies of the frequency domain series.

Given the discrete sampling of GW data, the *discrete Fourier transform* (DFT) is used, providing a discrete frequency-domain series as

$$\tilde{s}_k = \sum_{n=0}^{N-1} s_n e^{-i2\pi \frac{n}{N} k} \tag{2.4}$$

with its inverse

$$s_n = \sum_{k=0}^{N-1} \tilde{s}_k e^{i2\pi \frac{k}{N} n}. \tag{2.5}$$

Consisting of N points, the discrete frequency series $\tilde{s}$ is evaluated on a frequency grid with constant spacing $\Delta f = \frac{1}{T}$ with

$$f_k = \{-f_{Ny}, -f_{Ny} + \Delta f, ..., 0, ..., f_{Ny} - \Delta f, f_{Ny}\} \tag{2.6}$$

where $f_{Ny} = \frac{1}{2} f_s = \frac{1}{2\Delta t}$ is called the Nyquist frequency, which defines the maximum measurable frequency that can be uniquely represented in a discretely sampled signal given the time resolution.

Many analyses apply a window to the data which corresponds to multiplying the time series $s(t)$ by a so-called window function $W(t)$. The window function typically has zero values at the edges of the time grid, allowing edge effects to be avoided which arise from computing the DFT on time series with limited sample sizes.

We may also define the cross-correlation $\star$ between two time series $a(t)$ and $b(t)$ as

$$(a \star b)(t) = \int_{-\infty}^{+\infty} a^*(\tau)\, b(\tau + t)\, d\tau, \tag{2.7}$$

Here, τ is a dummy integration variable representing absolute time, while t plays the role of a time lag (or shift). The cross-correlation function $(a \star b)(t)$ therefore quantifies the similarity between a and a time-shifted version of b: for each lag t, it measures how well the two signals overlap when one is delayed relative to the other. In the special case $a = b$, this reduces to the autocorrelation, which characterizes periodicity and coherence within a single signal.

The cross-correlation has a convenient property: its Fourier transform is the product of the complex conjugate of one signal's transform and that of the other,

$$\mathcal{F}\{a \star b\}(f) = \tilde{a}^*(f)\tilde{b}(f). \tag{2.8}$$

This operation forms the basis of the matched-filtering algorithm used in GW searches (see Section 2.4.2), where the detector strain is correlated with model waveforms to identify signals with coherent phase evolution.

2.2.2 Power Spectral Density as a Statistical Model for Noise

The noise component in Eq 1.15, n_t, can be conveniently modelled as a stochastic process, since it takes some random value. A stochastic process is a collection of random variables indexed by a variable t which may be correlated with one another. We can define the mean $\mu(t)$ and the two-point correlation function $K(t_1, t_2)$ for a stochastic process as

$$\mu(t) = \mathbb{E}[n_t] \tag{2.9}$$

$$K(t_1, t_2) = \mathbb{E}[n_{t_1}, n_{t_2}] \tag{2.10}$$

where the expectation value $\mathbb{E}$ is computed over many different noise realizations, known as an ensemble average.

It is often assumed that the noise process n_t is fully characterized by its mean and two-point correlation function. This means that other quantities that may be derived (eg. three-point correlation) are functions of these two quantities. It can be shown

[126, 127] that access to $\mu(t)$ and $K(t_1, t_2)$ is sufficient to express the probability distribution p for a realization of the noise $(n_0, ..., n_{N-1})$ as

$$p(n_0, ..., n_{N-1}) \propto exp\left\{-\frac{1}{2}\sum_{ij}(n_i - \mu_i)K_{ij}^{-1}(n_j - \mu_j)\right\} \tag{2.11}$$

where $K_{ij} = K(t_i, t_j)$, $\mu_i = \mu(t_i)$ and K^{-1} denotes matrix inversion. This result is a consequence of applying the Maximum Entropy principle, which states that such a distribution maximises the entropy given the constraints provided by $\mu(t)$ and $K(t_1, t_2)$. The solution to this maximization problem is a multivariate Gaussian distribution. Consequently, any stochastic process whose realizations follow Equation 2.11—that is, one fully described by its mean and covariance—is referred to as Gaussian noise.

An important property is that the mean of the detector's noise is zero i.e., $\mu(t) = 0$. Furthermore, it is found that $K(t_1, t_2)$ depends only on $\tau = t_1 - t_2$ and is therefore invariant under a constant time shift $t_1, t_2 \to t_1 + C, t_2 + C$. A process with this property is described as *wide-sense stationary*, however, such an empirical observation is only valid when averaging over a 'short' period of time on the order of hours. This assumption of stationarity is safe for most GW data analyses, where durations on the order of $\mathcal{O}$(10 seconds) are considered. While the properties of stationary noise changes over longer timescales, this can be accounted for and there are approaches for dropping this assumption [128–132].

We define the autocorrelation function $C(\tau)$ of a wide-sense stationary process as

$$C(\tau) = K(\tau, 0) = \mathbb{E}[n_0 n_\tau], . \tag{2.12}$$

Under the assumption of Gaussian noise, the stochastic properties of the process are completely characterized by its autocorrelation function. This function, in turn, can be derived consistently using the Maximum Entropy principle.

The properties of stationary noise can be described further by considering the correlation of $\mathbb{E}[\tilde{n}_f^* \tilde{n}_{f'}]$ for different frequencies averaged over time.

$$\begin{aligned}
\mathbb{E}[\tilde{n}_f^* \tilde{n}_{f'}] &= \mathbb{E}\left[\int_{-\infty}^{+\infty} dt\, dt'\, n(t) n(t') e^{i2\pi f t} e^{-i2\pi f' t'}\right] \\
&= \int_{-\infty}^{+\infty} dt\, d\tau\, C(\tau) e^{-i2\pi t(f'-f)} e^{-i2\pi f'\tau} \\
&= \delta(f - f') \int_{-\infty}^{+\infty} d\tau\, C(\tau) e^{-i2\pi f'\tau} \\
&= \delta(f - f') S(f)\,.
\end{aligned} \tag{2.13}$$

Here, we used $C(t'-t) = \mathbb{E}[n_0\, n_{t'-t}]$ and performed a change of variables $\tau = t' - t$. The function $\delta(f - f')$ is the *Dirac delta function*, which is zero everywhere except at $f = f'$ and satisfies $\int \delta(f - f')\, df = 1$.

Finally, we define the *power spectral density* (PSD) as the Fourier transform of the autocorrelation function:

$$S(f) = \int_{-\infty}^{+\infty} d\tau\, C(\tau) e^{-i2\pi f\tau} \tag{2.14}$$

It is noted that $C(\tau)$ is a real function and that $S(f) = S(-f)$. The presence of the Dirac delta function in Equation 2.13 implies that, for a stationary process, the components in the frequency domain are uncorrelated with one another. Moreover, it implies that the variance $\mathbb{E}[|\tilde{n}_f|^2]$ in the frequency domain amounts to the Fourier transform of the autocorrelation function.

The function $S(f)$ is the PSD previously introduced in Section 1.4, and provides a statistical description of how the noise power is distributed across frequencies. From the standpoint of time-series analysis, it can be defined as the squared magnitude of the Fourier transform $\tilde{n}_T(f)$ of the noise time series $n_T(t)$ over an interval of duration T, in the limit of large T:

$$S(f) = \lim_{T\to\infty} \frac{1}{T}|\tilde{n}_T(f)|^2. \tag{2.15}$$

The PSD is commonly measured in units of 1/Hz, while we also saw in Section 1.4 that it is sometimes convenient to work with its square root, $\sqrt{S_n(f)}$, known as the *amplitude spectral density* (ASD), with units $\mathrm{Hz}^{-1/2}$.

Similarly to the time domain, the maximum entropy probability density can be written for obtaining a noise realization of frequency domain noise $(\tilde{n}_0, ..., \tilde{n}_{N-1})$. Assuming a zero mean and rearranging Equation 2.11 we yield

$$p(n_0, ..., n_{N-1}) \propto exp\left\{-\frac{1}{2}\sum_i \Delta f \frac{|\tilde{n}_i|^2}{S(f_i)}\right\} \tag{2.16}$$

where the frequencies f_i are of the same grid shown in Equation 2.6 with $\Delta f = 1/T$. In the continuous limit, the above equation is written as

$$p[\tilde{n}(f)] \propto exp\left\{-\frac{1}{2}\int_{-\infty}^{+\infty} df \frac{|\tilde{n}(f)|^2}{S(f)}\right\} \tag{2.17}$$

This expression can be used for sampling noise realizations according to $S(f)$, while it can also be used to compute the likelihood that a frequency series $\tilde{n}(f)$ results from the noise process that is described by $S(f)$.

Equation 2.17 can be simplified further by introducing the Wiener complex scalar product $\langle\cdot|\cdot\rangle$ between two time series $a(t)$ and $(b)t$ as

$$\langle a|b\rangle = \int_{-\infty}^{+\infty} df \frac{\tilde{a}^*(f)\tilde{b}(f)}{S(f)}. \tag{2.18}$$

The real and imaginary parts of the Wiener product $\langle\cdot|\cdot\rangle$ are denoted with $(\cdot|\cdot)$ and $[\cdot|\cdot]$, so that

$$\langle a|b\rangle = (a|b) + i[a|b]. \tag{2.19}$$

We can then simplify Equation 2.17 as

$$p[\tilde{n}(f)] \propto e^{-\frac{1}{2}\langle n|n\rangle}. \tag{2.20}$$

The symbol $\hat{}$ is used to denote a normalized time series

$$\hat{a} = \frac{1}{\sqrt{(a|a)}} a(t) \tag{2.21}$$

where $(\hat{a}|\hat{a}) = 1$

In practice, the detector strain is calibrated only within a finite frequency band $[f_{\min}, f_{\max}]$, the limits of which depend on the specific application. Furthermore, we can introduce the one-sided PSD $S_n(f)$, defined only for frequencies $f > 0$

$$S_n(f) = 2S(f). \tag{2.22}$$

Given a frequency range and using the one-sided PSD, the Wiener product can be rewritten as

$$\langle a|b\rangle = \int_{f_{min}}^{f_{max}} df \frac{\tilde{a}^*(f)\tilde{b}(f)}{S_n(f)}, \tag{2.23}$$

allowing the scalar product to be calculated only within the specified frequency range. Typically, $f_{min} = 10/15\,\text{Hz}$ and $f_{max} = 1024/2048\,\text{Hz}$, focusing on target signals within the detectors' most sensitive frequency band. Denoting the sampled frequencies as f with spacing Δf, the Wiener inner product takes the discrete form

$$\langle a|b\rangle = 4 \sum_{f=f_{\min}}^{f_{\max}} \Delta f \, \frac{\tilde{a}^*(f)\, \tilde{b}(f)}{S_n(f)}, \tag{2.24}$$

where the prefactor of 4 arises from the use of a one–sided power spectral density. Here, the sum runs only over the frequency bins lying within the chosen analysis band $[f_{\min}, f_{\max}]$, ensuring the inner product is evaluated in the most sensitive region of the detector.

It is evident that computing the PSD is critical for characterizing detector noise. While alternative approaches exist, such as Burg's method [133], GW data analysis predominantly relies on Welch's method [134]. Welch's method relies on Equation 2.15, estimating the PSD by computing the DFT of noise realizations. In practice, the time-domain signal is divided into overlapping segments, each of which is windowed, transformed via the DFT, and then averaged to reduce variance in the estimate.

This procedure reduces statistical fluctuations but comes at the cost of frequency resolution, particularly at low frequencies where fewer frequency bins are available. Although Welch's method is widely used due to its simplicity and robustness, careful tuning is required to balance resolution against variance, since poor choices of segment length can significantly degrade the PSD estimate. Moreover, the choice of windowing function plays an important role in minimizing spectral leakage and obtaining an effective estimate of the PSD.

2.3 Data Conditioning

The previous sections have highlighted the challenging noise environment present in the detector strain $s(t)$, which is influenced by numerous sources. Because GW

signals are typically buried deep within this noise, signal processing techniques are essential to enhance and isolate them. In this section, we outline key data conditioning approaches that support the detection and accurate measurement of GWs.

2.3.1 Low-Pass, High-Pass, and Band-Pass Filters

To ensure the condition discussed in Section 2.2.1, where $f_s > 2f_N$, a low-pass filter can be used which only allows frequencies below a specified cut-off frequency. Conversely, a high-pass filter can be employed to only allow frequencies above a certain threshold. Low-pass and high-pass filters can be combined to form band-pass filters that only allow frequencies within a range specified by high-frequency and low-frequency cut-offs. A schematic of impact that these three filters have on the frequency content of a signal is shown in Figure 2.1.

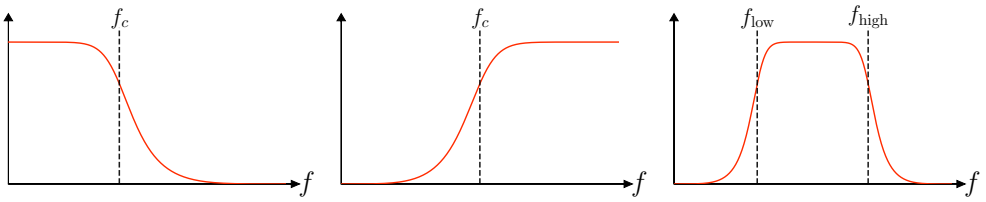


Figure 2.1: Idealized low-pass (left), high-pass (middle), and band-pass (right) filters, with cutoff frequencies indicated.

These filters are valuable in GW data analysis not only for preventing aliasing, but also for simplifying the data by isolating specific frequency bands of interest. By restricting the analysis to the relevant frequency range, one can avoid unnecessary complexity and often obtain more accurate results.

2.3.2 Whitening

As discussed in Section 2.2, Gaussian noise sampled at different times may exhibit correlations. These correlations are quantified by the autocorrelation function $C(\tau)$ in the time domain, or equivalently by the power spectral density $S(f)$ in the frequency domain. Removing such correlations from the detector strain $s(t)$ is a key step in GW data analysis, as it allows any residual correlations, such as those originating from astrophysical signals, to be more easily identified.

An important transformation for this purpose is *whitening*, a standard preprocessing technique in many GW analyses. Whitening produces an uncorrelated noise process $n^W(t)$, called *white noise*, in which each sample is statistically independent and the autocorrelation function reduces to $C(\tau) = \delta(\tau)$. Although glitches can also introduce residual correlations, whitening aids both their identification and mitigation.

Conceptually, whitening rescales each frequency component so that the variance of the power spectrum becomes uniform across frequency bins. In this sense, it acts as a normalization of the strain data across the frequency spectrum, enhancing excess power relative to the background noise. Whether localized in a single bin or spread

across multiple bins, such excess power manifests as correlations in the time domain, underscoring the central role of the whitening transform.

$$n_t^W \sim \mathcal{N}(0,1). \tag{2.25}$$

The whitening transformation is written as follows

$$n^W(t) = \int_{-\infty}^{+\infty} df \frac{\tilde{n}(f)}{\sqrt{S(f)}} e^{i2\pi ft} \tag{2.26}$$

Computing the two-point correlation function shows how Equation 2.26 produces white noise:

$$\begin{aligned}\mathbb{E}\big[n_{t_1}^W n_{t_2}^W\big] &= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} df\, df' \, \frac{\mathbb{E}[\tilde{n}^*(f)\tilde{n}(f')]}{S(f)} e^{i2\pi f t_1} e^{-i2\pi f' t_2} \\ &= \int_{-\infty}^{+\infty} df\, e^{i2\pi f(t_1-t_2)} = \delta(t_1-t_2)\end{aligned} \tag{2.27}$$

where Equation 2.13 was used to calculate the expected value over different noise realizations. It is apparent that a white noise process has a constant PSD with

$$S^W(f) = \int_{-\infty}^{+\infty} d\tau \delta(\tau) e^{-i2\pi f\tau} = 1. \tag{2.28}$$

It is important to note that any white noise process can be rescaled by a constant factor to set a given variance of the associated distribution, however, this rescaling also affects the PSD.

2.4 Transient Gravitational-Wave Searches

The detection of GWs requires disentangling weak, transient signals $h(t)$ from noisy detector data $s(t) = h(t)+n(t)$, where the noise $n(t)$ often dominates by several orders of magnitude. Search pipelines are thus designed to extract statistically significant signals from non-stationary, non-Gaussian data streams. This section outlines the essential formalism behind modelled and unmodelled searches.

Although this thesis does not develop new search pipelines directly, the DeepExtractor method introduced in Chapter 4 has potential applications in the construction of search statistics, as will be discussed in the Conclusions (see Chapter IV). Moreover, several metrics central to searches—such as the *match* and the *fitting factor*—are used in this thesis as evaluation tools.

2.4.1 Hypothesis Testing for Searches

GW searches can be formulated as a hypothesis-testing problem between two competing hypotheses:

Null hypothesis (H_0): $s(t) = n(t)$, corresponding to detector noise only.
Signal hypothesis (H_a): $s(t) = h(t)+n(t)$, where a GW signal is present in addition to noise.

To discriminate between these, a *test statistic* is computed, whose value determines whether to reject H_0 in favor of H_a. According to the Neyman–Pearson lemma [135], the likelihood ratio between these hypotheses provides the optimal statistic, maximizing the probability of detection for a fixed false alarm rate.

Assuming Gaussian noise, the log-likelihood ratio can be expressed as

$$\Lambda = \log \frac{p(s|h)}{p(s|n)} = (s|h) - \frac{1}{2}(h|h), \tag{2.29}$$

where $(a|b)$ denotes the Wiener inner product introduced in Section 2.2. A key observation is that the statistic depends on the data only through $(s|h)$. This means that, in Gaussian noise, the projection of the data onto the signal model is a sufficient statistic, and no other linear statistic constructed from the strain carries more information about whether h is present.

To see how this leads to an optimal decision rule, consider an unknown signal amplitude a multiplying the template h, corresponding to marginalization over overall signal strength. Maximizing Equation 2.29 with respect to a yields

$$\max_a \Lambda = \max_a \left\{ a(s|h) - \frac{a^2}{2}(h|h) \right\}, \tag{2.30}$$

which is achieved when the model waveform is aligned with the data in this inner product. Thus, the most powerful detector in Gaussian noise is one that evaluates the noise-weighted correlation between the strain and the expected waveform.

This result forms the theoretical foundation of matched filtering, discussed in the next section.

2.4.2 Modelled Searches

Modelled searches target sources such as CBCs, whose waveforms can be computed accurately using post-Newtonian expansions [136] or the effective-one-body formalism [137], discussed below. For these sources, the optimal detection method is *matched filtering*, which cross-correlates the data with a bank of modelled waveforms.

Matched Filtering and the Optimal Statistic

Matched filtering computes the correlation between a signal model $h(t)$ and the detector strain $s(t)$:

$$\rho = \int_{-\infty}^{+\infty} s(t)K(t)\,dt, \tag{2.31}$$

where $K(t)$ is a filter to be determined. The goal is to maximize the ratio S/N, where S is the expected signal contribution to ρ and N is the root-mean-square noise contribution.

Working in the frequency domain, and assuming stationary Gaussian noise with PSD $S_n(f)$, one obtains

$$N^2 = \int_{-\infty}^{+\infty} |\tilde{K}(f)|^2 S_n(f)\,df, \tag{2.32}$$

and the expected signal contribution

$$S = \int_{-\infty}^{+\infty} \tilde{K}^*(f)\tilde{h}(f)\, df. \tag{2.33}$$

Maximizing S/N yields the optimal (Wiener) filter

$$\tilde{K}(f) \propto \frac{\tilde{h}(f)}{S_n(f)}, \tag{2.34}$$

leading to the matched-filter signal-to-noise ratio (SNR) statistic

$$\rho_{\mathrm{mf}} = (s|\hat{h}) = 4\,\mathrm{Re}\int_{f_{\mathrm{min}}}^{f_{\mathrm{max}}} \frac{\tilde{s}^*(f)\tilde{\hat{h}}(f)}{S_n(f)}\, df, \tag{2.35}$$

where $\hat{h}$ denotes a normalized template. The Wiener filter down-weights frequencies dominated by detector noise while emphasizing bands where the signal is strongest. We can also define the expected optimal SNR as

$$\rho_{\mathrm{opt}} = (h|h) = 4\int_{f_{\mathrm{min}}}^{f_{\mathrm{max}}} \frac{|\tilde{h}(f)|^2}{S_n(f)}\, df. \tag{2.36}$$

This result shows that matched filtering is the optimal linear method for detecting a known signal buried in stationary Gaussian noise. In practice, the SNR time series $\rho(t)$ is computed for time-shifted versions of each template, and local maxima exceeding a threshold yield *triggers* (see Figure. 2.2).

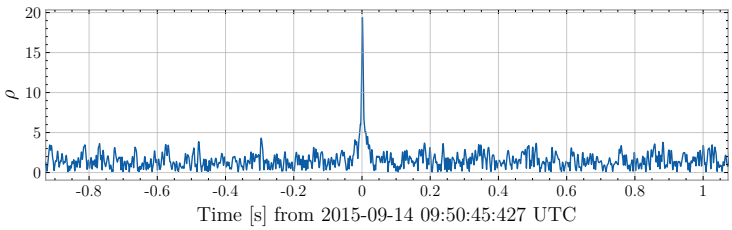


Figure 2.2: Matched-filter signal-to-noise ratio time series $\rho(t)$ obtained from LIGO Hanford data surrounding the binary black hole merger GW150914. A peak is observed at the merger time, where the corresponding template maximally overlaps with the data. Peaks exceeding a chosen threshold indicate candidate events (*triggers*), marking times at which the template matches the data most strongly.

Waveforms

CBC signals are fully characterized by a set of parameters, denoted by ϑ, numbering at least fifteen in the most general case. These parameters are divided into two categories: *intrinsic* and *extrinsic*.

The *intrinsic parameters* describe the physical properties of the binary system itself and determine the phase evolution of the emitted GWs. They include the

component masses (m_1, m_2) and the dimensionless spin vectors of the two objects, $(\boldsymbol{\chi}_1, \boldsymbol{\chi}_2)$, each with three components. Convenient derived quantities commonly used in waveform modelling are the total mass $M = m_1 + m_2$, the chirp mass $\mathcal{M} = (m_1 m_2)^{3/5}/M^{1/5}$—which largely sets the inspiral timescale—and the effective spin parameter $\chi_{\text{eff}} = (m_1\chi_1 + m_2\chi_2)/M$, which influences the phase evolution and merger frequency. If orbital eccentricity is included, two additional intrinsic parameters are required to describe the eccentricity and phase of periastron[1]. Furthermore, if one of the compact objects is a neutron star, an additional intrinsic parameter is introduced to describe its tidal deformability.

The *extrinsic parameters* specify how the source is oriented and observed from Earth, modulating the amplitude and polarization of the detected waveform. These include the luminosity distance D_L; the sky position of the source (θ, ϕ) in the detector frame; the inclination angle ι between the orbital angular momentum and the line of sight; the polarization angle ψ; the coalescence phase ϕ_c; and the coalescence time t_c. Together, these quantities define a generic binary system with a minimum of fifteen parameters (two masses, six spin components, and seven extrinsic parameters).

GW *waveforms* are solutions to Einstein's field equations that describe the spacetime dynamics of compact binaries based on their physical parameters. Since a complete analytic solution across the inspiral, merger, and ringdown (IMR) phases does not exist, waveform models rely on approximate formulations calibrated to numerical relativity simulations (see [138] for a review). While numerical relativity provides the most accurate representation of the full coalescence, its high computational cost makes it impractical for large-scale searches and parameter estimation, which may require generating millions of waveform evaluations. To overcome this limitation, several families of waveform approximants have been developed, each designed to capture the essential physics of the signal while remaining computationally tractable:

- **Post-Newtonian (PN)** expansions describe the inspiral regime for slowly moving, weakly gravitating binaries [139–141].
- **Numerical Relativity (NR)** solves the full Einstein equations for the highly nonlinear merger regime, at high computational cost.
- **Effective-One-Body (EOB)** models combine PN and NR results to generate accurate inspiral–merger–ringdown (IMR) waveforms [68–70].
- **Phenomenological (Phenom)** models provide analytic frequency-domain representations calibrated to NR simulations, enabling rapid and accurate waveform generation [56, 74, 75].

These waveform families form the foundation of CBC searches, enabling coherent matched filtering across a densely sampled parameter space while maintaining tractable computational cost.

Some experiments in this thesis utilize the IMRPhenom family of phenomenological waveforms, which provides a complete and computationally efficient description of compact binary coalescence. The IMRPhenomD model [142, 143] describes systems with spins aligned (or anti-aligned) with the orbital angular momentum and includes

[1]The periastron is the point in an eccentric orbit at which the two bodies reach their minimum separation. The *phase of periastron* specifies the orbital phase at which this closest approach occurs.

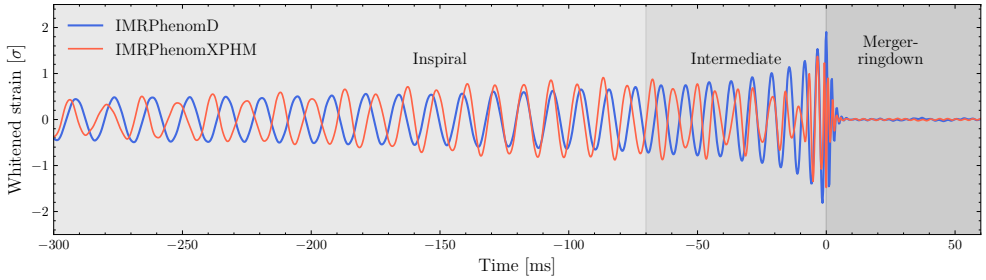


Figure 2.3: Whitened detector-frame waveforms corresponding to the maximum-likelihood parameters of GW190814. Two phenomenological waveforms are shown: IMRPhenomD (blue) and IMRPhenomXPHM (red). IMRPhenomD includes only the dominant mode and aligned-spin description, while IMRPhenomXPHM incorporates higher-order modes and full three-dimensional spin precession. The shaded regions approximately indicate the inspiral, merger, and ringdown regimes where different modelling methods are employed.

only the dominant quadrupole ($l = m = 2$) mode of emission. This model accurately captures most quasi-circular binary black hole signals but neglects effects from spin precession and higher-order harmonics.

These limitations are addressed in the IMRPhenomXPHM model [144], which extends IMRPhenomD by incorporating full three-dimensional spin precession and higher-order modes. Figure 2.3 illustrates the impact of these additional physical effects by comparing both waveform models for the same simulated system. Precession modulates the observed phase and amplitude as the orbital plane wobbles over time, while higher modes become important for systems with large mass ratios or near edge-on orientations. Together, these effects provide a more faithful representation of the GW signal, particularly for asymmetric or strongly spinning binaries.

Search Statistics and Parameter Maximization

In practice, searches maximize the statistic $\rho(\vartheta) = (s|\hat{h}(\vartheta))$ over all parameters ϑ:

$$\max_{\vartheta} \rho(\vartheta) = \max_{\vartheta}(s|\hat{h}(\vartheta)). \tag{2.37}$$

A key simplification arises from the fact that, for quasi-circular compact binaries, the dependence of the detector response on the extrinsic parameters enters only through an overall amplitude scaling and a constant phase shift. As a result, the effects of parameters such as the luminosity distance, inclination, polarization, and coalescence phase can be absorbed into an overall amplitude and phase.

It is therefore convenient to define two orthogonal waveform components $h_p(t; m_1, m_2, s_{1z}, s_{2z})$ and $h_c(t; m_1, m_2, s_{1z}, s_{2z})$ as

$$h_p(t) = A(t)\cos\big(\varphi(t)\big), \tag{2.38}$$
$$h_c(t) = A(t)\sin\big(\varphi(t)\big), \tag{2.39}$$

which satisfy $h_p \propto h_+$ and $h_c \propto h_\times$ in the detector response and are related in the frequency domain by

$$\tilde{h}_c(f) = i\,\tilde{h}_p(f). \tag{2.40}$$

Since these components are orthogonal, we also have $(h_c|h_p) = 0$.

It can be shown that the maximization problem can be written as

$$\max_{\text{extr.}} \rho^2 = (s|h_p)^2 + (s|h_c)^2. \tag{2.41}$$

This analytical maximization removes the need to explicitly search over these extrinsic degrees of freedom in aligned-spin binary black hole (BBH) systems. As a result, template banks—described in the next section—only need to cover a four-dimensional intrinsic parameter space rather than the full ten-dimensional space in the aligned-spin case. In more general cases, where these simplifying assumptions do not hold, some parameters can still be maximized over analytically [145]; however, the derivation lies outside the scope of this research. The reader is referred to Section 3.2.1 of [146] for a full derivation of these cases.

Template Banks

Since the intrinsic parameters cannot be maximized analytically, the parameter space must be discretized. A *template bank* is a discrete set of waveforms covering the parameter space of possible CBC signals [77, 147, 148]. Filtering data with imperfect templates leads to loss of recovered SNR, quantified by the *match*

$$M(\theta_1, \theta_2) = \max_t \left| 4 \int_{f_{\min}}^{f_{\max}} \frac{\tilde{h}^*(f;\theta_1)\tilde{h}(f;\theta_2)}{S_n(f)} e^{i2\pi f t}\, df \right|^2. \tag{2.42}$$

The match represents the overlap between two normalized signals after maximizing over relative time shifts. A match less than unity implies a loss of recovered SNR.

The *fitting factor* measures how well a bank recovers signals across parameter space, written as

$$FF(\theta) = \max_{\theta' \in \text{bank}} M(\theta, \theta'). \tag{2.43}$$

Template placement balances computational cost against SNR loss, often adopting a minimal match (MM) of 0.97 to limit the loss of detection efficiency to below $\sim 10\%$ [149].

Common placement strategies include metric-based [150], stochastic [151], and hybrid methods optimized for aligned-spin CBCs. The trade-off between coverage and efficiency directly impacts the sensitivity and false alarm rate of modern search pipelines. Figure 2.4 illustrates an example of a template bank generated with `mbank` [152] for a simplified non-spinning search.

Statistical Distribution and False Alarms

For Gaussian noise, the matched-filter output for a normalized template $\hat{h}$ follows a Gaussian distribution; $(\hat{h}|s) \sim \mathcal{N}(0,1)$ under H_0. The random variable, $\rho^2 = |x+iy|^2$,

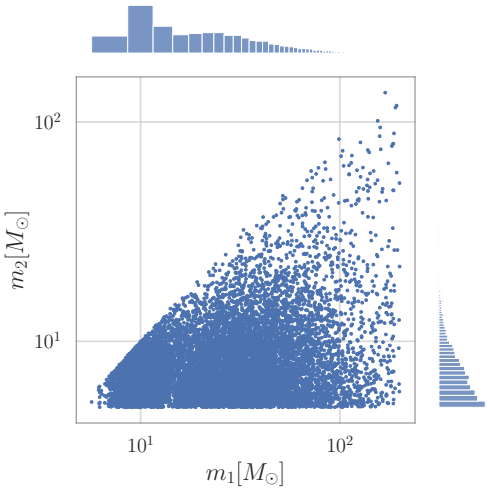


Figure 2.4: Example template bank constructed using the `mbank` framework [152] in the (m_1, m_2) parameter space for a nonspinning compact binary coalescence search. Each point corresponds to a single template, parameterized only by the component masses m_1 and m_2 (with $m_2 \leq m_1$), while all spin and angular parameters are fixed to zero.

is then the quadrature sum of two random normal variables $x, y \sim \mathcal{N}(0, 1)$ and follows a χ^2 distribution with two degrees of freedom i.e. $\rho^2 \sim \chi^2_2$. The corresponding probability density function for ρ is therefore

$$p(\rho) = \rho e^{-\rho^2/2}. \tag{2.44}$$

The *false alarm probability* (FAP) for an SNR threshold $\bar{\rho}$ is then given by

$$\mathrm{FAP}(\bar{\rho}) = \int_{\bar{\rho}}^{\infty} \rho\, e^{-\rho^2/2}\, d\rho = e^{-\bar{\rho}^2/2}, \tag{2.45}$$

which quantifies the probability that noise alone produces a value $\rho > \bar{\rho}$.

2.4.3 Features of a Realistic Search Pipeline

The theoretical matched-filter formalism must be adapted to real detector data, which are non-stationary, contaminated by transient artifacts, and recorded by multiple detectors. Operational search pipelines therefore extend the idealized framework into a multi-stage process [153, 154]. Key stages include:

- **Trigger generation:** identify candidate events by filtering the strain data with a large template bank.
- **Signal consistency tests:** reject glitches via time–frequency χ^2 tests [155].
- **Ranking:** combine SNR and consistency metrics into a detection statistic.

- **False alarm estimation:** compute background rates using time-shift analyses.
- **Astrophysical probability:** estimate p_{astro}—the probability a candidate is of astrophysical origin [11].

Pipelines such as GstLAL and PyCBC implement these steps in real time, producing statistically ranked candidate events.

2.4.4 Unmodelled Searches

While CBC signals are well modelled, other transient GW signals can arise from poorly understood or computationally intractable astrophysical processes. Such events, collectively referred to as *bursts*, are typically short in duration ($\lesssim 1$ s) and can span a wide range of frequencies. Unmodelled (burst) searches aim to detect these signals without assuming a specific waveform model, offering sensitivity to unexpected or weakly modelled astrophysical sources.

Excess Power Searches

Burst pipelines identify candidate signals by searching for statistically significant excess power in time–frequency representations of detector data. These algorithms may be *targeted*, focusing on specific astrophysical scenarios, or *generic*, relying only on minimal assumptions about signal morphology. A major challenge is distinguishing true astrophysical signals from glitches (see Section 1.4.4).

Since instrumental glitches are typically localized to individual detectors, coincident excess-power events appearing in multiple detectors are far more likely to be astrophysical. Burst pipelines therefore analyze data either *incoherently*, via independent single-detector searches followed by coincidence tests [40, 156], or *coherently*, combining multi-detector data to reconstruct a common sky-consistent signal [157].

In contrast to matched-filter CBC searches, burst analyses often use the *root-sum-square strain amplitude*,

$$h_{\text{rss}} = \sqrt{\int_{-\infty}^{\infty} h^2(t)\, dt}, \tag{2.46}$$

as a general measure of signal energy (in units of $\text{Hz}^{-1/2}$). Other intrinsic descriptors such as central time, frequency, duration, and bandwidth [158] can be derived to characterize a burst's morphology and facilitate cross-pipeline comparisons.

Given the unknown and expected broadband structure of burst signals, Fourier methods are limited by their fixed global basis. Instead, time–frequency analyses based on the Short-Time Fourier Transform (STFT) and wavelet transforms are widely employed.

Time-Frequency Transforms

Fourier analysis provides global frequency content but lacks temporal localization, making it poorly suited to transient signals. Burst pipelines therefore rely on time—frequency transforms that capture how spectral content evolves over time. The *Short-Time Fourier Transform* (STFT) introduces localization by applying a fixed window to the data and computing the Fourier transform within it. This produces a uniform

time–frequency grid and works well for compact events. However, its resolution is constrained by the window size: improving time resolution necessarily worsens frequency resolution, and vice versa. This fixed trade-off limits the STFT's effectiveness for signals that vary across multiple time–frequency scales.

Wavelet transforms overcome this limitation by adapting their time resolution to the frequency content of the signal [159]. High-frequency components are analyzed with short windows, while low-frequency components are analyzed with long windows. This multi-resolution behavior makes wavelets particularly effective for GW bursts, whose energy can span orders of magnitude in frequency over subsecond timescales. In practice, wavelets are also used for analyzing glitches, as discussed earlier in Section 1.4.4. A conceptual comparison between STFT and wavelets is shown in Figure 2.5.

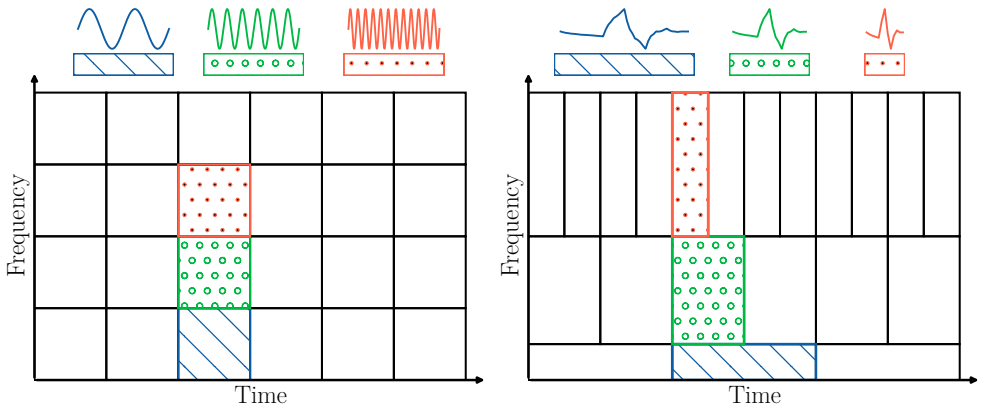


Figure 2.5: *Left:* Short-Time Fourier Transform (STFT) with fixed window size, resulting in uniform resolution across frequencies. *Right:* Wavelet transform using Daubechies wavelets, where variable window lengths yield high resolution for both short-duration, high-frequency and long-duration, low-frequency features [160]

A key concept in wavelet analysis is the *quality factor*, $Q = \sqrt{2} f_c/\sigma_f$, which controls the balance between time and frequency resolution. The *Q-transform*—a specific case of the continuous wavelet transform with constant Q—is widely used in GW burst searches [158]. It provides a convenient and physically meaningful way to visualize GW data, with approximately uniform resolution in logarithmic frequency.

Coherent WaveBurst and Multi-Resolution Analysis

A leading unmodelled search pipeline, Coherent WaveBurst (cWB), applies a coherent excess-power method across detectors to identify transient GW events [161, 162]. It performs the analysis in a wavelet domain using an orthogonal basis that preserves energy and minimizes redundancy, improving sensitivity to coherent multi-detector bursts.

cWB wavelet decomposition yields multi-resolution time–frequency maps, where short-duration, high-frequency wavelets capture compact structures (e.g., merger spikes), and long-duration, low-frequency wavelets capture extended features (e.g., inspirals

or echoes [163, 164]). This hierarchical decomposition enables the identification of clusters of excess energy consistent across detectors and sky positions.

To reduce background noise, cWB employs preprocessing techniques such as whitening and linear prediction error filtering, followed by a clustering stage that combines significant pixels across wavelet resolutions into coherent event candidates. Each candidate is then assigned a likelihood-based statistic that quantifies its consistency across detectors and its probability of astrophysical origin.

Overall, unmodelled burst searches complement modelled CBC analyses by providing sensitivity to unforeseen or weakly modelled phenomena, leveraging time–frequency methods such as the STFT and wavelet transforms to capture transient structure with high fidelity.

2.5 Measuring Gravitational-Wave Source Parameters

GW signals emitted by the merger of compact objects are now routinely detected by the current generation of detectors. We saw in Section 2.4.2 that GWs can typically be modelled using fifteen parameters. Accurate measurement of these parameters is crucial, as it enables a wide range of physical insights from GW data, including tests of general relativity and constraints on the neutron star equation of state. This chapter gives a brief overview of how the parameters of CBCs can be inferred from a GW signal using Bayesian parameter inference [81, 82].

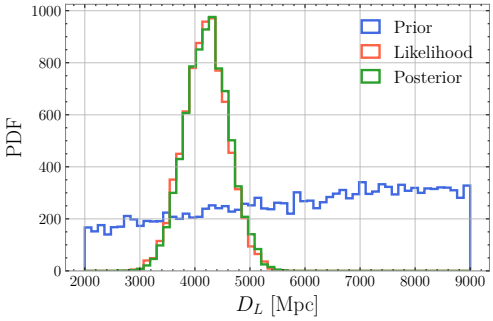


Figure 2.6: Illustrative probability distributions for the prior, likelihood, and posterior of the luminosity distance D_L, one of the extrinsic parameters measurable for BBH mergers.

Parameter estimation treats each model parameter as a random variable and infers its distribution conditioned on the observed data. Consider, for example, the luminosity distance D_{L} of the source. Before a signal is observed, we cannot make a measurement, and all values of D_{L} are equally likely—i.e., the *prior* distribution is uniform. After a GW signal is detected, the prior can be updated using the data to obtain the *posterior* probability distribution. This posterior represents the measurement: it describes the probability of the parameter values given the observed data.

2.5.1 Bayesian Analysis

The process of updating the prior distribution into the posterior is formalized using Bayes' theorem. We seek to estimate the posterior distribution $p(\vartheta|s)$, where p denotes a probability density function, ϑ is the set of GW source parameters, and s represents the data. Applying Bayes' theorem, we obtain:

$$p(\vartheta|s) = \frac{\pi(\vartheta)\, p(s|\vartheta)}{p(s)}, \tag{2.47}$$

where $\pi(\vartheta)$ is the prior probability distribution of ϑ, typically assumed to be uniform, and $p(s|\vartheta)$ is the likelihood of observing the data given the parameters. The term $p(s)$ serves as a normalization constant (called the *evidence*) when comparing hypotheses.

Figure 2.6 illustrates this framework for D_{L}. The posterior is obtained by reweighting the prior according to the likelihood: parameter values that better explain the data receive larger probability, whereas implausible values are suppressed.

2.5.2 Gravitational-Wave Likelihood

While the Bayesian framework is conceptually straightforward, parameter estimation in GW astronomy is computationally demanding. This is due to the high dimensionality of the parameter space ϑ (typically 15–17 parameters), the length of the detector strain data s, and the computational cost of generating accurate waveform models.

The likelihood of observing data s given a set of parameters ϑ is expressed as

$$\ln p(s|\vartheta) = -\frac{1}{2} \left\langle s - h(\vartheta) \mid s - h(\vartheta) \right\rangle, \tag{2.48}$$

where $h(\vartheta)$ denotes the GW signal predicted by one of the waveform models described in Section 2.4.2 for a given set of source parameters ϑ. These waveform families are employed to evaluate the likelihood function and to infer the most probable parameters consistent with the observed data.

The notation $\langle \cdot \mid \cdot \rangle$ represents the noise-weighted inner (Wiener scalar) product defined in Equation 2.24. For a network of detectors, the total log-likelihood is obtained by summing Equation (2.48) over all detectors.

2.5.3 Gaussian Noise Approximation

The likelihood in Equation (2.48) is derived from the Whittle likelihood [165], which is commonly used to model stationary Gaussian time series. As discussed in Section 2.4, this assumption is generally valid for the short data segments used in current GW analyses. Under the Gaussian noise approximation, the log-likelihood can be interpreted as a sum of independent Gaussian contributions across frequency bins, each with zero mean and variance given by the noise power spectral density $S_n(f)$. Figure 2.7 illustrates this assumption: the distribution of whitened detector data closely follows a standard normal distribution, as expected for stationary Gaussian noise.

However, the presence of glitches can violate the assumptions of stationarity and Gaussianity. We will examine in Chapter 5 how such artifacts can bias parameter estimation and how their impact can be mitigated through deep learning–based approaches.

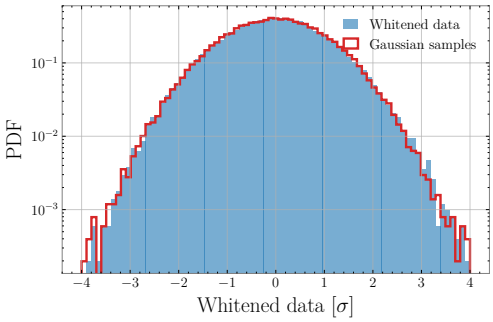


Figure 2.7: Comparison between a histogram of 16 s of whitened LIGO detector data (blue) and a standard normal distribution (red). Under the Gaussian noise approximation, the whitened strain samples should be distributed as $\mathcal{N}(0, 1)$, which forms the basis of the likelihood model used in GW parameter estimation.

2.5.4 Nested Sampling

Parameter estimation in GW astronomy involves exploring a highly multidimensional likelihood function, which is computationally intractable through uniform sampling. To address this, *nested sampling* [166, 167] is employed to efficiently compute the Bayesian evidence,

$$Z = \int \pi(\vartheta)\, p(s|\vartheta)\, \mathrm{d}\vartheta, \tag{2.49}$$

while simultaneously generating posterior samples $p(\vartheta|s)$. The algorithm achieves this by iteratively sampling from regions of increasing likelihood, effectively transforming the multidimensional integration over parameters into a one-dimensional integral over enclosed prior mass.

Nested sampling thus provides both the evidence used for model comparison and the posterior distributions used for parameter inference. It has become the standard approach for GW analyses due to its robustness and efficiency in exploring complex, correlated parameter spaces.

Chapter 3

Machine Learning

Over the past two decades, *Machine Learning* (ML) has emerged as one of the most transformative tools in science and engineering. By enabling algorithms to automatically extract patterns and predictive structures from data, ML has reshaped diverse fields including computer vision [168], natural language processing [169], healthcare [170], finance [171], and even the physical sciences [172]. The rapid progress of ML is largely driven by the confluence of increased computational power, massive data availability, and algorithmic innovations in both statistical modelling and neural network architectures.

GW physics is no exception to this trend. The detection and analysis of GW signals rely on sophisticated data analysis pipelines that must sift through terabytes of detector output dominated by noise. As discussed earlier in this thesis, traditional approaches based on matched filtering and Bayesian inference have proven remarkably successful [1–3]. However, the increasing sensitivity of current generation-detectors in upcoming observing runs demand faster, more adaptive, and more robust methods. These challenges will be amplified even further with next-generation observatories such as the Einstein Telescope (ET) [173] and Cosmic Explorer (CE) [174]. The ET alone is expected to detect on the order of 8×10^4 BBH mergers and 7×10^4 BNS mergers per year [175, 176], or more than 400 detections per day.

In these future observing scenarios, multiple source classes will often be simultaneously observable, and signals that currently appear as brief transients may persist for hours or even days within the detectors' sensitive frequency bands. For example, inspiral signals from the lightest binary systems could remain in-band for up to ten days, whereas those from the heaviest systems may last only a few hundred milliseconds. This dramatic increase in detection rate, signal overlap, and duration will substantially heighten the complexity of parameter estimation and introduce greater uncertainty in the inferred source properties. Moreover, next-generation detectors are expected to access entirely new astrophysical populations, including core-collapse supernovae and continuous emissions from rotating neutron stars [177, 178].

Aside from genuine astrophysical signals, the heightened sensitivity of GW detectors is expected to exacerbate issues relating to glitches [26, 27, 179, 180], as the lowered noise floor may expose new glitch morphologies originating from terrestrial phenomena that remain undetectable with current-generation instruments.

Matched-filter searches, will also become increasingly computationally demanding in next-generation detectors. The number of required waveform templates scales approximately as $f_s^{-11/3}$, where f_s is the lower cutoff frequency below which little SNR is accumulated. Because f_s will be 20–40 times smaller for next-generation

observatories such as ET and CE, the number of templates—and consequently the computational cost and false-alarm rate—could rise by several orders of magnitude, potentially rendering traditional matched-filtering approaches infeasible [181]. Machine learning offers powerful alternatives and complements to these conventional techniques, enabling fast, adaptive, and scalable solutions for tasks such as real-time detection, glitch classification, parameter estimation, and denoising. As a result, machine learning methods have gained increasing traction within the GW community [182–185].

In this chapter, we review the foundations of ML, beginning with the major learning paradigms—supervised and unsupervised—and the problem frameworks associated with each. We introduce common objectives, loss functions and evaluation metrics before moving to neural networks and the realm of deep learning, which provide the core methodology for this thesis. Finally, we connect these concepts to GW data analysis, showing how ML supports detection algorithms, parameter estimation, signal denoising, and data simulation. This discussion lays the groundwork for the following chapters, where deep learning methods are developed and applied to the challenges of real detector noise and transient GW signals.

3.1 Foundations of Machine Learning

ML encompasses several paradigms, defined by the type of supervision available and the nature of the task [186–188]. At the broadest level, we distinguish between *supervised* and *unsupervised* learning. Hybrid approaches exist between these extremes, and another major paradigm is *reinforcement learning* [189]; however, as these are not relevant to this thesis, we do not discuss them further.

3.1.1 Supervised learning

In supervised learning, the algorithm is trained on input–output pairs (x, y) to approximate a mapping $h : X \to Y$.

- *Regression:* the target variable y is continuous. Typical tasks include predicting real-valued physical parameters or denoising time series. Loss functions such as mean squared error (MSE) or mean absolute error (MAE) are commonly used. In this thesis, DeepExtractor (Chapters 4 and 5) is formulated as a regression model that learns to denoise and reconstruct GW signals and glitches.
- *Classification:* the target y takes discrete values (labels). Examples include image recognition or, in GW physics, distinguishing between different glitch morphologies e.g. Gravity Spy. Cross-entropy is the standard loss function for such tasks. In Chapters 6 and 7, we employ classification (detection) models to assess whether our simulated data can be used to train effective glitch and signal detection systems.

3.1.2 Unsupervised Learning

Unsupervised methods learn directly from the input distribution $p(x)$ without labelled outputs. They are widely used for:

- *Clustering:* grouping similar examples, e.g. organizing glitches by morphology.
- *Dimensionality reduction:* projecting data into lower-dimensional spaces (e.g., PCA, autoencoders) to capture essential features.
- *Anomaly detection:* identifying rare or unusual patterns that deviate from the bulk distribution.

Dimensionality Reduction: t-SNE and UMAP

While *Principal Component Analysis* (PCA) is a widely used linear method for dimensionality reduction, two prominent nonlinear alternatives for visualizing high-dimensional structures are *t-distributed Stochastic Neighbor Embedding* (t-SNE) [190] and *Uniform Manifold Approximation and Projection* (UMAP) [191][1]. t-SNE models pairwise similarities in the original and embedded spaces via probabilistic neighborhoods, while UMAP preserves both local and global topological structure based on manifold learning principles.

t-SNE. t-SNE maps high-dimensional data to a low-dimensional representation by minimizing the Kullback–Leibler divergence between two probability distributions that describe pairwise similarities:

$$p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2/2\tau_i^2)}{\sum_{k\neq i}\exp(-\|x_i - x_k\|^2/2\tau_i^2)}, \tag{3.1}$$

where τ_i is a perplexity-related parameter that sets the number of effective nearest neighbours (i.e. controls local vs global structure).

UMAP. UMAP constructs a high-dimensional graph based on nearest neighbors and optimizes its low-dimensional counterpart to preserve topological structure. It is computationally efficient and scales well to large datasets. Key hyperparameters include the number of neighbors (local vs. global structure), the minimum distance between embedded points (cluster compactness), the embedding dimension, and the choice of distance metric.

Throughout this thesis, PCA, t-SNE, and UMAP serve as key visualization tools. In Chapter 4, they are used to verify that DeepExtractor produces reconstructions that cluster coherently according to their known glitch classes from Gravity Spy. In Part III (Chapters 6–8), these methods enable a data-driven comparison between real data and data produced by our generative models, verifying whether they align in a fully data-driven manner.

[1] PCA is a linear technique that captures global variance in data but is limited to linear correlations among features. In contrast, t-SNE and UMAP are nonlinear manifold learning methods designed to preserve local structure and reveal complex, nonlinear relationships that PCA cannot represent.

3.1.3 Training, Validation, and Test Sets

Regardless of paradigm, ML requires a principled strategy to evaluate how well a model can generalize to unseen data. Typically, the available dataset in question is divided into three disjoint subsets:

- *Training set:* used to optimize the model parameters by minimizing a loss function (see Section 3.1.5 below).
- *Validation set:* used to tune hyperparameters and monitor performance during training, mitigating overfitting.
- *Test set:* held out until the final stage, providing an unbiased estimate of how well a model can generalize.

A common split is 80% training, 10% validation, and 10% test, although the exact proportions depend on dataset size and task [186].

Since much of the work in this thesis involves realistically simulated data, we are generally not constrained by limited dataset size and therefore do not need to adhere strictly to fixed train/validation/test splits. However, there are important exceptions. In Part III, when training generative models on real detector glitches, which are limited, we must hold out a reserved set of real glitches for validation and testing. These held-out samples enable a fair comparison between generated glitches and true detector artifacts and ensure that no information from the test set leaks into the training process.

3.1.4 Feature Scaling and Standardization

Before training, features are often transformed so that they lie on comparable numerical scales. Many machine-learning models are sensitive to the magnitude of input features. This is particularly true for *distance-based methods* (e.g., k-nearest neighbors, clustering) and *parametric models* (e.g., logistic regression, SVMs), where features with larger numerical ranges can disproportionately influence distance computations or parameter updates.

Two commonly used data preprocessing approaches are:

Standardization (z-score scaling). Features are transformed to have zero mean and unit variance:

$$x_{\text{standardized}} = \frac{x - \mu}{\sigma}, \tag{3.2}$$

where μ is the empirical mean and σ is the standard deviation computed from the training set. Standardization is appropriate when the data follow a roughly Gaussian distribution and is the default in many ML pipelines.

Min–Max Normalization. Features are rescaled to a fixed range, typically $[0, 1]$, or $[-1, 1]$:

$$x_{\text{min-max}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}. \tag{3.3}$$

This preserves relative spacing of values and is often preferred when the feature range is known (e.g., pixel intensities in images).

Neural networks, introduced in the next section, are particularly sensitive to feature scaling. Unscaled features can dominate gradient updates and lead to slow or unstable training.

Standardization is used in Chapters 4 and Chapter 5 for DeepExtractor, ensuring stable optimization and preventing differences in feature scale from hindering learning.

In Part III, min–max normalization is employed when training generative models on signal and glitch datasets. Here, the goal is to learn the *morphology* of the data rather than its amplitude. Normalizing inputs to a fixed range allows the network to focus on learning the underlying structure, and the generated samples can later be rescaled to any desired physical amplitude.

In general, all features across training, validation and test sets are standardized using the training set mean and standard deviation (or other relevant training set statistics) to avoid information leakage into the validation or test sets.

3.1.5 Loss Functions

Loss functions encode the learning objective and guide optimization or fit of a model with respect to a dataset. Common choices include MSE/MAE for regression, cross-entropy for classification and adversarial losses in generative modelling. Below we provide descriptions for the MSE and binary cross-entropy losses which are two of the most popular loss functions for regression and classification respectively.

