Mémoire

Auteur : Peters, Sacha
Promoteur(s) : Fays, Maxime
Faculté : Faculté des Sciences
Diplôme : Master en sciences spatiales, à finalité approfondie
Année académique : 2023-2024
URI/URL : http://hdl.handle.net/2268.2/20153

Enhancement of a machine learning algorithm for long-duration gravitational wave burst searches

Sacha Peters

Faculty of Sciences

Master in Space Sciences
Research Focus

Supervisor:
Pr. Maxime Fays

Reading Committee:
Pr. Jean-René Cudell
Pr. Gilles Louppe
Dr. Matteo Pracchia

2023-2024

Acknowledgments

I would first like to thank my supervisor, Pr. Maxime Fays, for his constant support and advice, as well as for all the technical help and the hours spent discussing this work. I also thank the members of my reading committee. I must acknowledge Pr. Jean-René Cudell for his straightforward yet insightful comments on this work, and Pr. Gilles Louppe for his thought-provoking lectures, which sparked some ideas for this thesis.

I thank my high school math teacher, M. Lesoinne, for showing me the beauty of mathematics. I am grateful to all my friends for making these years fun and memorable, special thanks to Adrien, the gravitational wave enjoyer who understands all my jokes, and to Guillaume and Cedric, the true coding masters. I am deeply thankful to my family for their support during difficult times. Lastly, I thank Sasha for her unconditional support and love.

Contents

Introduction

Gravitational waves are predictions of General Relativity, a theory that emerged at the start of the XX$^{\text{th}}$ century. Direct observation through advanced laser interferometers of gravitational waves generated by the coalescence of two black holes has been reported a century later. Since then many other Compact Binary Coalescence (CBC) events have been detected. Other astrophysical phenomena are expected to generate gravitational waves and are searched in the data stream of these detectors.

This work is part of the search for minute-long burst searches which are different from CBC events because they are longer and not well-modelled thus the more traditional template-based methods cannot be used. Long-duration burst searches use a time-frequency representation of the correlation of the data from multiple detectors. These are spectrograms which show the time-frequency evolution of signals. These spectrograms are dominated by the background noise of the detectors and transient non-astrophysical noise called glitches.

The GWpyxel pipeline searches for signals in spectrograms using fast and effective deep learning methods, namely the ALBUS model which acts as a non-linear noise removal filter for spectrograms to highlight potential signals and discriminate glitches. ALBUS takes as input spectrograms possibly containing signals and glitches partially hidden by the noise and outputs a spectrogram without the background noise, containing the reconstructed signals. Its output is then clustered to form triggers which indicate where each signal is seen in the spectrogram. Significant triggers can be sent to astronomers for further investigation and to potentially find their electromagnetic counterparts.

The current clustering process to obtain the triggers needs improvements as it finds an average of 5 to 6 low-significance non-physical triggers per 1000 seconds of data which unnecessarily increases the processing time. Furthermore, one signal is often clustered as two or more separate triggers, leading to the possibility that a significant astrophysical signal could be missed by being fragmented into several less significant triggers. This work presents a modification to ALBUS which changes its role from a noise removal filter to an image segmentation model which directly outputs the triggers.

The first chapter introduces the theory behind gravitational waves, from their emergence out of the theory of General Relativity to the interferometer detectors on Earth and the detected and potentially detectable astrophysical sources. The second chapter presents the technical aspects needed to analyse the data of the detectors. The third chapter is the practical work of this thesis which aims at enhancing the clustering process inside the minute-long gravitational wave burst search pipeline GWpyxel.

Chapter 1

Gravitational Waves

In this section, we will cover the theoretical origin of gravitational waves, the principles behind the detectors, and the potential sources of such waves, especially the long-duration burst sources.

1.1 General Relativity

General Relativity (GR), proposed by Albert Einstein in 1915 [1], is the prevailing theory of gravity. This section is adapted from [2] and will provide a short introduction to the theory and how GW emerge from it.

We start by introducing the basics of calculations in relativity. This is a four-dimensional theory, events $\mathbf{x}$ are notated as quadri-vectors,

$$\mathbf{x} = (ct, x, y, z), \tag{1.1}$$

With c being the speed of light, and t, x, y, z the coordinates in time and space. The speed of light is introduced to have the same dimensions across the vector's components. We can use natural units, where c = 1, to simplify the notations, this implies that the dimensions of space and time are equivalent. We remove the bold notation for vectors and introduce an index μ to notate the vector's components. Our quadri-vector becomes,

$$x^\mu = (x^0, x^1, x^2, x^3) = (t, x, y, z). \tag{1.2}$$

Higher order tensors are notated via two or more indices such as $T^{\mu\nu}$. One useful notation is the Einstein summation convention, where a sum is implicitly made over repeated lower and upper indices. The sums typically run over three to four elements, relating to the three spatial dimensions and the fourth time dimension. Greek indices run from 0 to 3 and Latin from 1 to 3. For two quadri-vectors x and y the convention is

$$\sum_{\mu=(0,1,2,3)} x_\mu y^\mu \equiv x_\mu y^\mu \qquad \text{and} \qquad \sum_{i=(1,2,3)} x_i y^i \equiv x_i y^i. \tag{1.3}$$

To measure distances we introduce the line element ds which is defined as follows,

$$ds^2 = g_{\mu\nu} dx^\mu dx^\nu, \tag{1.4}$$

with dx being an infinitesimal displacement and $g_{\mu\nu}$ being the metric tensor. In special relativity, spacetime is the in Minkowski spacetime and the metric tensor is

$$g_{\mu\nu} = \eta_{\mu\nu} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{1.5}$$

so the line element becomes

$$ds^2 = -dt^2 + dx^2 + dy^2 + dz^2. \tag{1.6}$$

In GR the metric tensor can have different values and non-null off-diagonal terms. GR states that the presence of mass and energy causes spacetime to curve. The Einstein field equations mathematically describe this curvature:

$$G_{\mu\nu} = 8\pi G T_{\mu\nu}, \tag{1.7}$$

where $G_{\mu\nu}$ is the Einstein tensor representing spacetime curvature, $T_{\mu\nu}$ is the energy-momentum tensor describing the mass and energy content in the spacetime, and G is the gravitational constant. To explore the emergence of gravitational waves, we employ the weak field approximation, where the metric tensor $g_{\mu\nu}$ can be expressed as the Minkowski metric $\eta_{\mu\nu}$ plus a small perturbation $h_{\mu\nu}$,

$$g_{\mu\nu} = \eta_{\mu\nu} + h_{\mu\nu}, \tag{1.8}$$

which allows the linearization of the Einstein field equation by inserting this expression inside the Einstein tensor $G_{\mu\nu}$

$$G_{\mu\nu} = R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R, \tag{1.9}$$

where $R_{\mu\nu}$ is the Ricci tensor, R is the Ricci scalar, and $g_{\mu\nu}$ is the metric tensor. The Ricci tensor is a contraction of the Riemann tensor $R^{\alpha}{}_{\mu\beta\nu}$

$$R_{\mu\nu} = R^{\alpha}{}_{\mu\alpha\nu}. \tag{1.10}$$

The Riemann tensor has the expression

$$R^{\alpha}{}_{\mu\beta\nu} = \partial_\beta \Gamma^{\alpha}_{\mu\nu} - \partial_\nu \Gamma^{\alpha}_{\mu\beta} + \Gamma^{\alpha}_{\sigma\beta} \Gamma^{\sigma}_{\mu\nu} - \Gamma^{\alpha}_{\sigma\nu} \Gamma^{\sigma}_{\mu\beta}, \tag{1.11}$$

where $\Gamma^{\alpha}_{\mu\nu}$ are the Christoffel symbols which describe how vectors change as they are transported along coordinates in spacetime and are defined as

$$\Gamma^{\alpha}_{\mu\nu} = \frac{1}{2} g^{\alpha\sigma} \left(\partial_\mu g_{\nu\sigma} + \partial_\nu g_{\mu\sigma} - \partial_\sigma g_{\mu\nu} \right). \tag{1.12}$$

These symbols depend on the inverse metric $g^{\alpha\sigma}$, which is given by

$$g^{\alpha\sigma} = \eta^{\alpha\sigma} + h^{\alpha\sigma}. \tag{1.13}$$

To simplify the linearized equation, a change of variable known as the trace-reverse transformation is applied, it is

$$\bar{h}_{\mu\nu} = h_{\mu\nu} - \frac{1}{2} \eta_{\mu\nu} h, \tag{1.14}$$

where $h = \eta^{\mu\nu} h_{\mu\nu}$ is the trace of $h_{\mu\nu}$. In the Lorentz gauge ($\partial_\mu \bar{h}^{\mu\nu} = 0$) and in the absence of any sources ($T_{\mu\nu} = 0$), the linearized gravitational wave equation becomes

$$\Box \bar{h}_{\mu\nu} = 0, \tag{1.15}$$

where $\Box$ represents the d'Alembertian operator. This wave equation describes the propagation of gravitational waves, that travel at the speed of light. For a plane-wave solution travelling in the z direction, we have

$$\bar{h}_{\mu\nu} = A_{\mu\nu} e^{ik_\sigma x^\sigma}, \tag{1.16}$$

with k being the wave vector, and A a 16 component tensor. Using the symmetry of $A_{\mu\nu}$, the Lorenz gauge ($A_{\mu\nu} k^\mu = 0$) which restricts the choice of coordinates, and the traceless-transverse gauge ($A^\mu_{\ \mu}$=0) which is another constraint allowed in the Lorenz gauge, these 16 components reduce to 2 independent components A_+ and $A_\times$,

$$A_{\mu\nu} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & A_+ & A_\times & 0 \\ 0 & A_\times & -A_+ & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}. \tag{1.17}$$

These two independent components relate to the two polarisations h_+ and $h_\times$ of a gravitational wave. The effect of two linearly polarized waves on a ring of matter is illustrated in Fig 1.1.

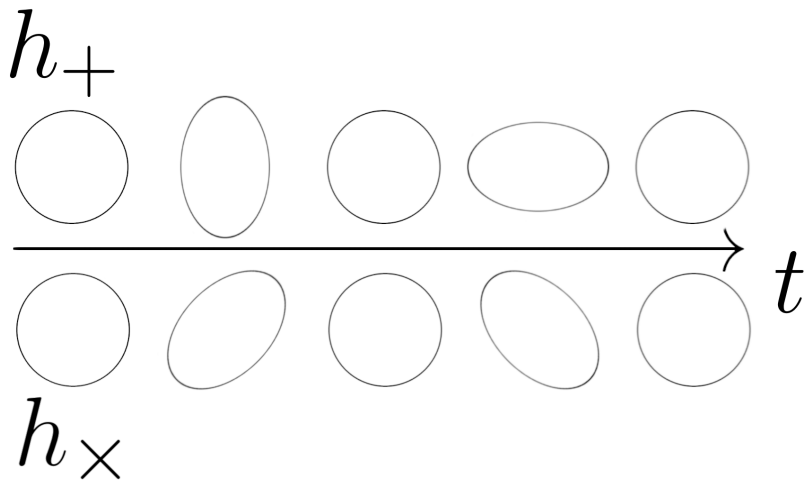


Figure 1.1: Representation of the effect of the passage of a linearly polarised (either h_+ or $h_\times$) GW on a ring of matter over time.

1.2 Detectors

General relativity predicts the existence of gravitational waves, this section will detail the instruments designed to detect those waves.

The first detectors were rigid bars of metal with a specific resonant frequency, if a gravitational wave of that frequency would pass through the detector bar, then a tiny

vibration could be sensed. Claims of detection were made [3] but have been proven to be wrong [4].

Figure 1.2: Joseph Weber besides one of his gravitational wave detector bars, at the University of Maryland [5].

The minimum attainable strain $h_{\min}$ for these detectors is given by

$$h_{\min} \leq \frac{1}{\omega_0 L Q} \left(\frac{15 k_B T}{M} \right)^{1/2}, \tag{1.18}$$

with ω_0 being the resonant frequency, L the length, Q a quality factor equal to $\tau\omega_0$ with τ being the damping time, k_B the Boltzmann constant, T the temperature, and M the mass of the bar.

In the case of Weber's bar, the mass of the detector M was 1,410 kg, the length L was 1.5 m, the resonant frequency ω_0 was 1,660 Hz (which was thought to be suited for detecting supernovae), and the quality factor Q was 2×10^5 at room temperature. With these parameters, the smallest detectable strain $h_{\min}$ would be of the order of 10^{-20}. We must note that this order of magnitude is only attainable if the signal perfectly matches the resonant frequency of the bar and if it lasts for at least τ, which is equal to 120 seconds in this case. Supernovae signals are now known to cover a wider range of frequencies and to last about 1 second [6] which renders them undetectable by this instrument.

Refinements of such detectors exist, such as cryogenically cooled spherical detectors [7]. Still, none of them have detected an astrophysical signal, due to their low sensitivity and narrow range of frequencies.

Modern-day detectors are based on the Michelson interferometer in which a laser sends light through a beam splitter, and the two rays of light travel through kilometre-long vacuum-sealed arms, and mirrors at each end of the arms reflect the two beams of light which are merged while crossing the beam splitter, and a photodetector measures any phase difference.

One key assumption that has to be taken to be able to detect anything is that the wavelength of the GW signal is much longer than the size of the detector so that during

the travel time of the light, the detector arms are not stretched by a significant amount [8]. These interferometers are kilometre-sized and they search for signals around 1000 Hz, which corresponds to a wavelength on the order of 100 kilometres. The wavelength is $\sim$100x times longer than the size of the interferometer so the long wavelength assumption is valid.

Multiple refinements to the original Michelson interferometer have been made to achieve an extremely high sensitivity capable of detecting faint signals sent from the edge of our observable universe. They are schematized in Fig 1.3 and they include [9]:

- Laser source: It is a Nd:YAG laser which is a common type of high-power infrared laser with a wavelength $\lambda = 1064$nm, it is stabilized in frequency, direction and intensity.
- Fabry Perot cavities: They are cavities bounded by partially silvered mirrors in each arm, photons make several round trips between these mirrors before leaving the arms. These cavities are effectively increasing the length of the arms, they also serve the purpose of increasing the laser power. The impact of these cavities on the long wavelength approximation is detailed in [10].
- Power Recycling: Placed between the laser and the beam splitter, it increases the laser's power to achieve greater sensitivity.
- Signal Recycling: It is used to maintain a broad frequency response but can also focus its sensitivity around one specific frequency, like the detector bar mentioned earlier.
- Suspension: They are used to reduce the vibrations of the mirrors. The test masses are suspended by quadruple pendulums which passively dampen vibration. This system is also stabilized by an active damping system which senses the seismic vibrations and cancels them out.

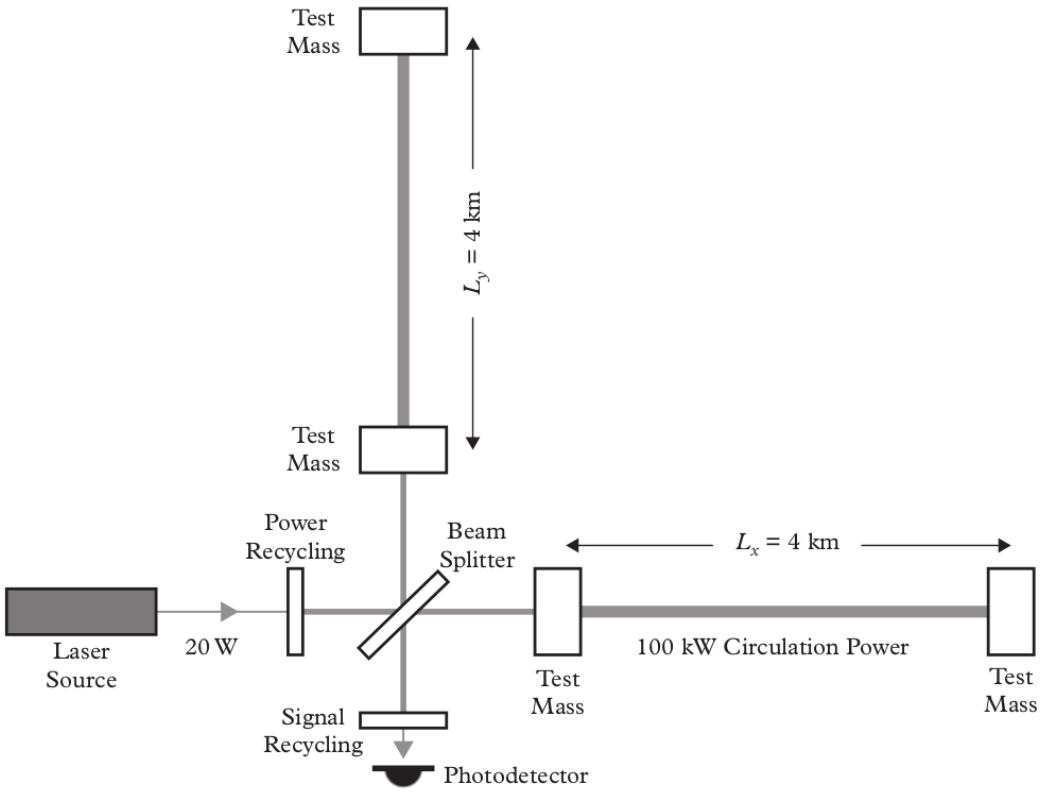


Figure 1.3: A schematic representation of the Advanced LIGO interferometer. The effective power of the laser is indicated at the exit of the laser and in the Fabry-Perot Cavities. [11]

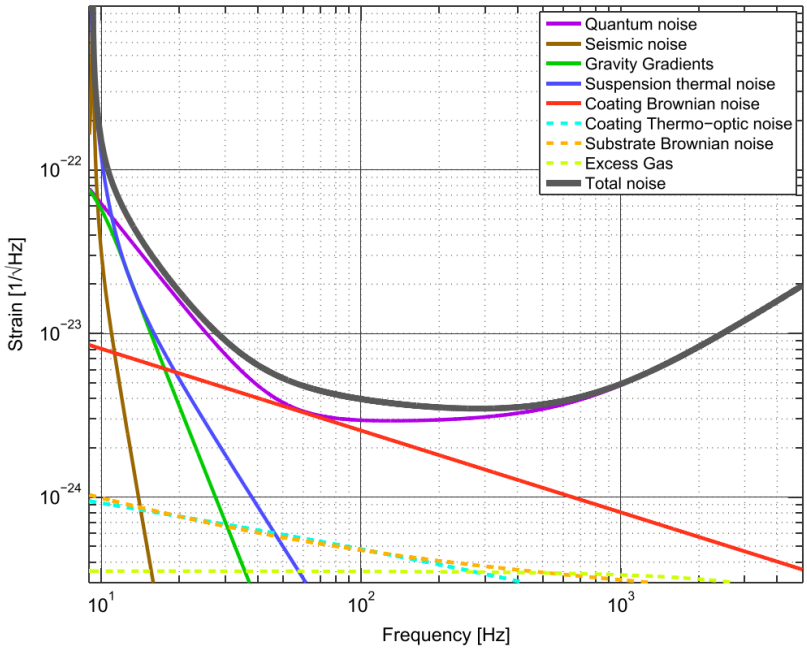


Figure 1.4: The strain noise spectrum of Advanced LIGO and its main noise sources. [9]

These detectors must face many sources of noise, the main ones are presented in Fig. 1.4:

- Quantum Noise: This is a combination of the shot noise due to the random times of the arrival of the photons at the photodiode and the radiation pressure exerted by the photons on the mirrors. Increasing the power of the laser decreases the shot noise but increases the radiation pressure. This is the main limiting noise source, especially at high frequencies.