Mean Squared Error (MSE)

For regression tasks, the *mean squared error* compares the predicted outputs $\hat{y}_i$ with the true values y_i across N samples:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2. \tag{3.4}$$

MSE penalizes large deviations quadratically, encouraging predictions to be as close as possible to the target values. It is simple, differentiable, and widely used when the underlying data is assumed to be Gaussian distributed. We will see in Chapters 4 and 5 that MSE is used as the loss function for DeepExtractor. In both cases, the objective is to minimize the MSE between the predicted and true noise components (and additionally the signal components in the case of Chapter 5).

Binary Cross-Entropy (BCE)

For binary classification, the *cross-entropy loss* quantifies the dissimilarity between the predicted probabilities $\hat{y}_i \in [0, 1]$ and the true binary labels $y_i \in \{0, 1\}$:

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{N}\sum_{i=1}^{N}\Big[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\Big]. \tag{3.5}$$

This loss encourages the model to assign high probability to the correct class. BCE is particularly suitable when outputs are interpreted as probabilities (e.g., with a sigmoid activation, see Section 3.2.2 below), and it has a natural interpretation in

terms of maximum likelihood estimation under a Bernoulli model [186]. BCE is used for training the downstream detection models in Chapters 6 and 7, enabling us to evaluate whether the simulated datasets produced by our generative models are effective training substitutes for real glitches and signals.

3.1.6 Model Evaluation Metrics

Loss functions quantify training objectives, but evaluation metrics determine how well the trained model performs on unseen data.

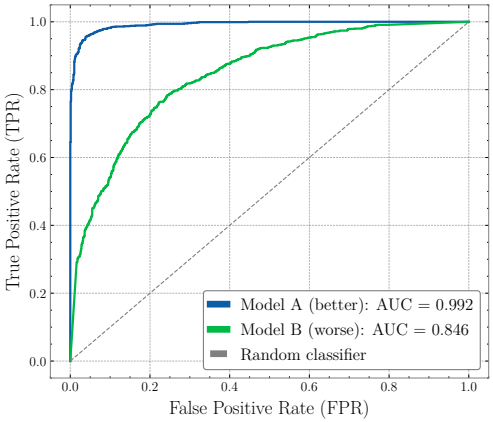


Figure 3.1: Example ROC curves for two different classification models. A classification model that curves further toward the upper-left corner has better separability and a larger area under the curve (AUC). The diagonal dashed line represents a random classifier (AUC = 0.5).

Regression Metrics. Common regression metrics include the MSE, MAE, root-mean-squared error (RMSE), and the coefficient of determination (R^2). These evaluate numerical accuracy: they measure how far predictions deviate from target values, or how much variance in the data the model explains in the case of R^2.

However, in GW physics the primary concern is not numerical deviation, but *waveform similarity*. A more appropriate metric is the *match* M—introduced in Equation 2.42 in the context of matched filtering—which quantifies how well two waveforms agree relative to the detector noise.

In this thesis we report either the match or the *mismatch*, defined as

$$\mathcal{M} = 1 - M. \tag{3.6}$$

The discretized form of the match used in practice is

$$M(\theta_1, \theta_2) = \max_t \left| 4 \sum_{k=f_{\min}}^{f_{\max}} \frac{\tilde{h}_p^*(f_k;\theta_1)\,\tilde{h}_p(f_k;\theta_2)}{S_n(f_k)}\, e^{i2\pi f_k t} \right|^2. \tag{3.7}$$

Unlike MSE, MAE, or R^2—which quantify numerical error—the match measures *scientific fidelity*: $M = 1$ means the two signals are indistinguishable in a GW detector.

Classification metrics. For binary classification, model predictions can be summarized using a *confusion matrix*, which tabulates how many samples were correctly or incorrectly classified. It compares the predicted label against the true class and categorizes each prediction as one of four outcomes:

	Predicted label	
	Positive	**Negative**
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

From these four values, several commonly used evaluation metrics are defined:

- *Accuracy*: fraction of correct predictions across all samples.
- *Precision*: among all samples predicted as positive, the fraction that are truly positive,
$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{3.8}$$
- *Recall* (also called *sensitivity* or *true positive rate*): among all truly positive samples, the fraction that are correctly identified,
$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{3.9}$$
- *F1-score*: the harmonic mean of precision and recall, providing a balanced measure when classes are imbalanced,
$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{3.10}$$

Depending on the application, different trade-offs are desirable. For example, in GW searches it might be considered more important to maximize *recall* (i.e., do not miss any astrophysical signal), even if this increases the number of false alarms and reduces precision.

Multi-class classification. For multi-class problems, the confusion matrix generalizes to an $N \times N$ table (where N is the number of classes). Each row corresponds to the *true* class, and each column to the *predicted* class. We will see examples of such confusion matrices in Chapters 4 and 8, where Gravity Spy is used to classify DeepExtractor glitch reconstructions and cDVGAN-generated glitches respectively, allowing us to assess whether the correct labels are recovered.

Precision, recall, and F1-score also extend naturally to multi-class classification by treating each class as "positive" and all others as "negative" (a *one-vs-all* strategy). Per-class scores can then be combined either by macro averaging (averaging

the metric computed for each class, treating all classes equally) or micro averaging (summing TP/FP/FN across classes before computing the metric, weighting classes by frequency). Although these metrics are not the primary evaluation tools in this thesis, they remain standard in ML and are useful when analyzing confusion matrices or comparing classifiers on imbalanced datasets—such as glitch classification in GW data.

Receiver Operating Characteristic (ROC) curve. The ROC curve, shown in Figure 3.1, is one of the most important tools for evaluating a classifier, because it assesses performance *independently of any chosen decision threshold* (typically chosen at 0.5). It visualizes how well a model separates the positive and negative classes by plotting the *true positive rate* (TPR) against the *false positive rate* (FPR):

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \qquad \mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}. \tag{3.11}$$

The area under the ROC curve (AUC) provides a threshold-independent measure of class separability: a value of 0.5 corresponds to random guessing, while 1.0 indicates perfect discrimination between classes. We will see examples of ROC curves in Chapter 7 for evaluating binary classifiers that are trained on cDVGAN-generated data but are evaluated on real data.

Ablation Studies

To assess the contribution of individual components of a proposed model, we conduct an *ablation study*. Ablation evaluates how performance changes when specific elements of the full model are removed or simplified. Typical variants include omitting a loss term, disabling a conditioning input, modifying preprocessing, or reducing network depth.

All variants are trained under identical conditions, using the same evaluation metrics. If performance degrades, the removed component contributes positively; if not, it may be unnecessary or redundant. Ablation studies therefore provide both scientific insight (identifying which design choices matter) and practical guidance (removing complexity without loss of performance).

In this thesis, ablation studies are used in Chapter 4 to compare different target-mapping strategies across multiple neural network architectures, including DeepExtractor. In Chapters 6 and 7, ablation is used to evaluate the effect of adding auxiliary derivative discriminators in DVGAN and cDVGAN, specifically examining how these components influence the quality and realism of the generated GW data.

3.2 Neural Networks and Deep Learning

Artificial neural networks (ANNs) form the backbone of modern deep learning and are inspired by the interconnected structure of biological neurons [186, 187, 192]. Their key strength lies in the ability to learn hierarchical representations of data directly from examples. They can be used in any of the paradigms described above i.e. supervised, unsupervised, semi-supervised etc.

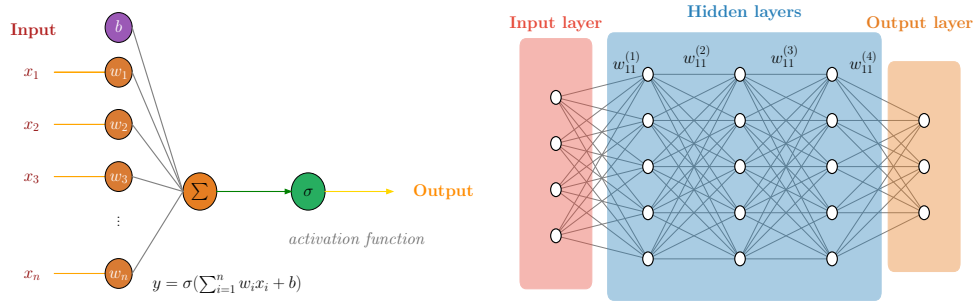


Figure 3.2: *Left:* The perceptron model described by Equation 3.12, which computes a weighted sum of the inputs (i.e. w_i) together with a bias term b before applying a nonlinear activation. *Right:* A feedforward neural network with three hidden layers, or multilayer perceptron (MLP), which extends this concept by interconnecting many perceptrons across layers.

3.2.1 From Perceptrons to Deep Networks

The simplest neural unit is the *perceptron*, introduced by Rosenblatt [193]. A perceptron maps an input vector $x \in \mathbb{R}^d$ to an output through a weighted sum followed by a non-linear activation function σ:

$$y = \sigma\left(\sum_{i=1}^{n} w_i x_i + b\right), \tag{3.12}$$

where w_i are learnable weights and b is a learnable bias term that allows the perceptron to represent functions that do not pass through the origin. Stacking multiple perceptrons in layers yields a *multilayer perceptron* (MLP), also referred to as a *fully connected* or *dense* neural network. When many such layers are combined, the resulting architecture is known as a *deep neural network* (DNN), which can approximate highly complex and non-linear functions (i.e. the universal approximation theorem [194]).

3.2.2 Activation Functions

Activation functions introduce nonlinearity into neural networks, allowing them to model complex relationships that cannot be captured by linear mappings alone. Without them, a stack of multiple layers would collapse into a single linear transformation [186, 187].

Common activation functions include:

- *Sigmoid:*

$$\sigma(x) = \frac{1}{1 + e^{-x}}, \tag{3.13}$$

  which maps inputs to the range $(0, 1)$ and was widely used in early neural networks. However, it suffers from vanishing gradients for large $|x|$, slowing down learning.

- *Hyperbolic tangent (tanh):*

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \tag{3.14}$$

  centered around zero with outputs in $(-1, 1)$, improving gradient propagation relative to the sigmoid.

- *Rectified Linear Unit (ReLU):*

$$\mathrm{ReLU}(x) = \max(0, x), \tag{3.15}$$

  introduced by [195], which promotes sparse activations and alleviates vanishing gradients, becoming the default in most modern deep architectures.

- *Leaky ReLU and ELU:* small variants that retain a nonzero gradient for negative inputs, addressing ReLU's "dying neuron" problem [196, 197], due to ReLU prohibiting negative values.

Smooth nonlinearities such as the *Gaussian Error Linear Unit (GELU)* [198] are also increasingly popular, particularly in Transformer-based models [169].

The choice of activation function significantly influences optimization dynamics, gradient stability, and the representational power of the network. In practice, ReLU and its variants dominate in feed-forward and convolutional networks, while tanh and sigmoid functions remain prevalent in recurrent architectures such as *Long-Short-Term-Memory* (LSTMs) [199].

3.2.3 Training Mechanics

Neural networks are trained to minimize a loss function $\mathcal{L}(\theta)$ with respect to parameters θ (weights and biases) like those shown above. Optimization is typically performed using *stochastic gradient descent* (SGD) and its variants (e.g., Adam [200]), which update parameters iteratively as:

$$\theta \leftarrow \theta - \eta \, \nabla_\theta \mathcal{L}(\theta), \tag{3.16}$$

where η is the learning rate. Gradients $\nabla_\theta \mathcal{L}$ are efficiently computed through *back-propagation* [201], which applies the chain rule layer by layer.

3.2.4 Regularization, Overfitting, and Generalization

A key challenge in training neural networks is preventing *overfitting*, where the model memorizes the training data but fails to generalize to unseen inputs. Common regularization techniques include:

- *Weight decay (L2 regularization)*: penalizes large weights.
- *Dropout* [202]: randomly deactivates units during training to encourage redundancy.
- *Early stopping*: halts training when performance on the validation set no longer improves.

Generalization is the ability to perform well on unseen data. It is often improved by careful architecture design, data augmentation, and large, diverse datasets.

3.2.5 Deep Architectures

While MLPs are universal function approximators [194], specialized architectures are often better suited to exploit the structure of specific data types:

- *Convolutional Neural Networks (CNNs)* [203]: exploit spatial locality via convolutional filters; widely used in image and spectrogram analysis. We will describe CNNs in more detail in the next section as they are particularly relevant to this research.
- *Recurrent Neural Networks (RNNs)* and their variants (LSTMs [199], GRUs [204]) are designed to capture sequential dependencies, important for time series.
- *Transformers* [169]: rely on self-attention mechanisms to model long-range dependencies without recurrence, and have become state-of-the-art in many domains, including natural language processing and physics applications.

In summary, neural networks provide a flexible modelling framework. Deep learning refers to training such networks with many layers and specialized architectures, enabling them to automatically extract features from raw data and achieve remarkable performance across scientific and industrial domains.

Convolutional Neural Networks (CNNs)

Fully connected networks such as multilayer perceptrons (MLPs) quickly become impractical for high-dimensional inputs. For example, a single hidden layer with $m = 50$ neurons processing a $32 \times 32 \times 3$ image requires $(32 \times 32 \times 3 + 1) \times 50 \approx 1.5 \times 10^5$ learnable parameters. Moreover, flattening the image destroys its spatial correlations, forcing the network to treat distant pixels as if they were equally related and to learn many irrelevant interactions. Flattening is required for MLPs because they operate on one-dimensional vector inputs, and thus a two-dimensional (or three-dimensional, in the case of colour images) array must be reshaped into a vector before it can be processed.

Convolutional Neural Networks (CNNs) address these limitations by introducing *local connectivity* and *parameter sharing*. Instead of connecting every input pixel to every neuron, CNNs use small filters (or *kernels*) of size $k \times k$, which convolve over the input to extract local features such as edges or textures [186, 203], as shown in Figure 3.3. Each kernel learns to detect a specific pattern, and the same set of weights is applied across the entire input, drastically reducing the number of parameters.

A convolutional layer can be described by a few key hyperparameters:

- *Kernel size* k: defines the receptive field of each neuron.
- *Stride* s: determines how far the filter moves across the input.
- *Padding* p: optionally adds zeros around the input to preserve spatial dimensions.
- *Dilation* d: controls the spacing between kernel elements to enlarge the receptive field without additional computation.

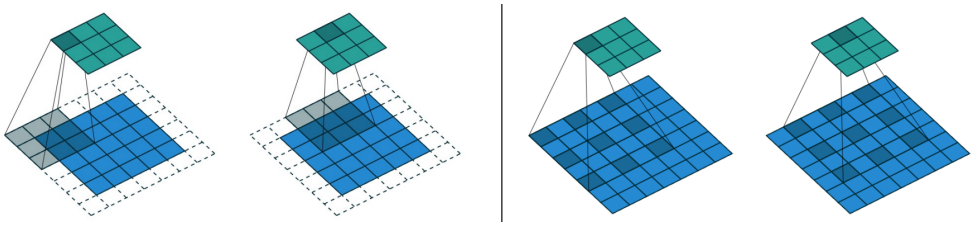


Figure 3.3: *Left:* Illustration of a convolutional kernel of size $k = 3$ applied to an input of size $6 \times 6 \times 1$, with dilation $d = 1$, stride $s = 2$, and padding $p = 1$. *Right:* Illustration of a kernel of size $k = 3$ applied to an input of size $7 \times 7 \times 1$, with dilation factor $d = 2$, stride $s = 1$, and padding $p = 0$. Adapted from [207].

Given an input of size $(h_{\text{in}}, w_{\text{in}}, d_{\text{in}})$ convolved with f filters, the output shape $(h_{\text{out}}, w_{\text{out}}, d_{\text{out}})$ is:

$$\begin{pmatrix} h_{\text{out}} \\ w_{\text{out}} \\ d_{\text{out}} \end{pmatrix} = \begin{pmatrix} \left\lfloor \frac{h_{\text{in}}+2p-d(k-1)-1}{s} \right\rfloor + 1 \\ \left\lfloor \frac{w_{\text{in}}+2p-d(k-1)-1}{s} \right\rfloor + 1 \\ f \end{pmatrix}. \tag{3.17}$$

Stacking multiple convolutional layers allows the network to build hierarchical feature representations: early layers capture low-level structures (e.g., edges), while deeper layers learn increasingly abstract concepts. CNN architectures typically include additional components such as pooling layers, batch normalization [205], and dropout [202] to improve generalization and training stability.

CNNs have become foundational in modern deep learning due to their efficiency and ability to exploit the spatial and temporal structure of data. While they are most commonly applied in computer vision, CNNs are equally well-suited to other forms of spatially correlated data, such as time series [206]. In GW analysis, they have been employed to process detector strain data for the detection of merging black holes [183]. Moreover, CNNs are particularly powerful when applied to time–frequency representations such as spectrograms, where they can identify localized features corresponding to astrophysical signals or glitches, analogous to how they detect visual features in images.

Autoencoders and U-Net Architectures

Autoencoders are a class of neural networks designed to learn efficient, low-dimensional representations of data in an unsupervised or self-supervised manner [186, 208]. They consist of two primary components: an *encoder*, which maps the input x to a latent representation z, and a *decoder*, which reconstructs the input from z:

$$z = f_{\text{enc}}(x), \qquad \hat{x} = f_{\text{dec}}(z). \tag{3.18}$$

Training minimizes a reconstruction loss, typically the mean squared error (MSE) between the input and output, encouraging the latent space to capture the most salient features of the data.

When convolutional layers are used in place of dense layers, the model is known as a *convolutional autoencoder*. Such models are particularly effective for spatial or time–frequency data, as they exploit local correlations through shared convolutional filters, much like CNNs. They have been developed for a wide range of applications, including image denoising [209], and have also been employed in GW analysis for tasks such as denoising, anomaly detection, and signal compression [210, 211].

A widely used network variant is the *U-Net* [212], originally developed for biomedical image segmentation. A schematic of a U-Net is shown in Figure 3.4. The U-Net extends the autoencoder by introducing *skip connections* between corresponding layers of the encoder and decoder, allowing fine-grained spatial information to be preserved during reconstruction. These connections concatenate encoder features with their corresponding decoder features, preserving fine-grained details while supporting a large receptive field. This architecture has made them a state-of-the-art choice for signal denoising [213], acoustic source separation [214, 215], and image denoising [216]. Unlike deep denoising autoencoders, which may lose information through aggressive compression, U-Nets preserve structural details throughout the network, enabling more accurate and faithful reconstructions [217]. In this thesis, U-Nets are employed as convolutional denoising models in the DeepExtractor framework for reconstructing GW signals and glitches from noisy time–frequency representations (see Chapters 4 and 5).

3.2.6 Generative Models

While discriminative models learn mappings from inputs to outputs, *generative models* aim to learn the underlying data distribution $p(x)$ itself, allowing the generation of new samples that resemble the training data. These models form the basis of many recent breakthroughs in unsupervised and self-supervised learning, enabling synthesis, data augmentation, and reconstruction across diverse modalities. Generative frameworks typically differ in how they parameterize and learn $p(x)$, with prominent examples including:

- **Variational Autoencoders (VAEs)** [218], which combine probabilistic inference with reconstruction losses;
- **Normalizing Flows** [219], which learn invertible transformations with tractable likelihoods;
- **Generative Adversarial Networks (GANs)** [220], which learn through adversarial optimization; and
- **Diffusion Models** [221–223], which learn to generate data by explicitly modelling the process of iteratively adding and then removing noise from samples drawn from a simple prior distribution.

We will see later in this thesis how generative modelling can be leveraged to simulate realistic GW signals and glitches in the time domain. Generating high-fidelity, synthetic time-domain signals also has applications in many other areas including medical [224] and music [225].

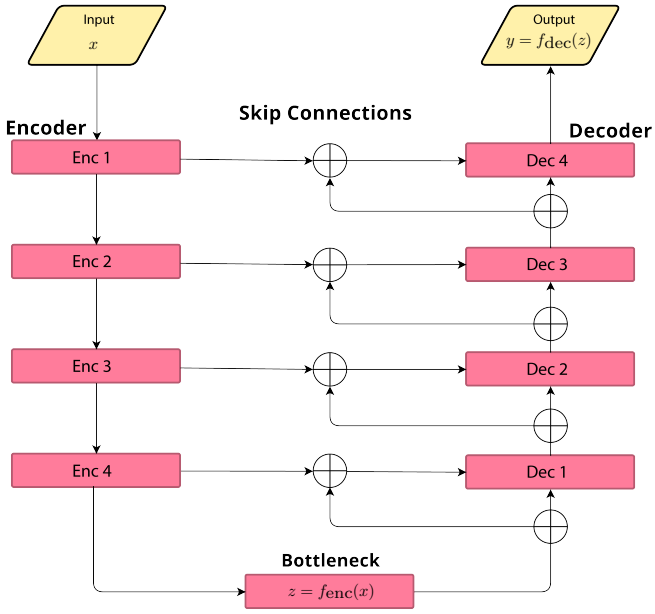


Figure 3.4: Schematic of a U-Net architecture, composed of an encoder, bottleneck, and decoder connected by skip connections that transfer feature maps between corresponding layers. When the skip connections are removed, the structure reduces to a standard autoeoncoder, where the encoder compresses the input into a latent representation in the bottleneck and the decoder reconstructs it.

Generative Adversarial Networks (GANs)

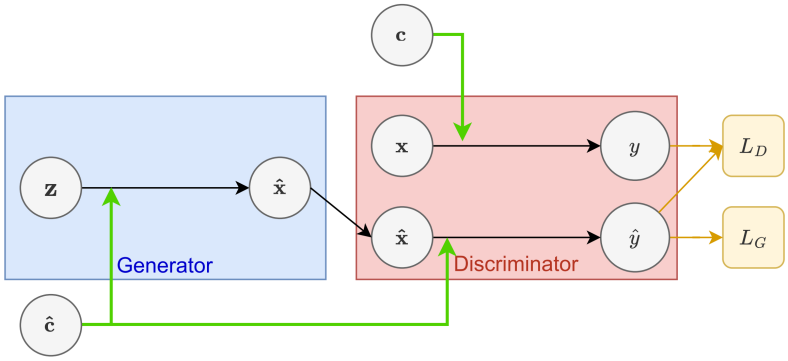


Figure 3.5: A diagram of a typical GAN architecture comprising a generator network G and a discriminator network D. Also shown are conditioned class vectors c, typically represented as one-hot encodings in the case of conditional GANs. A standard (non-conditional) GAN follows the same structure but without conditioning information provided in c.

GANs [220] are a class of generative models central to Part III of this thesis. They are composed of two neural networks trained in opposition: a *generator* G and a *discriminator* D. The generator learns to map latent variables z (sampled from a prior distribution p_z, typically Gaussian) to synthetic data $G(z)$, while the discriminator attempts to distinguish between real samples $x \sim p_{\text{data}}$ and generated samples $\hat{x} = G(z)$.

The discriminator typically outputs a probability $D(x) \in [0, 1]$ representing the likelihood that x is real. The generator aims to produce samples that maximize $D(G(z))$, effectively "fooling" the discriminator. Training proceeds as a two-player *min-max* game with the following objective function:

$$\min_G \max_D \mathcal{L}(D, G) = \mathbb{E}_{x \sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z)))]. \tag{3.19}$$

The discriminator seeks to maximize this objective by correctly identifying real and fake samples, while the generator minimizes it by producing samples that cause D to output values close to one. In the ideal case, training reaches a Nash equilibrium [226] where $p_g = p_{\text{data}}$, meaning that generated and real distributions are indistinguishable.

Loss formulation and divergence minimization. GANs can be interpreted as implicitly minimizing the Jensen–Shannon (JS) divergence between the true and generated data distributions [186]:

$$\text{JS}(p_{\text{data}} \,\|\, p_g) = \frac{1}{2}\text{KL}(p_{\text{data}} \,\|\, m) + \frac{1}{2}\text{KL}(p_g \,\|\, m), \tag{3.20}$$

where $m = \frac{1}{2}(p_{\text{data}} + p_g)$ and KL denotes the Kullback–Leibler divergence. This probabilistic interpretation links GAN training to divergence minimization in statistical inference.

Non-saturating generator loss. In practice, the original min-max generator loss (Equation 3.19) often leads to vanishing gradients when $D(G(z)) \approx 0$. To address this, a *non-saturating* variant is commonly used:

$$\mathcal{L}_G = -\mathbb{E}_{z\sim p_z}[\log D(G(z))], \tag{3.21}$$

which maximizes $\log D(G(z))$ directly, providing stronger gradients and faster convergence without changing the equilibrium.

Training challenges. Despite their success, GANs are notoriously difficult to train. The adversarial optimization can suffer from:

- *Mode collapse:* the generator produces limited sample diversity, producing only one or a few realistic samples.
- *Vanishing gradients:* the discriminator becomes too confident, halting generator learning.
- *Non-convergence:* oscillatory dynamics due to the min-max nature of the objective.

These issues stem primarily from the properties of the JS divergence used in Equation 3.19, which becomes saturated when the two distributions have little overlap.

Wasserstein GANs

Wasserstein GANs (WGANs) [227] improve training stability by replacing the JS divergence with the Wasserstein-1 distance (W_1) as the loss function, which measures the similarity between real and synthetic distributions. W_1 is continuous, differentiable, and provides reliable gradients even for disjoint distributions, avoiding vanishing gradients.

The optimization problem is formulated as:

$$\theta_{\text{opt}} = \arg\min_{\theta} \max_{\phi:||D(x,\phi)||_L \leq 1} L(\phi, \theta), \tag{3.22}$$

where θ and ϕ are the parameters of G and D, respectively. The discriminator (or critic) loss is:

$$\mathcal{L}(\phi, \theta) = \mathbb{E}_{x\sim P_x}[D(x, \phi)] - \mathbb{E}_{\hat{x}\sim P_{\hat{x}}}[D(\hat{x}, \phi)], \tag{3.23}$$

where P_x and $P_{\hat{x}}$ are the real and generated data distributions, and $\hat{x} = G(z, \theta)$.

To enforce the 1-Lipschitz constraint on D, WGAN-GP [228] introduces a gradient penalty (GP):

$$\mathcal{L}_D = -L(\phi, \theta) + \lambda GP(\phi), \tag{3.24}$$

with:

$$GP(\phi) = \mathbb{E}_{\hat{x}\sim P_{\hat{x}}}\left[(||\nabla_x D(\hat{x}, \phi)||_2 - 1)^2\right], \tag{3.25}$$

where λ is a regularization hyperparameter and $\hat{x}$ lies on a random linear interpolation between real and generated data.

The generator is updated to maximize the discriminator's response on generated data:

$$\mathcal{L}_G(\phi, \theta) = -\mathbb{E}_{\hat{x}\sim P_{\hat{x}}}[D(\hat{x}, \phi)]. \tag{3.26}$$

Conditional GANs

Conditional GANs (cGANs) [229] extend the standard GAN framework by introducing auxiliary conditioning information c (e.g., class labels or attributes) to both the generator G and discriminator D. This allows class-conditional or attribute-guided generation, where training samples are drawn from the joint distribution $P_{\text{data}}(x, c)$, and during inference, the condition $\hat{c}$ is sampled together with the latent vector z.

In the original formulation [229], conditioning is implemented by simply concatenating the class vector c with the latent vector z at the generator input and with the data sample x at the discriminator input. While effective, this concatenation-based approach provides only a limited interaction between the conditional information and learned features.

While multiple conditioning approaches have been proposed beyond naive concatenation [230–234], Miyato and Koyama [235] introduced a more principled conditioning mechanism known as the *projection discriminator*, which integrates class information directly within the discriminator's feature space as shown in Figure 3.6.

Specifically, the discriminator output is modified to:

$$D(x, c) = c^\top V h(x) + g(h(x)), \tag{3.27}$$

where $h(x)$ denotes the learned feature vector from the discriminator, V is an embedding matrix that projects the class label c into the same feature space, and $g(h(x))$ represents the unconditional discriminator output. The first term, $c^\top V h(x)$, measures the compatibility between the condition and the image features, effectively guiding the discriminator to enforce class consistency. This projection-based conditioning has been shown to substantially improve sample quality and training stability compared to simple concatenation.

In practice, the generator remains conditioned via concatenation of the class embedding with the latent vector z, consistent with the original cGAN design.

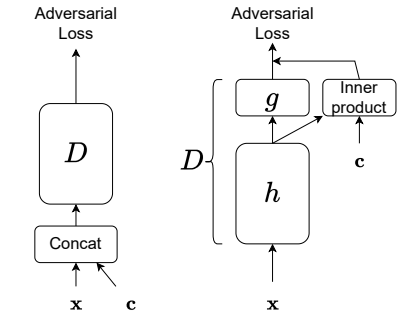


Figure 3.6: Comparison of conditioning strategies in conditional GANs. *Left:* The original cGAN discriminator, where class information is concatenated with the input. *Right:* The projection discriminator [235], which incorporates conditional information through an inner product between the class embedding ($c^\top V$)and the discriminator's learned feature vector ($h(x)$).

3.3 Machine Learning in Gravitational-Wave Data Analysis

In this section, we provide an overview of ML applications that have already been successful in GW physics. For a more exhaustive description, we recommend [185, 236] for in-depth reviews.

3.3.1 Detector Characterization

As discussed in Section 1.4, GW detectors are affected by numerous noise sources that must be monitored and mitigated to preserve detector sensitivity. Section 1.4.3 described that hundreds of thousands of auxiliary channels continuously track the detector subsystems and environment, yet identifying how disturbances in these channels couple into the strain data is often non-trivial. Many noise sources do not exhibit simple or direct correlations, making their characterization and removal an ongoing challenge.

To this end, DeepClean [237] was developed to use auxiliary channels to predict non-linear noise using a convolutional autoencoder. Another algorithmn that is actively used in LVK to supplement data analyses is iDQ [238], where an embedding of supervised ML classifiers combining auxiliary channels and the main detector strain is used to detect glitches in low latency. It notably supplemented the analysis of the BNS event GW170817, where an Extremely Loud glitch occured in LIGO-Livingston at the time of merger.

It was seen in Section 1.4.4 that deep learning based classification in the form of Gravity Spy [41, 45, 46, 239] has made significant strides in classifying the glitch space since manual characterization is unfeasible due to the abundance and diversity of glitches [240–242].

While iDQ estimates a probability for a glitch in the strain data in real time, and Gravity Spy classifies the glitch into prescribed glitch classes, GWitchHunters [243, 244] was developed for Virgo and aims to identify the likely instrumental or environmental source that is responsible for a given glitch. However, similarly to Gravity Spy, GWitchHunters requires manual human labelling to annotate the training data which carries with it several limitations. Firstly, the fixed class space is not guaranteed to be representative of all glitch morphologies, and there could be sub-classes yet to be discovered [46]. Secondly, as detector sensitivity improves and new detectors are introduced, new glitch morphologies could arise [45]. Finally, labelling enough samples to train ML models, particularly DNNs, is costly and time-consuming.

Such challenges highlight the need for unsupervised, data-driven approaches to glitch characterization that remain flexible across detectors and observing runs. One such strategy uses autoencoders to perform dimensionality reduction on glitch Q-scans, yielding compact latent representations that exhibit promising clustering behavior [245]. This enables a more data-driven and adaptive characterization of glitch morphologies, without relying on predefined labels or fixed class taxonomies. Another promising approach uses an autoencoder to compress a subset of LIGO's auxiliary channels around glitch times believed to be witnesses of certain disturbances [246]. By transforming the data using the fractal dimension [247, 248], the authors demonstrate that both the transformed data and the corresponding latent representations

can successfully cluster and separate three distinct glitch classes—Whistle, Tomte, and Scattered Light—from each other, as well as from quiet, glitch-free periods, without directly using the strain data. This result suggests that these auxiliary channels serve as viable witnesses, at least for these specific glitch types, and that they can be characterized without the need for human-labelled data.

3.3.2 Signal Denoising and Reconstruction

Recent studies have explored deep learning for denoising GW signals, focusing on isolating astrophysical events from noise in LIGO and Virgo data with promising results across various scenarios.

Recurrent autoencoders based on LSTM networks [199, 249] have demonstrated robust performance in recovering GW signals. Trained on simulated Gaussian noise, these networks excel in real, non-Gaussian LIGO noise, outperforming traditional methods like PCA and dictionary learning, especially at low SNR. Notably, the model generalizes well to eccentric GWs outside the training set, showcasing adaptability in diverse signal environments. Autoencoders using CNNs in the encoder and LSTMs in the decoder have been applied in the AWaRe framework to reconstruct GW events that align with benchmark algorithms. The framework also incorporates uncertainty estimation and can reconstruct signals that overlap with glitches [250, 251].

WaveNet architectures [211] have also been employed for GW denoising in simulated Gaussian and real non-Gaussian, non-stationary LIGO noise, achieving high overlaps ($O > 0.97$) between predicted and numerical relativity waveforms. These networks can additionally extract sine-Gaussian glitches due to structural similarities with GWs while remaining resilient to dissimilar glitches like Gaussian pulses.

U-Net models have been applied to denoise BBH signals using time-frequency representations [252]. Adapted from seismic signal separation studies [253], this approach leverages the real and imaginary parts of spectrograms to predict separate masks for signal and noise. These masks, applied to input spectrograms, allow reconstruction of the time-domain signal via inverse STFT, yielding strong performance on real astrophysical events. A U-Net-based architecture was also employed in the ALBUS framework [254] to search for minute-long GW transients by analyzing cross-correlated LIGO spectrograms, enabling pixel-wise localization of signal-like structures in the time-frequency domain.

NNETFIX [255] addresses signal reconstruction when portions are gated due to glitches. Simulating glitches of varying durations and offsets, this method effectively reconstructs BBH signals with single-interferometer SNRs above 20, even for glitches lasting hundreds of milliseconds near merger times. This highlights its relevance in the upcoming LIGO-Virgo-KAGRA observing run, where glitch overlap with longer-duration events like BNS signals is expected to increase.

3.3.3 Searches

Waveform generation has been significantly accelerated through the use of ML techniques such as Gaussian Processes [256–258], MLPs [259–264], DNNs [265–267], autoencoders [268], and transformers [269]. In addition, normalizing flows have been employed to efficiently sample templates and construct template banks with com-

prehensive coverage of the parameter space, including precessing binary configurations [270].

Several studies have applied CNNs to directly search for GWs from BBH [271–273] and BNS systems [274–277]. While these approaches offer an alternative to traditional matched-filtering methods, they currently do not match the precision of state-of-the-art modelled pipelines. Nevertheless, ML methods need not replace existing frameworks; rather, they can complement and enhance them, promoting a synergistic integration between data-driven and model-based techniques. With this motivation, the authors of [278] reformulated matched filtering as a multi-layer perceptron and optimized it via gradient descent, demonstrating improved performance over the classical implementation.

For unmodelled searches that rely on coherent excess power across detectors, various ML methods have been developed to distinguish between glitches and astrophysical burst signals [279–281], thereby improving the sensitivity of the cWB pipeline to genuine GW events. While some ML approaches have targeted generic burst searches [282–284], others have focused on specific astrophysical sources such as core-collapse supernovae [285–288], cosmic strings [289], and long-duration transient signals [290, 291]. Finally, ML has also been developed in the context of continuous wave [292–295] and stochastic background searches [296].

3.3.4 Parameter Estimation

Several studies have employed standard CNN architectures for parameter estimation [297, 298], while VAEs, such as Vitamin [299], have demonstrated improved performance in capturing posterior distributions. More recently, a number of works have explored the use of normalizing flows to reformulate Bayesian inference [300] in novel ways. For instance, Dingo [301, 302] employs an autoregressive normalizing flow for fast and accurate parameter estimation, whereas Nessai [303, 304] integrates normalizing flows within a nested sampling framework. Other studies [305, 306] combine CNN-based feature extraction with normalizing flows to enhance likelihood evaluation. Finally, truncated marginal neural ratio estimation has been introduced as an alternative simulation-based inference approach and is implemented in Peregrine for LIGO data and in Saqqara for applications to the European Space Agency's planned Laser Interferometer Space Antenna (LISA) mission [307–309].

3.3.5 Generative Models

Generative models have been developed for GW data analysis to enable computationally efficient waveform generation, facilitate realistic software simulations, and provide synthetic data for downstream applications such as class balancing, data augmentation, validation of detection pipelines via software injections [27, 310–312], and the construction of mock data challenges. Conditional autoencoders have been applied to generate accurate waveform models for non-spinning BBHs at a fraction of the computational cost required by traditional waveform models [313], while GANs have been shown to efficiently reproduce distributions of core-collapse supernova GW signals [314].

GANs have also been employed to augment glitch datasets for downstream clas-

sification tasks, though most studies have been limited to two-dimensional glitch Q-scans [315, 316]. A time-domain approach to simulating glitch events has been implemented using the gengli glitch generator [317], where a WGAN is trained on blip glitches extracted from detector background noise using BayesWave (see Section 1.4.4). The authors demonstrate that they can successfully isolate blips from their surrounding noise and learn their underlying time-domain distribution with WGANs, providing a useful framework for simulations and mock data challenges. Another study [318] proposes a cGAN called McGANn to simulate five distinct waveform classes analogous to GW bursts, while also enabling interpolation between classes to generate hybrid burst-like events that span the class space.

Part II

Noise Modelling and Signal Reconstruction

Chapter 4

Extracting Excess Power from Background Noise

GW detectors, such as LIGO, Virgo, and KAGRA, detect faint signals from distant astrophysical events. However, their high sensitivity also makes them susceptible to background noise, which can obscure these signals. This noise often includes transient artifacts called 'glitches', that can mimic genuine astrophysical signals or mask their true characteristics. In this study, we present DeepExtractor, a deep learning framework that is designed to reconstruct signals and glitches with power exceeding interferometer noise, regardless of their source. We design DeepExtractor to model the inherent noise distribution of GW detectors, following conventional assumptions that the noise is Gaussian and stationary over short time scales. It operates by predicting and subtracting the noise component of the data, retaining only the clean reconstruction of signal or glitch. We focus on applications related to glitches and validate DeepExtractor's effectiveness through three experiments: (1) reconstructing simulated glitches injected into simulated detector noise, (2) comparing its performance with the state-of-the-art BayesWave algorithm, and (3) analyzing real data from the Gravity Spy dataset to demonstrate effective glitch subtraction from LIGO strain data. We further demonstrate its potential by reconstructing three real GW events from LIGO's third observing run, without being trained on GW waveforms. Our proposed model achieves a median mismatch of only 0.9% *for simulated glitches, outperforming several deep learning baselines. Additionally, DeepExtractor surpasses BayesWave in glitch recovery, offering a dramatic computational speedup by reconstructing one glitch sample in approximately* 0.1 *seconds on a CPU, compared to BayesWave's processing time of approximately one hour per glitch.*

This chapter is based on the following publication:

Dooney et al.. *Time-domain reconstruction of signals and glitches in gravitational wave data with deep learning*, DOI: 10.1103/s91m-c2jw, Phys. Rev. D, 2025.

4.1 Introduction

GW data analysis procedures, such as those regarding the estimation of GW source parameters [319–321] (see Section 2.5), typically assume the detector noise to be stationary and Gaussian [322]. This assumption breaks down in the presence of glitches, which introduce non-Gaussian and non-stationary features in the data [27, 323, 324]. As discussed in Section 1.4.4, certain glitches can closely resemble astrophysical signals, leading to false positives [9, 32, 33, 37, 54, 58, 59, 325, 326], while others can bias parameter estimation results [36, 38, 39], a common issue observed across multiple detections [36, 106, 327–330]. Since glitches are unmodelled noise transients with diverse time-frequency morphologies, developing unified, holistic approaches for their reconstruction and mitigation is particularly challenging. Furthermore, as was discussed at the beginning of Chapter 3, the advent of next-generation detectors such as the ET and CE will introduce new challenges, requiring fast and accurate reconstruction algorithms capable of separating arbitrary signals from the detector background.

In this chapter, we present DeepExtractor, a PSD-informed deep learning framework designed to accurately reconstruct power excesses above a Gaussian detector background in the time domain. By simultaneously analyzing magnitude and phase spectrograms from the STFT, described in Section 2.4.4, DeepExtractor isolates the underlying noise from signal and glitch events using a U-Net architecture (see Section 3.2.5). The signal or glitch is then reconstructed by subtracting the predicted noise, following an additive model. Assuming additive interactions and linear approximations of signal and glitch components using time-domain wavelets, we simulate five waveform classes. By learning to isolate background noise from diverse combinations of waveforms from these classes, DeepExtractor is capable of interpolating to arbitrary signals and glitches.

DeepExtractor demonstrates superior accuracy in reconstructing simulated glitches and generalizes well to unseen distributions, outperforming several deep learning models. We also present a comparison of glitch reconstructions by DeepExtractor to those obtained using the state-of-the-art BayesWave [50], an advanced algorithm used in GW data analysis to model and subtract glitches from the strain data. We further demonstrate DeepExtractor's effectiveness in reconstructing real Gravity Spy [331] glitches from LIGO's third observing run, verified through Q-scans (examples shown in Figure 1.8) and time series plots of the detector data before and after glitch subtraction. This marks the first comprehensive application of deep learning for such tasks in GW data analysis. Finally, we use DeepExtractor to reconstruct three real GW events detected by LIGO during O3, showcasing its ability to generalize to GW signals without being explicitly trained on these waveforms. This work therefore bridges two active research directions in GW data analysis: glitch mitigation (see Section 1.4.4) and deep learning for signal reconstruction (see Section 3.3.2).

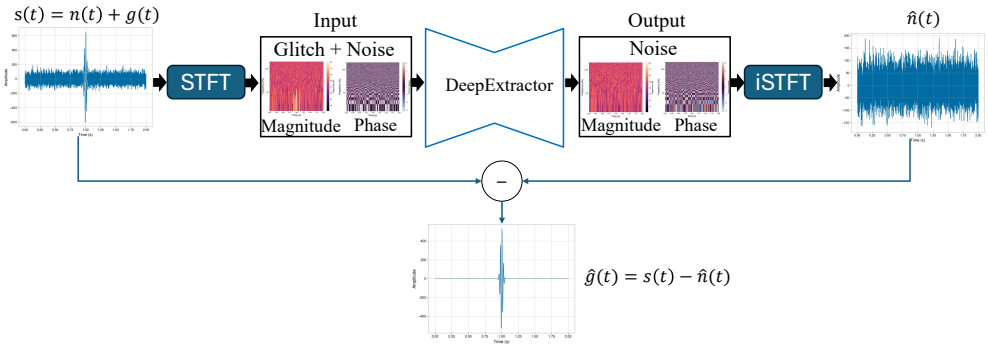


Figure 4.1: An overview of DeepExtractor's reconstruction approach. Two seconds of detector strain $s(t)$ is processed into magnitude and phase spectrogramns using an STFT. This is fed through our network, which maps the instance to the magnitude and phase spectrograms of the underlying background noise. We then use an inverse STFT to yield the corresponding time series $\hat{n}(t)$. An estimation of the underlying signal or glitch can be simply calculated as $\hat{g}(t) = s(t) - \hat{n}(t)$.

4.2 Methods

4.2.1 DeepExtractor

Overview of framework

DeepExtractor is a deep learning framework designed to reconstruct signals or glitches in GW data with power exceeding the underlying detector noise. An overview of the framework is illustrated in Figure 4.1. Built using PyTorch's deep learning library [332], the DeepExtractor codebase is publicly accessible on GitLab[1]. DeepExtractor is agnostic to the origin and morphology of excess power in the detector, whether it is a signal or a glitch. However, in this work, we primarily focus on the application of glitch reconstruction.

The framework operates under the first two models of BayesWave described in Section 1.4.4, where the data contains either an astrophysical signal (signal-only mode) **or** a glitch (glitch-only mode). In the next chapter, we present an approach to extend DeepExtractor to handle the third model assumption, where a glitch overlaps with an astrophysical signal (signal+glitch mode).

We follow the assumption that a glitch $g(t)$ interacts additively with the strain data in a real detector setting. Our goal is to map the input mixture $s(t)$ to the noise component $n(t)$, where $s(t)$ is represented as:

$$s(t) = n(t) + g(t) \tag{4.1}$$

We can then subtract the model output $\hat{n}(t)$ from $s(t)$ to yield an estimate of the underlying signal or glitch embedded in the detector background via $\hat{g}(t) = st) - \hat{n}(t)$.

[1] `https://git.ligo.org/tom.dooney/deepextractor.git`

Training strategy

The DeepExtractor framework utilizes a neural network that directly outputs the background noise component from the data in the spectrogram domain. Our core idea is that the distribution of whitened background, which is approximately Gaussian stationary, is easier to model by our network compared to modelling the diverse classes of glitches. A similar residual learning approach has been used to denoise endoscopic images by predicting and subtracting the noise component, demonstrating superior generalization in the presence of spatially varying noise [333]. Thus, to train DeepExtractor, access to the background noise target $n(t)$ is required.

To this end, we generate our own synthetic training data. We start with a background noise target $n(t)$, which represents 2 s of detector data. Next, we inject synthetic glitches $g(t)$ into $n(t)$ to create the input mixture $s(t)$, following Equation 4.1. A more detailed discussion on the process of synthesizing the data can be found in Section 4.2.2.

Neural networks can learn in either the time or time-frequency domain, each with trade-offs [334]. DeepExtractor is trained in the time-frequency domain using invertible STFTs, which provide rich signal representations that enhance learning [335–338]. While many STFT-based models use only the magnitude spectrogram—assuming identical mixture (input) and source (target) phases [339]—recent work shows that incorporating phase via complex spectrograms improves performance [340–342]. Accordingly, we model both magnitude and phase components of the complex STFT spectrograms to enable accurate time-domain reconstruction of signals and glitches.

To provide the inputs and targets, we first transform the time-domain signals $s(t)$ and $n(t)$ into the time-frequency domain by computing their complex spectrograms, $s(t, f)$ and $n(t, f)$, using the STFT (see Section 4.2.2). The spectrograms are then decomposed into magnitude and phase components, each with dimension $\mathbb{R}^{h \times w}$. These are then processed in two separate input and output channels of the network, using a data structure of dimension $\mathbb{R}^{2 \times h \times w}$ (see Figure 4.2). Thus, DeepExtractor takes the magnitude and phase components of $h(t, f)$ as input and outputs the corresponding components of the estimated noise $\hat{n}(t, f)$.

We optimize the network by minimizing a mean squared error (MSE) loss function (see Section 3.1.5), which compares the predicted components $\hat{n}(t, f)$ with the true background noise components $n(t, f)$, as defined below:

$$\frac{1}{N} \sum_{i=1}^{N} \left(\hat{n}_i(t, f) - n_i(t, f) \right)^2 , \tag{4.2}$$

where i is the index of the data sample and N is the batch size.

During inference, the predicted magnitude and phase components of $\hat{n}(t, f)$ are recombined into a complex spectrogram, which is then transformed back to the time domain using the inverse STFT to produce the noise estimate $\hat{n}(t)$. By subtracting $\hat{n}(t)$ from the input mixture $s(t)$, the framework isolates the underlying signal or glitch, $\hat{g}(t)$. This approach ensures the network focuses on learning the simpler distribution of Gaussian noise, avoiding the complexities associated with directly modelling the diverse combinations of glitches present in the data.

DeepExtractor employs the Adam optimizer [200] with a batch size of 32. Losses for DeepExtractor and other time-frequency models are calculated directly on spectro-

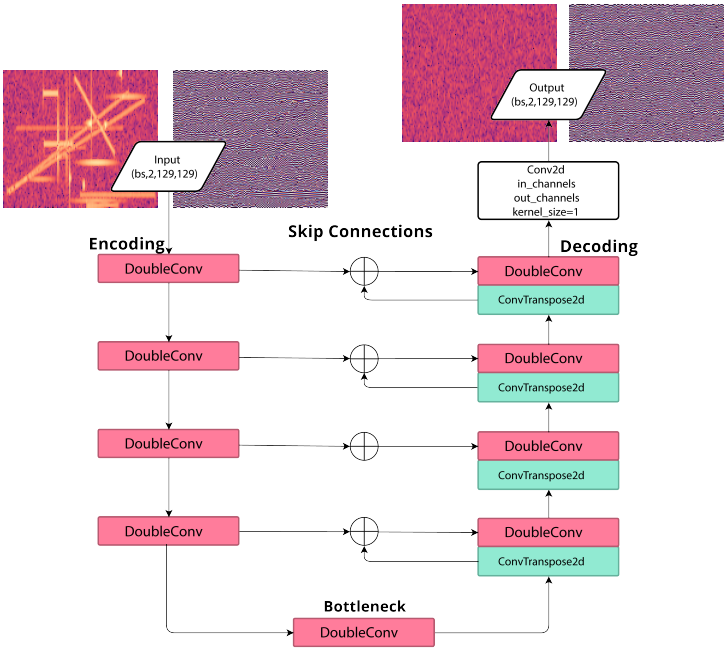


Figure 4.2: The U-Net architecture featured in DeepExtractor applied to batches (bs) of STFT data ($\mathbb{R}^{2\times h\times w}$), illustrating its characteristic 'U'-shaped structure. The network processes both the magnitude and phase components of the STFT simultaneously through two input and output channels.

gram data, while for time-domain benchmarks, they are computed on raw time-series data. To optimize the learning process, the training incorporates the `ReduceLROnPlateau` scheduler provided by Pytorch, which reduces the learning rate after four epochs without improvement, and early stopping is applied after ten such epochs. This ensures an efficient yet flexible training duration tailored to the model's performance on validation data.

All models in this study were trained on an NVIDIA A100 GPU (80GB memory). Our best-performing DeepExtractor configuration trained on the simulated dataset (see Section 4.2.2) in approximately 8 hours, converging after 24 epochs. The subsequent transfer learning phase on real LIGO data (see Section 4.2.2) required only 3 additional epochs to converge and completed in under 20 minutes.

U-Net Architecture

DeepExtractor employs a U-Net architecture [343], described in Section 3.2.5, for its ability to capture both high- and low-level features through skip connections. U-Nets are particularly well-suited for spectrogram-based representations of signals and glitches that span wide time–frequency ranges, where conventional deep autoencoders may struggle due to information loss from aggressive compression [217].

DeepExtractor is a 2D model designed to process magnitude and phase spectrograms. It accepts two input channels and outputs two spectrograms of the same dimensionality, as illustrated in Figure 4.2.

Encoder (Downsampling Path): The encoder extracts hierarchical features from the input through:

- **DoubleConvolution Blocks:** Each block consists of two 2D convolutional layers (kernel size 3), each followed by batch normalization [205] and ReLU activation [195], to enhance feature extraction while preserving spatial resolution.
- **MaxPooling:** A 2D max-pooling operation (kernel size 2) reduces spatial resolution by half after each DoubleConvolution block, enabling hierarchical feature representation.

Feature maps from the encoder are stored as skip connections for the decoder.

Decoder (Upsampling Path): The decoder reconstructs the original input resolution while integrating features from the encoder:

- **Transposed Convolutions:** Upsampling is performed using transposed convolutional layers that double the spatial resolution.
- **Skip Connections:** Features from the encoder at corresponding resolutions are concatenated with the upsampled features to preserve fine details.
- **DoubleConvolution Blocks:** Combined features are processed through DoubleConvolution blocks for refinement.

Bottleneck: At the deepest level, the bottleneck processes the feature map using a DoubleConvolution block, capturing the most abstract representations of the input.

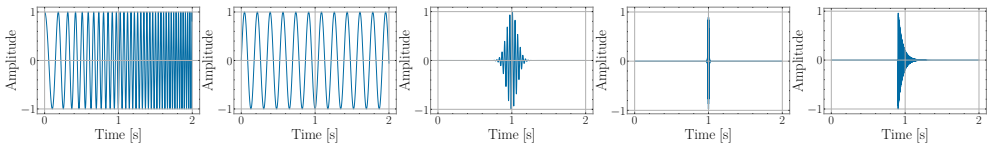
Output: The model concludes with a single 2D convolutional layer (kernel size 1) with a linear activation function, producing outputs with the desired dimensionality.

Time-Domain Variant: We also experiment using a 1D time-domain counterpart of DeepExtractor's U-Net architecture (UNET1D). This replaces all 2D operations with 1D equivalents and processes a single input/output channel. Specifically, the 1D model maps the raw, whitened time series $s(t)$ to the noise component $n(t)$, omitting the STFT and iSTFT transformations used in the 2D variant, as shown in Figure 4.1.

4.2.2 Datasets

Simulated data

To properly validate our model, we first implement a fully simulated approach for both training and validation, simulating the background noise and glitches. We simulate white noise training targets $n(t)$ with a duration of 2 seconds at a sampling rate of $f_s = 4,096$ Hz, yielding 8,092 data points. We generate 250,000 background samples for training and 25,000 for validation.



(a) Chirp (b) Sine (c) Sine-Gaussian (d) Gausian Pulse (e) Ringdown

Figure 4.3: Examples of each of the five training glitch classes; chirp, sine, sine-gaussian, gaussian pulse and ringdown. For each 2 s training sample, anywhere between 1 and 30 of these signals (selected randomly) are injected into the Gaussian background sample.

To create the corresponding input samples, we inject various analytical waveforms, represented by $g(t)$, into the background noise $n(t)$, as per Equation 4.1. While BayesWave models glitches exclusively as linear combinations of sine-Gaussian wavelets, our approach broadens the space of basis waveforms to improve the network's generalization capability. Specifically, we draw from five proxy glitch classes—*chirp*, *sine*, *sine-Gaussian*, *Gaussian pulse*, and *ringdown*—illustrated in Figure 4.3.

The chirp class was selected for its resemblance to the frequency evolution characteristic of CBC signals. The sine class captures narrow-band features across varying frequencies. The sine-Gaussian class, inspired by BayesWave, along with the Gaussian pulse and ringdown classes, serves as an analytical proxy for GW signals expected from a variety of burst sources [344]. Since these waveforms are commonly used in unmodelled burst searches, we investigate their suitability for unmodelled glitch reconstruction in our DeepExtractor framework.

Each waveform class has its own parameters that are randomly sampled during generation, such as frequency range (from 1 Hz to the Nyquist frequency of $2,048$ Hz), bandwidth, and duration (randomly chosen between 0.125 s and 2 s). The lower bound of 0.125 s was chosen since over 90% of high-confidence Gravity Spy glitches exceed this duration. Additionally, the analytical waveforms used for training often contain features—such as individual sine-Gaussian cycles—that evolve on timescales shorter than 0.125 s. While shorter durations could be considered, this threshold is expected to capture the majority of short-duration structures relevant for glitch reconstruction. For more information on these training classes, please see Appendix A.1 and our code repository. The SNR is varied between 1 and 250, scaled according to

$$\rho^2_{opt} = 4 \int_{f_{low}}^{f_{high}} \frac{|g(f)|^2}{S_n(f)} df \tag{4.3}$$

where $g(f)$ and $S_n(f)$ represent the Fourier transform of the injected waveform and the detector noise PSD, respectively [345]. This expression is equivalent to that in Equation 2.36, differing only by a constant scaling factor and by substituting the template signal h with our hypothetical glitch g over the frequency interval $[f_{\text{low}}, f_{\text{high}}]$. We set $S_n(f)$ to unity for convenience, since we are working with simulated whitened detector noise with a flat PSD.

To improve generalization and avoid overfitting, we inject linear combinations of glitches into each background sample, where:

$$\mathbf{g} = \sum_{i=1}^{k} g_i(t) \tag{4.4}$$

and $k \in [1, 30]$. Futhermore, each $g_i(t)$, with $i \in [1, k]$, is randomly time-shifted within the 2 s duration. This procedure produces a diverse set of composite proxy glitches, which may or may not overlap in time and frequency. In the context of the training data, $g(t)$ in Equation 4.1 can be replaced with $\mathbf{g}$ to reflect this linear combination. Inspired by BayesWave's approach of modelling glitches as sums of sine-Gaussian wavelets, our setup injects between 1 and 30 glitches per sample, drawn from a broader five-class proxy glitch space.

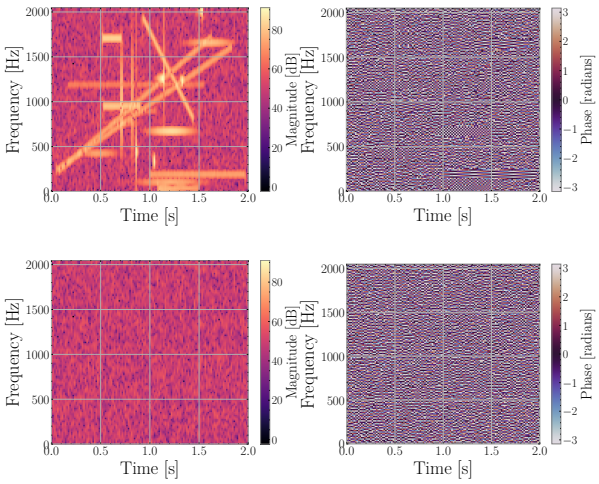
Additionally, glitches can be scaled to low SNRs, so some training samples include injections with very little (or even negligible) contribution to the data. We hypothesize that this approach helps the model to learn features inherent to the background noise when glitches are not present in the data. We test this hypothesis on both simulated (Section 4.2.3) and real detector noise (Section 4.2.3).

To prepare the data for training DeepExtractor, we scale (standardize) the time-domain inputs and outputs using `StandardScaler` from sklearn [346], which transforms the data so that it has a mean of 0 and a standard deviation of 1 (see Section 3.1.4). This standardization method assumes a Gaussian distribution of the data and is appropriate given the Gaussian-like characteristics of the background noise.

We fit the scaler exclusively to the model input (i.e., the glitch-injected samples) and use this input-fitted scaler to transform both the input and the background noise target. This ensures that both inputs and outputs occupy the same standardized space. While this data must be then transformed using the STFT to train DeepExtractor, it is this preprocessed data that is used to train the time-domain benchmarks (eg. UNET1D).

Short-time Fourier transform (STFT): As introduced in Section 2.4.4, the STFT is used to compute spectrograms from the preprocessed data. These spectrograms form the training inputs for DeepExtractor and the 2D benchmark models, as discussed in Section 4.2.1. Since the time-domain data is already standardized, no additional scaling is performed on the spectrograms. Figure 4.4 shows examples of the magnitude and phase components for both the input and target spectrograms.

We choose STFT parameters to ensure compliance with the Constant Overlap-Add (COLA) condition [347] (please see Appendix A.2 for more information). A Hann window (see Section 2.2.1 for details on windowing) is employed to minimize spectral leakage, further improving the quality of the spectrogram representation [348]. These chosen settings yield complex spectrograms with dimensions of 257×257, which we separate into magnitude and phase spectrograms. To investigate the effect of varying time and frequency resolutions on signal and glitch reconstruction, we also experiment with alternate parameter configurations that produce spectrograms of dimensions 129×129 and 65×129 while ensuring that all configurations satisfy the COLA condition.



(a) Magnitude (b) Phase

Figure 4.4: Magnitude and phase STFT spectrograms for noise+glitch input (top) and noise-only target (bottom) used to train DeepExtractor.

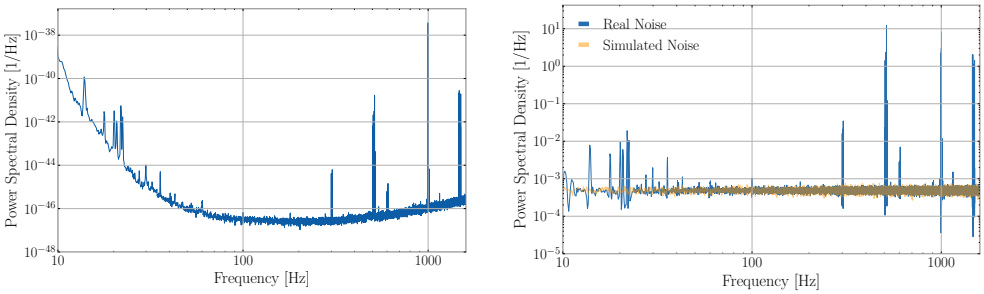
Real LIGO O3 backgrounds

GW detector noise is often modeled as Gaussian and stationary over short timescales; however, as shown in Section 1.4, it is generally non-Gaussian and exhibits a time-varying PSD. The sensitivity of GW detectors is constrained by broadband noise sources, such as seismic noise at low frequencies and quantum sensing noise at high frequencies [322]. An example PSD from LIGO Hanford is shown in Figure 4.5. The right plot compares the PSD of whitened data using Welch averaging [134], as described in Section 2.3.2, to that of simulated white noise described in the previous section. While whitening reduces broadband noise, narrow-band spectral lines-as discussed in Section 1.4.3-persist. In contrast, the simulated noise PSD shows a smooth, flat trend with no spectral lines, underscoring the complexities of real GW detector noise.

To accurately reconstruct signals and glitches in real GW data, it is essential to account for these unique noise features. A model trained exclusively on simulated data will inadvertently suppress spectral lines when applied to real data, leading to contaminated reconstructions of both signals and glitches. Therefore, incorporating real data into the training of DeepExtractor is crucial for adapting to the non-Gaussian, non-stationary nature of GW detector noise and ensuring accurate modelling of real events outside the training distribution.

To address this challenge, we apply transfer learning [349] to our DeepExtractor model, leveraging knowledge learned from simulated data to enhance performance or reduce training effort on real data. Transfer learning allows a model trained for one task to adapt to a related task, making it an effective strategy in this context.

We restrict the scope of this study to LIGO data during O3. We accumulate real backgrounds by identifying GPS times for all GW events and Omicron triggers. Background segments of 14 s are extracted from the *Gravitational Wave Open-Science*



(a) (b)

Figure 4.5: Power spectral density (PSD) of **(a)** real LIGO Hanford detector noise and **(b)** the same PSD after whitening, compared with our simulated white noise. The simulated noise lacks the instrumental and environmental lines characteristic of real detector data.

Centre (*GWOSC*) [350] data during O3a and O3b, ensuring these segments are at least 2 s away from any trigger. These segments are whitened using Welch's method (see Section 2.3.2), with the first and last 2 s discarded to mitigate boundary artifacts. The resulting 10 s segments are divided into 2 s samples. We perform a final validation step using Q-scans for each 2 s sample, discarding any sample with a maximum pixel (tile) energy — defined as the squared magnitude of the Q-transform coefficient for a time-frequency tile — exceeding 20 (see [351] for details on pixel energy). This ensures that no sample contains excessive power that could interfere with the training of DeepExtractor. After applying this criterion, we collect a total of $45,400$ background samples from LIGO's O3 run, distributed as follows: $18,400$ from Hanford O3a, $1,300$ from Livingston O3a, $18,600$ from Hanford O3b, and $7,100$ from Livingston O3b[2]. For injections, we follow the approach described in Section 4.2.2, with the key distinction that the injections are now added to real, whitened O3 backgrounds.

Unseen Test Classes

To evaluate the generalization capability of DeepExtractor, we employ two independent glitch generators: gengli [317] and cDVGAN [352], the latter being the focus of Chapter 7. Developed prior to this work, cDVGAN serves as an external test case for assessing DeepExtractor's reconstruction performance and is discussed in detail later in this thesis within the broader context of generative modelling for GW signals and glitches. Both of these generative deep learning models produce realistic glitches, also enabling robust testing on unseen glitch classes. Trained on Blip[3] glitches extracted from LIGO O2 data using BayesWave, gengli generates reliable representations of this specific glitch type in both Hanford and Livingston detectors separately. A conditional GAN trained on manually filtered LIGO O3 data, cDVGAN produces Blip

[2]While Livingston has nearly twice as many Omicron triggers as Hanford, a significant portion of Livingston background samples were also excluded during the Q-scan validation due to persistent excess power, even outside the Omicron triggers.

[3]https://gswiki.ischool.syr.edu/en/Glitches/Blip

and Tomte[4] glitches along with BBH signals with component masses in the range of $[30, 160]M_{\odot}$, a spin of zero and fixing $m_1 > m_2$. As a conditional model, cDVGAN can create hybrid samples (*simplex* and *uniform*) that blend features across all three training classes, introducing even greater diversity (see Chapter 7). By leveraging the class-mixing feature of cDVGAN, we create a diverse test dataset to assess DeepExtractor's ability to generalize to unseen classes. Examples of cDVGAN generations can be found in Figure 7.5 (Chapter 7), while gengli generations can be found in Appendix A.3.

4.2.3 Experiments

Simulated Experiments

We evaluate DeepExtractor's performance by injecting simulated glitch samples into simulated background noise. This controlled environment allows for a direct comparison between the extracted glitches and their ground-truth counterparts, helping identify the most effective data representations in terms of dimensionality and domain (time vs. time-frequency).

Simulated glitches are generated using both analytical waveforms and samples from the glitch generators discussed in the previous section. While the analytical waveforms represent known classes from training, we include unseen classes using glitches simulated with gengli and cDVGAN. Samples are injected into simulated white noise at SNRs $\in [7.5, 100]$ using Equation 4.3. We limit the maximum SNR of the test glitches to 100 (reduced from 250 in our training set) to create a controlled environment for glitch reconstruction. Furthermore, over 90% of high-confidence glitches[5] observed during O3 had an SNR below 100.

The results are evaluated using the *mismatch* metric, defined as

$$\mathcal{M}(g_1, g_2) = 1 - M(g_1, g_2) \tag{4.5}$$

where M denotes the match, the same quantity introduced in Section 2.4.2 used to quantify the similarity between a signal and a template within a template bank.

We evaluate the performance of DeepExtractor by comparing it against several deep learning benchmarks. These include a 1D denoising CNN (DnCNN1D) [353] and 1D and 2D autoencoders that are structurally similar to the U-Net, but without skip connections (see Section 3.2.5). We also assess the performance of the time-domain variant of DeepExtractor's U-Net architecture (UNET1D) to compare its effectiveness across domains. This comparison helps evaluate DeepExtractor's performance relative to other denoising and feature extraction architectures. Furthermore, we examine the performance of both DeepExtractor and UNET1D when directly mapping to $g(t)$, investigating the effectiveness of our residual approach in first mapping to $n(t)$ and subtracting it to yield $\hat{g}(t)$. Finally, we apply DeepExtractor to background samples without any injected signals, where the ideal outcome is for the output to reproduce the input and the inferred glitch component $\hat{g}(t)$ to vanish. This is important as it verifies that the model does not hallucinate or erroneously reconstruct glitches (or signals) in purely noise-dominated data..

[4] https://gswiki.ischool.syr.edu/en/Glitches/Tomte

[5] Those classified with a Gravity Spy confidence of at least 90%.

Glitch Type	UNET1D	DnCNN1D	Autoencoder1D	Autoencoder2D	DeepEx. (65x129)	DeepEx. (129^2)	DeepEx. (257^2)
chirp	$2.0^{+2.1}_{-0.8}$	$26.1^{+20.8}_{-13.9}$	$2.3^{+2.4}_{-0.9}$	$41.2^{+20.1}_{-9.3}$	$1.9^{+1.7}_{-0.6}$	$1.1^{+0.9}_{-0.4}$	$\mathbf{0.8}^{+0.7}_{-0.3}$
sine	$1.7^{+2.8}_{-0.6}$	$23.5^{+27.6}_{-11.5}$	$1.9^{+3.7}_{-0.7}$	$41.1^{+20.2}_{-9.7}$	$1.7^{+1.8}_{-0.6}$	$0.8^{+1.0}_{-0.2}$	$\mathbf{0.5}^{+0.7}_{-0.1}$
sine_gaussian	$1.4^{+1.8}_{-0.6}$	$8.9^{+13.5}_{-3.3}$	$1.6^{+2.0}_{-0.7}$	$40.2^{+16.3}_{-11.3}$	$1.7^{+1.8}_{-0.8}$	$0.6^{+0.7}_{-0.2}$	$\mathbf{0.4}^{+0.6}_{-0.1}$
gaussian_pulse	$\mathbf{0.4}^{+0.4}_{-0.2}$	$3.8^{+3.8}_{-1.7}$	$0.6^{+0.7}_{-0.3}$	$37.7^{+13.9}_{-11.5}$	$1.8^{+2.1}_{-0.8}$	$0.6^{+0.7}_{-0.2}$	$\mathbf{0.4}^{+0.4}_{-0.2}$
ringdown	$\mathbf{0.6}^{+0.8}_{-0.2}$	$5.2^{+4.3}_{-2.1}$	$0.9^{+1.0}_{-0.4}$	$39.3^{+15.0}_{-9.5}$	$2.2^{+2.4}_{-1.0}$	$0.9^{+0.8}_{-0.4}$	$0.7^{+0.6}_{-0.3}$
gengli_H1	$1.0^{+1.0}_{-0.3}$	$3.8^{+3.3}_{-1.4}$	$1.2^{+1.1}_{-0.4}$	$36.6^{+13.0}_{-9.0}$	$2.7^{+2.0}_{-1.1}$	$1.2^{+0.8}_{-0.4}$	$\mathbf{0.9}^{+0.7}_{-0.3}$
gengli_L1	$1.2^{+0.7}_{-0.4}$	$4.0^{+2.8}_{-1.1}$	$1.3^{+0.9}_{-0.4}$	$36.1^{+10.9}_{-7.6}$	$2.8^{+2.1}_{-1.1}$	$1.2^{+0.9}_{-0.3}$	$\mathbf{1.1}^{+0.7}_{-0.4}$
cdvgan_blip	$1.2^{+1.0}_{-0.4}$	$4.5^{+3.9}_{-1.7}$	$1.4^{+1.1}_{-0.5}$	$36.2^{+15.6}_{-9.1}$	$2.9^{+2.6}_{-1.0}$	$1.2^{+0.9}_{-0.4}$	$\mathbf{0.9}^{+0.8}_{-0.2}$
cdvgan_tomte	$1.0^{+0.7}_{-0.3}$	$4.5^{+3.4}_{-1.7}$	$1.2^{+0.8}_{-0.4}$	$38.0^{+12.4}_{-10.7}$	$2.6^{+2.5}_{-1.0}$	$1.0^{+0.7}_{-0.3}$	$\mathbf{0.9}^{+0.6}_{-0.3}$
cdvgan_bbh	$1.0^{+1.1}_{-0.3}$	$4.6^{+5.2}_{-1.5}$	$1.1^{+1.2}_{-0.3}$	$34.6^{+17.7}_{-6.7}$	$2.7^{+2.4}_{-0.9}$	$1.2^{+1.0}_{-0.5}$	$\mathbf{0.8}^{+0.9}_{-0.2}$
cdvgan_simplex	$1.8^{+1.4}_{-0.6}$	$5.2^{+5.4}_{-1.6}$	$2.0^{+1.6}_{-0.7}$	$36.9^{+16.1}_{-7.6}$	$3.5^{+3.5}_{-1.1}$	$1.8^{+1.4}_{-0.5}$	$\mathbf{1.6}^{+1.2}_{-0.5}$
cdvgan_uniform	$1.6^{+1.1}_{-0.5}$	$5.3^{+3.9}_{-1.9}$	$1.7^{+1.3}_{-0.5}$	$38.1^{+13.3}_{-10.0}$	$3.1^{+3.3}_{-1.1}$	$1.6^{+1.0}_{-0.5}$	$\mathbf{1.4}^{+0.9}_{-0.5}$
Total	$1.2^{+1.1}_{-0.4}$	$5.9^{+6.9}_{-2.4}$	$1.4^{+1.3}_{-0.5}$	$37.9^{+15.2}_{-9.2}$	$2.5^{+2.3}_{-0.9}$	$1.1^{+0.9}_{-0.4}$	$\mathbf{0.9}^{+0.7}_{-0.3}$

Table 4.1: Median mismatch (%) between injected and extracted glitches over 512 samples from each class. The bounds represent the range of the 1σ credible interval. Bold text highlights the best model for each signal type.

We explore additional target mappings for UNET1D by conducting several ablation studies (see Section 3.1.6).

Dual-Channel Output: We evaluate a dual-channel output configuration where the network predicts $\hat{n}(t)$ in one channel and $\hat{\mathbf{g}}(t)$ in the other, using both $n(t)$ and $\mathbf{g}(t)$ as targets. An extra loss ensures the sum of outputs equals the mixture input ($\hat{\mathbf{g}}(t) + \hat{n}(t) == s(t)$), enforcing consistency with Equation 4.1.

Difference Output Layer: In a variation of the dual-channel setup, we omit the explicit $\mathbf{g}(t)$ target. Instead, the second channel predicts $\hat{\mathbf{g}}(t)$ indirectly as $\hat{\mathbf{g}}(t) = s(t) - \hat{n}(t)$. This approach has been successfully applied in speech enhancement and audio source separation tasks [213, 354].

BayesWave Comparison

This experiment compares DeepExtractor with BayesWave (see Section 1.4.4), the current state-of-the-art method for glitch mitigation in GW physics. Following the identification of the optimal DeepExtractor configuration in Section 4.2.3, we conduct a controlled evaluation of both methods using simulated glitch injections.

In this experiment, we inject 30 samples from each of the 12 glitch classes mentioned in Table 4.1 (360 samples in total) into background noise at SNRs $\in [15, 100]$, then process the samples using both DeepExtractor and BayesWave for a direct, 1:1 comparison. We increase the lower SNR bound from 7.5 (as used in Section 4.2.3) to 15 to prevent potential convergence issues with BayesWave. The performance of each method is assessed by evaluating the mismatch (Equation 4.5) between the injected and reconstructed samples, following the same approach outlined in the previous section.

Gravity Spy Glitches

This experiment evaluates the performance of our model on Gravity Spy glitches, described in Section 1.4.4, from three complementary perspectives. First, we perform a qualitative assessment by visually inspecting Gravity Spy Q-scans before and after glitch subtraction, alongside the corresponding time-domain reconstructions, to

evaluate the effectiveness of our model in isolating and removing glitches. Second, we apply DeepExtractor to reconstruct a subset of 100 randomly selected glitches from each of seven of the most common glitch classes observed during O3 (25 from each observing run and each detector per class). The reconstructed signals are then reclassified using Gravity Spy. This allows us to examine whether the reconstructions preserve the representative morphology of their respective glitch classes, namely Blip, Fast Scattering, Koi Fish, Low Frequency Burst, Scattered Light, Tomte, and Whistle. Since Gravity Spy classifies glitches surrounded by detector background noise (i.e. $g(t) + n(t)$), each DeepExtractor reconstruction ($\hat{g(t)}$) is injected into a segment of 'quiet' Hanford detector noise from O3 prior to reclassification (representing $n(t)$). Finally, we embed the time-domain reconstructions produced by DeepExtractor into a three-dimensional latent space using both t-SNE and UMAP to investigate whether the resulting representations cluster consistently with the corresponding Gravity Spy labels in an unsupervised, data-driven manner. We fit both t-SNE and UMAP, described in Section 3.1.2, to a dataset consisting of over 30,000 randomly selected reconstructions from each class, and visualize 500 samples per class within the embedded space[6]. We use an extensive glitch dataset in this case, amounting to over 30,000 samples across the 7 classes that are as evenly distributed as possible across the detectors and observing runs. A breakdown of the dataset is described in Tab. 8.1 and the hyperparameters used to construct these spaces are shown in Appendix. A.4.

Since our fine-tuned DeepExtractor was trained on 2 s samples extracted from 14 s whitened segments, we need to use 14 s of real data surrounding each glitch during inference to ensure the whitening filter remains consistent. To avoid inadvertently suppressing the glitch during whitening, we estimate the PSD used for whitening from an adjacent 14 s segment via Welch's method that does not include the glitch. This approach ensures the glitch does not contaminate the PSD estimate, while still enabling effective whitening of the data. Although the adjacent 14 s segment is not vetoed for other glitches (which could result in imperfect whitening), this approach helps prevent the desired glitch from being suppressed during the whitening process.

Reconstructing O3 signals

To showcase the potential of DeepExtractor for reconstructing real gravitational wave signals in a model-agnostic way, we apply it to three events from the O3 observing run: GW190521_074359, GW200129_065458, and GW200224_222234. The component masses for these events were, respectively, 45.4 $M_\odot$ and 33.4 $M_\odot$, 34.5 $M_\odot$ and 29.0 $M_\odot$, and 40.0 $M_\odot$ and 32.7 $M_\odot$. These are among the loudest events reported during O3 that were confidently detected by both the Hanford and Livingston detectors. We follow the same preprocessing pipeline as in the previous section: whitening a 14 s segment centered around the event, and extracting a 2 s window around the merger, with the merger time aligned at 1.85 s within the segment. Similarly, we calculate the PSD from the previous 14 s segment to whiten the data, to ensure that the signal itself is not suppressed during the whitening process. We then compare DeepExtractor's reconstructions to the maximum likelihood templates obtained from parameter estimation (see Section 2.5), once again using the time-domain mismatch as the evaluation metric. The template itself is whitened using the same PSD as

[6] A smaller subset is displayed to ensure clarity and avoid visual clutter.

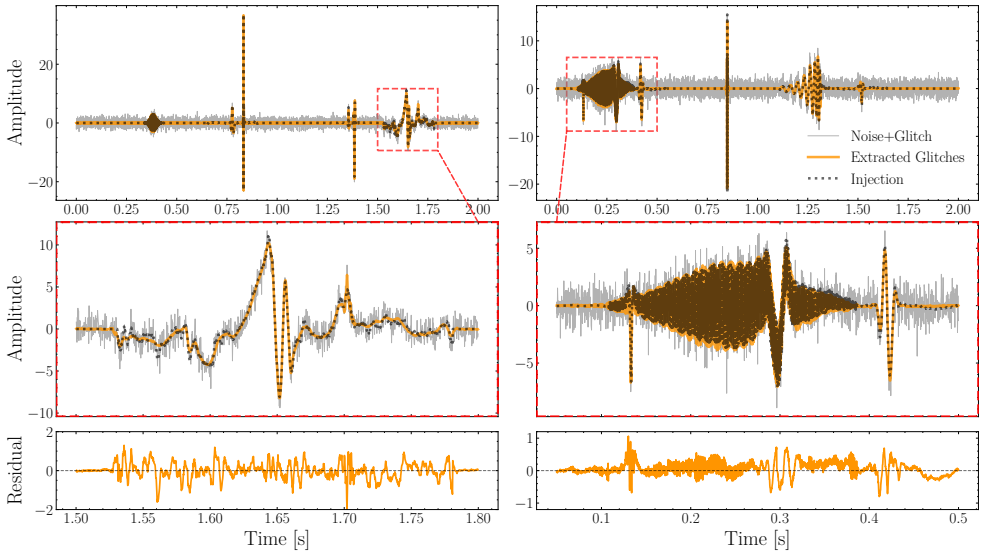


Figure 4.6: DeepExtractor reconstructions (orange) of samples comprising multiple injections (black). The upper panels display the full 2 s of whitened strain (gray), the middle panels provide an expanded view of the region outlined in red, emphasizing key features of the reconstructions and the bottom panels display the residual between the reconstructed and injected glitches.

above, since the original template is given according to coloured detector noise.

4.3 Results

4.3.1 Simulated Experiments

Overall evaluation methodology. Table 4.1 presents the median mismatch scores for each deep learning model across our test dataset, which consists of 512 samples from each class (totaling 6,144 test samples). The bounds represent the 1σ credible interval. All models in the table adhere to the framework outlined in Figure 4.1, where the background noise, $n(t)$, is first predicted and then subtracted from the input to reconstruct the glitch. The mismatch distributions are generally skewed towards low values for most models. However, a small subset of the samples includes very low SNR injections, located at the lower end of the SNR range (7.5). These low-SNR injections subtly affect the background noise, resulting in significantly higher mismatch results for these specific cases across all models. This trend is evident in the upper bounds shown in Table 4.1, which are consistently larger than the corresponding lower bounds.

Effect of Q-transform resolution. The high resolution (257x257) DeepExtractor outperforms the other deep learning benchmarks with a median mismatch score of 0.9% over the entire dataset. This marginally surpasses the (129x129) DeepExtractor, which yields a median mismatch of 1.1%, while the (65x129) DeepExtractor

yields a median mismatch of 2.5%. These results suggest that higher-resolution magnitude and phase spectrograms lead to better generalization. While using even higher resolutions may yield further gains, the diminishing improvements observed across increasing resolutions indicate that returns may taper off. When mapping directly to the glitch target $g(t, f)$ using the (129x129) architecture, we yield an overall median mismatch of $1.7^{+1.1}_{-0.6}\%$, compared to the median mismatch of $1.1^{+0.9}_{-0.4}\%$ by first mapping to the background, $n(t, f)$. This indicates that our approach of mapping to the background noise and subsequently subtracting it yields better reconstruction performance compared to mapping directly to the simulated glitch space.

Comparison with time-domain architectures. Although slightly trailing DeepExtractor in performance, the time-domain variant, UNET1D, still delivers impressive results, achieving a median mismatch of 1.2%. Similarly to DeepExtractor, mapping directly to $g(t)$ with UNET1D does not improve the performance of mapping to $n(t)$, where a median mismatch of $1.3^{+1.2}_{-0.5}\%$ is obtained across the test dataset. This highlights the utility of our framework across time and spectrogram domains. Furthermore, using both the dual-channel output and difference layer configuration make no notable improvement, with both yielding a median mismatch of 1.3%.

Autoencoder limitations. In contrast to the Autoencoder1D model, which produces mismatch results similar to UNET1D, the Autoencoder2D model exhibits notably large mismatch values between the reconstructed and injected glitches. This suggests that feature compression within the Autoencoder layers—particularly in the spectrogram domain—may be a critical issue. To explore this further, we removed one of the Autoencoder2D layers to assess whether reduced compression could lead to better results in the spectrogram domain, but this adjustment did not improve the performance. This finding highlights the importance of skip connections in U-Nets, which play a vital role in preserving key features and enhancing reconstruction accuracy.

Model behaviour on pure noise. Finally, applying DeepExtractor to 512 pure background samples without injections (i.e. where it is expected to reproduce the input) yields a maximum mismatch of 0.26% between the target and the reconstructed background noise. This demonstrates the model's ability to accurately replicate the background noise in the absence of signals or glitches, ensuring that no extraneous signals or glitches are reconstructed when they are not present in the data.

Generalization to multi-injection samples. Figure 4.6 shows reconstructions of multi-injection samples, similar to the training data, demonstrating DeepExtractor's versatility. Across 512 of such samples, DeepExtractor achieves a median mismatch of $0.7^{+0.2}_{-0.2}$. The model excels with multi-injection samples by effectively matching non-Gaussian components, which are more frequent per sample and better aligned with the training data than the single-class samples in Table 4.1[7].

[7]BayesWave was not applied to these samples, as it requires a single trigger time, while these samples have multiple trigger times. We instead maintain a controlled comparison with single-class samples, where results are easier to interpret.

4.3.2 BayesWave Comparison

Mismatch performance. Figure 4.7 presents a comparison of mismatch performance between DeepExtractor and BayesWave on a reduced dataset, consisting of 30 samples per class. Examples of reconstructions from both BayesWave and DeepExtractor are shown in Figure 4.8. DeepExtractor consistently outperforms BayesWave, even on previously unseen test classes. Specifically, DeepExtractor achieves a median mismatch of $0.3^{+0.2}_{-0.1}\%$ across the entire test dataset, showing a significant improvement over BayesWave, which has a median mismatch of $1.9^{+2.2}_{-1.1}\%$ (the bounds represent the 1σ confidence interval). The performance improvement of DeepExtractor compared to the previous section can be attributed to the lower SNR limit being raised to 15, making the reconstruction task easier.

Class-dependent behavior. The most notable improvements are observed in the chirp, sine, sine-Gaussian, and ringdown classes, where BayesWave reaches maximum mismatches as high as 81.8%, 66.8%, 72.0%, and 83.9%, respectively. In contrast, DeepExtractor shows much lower maximum mismatches of 4.0%, 3.0%, 1.9%, and 16.5% for these classes.

The discrepancies may be due to BayesWave's difficulty in converging due to high-frequency features in these samples. Furthermore, DeepExtractor demonstrates significant improvements when applied to realistic glitch samples from glitch generators, particularly hybrid samples from cDVGAN, which feature diverse glitch morphologies.

Runtime comparison. Finally, DeepExtractor also offers a substantial advantage in terms of speed, reconstructing a single glitch sample in an average of 0.14 seconds on a CPU (or 0.05 seconds per glitch when processing a batch of 256 samples). This represents a speedup of over $10,000$ times compared to BayesWave.

The significantly longer runtime of BayesWave stems from its use of RJMCMC (see Section 1.4.4) to explore a multidimensional parameter space and to construct posterior distributions. While computationally intensive, this Bayesian approach provides uncertainty estimates and flexible modelling of both glitches and astrophysical signals with minimal assumptions.

In contrast, DeepExtractor performs fast, deterministic inference via a single forward pass through a trained neural network, bypassing the need for iterative sampling or likelihood evaluation. This makes it well-suited for real-time or large-scale applications. While it trades some flexibility and uncertainty quantification for speed, DeepExtractor offers rapid, high-fidelity reconstructions of glitches in scenarios where fast turnaround is critical or full Bayesian inference is computationally prohibitive.

4.3.3 Gravity Spy Glitches

Transfer learning on real detector noise. This experiment applies transfer learning to fine-tune DeepExtractor on real LIGO O3 data from both the Hanford and Livingston detectors. The primary objective is to improve the model's ability to reconstruct real glitch events. We first take a qualitative approach by visualizing DeepExtractor's reconstruction performance through Q-scans and time-series plots.

Figure 4.9 illustrates the reconstruction of a blip glitch from LIGO Hanford during the O3a run, before and after transfer learning with DeepExtractor. This example

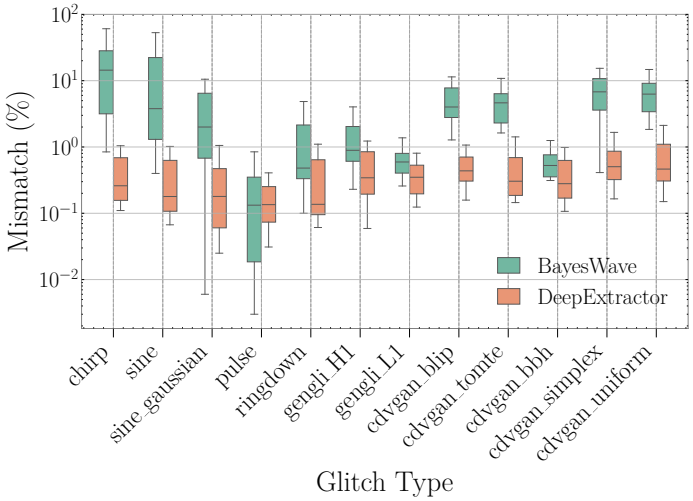


Figure 4.7: Box plots showing mismatch between injected and reconstructed samples per class. Lower boxplots for DeepExtractor indicate superior glitch reconstruction performance.

highlights the significance of further training with real detector data, as it includes features such as high-frequency lines that are common in actual LIGO data (see Section 1.4.3). Incorporating these realistic features is crucial for the model to accurately differentiate between detector noise and genuine glitch features, improving its reconstruction capabilities.

Qualitative reconstruction results across glitch classes. Figure 4.10 presents examples of DeepExtractor reconstructions following transfer learning, illustrating performance across three distinct glitch classes. Additional reconstruction examples are provided in Appendix A.5. The Q-scans show that, for most glitch classes, noise features are effectively preserved after glitch removal. However, for louder glitches, such as those from the 'Extremely Loud' and 'Koi Fish' classes, where the glitch dominates the sample, the removal process leads to a 'zeroing out' effect on the data, as seen in the Q-scans post-mitigation. However, in most cases, the time-series plots demonstrate that the reconstructions align closely with the actual data. Overall, DeepExtractor exhibits flexibility across glitch classes.

Classification evaluation using Gravity Spy. To quantitatively assess reconstruction quality, we classify the reconstructed glitches using Gravity Spy. Tables 4.2 and 4.3 summarize the results. Of the 700 reconstructed glitches, 627 are correctly identified by Gravity Spy, corresponding to $\approx 90\%$ accuracy. Correctly classified reconstructions have a mean confidence of **96.77%**, confirming that DeepExtractor effectively preserves the distinctive characteristics recognized by Gravity Spy. In contrast, misclassified samples have lower confidence (**84.84%**), indicating that the classifier confidence may serve as a useful veto.

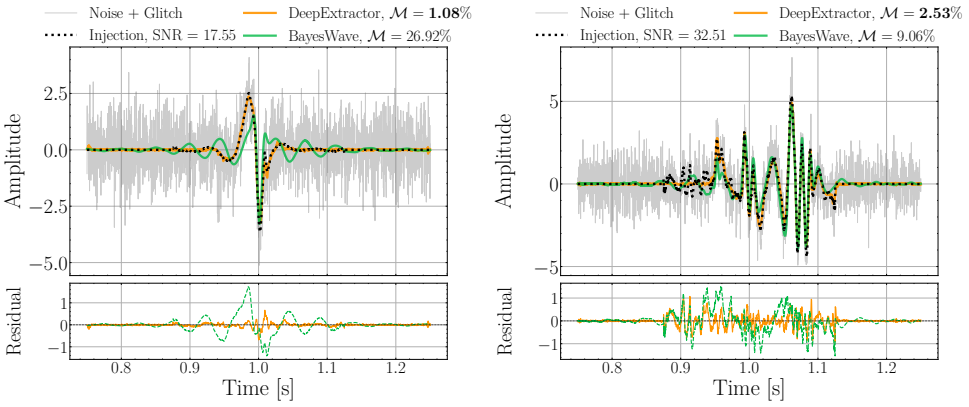
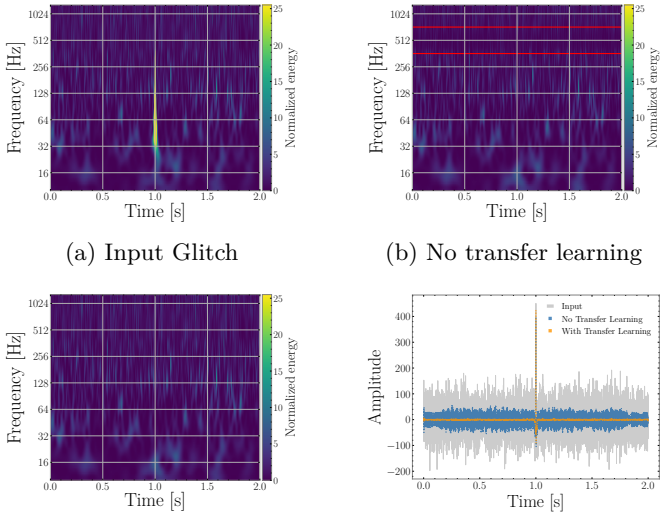


Figure 4.8: Examples from our comparison between BayesWave and DeepExtractor. Injected SNRs and mismatches yielded from both approaches are shown above each plot. The bottom panels display the residual between the reconstructed and injected glitches for BayesWave and DeepExtractor.



(a) Input Glitch

(b) No transfer learning

(c) With transfer learning

(d) Time-series outputs

Figure 4.9: An example of subtracting a Blip glitch with and without transfer learning DeepExtractor on real detector noise. Figure 4.9b shows that, without transfer learning on real detector noise, the line features at 512 Hz are also subtracted, highlighted within the red lines. This is because the model was trained on a purely flat, simulated PSD, causing the realistic line features to also be reconstructed as part of the excess power.

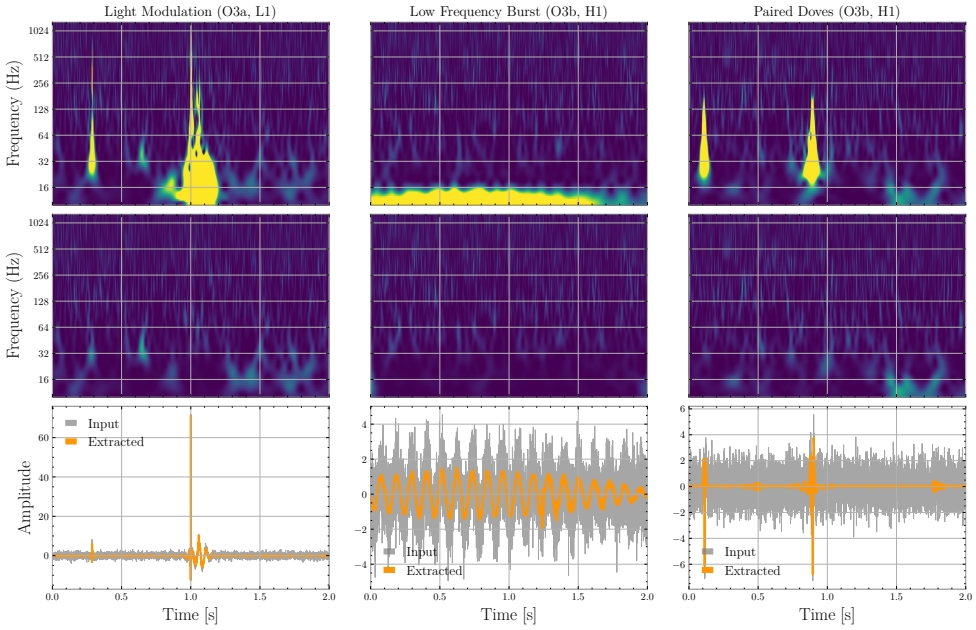


Figure 4.10: Reconstructions for three examples from distinct Gravity Spy glitch classes: Light Modulation, Low Frequency Burst, and Paired Doves. The top row displays Q-scans of the input to the network. The middle row shows Q-scans of the residual after subtracting DeepExtractor's reconstruction. The bottom row presents the corresponding time-domain input and reconstructed waveforms. The maximum color limit in the Q-scans is set to 25, similarly to the Q-scans shown in Figure 4.9.

Analysis of misclassifications. Table 4.3 shows that some classes are easier to identify than others. For example, all Blip and Low-Frequency Burst glitches were correctly classified, while Fast Scattering and Koi Fish achieved the lowest accuracies at 77% and 79%, respectively.

However, Figure 4.11[8] reveals that most misclassifications occur between morphologically similar classes. Specifically, among the 21 misclassified Koi Fish glitches, 14 were labelled as Extremely Loud. This outcome is unsurprising given that these glitches typically have exceptionally high SNRs, ranging from 220 to 1270. At such high energy levels, the large amplitudes may obscure finer morphological details typically used by Gravity Spy, leading to an Extremely Loud classification, particularly since that class lacks a well-defined morphological signature.

Similarly, the 23 misclassified Fast Scattering samples were distributed across several classes, most notably Scattered Light (7 instances), Tomte (4 instances), and Repeating Blips (3 instances). This confusion, especially with Scattered Light, is understandable given their known morphological similarities as low-frequency scattering noise types.

The 17 misclassified Scattered Light glitches were assigned either to other low-frequency categories (e.g., Fast Scattering, Low-Frequency Burst, Low-Frequency Lines), consistent with similar morphological features, or to the No Glitch class. In the latter case, this may indicate that DeepExtractor did not fully recover all excess power at very low frequencies, or that differences in the background noise segment affected the apparent morphology after injection. Overall, the remaining misclassifications are largely explainable, typically corresponding to morphologically related classes.

Unsupervised clustering with t-SNE and UMAP. The clustering results obtained using t-SNE and UMAP complement these classification findings. Figure 4.12 shows that the DeepExtractor glitch reconstructions are generally well-separated and align closely with the Gravity Spy labels, even after substantial dimensionality reduction using fully data-driven methods. Interestingly, the Blip and Koi Fish clusters exhibit partial overlap, supporting previous suggestions that these two glitch classes may be related [355]. These results demonstrate that DeepExtractor reconstructions, when combined with unsupervised clustering, could enable automated glitch characterization without human labelling while also preserving phase information, which is discarded in Gravity Spy Q-scans.

Context and limitations. Finally, it is important to contextualize the misclassifications within the Gravity Spy framework itself. As discussed in Section 1.4.4, Gravity Spy is trained on human-labelled data, which introduces a degree of subjectivity and potential labelling error. Additionally, injecting reconstructions into quiet Hanford segments may alter their interaction with background noise, producing subtle differences relative to the original events that impact Gravity Spy's classification. These considerations underscore the inherent challenges of glitch characterization and the limitations that arise from human interpretation and labelling variability.

[8] A rectangular confusion matrix is observed since only a subset of glitch classes was considered, and Gravity Spy may assign samples to classes outside this subset.

Table 4.2: Summary of Analysis Results

Total samples	700
Correct predictions	627
Incorrect predictions	73
Overall Accuracy	89.57 %
Average Confidence (Correct)	96.77 %
Average Confidence (Incorrect)	84.84 %

Table 4.3: Correct Predictions and Average Confidence per True Label

True Label	Correct Count	Avg. Confidence (%)
Blip	100	99.30 %
Fast Scattering	77	89.78 %
Koi Fish	79	91.59 %
Low Frequency Burst	100	99.85 %
Scattered Light	83	96.95 %
Tomte	96	98.21 %
Whistle	92	99.33 %

4.3.4 Reconstructing O3 signals

In this section, we present the reconstructions of three GW events detected during O3 observing run: GW190521_074359, GW200129_065458, and GW200224_222234. DeepExtractor is applied independently to whitened strain data from each detector around the time of merger.

Figure 4.13 shows the reconstructed signals for all three events in both LIGO detectors. For each case, we report the SNR, the mismatch relative to the maximum likelihood waveform [356], and the residual between the reconstructed and the maximum likelihood waveform.

Although DeepExtractor was not explicitly trained on astrophysical signals and is agnostic to their morphologies, it achieves mismatches below 10% for most cases. The exception is the GW190521_074359 event in the Hanford detector, where the mismatch reaches 16.45%.

This degradation in performance compared to the simulated setting is likely due to a misalignment between our preprocessing pipeline and that of the parameter estimation pipeline—specifically our whitening approach for both strain data and parameter estimation templates, which is necessary for input to DeepExtractor. Improving the alignment of DeepExtractor's preprocessing and training strategy with parameter estimation workflows could therefore enhance reconstruction accuracy. Additionally, incorporating physically motivated signal models into the training data, an approach explored in Chapter 5, offers further potential for improving performance.

This experiment demonstrates the promise of DeepExtractor as a model-agnostic tool for reconstructing GW signals. In cases where glitches are coincident with signals, DeepExtractor reconstructs both the signal and glitch simultaneously—a scenario we

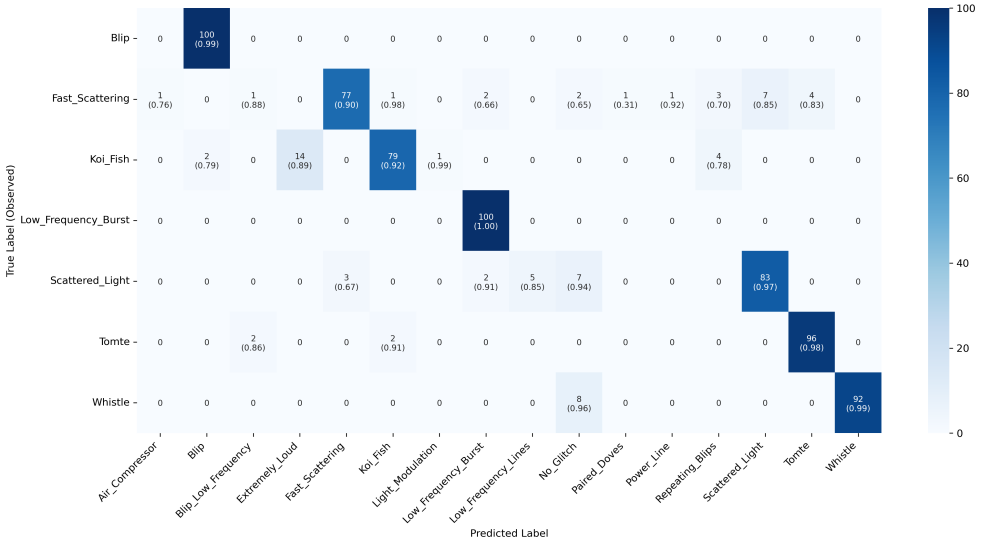


Figure 4.11: Confusion matrix for the Gravity Spy analysis on glitches reconstructed by DeepExtractor, with 100 samples per glitch class. Each square shows the number of predictions with the average classification confidence of those predictions in brackets.

explore in Appendix A.6. We will also see in Chapter 5 that incorporating signal models into DeepExtractor's training framework enables the model to separate GW signals from coincident glitches during reconstruction.

4.4 Conclusion

This chapter introduced DeepExtractor, a novel deep learning framework designed to reconstruct signals and glitches in gravitational wave detector data. Leveraging a PSD-informed U-Net architecture, DeepExtractor processes magnitude and phase spectrograms derived from the STFT. This spectrogram-based approach enables transformations between the frequency and time domains, essential for tasks such as glitch mitigation and signal reconstruction.

The key innovation lies in the training strategy. Rather than directly mapping input data to clean training waveforms, the model is trained using a residual approach to isolate the noise component from input data comprising background noise and injected signals or glitches. By subtracting the predicted noise, DeepExtractor reconstructs the signal or glitch. We show that this approach outperforms direct mapping in generalization. The framework is highly flexible, capable of reconstructing any power excess above the inherent detector noise. This flexibility stems from training on a diverse dataset of linear combinations of waveforms from five distinct analytical classes (proxy glitches), enabling effective interpolation to reconstruct unseen waveforms.

Four experiments validated DeepExtractor's efficacy. The first experiment involves simulated Gaussian noise with injected glitches, measuring time-domain mismatch

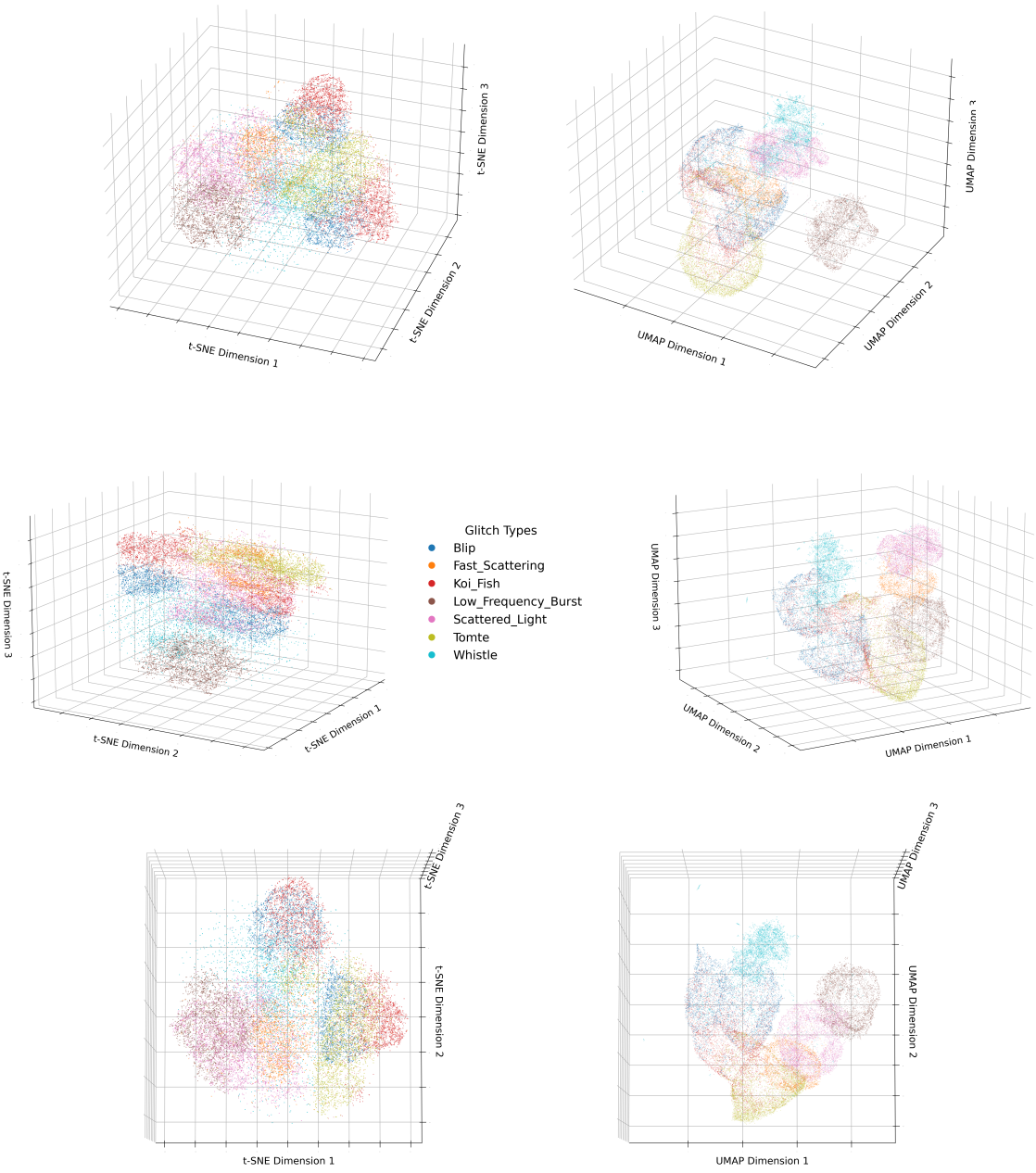


Figure 4.12: Representation of reconstructions of DeepExtractor in 3D t-SNE space (left) and 3D UMAP space (right) from different angles.

performance across several deep learning models. The results demonstrate that our U-Net architecture significantly outperforms autoencoders without skip connections, even when the overall architectures are otherwise identical. Additionally, U-Nets operating on magnitude and phase spectrograms are shown to deliver superior performance compared to a 1D U-Net applied directly to the corresponding time-series data. Finally, increasing the spectrogram resolution led to improved reconstruction performance in the time-domain. We experimented with three different resolutions and selected the highest-resolution configuration for all subsequent experiments.

In the second experiment, we compared DeepExtractor to the state-of-the-art BayesWave algorithm in a simulated environment. Our results reveal that DeepExtractor not only delivers higher accuracy and greater stability than BayesWave but also demonstrates a key advantage of deep learning over traditional Bayesian approaches: computational efficiency. While BayesWave relies on computationally intensive Bayesian inference, DeepExtractor achieves a remarkable speedup of over 10,000 times on a CPU, demonstrating its potential for real-time gravitational wave data analysis.

We evaluated DeepExtractor's capability to reconstruct real glitch events from LIGO's third observing run. This represents the first comprehensive application of deep learning to glitch reconstruction. By utilizing Q-scan and time-domain visualizations, we ensured effective glitch removal while preserving the underlying noise characteristics. The study highlighted the importance of integrating PSD knowledge through transfer learning, allowing the model to adapt effectively to the complexities of real detector data.

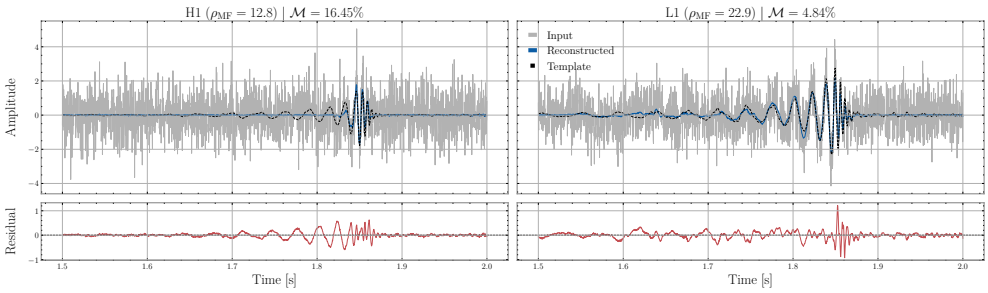
Furthermore, classifying reconstructed glitches with Gravity Spy and utilizing unsupervised algorithms strongly validate the fidelity of glitches reconstructed by DeepExtractor. The high classification accuracy of 89.6% achieved by Gravity Spy on DeepExtractor reconstructions, together with the interpretable and morphology-consistent misclassification patterns, demonstrates that the reconstructed glitches are representative of genuine transient detector artifacts. High-SNR glitches being labelled as Extremely Loud, confusion among morphologically similar classes, and the occasional misidentification of faint signals as No Glitch all align with known behavior in real detector data. Furthermore, the UMAP and t-SNE analyses revealed clear, structured clustering within the reconstructed sample, with distinct boundaries for several glitch types and partial overlap among closely related classes such as Blip, Koi Fish, and Tomte. These observations suggest that DeepExtractor not only preserves the morphological diversity of real glitches but also captures the underlying relationships between glitch families. Collectively, this establishes DeepExtractor as a reliable foundation for realistic glitch reconstruction, detector characterization, and future data-driven studies of non-Gaussian noise in GW detectors.

Finally, we applied DeepExtractor to three real GW events from LIGO's third observing run, demonstrating its potential as a model-agnostic tool for signal reconstruction. The resulting reconstructions generally showed mismatches under 10% against most of the parameter estimation templates, even without being explicitly trained on GW signals. Some of the observed mismatch is likely attributable to differences between our preprocessing pipeline and that used in parameter estimation. Aligning the preprocessing steps more closely with standard parameter estimation workflows may further enhance the reconstruction performance.