- Seismic noise: They are the vibrations from the activity around the detector that manages to pass through the suspension system. Since this system is very effective, this noise contribution is negligible beyond the low frequencies.

- Gravity gradients: Variation in the local density around the detector perturb the suspended mirrors (due to clouds or even tumbleweeds [12]). An underground detector would be less affected [13].

- Thermal noise: Suspension thermal noise, coating Brownian noise, coating thermo-optic noise, and substrate Brownian noise all relate to thermal loss in the fibres of the quadruple pendulum suspensions or the coatings of the mirrors.

- Excess Gas: Residual gases inside the arms modify the refractive index through the path of the light beam.

These sources of noise are the main contributors to the background we currently observe, they are present all the time but can vary in intensity. We call the background non-stationary, non-Gaussian and coloured, meaning that it can vary over time, it is not as well-behaved as Gaussian noise, and it varies across frequency. These detectors also suffer from transient noise artefacts which are called glitches and result from coupling within the various instruments of the detector.

Unlike traditional electromagnetic astronomy, where a telescope can be pointed at a specific region in the sky, GW interferometers are omnidirectional, meaning that signals are received from every direction in the sky. More specifically, they are called quasi-omnidirectional since they are less sensitive in certain directions. This is because the two arms have to be stretched by a different amount to produce a detectable phase shift. The antenna pattern for each polarisation is shown in Fig. 1.5, this shows that a single antenna is blind in some directions, proving the need for a multiple detector system. Another benefit of a multi-detector system is the use of time delay between the detections to pinpoint the localisation of the source in the sky. A single detector cannot tell the direction of an incoming signal. However, if another detector detects the same signal, the delay between the two detections can restrict the possible localisation of the source projected onto the sky. The more detectors the more precise the localisation can be.

The modern-day system of detectors consists of the two most sensitive ones, the LIGO interferometers located in the USA (a third LIGO interferometer is under construction in India), followed by the VIRGO interferometer in Italy which is less sensitive than the American ones but has detected signals [15]. And finally, KAGRA, located in Japan.

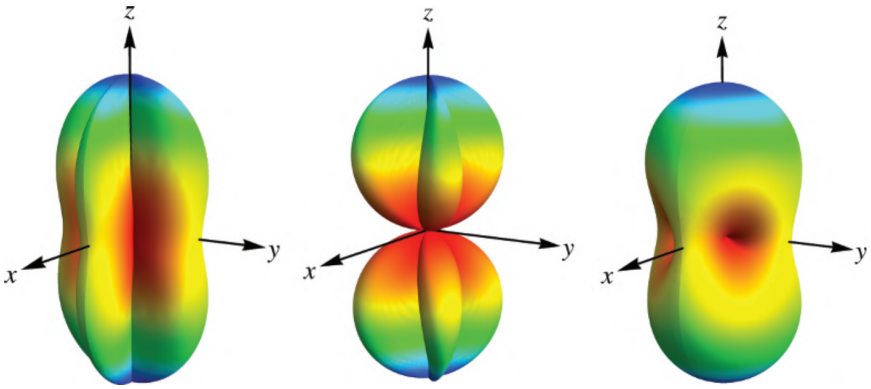


Figure 1.5: The antenna pattern in geocentric coordinates for the plus (left) and cross (middle) polarization, as well as the total receiving pattern (right) which is the root sum square of the two polarisations. These surfaces, as well as the colour, indicate the sensitivity of an interferometer with its arms along the x and y axis to a gravitational wave coming from a certain direction in the sky. (From [14])

Figure 1.6: The VIRGO observatory, with its two 3-kilometre arms. Located near Pisa, Italy. [16]

Future detectors could be the Einstein Telescope (ET)[13], composed of three two-armed interferometers of 10 kilometres each and the LISA (Laser Interferometer Space Antenna) which would be a GW space observatory composed of three satellites separated by 2.5 million kilometres. Due to its longer arms, it would observe at much lower frequencies ($10^{-4} - 10^{-1}$Hz), enabling new science opportunities such as measuring the Hubble constant [17].

1.3 Sources

General relativity predicts that gravitational waves exist and can travel through space from their source to our detectors on Earth. This section will present the detected astrophysical sources and those that should be detectable in the future.

The emission of gravitational waves is due to the acceleration of a non-axisymmetric distribution of matter (with a non-null quadrupole moment). Due to mass and momentum conservation, the monopole and dipole moment vanish and the first contributing term is the quadrupole moment of the mass distribution. This constrains the possible emitting astrophysical sources: a single rotating black hole would not produce gravitational waves since it is spherical, likewise a perfectly spherical supernova, even though violent, would still not generate gravitational waves.

We can characterise a GW by a quantity named h, called the strain, which is the fractional deformation of an object of length L caused by the GW

$$h \approx |h_{\mu\nu}| \approx \frac{\Delta L}{L}. \tag{1.19}$$

We can estimate the strain measured on Earth produced by a distant source using this relation that relates h to the energy radiated per unit time [14]

$$\frac{c^3}{16\pi G}|\dot{h}|^2 = \frac{1}{4\pi d^2}\dot{E}. \tag{1.20}$$

With c being the speed of light, G the gravitational constant, d the distance to the source, $\dot{E}$.

Supposing a highly energetic event, with an energy of about a thousandth of the mass of the Sun ($E \approx 10^{-3} M_\odot c^2$), located in the nearest galaxy cluster, the Virgo Cluster ($d \approx 15$ Mpc), that has a timescale τ and a signal of frequency f we can use

$$\dot{E} \approx \frac{E}{\tau} \quad \text{and} \quad \dot{h} \approx 2\pi f h, \tag{1.21}$$

to get an idea of the typical strain of such events, expressing f in kilo hertz and τ in milliseconds we get

$$h \approx 5 \times 10^{-22} \left(\frac{E}{10^{-3} M_\odot c^2}\right)^{\frac{1}{2}} \left(\frac{\tau}{1\,\text{ms}}\right)^{-\frac{1}{2}} \left(\frac{f}{1\,\text{kHz}}\right)^{-1} \left(\frac{d}{15\,\text{Mpc}}\right)^{-1}. \tag{1.22}$$

Using 1.19, a one-kilometre object on the surface of the Earth would be stretched by about a thousandth of a fermi (1 fm = 10^{-15} m, which is about the size of a proton).

Detecting such waves seems impossible but the advanced laser-interferometers described in the previous section have overcome this challenging task.

To get an idea of the typical mass of the most easily detectable sources we start with the orbital frequency of a gravitationally bound system of mass M and size R [14]

$$f \approx \frac{1}{2\pi}\left(\frac{GM}{R^3}\right)^{\frac{1}{2}}. \tag{1.23}$$

The radius of a non-spinning BH is the Schwarzschild radius, $R_{BH} = 2GM/c^2$ so

$$f_{BH} \approx 10^4 \left(\frac{M_\odot}{M} \right) \text{ Hz.} \tag{1.24}$$

Current detectors are the most sensitive between 100 to 1000 Hz, which means that a compact system of two BH of 10 to 100 $M_\odot$ would be the most easily detectable.

The first detected source of GW was the merging of two inspiralling black holes in 2015, with masses of 36 and 29 $M_\odot$, the resulting merged BH mass is 62 $M_\odot$, meaning that 3 $M_\odot c^2$ have been radiated through GW. The event happened about 430 Mpc away, was measured for about 0.1 seconds, ranged from 35 to 250 Hz and reached a strain of 10^{-21} [11]. Many binary black hole events have been detected since [18]. Then, merger events of a black hole and a neutron star [19] or two neutron stars have been detected [20]. Every observed GW event are compact binary coalescence.

Other astrophysical sources include continuous GW emitters such as single rapidly spinning neutron stars presenting a "mountain" on their surface, the height of these bumps is on the order of the millimetre [21] for a star radius of about 10 km. The fastest spinning neutron star reaches 716 Hz [22], potentially emitting GWs at twice that frequency. Another type of GWs are stochastic signals. The stochastic background of GWs is an analogue to the cosmic microwave background (CMB), it permeates the universe and carries insightful information about its content and cosmology. It could include phenomena ranging from the sum of unresolved astrophysical sources to inflationary effects occurring just after the Big Bang [23].

The last type of GWs are the burst signals, they can be either second-long or minute-long. Core-collapse supernovae can produce second-long burst signals [24]. This work focuses on the search for minute-long burst signals, which can be generated by many astrophysical processes. A representation of their signature in a spectrogram can be seen in Fig. 1.7 and their physical meaning is detailed below.

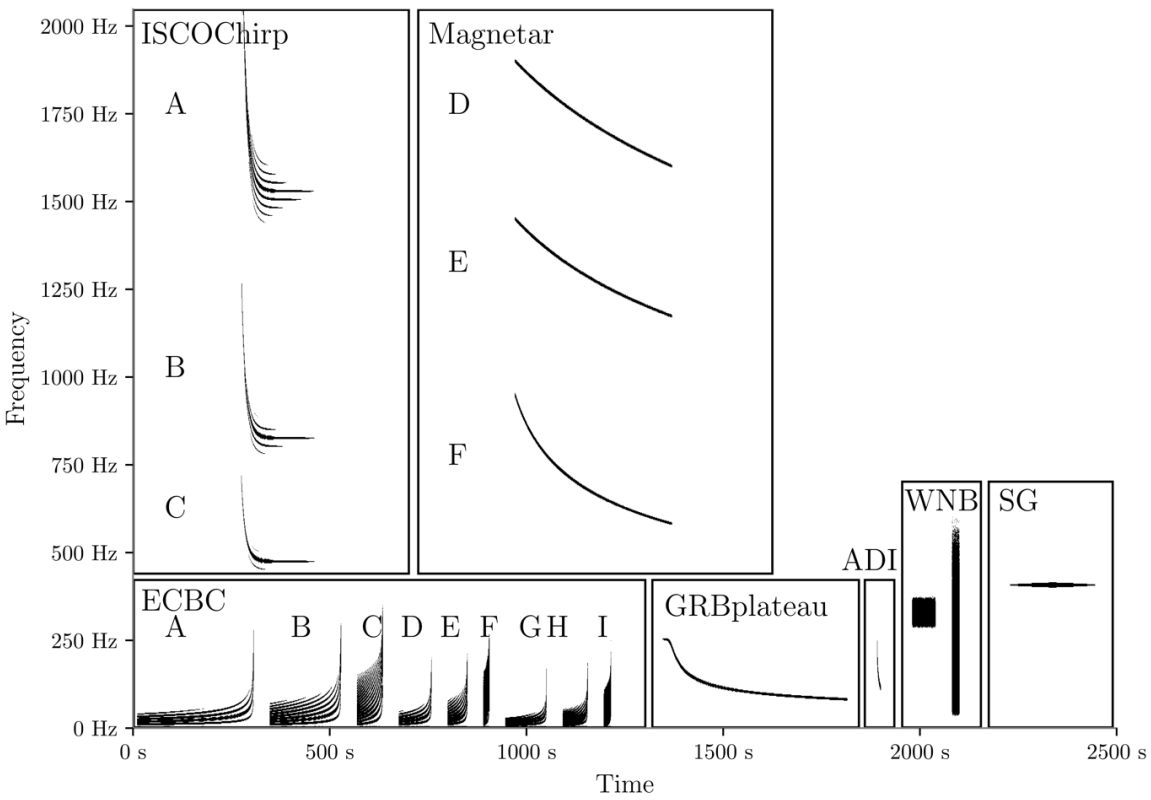


Figure 1.7: A spectrogram showing the signature of several astrophysical and ad-hoc signals [25]. The different letters correspond to different parameters. The detail of every given signal is in the text.

- ISCOChirp: Core collapse supernovae (CC-SNe) are extremely energetic events happening at the end of the lifetime of a massive star. A star, over its lifetime, fuses heavier and heavier atom cores, starting from hydrogen into helium and ending with the silicon-burning process that turns silicon into iron and nickel. The elements inside the star organize themselves in layers, with the lighter elements on top. The fusion of elements lighter than iron releases energy, this heat provides inner pressure to the star, preventing it from collapsing against its gravity. Once the star has reached the creation of an iron core, it can no longer create enough internal pressure to hold against its weight. The iron core under extreme pressure from the outer layers collapses into a neutron star or a black hole, while the outer layers are ejected outwards. Type Ib and Type Ic are CC-SNe that lack hydrogen (Ib) or hydrogen and helium (Ic), these outer layers have been expelled by strong stellar winds or the interaction with a companion star.

  In these systems, if the remnant of the CC-SNe is a BH, the expelled matter can fall back towards the BH, in the accretion disk. The interaction between the rotation of the BH and the matter at the Innermost Stable Circular Orbit (ISCO) may produce gravitational waves. The frequency evolution of the orbital frequency at the ISCO can be expressed by

  $$f_{ISCO}(t) = f_1 + (f_0 - f_1)e^{-at/T}, \tag{1.25}$$

  an exponentially decreasing function of time, where f_0 and f_1 are the starting and ending frequency, T is the total duration, and a a dimensionless coefficient. Harmonics are produced by multipole mass moments.[26]

- Magnetar: The remnant of a binary NS merger can either be a black hole or another neutron star. In the case of a neutron star remnant, if the final mass is too high it will collapse in seconds into a BH and if the mass is low enough the remnant will be stable. Magnetic field amplification during the merging phase can lead to a magnetar, a neutron star with a powerful magnetic field. The intense magnetic field can distort the newly born star into a prolate ellipsoid that could emit GW during its spindown [27].

- ECBC: Eccentric Compact Binary Coalescences, these signals are similar to CBC, which are well modelled and searched mainly through matched filtering. Eccentric coalescence events can present harmonics due to the eccentricity and if the two objects have different masses [28]. Lower mass binary systems emit at higher frequencies, for modern detectors this means that they spend more time in the high sensitivity frequency range, meaning that we would detect them as long-duration signals. Matched filtering searches require a bank of templates and are computationally heavier for longer signals. Unlike CBCs, ECBCs are not mainly searched through matched filtering since they are longer and their eccentricity complicates the generation of templates.

- GRBplateau: Gamma ray burst (GRBs) are burst of electromagnetic radiation at high energies observed isotropically in the sky. They have been classified in two subtypes, short GRBs and long GRBs. Long GRBs have been associated with core-collapse supernovae [29]. As stated earlier, CC-SNe can produce a neutron star, or even a magnetar. The X-ray light curves of long GRBs present a steep descent, followed by a plateau, and end with another steep descent. The presence of a plateau can be explained by energy being injected continuously with progressively reduced activity, for example by the magnetic dipole losses of the inner magnetar. However, this energy could also be provided by the emission of GW due to non-axisymmetric deformations of the magnetar [30].

- ADI: Accretion Disk Instabilities. Long GRBs can leave a black hole surrounded by an accretion disk. Turbulence in this torus of matter coupled to the spin of the black hole at its centre could produce GWs [31].

- WNB: This is not an astrophysical signal, it is an ad-hoc signal used to test the detection pipelines, in particular, it is a White Noise Burst, a signal which has the same power across a certain range of frequencies.

- SG: Same as WNB, Sine Gaussian, a sine wave of constant frequency with an amplitude modulated by a Gaussian function across time.

Chapter 2

Data analysis

The gravitational wave interferometers continuously measure a strain that demands analysis. This large amount of data requires signal, image processing, and more recently machine learning. Understanding the workings of those techniques is crucial for this work.

2.1 Machine Learning

Machine learning problems can be categorised into two groups: supervised learning and unsupervised learning.

Supervised learning is, given a database of inputs and outputs, finding a function that predicts the output of a given input. This is formally defined as follows: from a learning sample $(x_i, y_i | i = 1, ..., N)$ with $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$, find a function $f : \mathcal{X} \longrightarrow \mathcal{Y}$ that minimizes the expectation of some loss function $\ell : \mathcal{Y} \times \mathcal{Y} \longrightarrow \mathbb{R}$ over the joint distribution of input/output pairs: $\mathbb{E}_{x,y}\{\ell(f(x), y)\}$ [32].

An example of supervised learning could be to use a database of N galaxies with the x_i being physical measurements of those galaxies and the y_i their morphological classification, labelled by experts. From this database, find a function that automatically labels the morphology of a galaxy given its measurements.

Unsupervised learning is concerned with applications where the database is not labelled. It aims at finding regularities or patterns in the data. For example, one could use the same database of N galaxies without any label to try to recover or discover groups sharing similar morphology.

2.1.1 Clustering

Clustering is an unsupervised learning problem. The goal is to find groups within unlabelled data. There exist many ways to approach this task, we will go through some of

them that will be used in this work. First, we need to define some concepts.

A cluster is a group of data points, the distance between two clusters can be the largest distance between two points in the clusters, the shortest distance or the average distance. The distance between clusters can be measured with the Euclidian distance, the Manhattan distance, or other distances meaningful to our problem.

K-means

There exist many algorithms to find clusters in a dataset, we will start by explaining K-means clustering.

We start by specifying how many clusters we wish to find in the dataset. Then, so-called cluster centres are placed randomly in our input space. Datapoints are assigned to the cluster which corresponds to the nearest cluster centre. We then compute the centroid of every point belonging to each cluster and these centroid become the new cluster centres. We then iteratively repeat this procedure until the clustering stabilises. This process is illustrated in Fig. 2.1.

This final clustering depends on the initialisation of the cluster centres, we can thus rerun this algorithm several times to see the stability of clusters. One drawback is the need to provide the number of clusters in advance, which is sometimes unknown. One can empirically find the number of clusters using the so-called elbow method. Since it is based on the distance to the cluster centroid, this algorithm assumes that clusters are roughly spherical and not elongated.

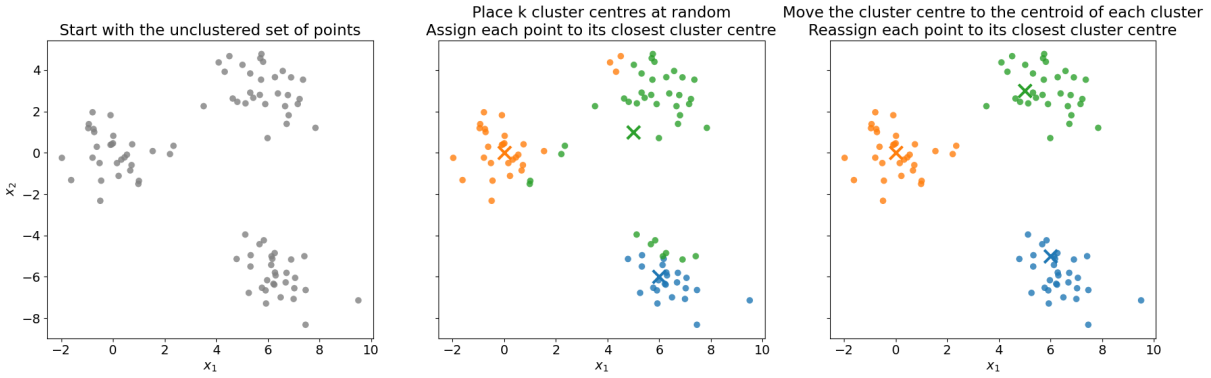


Figure 2.1: A visualisation of the Kmeans algorithm. We start with the unclustered 2D dataset (left). We place at random k cluster centres represented by coloured crosses, each colour corresponding to a cluster. We then assign each point to the cluster corresponding to the nearest cluster centre (middle). We move the cluster centres to the centroid of the previously clustered points. We then reassign each point to its closest cluster centre (right). We set $k = 3$ and for visualisation purposes, the cluster centres are placed so that a reasonable clustering is achieved in one step. In practice, setting k is not trivial and the final clustering depends on the initialisation of the cluster centres.

Hierarchical clustering

This algorithm can be used in two ways, a divisive way and an agglomerative way. We will first explain the most used one which is agglomerative clustering.

We start by assigning a different cluster to each datapoint, we thus have as many clusters as there are datapoints. We then merge the two clusters which are the closest to each other and keep a record of their distance. We repeat the process by iteratively merging the two closest clusters until there is only one cluster remaining. By keeping track of the distances between the clusters that are merged together we can create a visualisation of the clustering process called a dendrogram. An example is shown in Fig. 2.2. Large jumps in the dendrogram show that the two clusters were far apart. We can cut the iterative merging process at a certain step, this translates to cutting the dendrogram horizontally at a certain distance, for example, to keep a certain number of clusters or to fix a maximum distance between final clusters.

Divisive clustering is similar but it starts with one cluster for all the data points and tries every possible division into two clusters and keeps the one that maximises the distance between the two resulting clusters. This comes with a large computing cost, which renders agglomerative clustering more favourable.

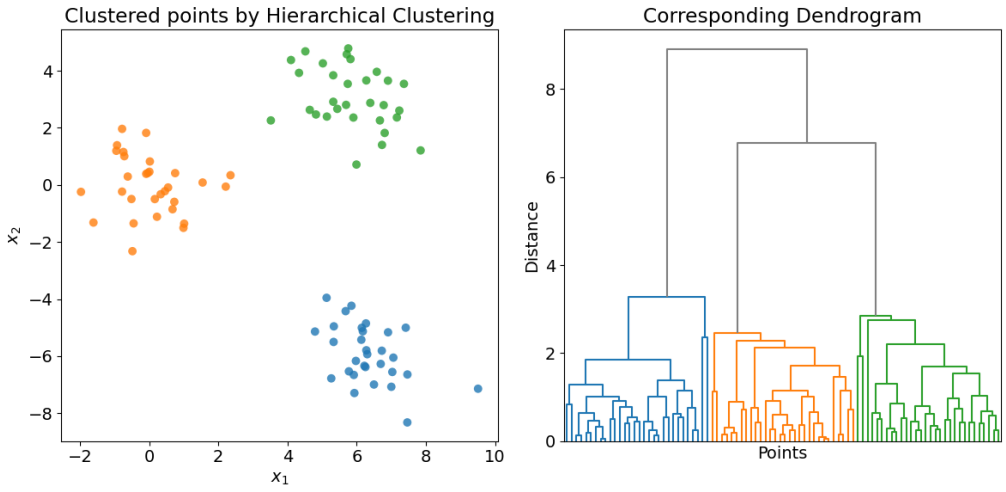


Figure 2.2: A visualisation of hierarchical clustering, with the final clustering (left) and the associated dendrogram (right). The dendrogram starts at the bottom with every point in a different cluster. The closest clusters are iteratively merged (we used the average distance), and we cut the dendrogram at a maximum distance between clusters of 4. This cut can be chosen by looking at the dendrogram, we see that between a wide range of distances (from $\approx$ 3.5 to 6.5) no clusters are merged, which means the clusters formed before this range of distances are far apart.

DBSCAN

Densitity-Based Spatial Clustering for Applications with Noise is an algorithm proposed in 1996 [33] that clusters points together using their density in the input space. It starts

by finding all so-called core points with a minimum of N neighbours below a distance of ϵ. It then assigns a cluster to each group of connected core points. If non-core points have core points in a circle of radius ϵ they are labelled with the cluster of the core point, otherwise they are labelled as noise. Such a clustering is shown in Fig. 2.3.

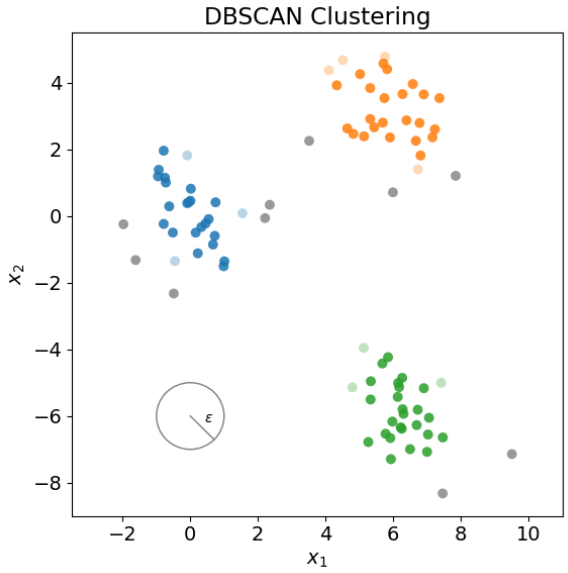


Figure 2.3: This is a visualisation of a clustering achieved with DBSCAN. A circle of radius $\epsilon = 1$ is shown. The minimum number of points in that circle required to be considered a core point is set to 5. Core points are bold and non-core points are light coloured. Noise points are grey.

HDBSCAN

Hierarchical Density-based Spatial Clustering for Applications with Noise is a clustering algorithm proposed in 2013 [34]. It is DBSCAN but with an integration over the ϵ parameter.

We need to specify the parameter N, the required number of samples inside a circle of radius ϵ to form a cluster. It then sweeps over increasing values of ϵ. When ϵ is small, every point is labelled as noise. As ϵ increases clusters appear in regions of N or more samples within a distance ϵ. These clusters grow as ϵ becomes larger, and merge to form larger and larger clusters. Once ϵ has reached a high enough value, all points are grouped within the same cluster.

The final clusters are chosen to be the most stable ones over ϵ. HDBSCAN removes one of the two parameters of DBSCAN, which allows the clustering of varying densities of points. The N parameter is still left to tune.

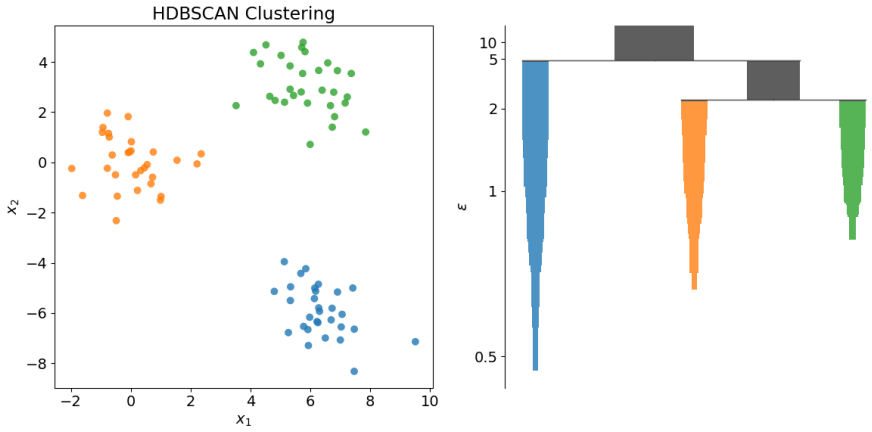


Figure 2.4: The clustering achieved by HDBSCAN (left) with $N = 5$, and the condensed tree provide a visualisation of the clustering process (right). We see that for low values of ϵ there are no clusters, then the the most dense points are labelled as a cluster, and as ϵ increases, two other clusters appear. The orange and green clusters are merged when ϵ gets larger than ≈ 2. All points are associated with the same cluster when ϵ reaches ≈ 5. The width of the tree branches is proportional to the size of the cluster. The final clusters are chosen to be the ones that are stable the longest while considering the number of points in the cluster (this can be seen as choosing the branches that "fill the most ink on the page").

Comparison of the different algorithms

A comparison of the strengths and weaknesses of each algorithm on a more complex and diverse dataset [35] is shown in Fig. 2.5.

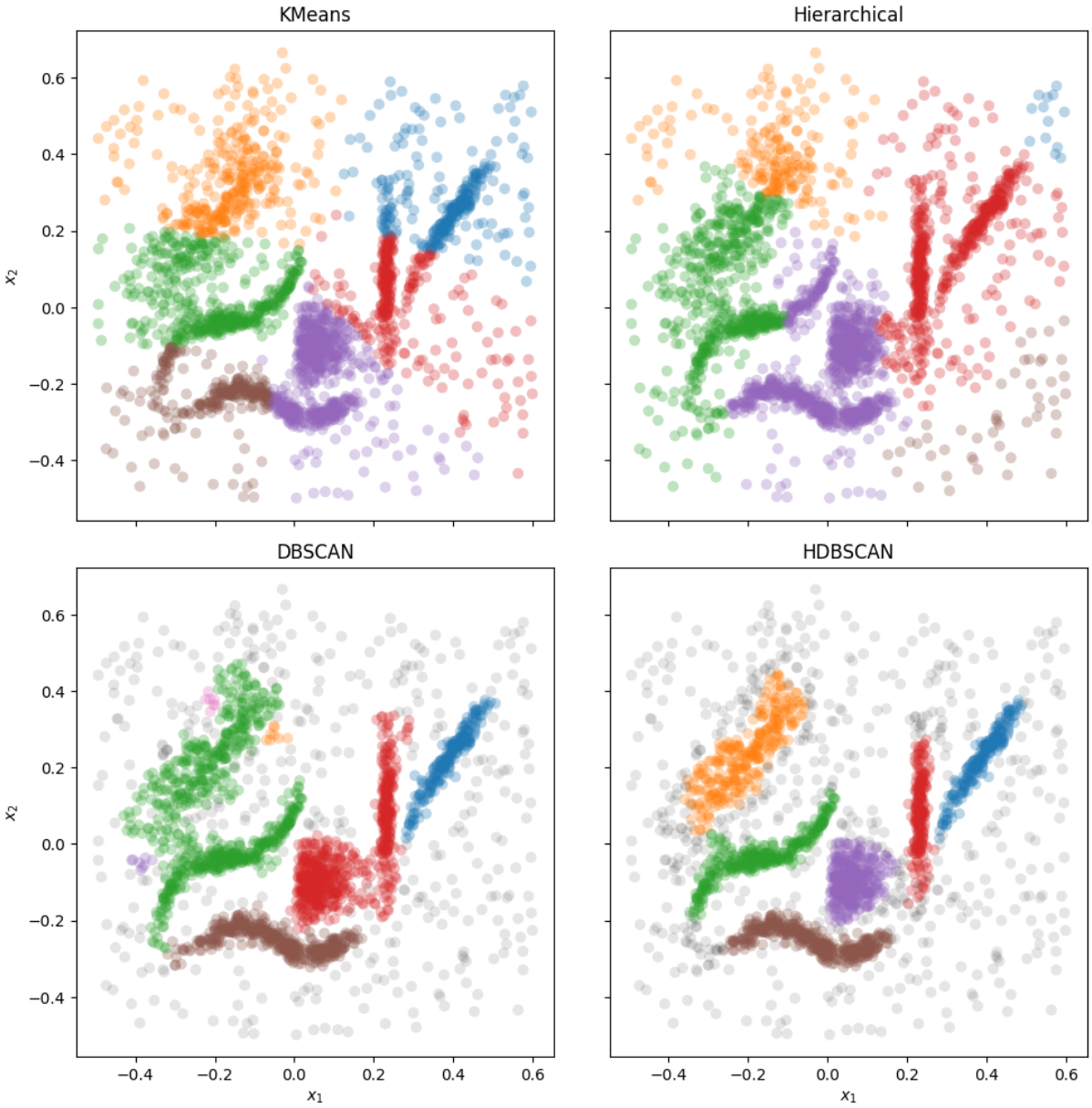


Figure 2.5: A comparison of four clustering algorithms on the same dataset. KMeans is used to find 6 clusters. The Hierarchical clustering uses the average distance and the dendrogram is cut to find 6 clusters. DBSCAN uses $\epsilon = 0.025$ and $N = 5$. HDBSCAN uses $N = 15$. We note that the different shapes present in this data are not well clustered by KMeans and the Hierarchical clustering. We see that the density-based approaches perform better, but DBSCAN, due to not being able to tackle varying densities links what seem to be separate clusters together, it also finds small clusters in pink, purple, and orange. HDBSCAN successfully finds six clusters while labelling the background points as noise.

We conclude that HDBSCAN performs better (on this dataset) than the other algorithms since it can classify points as noise, does not assume any shape for the clusters, and can cluster points of varying densities.

2.1.2 Neural Networks

This section will present models suited for supervised learning tasks, starting from simple fully connected networks and how to train them to more complex convolutional neural networks and their applications.

Fully Connected Network

Neural networks are functions that take several numbers as inputs and return other numbers as outputs. The inner workings of the function are layers of connected neurons. The neurons can be tuned to produce the desired output from the inputs.

The neurons are themselves functions. A neuron with a single input and a single output takes its input which is a number, multiplies it by a weight, adds a bias, and passes the resulting number into a non-linear activation function (some common activation functions are represented in Fig. 2.6b). The output of this function is called the activation of the neuron. The activation a of a neuron that is connected to n other neurons is

$$a = f\left(b + \sum_{i=1}^{n} w_i a_i\right), \tag{2.1}$$

with a_i and w_i being the activation of the i^{th} neuron in the previous layer and the weight linking them, b is the bias and f is the activation function.

The weight and bias are the parameters that are tuned during the training phase. Combining neurons can create algorithms ranging from simple logic gates to complex models provided that the parameters are set correctly or found during a training procedure.

A realisation of a neural network is presented in Fig. 2.6a. The depth is the number of layers, deep learning refers to the study and use of deep neural networks with millions or more parameters.

Training Neural Networks

The parameters of these neural networks are found by finding a minimum in the parameter space of a Loss function over the training set. One must remember that what is minimized is the empirical risk which is an approximation of the true error using the training data. For a regression problem, one can use the Mean Squared Error (MSE) as a Loss function.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2, \tag{2.2}$$

where y_i are the true outputs in the learning set $(x_i, y_i | i = 1, ..., N)$ and $\hat{y}_i$ are the predictions $f(x_i)$ of a model f. For a classification problem, the Cross-Entropy (CE) loss

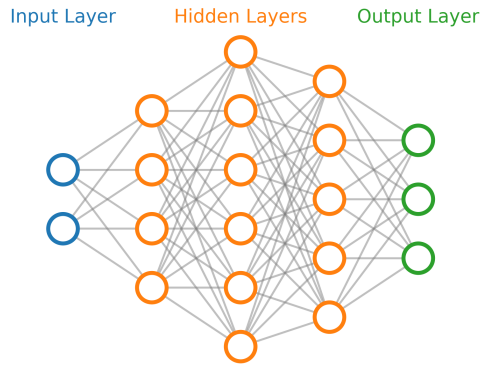


Fig. 2.6a: A schematic view of a neural network architecture. It is composed of an input layer, hidden layers, and an output layer. For this architecture, the depth of the hidden layers is 3, and their widths are 4, 6, and 5.

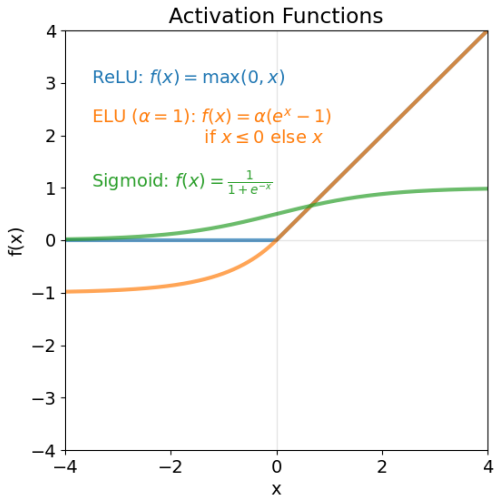


Fig. 2.6b: Illustration of three common activation functions: the ReLU (Rectified Linear Unit), the ELU (Exponential Linear Unit), and the Sigmoid.

function can be used,

$$CE = -\sum_{i=1}^{N} y_i \log(\hat{y}_i), \tag{2.3}$$

where y_i is the true label in a one-hot encoded form and $\hat{y}_i$ is the predicted class probabilities.

Due to the large number of parameters, the parameter space is of high dimension and cannot be sampled finely to find the minimum. Because these losses are differentiable, so as the operations within the network, gradients of the loss function with respect to the parameters can be computed, and are used by optimizers to find a minimum in the parameter space.

A simple optimizer is Stochastic Gradient Descent (SGD) where gradients are computed on a subset B of the dataset (called a batch) and the parameters θ are updated at every step t like so [36],

$$\theta_{t+1} \leftarrow \theta_t - \eta \cdot \sum_{i \in B_t} \frac{\partial \ell_i(\theta_t)}{\partial \theta}, \tag{2.4}$$

where η is the learning rate, a hyperparameter which sets the scale at which the loss landscape is probed, B_t is the set of the indices of the batch t and $\ell_i(\theta_t)$ is the loss of the sample i computed with the parameters θ_t. The size of the subset B is called the batch size and is a hyperparameter controlling the amount of randomness introduced in the computation of the gradients compared with the gradient computed on the full dataset. Introducing randomness can be beneficial for getting out of local minima.

One can refine SGD by adding inertia from the previous steps to the next steps, this is called SGD with momentum. The learning rate can also be adapted as a function of the gradients of the previous steps, this is called adaptive learning rate and one of the

most common optimizers, Adam (Adaptive moment estimation) implements both of these improvements.