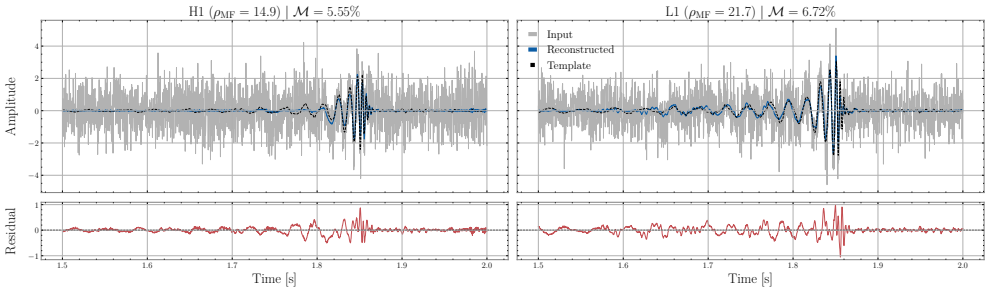
Several avenues remain open for future research. For example, modifying the training classes or tuning associated hyperparameters—such as the duration range of $[0.125, 2]$ s—could help assess whether these adjustments yield improved reconstruction performance. While BayesWave remains the most widely used tool for glitch mitigation in the GW community, future work could compare DeepExtractor with alternative approaches, such as gwsubtract, specifically in cases where glitches exhibit linear correlations with auxiliary channels. Comparisons with other deep learning–based signal reconstruction methods, such as AWaRe (see Section 3.3.2), are also of interest. However, to be applicable for arbitrary glitch reconstruction, such models would need to be adapted to incorporate DeepExtractor's training scheme or a comparable approach capable of recovering excess power directly from strain data.

A natural next step for signal reconstruction is to evaluate DeepExtractor's performance against established search and parameter estimation pipelines. While the current framework can jointly reconstruct signals and glitches when they overlap in the data, it does not yet support their explicit separation. This limitation can be addressed by extending the training framework to disentangle overlapping components—training the model not only to isolate background noise but also to reconstruct the individual glitch and signal contributions using simulated waveforms from standard signal models as supervised targets. These developments, along with the incorporation of data from multiple detectors to model coherent signal power across the network, are explored in detail in Chapter 5, where we demonstrate how such extensions can enhance reconstruction accuracy in the presence of coincident glitches and signals.

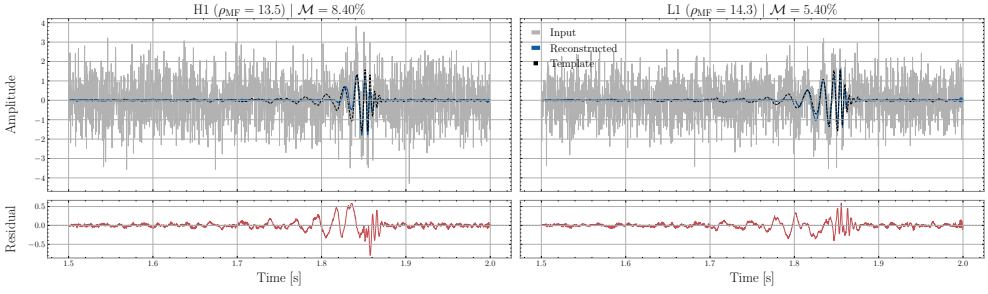
Recent research leveraging the null stream in the Einstein Telescope's triangular design has shown effective glitch mitigation in the presence of overlapping signals, marking a significant advancement for the third-generation detector era [357]. Building on this progress, DeepExtractor could further enhance the null stream's performance by precisely isolating uncorrelated glitches from signals. Integrating DeepExtractor into this framework would be a straightforward process and could strongly reinforce the scientific case for the Einstein Telescope's triangular design.



(a) GW190521_074359



(b) GW200129_065458



(c) GW200224_222234

Figure 4.13: Reconstruction of GW signals from three O3 events using DeepExtractor. Each panel shows the comparison between the reconstructed waveform, the template, and the residuals for H1 and L1 detectors. The matched-filter SNR (ρ_{MF}) in each detector frame is shown at the top of the respective plots, along with the mismatch ($\mathcal{M}$) between the reconstruction and the maximum likelihood template. We acknowledge that misalignment between our data preprocessing and that of the parameter estimation pipeline likely contributes to the observed mismatches.

Chapter 5

Separating Gravitational-Wave Signals, Glitches and Background Noise

GW detectors frequently encounter transient noise bursts, or glitches, arising from instrumental or environmental disturbances. Because glitches occur regularly, they can overlap with true GW signals and bias parameter estimation, as in the first binary neutron star event, GW170817. Accurately separating glitches from both the signal and background noise is therefore essential for reliable source parameter estimation. Even when temporally distinct from astrophysical signals, their unmodelled and variable morphology makes them difficult to reconstruct and remove. In the previous chapter, we showed that DeepExtractor can reconstruct arbitrary excess power in GW data above the Gaussian noise floor, recovering signals or glitches with minimal assumptions. It does so by modelling the detector's background noise distribution and subtracting it from the data, leaving only residual power from non-Gaussian features. Here, we extend DeepExtractor to handle data containing both signals and glitches. Using a two-channel input configuration representing LIGO Hanford and Livingston, the network maps noisy inputs to four outputs: the background noise and coherent signal components for each detector. Assuming glitches are incoherent across detectors, the model learns to reconstruct coherent GW signals while isolating incoherent glitch artifacts. Subtracting the predicted noise and signal components yields a clean glitch estimate for removal. Tests on realistic LIGO simulations show that the extended DeepExtractor effectively mitigates glitch contamination, restoring parameter estimation accuracy towards glitch-free levels.

This chapter is in preparation for submission to:

Tom Dooney et al.. *Physical Review Letters*, 2026.

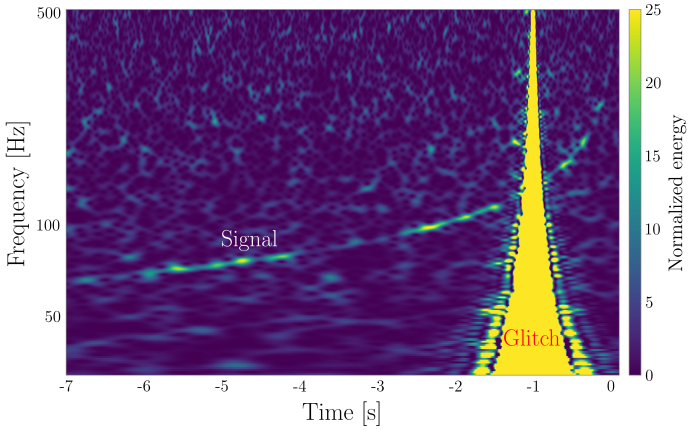


Figure 5.1: Time–frequency Q-transform of the LIGO–Livingston data around GW170817, showing a loud glitch overlapping with the BNS signal.

5.1 Introduction

As discussed in Section 2.5, extracting scientific insights from GW data fundamentally depends on accurately estimating the source parameters of detected signals [319–321]. This makes parameter estimation one of the most vulnerable stages of GW data analysis to the presence of glitches, which can bias inferred parameters when they overlap with astrophysical signals [38, 39, 358]. A well-known example is the BNS event GW170817 [106], where a loud glitch contaminated the LIGO–Livingston data near the merger, shown in Figure 5.1. The event required extensive mitigation efforts to assess its impact on the recovered parameters [36, 51]. Such occurrences are not rare: roughly 20 of the 90 confident detections reported in GWTC-3 [3] required some degree of glitch mitigation. As detector sensitivity continues to improve, these challenges are expected to become more frequent, since higher detection rates naturally increase the probability of coincident signals and glitches.

We have seen in Section 1.4.4 and the previous chapter that glitches are difficult to isolate from the detector background due to their unmodelled and highly variable nature. In the previous chapter, we demonstrated that DeepExtractor can overcome this challenge by learning to predict the background noise component $n(t)$ in the presence of diverse proxy glitch injections spanning the relevant time–frequency range, using a training dataset comprising hundreds of thousands of realistic samples. This residual learning approach encourages DeepExtractor to model the detector background—and thus the PSD—with high fidelity, enabling it to generalize to real glitch reconstructions without ever being explicitly trained on real glitch morphologies.

However, reconstructing a glitch that overlaps with a GW signal in both time and frequency presents an entirely new challenge, as it becomes difficult to avoid misattributing portions of the glitch to the signal, and vice versa [51, 52, 359]. DeepExtractor cannot distinguish between these two components, as it focuses solely on removing the background noise contribution from the data. This scenario is illustrated in Appendix A.6, where the combined signal–glitch mixture is reconstructed

from the noise background.

In this study, we extend DeepExtractor with the capability of disentangling the individual components that comprise GW data $s(t)$, namely the signal $h(t)$, glitch $g(t)$, and background noise $n(t)$:

$$s(t) = h(t) + g(t) + n(t) \tag{5.1}$$

While we demonstrated in the previous chapter that DeepExtractor can isolate $g(t)$ by predicting $n(t)$ in Equation 4.1 (i.e., assuming $h(t) = 0$), it could not separate $h(t)$ from $g(t)$ in the more general case of Equation 5.1, since it only models the noise component. We extend DeepExtractor in this study to predict both $n(t)$ and $h(t)$ simultaneously, enabling the glitch component $g(t)$ to be inferred using the same residual approach introduced in the previous chapter.

Furthermore, while the previous version of DeepExtractor operates on single-detector data, its separation mode accepts dual-detector input representing the LIGO Hanford and LIGO Livingston observatories, with the potential to be extended to additional detectors in future work. This design allows the network to model coherent signal power across the detector network and the noise component in the same way the previous version can, enabling the isolation of the glitch component via subtraction from the input.

In this preliminary study, we evaluate the performance of DeepExtractor for signal and glitch separation using simulated experiments that incorporate realistic parameter estimation simulations on two GW events before and after glitch mitigation. We inject representative *blip* glitches directly on the merger in one detector–a worst-case scenario due to the strong temporal and spectral overlap between glitch and signal. By comparing the recovered source parameters for (i) clean baseline data, (ii) glitch-contaminated data, and (iii) glitch-mitigated data, we show that DeepExtractor substantially reduces the bias introduced by glitches and restores parameter estimates toward baseline accuracy. Although these results are intentionally limited to two signals and a single glitch type, they demonstrate the viability of the approach and motivate a broader study that includes multiple glitch morphologies and a larger signal population.

5.2 Methods

5.2.1 DeepExtractor Separation Framework

The separation formulation of DeepExtractor builds on the core principles introduced in the previous chapter. In its original configuration, DeepExtractor learns to decompose an observed mixture into its underlying components using a U-Net architecture, and was primarily applied to isolating excess power (e.g., glitches) from background noise. In this chapter, we extend DeepExtractor to address a more challenging problem: separating *three* components present in GW strain data (Equation 5.1), simultaneously in two different detectors (Hanford and Livingston).

Input and output dimensionality. In the original configuration, DeepExtractor operated on 2 s of strain from a single detector and predicted only the excess-

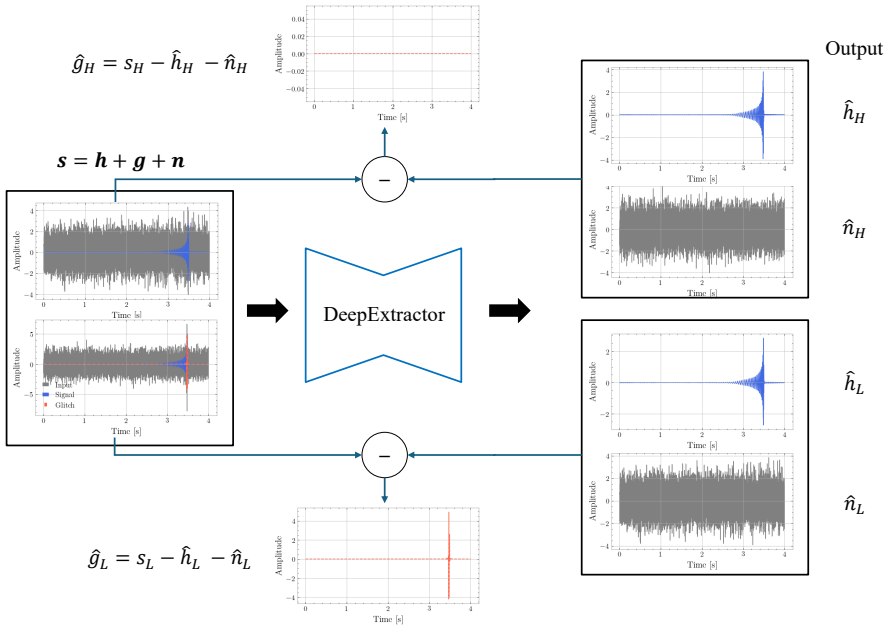


Figure 5.2: An overview of DeepExtractor's separation method. Four seconds of time-domain strain data from Hanford and Livingston, s_H and s_L respectively, is processed simultaneously. DeepExtractor outputs the signal ($\hat{h}_H$ and $\hat{h}_L$) and noise contributions ($\hat{n}_H$ and $\hat{n}_L$) in each detector. Subtracting the predicted components from the input yields predictions for glitch components in each detector as $\hat{g}_D(t) = s_D(t) - \hat{h}_D(t) - \hat{n}_D(t)$, with $D \in \{H, L\}$.

noise component. In this chapter, we extend DeepExtractor to a multi-detector setting: the model processes 4 s of time-domain strain from a two-detector network (LIGO–Hanford and LIGO–Livingston). The input to the network consists of the strain time series from each detector:

$$\{s_H(t),\, s_L(t)\},$$

and the network predicts four separate components,

$$\{\hat{h}_H(t),\, \hat{n}_H(t),\, \hat{h}_L(t),\, \hat{n}_L(t)\},$$

corresponding to the reconstructed signal and background noise contributions for each detector.

The residual glitch is then obtained by subtracting the predicted signal and noise components from the input strain:

$$\hat{g}_H(t) = s_H(t) - \hat{h}_H(t) - \hat{n}_H(t), \qquad \hat{g}_L(t) = s_L(t) - \hat{h}_L(t) - \hat{n}_L(t).$$

Because $s_H(t)$ and $s_L(t)$ are processed simultaneously as two input channels, DeepExtractor learns to exploit *cross-detector coherence*: true astrophysical signals are coherent across detectors (up to time-of-arrival and antenna-pattern differences),

whereas glitches are typically incoherent and appear in only one detector. This encourages the network to reconstruct $\hat{h}_H(t)$ and $\hat{h}_L(t)$ consistently across detectors while isolating incoherent glitch residuals.

Time-domain learning. A key change from the previous chapter is that DeepExtractor's separation mode operates directly in the time domain rather than on magnitude+phase STFT representations. This choice is motivated by three factors:

1. **Memory efficiency**: Handling two-detector inputs and four output components would require significantly more memory in the STFT domain for the same time duration.
2. **Training scale**: Learning to generalize over the full BBH parameter space requires more samples than that required in the previous chapter; the time domain enables larger batch sizes and faster training.
3. **Comparable performance**: Results from Chapter 4 showed that the performance gap between STFT-based DeepExtractor and a time-domain U-Net baseline was small (Table 4.1), making time-domain learning a practical and efficient choice.

Network architecture. Aside from these changes in dimensionality and data representation, DeepExtractor retains the same U-Net architecture used in the previous chapter (Figure 4.2). The only architectural modification is that the 2D convolutional layers used previously are replaced with 1D convolutions (i.e., `Conv2D` → `Conv1D`), enabling the network to operate directly in the time domain while keeping the hierarchical feature extraction and skip-connection structure that makes U-Net effective for separating overlapping components in noisy data.

A schematic overview of the full DeepExtractor separation workflow is shown in Figure 5.2.

5.2.2 Training Strategy

The training strategy of DeepExtractor in this chapter follows the same principles as its earlier formulation. To train the network, we require access to the background noise $n(t)$ and the target signal $h(t)$. We begin by generating whitened background noise for both LIGO–Hanford and LIGO–Livingston, denoted $n_H(t)$ and $n_L(t)$, corresponding to 4 s of detector data representative of the O3 observing run. A simulated GW signal is then injected into each noise segment by projecting the waveform onto each detector using the appropriate antenna-response and sky location. This produces the detector-frame signal components $h_H(t)$ and $h_L(t)$, and together the four targets

$$\{n_H(t),\, n_L(t),\, h_H(t),\, h_L(t)\}$$

form the supervised training targets for DeepExtractor.

To construct the network inputs $s_H(t)$ and $s_L(t)$, glitches are injected into *only one* detector per training sample, chosen at random. For the detector that does not receive a glitch injection, the glitch contribution is therefore zero, $g_i(t) = 0$. The glitch injection procedure matches that used in the previous chapter (Section 4.2.1), where

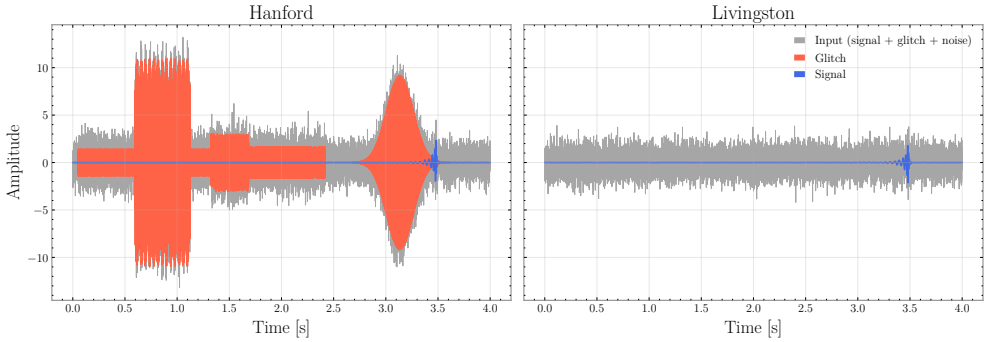


Figure 5.3: An example of an input training sample for DeepExtractor comprising data for Hanford (left) and Livingston (right), showing the injected signal and glitch components for clarity.

between 1 and 30 glitches are injected into the selected detector for every training sample (see Section 5.2.3 for more details).

We optimize the network by minimizing the MSE loss function (see Section 3.1.5, which compares the predicted components $\hat{n}_H(t)$, $\hat{n}_L(t)$, $\hat{h}_H(t)$, $\hat{h}_L(t)$ with the true background noise components $n_H(t)$, $n_L(t)$, $h_H(t)$, $h_L(t)$, as defined below:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{4N} \sum_{i=1}^{N} \left(\left\| \hat{n}_H^{(i)} - n_H^{(i)} \right\|_2^2 + \left\| \hat{n}_L^{(i)} - n_L^{(i)} \right\|_2^2 + \left\| \hat{h}_H^{(i)} - h_H^{(i)} \right\|_2^2 + \left\| \hat{h}_L^{(i)} - h_L^{(i)} \right\|_2^2 \right), \tag{5.2}$$

where N is the batch size and $\| \cdot \|_2^2$ denotes the sum of squared differences over the time dimension.

DeepExtractor employs the Adam optimizer [200] with a batch size of 64. To optimize the learning process, we once again incorporate a `ReduceLROnPlateau` scheduler, which reduces the learning rate after two epochs without improvement. Early stopping is triggered after five such plateaus, ensuring that training terminates once no further improvement is observed. This strategy provides an adaptive training duration governed by validation performance, consistent with the approach used in the previous chapter. Training DeepExtractor on our dataset (details in the next section) required approximately 37 hours on a NVIDIA H100 GPU and converged after 123 epochs.

5.2.3 Training Data

Unlike Chapter 4, which used both real and simulated data, the dataset for this study is *entirely simulated*. The objective is to evaluate whether deep learning can reliably separate signals, glitches, and background noise in a controlled environment before extending the method to real detector data.

We generate ~ 1.25 million training samples and ~ 130 thousand validation samples. Each sample consists of 4 s of whitened strain from both LIGO–Hanford and LIGO–Livingston, sampled at 4096 Hz. Figure 5.3 shows an example of the input

channels and the corresponding signal and glitch components.

Simulated BBH signals. We use *Bilby* [319] to generate realistic background noise and inject BBH waveforms using the `IMRPhenomXPHM` approximant (see Section 2.4.2). Waveforms are injected at a fixed geocentric time of $t = 3.5\,\mathrm{s}$ within the 4 s sample, ensuring that long-duration signals from low-mass systems fit within the analysis window.

CBC source parameters are drawn from broad astrophysical priors covering a diverse range of BBH configurations (Table 5.1). Luminosity distance is sampled uniformly in comoving volume out to 3000 Mpc, causing a fraction of injected signals to be extremely quiet; combined with a deliberate 5% "no-signal" probability, this prevents the network from overfitting to the presence of signals and encourages correct behavior when no signal is present.

Glitch injections. Blip glitches simulated with gengli are injected into only *one* detector per sample, chosen at random. Between 1 and 30 proxy glitches (from the same five proxy classes used in Chapter 4, see Figure 4.3) are inserted, with their parameters randomly sampled. Each glitch is scaled to a target SNR sampled from $[1, 250]$ (see Equation 4.3), ensuring a wide range of glitch amplitudes. Because some samples contain no signal and others contain negligible glitch power, the network learns to correctly map noise-to-noise when neither component is present.

Standardization. Before training, we standardize the input channels using `StandardScaler` from *scikit-learn* [346], setting the input distribution to zero mean and unit variance. The output targets (noise and signal components) are *not* scaled, which we found stabilizes training and prevents vanishing MSE values, unlike the approach taken for DeepExtractor in Chapter 4. At test time, the predicted noise and signal components are subtracted from the unscaled input strain, leaving the reconstructed glitch.

5.2.4 Experiments

This preliminary study evaluates how well a standard GW parameter estimation pipeline (Section 2.5) recovers posterior distributions when glitches are mitigated using DeepExtractor. We compare three cases: (i) clean baseline data (no glitch present), (ii) glitch–contaminated data (no mitigation), and (iii) glitch–mitigated data (processed by DeepExtractor).

We restrict this analysis to two simulated BBH signals, representing systems with parameters similar to those of GW150914 and GW200129. The latter was observed during O3, whereas GW150914 occurred in O1 and was loud relative to the detector sensitivity at that time. Injecting GW150914 into O3-like noise where the detector sensitivity significantly improved, using its original luminosity distance (450 Mpc) produces very high SNRs (approximately SNR 48 in Hanford and SNR 67 in Livingston). Although a glitch would be injected with comparable strength, this scenario would be artificially easy, where both the signal and glitch would stand out clearly above the noise. To keep the experiment realistic and challenging, we adjust only the luminosity distance to 1000 Mpc, reducing the SNRs values (about 22 and 30).

Table 5.1: BBH parameter ranges used when generating training data. Sampling matches the priors used in the training dataset generation script.

Parameter	**Sampling range / distribution**
Primary mass m_1	Uniform [5, 200] $M_\odot$
Secondary mass m_2	Uniform [5, 200] $M_\odot$ (swapped to enforce $m_1 \geq m_2$)
Spin magnitudes (a_1, a_2)	Uniform [0, 0.99]
Spin tilt angles $(\text{tilt}_1, \text{tilt}_2)$	Uniform [0, π]
Spin azimuthal angles (ϕ_{12}, ϕ_{jl})	Uniform [0, 2π]
Luminosity distance D_L	Uniform in comoving volume, $[0, 3000]$ Mpc
Right ascension RA	Uniform [0, 2π]
Declination Dec	Isotropic: $\sin(\text{Dec}) \sim U[-1, 1]$
Inclination angle θ_{JN}	Isotropic: $\cos(\theta_{\text{JN}}) \sim U[-1, 1]$
Polarization angle ψ	Uniform [0, π]
Phase at coalescence ϕ_c	Uniform [0, 2π]
Geocentric time	Fixed at 3.5 s within 4 s window

For each event, a Blip glitch is generated using gengli [317] and injected directly over the merger in one detector—an intentionally challenging configuration that maximally disrupts parameter estimation. Importantly, these Blip glitches were *not* included in the training data, allowing us to evaluate the ability of DeepExtractor to reconstruct and separate unseen glitch morphologies outside its training distribution. After applying DeepExtractor for glitch mitigation, we perform three separate parameter estimation runs (baseline, glitch-contaminated, and glitch-mitigated) to quantify how effectively the method reduces bias in the inferred posteriors.

5.3 Results

5.3.1 Reconstruction of Signal and Glitch Components

Figures 5.4 and 5.5 show time–domain reconstructions produced by DeepExtractor for GW150914–like and GW200129–like injections. Each figure compares the injected strain (top row) with the reconstructed signal (middle) and the reconstructed glitch (bottom) for both detectors. In each case, a blip-like glitch is injected directly over the merger in a single detector, creating strong time–frequency overlap with the astrophysical signal.

This overlap is visualized in the Q–transforms in Figures 5.6 and 5.7, which show (i) the clean signal (top), (ii) the glitch–contaminated data (middle), and (iii) the glitch–mitigated output of DeepExtractor (bottom).

For the GW150914–like event, DeepExtractor achieves good reconstructions of both signal and glitch components, all with mismatches below 10%. The difference is barely visible in the Q–transform (Figure 5.6), indicating that the morphology of the clean signal is preserved. Minor artifacts appear in the glitch estimate, but remain close to the noise level.

For the GW200129–like event, performance degrades: the reconstructed signal

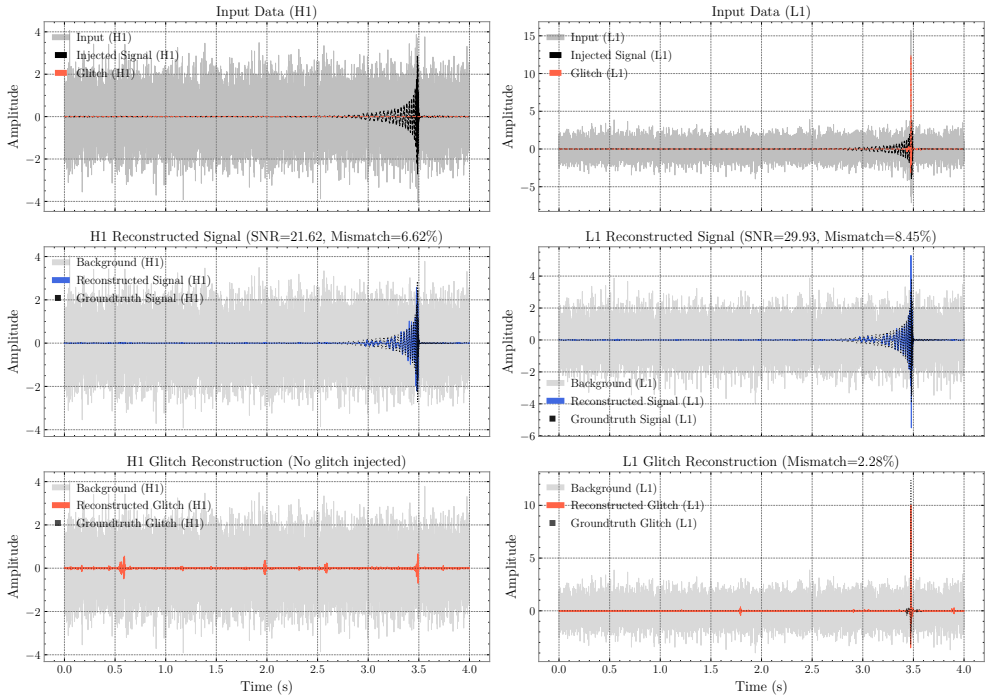


Figure 5.4: Time-domain plots for a simulated GW150914 in Hanford (H1) and Livingston (L1) with glitch injected in Livingston. The middle plots show the reconstructed signal in each detector with corresponding mismatches against the injected signal above each plot, while the bottom plots show the reconstructed glitches in each detector.

mismatch increases to nearly 20% in the glitch–affected detector, and the glitch mismatch remains $\sim 7\%$. The Q–transform (Figure 5.7) confirms that high–frequency glitch features are removed, but mid- and low-frequency components persist.

Across both events, the non-contaminated detector shows low signal mismatch and preserved signal power in the Q-scans, demonstrating that DeepExtractor does not introduce spurious glitches into clean channels and successfully preserves the astrophysical signal.

5.3.2 Impact on Parameter Estimation

It is clear from both events that the injected glitch introduces significant bias into the inferred parameters, whereas the DeepExtractor–mitigated posteriors recover distributions substantially closer to the baseline case. Comparing Figures 5.8 and 5.9, DeepExtractor performs better for GW150914, particularly in recovering the sky-location parameter DEC. This trend is consistent with the lower signal and glitch mismatches observed for GW150914 in Figure 5.4, and with the stronger residual glitch visible in Figure 5.7 compared to the much cleaner reconstruction in Figure 5.6.

Of particular importance for online parameter estimation is the recovery of the

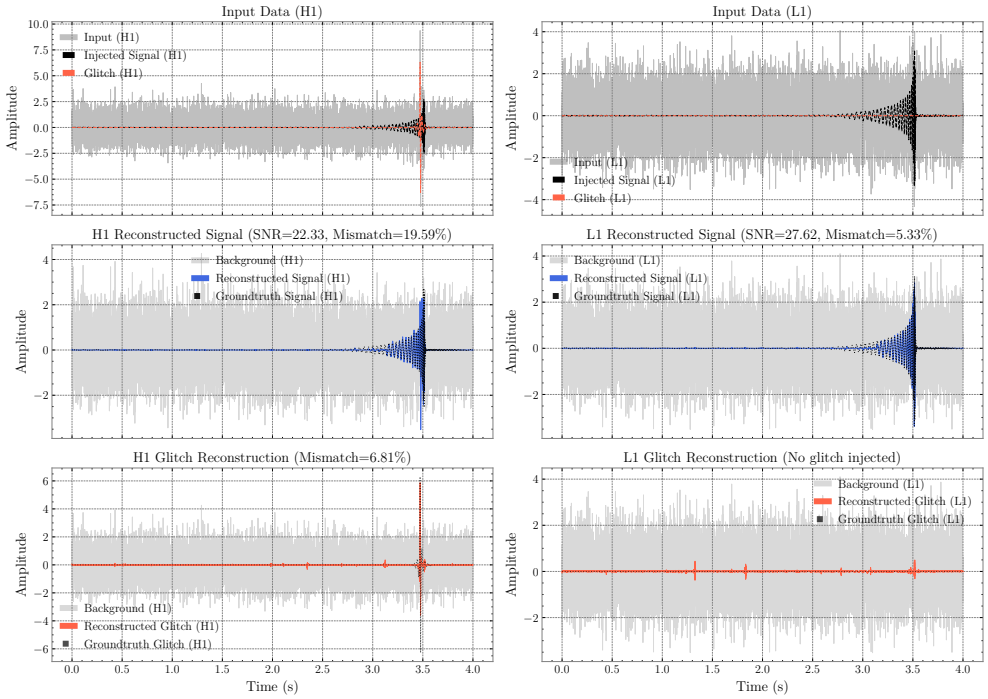


Figure 5.5: Time-domain plots for a simulated GW200129 with glitch injected in Hanford.

sky-position (RA, DEC) and luminosity distance D_L, as these quantities directly influence rapid sky localization and the triggering of electromagnetic follow-up in the case of BNS signals. Finally, because DeepExtractor processes 4 s of data in under 1 s, it is fast enough for online operation. While the DeepExtractor-mitigated posteriors do not fully match the baseline in all dimensions, they represent a substantial improvement over the glitch–contaminated case and demonstrate the feasibility of deep-learning–assisted glitch mitigation for low-latency PE pipelines.

5.4 Conclusion and Future Work

5.4.1 Conclusion

In this study, we introduced an extended version of DeepExtractor for separating GW signals, glitch transients, and detector noise in multi-detector data. Unlike prior approaches, this updated DeepExtractor configuration reconstructs both the signal and the noise components, enabling explicit extraction of incoherent glitch residuals. On two simulated gravitational wave events, DeepExtractor separates signal, glitch, and noise without requiring prior knowledge of glitch morphology, preserves coherent signal structure across detectors, and substantially reduces parameter bias introduced by glitches injected directly over the merger. Importantly, these improvements are

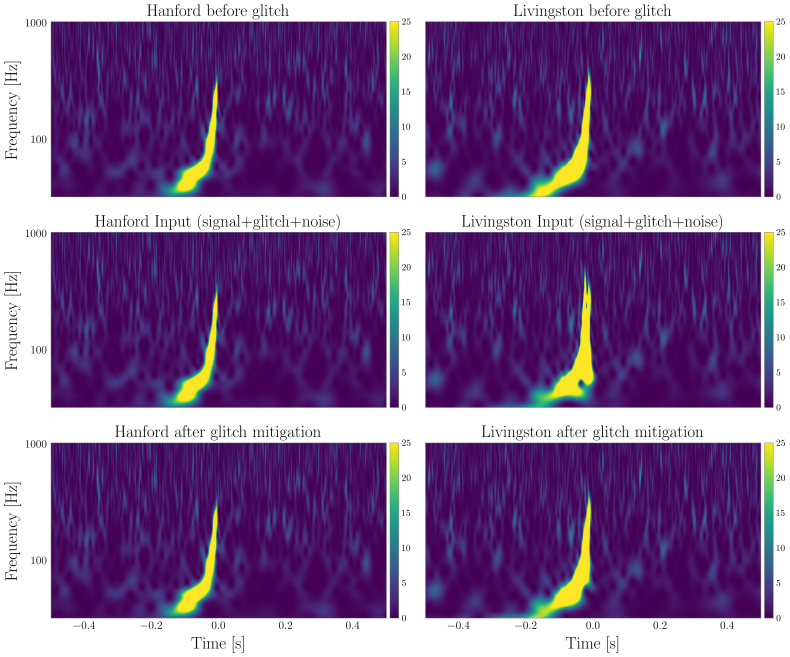


Figure 5.6: Q-scans of the GW150914 signal in Hanford (left) and Livingston (right) before the glitch injection (top), after glitch injection (middle), and after glitch mitigation.

achieved even though Blip glitches were *not included in training*, demonstrating generalization to unseen glitch morphologies. Furthermore, DeepExtractor processes 4 s of data in under 1 s on a CPU, indicating that real-time online mitigation is feasible. While preliminary, these results show that deep learning–based glitch removal is a viable path toward integrating automated glitch mitigation into online parameter estimation and rapid sky localization.

5.4.2 Future Work

This limited study serves to show preliminary results for the DeepExtractor separation framework, and several extensions are planned to build toward publication-ready performance.

Evaluation. Firstly, this evaluation only considers two simulated events, and one glitch class injected directly over the merger. This was intentional given the huge parameter space of signals and the diverse morphologies of glitches. A broader evaluation that broadly covers the signal parameters and one that includes representative samples across glitch classes will be required to investigate the robustness of DeepExtractor's separation framework. Furthermore, while we only showed the effect of glitch mitigation against the baseline data, a full 1:1 comparison against BayesWave will be required to understand the DeepExtractor performance relative to the state-of-the-art

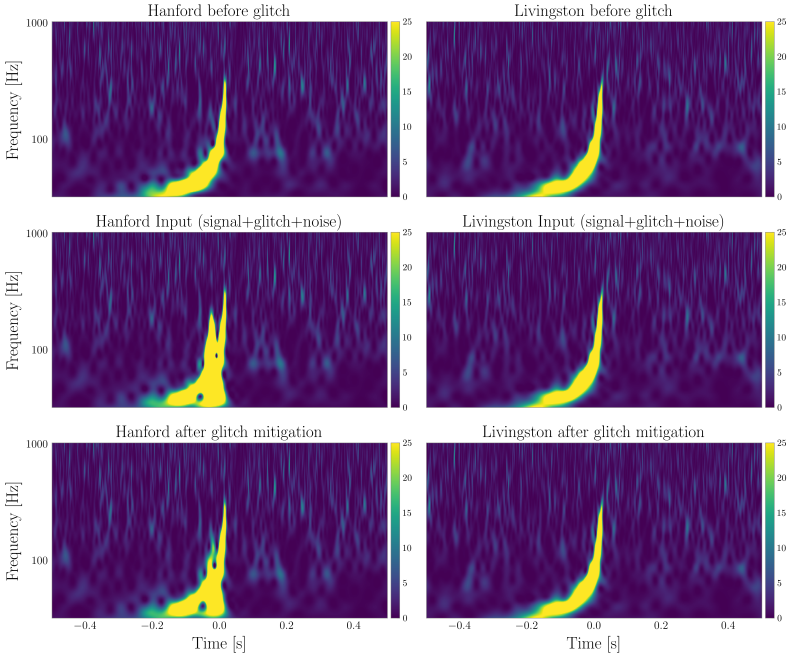


Figure 5.7: Q-scans of the GW200129 signal in Hanford (left) and Livingston (right) before the glitch injection (top), after glitch injection (middle), and after glitch mitigation.

algorithm employed for glitch mitigation in real data analysis.

Dataset and training improvements. While DeepExtractor performs well on the examples shown here, further work is needed to better match the baseline posteriors across a wider range of sources. A key next step is expanding the training dataset so that it provides denser and more uniform coverage of the compact binary coalescence parameter space. The current training set oversamples very distant sources, resulting in a large number of low-SNR examples that are less informative for learning signal–glitch separation. Reducing the maximum luminosity distance and better shaping the SNR distribution would make the training more representative of real detections. Additionally, refining the distribution of injected glitches to better approximate real glitch rates and amplitudes should improve generalization.

Architecture enhancements. The present network focuses primarily on per-detector reconstruction. Incorporating explicit cross-detector communication—via attention mechanisms, temporal aggregation modules, or lightweight Transformer blocks—may allow the model to better exploit inter-detector coherence, improving separation when only one detector is glitch contaminated. Similarly, experimenting with hybrid loss functions that balance waveform similarity (e.g., cosine similarity) with time-domain fidelity (e.g., mean squared error) could help stabilize training, particularly across a wide range of SNRs where the relative contributions of signal, noise, and glitch vary

significantly.

Physics realism. The analysis presented here is limited to two-detector simulations, but real observing runs frequently involve three or more instruments. Extending DeepExtractor to include Virgo (and later KAGRA) will allow the network to learn multi-detector coherence more effectively. Incorporating realistic glitch morphologies—either by injecting real transients from DeepExtractor's glitch reconstructions, as shown in the last chapter, or by sampling a diverse set of glitch families—will improve robustness to real detector behavior. Evaluating performance across these broader conditions will test whether the method generalizes beyond blip-like artifacts and remains effective for a wide range of non-Gaussian noise.

Ultimately, the goal is to develop a real-time, statistically robust separation of signals, noise, and glitches, deployable as a pre-processing module for low-latency parameter estimation pipelines in current and next-generation GW detectors.

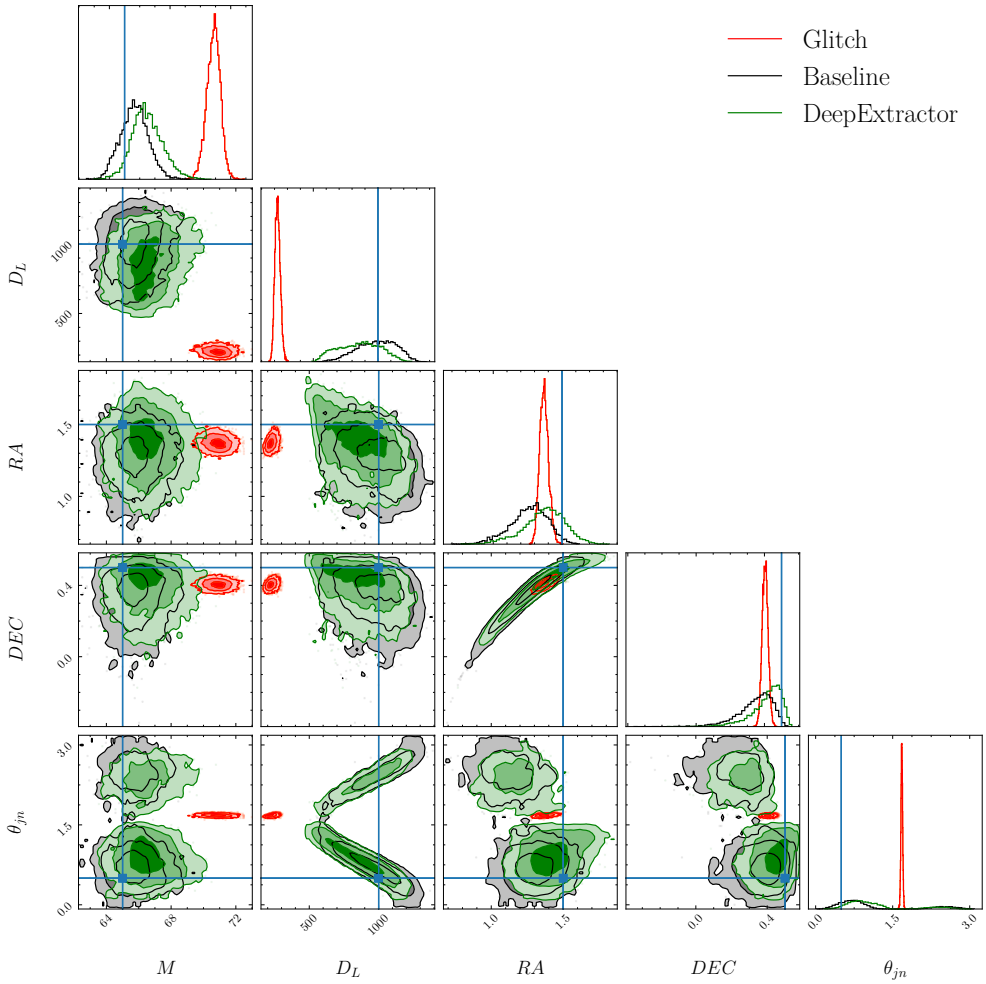


Figure 5.8: Parameter estimation results for the GW150914-like signal showing posterior distributions for five key source parameters before glitch injection (black), after glitch injection (red), and after glitch mitigation using DeepExtractor (green). The parameters shown are: total mass M, luminosity distance D_L, right ascension RA, declination DEC, and the binary inclination angle θ_{jn}. The injected (ground–truth) parameter values are indicated by blue lines.

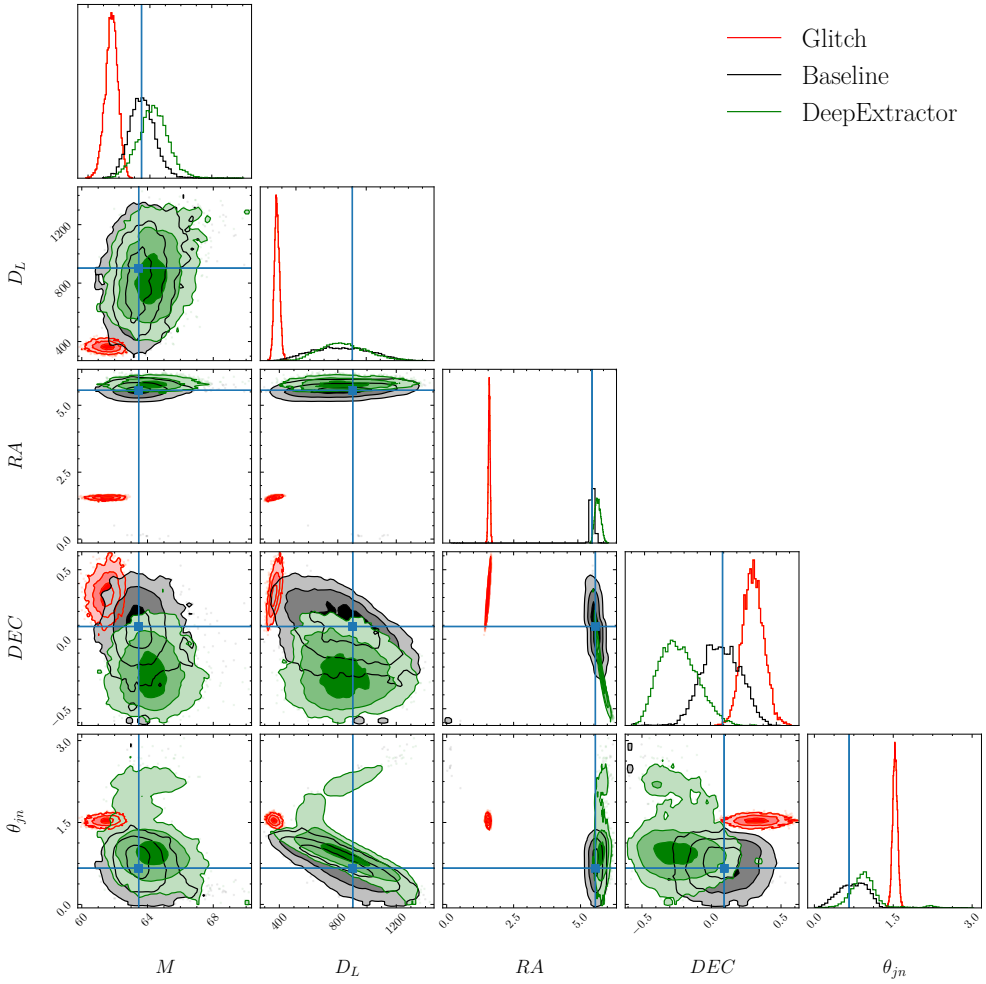


Figure 5.9: Parameter estimation results for the GW200129-like signal, showing posterior distributions for the same five parameters in Figure 5.8 before glitch injection (black), after glitch injection (red), and after glitch mitigation using DeepExtractor (green). The injected (ground–truth) parameter values are indicated by blue lines.

Part III

Generative Modelling of Gravitational-Wave Signals and Glitches

Chapter 6

Stabilizing Wasserstein GANs with Derivatives

Simulating realistic time-domain observations of GWs and glitches, can help in advancing GW data analysis. Simulated data can be used in downstream data analysis tasks by augmenting datasets for signal searches, balancing data sets for machine learning applications and validating detection schemes. This paper presents a novel approach to simulating fixed-length time-domain samples using a three-player Wasserstein GAN (WGAN), called DVGAN, that includes an auxiliary discriminator that discriminates on the derivatives of input samples. An ablation study is used to compare the effects of including adversarial feedback from an auxiliary derivative discriminator with a vanilla two-player WGAN. We show that discriminating on derivatives can stabilize the learning of GAN components on 1D continuous samples during their training phase. This results in smoother generated data that are less distinguishable from real samples and better capture the distributions of the training data. DVGAN is also used to simulate real transient noise events captured in the advanced LIGO GW detector.

This chapter is based on the following publication:

Dooney et al.. *DVGAN: Stabilize Wasserstein GAN training for time-domain Gravitational Wave physics*, DOI: 10.1109/BigData55660.2022.10021080, IEEE International Conference on Big Data, 2022.

6.1 Introduction

As discussed in Section 3.3.5, deep generative modelling of GW signals and realistic detector glitches, has wide-ranging applications in GW data analysis. Such models can provide computationally efficient alternatives to traditional waveform generation methods, enhance the realism of detector simulations for mock data challenges, and enable more rigorous testing and validation of detection pipelines and parameter estimation algorithms via software injections [310–312].

Among generative approaches, GANs [220] (see Section 3.2.6) have emerged as powerful tools for modelling complex data distributions. Although GANs have been primarily applied in computer vision, they have also demonstrated growing utility in GW physics [314–318]. Multiple studies that deal with glitch generation employ Q-scan representations of detector data that [315, 316], however, they overlook the intrinsic advantages of modelling directly in the time domain.

Time-domain generative modelling of GW strain data offers several benefits. First, strain data are measured natively as one-dimensional time series, and modelling directly in this space eliminates the need for computationally expensive transformations to and from the time–frequency domain. Second, time-domain samples are more flexible to manipulate and combine, facilitating the exploration of overlapping events—an increasingly relevant scenario for next-generation detectors as outlined in the beginning of Chapter 3. Finally, transforming from a high-fidelity time-domain representation to a spectrogram is straightforward, whereas reconstructing to the time-domain from a magnitude spectrogram is lossy due to the irreversible loss of phase information.

GANs have shown remarkable potential in generative modelling but are notoriously difficult to train and often prone to instability—challenges that become even more pronounced when simulating the noisy, non-stationary environments of GW detectors [317]. As discussed in Chapter 4, the complex and highly variable distributions underlying realistic glitch reconstructions from real GW data, such as those recovered by DeepExtractor, present additional difficulties for generative models to learn effectively.

To address these challenges, we introduce the Derivative GAN (DVGAN), a stabilized one-dimensional GAN framework designed specifically for generating GW signals, including bursts and glitches. DVGAN adopts a three-player adversarial structure comprising two discriminators and one generator—all convolutional with fully connected layers, enabling better scalability for long sequences. Previous three-player GANs have been explored in other domains, employing either dual discriminators or auxiliary classifiers to enhance adversarial feedback and mitigate mode collapse [360–362]. Building on these ideas, DVGAN introduces a derivative-based dual-discriminator design: one discriminator operates directly on the training sample, while the other acts on its temporal derivative, providing complementary feedback that stabilizes training and improves the physical realism of generated sampled. To the best of our knowledge, this derivative-informed dual-discriminator approach has not been previously explored in the context of GW data generation.

We train DVGAN on four distinct classes of time-domain data relevant to GW data analysis and use ablation studies to compare its performance against a baseline Wasserstein GAN (WGAN) (see Section 3.2.6). The quality of the generated data is

evaluated using three complementary analyses: (1) a t-SNE visualization to examine the distributional overlap between real and synthetic samples; (2) a discriminative score, where an independent CNN is trained to distinguish real from generated data; and (3) a match-based analysis assessing how well the generated signals reproduce the morphological and parametric diversity of the training set. Together, these metrics provide a comprehensive evaluation of both the realism and variability of data produced by DVGAN.

Our results show that DVGAN achieves more stable training dynamics than its baseline WGAN counterpart while generating samples that are equally—if not more—realistic and representative of the diversity present in the training data. In particular, DVGAN more effectively captures the morphological variety of challenging glitch datasets reconstructed from real LIGO data, underscoring the advantage of its derivative-informed dual-discriminator design. This work lays the groundwork for more advanced generative modelling of GW data and serves as a precursor to the conditional multi-class framework introduced later in this thesis (Chapter 7).

6.2 Methods

6.2.1 Derivative GAN (DVGAN)

This research introduces a novel GAN architecture, called DVGAN, designed to stabilize and enhance the generation of one-dimensional signals. The adversarial process that is featured in GANs, where two model components compete against one another, can often be unstable and prone to non-convergence. When dealing with challenging time-series morphologies, GANs lacking the appropriate architecture can struggle to converge or can fall into a local minimum and suffer mode collapse (where only a few high-fidelity samples can be generated with little variety). Achieving a stable balance between the generator and discriminator is therefore a central challenge, with failures in convergence typically revealed through oscillatory or divergent loss curves.

Motivated by the instability observed in standard Wasserstein GANs (WGANs) when applied to 1D data, we propose an additional discriminator that operates on the derivatives of both real and synthetic samples. This second adversary introduces gradient-level feedback, effectively regularizing the learning process and improving the fidelity of generated data. This dual-discriminator setup forms the core novelty of the proposed DVGAN, resulting in a three-player adversarial framework that is compared against the conventional two-player GAN baseline.

A visualization of the DVGAN architecture is provided in Figure 7.1 in the following chapter, which illustrates its conditional variant (cDVGAN) against a vanilla GAN architecture. This conditional form is structurally identical to DVGAN, differing only by the inclusion of a class-conditioning vector c that enables class-controllable generation. The additional model component acts as an extra form of regularization by discriminating on derivatives, providing information about the rate of change and encouraging smoother, more physically consistent outputs.

The derivative discriminator employs the same loss formulation as the standard discriminator introduced in Equation 3.23 in Section 3.2.6, expressed as

$$L(\phi_2, \theta) = -\mathbb{E}_{\frac{dx}{dt} \sim P_{\frac{dx}{dt}}} \left[D_2(\frac{dx}{dt}, \phi_2) \right] + \mathbb{E}_{\frac{d\hat{x}}{dt} \sim P_{\frac{d\hat{x}}{dt}}} \left[D_2(\frac{d\hat{x}}{dt}, \phi_2) \right] \tag{6.1}$$

where D_2 operates on the derivatives of real and generated data. The total generator loss combines the feedback from both discriminators and is written as

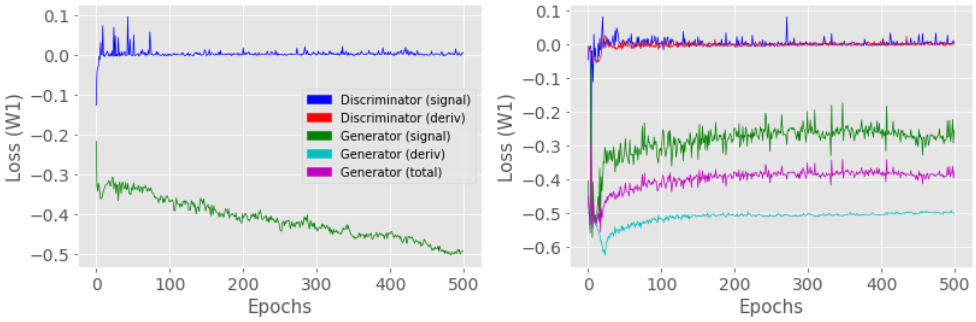
$$L_G(\phi_1, \phi_2, \theta) = -\eta_1 \mathbb{E}_{\hat{x} \sim P_{\hat{x}}} \left[D_1(\hat{x}, \phi_1) \right] - \eta_2 \mathbb{E}_{\frac{d\hat{x}}{dt} \sim P_{\frac{d\hat{x}}{dt}}} \left[D_2(\frac{d\hat{x}}{dt}, \phi_2) \right] \tag{6.2}$$

where D_1 and D_2 denote the primary and derivative discriminators, respectively, with corresponding weights ϕ_1 and ϕ_2.

The network architecture utilizes deep convolutional networks for the generator and both discriminators. GANs are sensitive to the size of components and their architectures, therefore, the discriminator and the generator of the WGAN maintain a similar number of parameters (4.1M), and mirror each other in terms of their architecture. The derivative discriminator of DVGAN comprises a significantly smaller architecture with only about 1.6% (68k) of the number of parameters of the base discriminator (the base discriminator and generator are identical to the corresponding components of the WGAN). The generator uses four layers of transposed convolutions with BatchNormalization in the first layer and a stride of 2 for upsampling.