Convolutional Neural Networks

As stated previously, neural networks take as inputs a list of numbers. This means that we can give it an image as input but only if we flatten it into a list of values. By flattening, we lose any spatial information previously contained in the image. Convolutional Neural Networks are a variation of the traditional neural networks which use the convolution operation to extract spatial information in the images. The convolution operation is illustrated in Fig. 2.7. The role that played the weights in the fully connected neural network is now played by the values inside the convolution kernels. Traditional fully connected layers are replaced by convolutional layers that can output several feature maps using several kernels. CNNs are composed of convolutional layers, but also max-pooling

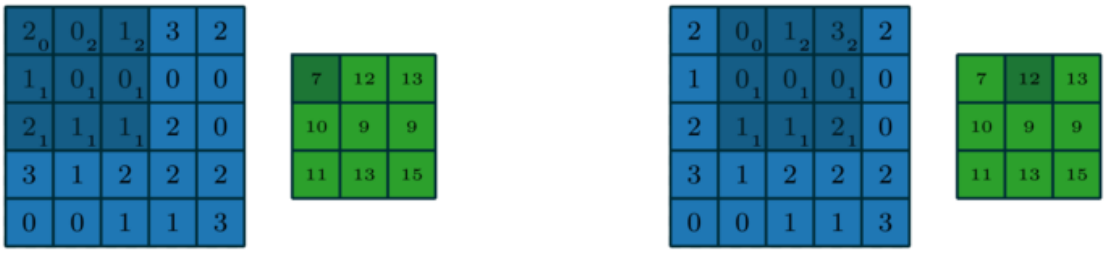


Figure 2.7: An illustration of the convolution operation of an input of size 5x5 by a kernel of size 3x3, with a stride of 1. Figures adapted from [37] using [38].

layers, and they are illustrated in Fig. 2.8. These layers are used to reduce the size of their input. Another type of layer is the transposed convolution layer, shown in Fig. 2.9

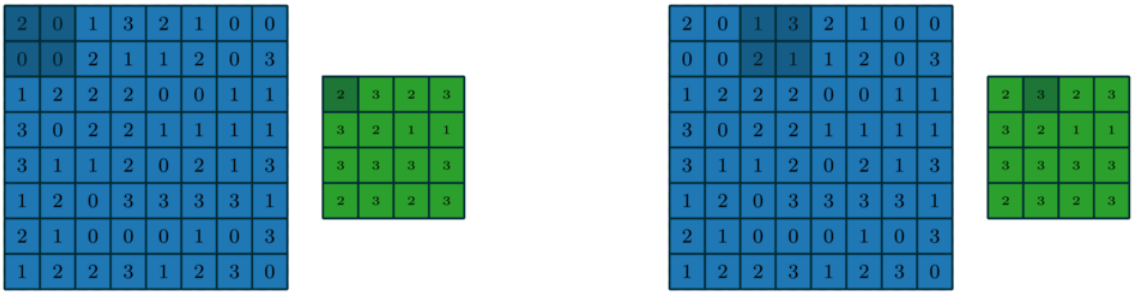


Figure 2.8: An illustration of the max pooling operation of an input of size 8x8 by a kernel of size 2x2, with a stride of 1. Figures adapted from [37] using [38].

which, contrary to the max pooling layer, increases the size of their inputs. Convolutional neural networks have proven to be powerful algorithms for image classification [39], object detection [40], and image segmentation [41].

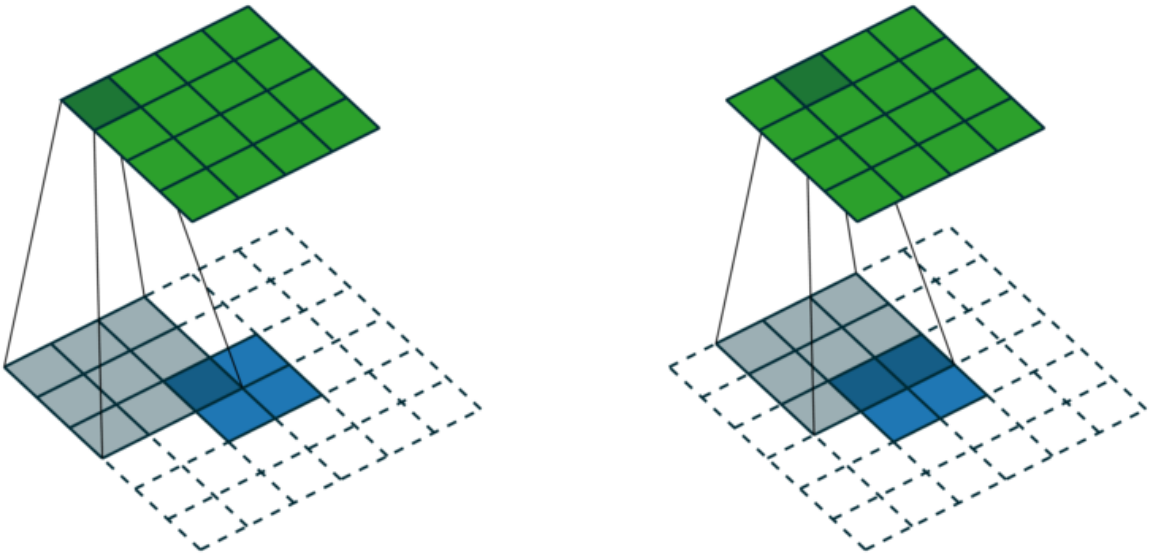


Figure 2.9: An illustration of the transposed convolution operation of an input of size 2x2 by a kernel of size 3x3, with a stride of 1. This is equivalent to a convolution of the 2x2 input with a padding of 2 with another kernel as shown in [37]. Figures adapted from [37] using [38].

The model presented in [41] will be of interest to this work. It is called the U-net due to its architecture shown in Fig. 2.10. It was first proposed for the segmentation of biomedical images, it is composed of two parts, an encoder part which uses convolutions and max-pooling to reduce the sizes of the input image to low-resolution feature maps and a decoder part which upscales those feature maps with transposed convolutions. Feature maps from the encoder part are concatenated with feature maps from the decoder part to retain precise spatial information.

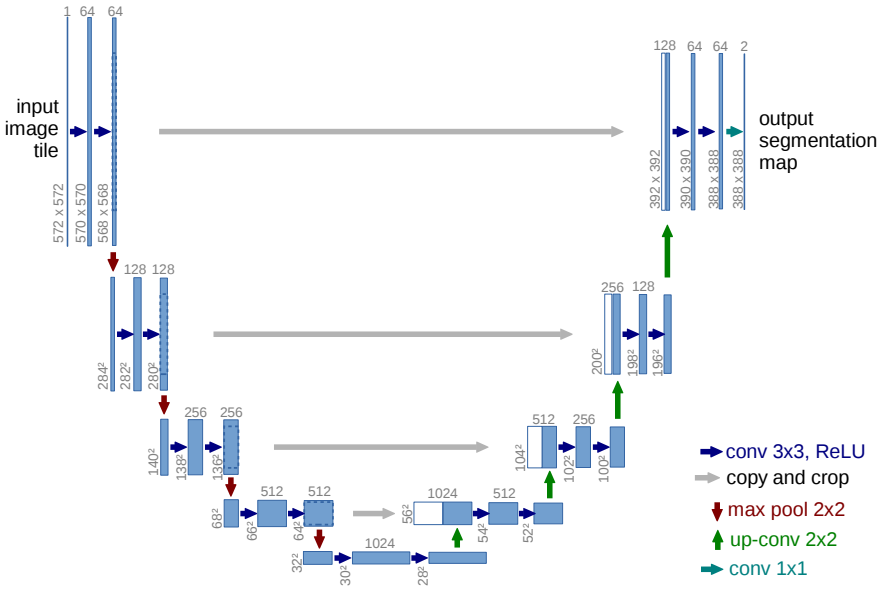


Figure 2.10: The architecture of the U-Net [41]. The input is a single 572x572 image which is convoluted to produce 64 feature maps, these feature maps are themselves convoluted and max-pooled while passing through activation functions multiple times until they reach the number of 1024 32x32 feature maps. They are then upscaled in the decoder part and concatenated with previous feature maps from the encoder part to produce a segmentation map.

2.2 Gravitational Wave Data Analysis

Gravitational wave laser interferometers, when operational, constantly output a strain over time. One can make an analogy with sound to better grasp the difficulty of finding signals in this data. Microphones also record a strain over time, so one can convert the output of the interferometer to sound, and play it through speakers, GW signals sound like pops or small chirps buried into a loud noise, (examples of such audio files can be found in [42]). Due to the large amount of data, the wide parameter space covered by potential signals, and their low signal-to-noise ratio, it is necessary to develop several pipelines to process and detect potential signals.

When we know precisely what our source sounds like we can use this knowledge to pierce through the background noise, but when no exact model is known, we must listen for when we hear something above the noise.

This translates to the GW world by the modelled vs unmodelled searches. Modelled searches use matched filtering with a bank of simulated waveforms to detect low SNR signals. Unmodelled searches have to use other means to detect signals. This work focuses on the unmodeled search for long-duration GW signals.

2.2.1 Unmodelled long-duration GW pipeline

The mathematical developments are adapted from [43] and [6]. One way to analyse a signal is to do a Fourier transform of it, which provides the amplitude at each frequency of the signal. The Fourier Transform of a function $x(t)$ is given by:

$$\tilde{x}(f) = \int_{-\infty}^{\infty} x(t) e^{-i2\pi ft}\, dt. \tag{2.5}$$

In practice our signal is a time series of a finite length with N samples at times $t_n = t_0 + n \times \Delta t$, we thus use the discrete version

$$\tilde{x}(f) = \sum_{n=0}^{N-1} x(t_n) e^{-i\frac{2\pi}{N} fn}. \tag{2.6}$$

Repeating this Fourier transform for several segments gives a view of the evolution of the different frequency components of the signal. Putting each of these segments together gives a spectrogram, a map of the evolution of the intensity across frequencies of our signal.

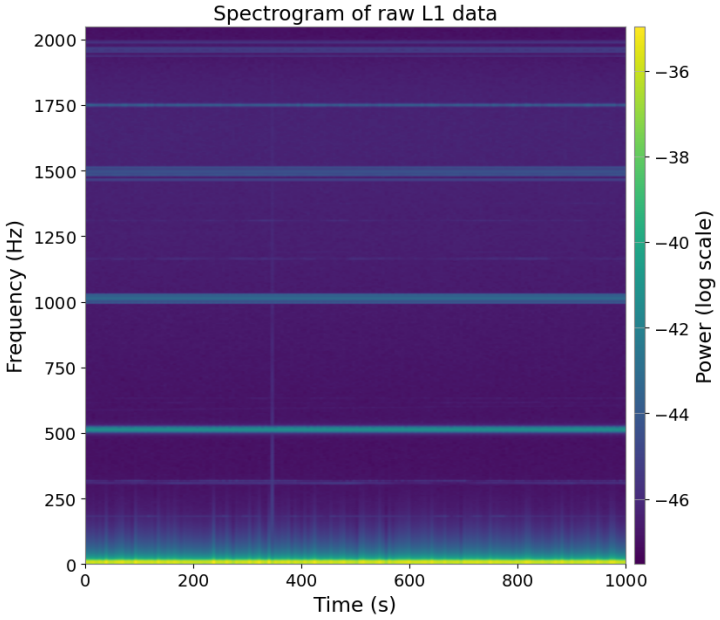


Figure 2.11: A 1000s second spectrogram of data from the Ligo Livingston (L1) interferometer. With a time resolution of 6 seconds and a frequency of resolution of 4 Hz. It is dominated by low-frequency noise and constant-frequency lines.

Fig. 2.11 shows such a spectrogram for data from a LIGO interferometer. We notice that the spectrogram is dominated by low-frequency noise and lines of constant frequency that last for the whole duration of the segment, these come from violin harmonics which are resonant frequencies in the silica fibre by which the mirrors are suspended, or even power lines at the harmonics of 60 Hz, the frequency of the alternating current in America. One can get rid of those artefacts by using a spectral whitening technique, the goal is to convert our noise into white noise. This means that we will render the intensity of

the noise across frequency constant, this is called white noise in analogy to white light. Technically, we normalise our signal by the noise spectrum of our detector shown in Fig. 1.4. To formally explain this procedure we must define the correlation between two signals $x(t)$ and $y(t)$ as

$$r_{xy}(t) = \int_{-\infty}^{\infty} x(\tau)y(t+\tau)\,d\tau, \tag{2.7}$$

and the cross-spectral density being the Fourier Transform of this correlation

$$S_{xy}(f) = \int_{-\infty}^{\infty} r_{xy}(t)e^{-i2\pi ft}\,dt. \tag{2.8}$$

The Power Spectral Density (PSD) is the Fourier Transform of the auto-correlation of a signal S_{xx}. Finaly, a whitened signal is $\tilde{x}/\sqrt{S_{xx}}$.

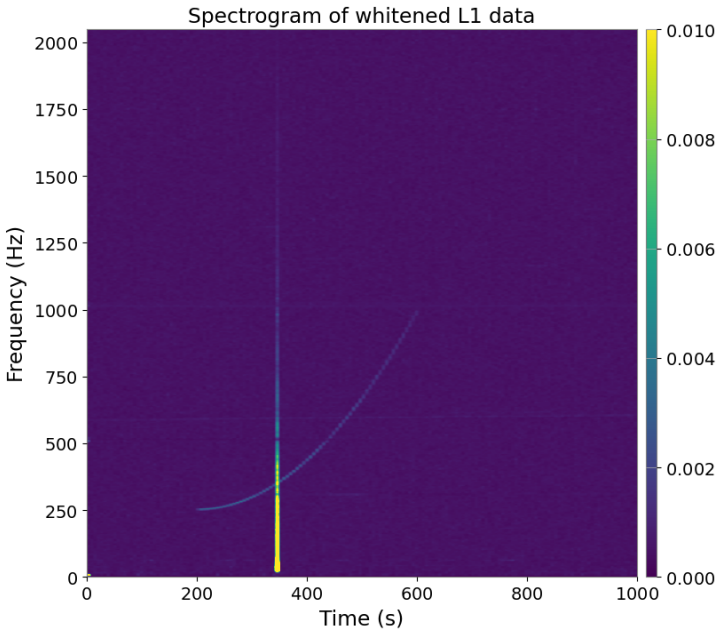


Figure 2.12: A spectrogram of the same data as in 2.11 but whitened. A wide-band glitch dominates the spectrogram but a signal is visible from 200 s to 600 s in time and 250 Hz to 1000 Hz in frequency. We still see the lines at constant frequencies that were not entirely removed by the spectral whitening procedure.

Fig. 2.12 shows the spectrogram of the same segment that has been whitened. What we see now is a loud, transient, wide-band signal, it is a glitch. Glitches come in a variety of shapes, and projects linking citizen science and machine learning like Gravity Spy [44] aim at classifying the glitches. Examples of different classes of glitches are shown in Fig. 2.13. We notice that these glitches are often short-lived, typically less than a few seconds [45], so using our time resolution they appear as vertical bands.

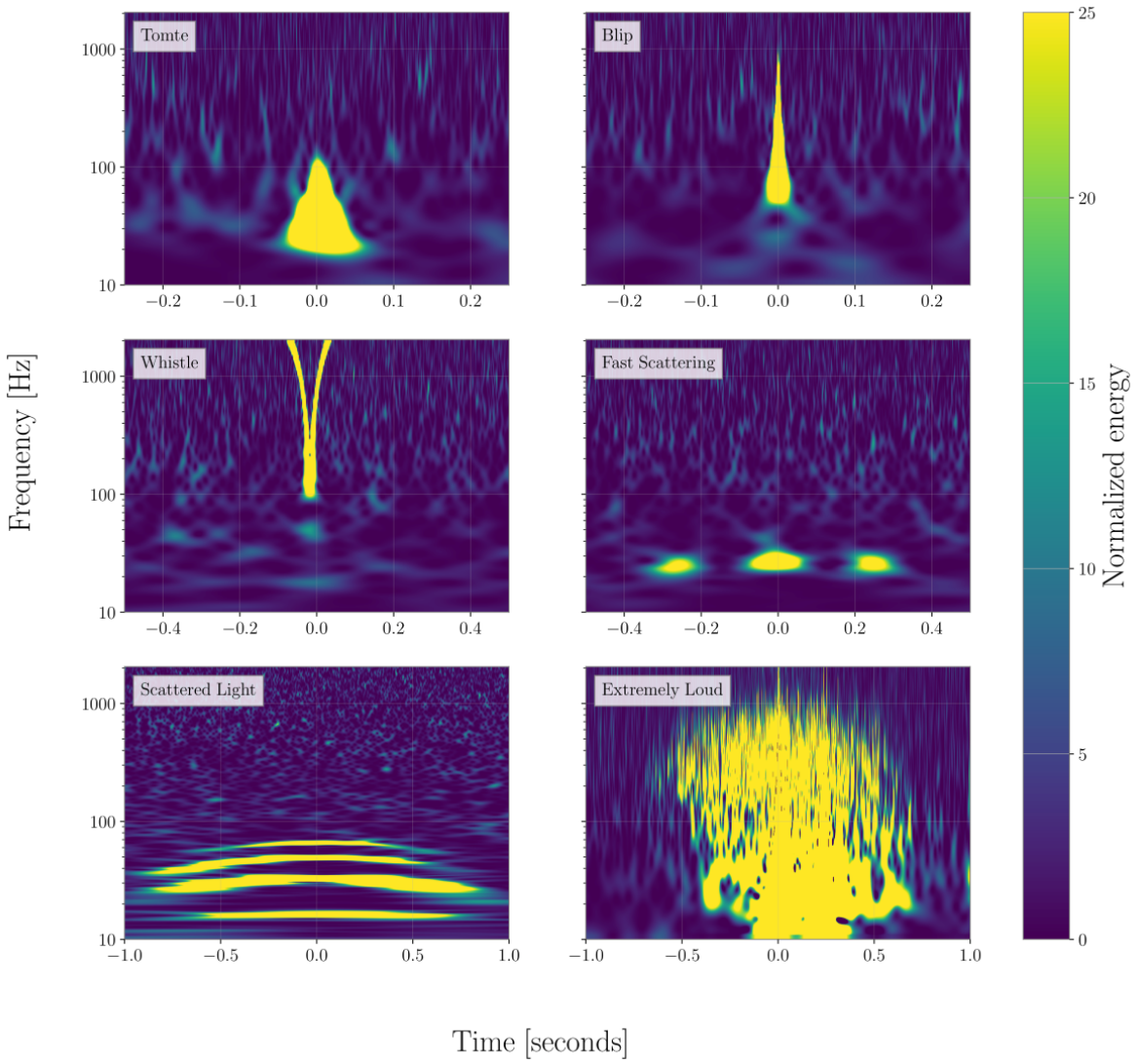


Figure 2.13: Spectrograms of interferometer data containing glitches of different classes. Figure taken from [45]. The different classes relate to different features of the glitches, such as their duration or morphology in the spectrogram.

One way to get rid of glitches is to use another detector, since the glitches are due to local noise happening at the detector, one glitch in Hanford should not happen exactly at the same time as another glitch detected three thousand kilometres away in Livingston, the location of the second LIGO interferometer. More precisely, if each detector has a glitch rate of R, and the glitches have a duration T, the rate of coincident glitches is R^2T. We thus define the coherence between two signals as

$$C_{xy}(f) = \frac{|S_{xy}(f)|^2}{S_{xx}(f)S_{yy}(f)} \tag{2.9}$$

We can thus make a spectrogram of the coherence between the two detectors. This is shown in Fig. 2.14. We see that the loud glitch goes away and we see a signal which was faint in the previous spectrograms now appear clearly in the coherence spectrogram. This is another benefit of the coherence operation, if signals happen during the same time-bin

in each detector then it gets amplified. Because the travel time between both detectors is about $10\mu s$, while our time resolution is 6s, any physical signal will be captured inside the same time-bin.

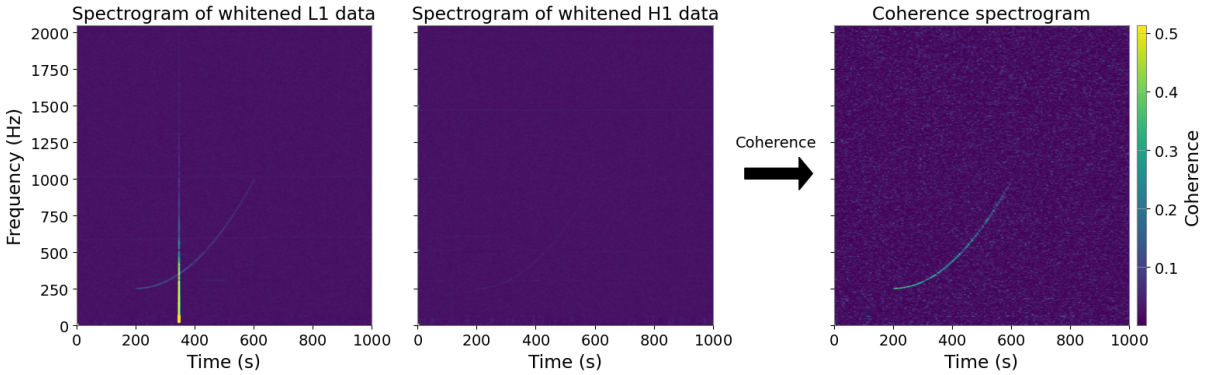


Figure 2.14: Coherence spectrogram of data from the H1 and L1 detector located in Hanford and Livingston. The glitch present in the L1 spectrogram disappears because no glitch happened at the same in H1 data. The signal is now clearly visible.

We notice that there are still remnants of the whitening procedure, the horizontal lines have not been totally removed, we thus apply a normalisation across time of all frequency bins so that they sum to one, this is called normalisation. The resulting spectrogram is shown Fig. 2.15.

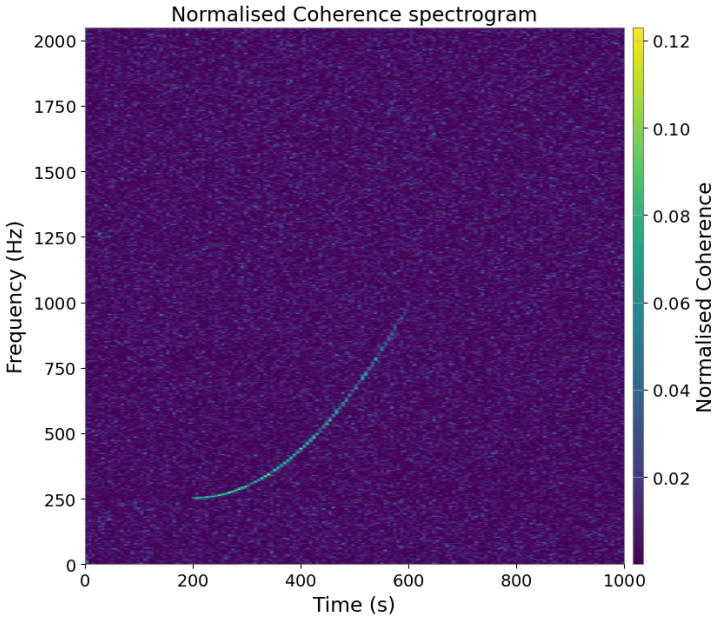


Figure 2.15: Normalised coherence spectrogram. The normalisation across time removes the lines remaining from the spectral whitening procedure. The spectrogram now only contains noise and a visible signal.

As no long-duration signal has presently been observed, the chirp seen in the spectrograms is manually injected. The challenge of our long-duration unmodeled GW burst

searches can be stated as follows: find potential traces of signals in coherent spectrograms, while disregarding glitches.

One way to detect such signals would be to use classical computer vision to seek patterns in the spectrogram, which are images. One could use the Hough Transform to detect lines like in [46], edge detection algorithms like the Sato Filter [47] or other more complicated algorithms but they have the disadvantage that they are slow.

A more powerful method would be to not hard-code an algorithm that detects those signals but to let an artificial intelligence train to recognize such signals in any given image. The use of artificial intelligence to detect objects in images is well researched as stated in section 2.1.2, and this exact problem of detecting chirping signals in a noisy environment polluted by transient artefacts has been researched in a different context but surprisingly similar way when searching for dolphin chirps in the noisy waters of South Korea's islands that are hidden by the clicks the claws of the snapping shrimps [48].

Knowing that the algorithms capable of training to achieve our task exist, having the computational power to train one, there is still one issue: the need for a dataset. Indeed modern deep-learning methods require large datasets of images on which to train, this renders the use of the templates in fig 1.7 impossible since they would take too long to generate and they are not numerous enough to provide a good dataset on which to learn. The next section will show how this problem was circumvented to allow the use of deep-learning methods inside a long-duration GW pipeline.

2.2.2 ALBUS

The problem of generating a dataset of spectrograms with injected signals on which to train a large neural network has been solved by using generic chirp-like signals. This was done using the scipy signals library [49] which contains linear, quadratic, hyperbolic, and logarithmic signals. These signals are represented in a spectrogram in Fig. 2.16.

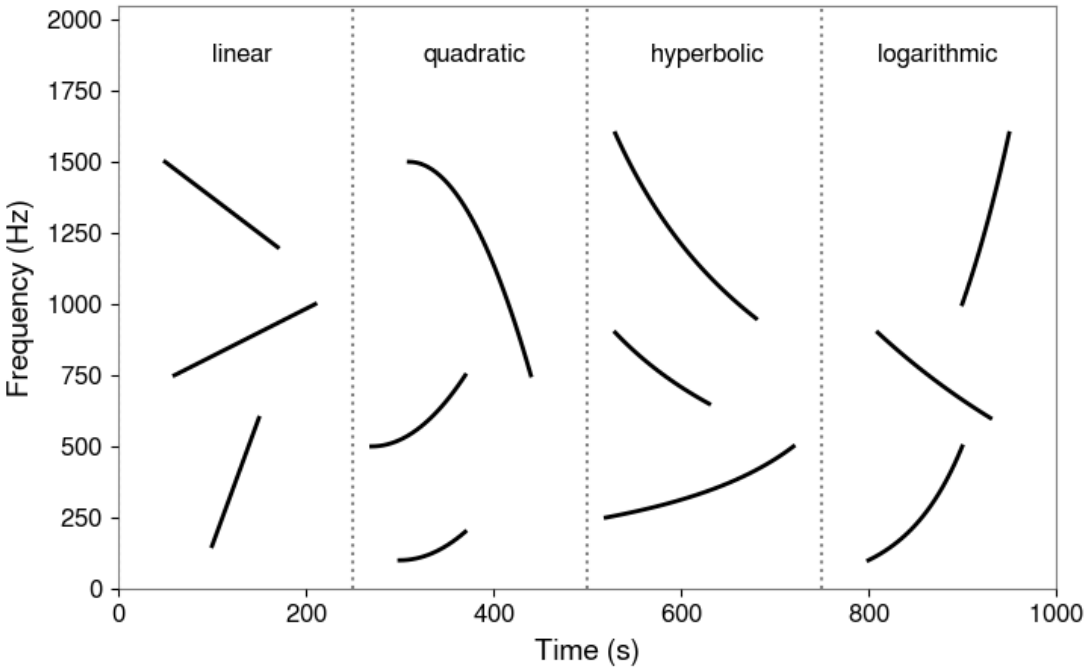


Figure 2.16: A spectrogram showing the signature of typical signals generated to create the dataset on which ALBUS is trained. The different frequency evolutions are highlighted. The parameters (starting and ending frequency, duration) have been chosen arbitrarily.

This idea led to the creation of ALBUS (Anomaly detection for Long-duration BUrst Searches) [50], a CNN that acts as a non-linear noise removal filter for spectrograms of gravitational wave data. The goal of CNN is to take as input a noisy spectrogram possibly containing signals and glitches, and output two spectrograms, which respectively contain the reconstructed signals and glitches without noise. To achieve such a behaviour we need to create a dataset of input of expected outputs and to choose an architecture.

Dataset

With the multiple publicly available deep-learning models and the relatively easy access to GPUs on which to train them, having a good dataset becomes the last key ingredient to successfully train a good algorithm.

The dataset of ALBUS contains the inputs and the expected outputs, called the target maps. For each input spectrogram, there are two output target maps, one for the signals and one for the glitches. We can subdivide the dataset into four parts, the backgrounds, chirps, glitches, and combined spectrograms, each having its inputs and target maps.

The generation of the inputs starts from coherence spectrograms of non-coincident (at different times) H1 and L1 data. The backgrounds have no further processing, as they are created from non-coincident data we expect no signal to appear in the spectrograms. In the chirp dataset, we inject one chirp of the types presented in Fig. 2.16 per spectrogram

with parameters selected in the parameter space of Table.2.1. For the glitch dataset, we carefully chose the two times of the H1 and L1 data so that one glitch lines up in our spectrogram. The combined dataset is created like the glitch dataset while also injecting one chirp in each spectrogram.

	Range of values
Duration	10 - 500 s
Delay	0 - 950 s
Frequency range	30 - 2000 Hz
Max. Bandwidth	1000 Hz
Frequency evolution	linear, quadratic, logarithmic or hyperbolic
β parameter - Kaiser	1 - 4
Coherent amplitudes	$2, 3, 4, 5, 6, 7, 8, 10, 12, 14, 16, 18, 20, 25, 30$
$N°$ harmonics	1, 2, 3
Frequency spacing	10, 15, 20, 25, 30 Hz
Attenuation coeff.	0.5, 0.6, 0.7, 0.8

Table 2.1: Range of parameters of the injected chirps. Delay is the starting time of the chirp in the spectrogram. The β parameter controls the amplitude evolution of the chirp that is modulated by a Kaiser window. The Coherent amplitude relates to the amplitude of the signal, it is computed as the sum over all the pixels of the difference of a spectrogram with and without the injection. Harmonics are also simulated by injecting chirps at spaced starting or ending frequencies with an attenuated Coherent amplitude.

The generation of the target maps is one of the most crucial elements as this is what the network will learn to output.

The background target maps are simply spectrograms where every pixel is zero, so when we give our network a spectrogram with no injected signals or glitches, it will learn to output blank spectrograms.

The target maps for the chirps are created by applying a threshold at the 99^{th} percentile on the input coherence spectrogram, this provides a binary image that highlights pixels of high coherence on the whole input spectrogram. Then we inject the same signal in no noise, this provides a clean spectrogram of the whole signal, which is thresholded above its mean value, this results in a binary image where the high energy pixels of the signal are highlighted. Then we take the intersection of those two binary images, this keeps only the bright pixels of the input spectrogram that belong to the signal. Finally, we multiply the coherence spectrogram by that binary image and normalise it by dividing it by its maximum value. This process is illustrated for a specific chirp in Fig. 2.17. One must consider this question: the algorithm presented above to generate target maps for chirps takes as input the coherence spectrogram and outputs the highlighted pixels belonging to the signal, why not use this algorithm to detect signals instead of the CNN since they serve the same purpose? The answer is that the target map generation algorithm also takes as input the signal injected in no noise, which we do not have access to in a real-life scenario.

Concerning the glitches target maps, the question asked above becomes relevant since

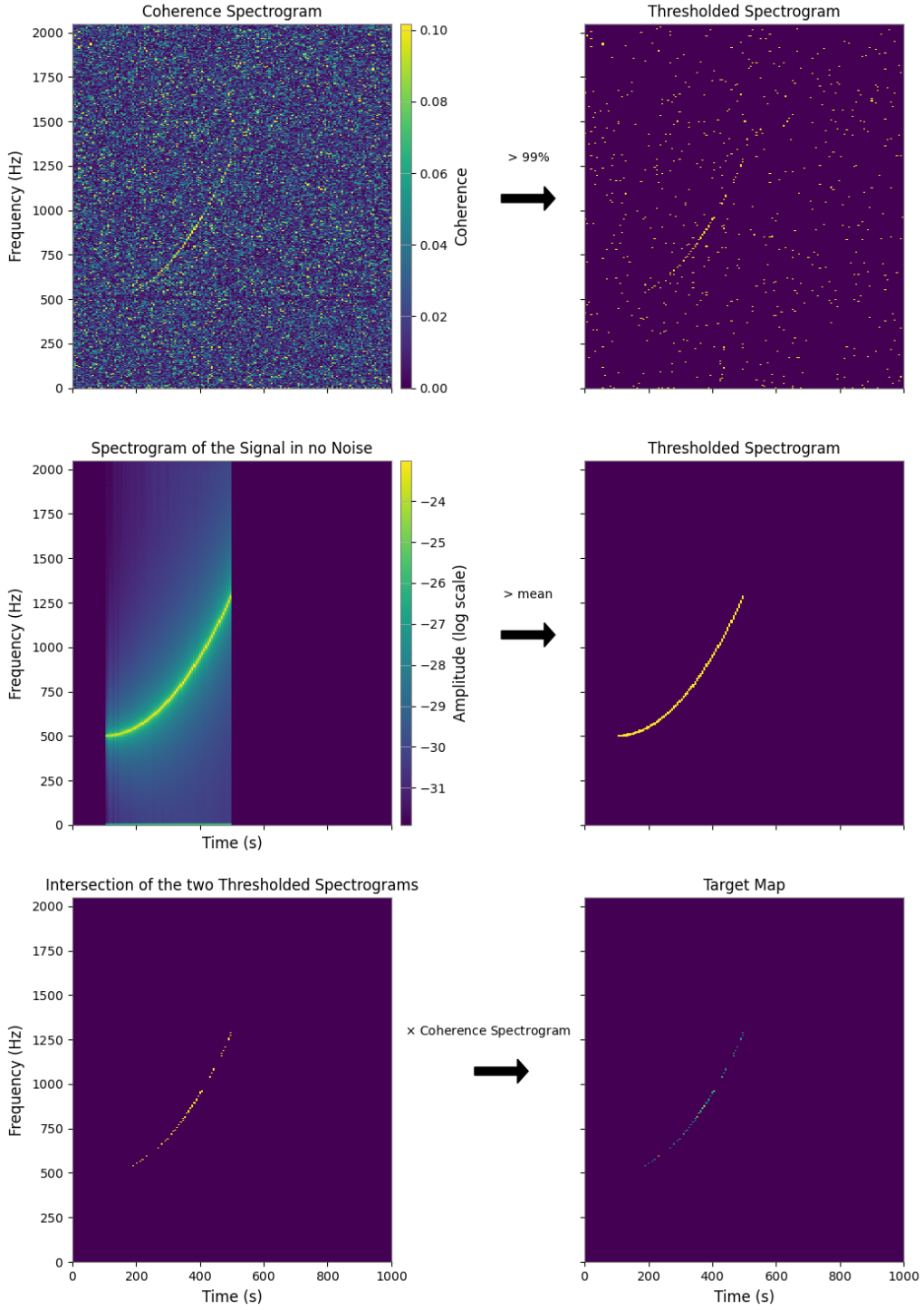


Figure 2.17: A visualisation of the generation of a target map for a chirp. It begins by thresholding the coherence spectrogram above its 99^{th} percentile. Then it thresholds the signal injected in no noise above its mean. Finally, it takes the values of the coherence spectrogram that belong to the intersection of the two thresholded images and divides them by their maximum value so that the values are in the [0,1] interval. The excess power at all frequencies in the duration of the signal is due to spectral leakage. It is an artefact of the discrete Fourier transform which assumes that the signal is periodic. The signal can be tapered by a window function to mitigate these effects.

we do not have access to the glitch injected in no noise because we do not inject the glitches. We thus correlate two noisy strains that contain a glitch. To generate the target maps for glitches we first train ALBUS on chirps only, and the target maps of input spectrograms containing a glitch will be the output of this first version of ALBUS for that spectrogram.

Finally, to create the target maps for a combined spectrogram, one containing a chirp and a glitch, we proceed the same way, we generate the glitch target map before injecting the chirp, and then we generate the chirp target map using the algorithm described above.

The LIGO interferometers are operational during periods lasting several months called observing runs. The dataset is generated using the noise from a specific observing run on which ALBUS will be used. This is done because the noise is not assumed to be the same for different observing runs.

Architechture

The architecture of the CNN is inspired by [51] which is itself inspired by the U-net presented in section 2.1.2. The exact architecture is shown in Fig. 2.18.

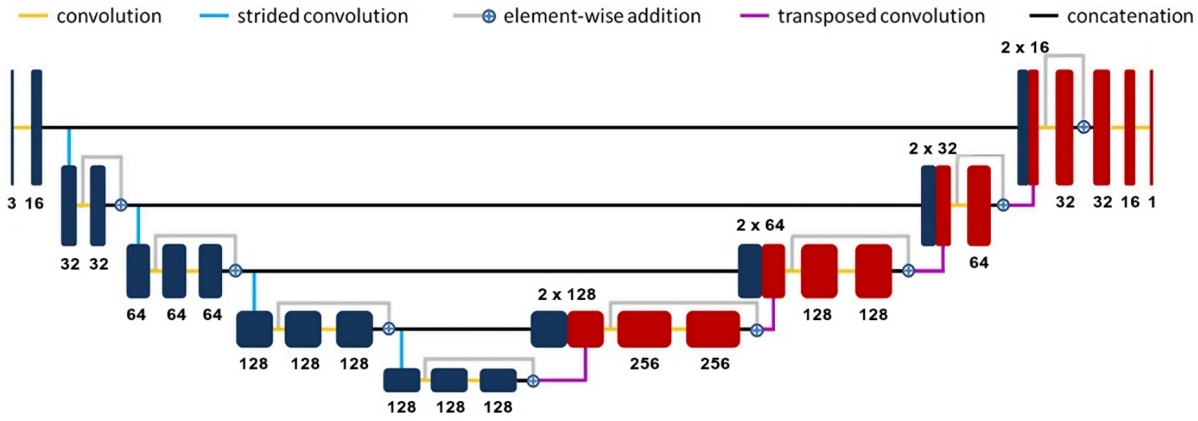


Figure 2.18: The architecture of ALBUS, with the encoder part in blue and the decoder part in red. Lines represent the concatenation of the previous feature maps and the numbers are the number of feature maps. [50]

Training and Results

The Loss function used was the MSE loss between the outputs of ALBUS and the target maps as they are real-valued. This sets ALBUS as a noise removal filter, that regresses a spectrogram of the expected signal/glitch in no noise.

It was trained for 20 epochs using Adam on a training-validation-testing split of 60% - 6.66% - 33.33% (i.e. 9000 - 1000 - 5000 images from each dataset). The loss curves are shown in Fig. 2.19. The output of ALBUS on two samples is shown in Fig. 2.20 and 2.21.

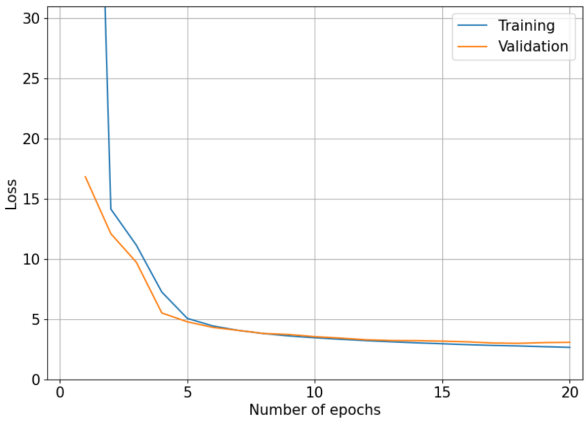


Figure 2.19: The training and validation loss curve of the training of ALBUS

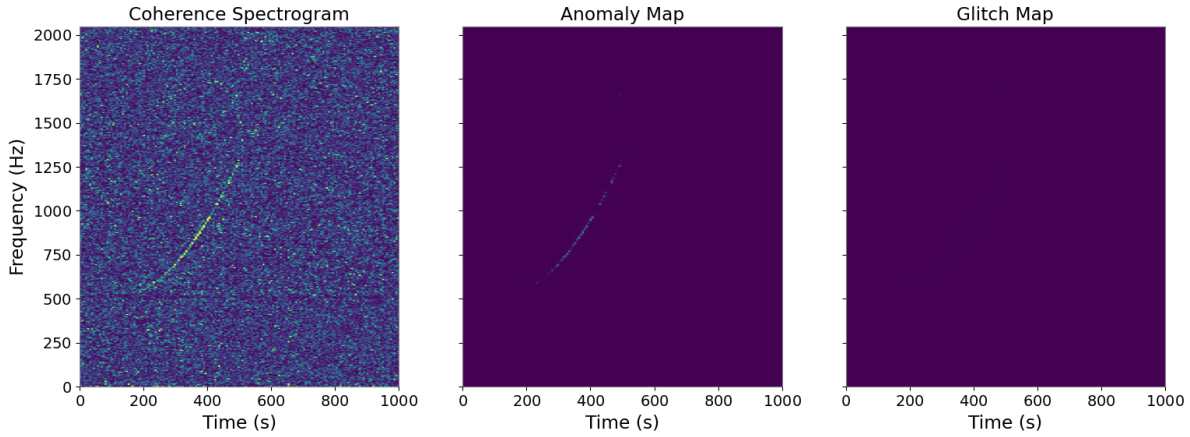


Figure 2.20: The outputs of ALBUS for this coherence spectrogram (left). The Anomaly Map (middle) is the output containing the signal, the anomaly. The glitch map (right) is the output containing the potential glitch, as there is no glitch in the input spectrogram, ALBUS correctly outputs a blank glitch map.