The base discriminator mirrors the generator architecture without BatchNormalization but uses dropout and a stride of 2 in the convolutional layers for downsampling. Both discriminators and the generator employ a kernel size of 5 and use LeakyReLU and ReLU activations respectively, except for their last layers. A linear activation is used for the final generator layer (guaranteeing positive and negative outputs), and sigmoid activations are used for the final layer of both discriminators (representing the probability of a sample being synthetic). It is found that a smaller network can be utilized in the derivative discriminator and still yield superior results. A Wasserstein loss function is used with RMSprop as the optimizer, with a learning rate of 10^{-4}. WGAN training can become unstable using momentum-based optimizers such as Adam, and it is reported in the original WGAN paper that RMSprop provides more stable training [363]. The *GP* regularization method described in Section 3.2.6 is also included in the losses of both discriminators. Throughout experimentation, η_1 and η_2 in Equation 6.2 are both maintained at 0.5 (arbitrarily, optimization is not the focus of this study).

When training a GAN, the generator and discriminator must be updated at particular rates to achieve convergence. Updating the discriminator is more challenging since samples from the generator can be anywhere in the signal space and can change over each iteration [364]. Therefore, the discriminator is updated 5 times for each update of the generator, according to recommendations from previous research [227, 228]. For each dataset (see Section 6.2.2), a DVGAN is trained for 500 epochs on 6000 samples with a batch size of 512. A WGAN with the same configuration except for the inclusion of the derivative discriminator is also trained on each of the datasets to enable a comparison with DVGAN. The learning behaviour of the model can be monitored by plotting the losses of GAN components. Figure 6.1 shows an example of stable DVGAN learning compared to that of a vanilla WGAN. Unstable loss trends commonly encountered when training WGANs are one of the motivating



(a) WGAN (b) DVGAN

Figure 6.1: Examples of an unstable learning process from a vanilla WGAN (left) and a stable learning process from a DVGAN (right) on the BBH dataset (see Section 6.2.2). The plots show the evolution of the Wasserstein-1 ($W1$) distance over 500 training epochs, with more stable, flat trends for DVGAN.

factors of this research. All models were designed using Keras [365] and Tensorflow [366] libraries.

It is important to note that effective GAN architectures are highly dependent on the data, and must be tuned according to the problem at hand. There is no optimization involved in either the WGAN or the DVGAN and there is no guarantee that the specified architectures are optimal in either case. Multiple model configurations were implemented that include different hyperparameters, three of which are presented in Appendix B.1, along with the final, most optimal model architecture (in bold text) found through trial and error (based on the evaluation followed in this paper). It is important to note that in many of the configurations attempted, the DVGAN offers more stable training than (or at least similar to) WGAN counterparts. A uniform architecture is maintained across all experiments for a 1:1 ablation study against the vanilla method. An ablation study in machine learning is a systematic experimentation technique used to understand the contribution of individual components of a model to its overall performance. Thus, this research shows that it is possible to stabilize GAN training by bolstering the model with a second discriminator applied to another representation of the data.

6.2.2 Training Data and Procedures

Four datasets comprising 1D differentiable time-series are utilized to show the robustness of DVGAN in this setting. The first three are inspired by [318], representing proxy waveforms analogous to GW waveforms and transients. Notably, the first two were also included in the DeepExtractor training dataset described in the previous chapter. The final dataset is compiled from actual 'blip' glitch events and is obtained from [317].

1. Gaussian pulses: short exponential increase then decrease in amplitude that follows the equation $h_{GP}(t) = exp(\frac{-t^2}{\tau^2})$. Gaussian pulses can represent anomalous glitch events observed in GW detectors.

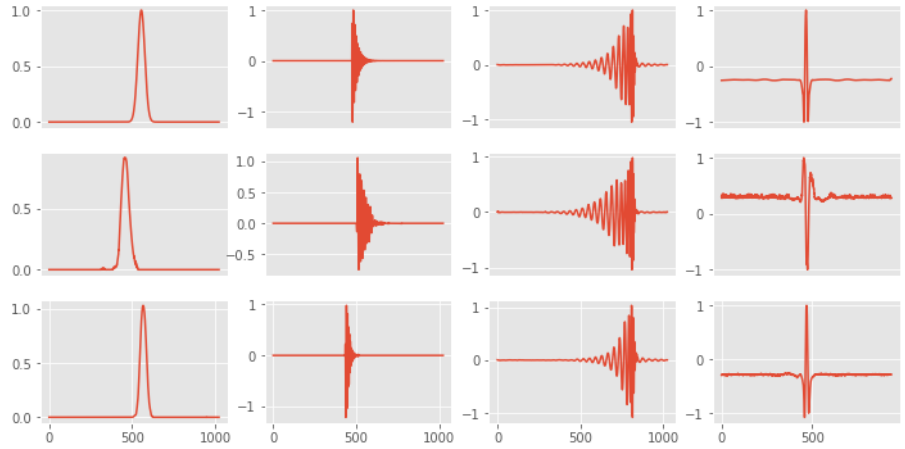


Figure 6.2: Example waveforms (*top*) for all datasets (pulse, ringdown, BBH, blip) with example generations from WGAN (*middle*) and DVGAN (*bottom*). Noise artifacts are more significant in the WGAN generations compared with DVGAN, in particular for the pulse and blip examples.

2. Ringdown: damped oscillations mimicking the signals following a binary black hole collision, following the equation $h_{RD}(t) = Aexp[-(t-t_0)/\tau]sin(2\pi f_0(t-t_0)+\phi)$ with frequency f_0, duration parameter τ, amplitude A, time of arrival t_0 and phase ϕ (uniformly sampled between [0, 2π]).

3. BBH: The inspiral and merger of a BBH system. All BBH signals were simulated using the IMRPhenom waveform routine from LALSuite [367], which generates the inspiral, merger and ringdown of a BBH waveform. The component masses are restricted to the range of $[30, 70]M_{\odot}$ with a spin of zero and fixing $m_1 > m_2$. The mass distribution is approximated by a power law with an index of 1.6 [368]. The inclinations of the BBH signals are drawn so that the cosine of the angles lies uniformly in the range of $[-1, 1]$ using only plus polarization.

4. Blip glitches: One glitch class of 22 that have been classified by Gravity Spy, reconstructed from actual LIGO strain data using BayesWave.

The location of the peak amplitude of the Gaussian pulse and ringdown datasets are randomly drawn from a uniform distribution within $[0.4, 0.6]$ from the start of the 1 s time interval, with all training datasets except the Blip glitch dataset sampled at $1,024Hz$. This corresponds to samples of 1,024 data points in length for the first three datasets. GANs can generate higher dimensional spaces, however, larger networks and longer training times are generally required. The masses of the BBH dataset are restricted to be above 30 solar masses. Lower mass systems would produce longer-lasting waveforms that would fall outside the $1s$ interval defined. All training datasets are rescaled so that their peak amplitude is at 1. The Gaussian pulse and ringdown datasets are analytic proxy waveforms of signals expected from burst GW sources. These two datasets and their suitability for this research are also described in [369].

The final Blip glitch dataset was constructed using confidences from Gravity Spy applied to the blip glitch triggers in LIGO's second observing run (O2). In [317], they select Blip glitch triggers from LIGO's Hanford (H1) and Livingston (L1) detectors with a high Gravity Spy confidence, $c^1_{GS} \geq 0.9$, with c^1_{GS} representing the base confidence provided by Gravity Spy when applied to raw strain data. The glitches are extracted but are surrounded by stationary and uncorrelated noise, which would hinder the learning of GAN models.

The dataset requires multiple preprocessing steps [317], which are also followed in this study, except for the final denoising step, called rROF [370], which cannot remove some high-frequency artifacts that remain after previous preprocessing. Instead, to further reduce artifacts and assist in GAN training, two Savitzky-Golay smoothing filters [371] are applied to the glitches one after another (polynomials of order 3, windows of 21 and 11 respectively), which can smooth most noise artifacts in the dataset, and the smoothed blips are again rescaled. It is not investigated whether the characteristics of the Blips are preserved after filtering. However, the glitch morphologies are analogous and can serve experimental purposes. With a sampling rate of $4,096Hz$, the samples of the final training set have 938 data points, comprising $0.23s$ of LIGO strain data.

6.2.3 Data Generation and Experiments

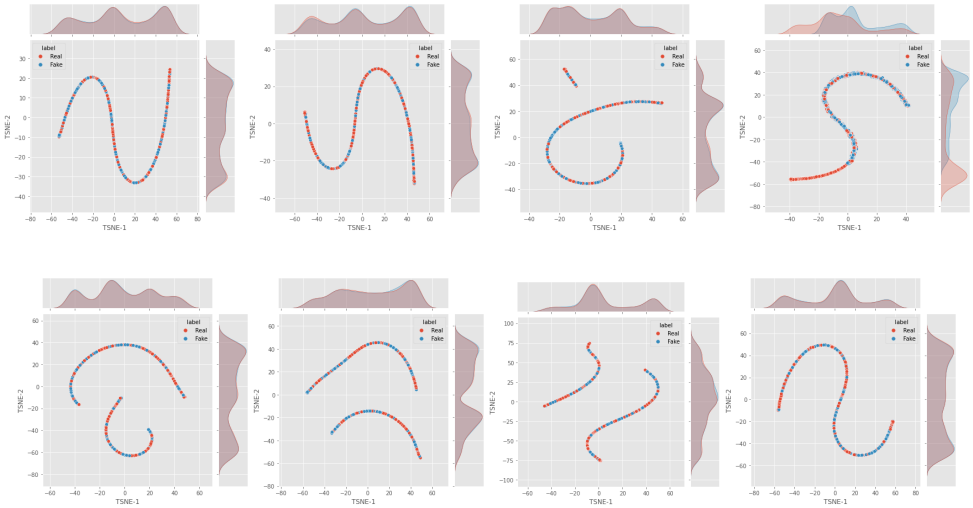
After training the WGAN and DVGAN on each training dataset, synthetic samples are generated by each network, examples of which can be viewed in Figure 6.2. This is done by sampling a 100-dimensional vector from a normal distribution, typical of GAN studies. Using the generated waveforms and corresponding training data, three experiments are used to achieve a broad view of the performance on respective datasets:

1. *Diversity* - A t-SNE analysis to ensure that the distribution of the generated samples aligns with those of the training set (3000 samples each), showing that the diversity of the training set is captured.

2. *Fidelity*[1] - A discriminative analysis that uses a 2-layer, one-dimensional CNN to classify a balanced dataset of training and generated samples (3000 samples each, trained over 20 epochs), giving a discriminative score, defined as $|0.5 - HoldoutError|$.

3. *Match Consistency*: A final experiment, only applied to the BBH dataset, involves a match analysis on 500 generated samples from each GAN, to investigate the parameter space covered by the generated signals compared to that of the training set. Metrics such as the mean maximum match (same as that defined in Section 2.4.2 and used in Chapter 4), RMSE and the signal shift required to achieve the maximum match are used to show how well the generated data matches the training data, while the mass distributions are compared with that of the training data under a two-sample *Kolmogorov–Smirnov* (KS) Test[372] at the 95% confidence interval.

[1]The model architecture involved in the fidelity experiments is shown in the corresponding publication and code repository



(a) Pulse (b) Ringdown (c) BBH (d) Blip

Figure 6.3: t-SNE plots comparing real and synthetic data for a vanilla WGAN (*top*) and DVGAN (*bottom*). Both models capture the diversity of the first three classes; however, DVGAN more effectively reproduces the diversity of the final blip glitch dataset.

6.3 Results

6.3.1 Experimental Analysis

Visualization - Diversity

Visualizing the synthetic data using t-SNE provides insight into how well the generated samples capture the diversity of the training data. Figure 6.3 presents t-SNE projections comparing real and synthetic distributions in a reduced two-dimensional space. For the first three datasets, both WGAN and DVGAN successfully reproduce the diversity of the training sets, as indicated by the overlap in their respective distributions along the two t-SNE dimensions. However, as shown in Figure 6.3d, DVGAN demonstrates a markedly superior ability to capture the variability of the blip dataset, highlighting its advantage over WGAN in reproducing morphology of glitch waveforms derived from real detector noise.

Discriminative Score - Fidelity

Table 6.1 reports the discriminative scores of WGAN and DVGAN across all datasets. DVGAN consistently achieves lower discriminative scores than WGAN, indicating that its generated samples are more difficult to distinguish from real data. This suggests that DVGAN produces higher-fidelity and more realistic data compared to its WGAN counterparts.

	Discriminative	Score		Match Analysis	
Dataset	**WGAN**	**DVGAN**	**Metric**	**WGAN**	**DVGAN**
Pulse	$0.101 \pm \frac{0.051}{0.041}$	$\mathbf{0.009} \pm \frac{0.015}{0.007}$	Match	$\mathbf{0.9918} \pm \frac{0.000}{0.000}$	$0.9917 \pm \frac{0.000}{0.000}$
Ringdown	$0.199 \pm \frac{0.025}{0.038}$	$\mathbf{0.108} \pm \frac{0.014}{0.018}$	RMSE	$0.0486 \pm \frac{0.0005}{0.0007}$	$\mathbf{0.0428} \pm \frac{0.0001}{0.0002}$
BBH	$0.496 \pm \frac{0.003}{0.004}$	$\mathbf{0.369} \pm \frac{0.009}{0.026}$	Shift	$0.1740 \pm \frac{0.0019}{0.0020}$	$\mathbf{0.1535} \pm \frac{0.0015}{0.0015}$
Blip	$0.453 \pm \frac{0.012}{0.010}$	$\mathbf{0.129} \pm \frac{0.012}{0.011}$	(KS1, KS2)	$(0.106 \pm \frac{0.007}{0.010}, 0.114 \pm \frac{0.014}{0.008})$	$(0.102 \pm \frac{0.011}{0.021}, 0.143 \pm \frac{0.009}{0.005})$

Table 6.1: Discriminative scores (lower is better) and match metrics for WGAN and DVGAN on respective datasets. The results represent the mean over 5 iterations of experimentation, with bold text indicating a superior results across both models. Bounds are provided by the maximum and minimum scores over 5 iterations. The Mean Max Match indicates, on average, how well the generated data matches training data, two KS test statistics measure the discrepancy between training data and estimated (synthetic) mass distributions (M1 and M2 respectively), and the shift describes how many data points the time-series must be shifted to maximize their match. Bounds are provided by the maximum and minimum scores over all iterations.

Match Analysis for BBH Dataset

Aside from comparing the generated data from GANs directly with the training data, it is also useful to compare the underlying parameters that result in such waveforms. For this purpose, a match analysis is conducted on the BBH waveforms from the WGAN and DVGAN, using a large bank of 20,000 simulated BBH waveforms with known masses in the range $[20, 80]M_{\odot}$ (to investigate if the GANs generate waveforms with masses outside of the $[30, 70]M_{\odot}$ range in the training set). Both GANs are used to generate 500 BBH waveforms for each iteration, which are compared with the bank of simulated BBHs using *PyCBC's* [373] match function. Taking the masses of the maximum matches allows an investigation into the parameter space of the training data that is covered by the respective GANs.

Figure 6.4 shows the coverage of the parameter space by the generated waveforms from respective models, compared with same-size subsets of the template bank. A lower triangular plot is observed in the template distribution, due to the fixing of $m_1 > m_2$. For each GAN-generated sample, the maximum match is recorded between it and its matched waveform to compare how well the respective waveforms fit. Table 6.1 shows that a similar mean match is yielded by both models, however, the average RMSE from DVGAN generations is lower than that of WGAN, while the average shift is also lower. Since the time of coalescence of the BBH signals is kept constant within each sample, maximum matches should be achieved from a shift of zero. The lower shift value yielded by DVGAN generations implies that the generated data better matches the training data compared to WGAN. It is observed that the generated waveforms from both models have mass distributions that are different from that of the training data, and even match with waveforms outside of the mass range of the training data. This is confirmed by the two-sample KS tests, where the null hypothesis (same distribution) is rejected for each iteration of the experiment. However, it is unclear whether this is beneficial, given that interpolations in the signal space might correspond to extrapolations in the parameter space.

(a) WGAN (b) DVGAN

Figure 6.4: Match Analysis on BBH generations from a Vanilla WGAN (*left*) and DVGAN (*right*).

6.4 Conclusion

This study has demonstrated that the proposed DVGAN consistently outperforms the baseline WGAN across multiple datasets and evaluation experiments, producing samples of higher fidelity and smoother morphology. Through ablation-style comparisons, we find that the dual-discriminator design yields more stable learning dynamics and improved output quality with only minimal additional computational overhead—provided the auxiliary discriminator remains lightweight. Importantly, the method has also been successfully applied to real (filtered) Blip glitch samples from LIGO, illustrating its potential applicability to realistic GW data. This capability is further explored in the next two chapters, where we extend the DVGAN framework to conditional settings on diverse and realistic glitch datasets and evaluate its use for data augmentation in hypothetical search experiments. Beyond GW physics, the approach also holds promise for generative modelling in other domains involving temporal data, such as medical or financial time series, as well as broader applications where leveraging multiple data representations can guide and stabilize generator learning.

There remain several promising directions for future work. Instead of restricting the auxiliary discriminator to derivatives, alternative data representations—such as spectrograms, moving averages, or other physically meaningful transforms—could be employed to guide adversarial learning. Higher-order derivatives could also be incorporated, for example by stacking them with first derivatives into a joint feature matrix processed via one-dimensional convolutions. However, care must be taken to avoid over-empowering the discriminator, as excessive representational information can hinder convergence—a common challenge in GAN training for high-dimensional data [374]. One potential mitigation strategy is to expand the architecture to include

additional discriminators, each operating on different data representations, rather than overloading a single one. Additionally, scaling the auxiliary discriminator to match the capacity of the primary discriminator could further enhance training stability and generative fidelity, particularly when computational resources allow. These ideas, along with alternative GAN architectures beyond the WGAN baseline, is further investigated in the following chapter.

Chapter 7

Class Conditioning Derivative GANs

As discussed in the previous chapter, simulating realistic time-domain observations of GWs and glitches can help in advancing GW data analysis. In this work, we present Conditional Derivative GAN (cDVGAN), a novel conditional model in the GAN framework for simulating multiple classes of time-domain observations that represent GWs and detector glitches. cDVGAN can also generate generalized hybrid samples that span the variation between classes through class interpolation in the conditioned class vector. cDVGAN introduces an additional player into the typical 2-player adversarial game of GANs, where an auxiliary discriminator analyzes the first-order derivative time-series. Our results show that this provides synthetic data that better captures the features of the original data. cDVGAN conditions on three classes in the time-domain, two denoised from LIGO Blip and Tomte glitch events from its 3rd observing run (O3), and the third representing BBH mergers. Our proposed cDVGAN outperforms 4 different baseline GAN models in replicating the features of the three classes. Specifically, our experiments show that training CNNs with our cDVGAN-generated data improves the detection of samples embedded in detector noise beyond the synthetic data from other state-of-the-art GAN models. Our best synthetic dataset yields as much as a 4.2% increase in AUC performance compared to synthetic datasets from baseline GANs. Moreover, training the CNN with class-interpolated hybrid samples from our cDVGAN outperforms CNNs trained only on the standard classes, when detecting real samples embedded in LIGO detector background between signal-to-noise ratios ranging from 1 to 16 (4% AUC improvement for cDVGAN). We also demonstrate cDVGAN for use in data augmentation, showing that it is competitive with a traditional augmentation approach. Lastly, we test cDVGAN's BBH signals in a fitting-factor study, showing that the synthetic signals are generally consistent with the semi-analytical model used to generate the training signals and the corresponding parameter space.

This chapter is based on the following publication:

Dooney et al.. *One flexible model for multiclass gravitational wave signal and glitch generation*, DOI: 10.1103/PhysRevD.110.022004, Phys. Rev. D, 2024.

7.1 Introduction

In the previous chapter, we demonstrated that incorporating derivative-based adversarial feedback can significantly stabilize the training of GANs for time-domain GW data. By introducing an auxiliary discriminator that operates on sample derivatives, the proposed DVGAN framework improved convergence and enhanced the fidelity of generated one-dimensional signals.

Building on these results, this chapter extends the derivative-discriminator concept into the conditional domain, introducing a framework capable of modelling distinct classes of time-domain observations from GW detectors. We propose the conditional Derivative GAN (cDVGAN), a model designed to simulate multiple types of LIGO time-domain signals, including both glitch morphologies and astrophysical waveforms, within a single unified architecture. In this study, cDVGAN is conditioned on three classes: two denoised glitch types from the LIGO O3 run, Blip and Tomte, and one representing BBH mergers. Unlike traditional GANs trained separately for each class, cDVGAN learns a joint latent space across multiple categories, enabling the generation of both in-class samples and hybrid signals that smoothly interpolate between classes via a user-controlled conditioning vector. Once trained, the model can generate hundreds of thousands of diverse, high-fidelity samples within seconds, making it a scalable and flexible tool for GW data analysis.

To evaluate its performance, we first assess the quality of cDVGAN-generated data using PCA visualizations, which reveal strong alignment with the corresponding training distributions—consistent with the t-SNE-based analysis performed in the previous chapter. We also demonstrate that the model's hybrid samples smoothly span the variation between standard classes in the reduced three-dimensional PCA space, confirming its ability to capture continuous transitions between distinct classes. Next, we examine the practical utility of the generated data in downstream GW analysis tasks. Specifically, we train a CNN binary classifier to detect real samples embedded in real LIGO detector noise using synthetic training data produced by cDVGAN, and benchmark its performance against other state-of-the-art GAN models using the AUC metric. Our results show that cDVGAN better captures the morphological and temporal characteristics of real detector data, owing to its derivative-informed auxiliary discriminator. Furthermore, we demonstrate that hybrid samples generated by cDVGAN can serve as valuable data for improving the robustness of detection algorithms to signals that fall outside standard glitch or astrophysical waveform classes.

Since cDVGAN is primarily intended for use as a glitch generator, as will be demonstrated in the next chapter, the astrophysical nature of the BBH signals is considered arbitrary for most of the experiments. However, we also confirm our cDVGAN data against a semi-analytical model used in GW searches, as discussed in Section 2.4, exploring cDVGAN's application as a BBH signal generator. To this end, we implement a fitting-factor study to evaluate the faithfulness of our generated data against the templates of a template bank. The results show that cDVGAN's synthetic signals are generally consistent with signals from the waveform routine used to generate the original training signals.

Finally, this work positions cDVGAN as a core component of a next-generation glitch generator framework, extending beyond the capabilities of existing systems such as gengli [317]. This broader application is explored in detail in the next chapter of

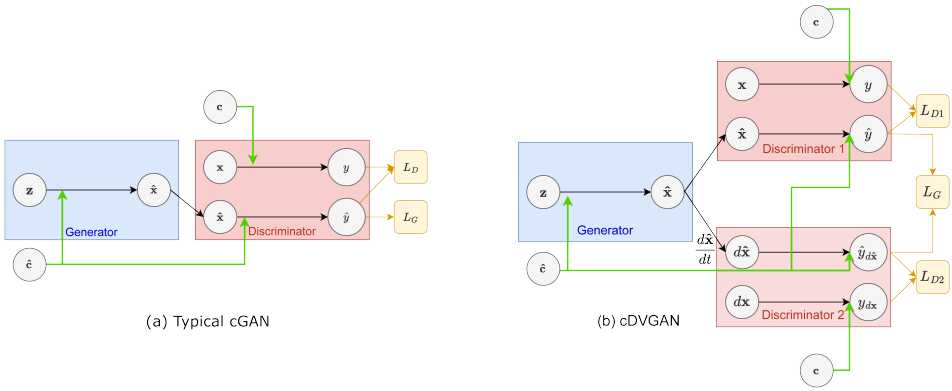


Figure 7.1: Diagrams of a typical cGAN architecture (left), comprising one discriminator, and cDVGAN (right), comprising two discriminators. Class vectors c (real) and $\hat{c}$ (fake), are fed to all model components in both cases. An intermediate derivative calculation is observed in the cDVGAN plot, where the derivative of the synthetic sample is calculated. cDVGAN2 includes yet another discriminator applied to second-order derivatives. In cDVGAN and cDVGAN2, the total generator loss is calculated as a linear combination of the discriminator losses applied to synthetic samples.

this thesis.

7.2 Methods

7.2.1 Conditional Derivative GAN (cDVGAN)

It was discussed in the last chapter in Section 3.3.5 that the adversarial training process for GANs is known to be volatile. These problems are exacerbated when conditioning on multi-modal distributions of GW and glitch time-series. We saw in the last chapter that incorporating an auxiliary derivative discriminator leads to increased training stability of the model components and minimizes high-frequency artefacts in the time-domain GAN output, generating smoother and more faithful data [375]. In this section, we introduce two new conditional GAN designs, cDVGAN and cDVGAN2, that can help overcome these limitations. Under the conditional scheme, the cDVGAN generator loss is calculated as a linear combination of the two discriminator outputs. The standard GAN loss in 3.26 can be rewritten as

$$L_G(\phi_1, \phi_2, \theta) = -\eta_1 \mathbb{E}_{\hat{x}_1 \sim P_{\hat{x}_1}}[D_1(\hat{x}_1, \phi_1, \hat{c})] - \eta_2 \mathbb{E}_{\hat{x}_2 \sim P_{\hat{x}_2}}[D_2(\hat{x}_2, \phi_2, \hat{c})] \tag{7.1}$$

where D_1 and D_2, represent the first and second discriminator respectively. Here $\hat{x}_1$ represents synthetic samples while $\hat{x}_2 = d\hat{x}_1/dt$ represents the corresponding temporal derivatives and $\hat{c}$ represents the class vector for the synthetic samples. This formulation extends the generator loss of DVGAN (Equation 6.2) by incorporating the conditional term $\hat{c}$, thereby enabling class-controlled generation within the same

adversarial framework. $P_{\hat{x}_1}$ and $P_{\hat{x}_2}$ are the distributions of the two representations of generated data, and η_1 and η_2 are hyperparameters that control the relative strength of the discriminator losses. The cDVGAN method is not restricted to two discriminators. We extend cDVGAN to cDVGAN2, which includes a third discriminator that is applied to second-order derivatives. The models are identical except for the addition of the second-order derivative discriminator in cDVGAN2, while the second-order derivative discriminator is identical to the first-order discriminator except for the input size. As mentioned in the conclusions of Chapter 6, additional representations of the data (eg. time-frequency representations) can also be provided to additional discriminators, depending on the problem. For k discriminators applied to k representations of the data, the generator loss is written generally as

$$L_G(\phi_1, ..., \phi_k, \theta) = -\sum_{i=1}^{k} \eta_i \mathbb{E}_{\hat{x}_i \sim P_{\hat{x}_i}} [D_i(\hat{x}_i, \phi_i, \hat{c})] \tag{7.2}$$

Our cDVGAN and cDVGAN2 models are conditioned via projection, as described in Section 3.2.6. Since hyperparameter optimization is not the focus of this research, the hyperparameters $\eta_1 = \eta_2 = 0.5$ in cDVGAN and $\eta_1 = \eta_2 = \eta_3 = 0.33$ in cDVGAN2, meaning all discriminators contribute equally to the respective generator losses. The discriminators are updated 5 times for each generator update, similarly to DVGAN in the last chapter. The full model architecture can be viewed in the Appendix (C.1)[1]. While the overall architecture is structurally similar to DVGAN apart from the inclusion of the class-conditioning vector, additional modifications were made, notably, larger kernel sizes were adopted across the model components (14 and 18 for discriminator and generator, respectively), and the derivative discriminator was substantially expanded in size.

7.2.2 Training Data and Preprocessing

The GAN models in this study are conditioned on three different classes; two classes that are derived from Gravity Spy glitch classes called Blip and Tomte, and a third BBH signal class. The astrophysical nature that the BBH signals represent is not considered important during experimentation, and they are treated as another class of transient, glitch-like time-series for experimental purposes. Examples of the three classes can be observed in Figure 7.2. The Blip and Tomte glitches were described in detail in Section 1.4.4, while the BBH class corresponds to simulated inspiral and merger waveforms of binary black hole systems. All samples are of length $1,024$ and have a sampling rate of $4,096$ Hz, corresponding to 0.25 s of data. The GANs are

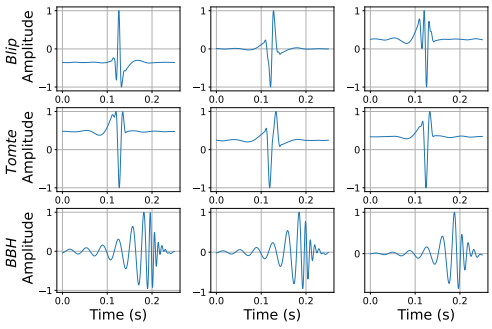
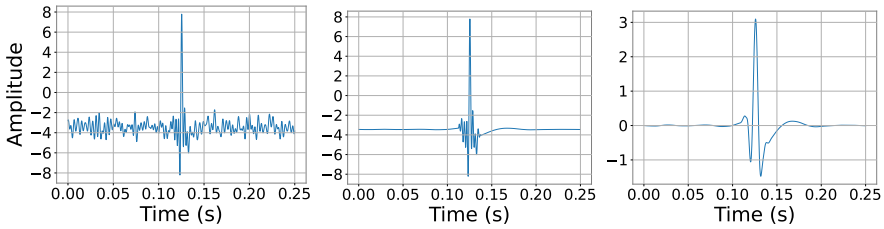


Figure 7.2: Examples of Blip (top), Tomte (middle), and BBH signals (bottom) used to train GAN models.

[1]Python code can be found at https://git.ligo.org/tom.dooney/cdvgan_paper.



(a) Whitened and band-passed sample

(b) Filtering either side of the peak

(c) A fully preprocessed blip (before rescaling)

Figure 7.3: Visualizations of the preprocessing steps applied to a blip glitch event.

trained on $7,500$ samples ($2,500$ samples from each class).

The Blip and Tomte datasets are constructed using confidences from Gravity Spy applied to the glitch triggers in LIGO's O3 run. Only Blip and Tomte events with Gravity Spy confidences of $c^1_{GS} \geq 0.9$ for their respective class are used in this study, and are extracted using GWpy's [376] `fetch_open_data` method, which provides an interface to the GWOSC[2] [350] data archive. The glitches are surrounded by stationary and uncorrelated noise, which can hinder the learning of GAN models.

To avoid the computational expense of BayesWave, as done in [317], the glitches are isolated from the background using Savitsky-Golay [371] filters. This requires the following preprocessing steps:

1. Firstly, 20 s of strain data is extracted, centred around the glitch GPS time provided by Gravity Spy.
2. The data is whitened, and a bandpass filter between $(20, 350)$ Hz is applied to the 20 s of data.
3. The data is cropped at 8,192 datapoints, centered around the GPS time of the glitch, corresponding to 2 s of data.
4. A window of 100 data points is isolated around the centre of the glitch, and two consecutive Savitsky-Golay filters with a polynomial of order 3 are applied to each side around the glitch centre iteratively, with window sizes of 501 and 301 respectively.
5. The two smoothed sides and the unsmoothed central peak are then concatenated, followed by applying 3 additional Savitsky-Golay filters iteratively with window sizes 41, 31 and 21 to the entire concatenated sample. The entire sample is finally cropped at 1,024 data points around the event GPS time, centered around 0 and rescaled to $(-1, 1)$.

The results of the above preprocessing steps are shown in Figure 7.3. The extensive use of Savitsky-Golay filters ensures that high-frequency noise artefacts are removed while preserving the overall shape. An analysis was made to investigate whether the characteristics of Blips and Tomtes are preserved after filtering. Gravity Spy

[2]The Gravitational Wave Open Science Centre

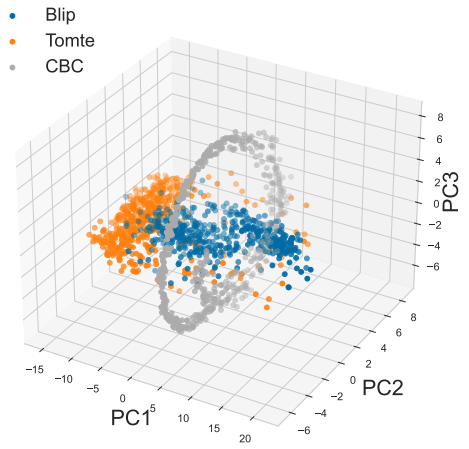


Figure 7.4: A plot of the first 3 principal components of the original samples. The separability of the three classes in this compressed representation indicates the diversity of the data.

generally classifies the preprocessed samples as their correct class when embedded in detector background for the SNR ranges considered in experiments (see section 7.2.3), indicating that their morphologies are analogous to real glitches.

All BBH signals are simulated with PyCBC [373] using the IMRPhenomD waveform routine from LALSuite [367], which generates the inspiral, merger and ringdown of a BBH waveform. The component masses are restricted to the range of $[30, 160]M_{\odot}$ with a spin of zero and fixing $m_1 > m_2$ and using only plus polarization.

We show the diversity of the three classes in Figure 7.4, where the dataset is represented by three separable clusters in a reduced principal component space.

7.2.3 Experimental Procedure

GAN Benchmarks

Similarly to the previous chapter, this work uses ablation studies to compare cDVGAN and its cDVGAN2 extension with three other baseline GAN models in their ability to generate data useful for training detection algorithms. In this case, the ablation involves selectively disabling auxiliary first and second-order derivative discriminators during the training of GAN models to investigate if they are successful in improving the features captured in synthetic GW signals and glitches. To construct an appropriate ablation study, the first benchmark is a conditional variant of a vanilla Wasserstein GAN (cWGAN), which comprises the same architecture as cDVGAN except for the derivative discriminator. This allows us to investigate the effect of including a first-order derivative discriminator during GAN training. cDVGAN2 allows us to investigate the effect of including a second-order derivative discriminator on top of the first-order derivative discriminator.

The second benchmark is McGANn [318], described in Section 3.3.5, since it suc-

cessfully replicates the features of simulated GW bursts. As a third benchmark, we developed a modified McGANn model, called McDVGANn, that uses a second auxiliary discriminator applied to first-order derivatives, similar to cDVGAN. The architecture is identical to McGANn except for the addition of a derivative discriminator. In McDVGANn, the derivative discriminator is identical to the base discriminator except for the input size. This benchmark investigates whether the idea of derivative discriminators can generalize to other GAN models.

McGANn and McDVGANn are conditioned using concatenation similar to their paper [318] and are trained with an Adam optimizer, binary cross-entropy loss function and learning rate 2×10^{-4}. Conversely, cWGAN and cDVGAN2 are conditioned via projection similarly to cDVGAN and are trained with RMSProp, a Wasserstein loss function and a learning rate of 2×10^{-4}. All models are trained for 500 epochs with a batch size of 512. Unless otherwise specified, all GANs are trained using the same standard hyperparameters as prior works.

GAN-generated datasets

We construct 3 different datasets from each GAN by sampling the respective generator's class space, as suggested by [318]. The three variants of GAN-generated datasets are as follows:

1. **Vertex:** The vertex class space corresponds to the vertices of the three-dimensional class space. The locations of the vertex class space are the same as the training set class space locations used to train the GAN models and are the closest representation to the training set. The vertex class vector is one-hot encoded corresponding to one of the three GAN training classes eg. [1, 0, 0] corresponds to the Blip class.

2. **Simplex:** The simplex class space corresponds to points on a $k = 2$ simplex (2D triangle) in the case of three classes. Class vectors are constructed by sampling points uniformly on this simplex. The simplex can be considered the simplest surface that intersects all three training classes, and all simplex class vectors sum to 1 (i.e. $\sum c_i = 1$). Variations are observed in the samples, with some having characteristics that strongly resemble the training classes, due to one class dominating the others. The simplex dataset is a superset of the vertex dataset and represents synthetic data outside the training data distribution.

3. **Uniform:** For the uniform dataset, each entry in the class vector is uniformly sampled from $\mathrm{U}[0, 1]$, corresponding to sampling uniformly within a cube with dimensions 1x1x1. The uniform dataset is a superset of the simplex and vertex datasets and, like the simplex dataset, represents data from outside of the training data distribution. The uniform dataset exhibits the largest variety since it explores regions of the class space further than the simplex dataset relative to the training set vertices.

Figure 7.5 shows examples from each of the cDVGAN-generated datasets (vertex, simplex and uniform). Figure 7.6 shows plots of the first 3 principal components (PCs) of real and GAN-generated samples from cDVGAN. The vertex samples (Figure 7.6a) generally match their corresponding classes from the real data distribution. Figure

7.6b shows that the simplex and uniform hybrid datasets populate spaces between the class clusters, while Figure 7.6c shows that the uniform dataset covers a larger part of the PC space than the simplex dataset. This is intuitive since the simplex space is a subset of the uniform space.

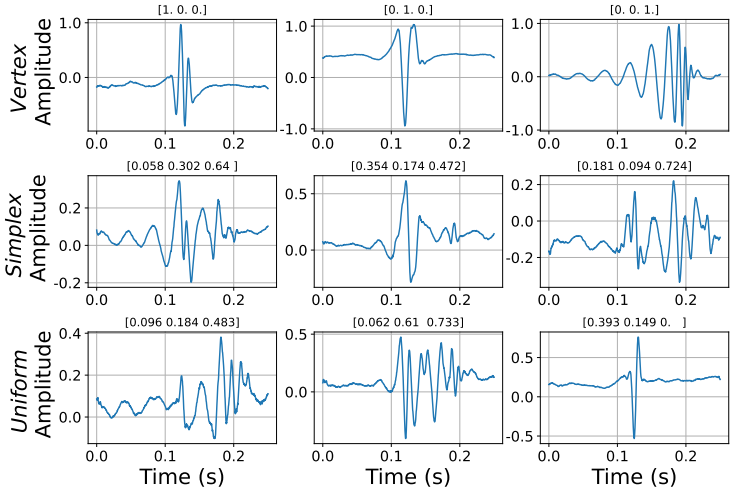


Figure 7.5: Examples of vertex (top), simplex (middle) and uniform samples (bottom) from cDVGAN. The corresponding class vector is shown above each sample.

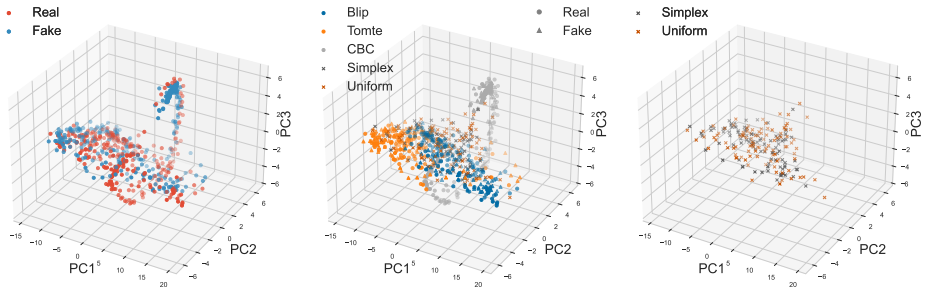
Downstream search with CNNs

The experiments followed in this study investigate the effectiveness of GAN-generated data for training CNNs (see Section 3.2.5) to detect real data in additive Gaussian noise.

The CNNs' objective is to perform the binary classification of two classes: samples in additive detector noise and detector noise only, examples of which are shown in Figure 7.7. It takes a time-series input of dimension $1,024$, representing $0.25\,\mathrm{s}$ of LIGO strain data. The CNN architecture is kept constant for training with each dataset and can be viewed in Appendix C.2.

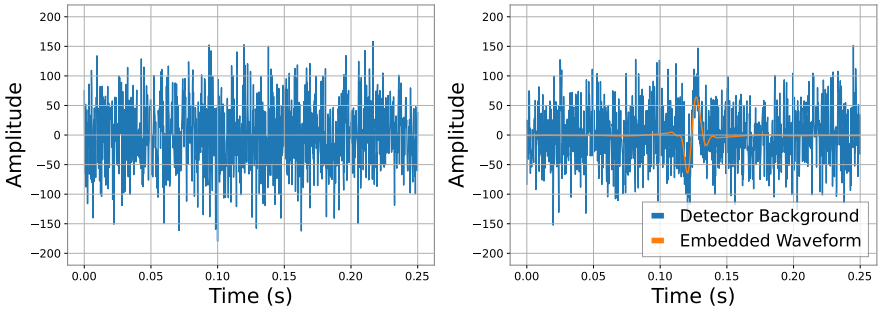
Better synthetic data will result in CNNs with better detection efficiency on a real data subset from the GAN training data distribution. We also investigate if training CNNs with GAN-generated hybrid datasets (simplex and uniform) can improve the detection efficiency beyond training solely with the standard three classes.

The training and testing samples are injected additively into detector noise from Hanford (H1) or Livingston detectors (L1) during O3 for each of the above datasets. The detector noise is extracted from GWOSC, using trigger times for Gravity Spy's No_glitch class, extracing $14\,\mathrm{s}$ around each No_glitch GPS time. The background is sampled at $4,096\,\mathrm{Hz}$, similar to the GAN training data. The LIGO detector noise is first whitened using PyCBC's whitening function, with the first and last $2.5\,\mathrm{s}$ removed due to artefacts at the noise boundaries. The preprocessed detector noise is then split into chunks of length $1,024$, the same dimensionality as the GAN input and output.



(a) Real and Vertex datasets (b) All datasets (c) Hybrid datasets

Figure 7.6: The first 3 principal components (PCs) of real and GAN-generated samples from cDVGAN. The vertex samples from cDVGAN generally match the real samples in the PC space while hybrid samples populate intermediate regions between the clusters for the 3 classes. Figure 7.6c shows that the uniform dataset covers a larger space than the simplex dataset.



(a) Pure LIGO detector noise (0 class) (b) Detector noise + glitch (1 class)

Figure 7.7: Examples of the two classes predicted by CNN models. The injected sample on the right is scaled to an SNR of 8 before injection (shown in orange for clarity).

Following this procedure, we accumulate just over $250,000$ background samples in total. Since the GAN training data and output are scaled between $[-1, 1]$ the samples are scaled to a signal-to-noise ratio (SNR) ratio that is sampled uniformly on $\mathrm{U}[1, 16]$ before injecting them into the preprocessed detector noise. This is done in the same way as the scaling of the training data for DeepExtractor in Chapter 4, following Equation 4.3.

For each GAN, we train three separate CNNs on the three generated datasets (vertex, simplex, uniform). For the vertex dataset the three different vertex locations in the class space are sampled with equal probability. For the uniform and simplex datasets, samples are drawn uniformly from their respective spaces.

Each training dataset comprises $100,000$ samples, with 50% glitch/signal plus Gaussian noise and 50% Gaussian noise only. We test each CNN on a real data

(a) cDVGAN (b) cDVGAN2 (c) cWGAN (d) McGANn (e) McDVGANn

Figure 7.8: ROC-Curves for the different GAN-generated datasets from each GAN.

Dataset	cDVGAN (ours)	cDVGAN2 (ours)	cWGAN	McGANn	McDVGANn (ours)
Vertex-Trained	0.771 ± 0.012	0.758 ± 0.008	0.762 ± 0.018	0.768 ± 0.016	0.768 ± 0.022
Simplex-Trained	***0.802*** ± 0.019	0.789 ± 0.010	*0.788* ± 0.009	0.759 ± 0.010	*0.786* ± 0.012
Uniform-Trained	0.797 ± 0.022	*0.791* ± 0.009	0.777 ± 0.014	*0.770* ± 0.014	0.778 ± 0.012

Table 7.1: The Area-Under-Curve (AUC) yielded on a real test set by CNNs trained on synthetic datasets from each GAN. The results represent the mean AUC over 5 iterations, with bounds given by the standard deviations. The overall best result is shown in **bold**, while the best result per GAN is shown in *italic*.

subset comprising 7,500 samples (2,500 from each class), sampled randomly from a distribution of 20,850 samples (6,950 from each class). This results in a test dataset of 15,000 samples in total (50% LIGO noise only) in each iteration of training/testing. We ensure that none of the GAN training data appear in the test set, although they are taken from the same distribution.

The results are presented using the area-under-the-curve (AUC) metric (see Section 3.1.6). It provides a single metric between 0 and 1 that summarizes the overall discriminatory power of a model across different classification thresholds. We repeat the experiment 5 times and report the mean results, randomly generating the training data and randomly sampling the backgrounds and test data in each iteration. We keep the real test datasets constant for each GAN-generated dataset in each iteration.

Fitting-factor study

We investigate the accuracy of cDVGAN's BBH signals under a fitting-factor study, showing that most of the synthetic signals are consistent with the original signals simulated with IMRPhenomD. As discussed in Section 2.4.2, the fitting-factor of a signal is defined as the maximum match of that signal over the templates of a template bank [377].

While cDVGAN's signals are used as the signal injections in a matched-filter search, the template bank is created using `mbank` [152], and the corresponding templates are simulated using IMRPhenomD. We aim for good coverage of the parameter space of cDVGAN's training signals ($30 \leq m2 \leq m1 \leq 160$) beyond the 97% requirement for matched-filter standards, using a holdout dataset from the cDVGAN BBH distribution to validate the template bank[3]. We sample `mbank's` normalizing flow until the parameter space is covered evenly with an average fitting factor of over 99% (only approximately 5% of signals yield a fitting factor of under 99%. This yields

[3]We validate the bank using a flat PSD, required for time-domain match calculations using `mbank`

1,500 templates, allowing for a thorough exploration of the parameter space.

We calculate the fitting factor in the time-domain on a fixed time grid since the training signals and generated signals are truncated at 0.25 s around the merger time. We compute the fitting-factor over the 2,500 GAN training samples simulated using IMRPhenomD and compare it to the fitting-factor computed on 2,500 synthetic signals from cDVGAN.

An important caveat to highlight is that the class-conditional approach featured in cDVGAN is focused towards a glitch generator application rather than a waveform generator for GW searches. To achieve better accuracy in the latter case, it would likely be better to modify the cDVGAN architecture to learn BBH signals only, reducing it to the DVGAN architecture in the previous chapter. This would simplify the training schedule to learn one distribution of signals rather than three distributions in one model. Furthermore, the model could also be conditioned on continuous source parameters rather than discrete class information, giving the user control over the parameter space they wish to generate signals in.

7.3 Results

7.3.1 Training with GAN data

Dataset	cDVGAN (ours)	cDVGAN2 (ours)	cWGAN	McGANn	McDVGANn (ours)
Vertex-Trained	0.689 ± 0.009	0.680 ± 0.006	0.685 ± 0.017	*0.687* ± 0.010	0.668 ± 0.020
Simplex-Trained	0.698 ± 0.013	0.686 ± 0.007	*0.695* ± 0.010	0.673 ± 0.008	*0.676* ± 0.010
Uniform-Trained	***0.702*** ± 0.014	*0.693* ± 0.008	0.692 ± 0.009	0.680 ± 0.013	0.671 ± 0.006

Table 7.2: The AUC for test samples under an SNR of 8. The results represent the mean AUC over 5 iterations, where the bounds are calculated using the standard deviations over the 5 iterations. The best result overall is shown in bold text, while the best result per GAN is shown in italic text.

Dataset	cDVGAN (ours)	cDVGAN2 (ours)	cWGAN	McGANn	McDVGANn (ours)
Vertex-Trained	0.844 ± 0.015	0.827 ± 0.010	0.830 ± 0.022	0.841 ± 0.021	0.856 ± 0.027
Simplex-Trained	***0.893*** ± 0.029	*0.881* ± 0.013	*0.867* ± 0.010	0.836 ± 0.012	*0.885* ± 0.013
Uniform-Trained	0.883 ± 0.031	0.878 ± 0.011	0.851 ± 0.020	*0.852* ± 0.020	0.873 ± 0.014

Table 7.3: The AUC for samples above an SNR of 8. The results represent the mean AUC over 5 iterations, where the bounds are calculated using the standard deviations over the 5 iterations. The best result overall is shown in **bold** text, while the best result per GAN is shown in *italic* text.

The AUC values for each CNN model over the entire SNR range (1-16) are shown in Table 7.1. We also scale the test sets between SNRs of 1-8 and 8-16 and record the AUC of the same CNNs to investigate performance for quieter and louder samples, which can be seen in Tables 7.2 and 7.3 respectively. Examples of ROC curves from one of the testing iterations are shown in Figure 7.8. All tables show that the simplex and uniform datasets from cDVGAN yield the highest AUC results for CNNs, with

the simplex dataset yielding the highest overall AUC.

Ablation Studies. The ablation between cDVGAN and cWGAN in Table 7.1 shows that the adversarial feedback from the first-order derivative discriminator improves the features of the synthetic data in all three datasets. This suggests that the first-order derivative discriminator was successful in improving the GAN output. cDVGAN2 was unsuccessful in improving upon cDVGAN's results, yielding a slightly lower performance, but improves upon the performance of cWGAN's simplex and uniform datasets, particularly for the higher SNR range (Table 7.3).

The AUC performance yielded from McDVGANn's datasets is competitive among the GAN models. Both its simplex and uniform datasets yield better AUC performance than McGANn's counterparts, with the simplex dataset yielding the highest overall AUC performance between the two models. The vertex datasets from McGANn and McDVGANn yield comparable performance. This suggests that the derivative discriminator can be effective in a more traditional GAN architecture.

These results show that incorporating derivative discriminators can improve the synthetic data in multiple conditional GAN architectures, and indicate that analyzing first-order derivatives in a separate auxiliary discriminator is superior to using both first and second-order derivative discriminators for modelling the dataset covered in this study (in the cWGAN architecture). For all GAN models, hybrid datasets provide the best overall AUC performance for the CNN. This suggests that GAN-generated hybrid samples are useful for searching for multiple classes of real data when obscured by the detector background. This might offer interesting applications in glitch searches, particularly for those with no clear correlation to auxiliary channels. Such glitches could be conditioned into cDVGAN to generate hybrid samples specific to a subset of LIGO glitch classes for use in a glitch detection algorithm. Combining all three GAN-generated datasets for training CNNs may improve upon these results yet again, although this is left to future work.

SNR	100:0	75:25	50:50	25:75	0:100
1–16	0.900 ± 0.001	0.898 ± 0.002	0.893 ± 0.002	0.887 ± 0.002	0.802 ± 0.019
1–8	0.799 ± 0.001	0.792 ± 0.005	0.787 ± 0.003	0.777 ± 0.003	0.698 ± 0.013
8–16	0.988 ± 0.001	0.987 ± 0.001	0.987 ± 0.001	0.985 ± 0.001	0.893 ± 0.029

Table 7.4: AUC values over three SNR ranges for different proportions of real:synthetic samples for a training set fixed at 100,000 samples. The results are represented by the mean AUC and standard deviation over 5 iterations.

7.3.2 Combining real and cDVGAN data for improved training

In this section, we augment real datasets with GAN-generated data to improve upon the classification performance in the previous section. We augment the real data with the simplex dataset from cDVGAN since it yields the best performance. We compare cDVGAN data with traditional duplication of real training samples for data augmentation.

Maintaining a real hold-out test set of 7,500 samples (2,500 from each class) as

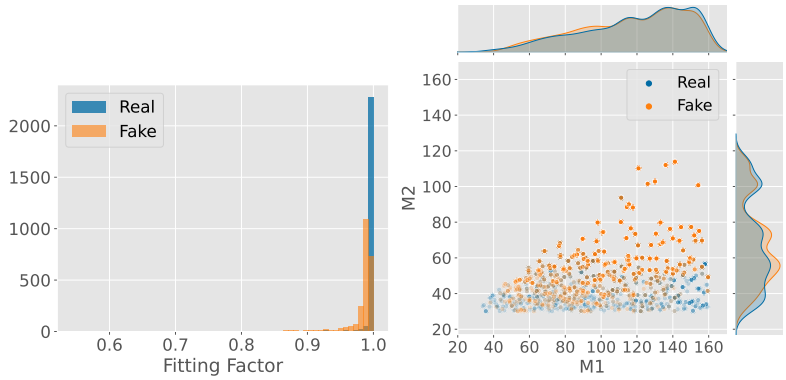
in the previous section, we use all remaining samples from the real distribution for training, which amounts to 13,350 samples (4,450 from each class). Fixing the training data size again at 100,000 (50,000 glitch/signal + noise samples, 50,000 noise-only samples), we vary the proportion of real and GAN-generated samples. Since there is no limit to generating cDVGAN data, we duplicate the real training data before injection to reach the required number of samples. This is done to control the effects of the background noise on the training of CNNs, since CNNs are sensitive to the number of different backgrounds seen during training.

The results in Table 7.4 show that the performance drops only slightly with smaller proportions of real data. The CNN performance remains competitive even with only 25% of real training samples with only a 1% drop in overall AUC performance compared to only using real data. The second and third rows of Table 7.4 show that this decrease in AUC performance occurs mostly for lower SNR (<8) samples. The decrease in performance for louder samples is minimal (<1% decrease). This indicates that the synthetic data is competitive for augmenting the training data for CNNs when including a relatively small amount of real data in the training set. Although there is a more substantial decrease in performance when using 100% synthetic samples, this might be improved by including other synthetic datasets from cDVGAN in the training schedule.

7.3.3 Fitting-factor results

The results of the fitting factor study indicate that cDVGAN's BBH signals are generally consistent with IMRPhenomD. The search calculates an average fitting-factor of 0.994 ± 0.0423 for the $2,500$ real cDVGAN training signals with 5^{th} and 10^{th} percentiles at 0.972 and 0.994 respectively. Conversely, the experiment yields an average fitting-factor of 0.976 ± 0.045 for $2,500$ synthetic cDVGAN signals with 5^{th} and 10^{th} percentiles at 0.893 and 0.951 respectively.

Figure 7.9 shows a histogram of the fitting-factors yielded from the real and syn-



(a) Fitting-factor (b) Match parameters

Figure 7.9: Plots of fitting-factors and corresponding best-fit template parameters for real GAN training signals and synthetic cDVGAN signals.

thetic signals and the distribution of the recovered parameters of both datasets. Figure 7.9a shows the fitting-factor distribution of cDVGAN's signals is similar to that of the real training signals, although a minor decrease in accuracy is observed. Figure 7.9b shows that cDVGAN's signals cover most of the parameter space well, although there is a slight mismatch in the lower ends of both the $M1$ and $M2$ distributions.