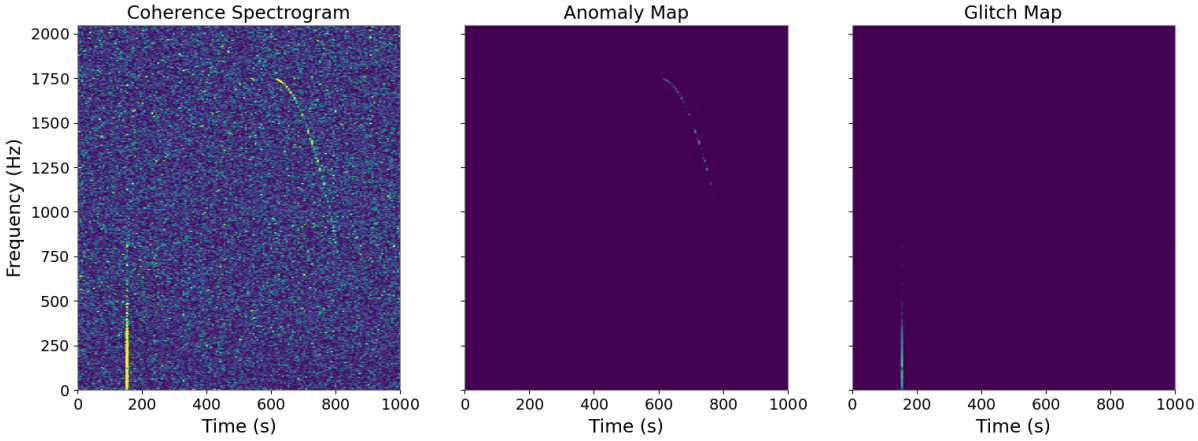


Figure 2.21: The outputs of ALBUS for this coherence spectrogram (left). The Anomaly Map (middle) is the output containing the signal, the anomaly. The glitch map (right) is the output containing the glitch.

Clustering of ALBUS outputs

To be used in a GW pipeline, the output of ALBUS must be transformed into triggers, triggers are potential signals that are ranked according to detection statistics based on their intensity and their shape. The triggers allow for a statistical analysis of the pipeline and relevant triggers can be sent to astronomers for further investigation. The current clustering procedure to transform the anomaly map into a set of triggers is as follows:

- Apply Yen's Threshold as described in [52]. Which automatically finds threshold values different for each spectrogram. This produces a binary map.
- Apply the Euclidian Distance Transform (EDT) on the binary map, this sets the value for each pixel as its Euclidian distance to the closest thresholded pixel.
- Threshold this EDT at a value of 5, which essentially enlarges all previously found clusters, and links clusters which were 10 pixels apart.
- All non-connected clusters are labelled as triggers that can be statistically analysed.

This exact procedure is illustrated for a certain input spectrogram in Fig. 2.22.

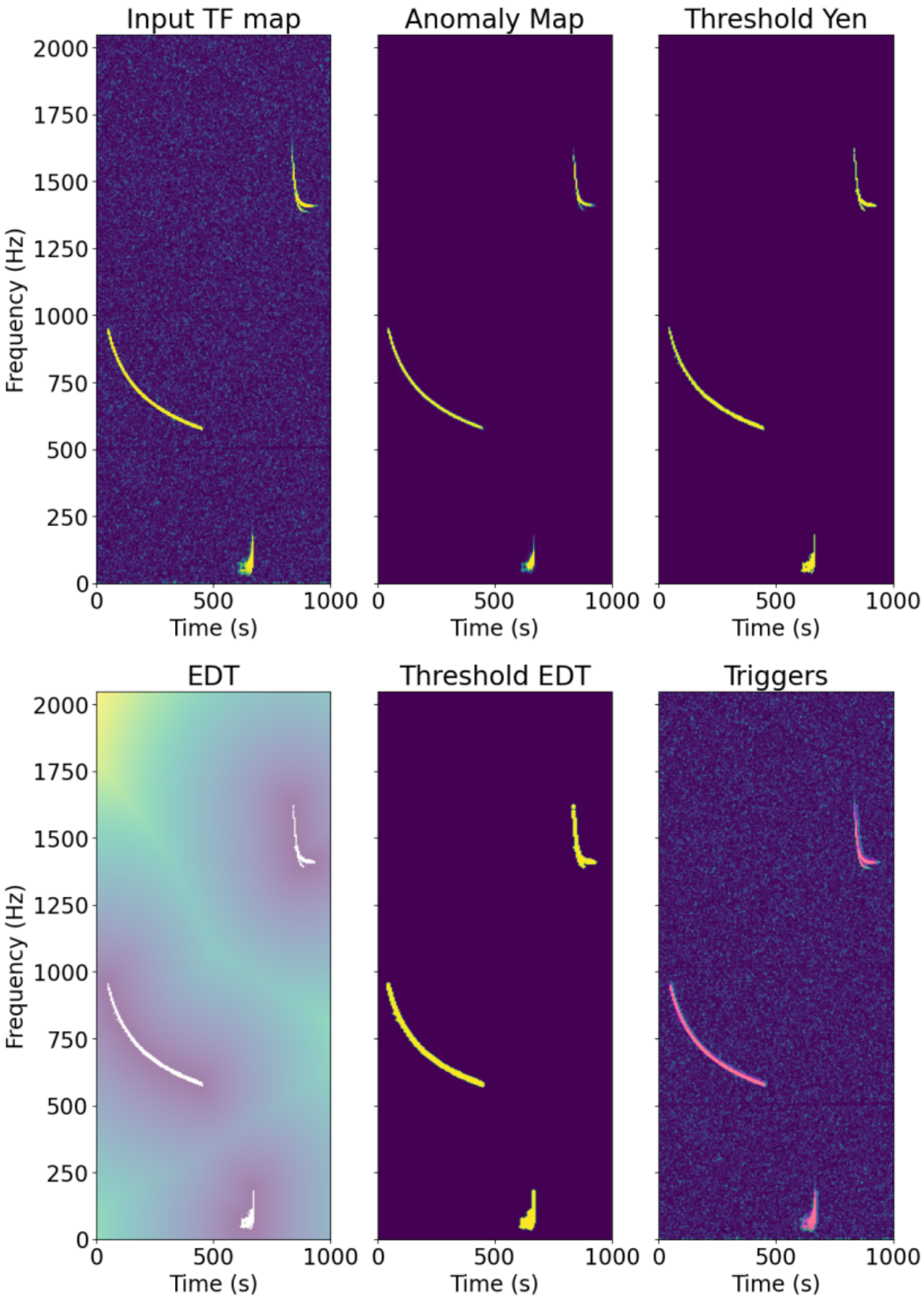


Figure 2.22: The clustering procedure for a certain TF map (Time-Frequency map, a coherence spectrogram). The anomaly map is the output of ALBUS containing potential signals. It is thresholded using Yen's threshold, and then EDT is applied and thresholded at 5 pixels, this forms the triggers shown in light red.

Chapter 3

Improving ALBUS's Clustering

This work aimed to improve the clustering steps of ALBUS integrated into the GW-pyxel pipeline. As explained earlier, the output of ALBUS, the reconstructed chirp or glitch, passed through a Yen's threshold to become binary, and this binary map was then transformed using a thresholded Euclidian Distance Transform to obtain triggers. The problem with this method was that the reconstructed signals were often not continuous. Yen's threshold can binarise a single signal into multiple separate regions that cannot be linked with the EDT. Also, the EDT greatly enlarges triggers and thus does not provide an exact mask of the potential signal. Finally, Yen's threshold on a spectrogram containing no signals still leads to 5-6 low-significance triggers, which increases the processing time.

We will start by exploring the idea of using an unsupervised clustering algorithm on the output of ALBUS. This was not optimal as it required many free parameters to be tuned. This led to the fundamental question of why the signals in the outputs of ALBUS were sometimes discontinuous. This proved to be due to the generation of the dataset on which ALBUS was trained. This led to the idea of entirely skipping the clustering algorithms and training ALBUS to output the trigger masks directly, which required changing the dataset generation and the training procedure. Results of this newly trained version of ALBUS are then shown and compared to the previous one. Finally, we perform the entire analysis necessary to assess the performance of this new detection pipeline for long-duration burst searches and critically discuss its results.

3.1 Clustering Algorithms

The outputs of ALBUS are spectrograms with continuous values, these are thresholded to become binary, these binary regions are then connected by enlarging them using the EDT. One can replace the use of the EDT by an unsupervised clustering algorithm. This is illustrated in Fig. 3.1 which shows the points in the output spectrogram of ALBUS which are above Yen's threshold and Fig. 3.2 shows the application of clustering algorithms presented in section 2.1.1 on those points.

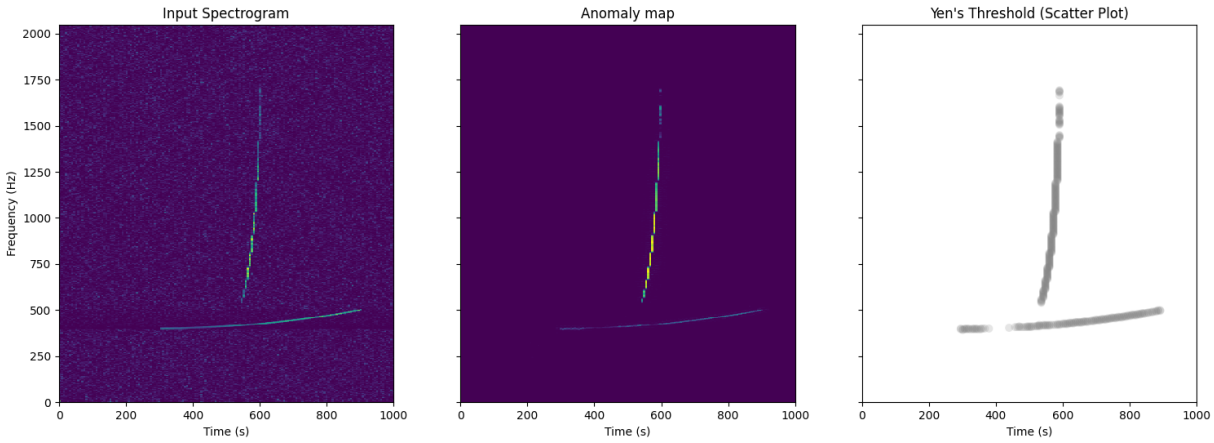


Figure 3.1: A spectrogram containing two signals (left). The output of ALBUS for this spectrogram (middle), only the anomaly map is shown, the glitch map is empty for this spectrogram. A scatter plot of the pixels above the threshold set by Yen's threshold method (right). We observe that discontinuities in the distribution of the points appear when going through ALBUS and Yen's threshold.

Using clustering algorithms to group the points above Yen's threshold in the output of ALBUS is thus not optimal since it requires the tuning of several parameters. For example, Kmeans requires the number of signals to cluster to be known in advance. The cluster densities have to be known for DBSCAN to perform well, but they can vary depending on the signals. HDBSCAN requires a minimum number of points to form a cluster but signals can greatly vary in size so this also requires tuning.

Provided that a satisfactory tuning of these parameters can be found, clustering algorithms are only a fix to a more profound problem. The fundamental problem is the discontinuity of the signals produced by ALBUS and thresholded by Yen's threshold. Fixing this problem can be achieved by modifying these two steps.

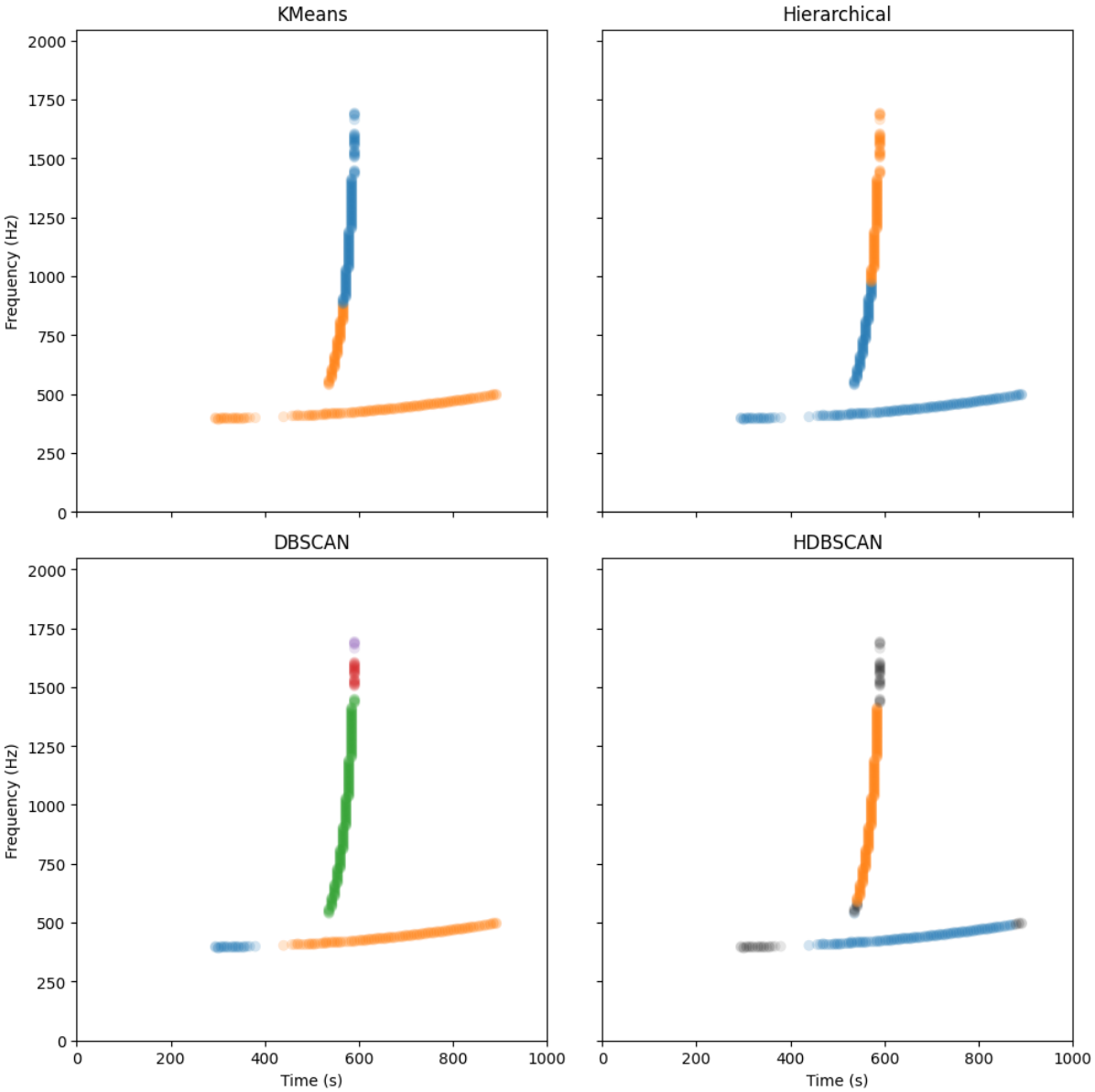


Figure 3.2: A comparison of the four clustering algorithms presented in section 2.1.1 on the pixels above Yen's threshold in the output of ALBUS. KMeans is used to find 2 clusters. Hierarchical clustering uses the average distance and the dendrogram is cut to find 2 clusters. DBSCAN uses $\epsilon = 50$ and $N = 5$. HDBSCAN uses $N = 42$. Even though the parameters are set to produce relevant results no clustering algorithm performs as wanted. Kmeans and Hierarchical clustering assume spherical distributions of points and thus fail to cluster the two signals separately. DBSCAN does not make this assumption and thus it clusters the different lines separately. However, it clusters the discontinuous parts of those signals separately since they are further apart than the distance between the two signals. HDBSCAN performs well but classifies the edges of the signals as noise which is not wanted.

3.2 Directly Finding the Triggers

The discontinuities in the reconstructed signals arise in part due to ALBUS, and more precisely due to its target maps which are generated as discontinuous reconstructions of signals. The target maps are what ALBUS learns to output and thus this CNN reproduces signals with discontinuities. As the outputs of ALBUS are spectrograms of continuous values, the discontinuities are often only drops in the intensity profile of the reconstructed signal. Because of those drops in intensity, the signals are fragmented by Yen's threshold.

One way to prevent this fragmentation would be to train ALBUS to output signals with a smooth intensity profile, the threshold would thus not cut the signal into several parts. Another more elegant way would be to remove Yen's threshold and train ALBUS to directly output the trigger masks. The trigger mask is what was obtained by going through the previous clustering procedure, it is a binary spectrogram containing a connected region for each signal.

One can achieve this goal by changing the role of ALBUS from a noise-removal filter to an image segmentation algorithm. ALBUS as an image segmentation CNN would associate each pixel in a spectrogram with a label, these labels are "background", "chirp" or "glitch". Several changes are required for ALBUS to output segmented spectrograms, these are discussed in the following sections.

3.2.1 New Target Maps

The outputs of ALBUS are determined by the target maps on which it trains. Changing what ALBUS outputs thus requires changing the dataset, specifically the target maps. We need target maps which now are binary spectrograms containing pixels highlighting the signals or glitches visible in the input spectrogram.

To create the chirp target maps, we have access to the signal injected in no noise, which when thresholded, gives a binary footprint of the signal. One cannot use this footprint as a target map because when the signal is injected in noise, its edges can get buried in the noise and the signal will appear shorter. One must not train ALBUS to output those invisible parts of the signal.

The algorithm producing the chirp target maps is presented in Fig. 3.3. It starts by thresholding the signal in no noise at an amplitude outside of our current detectors' sensitivity, this creates a mask containing the footprint of our signal. It then, through the use of a Sato edge detection filter [47], computes the edge strength of the coherence spectrogram inside this mask. This edge strength is then thresholded at an empirical value to produce a binary spectrogram containing pixels of high edge strength inside the footprint of the signal. We need to filter potential pixels of noise in the spectrogram that have a high edge strength inside a part of the footprint buried into the noise. As a filter, we use the HDBSCAN algorithm presented in section 2.1.1 which can find clusters of different densities and labels as noise the points which are not part of any cluster. Further filtering of those noise pixels is applied using the `remove_small_objects()` function from the skimage library [53]. A bounding box containing all the non-filtered pixels is then

used to crop the mask, forming our target map.

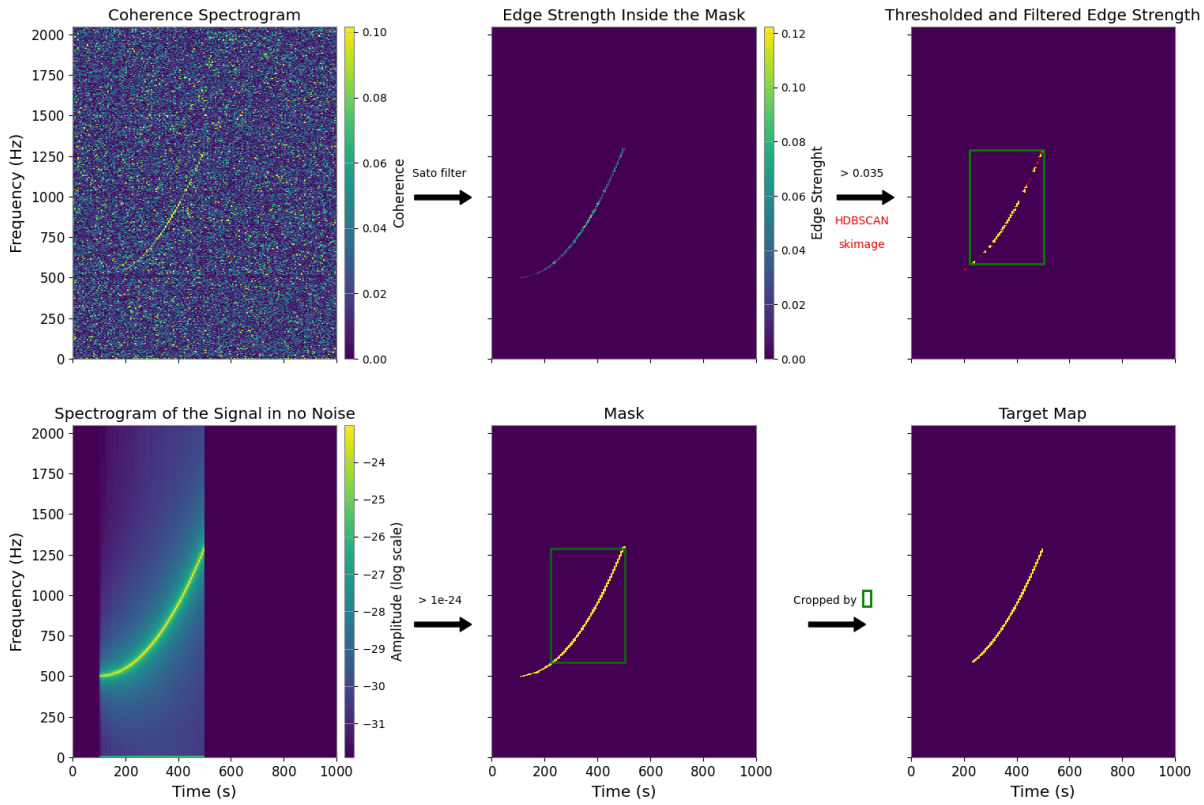


Figure 3.3: The algorithm to produce the binary target maps. It uses the coherence spectrogram and the spectrogram of the signal injected in no noise, which is thresholded above an amplitude of 10^{-24} $1/\sqrt{Hz}$. The edge strength of the coherence spectrogram inside the mask is computed using a Sato filter. The edge strength is thresholded at an empirical value of 0.035, this thresholded spectrogram is then filtered using HDBSCAN and `remove_small_objects()`. The points passing these filters are shown in yellow and the others in red. The target map is the mask cropped by a bounding box containing all the filtered pixels

The generation of these binary target maps is more convoluted than the previous target maps, this is not an issue since they only serve during training and, as long as they are generated correctly, they can be done by any means. Many datasets are even labelled by hand, like in [48].

The background target maps are spectrograms with every pixel labelled as background, and the glitches target maps are generated as before, they are the outputs of an intermediate version of ALBUS trained only on chirps and background. The combined target maps are generated by first creating the glitch target map, then injecting a chirp and creating the chirp target map.

3.2.2 Loss Function and Architecture

Now that the target maps are segmented spectrograms, we use the cross entropy loss function to train ALBUS. This loss function expects as input the probabilities for each class for each pixel. We need to modify the last convolutional layer of ALBUS that previously outputted two spectrograms to output three spectrograms. The value of each pixel inside the three spectrograms will be the probability for the pixel to belong to the "background", "chirp", or "glitch" class respectively. The probabilities need to be in the range $[0, 1]$ and sum to 1. We thus add a SoftMax layer which normalises the three outputs x_i from the last convolutional layer,

$$\text{SoftMax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{C} e^{x_j}}. \tag{3.1}$$

Where C is the number of classes, three in our case.

3.2.3 Training

A training-validation-test split of 75%-15%-10% was used (ie. 36689-7336-4891). We used batches of size of 32, containing images randomly sampled from the four datasets. We performed hyperparameter tuning using the validation set for 34 models using different learning rates, learning rate schedulers, weight decay, initialisations, architectures, widths, and data augmentation.

The learning rate schedulers tried were `ReduceLROnPlateau` and `CyclicLR` from PyTorch [54]. `ReduceLROnPlateau` reduces the learning rate by a certain factor when a metric has stopped improving for a certain number of epochs. `CyclicLR` implements a triangular cyclical learning rate as detailed in [55]. This cycle helps in deciding when to stop the training phase because the losses become minimal when the learning rate is minimal.

We used the Adam optimizer, which has a weight decay parameter that prevents the weights from becoming too large and thus acts as a regularisation method against overfitting. The initialization of the weights tried were the Xavier and He initialisations, as well as starting from the weights of the previous version of ALBUS. These parameters were found to not greatly affect the training nor reduce overfitting.

As in [48], the DeepLabV3 model was also tried as a replacement for the U-Net architecture of ALBUS. This led to poorer results because the segmentation was not as precise as with ALBUS, this could be investigated further using different resolutions.

The first layer of ALBUS produces 16 feature maps, which are doubled several times to reach 256 individual feature maps at the centre of the network. Different widths were tried by changing the number of feature maps by a factor of 1/4 to 2.

Data augmentation was used, which is a technique to artificially increase the size of the dataset by applying modifications to the images ranging from random flips, rotations, blurring, changes in colour, etc. As the model is purposefully trained only on real noise

spectrogram, we restrict the data augmentation to a horizontal flip of the image which is equivalent to reversing the time in our spectrogram. The signals are generated with parameters randomly taken from Table 2.1, flipping the time axis is effectively another realisation of a randomly generated signal. At the scale of our spectrogram, the coherence of the noise does not appear to depend on time, horizontally flipping our images does not alter their physical plausibility. Since our spectrograms are spectrally whitened, we could even flip the frequency axis, but this was not done since glitches seem to be more frequent in the lower frequencies.

The final parameters were chosen as the ones that performed best on the validation set. This model uses a cyclical learning rate ranging from 10^{-4} to 10^{-2}, the training is stopped after 2 cycles (6 epochs) to avoid the apparent overfitting problem appearing after 10 epochs. Stopping the training is called early stopping and is a regularisation method against overfitting. It uses no weight decay, He initialization, the ALBUS architecture with 16 feature maps and data augmentation through horizontal flipping. The loss curve of this model is shown in 3.4a. Then, the model was trained on the training and validation sets and the final performances were computed on the test set. The metric used is the intersection over union (IoU) which computes, the ratio between the intersection and union of the outputs of ALBUS and the target maps. The IoU can range from 0 to 1, the IoU is 0 when the highlighted pixels do not overlap with the target map, and it is 1 when they are the same. It achieved an IoU of 0.82 ± 0.15 for the chirp class and an IoU of 0.84 ± 0.2 for the glitch class. The evolution of the metric is shown in Fig.3.4b.

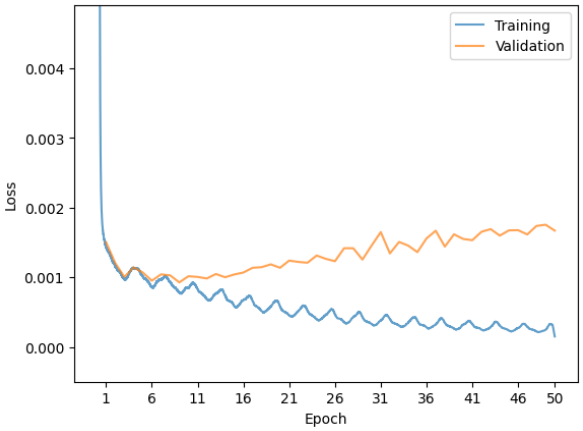


Fig. 3.4a: The loss curves of the final model. The training loss is shown per batch to visualize the effect of the cyclical triangular learning rate scheduler, it is smoothed over 1000 batches for better visualisation. The training loss decreases near 0 while the validation loss stagnates after several epochs and starts to increase after 10 epochs, we perform early stopping at 6 epochs to avoid this problem.

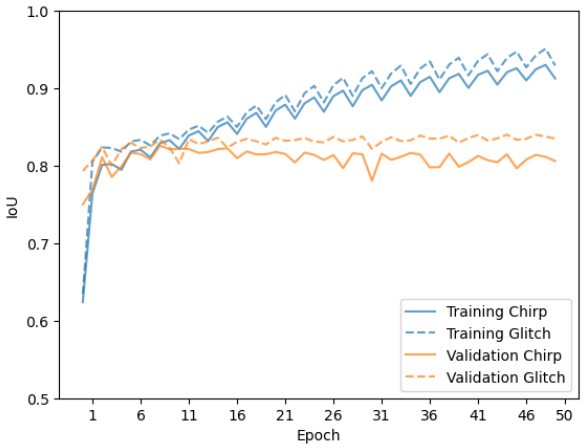


Fig. 3.4b: The IoU curves of the final model for the chirp and glitch class. The training IoU metric increases towards 1 while the validation metric stagnates after several epochs, we perform early stopping at 6 epochs to avoid overfitting. The model generally performs best on glitches which can be understood since glitches have a simpler morphology than chirps, they often appear as vertical lines.

3.3 Results

This section illustrates the results obtained with this newly trained version of ALBUS. We first detail the enhanced clustering process. We then perform the analysis of the performance of this model as part of a burst detection pipeline.

3.3.1 Clustering

The output of ALBUS on spectrograms containing chirps and glitches is shown in Fig. 3.5 and Fig. 3.6. This new version of ALBUS successfully segments spectrograms by highlighting pixels belonging to chirps or glitches.

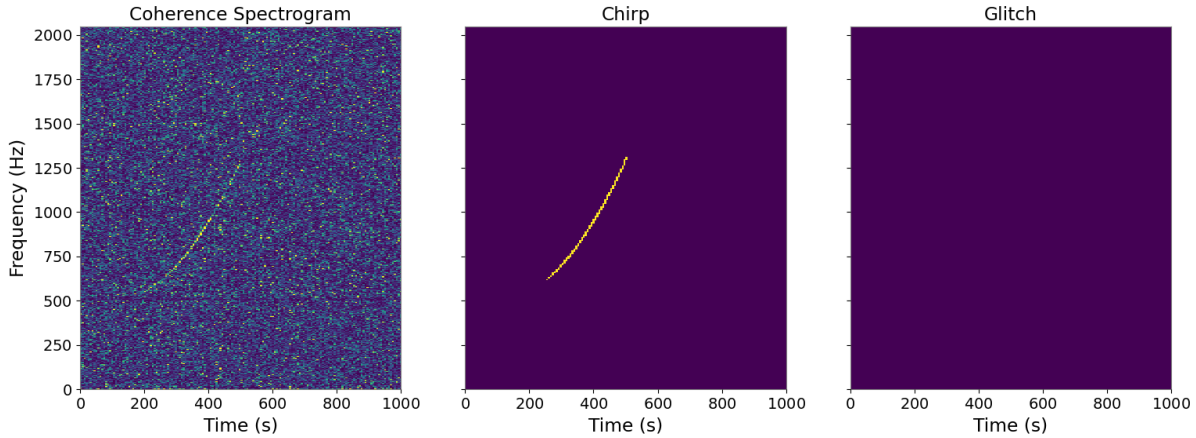


Figure 3.5: The input coherence spectrogram containing a signal (left). The output of the new version of ALBUS which segments the input spectrogram into the chirp (middle) and glitch (right) class. The signal is well recovered. There is no glitch present in the spectrogram, no pixels are labelled as glitch by the network.

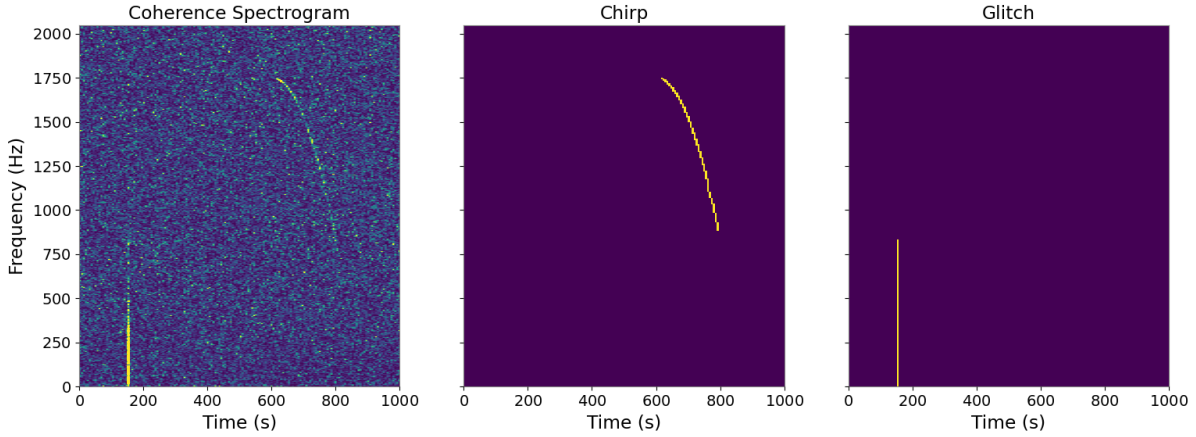


Figure 3.6: The input coherence spectrogram containing a signal and a glitch (left). The output of the new version of ALBUS which segments the input spectrogram into the chirp (middle) and glitch (right) class. The signal and the glitch are well recovered.

The clustering process for a spectrogram containing multiple astrophysical signals is shown in Fig. 3.7. We notice that it highlights similar triggers as the previous clustering process shown in Fig. 2.22 but in only one step.

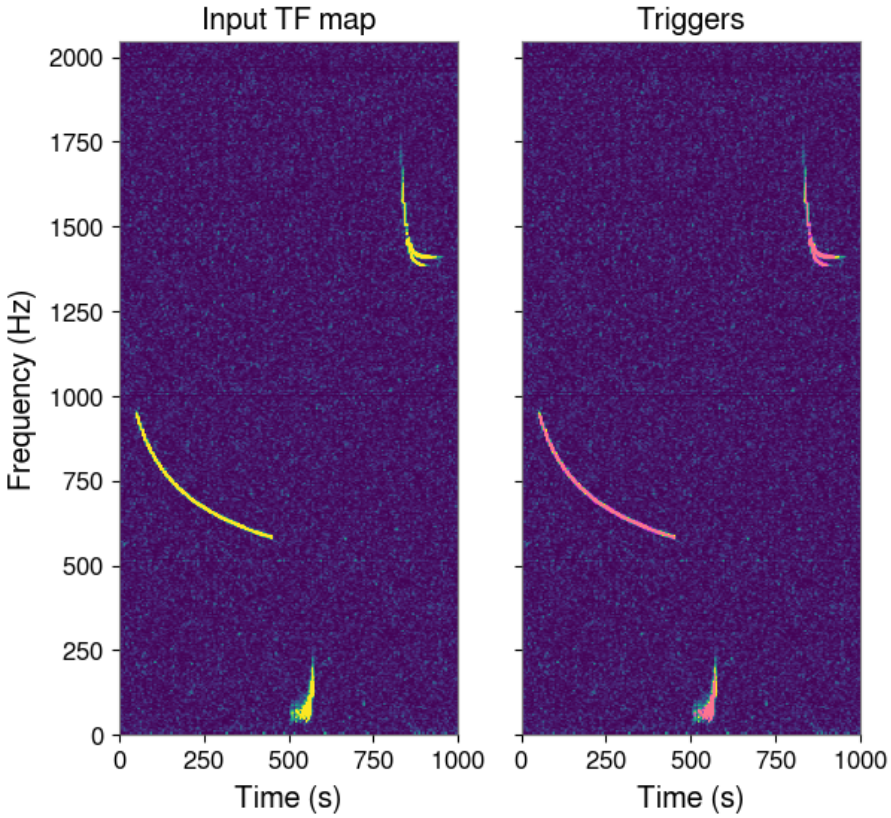


Figure 3.7: The input time-frequency spectrogram in which three simulated astrophysical signals have been injected (left). The triggers directly found by the new version of ALBUS (right), the triggers are highlighted in pink on top of the input TF map.

We can use the test set as a way to quantify the improvement in the clustering procedure. We first compare the IoUs of the new outputs and the target maps, against the IoUs of the Yen-thresholded previous outputs and the target maps. This is done in Fig. 3.8 which shows the distribution over the test set of the IoUs of the chirp and glitch class for the old and new version of ALBUS. The previous model reaches a chirp IoU of 0.65 ± 0.18 and a glitch IoU of 0.65 ± 0.2. The new model reaches higher IoUs (0.82 ± 0.15 and 0.84 ± 0.2) meaning that its outputs more closely resemble the target maps than the outputs of the previous model that have been thresholded. Provided that our target maps always show the right footprint of the signal, our model thus more precisely highlights each pixel belonging to a chirp or a glitch.