Although the accuracy of cDVGAN's signals is lower than that of IMRPhenomD, the quality of cDVGAN's signals and coverage of the parameter space could be improved with further training, or by considering a different modelling approach as outlined in Section 7.2.3. Furthermore, cDVGAN is competetive with state-of-the-art surrogate models in terms of inference speed on a CPU [378, 379], and far surpasses them with the use of a GPU. For example, cDVGAN can generate $2,500$ signals of the learned BBH class in approximately 18 s on a CPU, and only 0.04 s on a GPU.

7.4 Conclusion

This chapter extended the DVGAN framework introduced in the previous chapter into a novel conditional setting, presenting the Conditional Derivative GAN (cDVGAN). The model was developed to generate diverse classes of time-domain GW signals, including two unmodeled glitch types (Blip and Tomte) and one class of modeled BBH waveforms. Like DVGAN, cDVGAN incorporates auxiliary adversarial feedback on the first-order derivatives of the data through a secondary discriminator, encouraging smoother, more physically consistent data generation on a multi-modal dataset. In addition, its conditional formulation allows explicit control over class composition, enabling the generation of both class-specific and hybrid samples that continuously interpolate between known categories—effectively extending the model's expressive power beyond the training distribution.

Through a series of ablation studies, we evaluated the impact of derivative-informed adversarial feedback on model performance in a downstream detection task. Specifically, we trained CNNs to detect real samples embedded in simulated LIGO noise using synthetic data generated by different GAN variants. Comparisons between cDVGAN and its baseline cWGAN demonstrated that the inclusion of derivative-based discrimination improves the utility of synthetic data for CNN training, yielding higher detection performance. Further experiments comparing cDVGAN2—which incorporates second-order derivative feedback—showed that while its performance was competitive, the additional derivative information did not lead to measurable gains under this experimental setup. A third ablation comparing McDVGANn with the existing McGANn [318] further confirmed that derivative-informed adversarial feedback can enhance training stability and realism, even in traditional conditional GAN architectures. Overall, these results indicate that incorporating adversarial feedback on temporal derivatives strengthens GAN learning on continuous time-series data, particularly for transient phenomena relevant to GW physics.

We also demonstrated the practical value of diverse, GAN-generated hybrid samples for training detection algorithms. Among all tested datasets, cDVGAN's simplex-generated hybrids yielded the best overall CNN performance, while hybrid datasets from other GANs also outperformed those trained solely on pure (vertex) samples. Lastly, our final experiment combines GAN-generated data with real data to improve the performance of CNN models for glitch and signal searches, showing cDVGAN as

a viable approach for data augmentation.

Lastly, we implemented a fitting-factor study that shows cDVGAN's BBH signals are consistent with the IMRPhenomD waveform routine used to generate the cDVGAN training signals. Although there are some inconsistencies and a small decrease in accuracy, the synthetic signals generally match well with a template bank for the corresponding parameter range. The cDVGAN signals have good coverage on most of the parameter range, and can be generated efficiently, particularly with the use of a GPU.

Although hyperparameter optimization was not the primary focus of this study, several avenues for improvement remain. Future work could explore optimized weighting of the discriminator losses (the η parameters in Equation 7.1), or introduce consistency regularization [380] to further stabilize training. Additionally, expanding the model to operate on alternative data representations—such as spectrograms or wavelet coefficients—may enhance generative fidelity. Finally, the use of more advanced CNN architectures or alternative detection frameworks could better exploit the diversity of GAN-generated training data, paving the way for efficient and scalable analysis methods suitable for next-generation detectors.

In the next and final chapter, we demonstrate how DeepExtractor can be used to extract realistic glitch events from detector noise to construct high-fidelity datasets for conditioning cDVGAN, thereby expanding its generative scope. Such generalized hybrid datasets, spanning broader class spaces, may further improve data augmentation and robustness in future GW analysis pipelines.

Chapter 8

Learning the Glitch Space with Derivative GANs

This work develops a class-conditional generative model, cDVGAN, capable of synthesizing seven of the most common glitch types observed during LIGO's third observing run: Blip, Fast Scattering, Koi Fish, Low-Frequency Burst, Scattered Light, Tomte, and Whistle. The model is trained on high-quality reconstructions produced by the DeepExtractor framework, which accurately represents the morphology of real LIGO glitches. While the previous chapter demonstrated cDVGAN as effective for reproducing proxy signals and simplified glitch datasets, this study extends its application to a more complex and realistic glitch population. We show that cDVGAN generalizes effectively, learning to reproduce a diverse and physically consistent glitch space directly from these reconstructions. Moreover, because the model is conditioned on glitch class, it can generate hybrid or transitional glitch morphologies by interpolating across the class-conditioning vector after training. Model outputs are validated against real data using unsupervised UMAP embeddings and Gravity Spy, widely used in the GW community for glitch classification. The UMAP analysis shows substantial overlap between real and synthetic samples in the reduced latent space, while Gravity Spy classifies the majority of cDVGAN's synthetic glitches as the correct class. These analyses demonstrate the capability of generative models to produce realistic synthetic glitches suitable for a range of downstream applications. At the same time, these results reveal the limitations of relying solely on Q-transform–based (magnitude spectrogram) representations, as employed by classifiers such as Gravity Spy, which can confidently misclassify unrealistic samples from less robust generative models. They highlight the critical role of phase information for accurately capturing glitch morphology, which is discarded in standard Q-transforms used for glitch classification, and underscore the need for complementary unsupervised validation methods to achieve more reliable glitch characterization.

This chapter is in preparation for submission to:

Tom Dooney et al.. *Monthly Notices of the Royal Astronomical Society*, 2026.

8.1 Introduction

It has been emphasized throughout this thesis that the ability to synthesize accurate and reliable glitch distributions is of growing importance. Such capability enhances the realism of detector simulations (e.g., mock data challenges), enabling more rigorous testing and validation of GW detection pipelines and parameter estimation algorithms. In this final chapter, we advance these efforts by extending the development of generative models beyond the research presented in previous chapters. While the preceding chapters introduced novel GAN architectures for generating simulated and proxy time-domain signals, this work focuses on modelling a broader set of *realistic* glitch samples in a conditional framework.

The previous chapter demonstrated that independent GAN models such as DVGAN can successfully reproduce individual classes—notably the Blip glitch class—with high fidelity, benefiting from the simplicity of learning a single distribution. These results were comparable to using gengli [317], further highlighting the promise of generative modelling in replicating time-domain glitch morphologies. Other approaches, such as those presented in [316], extended this concept to the spectrogram domain by training separate GANs on Q-scan representations of each Gravity Spy class. However, extending this holistic approach to the time domain remains unexplored, despite its advantages in preserving phase information and enabling the direct use of time-series data in simulations and mock data challenges.

These studies suggest that generative modelling can be applied across the full glitch class space, provided that effective and computationally feasible reconstruction methods are available; a condition now satisfied with the development of DeepExtractor[1]. Earlier models such as gengli and DVGAN were trained on BayesWave reconstructions, which produce smooth time series composed of Gabor–Morlet wavelets (see Section 1.4.4). While these models accurately reproduce such smooth signals, their capacity to model DeepExtractor reconstructions remains untested, which impose no assumptions about waveform smoothness and may include residual noise artifacts. This introduces a new challenge, as DeepExtractor reconstructions are inherently more realistic but likely more complex to model using deep generative models.

Chapter 7 further demonstrated that cDVGAN can conditionally generate multiple classes within a single model, including two proxy glitch classes (Blip and Tomte) and BBH signal class. Since the glitch datasets were preprocessed with smoothing filters that facilitated convergence, and the BBH class was generated using waveform models, it remains unclear whether this approach generalizes to the reconstructions produced by DeepExtractor. If successful, however, a conditional model capable of learning multiple glitch classes within one framework would offer several important advantages: (i) reduced training effort compared to separate per-class models; (ii) improved data efficiency and generalization by combining all glitch classes into a single training set; (iii) potential for knowledge transfer between morphologically related classes; and (iv) the ability to interpolate between classes through conditioning, generating diverse hybrid glitches useful for tasks such as detection, classification, and robustness testing, which we saw in the previous chapter.

In this chapter, we evaluate cDVGAN's capability to generate time-domain glitches

[1]The computational cost of BayesWave makes it impractical for constructing the large, holistic datasets required to train deep generative models, as discussed in Chapter 7.

from a class space that is substantially larger, more realistic, and more complex than the well-behaved glitches and simulated signals examined in Chapters 6 and 7. The training dataset is constructed from DeepExtractor reconstructions of seven of the most common glitch classes observed during O3: Blip, Fast Scattering, Koi Fish, Low-Frequency Burst, Scattered Light, Tomte, and Whistle. After training, we validate the model by comparing the generated glitches with a real data subset using UMAP to visualize overlap between the corresponding distributions in a reduced latent space. We further assess the fidelity of the synthetic glitches using the Gravity Spy classifier to determine whether they are identified as their intended class.

While some classes are reproduced more accurately than others, the results demonstrate that cDVGAN captures much of the morphological diversity across glitch types, with substantial overlap between real and generated classes in the UMAP projection. Moreover, most synthetic glitches are correctly classified by Gravity Spy, indicating that cDVGAN effectively learns the salient morphological features of real glitches as recognized by a state-of-the-art classifier trained on extensive real data. Finally, we highlight a limitation of relying solely on magnitude-based time–frequency representations, such as the Q-transform used by Gravity Spy. When classifying unrealistic Low-Frequency Burst samples generated by a less robust diffusion-based model [222], Gravity Spy still assigns them to the correct class with high confidence. Although the corresponding Q-scans resemble those of real glitches, the samples are clearly inconsistent with their real counterparts in the time domain, underscoring the need to complement supervised classifiers with approaches, potentially unsupervised, that employ representations preserving phase information.

8.2 Data

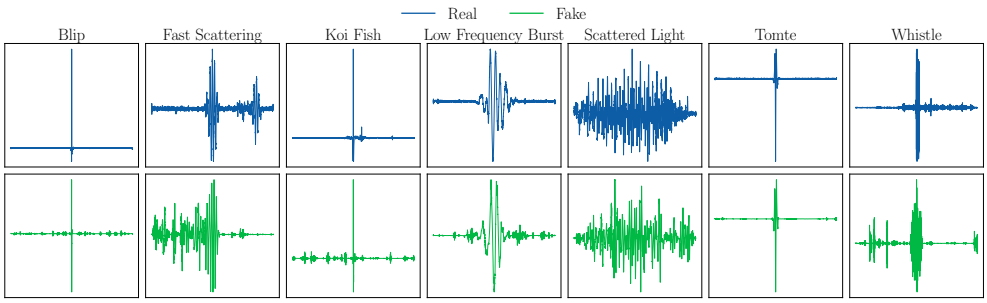


Figure 8.1: Comparison between real and synthetic glitch waveforms across the seven glitch classes used for training. **Top:** Real glitches reconstructed with DeepExtractor. **Bottom:** Synthetic glitches generated by cDVGAN after training. Each panel shows one representative glitch per class, illustrating the model's ability to reproduce distinct morphological features observed in the real data.

We curate the training data for our generative models by using DeepExtractor to reconstruct glitch samples from the Gravity Spy dataset. Our focus is restricted

to the LIGO's Hanford and Livingston detectors during both O3a and O3b, aiming to collect an approximately equal number of samples per class across detectors and observing runs. To ensure reliability, we further limit the selection to glitches with a Gravity Spy confidence $\sigma_{GS} \geq 0.9$ for their assigned class and an SNR ≥ 15. This ensures that each glitch is morphologically representative of its class and sufficiently loud relative to the background, minimizing the presence of residual noise artifacts that DeepExtractor might otherwise reconstruct.

We focus on the seven most prominent glitch classes observed during O3: Blip, Fast Scattering, Koi Fish, Low-Frequency Burst, Scattered Light, Tomte, and Whistle. Although the Extremely Loud class is also common, it is excluded from this study since its characterization is dominated by exceptionally high SNR rather than distinctive morphology. Beyond their prevalence, these seven classes collectively span wide time and frequency ranges, encompassing much of the morphological range relevant to transient noise in GW detectors. This diversity makes them an ideal testbed for generative modelling, as a single framework capable of reproducing such a broad spectrum of glitch morphologies can be leveraged for multiple downstream applications, including realistic detector simulations, mock data challenges, and the development of more resilient data analysis pipelines.

The distribution of the curated data across glitch classes, detectors, and observing runs is summarized in Table 8.1. While the goal was to obtain 5,000 samples per class (1,250 per detector per observing run) under the selection criteria of $\sigma_{GS} \geq 0.9$ and SNR ≥ 15, some corresponding GPS times recorded in the Gravity Spy dataset were unavailable from GWOSC, preventing the full target from being reached for all classes. The most affected classes were Fast Scattering with 2,774 samples and Low-Frequency Burst with 4,316 samples, giving a total dataset size of 31,511 samples. Such class imbalances can adversely affect GAN training, as the model may bias toward classes with larger sample counts. To ensure stable training of cDVGAN, we applied bootstrapping (sampling with replacement) to oversample all classes to 5,000 samples each, yielding a balanced training set of 35,000 samples in total.

Before training, each reconstructed time series is normalized to the range $[-1, 1]$, and its mean is subsequently subtracted. This standardization ensures consistent scaling across samples, simplifying the learning process by allowing the model to focus on class-specific morphological features rather than variations in scale. While this normalization constrains the generative models to operate at a single effective SNR scale, the generated signals can later be rescaled to any desired SNR, depending on the target application. As in Chapter 7, we compute the time-domain derivative of each glitch waveform, which is provided to the auxiliary discriminator during training, and transform the class labels into one-hot vectors. The main difference between this dataset and that used in Chapter 7 lies in the use of *reconstructed* glitch time series using DeepExtractor rather than the smoother proxy glitches curated for that earlier study, as well as the higher temporal resolution of 8192 samples (corresponding to 2 s at a sampling rate of 4096 Hz), compared to $1,024$ data points in Chapter 7.

Table 8.1: Complete size of dataset. Amount of entries per glitch class per observing run and detector Hanford (H1) and Livingston (L1)

Glitch Class	H1		L1	
	O3a	O3b	O3a	O3b
Blip	1246	1247	1245	1243
Fast_Scattering	143	143	1244	1244
Koi_Fish	1247	1248	1243	1247
Low_Frequency_Burst	1246	1246	912	912
Scattered_Light	1236	1243	1246	1248
Tomte	1031	1031	1247	1246
Whistle	1232	1232	1231	1232

8.3 Methods

8.3.1 Generative models

The cDVGAN model used in this study follows the configuration described in Chapter 7, trained for 500 epochs with a batch size of 64. The only modification concerns the dimensionality of the input and class-conditioning vectors, adjusted to match the dataset described in the previous section. Although several architectural or training refinements could improve cDVGAN's performance (see Section 8.4), optimizing GANs for specific datasets is computationally intensive, requiring multiple training and validation cycles, and therefore lies beyond the scope of this work. Instead, we evaluate the 'out-of-the-box' generalization capability of cDVGAN, previously validated on a simpler problem in the previous chapter, when applied to a more complex and realistic glitch dataset spanning a wide range of time–frequency morphologies. This represents a challenging test for cDVGAN and generative models more broadly, as capturing such diverse temporal and spectral structures within a single configuration is inherently difficult.

We also experimented with diffusion-based generative models trained on the same glitch dataset, adapting the original diffusion framework [222] for one-dimensional GW time series. All architectures were modified to use 1D convolutional layers to accommodate temporal rather than spatial data. An initial U-Net was employed to model the reverse diffusion process but struggled to capture long-range temporal dependencies. To address this, we replaced the U-Net with a transformer-based predictor, yielding a Diffusion Transformer (DiT) architecture [381]. However, the full-resolution DiT proved computationally prohibitive, motivating the introduction of a convolutional autoencoder for dimensionality reduction.

The autoencoder compresses each input glitch from (8192, 1) to (256, 16), reducing the temporal dimension by a factor of 32 while expanding the feature depth. Although this introduces minor reconstruction dependence, it significantly decreases input dimensionality, reducing the diffusion model's computational complexity to approximately 1/1024 of the original. Even with doubled internal width to accommodate additional channels, the expected generation time decreases from roughly 120 s to 0.2 s. As shown in Appendix D.1, the reconstructed waveforms closely match the inputs,

demonstrating that the autoencoder preserves essential morphological features and provides a compact latent representation well-suited for diffusion-based generation.

8.4 Results

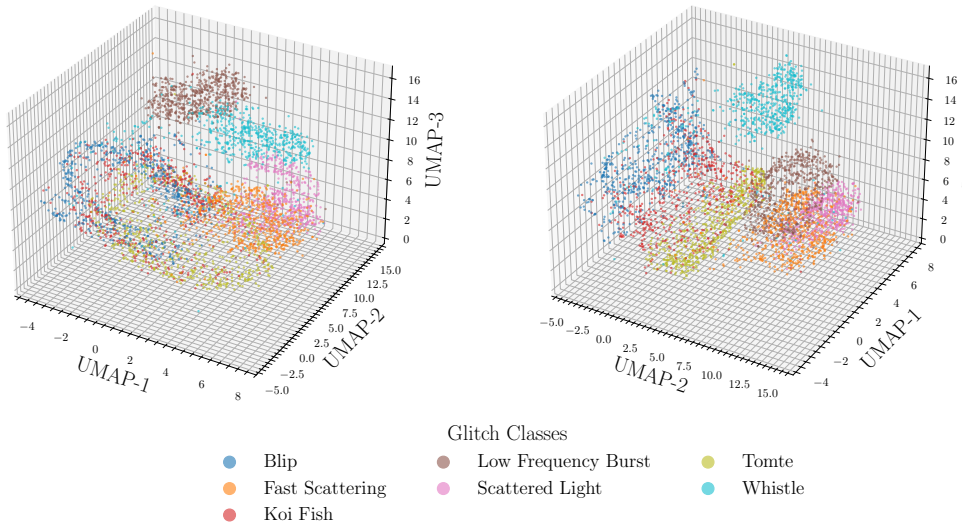


Figure 8.2: Two complementary 3D views of the UMAP embeddings showing real and generated glitch samples. Each color corresponds to one of the seven glitch classes used during training.

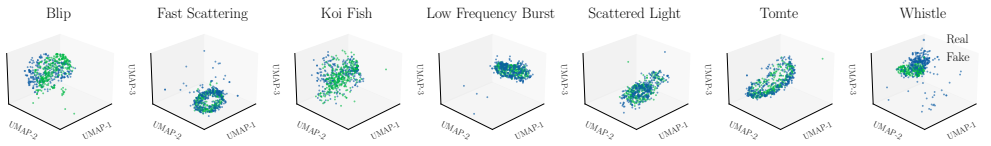
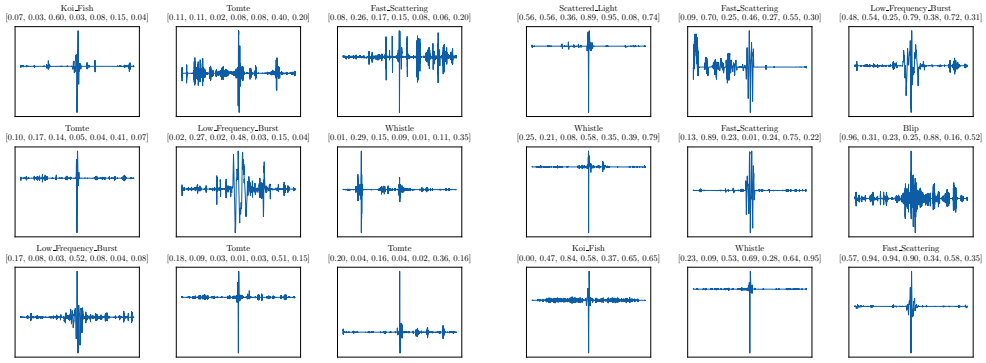


Figure 8.3: Three-dimensional UMAP embeddings for each glitch class, comparing real (blue) and cDVGAN-generated (green) samples. Each panel corresponds to one glitch type and is shown from a rotated viewpoint to highlight intra-class structure. The close overlap between real and synthetic distributions across most classes demonstrates that cDVGAN effectively learns class-specific morphological features, though some variation remains for more complex glitch morphologies.

Global similarity between real and synthetic glitches. We assess similarity between real and generated glitches by projecting both datasets into a three-dimensional latent space using UMAP. Figures 8.2 and 8.3 show the global and class-wise embeddings (500 samples per class).



(a) Simplex mixing ($\sum c_i = 1$). (b) Uniform mixing ($c_i \in [0, 1]$).

Figure 8.4: Examples of hybrid glitch waveforms generated by cDVGAN using two class-conditioning strategies. **(a)** Simplex mixing samples each class vector from $k = 7$ simplex, ensuring coefficients sum to one. **(b)** Uniform mixing samples each coefficient independently from a uniform distribution in $\mathrm{U}[0, 1]$. The corresponding sampled class vectors are shown above each plot, with the dominant glitch label also shown. Both methods produce hybrid morphologies interpolating between glitch classes, demonstrating cDVGAN's ability to explore smooth transitions across the learned latent class space.

Overall, cDVGAN reproduces the structure of the real data distribution with strong overlap between real and synthetic samples. The main exception is the Whistle class, where synthetic samples form a partially separate cluster, indicating that the full morphological diversity of this class is not fully captured. Nevertheless, the generated samples remain close in latent space, showing that cDVGAN learns the dominant morphology of each glitch class.

Generating hybrid glitch morphologies. Similarly to our study in Chapter 7, the conditional nature of cDVGAN allows it to generate class-interpolated samples that blend features across the learned glitch classes. Figure 8.4 illustrates this capability, showing two approaches for sampling the class vector to produce hybrid datasets: simplex and uniform. In the simplex case, the class vector is normalized to sum to one (i.e. sampled from a $k = 7$ simplex), while in the uniform case, each component is independently sampled from $[0, 1]$. As shown, samples from these datasets exhibit mixed morphological characteristics drawn from multiple glitch classes as per the sampled class vector.

As described in the last chapter, the standard vertex class space is a subset of the simplex space, which itself is contained within the uniform space, with each step expanding the potential diversity of generated samples. This hierarchy is reflected in the UMAP projections shown in Figure 8.5, where simplex and uniform samples clearly span the variability of the standard vertex classes, with the uniform dataset covering the broadest region in embedding space, indicating enhanced diversity.

Although we do not apply these hybrid datasets further in this study, as we did with the hybrid samples in Chapter 7, such data could serve several purposes: en-

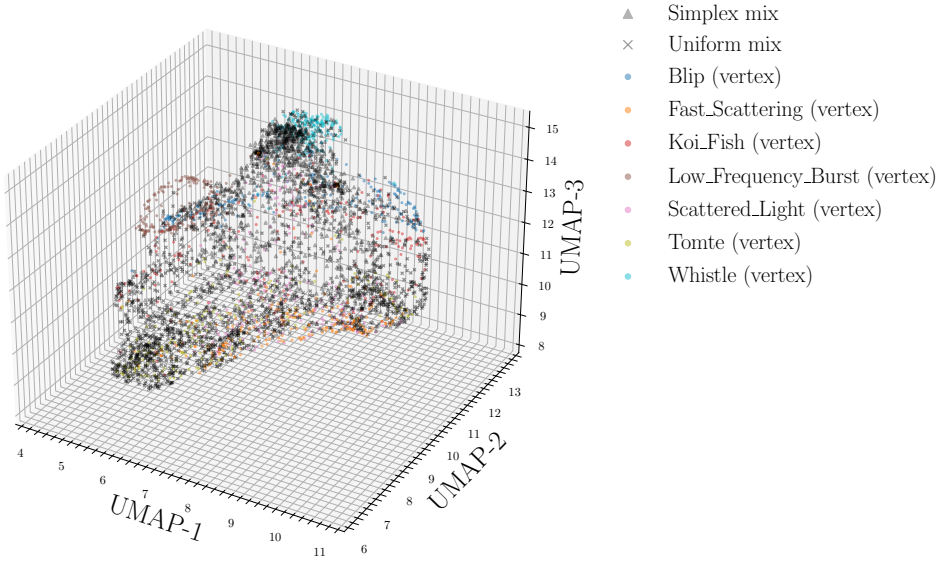


Figure 8.5: Three-dimensional UMAP embedding showing the distribution of class-mixed glitch samples relative to the vertex (pure-class) glitches. Vertex samples are colored by their respective glitch class, while class-mixed samples generated using simplex and uniform class-conditioning vectors are shown in black. The smooth interpolation of mixed samples between class clusters illustrates how cDVGAN captures continuous transitions across glitch morphologies in the latent class space, spanning the diversity of glitch types represented in the training set.

riching MDCs, aiding pipeline validation, or augmenting training datasets for glitch detection and classification algorithms.

Validation via Gravity Spy classification. To quantify realism, we classify generated samples using Gravity Spy, similarly to DeepExtractor's reconstructions in Chapter 4. For each class, 100 generated glitches were injected into quiet Hanford O3 background noise at SNR = 50 and classified using Gravity Spy. The resulting confusion matrix shown in Figure 8.6 reflects classifier predictions and confidence scores in brackets. Because Gravity Spy supports additional classes beyond our seven training classes, the matrix is asymmetric.

The overall classification accuracy is $\sim$ 72%. While this may appear low, the misclassifications are consistent with known limitations of Gravity Spy. Most misclassifications occur in the Koi Fish and Scattered Light classes. While only 20% of the generated Koi Fish samples were correctly classified, the majority of the remaining samples were misidentified as Blip, Blip Low Frequency, or Tomte. This behaviour is consistent with known morphological similarities between Koi Fish and these glitch classes, which may arise from related physical mechanisms differing primarily in SNR [355]. Moreover, increasing the injected SNR of the generated Koi Fish samples to 150 raises the classification accuracy to 79% with an average confidence

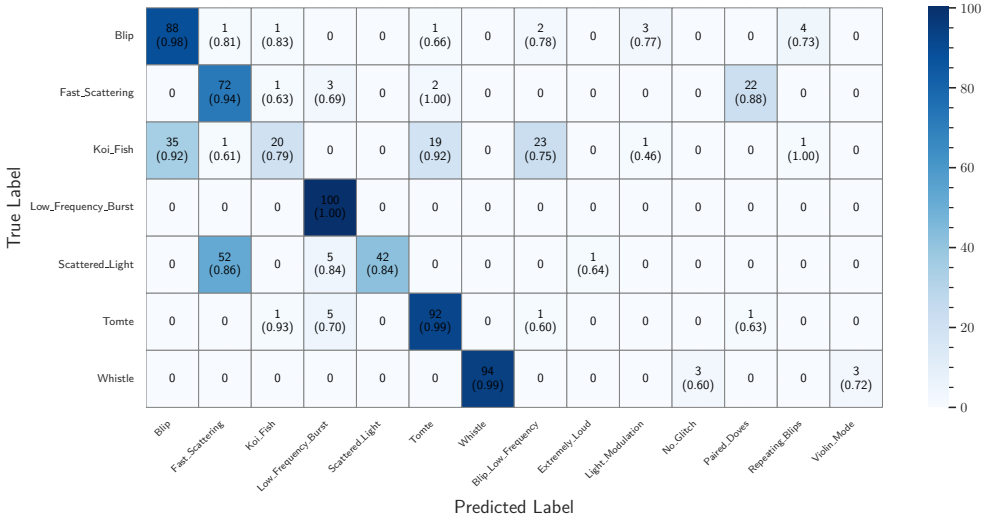


Figure 8.6: Confusion matrix showing Gravity Spy classifications of glitches generated by cDVGAN, with 100 samples per class. Each cell indicates the number of samples predicted for each class, with the average Gravity Spy confidence shown in brackets. The strong diagonal structure demonstrates that most synthetic glitches are correctly classified according to their intended class.

of ∼95%, further illustrating the dependence of Gravity Spy classifications on the injected SNR.

This behaviour is further supported by the analysis in Appendix D.2, where glitches were injected at an SNR of 20. In this lower-SNR regime, none of the generated Koi Fish samples were correctly classified; instead, they were misidentified as other morphologically similar classes. A comparable SNR dependence is observed for the Whistle class. Injecting the generated Whistle glitches at an SNR of 50 yields an accuracy of 94%, whereas injecting them at an SNR of 20 leads to frequent misclassification as No Glitch.

For Scattered Light, Gravity Spy often predicts Fast Scattering, which is expected given their related scattering origins at low frequency. This was also observed for DeepExtractor's original reconstructions of Scattered Light shown in Chapter 4.

Thus, if Blip, Tomte, and Blip Low Frequency are regarded as acceptable misclassifications for the Koi Fish class, and Fast Scattering is considered an acceptable misclassification for Scattered Light, then the effective classification accuracy for this experiment exceeds 90%. It is also important to note that some misclassifications may arise from stochastic interactions between the injected glitches and the specific noise realization into which they are injected, which can differ from the original training conditions. Additionally, labelling uncertainties in the Gravity Spy training set may contribute, and the discarded phase information in Q-scans may play a role—an issue we discuss further below.

Comparison with the DiT generative model. We also evaluate our DiT model. Figures 8.7 and 8.8 show its UMAP and Gravity Spy classification performance respectively. DiT overlaps with real glitches primarily for low-frequency classes (e.g., Fast Scattering, Scattered Light) but shows reduced diversity and separable clusters for most other classes where cDVGAN was more successful.

A notable case is Low Frequency Burst. UMAP reveals two distinct clusters (real vs. synthetic), yet Gravity Spy classifies 87% of generated samples correctly with high confidence (97%). This discrepancy arises because UMAP preserves phase information, whereas Gravity Spy operates on magnitude-only Q-scans. Figure 8.9 illustrates how time-domain differences can be masked in magnitude-only spectrogram representations.

This result highlights a key limitation of using magnitude-based, supervised classifiers like Gravity Spy for glitch characterization, as they may overlook physically meaningful phase information. Unsupervised latent-space analysis complements such classification by revealing differences invisible to Q-scan–based methods.

In summary, while DiT demonstrates progress toward conditional time-domain glitch generation, its ability to reproduce full morphological diversity—especially mid- and high-frequency content—is limited relative to cDVGAN. Future improvements include reintroducing dilated convolutional backbones (e.g., U-Net/WaveNet), alternative normalization strategies, or training class-specific diffusion models without dimensionality reduction via an autoencoder.

8.5 Conclusion

This study demonstrated the capability of cDVGAN to generate realistic glitch datasets that extend beyond idealized proxy data. Within a single, flexible framework, the model reproduced a broad range of glitch morphologies in both the time and frequency domains—consistent with those faithfully reconstructed by DeepExtractor from real LIGO data, as shown in Chapter 4. It can generate specific glitch types based on user-defined conditions and produce hybrid glitches by sampling the conditioned class vector. These hybrid samples interpolate between classes, capturing the morphological variation across the dataset and enabling the creation of diverse glitch datasets for downstream applications such as data challenges, pipeline validation, and data augmentation, as shown in Chapter 7.

We evaluated cDVGAN using two complementary approaches: an unsupervised, data-driven UMAP analysis to visualize and compare the class distributions of real and synthetic glitches in a projected 3D space, and a supervised classification analysis using Gravity Spy, a state-of-the-art glitch classification algorithm trained on an extensive labeled dataset.

The UMAP results showed that cDVGAN-generated glitches generally occupy the same regions as real data in the embedded space, indicating that the model captures much of the morphological diversity within each glitch class. One exception is the Whistle class, which consists of the highest-frequency glitches in the dataset. For this class, synthetic samples formed a partially distinct cluster from the real data, suggesting that cDVGAN did not fully capture the fine-scale, high-frequency structure characteristic of these glitches.

The Gravity Spy classification results broadly aligned with the unsupervised UMAP

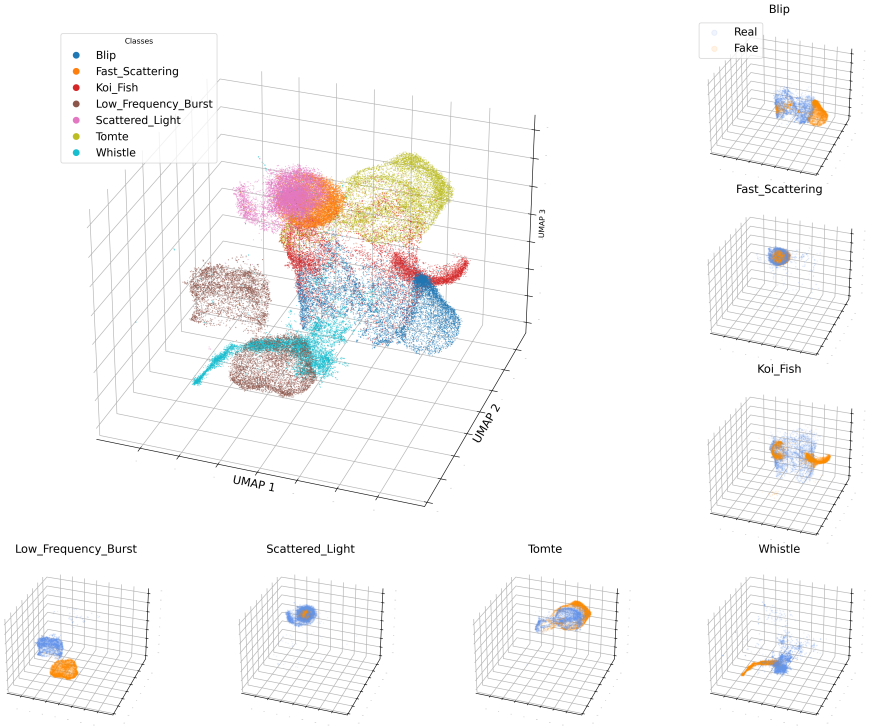


Figure 8.7: Three-dimensional UMAP projection of real (orange) and DiT-generated (blue) data across all glitch classes. The large panel shows the overall structure, while smaller panels display each class individually. Overlapping clusters indicate morphological similarity between real and synthetic samples.

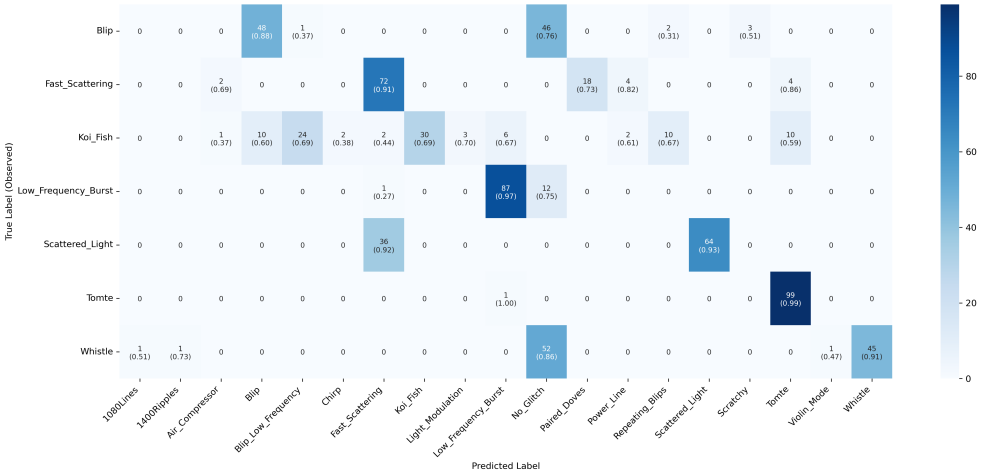


Figure 8.8: Confusion matrix showing Gravity Spy classifications of samples generated by the DiT model. Each entry shows the number of predictions and average Gravity Spy confidence for those predictions in brackets.

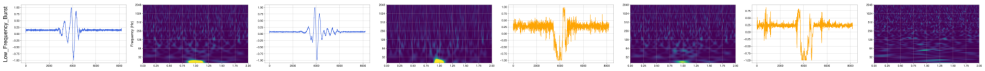


Figure 8.9: Two examples each of real (left) and DiT generated (right) samples from the Low Frequency Burst class, showing both the time-domain and magnitude Q-scan representations.

analysis, with most cDVGAN samples correctly classified. Misclassifications primarily occured for the Koi Fish and Scattered Light classes, which can largely be attributed to morphological similarities between similar classes and Gravity Spy's classification being dependent on SNR. For instance, Scattered Light glitches were often confused with Fast Scattering, while Koi Fish are sometimes misclassified as Blip or Tomte. These overlaps are expected, as the classes share similar time-frequency signatures. This is confirmed by increasing the injected SNR to 150, where 79% of generated Koi Fish samples are correctly classified.

From an architectural standpoint, the large convolutional kernels used in cDVGAN may also limit its ability to resolve fine, high-frequency details—particularly for the Koi Fish and Whistle classes. Future work could address this by incorporating smaller kernels (balanced against the need for long-range temporal modelling), dilated convolutions to better capture hierarchical structure, or attention mechanisms to enhance global feature representation.

Finally, this study highlights a fundamental limitation in using purely magnitude-based Q-scan representations for supervised glitch classification. We observed that even a less robust diffusion-based model, which produces unrealistic Low Frequency Burst glitches in the time domain, can still yield correct Gravity Spy labels with high confidence. Although the time-domain waveforms of real and synthetic samples are clearly distinct, their Q-scan representations appear similar, explaining the classifier's

high confidence. This discrepancy is evident in the corresponding UMAP analysis of time-domain data, which reveals two distinct clusters separating real and synthetic samples. Such findings underscore the limitations of magnitude-only training that discards phase information—an essential feature for accurately capturing glitch morphology. They further highlight the importance of preserving phase information and integrating unsupervised analyses to complement and strengthen glitch characterization efforts.

Part IV

Final Remarks

Conclusions

This thesis has explored deep learning–based methods for improving the analysis of GW data, focusing on two major challenges in current and future detector networks: (1) the model-agnostic reconstruction and mitigation of non-Gaussian transient noise artifacts, known as glitches, and (2) the scalable and realistic generation of simulated data for analysis development and validation.

Through a series of interconnected studies, this work introduced novel deep neural frameworks for the reconstruction and generation of time-domain GW signals and glitches, laying a foundation for more robust, data-driven approaches to GW astronomy in both the current and next-generation eras.

Context and Motivation

The first part of this thesis established the physical and computational context for the research. Chapters 1 and 2 reviewed the theoretical foundations of general relativity and GW emission, the operation of current detectors, and the principles of traditional data analysis methods such as matched filtering and Bayesian inference.

After outlining the foundational concepts, Chapter 3 introduced fundamentals of machine learning, and discussed it as a modern complement to traditional analysis pipelines. It surveyed how deep learning techniques, spanning various architectures, are being used to accelerate detection, improve parameter estimation, and enhance noise mitigation.

The Reconstruction and Separation of Signals and Glitches

The second part of this thesis introduces deep learning frameworks capable of reconstructing arbitrary signals and glitches in GW data, demonstrating that such models can accurately recover diverse transient morphologies. While full holistic glitch mitigation remains a challenging and open problem, the results presented here offer a promising step towards that goal and outline a pathway for future development.

Model-Angostic Signal and Glitch Reconstruction with DeepExtractor

Part II presented DeepExtractor, a novel deep learning framework for reconstructing transient signals and glitches in GW data. As described in Chapter 4, DeepExtractor employs a PSD-informed U-Net architecture operating on magnitude and phase spectrograms, learning to isolate detector noise from input data that may contain both astrophysical signals and glitches. The key innovation lies in its residual formulation and the design of its training data: by learning to isolate the background noise component from diverse combinations of simulated waveforms spanning the relevant time–frequency ranges for both signals and glitches, DeepExtractor learns to

reconstruct virtually any excess power above the detector's noise floor. Rather than predicting simulated the proxy training signals and glitches directly, it predicts the noisy component and learns the structure of the noise itself, leading to superior generalization across a wide variety of morphologies. This residual learning strategy allows DeepExtractor to recover both modelled and unmodelled transients, including glitches, outperforming baseline deep learning architectures such as autoencoders and one-dimensional U-Nets in terms of time-domain mismatch and reconstruction fidelity.

Comparative experiments with the Bayesian inference framework BayesWave demonstrated that DeepExtractor typically achieves better reconstruction accuracy while reducing computation time by over $10,000$ times, enabling real-time processing on standard hardware. Applications to real LIGO data confirmed that DeepExtractor effectively reconstructs realistic glitch morphologies consistent with those identified by Gravity Spy, preserving the statistical properties of detector noise. Dimensionality-reduction analyses using UMAP and t-SNE further verified that reconstructed glitches cluster according to their classes in the feature space, confirming both the fidelity and diversity of the reconstructed glitches. Finally, we demonstrated its capability as a model-agnostic approach for reconstructing real GW events, successfully recovering three observed signals despite never being trained on them.

Signal and Glitch Separation with DeepExtractor

Chapter 5 extended the DeepExtractor framework to address one of the most persistent challenges in GW data analysis; glitch mitigation. To this end, we modified the DeepExtractor framework for the simultaneous reconstruction of the three components of a detector output: the astrophysical signal $s(t)$, glitches $g(t)$, and stationary background noise $n(t)$.

Whereas the initial signal-/glitch-only mode of DeepExtractor focused solely on modelling the stationary noise background for a single detector, the extended framework reconstructs both the signal and noise in each detector. However, similarly to its initial version, the glitch is then obtained as the residual between the input strain and the reconstructed signal-plus-noise output in each detector. In this multi-detector configuration, the residual formulation allows the model to separate incoherent transient noise, which appears in only a single detector, from the coherent astrophysical signal present across the network. We restricted this preliminary demonstration to the two LIGO detectors (Hanford and Livingston) and assume that glitches occur in only one detector at a time—an assumption consistent with all glitch-contaminated events identified so far.

DeepExtractor's signal+glitch mode adopts a U-Net architecture similar to the first version but operates in the time-domain to reduce computational cost, which is important when scaling to multi-detector analyses that processes substantially larger volumes of data. To ensure generalization across the high-dimensional signal parameter space, we train on a set of over a million simulated waveforms spanning the relevant physical parameters. While deep learning models can interpolate within their training distributions and therefore require far fewer templates than traditional matched filtering (which may rely on millions of templates; see Section 2.4.2), a sufficiently rich training set remains essential.

We evaluated DeepExtractor in a controlled study using two simulated GW events, injecting blip glitches directly over the merger to create strong temporal and spectral overlap and therefore a challenging mitigation scenario. Although the DeepExtractor-corrected posteriors do not perfectly match the baseline for all parameters, they show a clear improvement relative to the glitch-contaminated case and recover the underlying signal parameters far more accurately. Combined with its computational efficiency, processing 4 s of data in under 1 s on a CPU, these results demonstrate that deep learning–based glitch mitigation is feasible for low-latency parameter estimation and rapid sky localization in future observing runs.

Conceptually, DeepExtractor can be viewed as a deterministic deep-learning analogue to BayesWave in its signal+glitch separation mode, but with the potential for faster, fully data-driven, and more scalable operation. Results from Chapter 4 suggest that, with the right architecture and training strategy, deep learning approaches like DeepExtractor may ultimately exceed the performance of traditional Bayesian methods. With further refinement and training on real detector data, the framework could progress towards real-time applications, including rapid glitch removal, online searches, and low-latency parameter estimation.

Generative Modelling of Gravitational-Wave Data

The final part of the thesis shifted focus from reconstruction to generation, introducing a family of GANs for simulating realistic GW signals and glitches. This line of work addresses two key needs in GW data analysis: (1) scalable data augmentation for machine learning–based pipelines, and (2) the ability to generate realistic mock data for validation and training of data analysis algorithms.

DVGAN: Derivative-Enhanced Generation

Chapter 6 introduced Derivative GAN (DVGAN), a novel 1D GAN architecture designed to stabilize adversarial training for time-domain signals. Unlike standard GANs that include a single discriminator, DVGAN employs an auxiliary discriminator that operates on the temporal derivatives of both real and generated samples. This derivative-informed feedback provides richer gradient information to the generator, reducing mode collapse and improving training stability without significant computational overhead.

Through ablation studies across multiple differentiable signal datasets, DVGAN demonstrated smoother and more physically realistic waveform generation than its baseline WGAN counterpart. It also generalized effectively to real (filtered) LIGO blip glitch data, highlighting its robustness on non-synthetic samples. Beyond GW physics, the derivative-based dual-discriminator concept offers broader potential for modelling signals in other domains such as medicine and finance. Chapter 7 concluded by outlining extensions to multi-discriminator and multi-representation frameworks, a direction pursued in the subsequent chapter.

cDVGAN: Conditional Generative Modelling

Building on these results, Chapter 7 presented the Conditional Derivative GAN (cDVGAN); a class-conditional extension of DVGAN capable of generating multiple time-

domain classes within a single unified model. Trained on two unmodelled LIGO glitch types (Blip and Tomte) and one modelled class of binary black hole (BBH) signals, cDVGAN learns a joint latent space across all categories, allowing for both class-specific and hybrid generation. By interpolating between conditioning vectors, the model can produce hybrid samples that capture smooth transitions between physical and instrumental transients, effectively synthesizing events beyond the training distribution.

Comprehensive ablation studies compared cDVGAN with baseline conditional WGANs and alternative dual-discriminator architectures. Results showed that the derivative-informed discriminator enhances generative fidelity and improves the usefulness of synthetic data for downstream tasks, such as training CNNs for signal detection in noisy data. Hybrid datasets produced by cDVGAN—particularly those from simplex interpolations—were found to be especially effective for data augmentation, improving CNN detection performance relative to models trained on purely class-specific samples.

Learning the Glitch Space with cDVGAN

The final research chapter applied cDVGAN to realistic glitch datasets derived from LIGO's O3 run, reconstructed using DeepExtractor. Within a single framework, cDVGAN successfully reproduced seven diverse glitch classes, closely matching those of real detector artifacts produced by DeepExtractor. The model could generate user-specified classes or hybrid morphologies that interpolate between known glitch types, providing a powerful tool for creating diverse glitch datasets.

Performance was assessed through unsupervised UMAP analyses and supervised classification using the Gravity Spy classifier. Results showed strong overlap between real and generated samples in embedded space and high classification consistency, confirming that the synthetic glitches captured the key features of the real data. Remaining discrepancies—particularly for high-frequency classes such as Whistle—highlighted the need for improved modelling of fine temporal structure, potentially achievable through smaller convolutional kernels, dilated convolutions, or attention mechanisms. This chapter also revealed the limitations of magnitude-only Q-scan representations for glitch classification, emphasizing the importance of incorporating phase information to fully capture the glitch morphology relevant for robust characterization.

Future Work and Broader Impact

DeepExtractor

DeepExtractor already functions as a fast, model-agnostic method for reconstructing arbitrary excess power in GW data. In its current form, it is well suited for downstream applications such as rapid glitch inspection, detector characterization, and low-latency data-quality assessment. However, there remains many avenues for future development.

DeepExtractor for Virgo

This thesis focused on DeepExtractor for LIGO data, but the method is now being adapted for Virgo, which faces many of the same challenges posed by transient non-Gaussian noise. Although only limited transfer learning has been applied so far—and additional fine-tuning phases are planned—the model has already been used to support glitch investigations during the O4 observing run.

DeepExtractor for the Einstein Telescope

DeepExtractor also has direct applicability as a glitch–removal tool for the triangular configuration of the Einstein Telescope (ET). A key advantage of the ET geometry is the *null stream*: a linear combination of the three interferometer outputs that cancels any true GW signal purely through detector geometry. Any residual excess power in the null stream must therefore originate from incoherent noise, such as glitches.

After transfer learning on representative ET data, DeepExtractor's signal/glitch-only mode could be applied directly to the null stream to reconstruct the incoherent glitch component. The reconstructed glitch could then be subtracted from the affected detector, avoiding the need to model signal, noise, and glitch components simultaneously, as required in DeepExtractor's signal+glitch separation mode in Chapter 5. This represents a significant simplification of the glitch mitigation problem in the ET context.

Integrating DeepExtractor into the null-stream framework would be technically straightforward and could strengthen the scientific case for the ET's triangular design by providing a practical, low-latency method for glitch removal.

DeepExtractor for LISA Data Analysis

With an expected launch in the mid-2030s, the Laser Interferometer Space Antenna (LISA) [307] will open an entirely new observational window on the GW universe. Unlike current ground-based detectors, LISA will operate in a regime where signals

are continuously present in the data, requiring fundamentally different analysis strategies. Noise characterization will be especially critical, as disentangling overlapping astrophysical sources from instrumental artifacts becomes significantly more complex.

Experience from the LISA Pathfinder mission [382] revealed the presence of glitch transients even in a reduced configuration of the instrument. Motivated by these findings, we are currently collaborating with Eurpean Space Agency (ESA) researchers to explore how DeepExtractor can be adapted to identify and mitigate glitches in the LISA data stream. In parallel, we are investigating its potential use for reconstructing and characterizing specific classes of expected LISA signals.

Beyond glitch mitigation, DeepExtractor may also contribute to search or triggering strategies by providing rapid, model-agnostic transient reconstructions. Similar applications are possible for current ground-based detectors, discussed in the next section.

DeepExtractor as a model-agnostic search pipeline

Another promising future direction is the use of DeepExtractor as the core of a model-agnostic GW search pipeline, an approach we are already investigating. Because DeepExtractor can operate in real time and can reconstruct signals of arbitrary morphology, it could be used to build a search statistic without relying on the construction of computationally expensive template banks, as required in matched-filter pipelines.

In this configuration, DeepExtractor would run continuously on the data streams from multiple detectors. When a GW signal is present and coherent across the network, the reconstructions in each detector would also be coherent (up to time of arrival and antenna patterns) and could be used to compute an SNR-like search statistic, analogous to Equation 2.35 (Section 2.4), with the reconstruction replacing the template h. This statistic could be computed either directly in the time domain or after transformation to the frequency domain.

As with standard searches, glitches would also generate triggers; however, a simple veto could be constructed using the reconstructions across the detector network. Triggers produced by incoherent transients could be suppressed by weighting the search statistic by the overlap (match M) of the reconstructions across detectors. Glitches, being incoherent, would have no overlap, causing their weighted statistic to collapse, whereas true astrophysical signals would retain a non-zero match across the network[2].

Once validated on real data, this approach could enable a fast, online search pipeline capable of detecting both modelled and unmodelled signals. Because DeepExtractor is agnostic to waveform morphology, such a pipeline could supplement or complement existing algorithms such as cWB, particularly for poorly modelled sources like core-collapse supernovae.

Next Steps for DeepExtractor

Several promising directions can further develop DeepExtractor into a more robust reconstruction framework for GW data analysis.

[2]The match between reconstructed signals from different detectors will in general depend on the source's sky location, the detectors' antenna patterns, and the relative SNRs. This limitation applies to all coherent search pipelines, not only to the approach proposed here.

Incorporating probabilistic outputs and uncertainty estimates. A key next step is transitioning from deterministic predictions to probabilistic reconstructions, analogous to the posterior distributions produced by BayesWave. By allowing DeepExtractor to output not only a reconstruction but also an associated uncertainty estimate—e.g., via variational techniques, Bayesian neural networks, or deep ensembles—the model could quantify confidence. Such uncertainty information would (i) increase interpretability and user trust, (ii) allow glitch uncertainty to propagate into downstream analyses such as parameter estimation, and (iii) enable principled thresholding based on reconstruction confidence rather manually chosen thresholds (e.g. based on SNR).

Iterative self-improvement through reconstruction-driven training. Another promising direction is iterative training, in which the model improves using its own reconstructions. In this thesis, DeepExtractor was trained solely on linear combinations of proxy glitches and waveforms, yet it demonstrated strong generalization to realistic glitches. An iterative training loop—where reconstructed glitches are reinjected into noise and added to the training set—could progressively refine the reconstructions of real glitch morphologies. Such a self-supervised refinement scheme would enable DeepExtractor to learn from true detector artifacts without requiring manual labelling or curated glitch catalogs, and may ultimately converge towards increasingly accurate, morphology-agnostic reconstructions.

Additional future directions. Beyond these core improvements, several extensions are worth pursuing:

- **Calibration and nonstationarity robustness:** training across varying noise PSDs and calibration conditions would increase reliability during detector commissioning and early observing runs.
- **Multi-resolution and multi-scale architectures:** hybrid approaches that combine time- and frequency-domain inputs may improve the capture of both fine and broadband structure.
- **Integration into real-time systems:** deploying DeepExtractor within data-quality monitoring dashboards or continuous low-latency pipelines.

Together, these directions outline a clear pathway for transforming DeepExtractor from a fast, accurate reconstruction tool into a probabilistic, iterative, and fully autonomous deep-learning component of future GW analyses.

Applications in Other Domains

The residual learning strategy at the core of DeepExtractor—reconstructing the noise component rather than predicting the signal directly—enables generalization to signal and noise patterns that were not present in the training set. This makes the approach relevant far beyond GW data analysis.

Because DeepExtractor operates on spectrograms (or more broadly, two-dimensional time–frequency representations), it can be applied to domains where transient or poorly modelled features must be isolated from noise. Examples include:

- **Particle physics**, where denoising or isolating anomalous detector activity could aid particle-identification pipelines
- **Industrial monitoring and fault detection**, where unexpected transient patterns in sensor spectrograms indicate equipment failure;
- **Seismology and geophysics**, where separating coherent seismic waves from transient anthropogenic noise (e.g., explosions, machinery) is analogous to separating signals and glitches
- **Audio signal processing**, such as speech enhancement or separating transient artifacts in acoustic recordings.

More generally, the architecture is suitable for any domain requiring fast reconstruction of localized excess power from noisy data, especially where the underlying phenomena are not easily modelled. Its ability to perform real-time reconstruction and to interpolate across previously unseen morphologies makes DeepExtractor a strong candidate for anomaly detection, rapid diagnostics, and data-quality monitoring across the physical sciences and applied engineering.

DeepExtractor's Signal+Glitch Mode

DeepExtractor's signal + glitch mode represents an early but promising attempt at holistic separation of GW signals, stationary detector noise, and incoherent transient glitches. The results in this thesis demonstrate that such separation is feasible using deep learning, even in challenging cases where the glitch overlaps the merger. However, significant development is required before the model can achieve the reliability and stability needed for deployment in active observing runs.

Evaluation and Benchmarking

This initial study was intentionally limited to two simulated events and a single glitch morphology (blip), injected directly over the merger to create strong temporal and spectral overlap. A broader evaluation is needed to assess robustness across the full BBH parameter space and across realistic glitch diversity. Future work will include parameter sweeps over mass, distance, sky position, and orientation, together with injections from the full Gravity Spy glitch taxonomy. In addition, the current analysis compares DeepExtractor only to the baseline (clean) data. A direct 1:1 comparison with BayesWave will be essential to quantify relative reconstruction fidelity and parameter accuracy.

Dataset and Training Improvements

Improving the realism and coverage of the training dataset is a critical next step. The current dataset oversamples very distant mergers, leading to many low-SNR examples that contribute little to learning signal–glitch separation. Future versions of DeepExtractor will train on a more representative population of signals, providing denser and more uniform sampling of the CBC parameter space. The injected glitch population will also be reshaped to better reflect real detector statistics—distribution

of SNR, duration, and frequency structure. Incorporating realistic transients generated by DeepExtractor will expose the network to the true variability of instrument behavior, improving generalization beyond idealized proxy glitches.

Architecture Enhancements

Integrating attention mechanisms or temporal sequence modules (e.g., Transformers or LSTMs) may allow the network to reason jointly across detectors, improving separation when a glitch affects only one instrument. Hybrid loss functions that combine waveform similarity metrics with time-domain reconstruction losses may provide more stable optimization across a broad SNR range, particularly for signals with extreme mass ratios or weak amplitudes. Multi-resolution and dilated convolutions will be explored to simultaneously capture short-duration glitch bursts and broadband inspiral–merger–ringdown morphology.

Towards Physics Realism and Deployment

A critical milestone is extending training to full multi-detector operation. Future development will include Virgo and KAGRA, enabling the network to exploit the redundancy and coherence afforded by three or more detectors. Incorporating variations in the noise power spectral density and calibration uncertainty will further increase realism and prepare DeepExtractor for deployment on real data. Ultimately, the framework is intended to serve as a pre-processing module for low-latency searches and parameter estimation pipelines, reducing the impact of glitches on sky localization and enabling more reliable electromagnetic follow-up.

The overarching vision is a robust, real-time glitch mitigation system—a deep-learning analogue to BayesWave, but one that operates orders of magnitude faster and integrates seamlessly into future multi-detector GW analyses.

cDVGAN

Towards a Holistic Glitch Generator

Realistic glitch generators are increasingly valuable to the collaboration, as they can supplement MDCs, pipeline validation efforts, and commissioning studies. cDVGAN demonstrated strong capability in learning a diverse space of glitch morphologies and generating synthetic samples that overlap with real glitches in latent space. Although the model was trained on only seven classes, these classes span a significant fraction of the glitch population currently observed in LIGO and Virgo and are already suitable for use in simulation pipelines.

A natural next milestone is to extend cDVGAN to cover the full set of Gravity Spy classes, enabling analysts to generate any glitch on demand. Such a “holistic glitch generator” would allow researchers to stress-test search pipelines with any glitch type. Additionally, because the model supports interpolation within its conditioned class space, it can generate hybrid morphologies, bridging between discrete glitch types, and therefore produce extremely diverse datasets for downstream applications such as training robust glitch classifiers or anomaly-detection models.

Next Steps for cDVGAN

To better capture the full morphological diversity of real glitches, several architectural and training improvements are planned:

Capturing multi-scale structure. Glitches span a wide range of timescales and frequencies, from short, high-frequency bursts to long-duration, low-frequency scattering events. Potential enhancements include:

- Reducing kernel sizes or introducing **dilated convolutions** to better resolve fine time–frequency structure.
- Incorporating **attention mechanisms** to emphasize transient temporal features [383].
- Using **multi-resolution pathways** to jointly model short- and long-duration glitches.

Care must be taken to balance receptive-field size: overly small kernels may improve fine detail but at the cost of degrading low-frequency morphologies.

Conditioning on physical glitch features. A further extension is to condition glitch generation directly on measurable physical quantities rather than human-assigned glitch labels. Tools such as Omicron routinely extract features including SNR, central frequency, bandwidth, and duration for each transient. Conditioning cDVGAN on these continuous parameters would enable controlled generation of glitches with specific, user-defined physical characteristics—e.g., "generate a transient with SNR 25, central frequency 120 Hz, bandwidth 40 Hz." This removes reliance on subjective class labels, better reflects the physical parameter space explored by commissioning teams, and allows analysts to probe pipeline behavior across precise regions of glitch morphology. In the long term, such parameterized control could support systematic stress testing of search pipelines, uncertainty studies for glitch–signal overlap scenarios, and targeted data augmentation for machine-learning–based glitch classifiers.

Richer supervision via auxiliary discriminators. The dual-discriminator framework in cDVGAN is not limited to time-domain or derivative representations. Additional discriminators can be attached to alternate representations of the data, such as spectrograms or Q-transforms. This approach was explored in the subsequent *cDVGAN2* prototype and can be expanded further—enabling the generator to learn complementary structure from multiple views of the same glitch. For image-based applications, auxiliary discriminators receiving different resolutions or feature scales could help balance global structure and local detail, similar to multi-scale GAN architectures.
In the longer term, a fully conditioned generative model that spans the entire glitch taxonomy could serve as a community resource for MDC construction, commissioning studies, and machine-learning–based pipeline development, allowing analysts to generate realistic, diverse, and controllable glitch populations on demand.

Applications in Other Domains

Although cDVGAN was developed for GW glitch generation, its core ideas—the conditional generation of diverse transient morphologies and the use of auxiliary discriminators to guide fidelity—extend naturally to many other fields where labelled data are scarce, sensitive, or costly to obtain.

High-fidelity time–series generation is particularly valuable in domains such as:

- **Medicine**, where synthetic ECG/EEG waveforms can support algorithm development without requiring access to patient data.
- **Finance**,where realistic but privacy-preserving market simulations are needed for stress testing and forecasting.
- **Audio and Speech**, including music synthesis, where fine temporal detail must be preserved.

More broadly, cDVGAN's class-conditioning and interpolation properties make it suitable for any domain where practitioners need controllable synthetic data that span a diverse space of morphologies. In applications ranging from medical diagnostics to anomaly detection and industrial monitoring, such generative tools can supplement limited datasets, enable stress-testing of algorithms, and accelerate the development of data-driven pipelines without relying exclusively on sensitive or difficult-to-acquire real data.

Bibliography

[1] B. P. Abbott et al. (LIGO Scientific Collaboration and 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**, 10.1103/PhysRevX.9.031040 (2019).

[2] R. Abbott et al. (LIGO Scientific Collaboration and 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**, 10.1103/PhysRevX.11.021053 (2021).

[3] R. Abbott et al. (LIGO Scientific Collaboration, Virgo Collaboration, and KAGRA Collaboration), "Gwtc-3: compact binary coalescences observed by ligo and virgo during the second part of the third observing run", Phys. Rev. X **13**, 10.1103/PhysRevX.13.041039 (2023).

[4] B. P. Abbott et al. (LIGO Scientific, Virgo), "A First Targeted Search for Gravitational-Wave Bursts from Core-Collapse Supernovae in Data of First-Generation Laser Interferometer Detectors", Phys. Rev. D **94**, 10.1103/PhysRevD.94.102001 (2016).

[5] R. Abbott et al. (LIGO Scientific, VIRGO, KAGRA), "Search for Gravitational Waves Associated with Gamma-Ray Bursts Detected by Fermi and Swift during the LIGO–Virgo Run O3b", Astrophys. J. **928**, 10.3847/1538-4357/ac532b (2022).

[6] R. Abbott et al. (LIGO Scientific, Virgo, KAGRA), "Constraints on Cosmic Strings Using Data from the Third Advanced LIGO–Virgo Observing Run", Phys. Rev. Lett. **126**, 10.1103/PhysRevLett.126.241102 (2021).

[7] P. D. Lasky, "Gravitational Waves from Neutron Stars: A Review", Publ. Astron. Soc. Austral. **32**, 10.1017/pasa.2015.35 (2015).

[8] N. Christensen, "Stochastic Gravitational Wave Backgrounds", Rept. Prog. Phys. **82**, 10.1088/1361-6633/aae6b5 (2019).

[9] R. Abbott et al. (KAGRA, Virgo, LIGO Scientific), "Upper limits on the isotropic gravitational-wave background from Advanced LIGO and Advanced Virgo's third observing run", Phys. Rev. D **104**, 10.1103/PhysRevD.104.022004 (2021).

[10] J. Antoniadis et al. (EPTA), "The second data release from the European Pulsar Timing Array III. Search for gravitational wave signals", Astron. Astrophys. **678**, 10.1051/0004-6361/202346844 (2023).

[11] B. P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), "Observation of gravitational waves from a binary black hole merger", Phys. Rev. Lett. **116**, 10.1103/PhysRevLett.116.061102 (2016).

[12] LIGO Scientific Collaboration and others, "Advanced LIGO", Class. Quant. Grav. **32**, 074001, 10.1088/0264-9381/32/7/074001 (2015).

[13] F. Acernese et al. (VIRGO), "Advanced Virgo: a second-generation interferometric gravitational wave detector", Class. Quant. Grav. **32**, 10.1088/0264-9381/32/2/024001 (2015).

[14] T. Akutsu, M. Ando, K. Arai, Y. Arai, S. Araki, A. Araya, N. Aritomi, H. Asada, Y. Aso, and S. Atsuta, "Kagra: 2.5 generation interferometric gravitational wave detector", Nature Astronomy **3**, 10.1038/s41550-018-0658-y (2019).

[15] B. S. Sathyaprakash and B. F. Schutz, "Physics, astrophysics and cosmology with gravitational waves", Living Reviews in Relativity **12**, 10.12942/lrr-2009-2 (2009).

[16] B. F. Schutz, "Networks of gravitational wave detectors and three figures of merit", Class. Quant. Grav. **28**, 10.1088/0264-9381/28/12/125023 (2011).

[17] C. Fabry and A. Pérot, "Théorie et applications d'une nouvelle méthode de spectroscopie interférentielle", Annales de Chimie et de Physique **16** (1899).

[18] M. Maggiore, *Gravitational Waves. Vol. 1: Theory and Experiments* (Oxford University Press, 2007).

[19] B. P. Abbott et al. (LIGO Scientific Collaboration), "Calibration of the advanced ligo detectors for the discovery of the binary black-hole merger gw150914", Phys. Rev. D **95**, 10.1103/PhysRevD.95.062003 (2017).

[20] C. Cahillane et al., "Calibration uncertainty for advanced ligo's first and second observing runs", Phys. Rev. D **96**, 10.1103/PhysRevD.96.102001 (2017).

[21] L. Sun et al., "Characterization of systematic error in advanced ligo calibration", Classical and Quantum Gravity **37**, 10.1088/1361-6382/abb14e (2020).

[22] M. Wade, A. D. Viets, T. Chmiel, M. Stover, and L. Wade, "Improving LIGO calibration accuracy by using time-dependent filters to compensate for temporal variations", Class. Quant. Grav. **40**, 10.1088/1361-6382/acabf6 (2023).

[23] A. Freise and K. Strain, "Interferometer techniques for gravitational-wave detection", Living Reviews in Relativity **13**, 10.12942/lrr-2010-1 (2010).

[24] B. P. Abbott et al., "A guide to ligo–virgo detector noise and extraction of transient gravitational-wave signals", Classical and Quantum Gravity **37**, 10.1088/1361-6382/ab685e (2020).

[25] L. K. Nuttall, T. J. Massinger, J. Areeda, J. Betzwieser, S. Dwyer, A. Effler, R. P. Fisher, P. Fritschel, J. S. Kissel, A. P. Lundgren, D. M. Macleod, D. Martynov, J. McIver, A. Mullavey, D. Sigg, J. R. Smith, G. Vajente, A. R. Williamson, and C. C. Wipf, "Improving the data quality of advanced ligo based on early engineering run results", Classical and Quantum Gravity **32**, 10.1088/0264-9381/32/24/245005 (2015).

[26] D. Davis et al., "Ligo detector characterization in the second and third observing runs", Classical and Quantum Gravity **38**, 10.1088/1361-6382/abfd85 (2021).

[27] F. Acernese et al. (Virgo), "Virgo detector characterization and data quality: results from the O3 run", Class. Quant. Grav. **40**, 10.1088/1361-6382/acd92d (2023).

[28] B. P. Abbott et al., "Sensitivity of the Advanced LIGO detectors at the beginning of gravitational wave astronomy", Phys. Rev. D **93**, [Addendum: Phys.Rev.D 97, 059901 (2018)], 10.1103/PhysRevD.93.112004 (2016).

[29] T. T. Lyons, M. W. Regehr, and F. J. Raab, "Shot noise in gravitational-wave detectors with fabry–perot arms", Appl. Opt. **39**, 10.1364/AO.39.006761 (2000).

[30] P. B. Covas et al. (LSC Instrument Authors), "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 **97**, 10.1103/PhysRevD.97.082002 (2018).

[31] J. R. Smith et al., "A Hierarchical method for vetoing noise transients in gravitational-wave detectors", Class. Quant. Grav. **28**, 10.1088/0264-9381/28/23/235005 (2011).

[32] L. Blackburn et al., "The LSC Glitch Group: Monitoring Noise Transients during the fifth LIGO Science Run", Class. Quant. Grav. **25**, edited by S. Hughes and E. Katsavounidis, 10.1088/0264-9381/25/18/184004 (2008).

[33] B. P. Abbott et al. (LIGO Scientific, Virgo), "Characterization of transient noise in Advanced LIGO relevant to gravitational wave signal GW150914", Class. Quant. Grav. **33**, 10.1088/0264-9381/33/13/134001 (2016).

[34] S. Soni et al. (LIGO), "Reducing scattered light in LIGO's third observing run", Class. Quant. Grav. **38**, 10.1088/1361-6382/abc906 (2020).

[35] M. Cabero et al., "Blip glitches in Advanced LIGO data", Class. Quant. Grav. **36**, 10.1088/1361-6382/ab2e14 (2019).

[36] C. Pankow et al., "Mitigation of the instrumental noise transient in gravitational-wave data surrounding gw170817", Physical Review D **98** (2018).

[37] J. C. Driggers et al. (LIGO Scientific), "Improving astrophysical parameter estimation via offline noise subtraction for Advanced LIGO", Phys. Rev. D **99**, 10.1103/PhysRevD.99.042001 (2019).

[38] E. Payne, S. Hourihane, J. Golomb, R. Udall, D. Davis, and K. Chatziioannou, "Curious case of gw200129: interplay between spin-precession inference and data-quality issues", Phys. Rev. D **106**, 10.1103/PhysRevD.106.104017 (2022).

[39] A. Gupta et al., *Possible causes of false general relativity violations in gravitational wave observations*, 2024.

[40] F. Robinet, N. Arnaud, N. Leroy, A. Lundgren, D. Macleod, and J. McIver, "Omicron: a tool to characterize transient noise in gravitational-wave detectors", SoftwareX **12**, https://doi.org/10.1016/j.softx.2020.100620 (2020).

[41] M. Zevin et al., "Gravity Spy: Integrating Advanced LIGO Detector Characterization, Machine Learning, and Citizen Science", Class. Quant. Grav. **34**, 10.1088/1361-6382/aa5cea (2017).

[42] J. C. Brown, "Calculation of a constant q spectral transform", The Journal of the Acoustical Society of America **89**, 10.1121/1.400476 (1991).

[43] J. Glanzer et al., "Data quality up to the third observing run of advanced LIGO: Gravity Spy glitch classifications", Class. Quant. Grav. **40**, 10.1088/1361-6382/acb633 (2023).

[44] M. López, "Exploring the frontier of transient gravitational wave detection", An optional note, PhD thesis (Universiteit Utrecht, x, July 2025).

[45] S. Soni et al., "Discovering features in gravitational-wave data through detector characterization, citizen science and machine learning", Class. Quant. Grav. **38**, 10.1088/1361-6382/ac1ccb (2021).

[46] S. Bahaadini, V. Noroozi, N. Rohani, S. Coughlin, M. Zevin, J. Smith, V. Kalogera, and A. Katsaggelos, "Machine learning for gravity spy: glitch classification and dataset", Information Sciences **444**, https://doi.org/10.1016/j.ins.2018.02.068 (2018).

[47] A. Torres-Forné, E. Cuoco, J. A. Font, and A. Marquina, "Application of dictionary learning to denoise ligo's blip noise transients", Phys. Rev. D **102**, 10.1103/PhysRevD.102.023011 (2020).

[48] J. D. Merritt, B. Farr, R. Hur, B. Edelman, and Z. Doctor, "Transient glitch mitigation in advanced ligo data", Phys. Rev. D **104**, 10.1103/PhysRevD.104.102004 (2021).

[49] A. E. Tolley, G. S. C. Davies, I. W. Harry, and A. P. Lundgren, "Archenemy: removing scattered-light glitches from gravitational wave data", Classical and Quantum Gravity **40**, 10.1088/1361-6382/ace22f (2023).

[50] N. J. Cornish and T. B. Littenberg, "Bayeswave: bayesian inference for gravitational wave bursts and instrument glitches", Classical and Quantum Gravity **32**, 10.1088/0264-9381/32/13/135012 (2015).

[51] K. Chatziioannou, N. Cornish, M. Wijngaarden, and T. B. Littenberg, "Modeling compact binary signals and instrumental glitches in gravitational wave data", Phys. Rev. D **103**, 10.1103/PhysRevD.103.044013 (2021).

[52] S. Hourihane, K. Chatziioannou, M. Wijngaarden, D. Davis, T. Littenberg, and N. Cornish, "Accurate modeling and mitigation of overlapping signals and glitches in gravitational-wave data", Physical Review D **106**, 10.1103/physrevd.106.042006 (2022).

[53] B. Allen, W. Hua, and A. Ottewill, *Automatic cross-talk removal from multi-channel data*, 1999.

[54] D. Davis et al., "Improving the sensitivity of advanced ligo using noise subtraction", Class. Quant. Grav. **36** (2019).

[55] D. Davis, T. B. Littenberg, I. M. Romero-Shaw, M. Millhouse, J. McIver, F. Di Renzo, and G. Ashton, "Subtracting glitches from gravitational-wave detector data during the third ligo-virgo observing run", Classical and Quantum Gravity **39**, 10.1088/1361-6382/aca238 (2022).

[56] S. Khan et al., "Frequency-domain gravitational waves from nonprecessing black-hole binaries. ii. a phenomenological model for the advanced detector era", Phys. Rev. D **93**, 10.1103/PhysRevD.93.044007 (2016).

[57] T. B. Littenberg and N. J. Cornish, "Bayesian inference for spectral estimation of gravitational wave detector noise", Phys. Rev. D **91**, 10.1103/PhysRevD.91.084034 (2015).

[58] R. Abbott et al. (KAGRA, LIGO Scientific, VIRGO), "All-sky search for continuous gravitational waves from isolated neutron stars using Advanced LIGO and Advanced Virgo O3 data", Phys. Rev. D **106**, 10.1103/PhysRevD.106.102008 (2022).

[59] B. Steltner et al., "Deep Einstein@Home All-sky Search for Continuous Gravitational Waves in LIGO O3 Public Data", Astrophys. J. **952**, 10.3847/1538-4357/acdad4 (2023).

[60] R. Abbott et al., "Narrowband searches for continuous and long-duration transient gravitational waves from known pulsars in the ligo-virgo third observing run", The Astrophysical Journal **932**, 10.3847/1538-4357/ac6ad0 (2022).

[61] R. Abbott et al. (LIGO Scientific Collaboration, Virgo Collaboration, and KAGRA Collaboration), "Search for continuous gravitational wave emission from the milky way center in o3 ligo-virgo data", Phys. Rev. D **106**, 10.1103/PhysRevD.106.042003 (2022).

[62] B. P. Abbott et al. (LIGO Scientific, Virgo), "Optically targeted search for gravitational waves emitted by core-collapse supernovae during the first and second observing runs of advanced LIGO and advanced Virgo", Phys. Rev. D **101**, 10.1103/PhysRevD.101.084002 (2020).

[63] R. Abbott et al., "Searches for continuous gravitational waves from young supernova remnants in the early third observing run of advanced ligo and virgo", The Astrophysical Journal **921**, 10.3847/1538-4357/ac17ea (2021).

[64] R. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), "Search of the early o3 ligo data for continuous gravitational waves from the cassiopeia a and vela jr. supernova remnants", Phys. Rev. D **105**, 10.1103/PhysRevD.105.082005 (2022).

[65] B. P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), "Upper limits on the stochastic gravitational-wave background from advanced ligo's first observing run", Phys. Rev. Lett. **118**, 10.1103/PhysRevLett.118.121101 (2017).

[66] B. Abbott et al., "Search for the isotropic stochastic background using data from advanced ligo's second observing run", Physical Review D **100**, 10.1103/physrevd.100.061101 (2019).

[67] R. Abbott et al., "Search for anisotropic gravitational-wave backgrounds using data from advanced ligo and advanced virgo's first three observing runs", Physical Review D **104**, 10.1103/physrevd.104.022005 (2021).

[68] A. Nagar et al., "Multipolar effective one body waveform model for spin-aligned black hole binaries", Phys. Rev. D **102**, 10.1103/PhysRevD.102.024077 (2020).

[69] S. Ossokine et al., "Multipolar effective-one-body waveforms for precessing binary black holes: construction and validation", Phys. Rev. D **102**, 10.1103/PhysRevD.102.044055 (2020).

[70] A. Nagar, A. Bonino, and P. Rettegno, "Effective one-body multipolar waveform model for spin-aligned, quasicircular, eccentric, hyperbolic black hole binaries", Phys. Rev. D **103**, 10.1103/PhysRevD.103.104021 (2021).

[71] A. Ramos-Buades et al., "Next generation of accurate and efficient multipolar precessing-spin effective-one-body waveforms for binary black holes", Phys. Rev. D **108**, 10.1103/PhysRevD.108.124037 (2023).

[72] A. Bohé 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**, 10.1103/PhysRevD.95.044028 (2017).

[73] R. Cotesta et al., "Enriching the symphony of gravitational waves from binary black holes by tuning higher harmonics", Phys. Rev. D **98**, 10.1103/PhysRevD.98.084028 (2018).

[74] H. Estellés et al., "Phenomenological time domain model for dominant quadrupole gravitational wave signal of coalescing binary black holes", Phys. Rev. D **103**, 10.1103/PhysRevD.103.124060 (2021).

[75] H. Estellés et al., "New twists in compact binary waveform modeling: a fast time-domain model for precession", Phys. Rev. D **105**, 10.1103/PhysRevD.105.084040 (2022).

[76] G. Pratten et al., "Computationally efficient models for the dominant and subdominant harmonic modes of precessing binary black holes", Phys. Rev. D **103**, 10.1103/PhysRevD.103.104056 (2021).

[77] B. Allen et al., "FINDCHIRP: An Algorithm for detection of gravitational waves from inspiraling compact binaries", Phys. Rev. D **85**, 10.1103/PhysRevD.85.122006 (2012).

[78] M. Drago, V. Gayathri, S. Klimenko, C. Lazzaro, E. Milotti, G. Mitselmakher, V. Necula, B. O'Brian, G. A. Prodi, F. Salemi, M. Szczepanczyk, S. Tiwari, V. Tiwari, G. Vedovato, and I. Yakushin, *Coherent waveburst, a pipeline for unmodeled gravitational-wave data analysis*, 2020.

[79] S. Klimenko, I. Yakushin, A. Mercer, and G. Mitselmakher, "A coherent method for detection of gravitational wave bursts", Classical and Quantum Gravity **25**, 10.1088/0264-9381/25/11/114029 (2008).

[80] S. Klimenko et al., "Method for detection and reconstruction of gravitational wave transients with networks of advanced detectors", Phys. Rev. D **93**, 10.1103/PhysRevD.93.042004 (2016).

[81] J. Veitch et al., "Parameter estimation for compact binaries with ground-based gravitational-wave observations using the LALInference software library", Phys. Rev. D **91**, 10.1103/PhysRevD.91.042003 (2015).

[82] I. M. Romero-Shaw et al., "Bayesian inference for compact binary coalescences with bilby: validation and application to the first LIGO–Virgo gravitational-wave transient catalogue", Mon. Not. Roy. Astron. Soc. **499**, 10.1093/mnras/staa2850 (2020).

[83] G. Ashton, M. Huebner, P. D. Lasky, C. Talbot, K. Ackley, S. Biscoveanu, Q. Chu, A. Divarkala, P. J. Easter, B. Goncharov, F. H. Vivanco, J. Harms, M. E. Lower, G. D. Meadors, D. Melchor, E. Payne, M. D. Pitkin, J. Powell, N. Sarin, R. J. E. Smith, and E. Thrane, "Bilby: A user-friendly Bayesian inference library for gravitational-wave astronomy", The Astrophysical Journal Supplement Series **241**, 10.3847/1538-4365/ab06fc (2018).

[84] R. Biswas, P. R. Brady, J. D. E. Creighton, and S. Fairhurst, "The Loudest event statistic: General formulation, properties and applications", Class. Quant. Grav. **26**, [Erratum: Class.Quant.Grav. 30, 079502 (2013)], 10.1088/0264-9381/26/17/175009 (2009).

[85] W. M. Farr, J. R. Gair, I. Mandel, and C. Cutler, "Counting and confusion: bayesian rate estimation with multiple populations", Phys. Rev. D **91**, 10.1103/PhysRevD.91.023005 (2015).

[86] M. Fishbach, D. E. Holz, and W. M. Farr, "Does the black hole merger rate evolve with redshift?", The Astrophysical Journal Letters **863**, 10.3847/2041-8213/aad800 (2018).

[87] M. Fishbach, R. Essick, and D. E. Holz, "Does matter matter? using the mass distribution to distinguish neutron stars and black holes", The Astrophysical Journal Letters **899**, 10.3847/2041-8213/aba7b6 (2020).

[88] A. Farah, M. Fishbach, R. Essick, D. E. Holz, and S. Galaudage, "Bridging the gap: categorizing gravitational-wave events at the transition between neutron stars and black holes", The Astrophysical Journal **931**, 10.3847/1538-4357/ac5f03 (2022).

[89] I. Mandel, W. M. Farr, A. Colonna, S. Stevenson, P. Tino, and J. Veitch, "Model-independent inference on compact-binary observations", Monthly Notices of the Royal Astronomical Society **465**, 10.1093/mnras/stw2883 (2016).

[90] V. Tiwari, "VAMANA: modeling binary black hole population with minimal assumptions", Classical and Quantum Gravity **38**, 10.1088/1361-6382/ac0b54 (2021).

[91] S. Rinaldi and W. Del Pozzo, "(H)DPGMM: a hierarchy of dirichlet process gaussian mixture models for the inference of the black hole mass function", Monthly Notices of the Royal Astronomical Society **509**, 10.1093/mnras/stab3224 (2021).

[92] T. A. Callister and W. M. Farr, "Parameter-free tour of the binary black hole population", Phys. Rev. X **14**, 10.1103/PhysRevX.14.021005 (2024).

[93] A. Ray, I. M. Hernandez, S. Mohite, J. Creighton, and S. Kapadia, "Nonparametric inference of the population of compact binaries from gravitational-wave observations using binned gaussian processes", The Astrophysical Journal **957**, 10.3847/1538-4357/acf452 (2023).

[94] A. Ghosh, N. K. Johnson-McDaniel, A. Ghosh, C. K. Mishra, P. Ajith, W. D. Pozzo, C. P. L. Berry, A. B. Nielsen, and L. London, "Testing general relativity using gravitational wave signals from the inspiral, merger and ringdown of binary black holes", Classical and Quantum Gravity **35**, 10.1088/1361-6382/aa972e (2017).

[95] R. Cotesta, G. Carullo, E. Berti, and V. Cardoso, "Analysis of ringdown overtones in gw150914", Phys. Rev. Lett. **129**, 10.1103/PhysRevLett.129.111102 (2022).

[96] B. P. Abbott et al. (LIGO Scientific and Virgo Collaborations), "Tests of general relativity with gw150914", Phys. Rev. Lett. **116**, 10.1103/PhysRevLett.116.221101 (2016).

[97] B. P. Abbott et al. (LIGO Scientific, Virgo), "Tests of General Relativity with GW170817", Phys. Rev. Lett. **123**, 10.1103/PhysRevLett.123.011102 (2019).

[98] T. E. Collett and D. Bacon, "Testing the speed of gravitational waves over cosmological distances with strong gravitational lensing", Physical Review Letters **118**, 10.1103/PhysRevLett.118.091101 (2016).

[99] R. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), "Tests of general relativity with binary black holes from the second ligo-virgo gravitational-wave transient catalog", Phys. Rev. D **103**, 10.1103/PhysRevD.103.122002 (2021).

[100] J. A. Faber and F. A. Rasio, "Binary neutron star mergers", Living Reviews in Relativity **15** (2012).

[101] K. Chatziioannou, "Neutron star tidal deformability and equation of state constraints", Gen. Rel. Grav. **52**, 10.1007/s10714-020-02754-3 (2020).

[102] E. NAKAR, "Short-hard gamma-ray bursts", Physics Reports **442**, 10.1016/j.physrep.2007.02.005 (2007).

[103] F. Piron, "Gamma-ray bursts at high and very high energies", Comptes Rendus. Physique **17**, 10.1016/j.crhy.2016.04.005 (2016).

[104] B. D. Metzger, "Kilonovae", Living Reviews in Relativity **23**, 10.1007/s41114-019-0024-0 (2019).

[105] B. P. Abbott et al., "Properties of the Binary Neutron Star Merger GW170817", Physical Review X **9**, 011001, 10.1103/PhysRevX.9.011001 (2019).

[106] B. P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), "Gw170817: observation of gravitational waves from a binary neutron star inspiral", Phys. Rev. Lett. **119**, 10.1103/PhysRevLett.119.161101 (2017).

[107] B. P. Abbott et al. (LIGO Scientific, Virgo, Fermi-GBM, INTEGRAL), "Gravitational Waves and Gamma-rays from a Binary Neutron Star Merger: GW170817 and GRB 170817A", Astrophys. J. Lett. **848**, 10.3847/2041-8213/aa920c (2017).

[108] B. P. Abbott et al., "Estimating the contribution of dynamical ejecta in the kilonova associated with gw170817", The Astrophysical Journal Letters **850**, 10.3847/2041-8213/aa9478 (2017).

[109] B. P. Abbott et al. (LIGO Scientific, Virgo), "GW170817: Measurements of neutron star radii and equation of state", Phys. Rev. Lett. **121**, 10.1103/PhysRevLett.121.161101 (2018).

[110] P. T. H. Pang et al., "An updated nuclear-physics and multi-messenger astrophysics framework for binary neutron star mergers", Nature Communications **14**, 10.1038/s41467-023-43932-6 (2023).

[111] B. P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), "Tests of general relativity with gw170817", Phys. Rev. Lett. **123**, 10.1103/PhysRevLett.123.011102 (2019).

[112] A. Albert et al., "Search for high-energy neutrinos from binary neutron star merger gw170817 with antares, icecube, and the pierre auger observatory", The Astrophysical Journal Letters **850**, 10.3847/2041-8213/aa9aed (2017).

[113] B. P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), "Constraining the p-mode–g-mode tidal instability with GW170817", Phys. Rev. Lett. **122**, 10.1103/PhysRevLett.122.061104 (2019).

[114] B. P. Abbott et al., "Model comparison from ligo-virgo data on gw170817's binary components and consequences for the merger remnant", Classical and Quantum Gravity **37**, 10.1088/1361-6382/ab5f7c (2020).

[115] B. P. Abbott et al., "A gravitational-wave standard siren measurement of the Hubble constant", Nature **551**, 10.1038/nature24471 (2017).

[116] S. Vitale and H.-Y. Chen, "Measuring the hubble constant with neutron star black hole mergers", Physical Review Letters **121**, 10.1103/physrevlett.121.021303 (2018).

[117] B. P. Abbott et al. (LIGO Scientific, Virgo, VIRGO), "A Gravitational-wave Measurement of the Hubble Constant Following the Second Observing Run of Advanced LIGO and Virgo", Astrophys. J. **909**, 10.3847/1538-4357/abdcb7 (2021).

[118] R. Abbott et al. (LIGO Scientific, VIRGO, KAGRA), "Constraints on the cosmic expansion history from GWTC-3", (2021).

[119] T. L. S. Collaboration, the Virgo Collaboration, and the KAGRA Collaboration, *Gwtc-4.0: constraints on the cosmic expansion rate and modified gravitational-wave propagation*, 2025.

[120] K. Hotokezaka et al., "A Hubble constant measurement from superluminal motion of the jet in GW170817", Nature Astron. **3**, 10.1038/s41550-019-0820-1 (2019).

[121] M. Soares-Santos et al. (DES, LIGO Scientific, Virgo), "First Measurement of the Hubble Constant from a Dark Standard Siren using the Dark Energy Survey Galaxies and the LIGO-Virgo Binary Black-hole Merger GW170814", Astrophys. J. Lett. **876**, 10.3847/2041-8213/ab14f1 (2019).

[122] J.-P. Hu and F.-Y. Wang, *Hubble tension: the evidence of new physics*, 2023.

[123] N. Aghanim et al., "Planck 2018 results vi. cosmological parameters", Astronomy & Astrophysics **641** (2020).

[124] A. G. Riess et al., "A Comprehensive Measurement of the Local Value of the Hubble Constant with 1 km s^{-1} Mpc^{-1} Uncertainty from the Hubble Space Telescope and the SH0ES Team", The Astrophysical Journal Letters **934**, 10.3847/2041-8213/ac5c5b (2022).

[125] D. C. T. M. C. Abbott et al., "The Dark Energy Survey: Cosmology Results with approximately 1500 New High-redshift Type Ia Supernovae Using the Full 5 yr Data Set", The Astrophysical Journal Letters **973**, 10.3847/2041-8213/ad6f9f (2024).

[126] C. Chatfield, *The analysis of time series: an introduction*, 6th (CRC Press, Florida, US, 2004).

[127] P. Gregory, "Multivariate gaussian from maximum entropy", in *Bayesian logical data analysis for the physical sciences* (Cambridge University Press, 2005).

[128] G. Vajente, Y. Huang, M. Isi, J. C. Driggers, J. S. Kissel, M. J. Szczepanczyk, and S. Vitale, "Machine-learning nonstationary noise out of gravitational-wave detectors", Phys. Rev. D **101**, 10.1103/PhysRevD.101.042003 (2020).

[129] B. Zackay, T. Venumadhav, J. Roulet, L. Dai, and M. Zaldarriaga, "Detecting gravitational waves in data with non-stationary and non-gaussian noise", Phys. Rev. D **104**, 10.1103/PhysRevD.104.063034 (2021).

[130] O. Edy, A. Lundgren, and L. K. Nuttall, "Issues of mismodeling gravitational-wave data for parameter estimation", Phys. Rev. D **103**, 10.1103/PhysRevD.103.124061 (2021).

[131] S. Kumar, A. H. Nitz, and X. J. Forteza, *Parameter estimation with non stationary noise in gravitational waves data*, Feb. 2022.

[132] O. Edy, "Non-stationarity in gravitational-wave analysis", PhD thesis (University of Portsmouth, Portsmouth, UK, Sept. 2022).

[133] J. P. Burg, "Maximum entropy spectral analysis", Stanford Exploration Project, PhD thesis (Stanford University, Stanford, CA, 1975).

[134] P. Welch, "The use of fast fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms", IEEE Transactions on Audio and Electroacoustics **15**, 10.1109/TAU.1967.1161901 (1967).

[135] J. Neyman and E. S. Pearson, "On the problem of the most efficient tests of statistical hypotheses", Philosophical Transactions of the Royal Society of London. Series A **231**, 10.1098/rsta.1933.0009 (1933).

[136] L. Blanchet et al., "Gravitational-radiation damping of compact binary systems to second post-newtonian order", Phys. Rev. Lett. **74**, 10.1103/PhysRevLett.74.3515 (1995).

[137] A. Buonanno and T. Damour, "Effective one-body approach to general relativistic two-body dynamics", Phys. Rev. D **59**, 10.1103/PhysRevD.59.084006 (1999).

[138] V. Cardoso, L. Gualtieri, C. Herdeiro, and U. Sperhake, "Exploring New Physics Frontiers Through Numerical Relativity", Living Rev. Relativity **18**, 10.1007/lrr-2015-1 (2015).

[139] C. M. Will and A. G. Wiseman, "Gravitational radiation from compact binary systems: gravitational waveforms and energy loss to second post-newtonian order", Phys. Rev. D **54**, 10.1103/PhysRevD.54.4813 (1996).

[140] M. E. Pati and C. M. Will, "Post-newtonian gravitational radiation and equations of motion via direct integration of the relaxed einstein equations: foundations", Phys. Rev. D **62**, 10.1103/PhysRevD.62.124015 (2000).

[141] M. E. Pati and C. M. Will, "Post-newtonian gravitational radiation and equations of motion via direct integration of the relaxed einstein equations. ii. two-body equations of motion to second post-newtonian order, and radiation reaction to 3.5 post-newtonian order", Phys. Rev. D **65**, 10.1103/PhysRevD.65.104008 (2002).

[142] S. Khan et al., "Frequency-domain gravitational waves from nonprecessing black-hole binaries. II. A phenomenological model for the advanced detector era", Phys. Rev. D **93**, 10.1103/PhysRevD.93.044007 (2016).

[143] S. Husa et al., "Frequency-domain gravitational waves from nonprecessing black-hole binaries. I. New numerical waveforms and anatomy of the signal", Phys. Rev. D **93**, 10.1103/PhysRevD.93.044006 (2016).

[144] G. Pratten et al., "Computationally efficient models for the dominant and subdominant harmonic modes of precessing binary black holes", Phys. Rev. D **103**, 10.1103/PhysRevD.103.104056 (2021).

[145] I. Harry, J. Calderón Bustillo, and A. H. Nitz, "Searching for the full symphony of black hole binary mergers", Phys. Rev. D **97**, 10.1103/PhysRevD.97.023004 (2018).

[146] S. Schmidt, "Searching for precessing black hole binaries in gravitational-wave data", English, Doctoral thesis 1 (Research UU / Graduation UU) (Universiteit Utrecht, May 2025).

[147] K. Cannon et al., "Toward early-warning detection of compact binary coalescence", Astrophys. J. **748**, 10.1088/0004-637X/748/2/136 (2012).

[148] B. J. Owen and B. S. Sathyaprakash, "Matched filtering of gravitational waves from inspiraling compact binaries: computational cost and template placement", Phys. Rev. D **60**, 10.1103/PhysRevD.60.022002 (1999).

[149] R. Prix, "Template-based searches for gravitational waves: efficient lattice covering of flat parameter spaces", Classical and Quantum Gravity **24**, 10.1088/0264-9381/24/19/S18 (2007).

[150] B. J. Owen, "Search templates for gravitational waves from inspiraling binaries: choice of template spacing", Phys. Rev. D **53**, 10.1103/PhysRevD.53.6749 (1996).

[151] S. Babak, "Building a stochastic template bank for detecting massive black hole binaries", Classical and Quantum Gravity **25**, 10.1088/0264-9381/25/19/195011 (2008).

[152] S. Schmidt, B. Gadre, and S. Caudill, "Gravitational-wave template banks for novel compact binaries", Phys. Rev. D **109**, 10.1103/PhysRevD.109.042005 (2024).

[153] S. A. Usman et al., "The pycbc search for gravitational waves from compact binary coalescence", Class. Quantum Grav. **33**, 10.1088/0264-9381/33/21/215004 (2016).

[154] C. Messick et al., "Analysis framework for the prompt discovery of compact binary mergers in gravitational-wave data", Phys. Rev. D **95**, 10.1103/PhysRevD.95.042001 (2017).

[155] B. Allen, "A χ^2 time-frequency discriminator for gravitational wave detection", Phys. Rev. D **71**, 10.1103/PhysRevD.71.062001 (2005).

[156] R. Lynch et al., "Information-theoretic approach to the gravitational-wave burst detection problem", Phys. Rev. D **95**, 10.1103/PhysRevD.95.104046 (2017).

[157] M. Drago et al., "Coherent WaveBurst, a pipeline for unmodeled gravitational-wave data analysis", arXiv pre-print (2020).

[158] S. Chatterji, L. Blackburn, G. Martin, and E. Katsavounidis, "Multiresolution techniques for the detection of gravitational-wave bursts", Classical and Quantum Gravity **21**, 10.1088/0264-9381/21/20/024 (2004).

[159] S. Mallat, *A wavelet tour of signal processing*, 2nd (Academic Press, 1999).

[160] A. Graps, ""an introduction to wavelets"", IEEE Comp. Sci. Engi. **2**, 10.1109/99.388960 (1995).

[161] V. Necula, S. Klimenko, and G. Mitselmakher, "Transient analysis with fast wilson-daubechies time-frequency transform", Journal of Physics: Conference Series **363**, 10.1088/1742-6596/363/1/012032 (2012).

[162] M. Drago, "Search for transient gravitational wave signals with unknown waveform in the ligo-virgo network of interferometric detectors using a fully coherent algorithm", PhD thesis (University of Padua. Dept. of Physics., 2010).

[163] V. Cardoso, E. Franzin, and P. Pani, "Is the gravitational-wave ringdown a probe of the event horizon?", Physical Review Letters **116**, 10.1103/physrevlett.116.171101 (2016).

[164] V. Cardoso, S. Hopper, C. F. B. Macedo, C. Palenzuela, and P. Pani, "Gravitational-wave signatures of exotic compact objects and of quantum corrections at the horizon scale", Physical Review D **94**, 10.1103/physrevd.94.084031 (2016).

[165] P. Whittle, "The analysis of multiple stationary time series", Journal of the Royal Statistical Society: Series B (Methodological) **15** (1953).

[166] J. Skilling, "Nested sampling for general Bayesian computation", Bayesian Analysis **1**, 10.1214/06-BA127 (2006).

[167] J. Veitch and A. Vecchio, "Bayesian coherent analysis of in-spiral gravitational wave signals with a detector network", Phys. Rev. D **81**, 10.1103/PhysRevD.81.062003 (2010).

[168] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", Nature **521** (2015).

[169] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need", in Advances in neural information processing systems, Vol. 30 (2017).

[170] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun, "Dermatologist-level classification of skin cancer with deep neural networks", Nature **542** (2017).

[171] J. B. Heaton, N. G. Polson, and J. H. Witte, "Deep learning in finance", arXiv preprint arXiv:1602.06561 (2017).

[172] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová, "Machine learning and the physical sciences", Reviews of Modern Physics **91** (2019).

[173] M. Punturo et al., "The einstein telescope: a third-generation gravitational wave observatory", Classical and Quantum Gravity **27**, 10.1088/0264-9381/27/19/194002 (2010).

[174] D. Reitze et al., "Cosmic Explorer: The U.S. Contribution to Gravitational-Wave Astronomy beyond LIGO", in, Vol. 51 (Sept. 2019).

[175] F. Iacovelli et al., "Forecasting the detection capabilities of third-generation gravitational-wave detectors using GWFAST", The Astrophysical Journal **941**, 10.3847/1538-4357/ac9cd4 (2022).

[176] V. Kalogera et al., "The next generation global gravitational wave observatory: the science book", in (2021).

[177] C. Fryer and K. New, "Gravitational waves from gravitational collapse", Living Reviews in Relativity **14**, 10.12942/lrr-2011-1 (2011).

[178] L. Baiotti, I. Hawke, L. Rezzolla, and E. Schnetter, "Details on the gravitational-wave emission from rotating gravitational collapse in 3d", in (2007).

[179] T. Akutsu et al., "Overview of KAGRA: Calibration, detector characterization, physical environmental monitors, and the geophysics interferometer", Progress of Theoretical and Experimental Physics **2021**, 10.1093/ptep/ptab018 (2021).

[180] B. P. Abbott et al., "Prospects for observing and localizing gravitational-wave transients with advanced ligo, advanced virgo and kagra", Living Reviews in Relativity **23**, 10.1007/s41114-020-00026-9 (2020).

[181] S. H. et al., "Sensitivity studies for third-generation gravitational wave observatories", Classical and Quantum Gravity **28**, 10.1088/0264-9381/28/9/094013 (2011).

[182] H. Gabbard, M. Williams, F. Hayes, and C. Messenger, "Matching matched filtering with deep networks for gravitational-wave astronomy", Phys. Rev. Lett. **120**, 10.1103/PhysRevLett.120.141103 (2018).

[183] T. D. Gebhard, N. Kilbertus, I. Harry, and B. Schölkopf, "Convolutional neural networks: a magic bullet for gravitational-wave detection?", Phys. Rev. D **100**, 10.1103/PhysRevD.100.063015 (2019).

[184] P. G. Krastev, "Real-time detection of gravitational waves from binary neutron stars using artificial neural networks", Physics Letters B **803**, 10.1016/j.physletb.2020.135330 (2020).

[185] E. Cuoco et al., "Enhancing gravitational-wave science with machine learning", Machine Learning: Science and Technology **2**, 10.1088/2632-2153/abb93a (2020).

[186] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning* (MIT Press, 2016).

[187] C. M. Bishop, *Pattern recognition and machine learning* (Springer, 2006).

[188] K. P. Murphy, *Machine learning: a probabilistic perspective* (MIT Press, 2012).

[189] R. Sutton and A. Barto, "Reinforcement learning: an introduction", IEEE Transactions on Neural Networks **9**, 10.1109/TNN.1998.712192 (1998).

[190] T. T. Cai and R. Ma, *Theoretical foundations of t-sne for visualizing high-dimensional clustered data*, 2022.

[191] L. McInnes, J. Healy, and J. Melville, *Umap: uniform manifold approximation and projection for dimension reduction*, 2020.

[192] J. Schmidhuber, "Deep learning in neural networks: an overview", Neural Networks **61** (2015).

[193] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain", Psychological Review **65** (1958).

[194] K. Hornik, "Multilayer feedforward networks are universal approximators", Neural Networks **2** (1989).

[195] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines", in Proceedings of the 27th international conference on machine learning (icml) (2010).

[196] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models", in Proc. icml (2013).

[197] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)", in Iclr (2016).

[198] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)", arXiv preprint arXiv:1606.08415 (2016).

[199] S. Hochreiter and J. Schmidhuber, "Long short-term memory", Neural Computation **9** (1997).

[200] D. Kingma and J. Ba, "Adam: a method for stochastic optimization", in Proceedings of the 3rd international conference on learning representations (iclr 2015) (2015).

[201] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors", Nature **323** (1986).

[202] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting", Journal of Machine Learning Research **15** (2014).

[203] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition", Proceedings of the IEEE **86** (1998).

[204] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation", arXiv preprint arXiv:1406.1078 (2014).

[205] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift", in Proceedings of the 32nd international conference on machine learning (icml) (2015).

[206] H. I. Fawaz et al., "Deep learning for time series classification: a review", Data Mining and Knowledge Discovery **33**, 10.1007/s10618-019-00619-1 (2019).

[207] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning", 10.48550/arXiv.1603.07285 (2016).

[208] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks", Science **313**, 10.1126/science.1127647 (2006).

[209] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders", in (Jan. 2008).

[210] D. George and E. A. Huerta, "Deep learning for real-time gravitational wave detection and parameter estimation: results with advanced ligo data", Physics Letters B **778**, 10.1016/j.physletb.2017.11.061 (2018).

[211] W. Wei and E. Huerta, "Gravitational wave denoising of binary black hole mergers with deep learning", Physics Letters B **800**, 10.1016/j.physletb.2019.135081 (2020).

[212] O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation", in Medical image computing and computer-assisted intervention (miccai), Vol. 9351 (2015).

[213] D. Stoller, S. Ewert, and S. Dixon, *Wave-u-net: a multi-scale neural network for end-to-end audio source separation*, 2018.

[214] S. A. Nossier, J. Wall, M. Moniri, C. Glackin, and N. Cannings, "An experimental analysis of deep learning architectures for supervised speech enhancement", Electronics **10**, 10.3390/electronics10010017 (2021).

[215] A. Pandey and D. Wang, "A new framework for cnn-based speech enhancement in the time domain", IEEE/ACM Trans. Audio, Speech and Lang. Proc. **27**, 10.1109/TASLP.2019.2913512 (2019).

[216] K. Zhang, Y. Li, J. Liang, J. Cao, Y. Zhang, H. Tang, R. Timofte, and L. V. Gool, "Practical blind image denoising via swin-conv-unet and data synthesis", Machine Intelligence Research **20** (2022).

[217] G. Frusque and O. Fink, "Robust time series denoising with learnable wavelet packet transform", Advanced Engineering Informatics **62** (2024).

[218] D. P. Kingma and M. Welling, *Auto-encoding variational bayes*, 2013.

[219] D. J. Rezende and S. Mohamed, *Variational inference with normalizing flows*, 2015.

[220] I. J. Goodfellow et al., *Generative adversarial networks*, 2014.

[221] J. Ho, A. Jain, and P. Abbeel, *Denoising diffusion probabilistic models*, 2020.

[222] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli, *Deep unsupervised learning using nonequilibrium thermodynamics*, 2015.

[223] J. Ho and T. Salimans, *Classifier-free diffusion guidance*, 2022.

[224] S. Dash, A. Yale, I. Guyon, and K. Bennett, "Medical time-series data generation using generative adversarial networks", in (Sept. 2020).

[225] S. Ji, J. Luo, and X. Yang, *A comprehensive survey on deep music generation: multi-level representations, algorithms, evaluations, and future directions*, 2020.

[226] J. F. Nash, "Non-cooperative games", Annals of Mathematics **54** (1951).

[227] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks", in Proceedings of the 34th international conference on machine learning, Vol. 70, edited by D. Precup and Y. W. Teh, Proceedings of Machine Learning Research (Aug. 2017).

[228] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, *Improved training of wasserstein gans*, 2017.

[229] M. Mirza and S. Osindero, *Conditional generative adversarial nets*, 2014.

[230] H. de Vries, F. Strub, S. Chandar, O. Pietquin, H. Larochelle, and A. C. Courville, "Modulating early visual processing by language", in Advances in neural information processing systems (neurips) (2017).

[231] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans", in Proceedings of the 34th international conference on machine learning (icml) (2017).

[232] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville, "Film: visual reasoning with a general conditioning layer", in Proceedings of the aaai conference on artificial intelligence, Vol. 32, 1 (2018).

[233] A. Brock, J. Donahue, and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis", in International conference on learning representations (iclr) (2019).

[234] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He, "Attngan: fine-grained text to image generation with attentional generative adversarial networks", in Proceedings of the ieee conference on computer vision and pattern recognition (cvpr) (2018).

[235] T. Miyato and M. Koyama, "CGANs with projection discriminator", in International conference on learning representations (2018).

[236] T. Zhao et al., "Dawning of a new era in gravitational wave data analysis: unveiling cosmic mysteries via artificial intelligence – a systematic review", arXiv preprint arXiv:2311.06571 (2023).

[237] R. Ormiston and tohers, "Noise Reduction in Gravitational-wave Data via Deep Learning", Phys. Rev. Res. **2**, 10.1103/PhysRevResearch.2.033066 (2020).

[238] R. Essick, P. Godwin, C. Hanna, L. Blackburn, and E. Katsavounidis, "Idq: statistical inference of non-gaussian noise with auxiliary degrees of freedom in gravitational-wave detectors", Machine Learning: Science and Technology **2**, 10.1088/2632-2153/abab5f (2020).

[239] J. Glanzer and et al, *Data quality up to the third observing run of advanced ligo: gravity spy glitch classifications*, 2022.

[240] D. Davis, L. V. White, and P. R. Saulson, "Utilizing aligo glitch classifications to validate gravitational-wave candidates", Classical and Quantum Gravity **37** (2020).

[241] M. Zevin, S. Coughlin, S. Bahaadini, E. Besler, N. Rohani, S. Allen, M. Cabero, K. Crowston, A. K. Katsaggelos, S. L. Larson, T. K. Lee, C. Lintott, T. B. Littenberg, A. Lundgren, C. Østerlund, J. R. Smith, L. Trouille, and V. Kalogera, "Gravity spy: integrating advanced ligo detector characterization, machine learning, and citizen science", Classical and Quantum Gravity **34**, 10.1088/1361-6382/aa5cea (2017).

[242] J. Powell et al., "Classification methods for noise transients in advanced gravitational-wave detectors II: performance tests on Advanced LIGO data", Classical and Quantum Gravity **34**, 034002, 10.1088/1361-6382/34/3/034002 (2017).

[243] M. Razzano and E. Cuoco, "Image-based deep learning for classification of noise transients in gravitational wave detectors", Classical and Quantum Gravity **35**, 10.1088/1361-6382/aab793 (2018).

[244] M. Razzano et al., "Gwitchhunters: machine learning and citizen science to improve the performance of gravitational wave detector", Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **1048**, https://doi.org/10.1016/j.nima.2022.167959 (2023).

[245] Y. Li, Y. Wu, and A. K. Katsaggelos, "Cross-temporal spectrogram autoencoder (ctsae): unsupervised dimensionality reduction for clustering gravitational wave glitches", 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2024).

[246] P. Laguarta et al., "Detection of anomalies amongst LIGO's glitch populations with autoencoders", Class. Quant. Grav. **41**, 10.1088/1361-6382/ad1f26 (2024).

[247] J. Theiler, "Estimating fractal dimension", J. Opt. Soc. Am. A **7**, 10.1364/JOSAA.7.001055 (1990).

[248] M. Cavaglià, "Characterization of gravitational-wave detector noise with fractals", Classical and Quantum Gravity **39**, 10.1088/1361-6382/ac7325 (2022).

[249] H. Shen, D. George, E. A. Huerta, and Z. Zhao, "Denoising gravitational waves with enhanced deep recurrent denoising auto-encoders", in Icassp 2019-2019 ieee international conference on acoustics, speech and signal processing (icassp) (IEEE, 2019).

[250] C. Chatterjee and K. Jani, "Navigating unknowns: deep learning robustness for gravitational-wave signal reconstruction", The Astrophysical Journal **973**, 10.3847/1538-4357/ad6984 (2024).

[251] C. Chatterjee and K. Jani, *No glitch in the matrix: robust reconstruction of gravitational wave signals under noise artifacts*, 2024.

[252] C. Murali and D. Lumley, "Detecting and denoising gravitational wave signals from binary black holes using deep learning", Phys. Rev. D **108**, 10.1103/PhysRevD.108.043024 (2023).

[253] Z. Weiqiang, S. Mousavi, and G. Beroza, "Seismic signal denoising and decomposition using deep neural networks", IEEE Transactions on Geoscience and Remote Sensing **PP**, 10.1109/TGRS.2019.2926772 (2019).

[254] V. Boudart and M. Fays, "Albus: a machine learning algorithm for gravitational wave burst searches", in 2022 ieee international conference on big data (big data) (2022).

[255] K. Mogushi, R. Quitzow-James, M. Cavaglià, S. Kulkarni, and F. Hayes, "Nnetfix: an artificial neural network-based denoising engine for gravitational-wave signals", Machine Learning: Science and Technology **2**, 10.1088/2632-2153/abea69 (2021).

[256] E. A. Huerta, P. Kumar, M. Giesler, D. George, A. J. K. Chua, R. Haas, S. Habib, D. E. Johnson, M. L. Katz, A. H. Mroue, B. O'Brien, K. Panchapakesan, and et al., "Eccentric, nonspinning, inspiral, gaussian-process merger approximant for the detection and characterization of eccentric binary black hole mergers", Physical Review D **97**, 10.1103/PhysRevD.97.024031 (2018).

[257] B. D. Lackey, S. Bernuzzi, C. R. Galley, J. Meidam, and C. Van Den Broeck, "Effective-one-body waveforms for binary neutron stars using surrogate models", Physical Review D **95**, 10.1103/PhysRevD.95.104036 (2017).

[258] B. D. Lackey, M. Pürrer, and A. Taracchini, "Surrogate model for an aligned-spin effective one body waveform model of binary neutron star inspirals using gaussian process regression", Physical Review D **100**, 10.1103/PhysRevD.100.024002 (2019).

[259] S. Khan and R. Green, "Gravitational-wave surrogate models powered by artificial neural networks", Physical Review D **103**, 10.1103/PhysRevD.103.064015 (2021).

[260] L. M. Thomas, G. Pratten, and P. Schmidt, "Accelerating multimodal gravitational waveforms from precessing compact binaries with artificial neural networks", arXiv preprint arXiv:2205.05636 (2022).

[261] S.-C. Fragkouli, N. Chatzifotis, K. Chatziioannou, N. J. Cornish, J. R. Gair, C. J. Moore, and A. Vecchio, "Deep residual error and bag-of-tricks learning for gravitational wave surrogate modeling", arXiv preprint arXiv:2203.14981 (2022).

[262] B. Keith, A. Khadse, and S. E. Field, "Learning orbital dynamics of binary black hole systems from gravitational wave measurements", Physical Review Research **3**, 10.1103/PhysRevResearch.3.043101 (2021).

[263] J. Tissino, M. Pürrer, T. Dietrich, B. D. Lackey, and J. R. Gair, "Combining effective-one-body accuracy and reduced-order-quadrature speed for binary neutron star merger parameter estimation with machine learning", Physical Review D **107**, 10.1103/PhysRevD.107.084037 (2023).

[264] S. Schmidt, H. Witek, J. Monaco, A. J. K. Chua, M. Giesler, S. E. Field, R. Haas, A. H. Mroué, H. P. Pfeiffer, and F. Pretorius, "Machine learning gravitational waves from binary black hole mergers", Physical Review D **103**, 10.1103/PhysRevD.103.043020 (2021).

[265] A. J. K. Chua, C. R. Galley, and M. Vallisneri, "Reduced-order modeling with artificial neurons for gravitational-wave inference", Physical Review Letters **122**, 10.1103/PhysRevLett.122.211101 (2019).

[266] J. Lee, W. Kim, H. Kim, and H. M. Lee, "Deep learning model on gravitational waveforms in merging and ringdown phases of binary black hole coalescences", Physical Review D **103**, 10.1103/PhysRevD.103.123023 (2021).

[267] A. J. K. Chua, M. L. Katz, N. Warburton, and S. A. Hughes, "Rapid generation of fully relativistic extreme-mass-ratio-inspiral waveform templates for LISA data analysis", Physical Review Letters **126**, 10.1103/PhysRevLett.126.051102 (2021).

[268] P. Nousi, S.-C. Fragkouli, N. Passalis, P. Iosif, T. Apostolatos, G. Pappas, N. Stergioulas, and A. Tefas, "Autoencoder-driven spiral representation learning for gravitational wave surrogate modelling", Neurocomputing **491**, https://doi.org/10.1016/j.neucom.2022.03.052 (2022).

[269] A. Khan, E. A. Huerta, and H. Zheng, "Interpretable AI forecasting for numerical relativity waveforms of quasicircular, spinning, nonprecessing binary black hole mergers", Physical Review D **105**, 10.1103/PhysRevD.105.024024 (2022).

[270] S. Schmidt, B. Gadre, and S. Caudill, "Gravitational-wave template banks for novel compact binaries", Phys. Rev. D **109**, 10.1103/PhysRevD.109.042005 (2024).

[271] D. George and E. A. Huerta, "Deep neural networks to enable real-time multimessenger astrophysics", Physical Review D **97**, 10.1103/PhysRevD.97.044039 (2018).

[272] H. Gabbard, M. Williams, F. Hayes, and C. Messenger, "Matching matched filtering with deep networks for gravitational-wave astronomy", Physical Review Letters **120**, 10.1103/PhysRevLett.120.141103 (2018).

[273] K. Sharma, K. Chandra, and A. Pai, "Fishing massive black hole binaries with THAMES", arXiv preprint arXiv:2208.02299 (2022).

[274] P. G. Krastev, "Real-time detection of gravitational waves from binary neutron stars using artificial neural networks", Physics Letters B **803**, 10.1016/j.physletb.2020.135330 (2020).

[275] A. Menéndez-Vázquez, A. Torres-Forné, J. A. Font, O. Pujol, et al., "Searches for compact binary coalescence events using neural networks in the ligo/virgo second observation period", Physical Review D **103**, 10.1103/PhysRevD.103.062004 (2021).

[276] G. Baltus, H. Daub, C. A. Da Silva, A. Lemaître, A. Lundgren, et al., "Convolutional neural networks for the detection of the early inspiral of a gravitational-wave signal", Physical Review D **103**, 10.1103/PhysRevD.103.102003 (2021).

[277] G. Baltus, H. Daub, C. A. Da Silva, A. Lemaître, A. Lundgren, et al., "Convolutional neural network for gravitational-wave early alert: going down in frequency", Physical Review D **106**, 10.1103/PhysRevD.106.042002 (2022).

[278] J. Yan et al., "Generalized approach to matched filtering using neural networks", Phys. Rev. D **105**, 10.1103/PhysRevD.105.043006 (2022).

[279] T. Mishra, A. K. Singh, S. Mitra, and S. Bose, "Optimization of model independent gravitational wave search for binary black hole mergers using machine learning", Physical Review D **104**, 10.1103/PhysRevD.104.023014 (2021).

[280] M. J. Szczepańczyk, S. Klimenko, V. Tiwari, G. Vedovato, M. Drago, et al., "Search for gravitational-wave bursts in the third advanced ligo-virgo run with coherent waveburst enhanced by machine learning", Physical Review D **107**, 10.1103/PhysRevD.107.062002 (2023).

[281] D. Lopez, M. Drago, M. Vavoulidis, G. Vedovato, S. Klimenko, et al., "Utilizing gaussian mixture models in all-sky searches for short-duration gravitational wave bursts", Physical Review D **105**, 10.1103/PhysRevD.105.063024 (2022).

[282] V. Skliris, M. R. K. Norman, and P. J. Sutton, "Real-time detection of unmodelled gravitational-wave transients using convolutional neural networks", arXiv preprint arXiv:2009.14611 (2020).

[283] R. Raikman, S. J. Kapadia, C. Wang, X. Liang, and T. G. F. Li, "GWAK: gravitational-wave anomalous knowledge with recurrent autoencoders", arXiv preprint arXiv:2309.14929 (2023).

[284] F. Morawski, M. Bejger, S. Bini, M. Drago, S. Klimenko, et al., "Anomaly detection in gravitational waves data using convolutional autoencoders", Machine Learning: Science and Technology **2**, 10.1088/2632-2153/ac2dcb (2021).

[285] M. Cavaglià, K. Staats, K. Gill, T. Summerscales, and M. J. Szczepańczyk, "Improving the background of gravitational-wave searches for core collapse supernovae: a machine learning approach", arXiv preprint arXiv:2002.04591 (2020).

[286] J. M. Antelis, C. Moreno, J. A. Moreno, A. Iess, G. Pagliaroli, and E. Vigezzi, "Using supervised learning algorithms as a follow-up method in the search of gravitational waves from core-collapse supernovae", Physical Review D **105**, 10.1103/PhysRevD.105.084054 (2022).

[287] A. Iess, F. Ricci, G. Pagliaroli, E. Vigezzi, et al., "Core-collapse supernova gravitational-wave search and deep learning classification", arXiv preprint arXiv:2001.11650 (2020).

[288] A. Iess, G. Pagliaroli, E. Vigezzi, and F. Ricci, "LSTM and CNN application for core-collapse supernova search in gravitational wave real data", Astronomy & Astrophysics **669**, 10.1051/0004-6361/202244364 (2023).

[289] Q. Meijer, M. Lopez, D. Tsuna, and S. Caudill, "Gravitational-wave searches for cosmic string cusps in einstein telescope data using deep learning", Physical Review D **109**, 10.1103/physrevd.109.022006 (2024).

[290] V. Boudart and M. Fays, "Machine learning algorithm for minute-long burst searches", Physical Review D **105**, 10.1103/PhysRevD.105.083007 (2022).

[291] V. Boudart, "Convolutional neural network to distinguish glitches from minute-long gravitational wave transients", Physical Review D **107**, 10.1103/PhysRevD.107.024007 (2023).

[292] J. Bayley, C. Messenger, and G. Woan, "Robust machine learning algorithm to search for continuous gravitational waves", Physical Review D **102**, 10.1103/PhysRevD.102.083024 (2020).

[293] L. M. Modafferi, R. R. Tenorio, and D. Keitel, "Convolutional neural network search for long-duration transient gravitational waves from glitching pulsars", Physical Review D **108**, 10.1103/PhysRevD.108.023005 (2023).

[294] P. M. Joshi and R. Prix, "Novel neural-network architecture for continuous gravitational waves", Physical Review D **108**, 10.1103/PhysRevD.108.063021 (2023).

[295] J. Bayley, C. Messenger, and G. Woan, "Generalized application of the viterbi algorithm to searches for continuous gravitational-wave signals", Physical Review D **100**, 10.1103/PhysRevD.100.023006 (2019).

[296] A. Utina, F. Marangio, F. Morawski, A. Iess, T. Regimbau, G. Fiameni, and E. Cuoco, "Deep learning searches for gravitational wave stochastic backgrounds", in (June 2021).

[297] O. G. Freitas, M. Cavaglià, K. Staats, and K. Gill, "Comparison of neural network architectures for feature extraction from binary black hole merger waveforms", arXiv preprint arXiv:2307.06627 (2023).

[298] M. Andrés-Carcasona, M. Martínez, and L. M. Mir, "Fast bayesian gravitational wave parameter estimation using convolutional neural networks", Monthly Notices of the Royal Astronomical Society **527**, 10.1093/mnras/stad3257 (2023).

[299] H. Gabbard, M. Williams, F. Hayes, and C. Messenger, "Bayesian parameter estimation using conditional variational autoencoders for gravitational-wave astronomy", Nature Physics **18**, 10.1038/s41567-021-01425-7 (2022).

[300] T. Bayes, "An essay towards solving a problem in the doctrine of chances", Philosophical Transactions of the Royal Society of London **53**, 10.1098/rstl.1763.0053 (1763).

[301] S. R. Green, C. Simpson, and J. Gair, "Gravitational-wave parameter estimation with autoregressive neural network flows", Physical Review D **102**, 10.1103/PhysRevD.102.104057 (2020).

[302] S. R. Green and J. Gair, "Complete parameter inference for gw150914 using deep learning", Machine Learning: Science and Technology **2**, 10.1088/2632-2153/abe4b3 (2021).

[303] M. J. Williams, J. Veitch, and C. Messenger, "Nested sampling with normalizing flows for gravitational-wave inference", Physical Review D **103**, 10.1103/PhysRevD.103.103006 (2021).

[304] M. J. Williams, J. Veitch, and C. Messenger, "Importance nested sampling with normalising flows", Machine Learning: Science and Technology **4**, 10.1088/2632-2153/accb18 (2023).

[305] A. Kolmus, A. J. K. Chua, J. R. Gair, et al., "Fast sky localization of gravitational waves using deep learning seeded importance sampling", Physical Review D **106**, 10.1103/PhysRevD.106.013009 (2022).

[306] J. Langendorff, D. Wysocki, J. Gair, and A. J. K. Chua, "Normalizing flows as an avenue to studying overlapping gravitational wave signals", Physical Review Letters **130**, 10.1103/PhysRevLett.130.171402 (2023).

[307] J. I. Thorpe et al., "The Laser Interferometer Space Antenna: Unveiling the Millihertz Gravitational Wave Sky", in Bulletin of the american astronomical society, Vol. 51 (Sept. 2019).

[308] A. Cole, C. Weniger, J. Hermans, G. Louppe, et al., "Fast and credible likelihood-free cosmology with truncated marginal neural ratio estimation", Journal of Cosmology and Astroparticle Physics **2022**, 10.1088/1475-7516/2022/09/004 (2022).

[309] J. Alvey, T. D. P. Edwards, F. Iacovelli, M. Pieroni, et al., "Simulation-based inference for stochastic gravitational wave background data analysis", arXiv preprint arXiv:2309.07919 (2023).

[310] J. Abadie et al., "Search for gravitational waves from compact binary coalescence in ligo and virgo data from s5 and vsr1", Physical Review D **82**, 10.1103/physrevd.82.102001 (2010).

[311] J. Abadie et al., "All-sky search for gravitational-wave bursts in the first joint ligo-geo-virgo run", Physical Review D **81**, 10.1103/physrevd.81.102001 (2010).

[312] C. Biwer et al., "Validating gravitational-wave detections: the advanced ligo hardware injection system", Phys. Rev. D **95**, 10.1103/physrevd.95.062002 (2017).

[313] C.-H. Liao and F.-L. Lin, "Deep generative models of gravitational waveforms via conditional autoencoder", Phys. Rev. D **103**, 10.1103/PhysRevD.103.124051 (2021).

[314] T. Eccleston and M. C. Edwards, "Generative adversarial network for stellar core-collapse gravitational waves", Phys. Rev. D **110**, 10.1103/PhysRevD.110.104055 (2024).

[315] J. Yan, A. P. Leung, and D. C. Y. Hui, *On improving the performance of glitch classification for gravitational wave detection by using generative adversarial networks*, 2022.

[316] J. Powell, L. Sun, K. Gereb, P. D. Lasky, and M. Dollmann, "Generating transient noise artefacts in gravitational-wave detector data with generative adversarial networks", Classical and Quantum Gravity **40**, 10.1088/1361-6382/acb038 (2023).

[317] M. Lopez, V. Boudart, K. Buijsman, A. Reza, and S. Caudill, *Simulating transient noise bursts in ligo with generative adversarial networks*, 2022.

[318] J. McGinn, C. Messenger, M. J. Williams, and I. S. Heng, "Generalised gravitational wave burst generation with generative adversarial networks", Classical and Quantum Gravity **38**, 10.1088/1361-6382/ac09cc (2021).

[319] G. Ashton et al., "Bilby: a user-friendly bayesian inference library for gravitational-wave astronomy", The Astrophysical Journal Supplement Series **241**, 10.3847/1538-4365/ab06fc (2019).

[320] I. M. Romero-Shaw et al., "Bayesian inference for compact binary coalescences with bilby: validation and application to the first ligo–virgo gravitational-wave transient catalogue", Monthly Notices of the Royal Astronomical Society **499**, 10.1093/mnras/staa2850 (2020).

[321] H. Narola, J. Janquart, Q. Meijer, K. Haris, and C. V. D. Broeck, *Relative binning for complete gravitational-wave parameter estimation with higher-order modes and precession, and applications to lensing and third-generation detectors*, 2023.

[322] B. P. A. et al., "A guide to LIGO–virgo detector noise and extraction of transient gravitational-wave signals", Classical and Quantum Gravity **37**, 10.1088/1361-6382/ab685e (2020).

[323] D. Davis et al., "Ligo detector characterization in the second and third observing runs", Classical and Quantum Gravity **38**, 10.1088/1361-6382/abfd85 (2021).

[324] T. Akutsu et al., "Overview of KAGRA: Calibration, detector characterization, physical environmental monitors, and the geophysics interferometer", Progress of Theoretical and Experimental Physics **2021**, 10.1093/ptep/ptab018 (2021).

[325] B. P. Abbott et al. (LIGO Scientific, Virgo), "Effects of data qualKAGRA:2022dwbity vetoes on a search for compact binary coalescences in Advanced LIGO's first observing run", Class. Quant. Grav. **35**, 10.1088/1361-6382/aaaafa (2018).

[326] B. Steltner, M. A. Papa, and H. B. Eggenstein, "Identification and removal of non-Gaussian noise transients for gravitational-wave searches", Phys. Rev. D **105**, 10.1103/PhysRevD.105.022005 (2022).

[327] R. Abbott 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**, 10.1103/physrevx.13.041039 (2023).

[328] B. P. Abbott et al., "Gw190425: observation of a compact binary coalescence with total mass $\sim 3.4\,M_{\odot}$", The Astrophysical Journal Letters **892**, 10.3847/2041-8213/ab75f5 (2020).

[329] R. Abbott et al., "Gw190814: gravitational waves from the coalescence of a 23 solar mass black hole with a 2.6 solar mass compact object", The Astrophysical Journal Letters **896**, 10.3847/2041-8213/ab960f (2020).

[330] R. Abbott et al., "Observation of gravitational waves from two neutron star–black hole coalescences", The Astrophysical Journal Letters **915**, 10.3847/2041-8213/ac082e (2021).

[331] M. Z. et al., "Gravity spy: integrating advanced LIGO detector characterization, machine learning, and citizen science", Classical and Quantum Gravity **34**, 10.1088/1361-6382/aa5cea (2017).

[332] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: an imperative style, high-performance deep learning library", Advances in Neural Information Processing Systems **32** (2019).

[333] B. Zhang, M. Li, C. Zhou, Q. Yang, L. Wang, and B. Yuan, "Endoscopic image denoising algorithm based on spatial attention unet", Journal of Physics: Conference Series **2400**, 10.1088/1742-6596/2400/1/012026 (2022).

[334] S. Nossier, J. Wall, M. Moniri, C. Glackin, and N. Cannings, "A comparative study of time and frequency domain approaches to deep learning based speech enhancement", in (July 2020).

[335] L. Hertel, H. Phan, and A. Mertins, "Comparing time and frequency domain for audio event recognition using deep learning", 2016 International Joint Conference on Neural Networks (IJCNN) (2016).

[336] D. Wang, "Deep learning reinvents the hearing aid", IEEE Spectrum **54**, 10.1109/MSPEC.2017.7864754 (2017).

[337] Y. Pu and H. Yu, "Resunet: a fully convolutional network for speech enhancement in industrial robots", in Advances and trends in artificial intelligence. theory and practices in artificial intelligence: 35th international conference on industrial, engineering and other applications of applied intelligent systems, iea/aie 2022, kitakyushu, japan, july 19–22, 2022, proceedings (2022).

[338] S. Chakrabarty, D. Wang, and E. A. P. Habets, "Time-frequency masking based online speech enhancement with multi-channel data using convolutional neural networks", in 2018 16th international workshop on acoustic signal enhancement (iwaenc) (2018).

[339] D. Wang and J. Lim, "The unimportance of phase in speech enhancement", IEEE Transactions on Acoustics, Speech, and Signal Processing **30**, 10.1109/TASSP.1982.1163920 (1982).

[340] S.-W. Fu, T.-y. Hu, Y. Tsao, and X. Lu, "Complex spectrogram enhancement by convolutional neural network with multi-metrics learning", 2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP) (2017).

[341] Z. Ouyang, H. Yu, W. Zhu, and B. Champagne, "A fully convolutional neural network for complex spectrogram processing in speech enhancement", ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2019).

[342] T. Gerkmann, M. Krawczyk-Becker, and J. Le Roux, "Phase processing for single-channel speech enhancement: history and recent advances", IEEE Signal Processing Magazine **32**, 10.1109/MSP.2014.2369251 (2015).

[343] O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation", in Medical image computing and computer-assisted intervention–miccai 2015: 18th international conference, munich, germany, october 5-9, 2015, proceedings, part iii 18 (Springer, 2015).

[344] B. P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), "All-sky search for short gravitational-wave bursts in the second advanced ligo and advanced virgo run", Phys. Rev. D **100**, 10.1103/PhysRevD.100.024017 (2019).

[345] B. Allen, W. G. Anderson, P. R. Brady, D. A. Brown, and J. D. E. Creighton, "Findchirp: an algorithm for detection of gravitational waves from inspiraling compact binaries", Physical Review D **85**, 10.1103/physrevd.85.122006 (2012).

[346] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., "Scikit-learn: machine learning in python", Journal of machine learning research **12** (2011).

[347] J. Smith, S. U. C. for Computer Research in Music, Acoustics, and S. U. D. of Music, *Spectral audio signal processing* (W3K, 2011).

[348] F. Harris, "On the use of windows for harmonic analysis with the discrete fourier transform", Proceedings of the IEEE **66** (1978).

[349] S. J. Pan and Q. Yang, "A survey on transfer learning", IEEE Transactions on Knowledge and Data Engineering **22**, 10.1109/TKDE.2009.191 (2010).

[350] R. Abbott et al. (KAGRA, VIRGO, LIGO Scientific), "Open Data from the Third Observing Run of LIGO, Virgo, KAGRA, and GEO", Astrophys. J. Suppl. **267**, 10.3847/1538-4365/acdc9f (2023).

[351] S. Soni, E. Marx, E. Katsavounidis, R. Essick, G. S. Cabourn Davies, P. Brockill, M. W. Coughlin, S. Ghosh, and P. Godwin, "Qoq: a q-transform based test for gravitational wave transient events", Classical and Quantum Gravity **41**, 10.1088/1361-6382/ad0922 (2023).

[352] T. Dooney, R. L. Curier, D. S. Tan, M. Lopez, C. Van Den Broeck, and S. Bromuri, "One flexible model for multiclass gravitational wave signal and glitch generation", Phys. Rev. D **110**, 10.1103/PhysRevD.110.022004 (2024).

[353] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: residual learning of deep cnn for image denoising", IEEE Transactions on Image Processing **26**, 10.1109/TIP.2017.2662206 (2017).

[354] D. Rethage, J. Pons, and X. Serra, "A wavenet for speech denoising", 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2017).

[355] J. Areeda, *Are blips and koi fish related?*, https://blog.gravityspy.org/2017/10/19/are-blips-and-koi-fish-related/, Accessed on 11-06-2025, Oct. 2017.

[356] L. S. Collaboration, V. Collaboration, and K. Collaboration, *Gwtc-3: compact binary coalescences observed by ligo and virgo during the second part of the third observing run — parameter estimation data release* (Zenodo, Nov. 2021).

[357] H. Narola, T. Wouters, L. Negri, M. Lopez, T. Dooney, F. Cireddu, M. Wils, I. C. F. Wong, P. T. H. Pang, J. Janquart, A. Samajdar, C. Van Den Broeck, and T. G. F. Li, "Null-stream-based third-generation-ready glitch mitigation for gravitational wave measurements", Phys. Rev. D **112**, 10.1103/l6tp-ykxp (2025).

[358] R. Udall et al., "The anti-aligned spin of GW191109: glitch mitigation and its implications", (2024).

[359] S. Ghonge et al., "Assessing and Mitigating the Impact of Glitches on Gravitational-Wave Parameter Estimation: a Model Agnostic Approach", (2023).

[360] S. Vandenhende, B. De Brabandere, D. Neven, and L. Van Gool, *A three-player gan: generating hard samples to improve classification networks*, 2019.

[361] Y. Shen, P. Luo, P. Luo, J. Yan, X. Wang, and X. Tang, "Faceid-gan: learning a symmetry three-player gan for identity-preserving face synthesis", in 2018 ieee/cvf conference on computer vision and pattern recognition (2018).

[362] T. D. Nguyen, T. Le, H. Vu, and D. Phung, *Dual discriminator generative adversarial nets*, 2017.

[363] M. Arjovsky, S. Chintala, and L. Bottou, *Wasserstein gan*, 2017.

[364] N. Kodali, J. Abernethy, J. Hays, and Z. Kira, *On convergence and stability of gans*, 2017.

[365] F. Chollet et al., *Keras*, (2015) `https://github.com/fchollet/keras`.

[366] M. A. et al., *TensorFlow: large-scale machine learning on heterogeneous systems*, Software available from tensorflow.org, 2015.

[367] LIGO Scientific Collaboration, *LIGO Algorithm Library - LALSuite*, free software (GPL), 2018.

[368] B. P. A. et al., "Binary black hole population properties inferred from the first and second observing runs of advanced LIGO and advanced virgo", The Astrophysical Journal **882**, `10.3847/2041-8213/ab3800` (2019).

[369] B. P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), "All-sky search for short gravitational-wave bursts in the second advanced ligo and advanced virgo run", Phys. Rev. D **100**, `10.1103/PhysRevD.100.024017` (2019).

[370] A. Torres et al., "Total-variation-based methods for gravitational wave denoising", Phys. Rev. D **90**, `10.1103/PhysRevD.90.084029` (2014).

[371] N. Gallagher, *Savitzky-golay smoothing and differentiation filter*, Jan. 2020.

[372] "Kolmogorov–smirnov test", in *The concise encyclopedia of statistics* (Springer New York, New York, NY, 2008).

[373] A. N. et al., *Gwastro/pycbc: v2.0.4 release of pycbc*, version v2.0.4, June 2022.

[374] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, *Spectral normalization for generative adversarial networks*, 2018.

[375] T. Dooney, S. Bromuri, and L. Curier, "Dvgan: stabilize Wasserstein GAN training for time-domain gravitational wave physics", in 2022 ieee international conference on big data (big data) (2022).

[376] D. M. Macleod, J. S. Areeda, S. B. Coughlin, T. J. Massinger, and A. L. Urban, "GWpy: A Python package for gravitational-wave astrophysics", SoftwareX **13**, `10.1016/j.softx.2021.100657` (2021).

[377] P. Ajith, S. Babak, Y. Chen, M. Hewitson, B. Krishnan, A. M. Sintes, J. T. Whelan, B. Brügmann, P. Diener, N. Dorband, J. Gonzalez, M. Hannam, S. Husa, D. Pollney, L. Rezzolla, L. Santamaría, U. Sperhake, and J. Thornburg, "Template bank for gravitational waveforms from coalescing binary black holes: nonspinning binaries", Phys. Rev. D **77**, `10.1103/PhysRevD.77.104017` (2008).

[378] V. Varma, S. E. Field, M. A. Scheel, J. Blackman, L. E. Kidder, and H. P. Pfeiffer, "Surrogate model of hybridized numerical relativity binary black hole waveforms", Phys. Rev. D **99**, `10.1103/PhysRevD.99.064045` (2019).

[379] M. Tiglio and A. Villanueva, "Reduced order and surrogate models for gravitational waves", Living Rev. Rel. **25**, `10.1007/s41114-022-00035-w` (2022).

[380] X. Wei, B. Gong, Z. Liu, W. Lu, and L. Wang, *Improving the improved training of wasserstein gans: a consistency term and its dual effect*, 2018.

[381] W. Peebles and S. Xie, *Scalable diffusion models with transformers*, 2023.

[382] M. Armano et al. (LISA Pathfinder Collaboration), "Sub–femtog $s^{-2}/\sqrt{\text{Hz}}$ space inertial sensor performance from lisa pathfinder", Phys. Rev. Lett. **116**, `10.1103/PhysRevLett.116.231101` (2016).

[383] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks", in Proceedings of the 36th international conference on machine learning (icml) (2019).

Appendix

Chapter A

DeepExtractor

A.1 Simulated Training Glitches

Table A.1 outlines the analytical models employed to generate the proxy glitch classes for training DeepExtractor, along with the parameters used in their functions.

Function	Description	Key Parameters
chirp	Generates a linear chirp signal with frequency varying from `f0` to `f1` over the duration.	`f0_min, f0_max, f1_min, f1_max, duration, sample_rate`
sine	Generates a simple sine wave with a frequency randomly chosen between `freq_min` and `freq_max`.	`freq_min, freq_max, duration, sample_rate`
sine-gaussian	Generates a sine wave modulated by a Gaussian envelope.	`freq_min, freq_max, duration, sample_rate`
Gaussian pulse	Generates a Gaussian pulse with randomized frequency and bandwidth characteristics.	`fc_min, fc_max, bw_min, bw_max, bwr_min, bwr_max, tpr_min, tpr_max, duration, sample_rate`
ringdown	Generates a ringdown signal based on a damped sinusoidal model with a random frequency `f0`, decay time `tau`, and quality factor `Q`. There is also a 50% probability to flip the signal horizontally.	`duration, sample_rate, n_signals`

Table A.1: Summary of analytical glitch models used to simulate glitches for training DeepExtractor.

A.2 STFT Parameters

Table A.2 details the parameters and their respective values used for STFT computation, enabling transformations to and from the time-frequency (spectrogram) domain.

Parameter	Description	Value
`n_fft`	Size of the Fast Fourier Transform (FFT), typically a power of two. Determines the frequency resolution.	512
`window_length`	Length of the window function used in the FFT, defining the signal segment analyzed at once.	64
`hop_length`	Step size between consecutive FFT windows, determining their overlap.	32
Hann window	A smooth tapering window that reduces spectral leakage and satisfies the COLA condition when `hop_length` = `window_length`/2.	Used
COLA condition	The Constant Overlap-Add (COLA) condition ensures continuous signal reconstruction from overlapped segments.	Satisfied by Hann window and chosen hop length.

Table A.2: Parameters used in the short-time Fourier transform (STFT) for transforming between the time and spectrogram domains in DeepExtractor.

A.3 Test samples generated by gengli

FIG. A.1 shows samples generated by gengli, used to evaluate the generalization capability of DeepExtractor in Chapters 4 and 5.

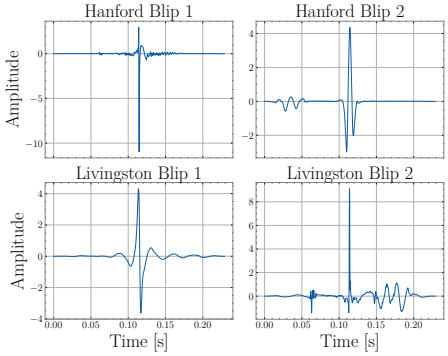


Figure A.1: Two example Blip glitches from the Hanford (top row) and Livingston (bottom row) detectors, generated using gengli.

A.4 Hyperparameters for t-SNE and UMAP

Table A.3: Hyperparameters for t-SNE and UMAP

Algorithm	Hyperparameter	Value
t-SNE	`N_components`	3
	`Perplexity`	40
	`Metric`	`euclidean`
UMAP	`N_neighbors`	15
	`Min_dist`	0.6
	`N_components`	3
	`Metric`	`correlation`

A.5 Extracted Gravity Spy Glitches

FIG. A.2, A.3 and A.4 show DeepExtractor reconstructions for glitch classes from the *Gravity Spy* dataset. Each figure shows reconstructions for three glitch classes which are named above each column. The top row shows a Q-scan of the input to the network. The middle row shows a Q-scan after removing DeepExtractor's reconstruction. The bottom plots show the time series of the input and the reconstruction. The maximum color limit in the Q-scans is set to 25, similarly to the Q-scans shown in FIG. 4.9.

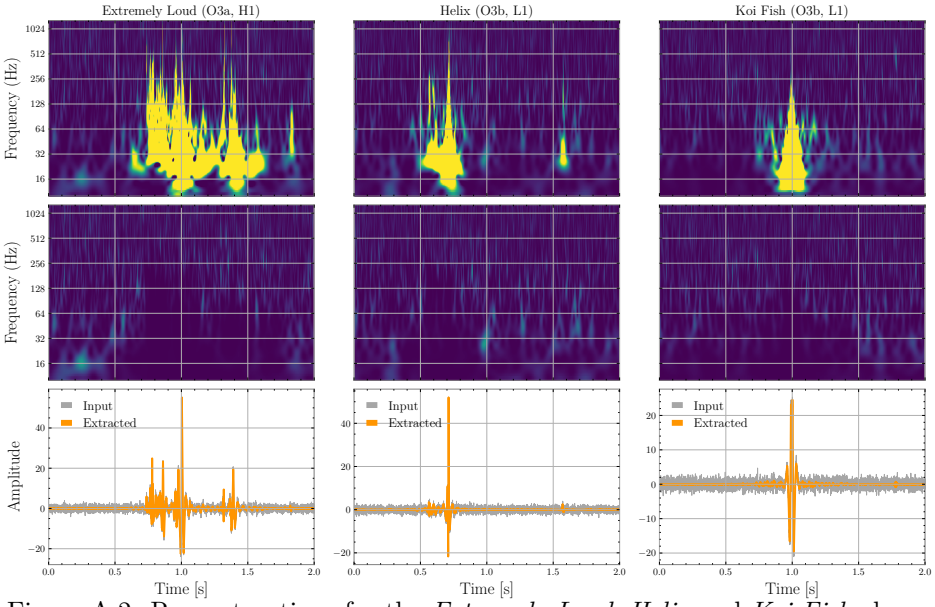


Figure A.2: Reconstructions for the *Extremely Loud*, *Helix* and *Koi Fish* classes

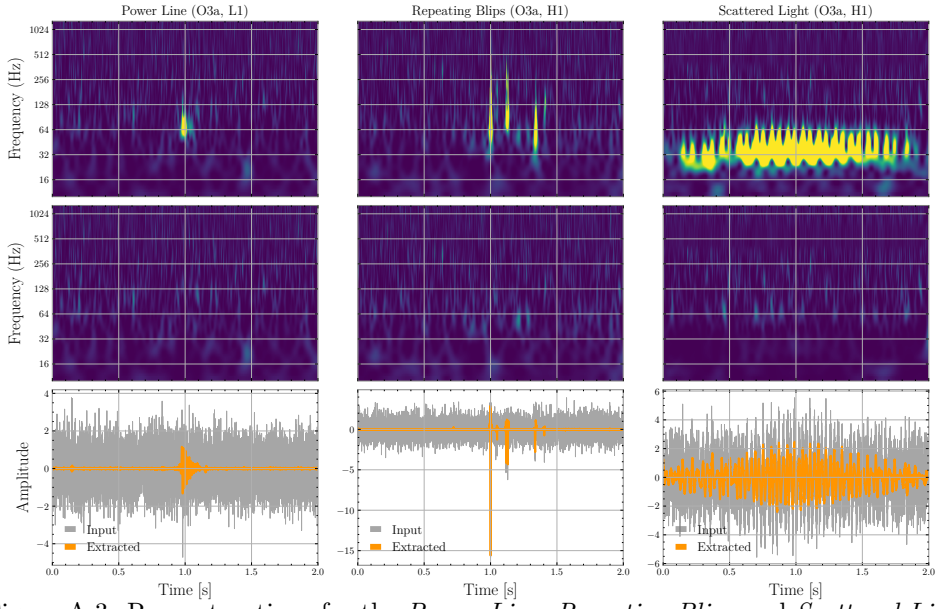


Figure A.3: Reconstructions for the *Power Line*, *Repeating Blips* and *Scattered Light* classes

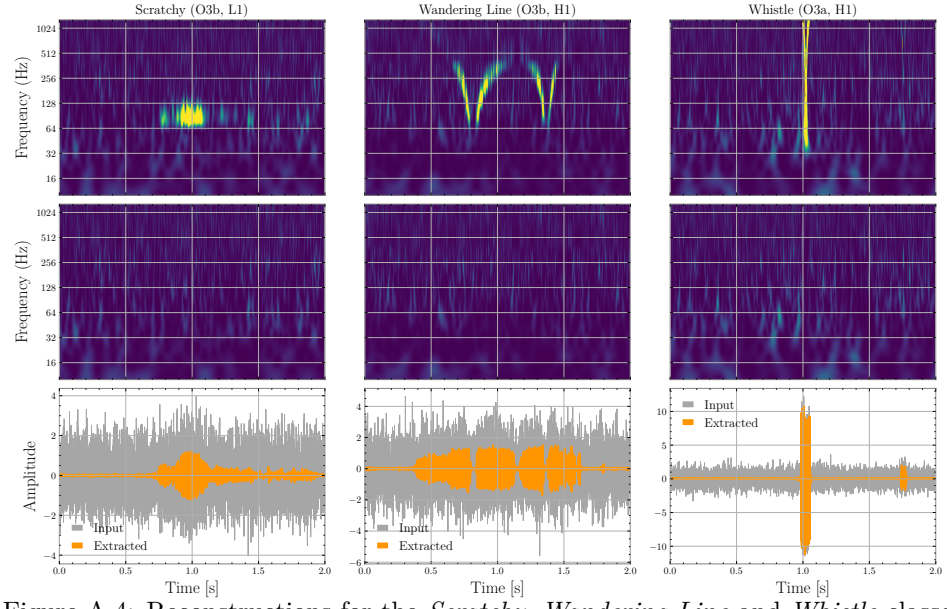
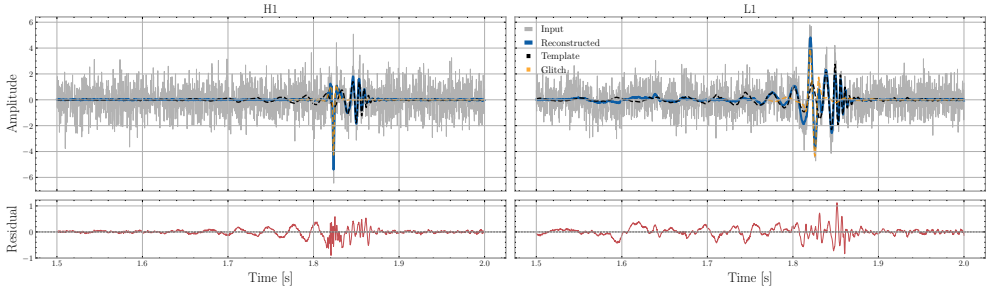


Figure A.4: Reconstructions for the *Scratchy*, *Wandering Line* and *Whistle* classes

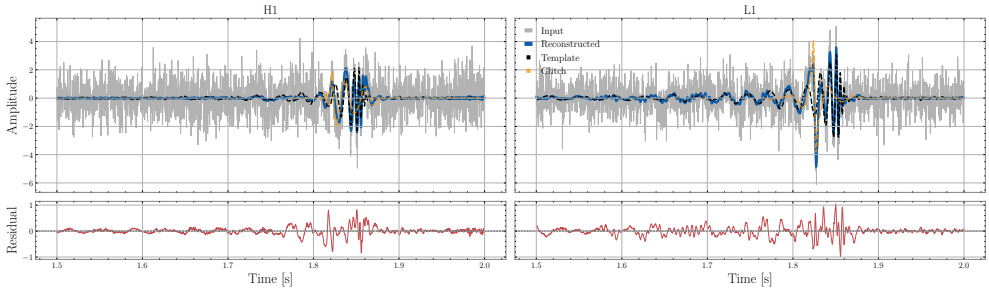
A.6 Reconstructing O3 GWs with glitches

FIG. A.5 shows DeepExtractor reconstructions for the same events shown in Section 4.3.4 but with blip glitches injected close to the merger using gengli. It is observed

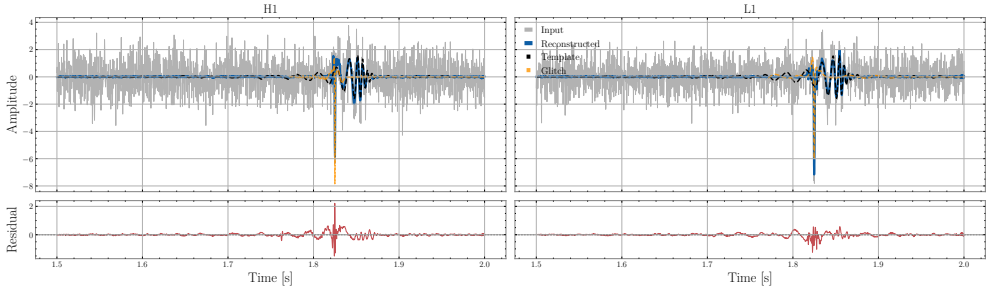
that when glitches are coincident with GWs, that DeepExtractor will reconstruct both sources of excess power without distinguishing between them.



(a) *GW190521_074359*



(b) *GW200129_065458*



(c) *GW200224_222234*

Figure A.5: DeepExtractor reconstructions of GW signals from the same three O3 events shown in Section 4.3.4, now with blip glitches injected near the merger in each detector. Glitches are injected with SNRs comparable to the matched-filter SNR of the corresponding maximum likelihood waveform. Each panel displays the model reconstruction overlaid with the aligned template and injected glitch. The residual plot compares the reconstruction to the linear combination of the signal and glitch.

Chapter B

DVGAN

B.1 DVGAN Architecture

		Discriminator			(2.5M, 3.5M, **4.1M** param.)
Operation	Output shape	Kernel size	Stride	Dropout	Activation
Signal input	(1024)	-	-	0	-
Reshape	(1024,1)	-	-	0	-
Convolutional	(512,32), (-), **(-)**	5	2	0.5	Leaky ReLU
Convolutional	(256,64), (-), **(512,64)**	5	2	0.5	Leaky ReLU
Convolutional	(128, 128), (512,64), **(256,128)**	5	2	0.5	Leaky ReLU
Convolutional	(64, 256), (256,128), **(128,256)**	5	2	0.5	Leaky ReLU
Convolutional	(32, 512), (128,256), **(64,512)**	5	2	0.5	Leaky ReLU
Flatten	(16384), (32768), **(32768)**	-	-	0.5	-
Dense	(100)	-	-	0.2	Leaky ReLU
Dense	(1)	-	-	0	Sigmoid
		DV Discriminator			(300k param.)
Operation	Output shape	Kernel size	Stride	Dropout	Activation
Signal input	(1023)	-	-	0	-
Dense	(1024)	-	-	0	-
Reshape	(512,2)	-	-	0	-
Convolutional	(256,64)	5	2	0.5	Leaky ReLU
Convolutional	(128,128)	5	2	0.5	Leaky ReLU
Convolutional	(64,256)	5	2	0.5	Leaky ReLU
Flatten	(16384)	-	-	0.5	-
Dense	(1)	-	-	0	Sigmoid
		Generator			(2.5M, 3.5M, **4.1M** param.)
Operation	Output shape	Kernel size	Stride	BN	Activation
Latent input	(100)	-	-	–	-
Dense	(16384), (32768), **(32768)**	-	-	–	ReLU
Reshape	(32,512), (128,256), **(64,512)**	-	-	–	-
Transposed conv.	(64,256), (-), **(-)**	5	2	✓	ReLU
Transposed conv.	(128,128), (-), **(128,256)**	5	2	✓	ReLU
Transposed conv.	(256,64), (256,128), **(256,128)**	5	2	–	ReLU
Transposed conv.	(512,32), (512,64), **(512,64)**	5	2	–	ReLU
Transposed conv.	(1024,1), (1024, 1), **(1024,1)**	5	2	–	Linear
Reshape	(1024)	-	-	–	-
Optimizer	RMSprop($\alpha = 0.0001$)				
Batch size	512				
Epochs	500				
Loss	Wasserstein				

Table B.1: Architecture and hyperparameters of DVGAN, including the discriminator, derivative discriminator (DV), and generator networks. Bold entries indicate the optimal configuration used in results.

B.2 CNN Network for Discriminative Score

Operation	Output shape	Kernel size	Stride	Dropout	Activation
Signal input	(1024)	–	–	0	–
Reshape	(1024,1)	–	–	0	–
Convolutional	(512,32)	5	2	0.5	Leaky ReLU
Convolutional	(256,64)	5	2	0.5	Leaky ReLU
Flatten	(16384)	–	–	0.5	–
Dense	(1)	–	–	0	Sigmoid

(a) Layer architecture.

Hyperparameter	Value
Optimizer	RMSprop ($\alpha = 0.0001$)
Batch size	64
Epochs	20
Loss	Binary Crossentropy

(b) Training configuration.

Table B.2: Post-hoc Discriminative CNN (27k parameters). The CNN applied to blip glitches is identical except for the input signal shape (938).

Chapter C

cDVGAN

C.1 cDVGAN Architecture

		Discriminator			(3.5M parameters)
Operation	**Output shape**	**Kernel size**	**Stride**	**Dropout**	**Activation**
Input	(1024)	–	–	0	–
Reshape	(64,16)	–	–	0	–
Convolutional	(64,128)	14	2	0.5	Leaky ReLU
Convolutional	(32,128)	14	2	0.5	Leaky ReLU
Convolutional	(16,256)	14	2	0.5	Leaky ReLU
Convolutional	(8,256)	14	2	0.5	Leaky ReLU
Convolutional	(4,512)	14	2	0.5	Leaky ReLU
Global Avg. Pooling	(512)	–	–	0.5	–
Avg. Pooling Dense	(128)	–	–	0.2	Leaky ReLU
Dense	(1)	–	–	0	Linear
Class input	(3)	–	–	–	–
Class Dense	(128)	–	–	0	Linear
Scalar product	(1)	–	–	–	–
Dense + Scalar product	(1)	–	–	–	–
		DV Discriminator			(1.1M parameters)
Operation	**Output shape**	**Kernel size**	**Stride**	**Dropout**	**Activation**
Input	(1023)	–	–	0	–
Dense	(512)	–	–	0	Leaky ReLU
Reshape	(32,16)	–	–	0	–
Convolutional	(32,64)	5	2	0.5	Leaky ReLU
Convolutional	(16,128)	5	2	0.5	Leaky ReLU
Convolutional	(8,256)	5	2	0.5	Leaky ReLU
Convolutional	(4,256)	5	2	0.5	Leaky ReLU
Global Avg. Pooling	(256)	–	–	0.5	–
Avg. Pooling Dense	(128)	–	–	0.2	Leaky ReLU
Dense	(1)	–	–	0	Linear
Class input	(3)	–	–	–	–
Class Dense	(128)	–	–	0	Linear
Scalar product	(1)	–	–	–	–
Dense + Scalar product	(1)	–	–	–	–
		Generator			(3.5M parameters)
Operation	**Output shape**	**Kernel size**	**Stride**	**BN**	**Activation**
Latent input	(100)	–	–	–	–
Class input	(3)	–	–	–	–
Class Dense	(32)	–	–	–	–
Concatenate	(132)	–	–	–	–
Dense	(1024)	–	–	–	ReLU
Reshape	(32,32)	–	–	–	–
Transposed conv.	(64,512)	18	2	✓	ReLU
Transposed conv.	(128,256)	18	2	✓	ReLU
Transposed conv.	(256,128)	18	2	✓	ReLU
Transposed conv.	(512,64)	18	2	✓	ReLU
Transposed conv.	(1024,1)	18	2	–	Linear
Flatten	(1024)	–	–	–	–
Optimizer	RMSprop ($\alpha = 0.0001$)				
Batch size	512				
Epochs	500				
Loss	Wasserstein				

Table C.1: Architecture and hyperparameters of cDVGAN. The model includes a base discriminator, derivative (DV) discriminator, and generator. The cDVGAN2 variant introduces an additional discriminator identical to the DV discriminator but with an input length of 1022

C.2 CNN Architecture

Operation	Output shape	Kernel size	Stride	Dropout	Activation
Input	(1024)	–	–	0	–
Reshape	(1024,1)	–	–	0	–
Convolutional	(512,256)	5	2	0.5	Leaky ReLU
Convolutional	(256,128)	5	2	0.5	Leaky ReLU
Convolutional	(128,64)	5	2	0.5	Leaky ReLU
Convolutional	(64,32)	5	2	0.5	Leaky ReLU
Flatten	(2048)	–	–	0.5	–
Dense	(512)	–	–	0	Leaky ReLU
Dense	(1)	–	–	0	Sigmoid

(a) CNN architecture (1.2M parameters).

Hyperparameter	Value
Optimizer	Adam ($\alpha = 0.001$)
Batch size	64
Epochs	20
Loss	Binary Crossentropy

(b) Training configuration.

Table C.2: The architecture of the CNN (1.2M parameters) used during experiments.

C.3 Class Interpolation

Figure C.1 shows smooth interpolation between the trained classes by sampling the conditioned class vector.

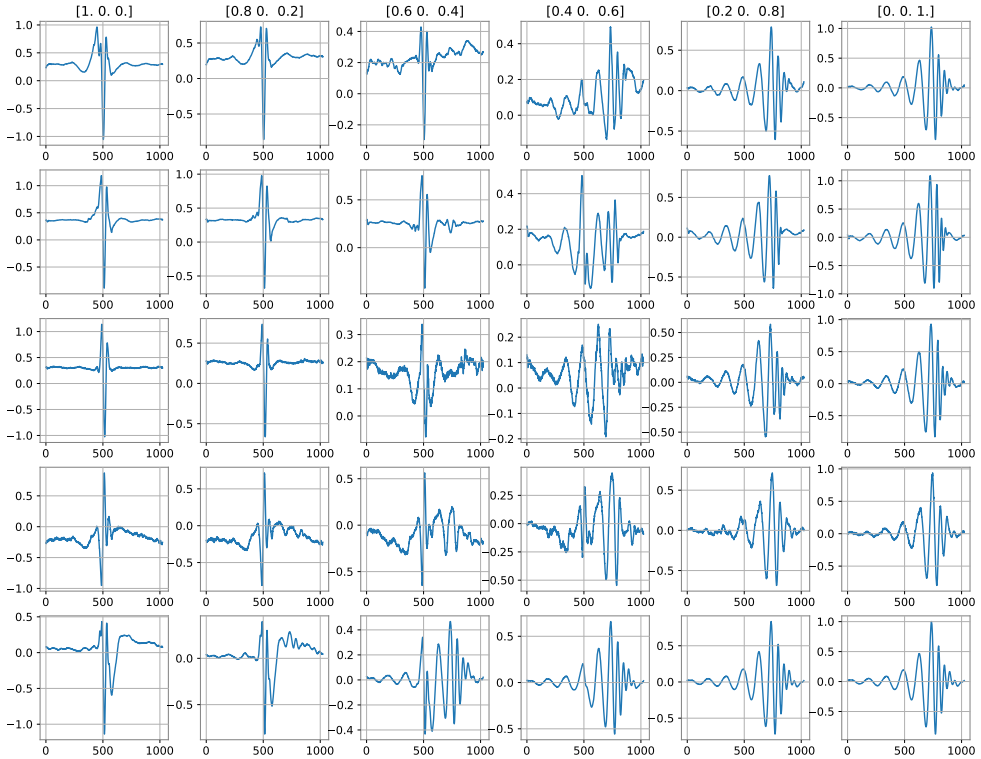


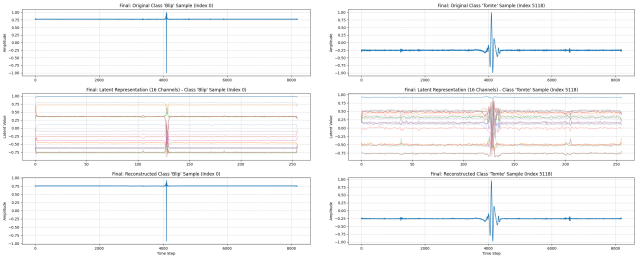
Figure C.1: Interpolation between blip and BBH classes for cDVGAN (1st row), cDVGAN2 (2nd row), cWGAN (3rd row), McGANn (4th row) and McDVGANn (5th row). The class input is shown at the top of each column, while the latent input of the generator is kept constant.

Chapter D

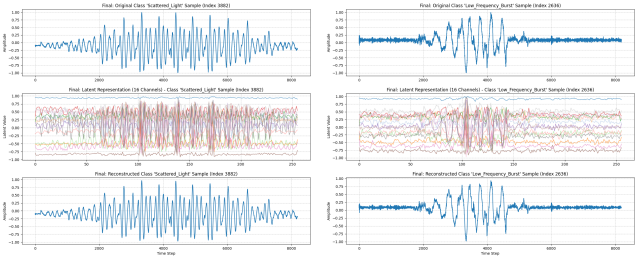
Learning the Glitch Space with cD-VGAN

D.1 Results of autoencoder

In Fig D.1 the results of the trained autoencoder with a bottleneck size of $[16, 256]$ are shown. For each class the top plot is the input data. The middle plot shows the 16 bottleneck channels and the bottom plot shows the reconstruction of made from autoencoder.



(a) Blip (b) Tomte



(c) Scattered light (d) Low frequency burst

Figure D.1: Results of trained Autoencoder. Each plot contains the input data, the 16 channels of the bottleneck data and the reconstructed data.

D.2 Varying the Injected SNR in the Gravity Spy classification Experiment

Figure D.2 shows *Gravity Spy* classifications for cDVGAN-generated samples injected into background noise at an SNR of 20, highlighting the dependence of *Gravity Spy* classifications on SNR when compared with the analysis in Chapter 8.

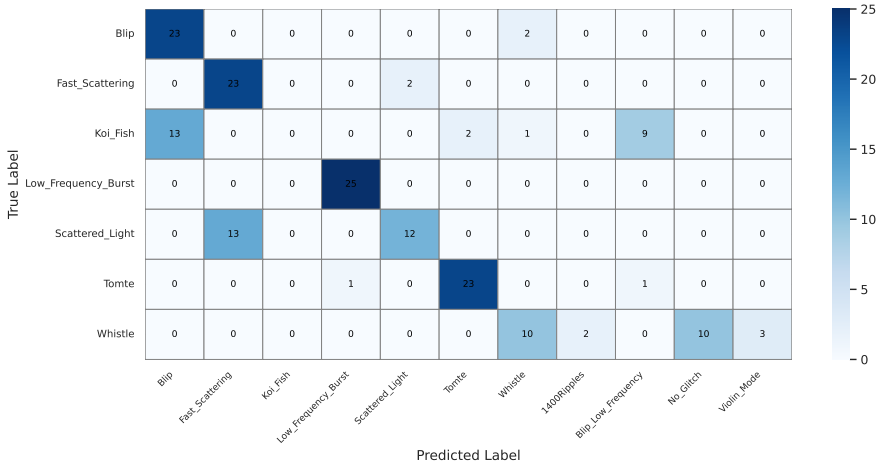


Figure D.2: Confusion matrix showing *Gravity Spy* classifications of glitches generated by cDVGAN, with 25 samples per class with an injected SNR of 20, compared to that shown in Chapter 8 where an injected SNR of 100 was used.