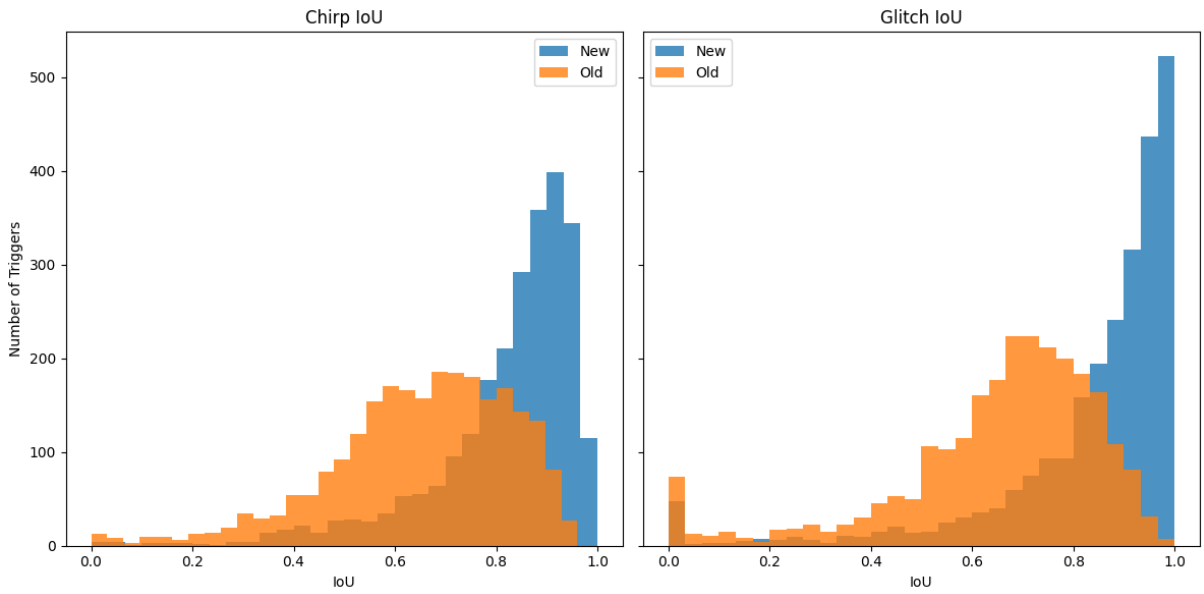


Figure 3.8: Histogram of the IoU between the new target maps and the outputs of the new version of ALBUS (blue), and the IoU between the new targets maps and the Yen thresholded outputs of the previous version of ALBUS (orange). The left histogram is for the chirp class and the right spectrogram is for the glitch class. The new models more closely match the new target maps compared to the Yen thresholded outputs of the previous model which are more discontinuous. We note as seen in Fig. 3.4b that the new model performs slightly better on glitches.

The test set contains spectrograms with a single injected chirp. We thus run the old and new clustering procedure on the test set and record the number of triggers per spectrogram, which should always be 1 if a signal is visible. The histogram of the number of triggers per spectrogram is shown in Fig. 3.9.

We notice that the new model assigns fewer triggers per signal. This means that the triggers are less discontinuous compared to those obtained by the previous version of ALBUS. However, there are still some signals and target maps which are fragmented into several triggers. This can happen when the signal is highly vertical as shown in Fig.3.10. We thus enlarge the triggers as before using the EDT at a reasonable distance of 1 pixel to link such fragmented signals.

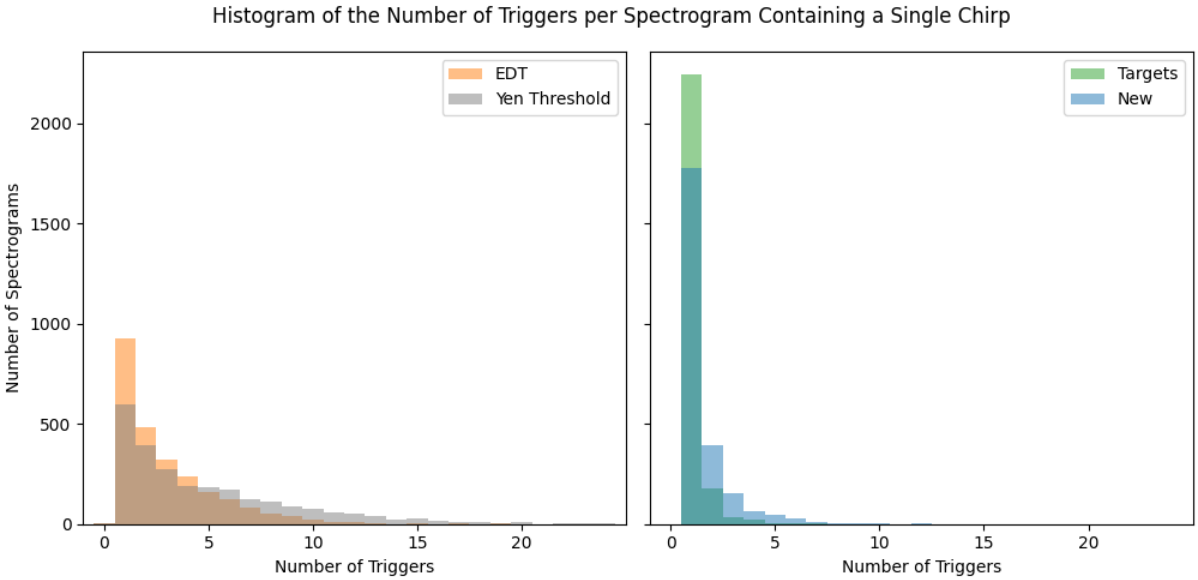


Figure 3.9: Histogram of the number of triggers per spectrogram containing a single injected chirp. A perfect model would always produce one trigger per signal. The results of the previous model are shown on the left, with the number of triggers generated by Yen's threshold in grey and the final number of triggers after applying the EDT in orange. The results of the new model are shown on the right, with the total number of triggers generated by the new ALBUS in blue, and the number of triggers per target map in green. This shows that the triggers are less fragmented using the new model. The target maps as well as the new outputs are still sometimes fragmented this can happen for highly vertical signals.

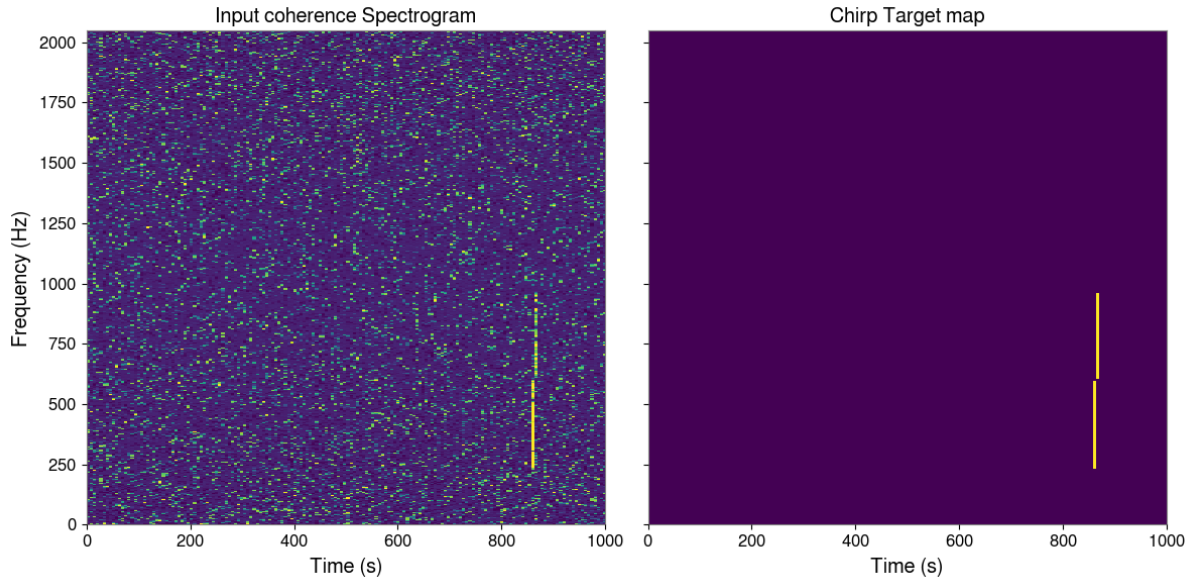


Figure 3.10: A highly vertical signal and its target map which is fragmented into two parts. This is due to the resolution of our spectrograms and the threshold applied to produce the mask during the generation of our target maps.

3.3.2 Analysis

ALBUS is the detection engine inside the gravitational wave burst search pipeline GW-pyxel. We must verify that the changes to ALBUS did not decrease the previous detection capacity.

To assess ALBUS's sensitivity we will first generate several years of data that does not contain any signals. We will then analyse the distribution of triggers found by ALBUS on this data. We will compare the distributions of triggers found on coincident data that potentially contain real signals. Next, we will inject simulated astrophysical signals at increasing amplitudes to determine the amplitude at which ALBUS can detect them.

Background and Foreground Analysis

Having access to the data from two detectors, we can generate several years of data using time slides. This process starts by splitting the multiple days of data available into segments of 1000 seconds and then generates spectrograms of the coherence between two segments at different times. We used 36 days of data to generate 30 years of non-coincident data. These spectrograms do not contain any physical signals since they use data at different times for each detector. We then run this new version of ABLUS on the 30 years of data and record every trigger it found. We associate each trigger with a detection statistic, called the anomaly score, which is the sum inside the coherence spectrogram of the pixels belonging to the trigger. The distribution of the background triggers is represented in red as a reverse cumulative histogram in Fig. 3.14.

This histogram informs us that a total of $\sim$ 3000 triggers were detected above a minimum anomaly score of $\sim 3 \times 10^{-3}$. Most triggers are found around a score of 0.1. A few loud triggers have a score above 0.6, this is the tail of our distribution and we can visualize the loudest in Fig. 3.11 to understand their origin.

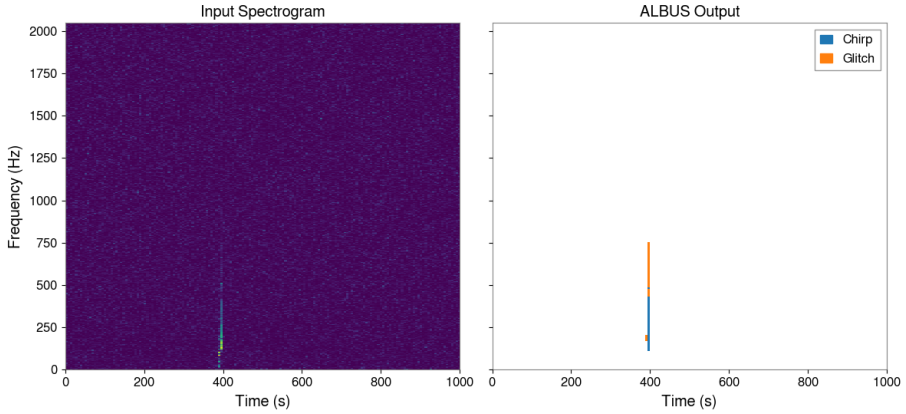


Figure 3.11: The loudest trigger detected in 30 years of non-coincident data. This broad-band trigger is the correlation of two glitches that was partly misclassified as a chirp probably because it spans two pixels.

This spectrogram shows that the loudest trigger is the correlation of two glitches that

was partly misclassified as a chirp probably because it spans two pixels. The tail of the background distribution contains about 5 triggers which are all similar to the one shown in Fig. 3.11, since they are associated with glitches and we expect glitches to have a duration of 6 seconds (1 pixel in our spectrogram) we can decide to disregard any trigger with a duration of 6 seconds. This cut to the background distribution is shown in grey in Fig. 3.14. The loudest trigger of this new distribution is shown in Fig. 3.12, where the noise appears to form a faint line that is detected by the network.

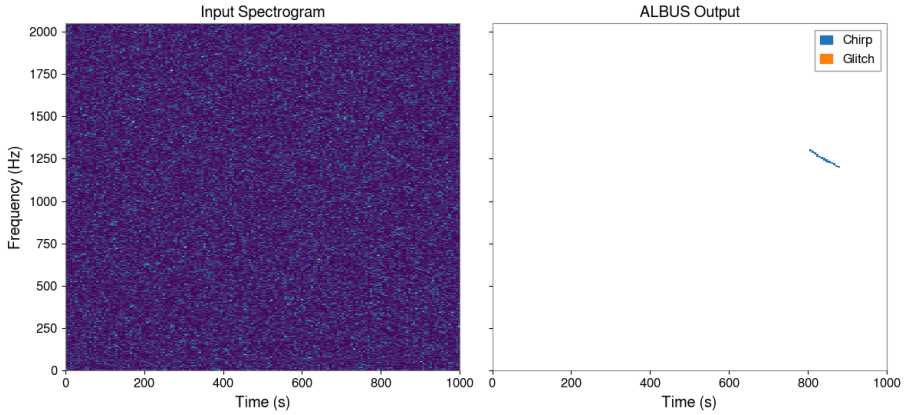


Figure 3.12: The loudest trigger of the cut background distribution. This distribution contains the triggers from the 30 years of non-coincident data (background distribution) that are longer than 6 seconds. It is the 6^{th} loudest trigger of the background distribution. We see in the input spectrogram (left) that the noise randomly formed a line that is detected as a chirp by ALBUS (right).

Finally, the 30 days of coincident data is analysed by ALBUS and the distribution of the 20 triggers found is shown in green in Fig. 3.14. No trigger is found above the background distribution, meaning there is no significant detection. The loudest foreground trigger is shown in Fig. 3.13.

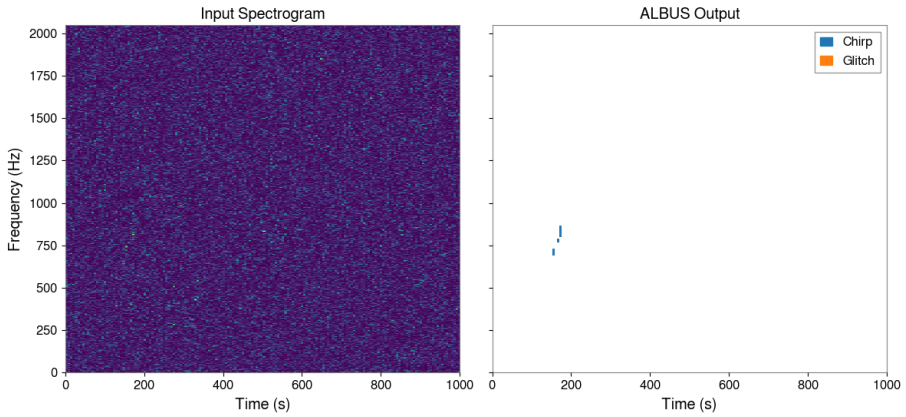


Figure 3.13: The loudest foreground (coincident data) event. This event has an anomaly score below the loudest event of the background so it is not considered significant. It is recovered by the network as several separate triggers because it is uncertain that it is a chirp.

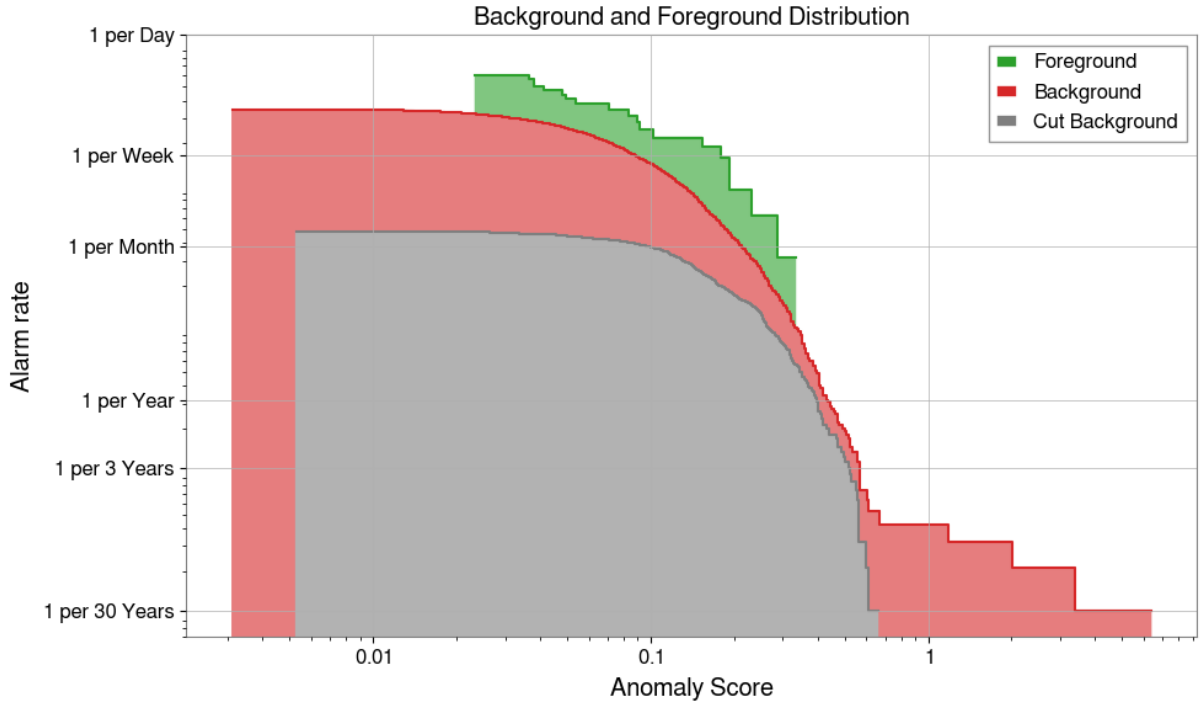


Figure 3.14: Reverse cumulative histogram of the background, cut background and foreground distribution of triggers, expressed using the alarm rate corresponding to the background distribution. The tail of the background distribution is removed by applying a cut to remove every trigger of one pixel. The foreground distribution is slightly above the background which might indicate a difference between coincident and non-coincident data. No foreground trigger has an anomaly score above the background.

The apparent excess of triggers in the foreground distribution compared to the background distribution could be due to a significant difference between the coincident and non-coincident data. We can test the significance of this discrepancy by assuming that the probability for a certain number of spectrograms to contain triggers follows a Poisson distribution. Under this assumption, we can test the hypothesis that the foreground and background distribution have different rates λ of spectrograms containing triggers. Let λ_1 be the background rate and λ_2 be the foreground rate. The null hypothesis H_0 is $\lambda_1 = \lambda_2$, and the alternative H_1 is $\lambda_1 \neq \lambda_2$. We found that 2548 spectrograms contained triggers in the 942 001 background spectrograms and 15 spectrograms contained triggers in the 3178 foreground spectrograms. We can perform a Poisson means test [56] and the probability of finding at least 15 events in the foreground spectrograms under the H_0 hypothesis is 7.3%. This does not provide enough evidence to reject the null hypothesis. However, further analysis with more foreground data could better assess any potential discrepancy. We note that the previous rate of triggers per background spectrogram was 5-6, and is now 1 trigger per 288 spectrograms.

Injection Analysis

The second part of the analysis consists of injecting the simulated astrophysical signals shown in Fig. 1.7 into background noise, at increasing h_{rss} values. The h_{rss} stands for the

root-sum-square gravitational-wave amplitude,

$$h_{\text{rss}} = \sqrt{\int_{-\infty}^{\infty} (h_+^2(t) + h_\times^2(t))\, dt}, \tag{3.2}$$

with h_+ and $h_\times$ being the two signal polarisations. We inject 20 different models at 18 different h_{rss} values 100 times. The proportion of signals detected by the network is called the efficiency and we compute efficiency against h_{rss}. We report the efficiency for every detected signals as well as the efficiency for detected signals that have an anomaly score above the loudest event in the cut background distribution. This ensures that the detection is due to the signal and not a fluctuation in the noise. The h_{hrss} at which the efficiency above the cut background equals 50% is reported as the "50% h_{rss}" and determines the sensitivity of our network to the signal.

The efficiency curves for representative signals of different families are shown in Fig.3.15 for the ISCOchirp, in Fig. 3.16 for the Magnetar (also called magXnetar), in Fig. 3.17 and 3.18 for the ECBC (also called NCSACAM), in Fig. 3.19 for the GRBplateau, in Fig.3.20 for the ADI, and finally in Fig.3.21 for the PT waveforms. The PT (Piro-Thrane) waveforms are not represented in Fig. 1.7, they are the result of fallback accretion on a neutron star [57].

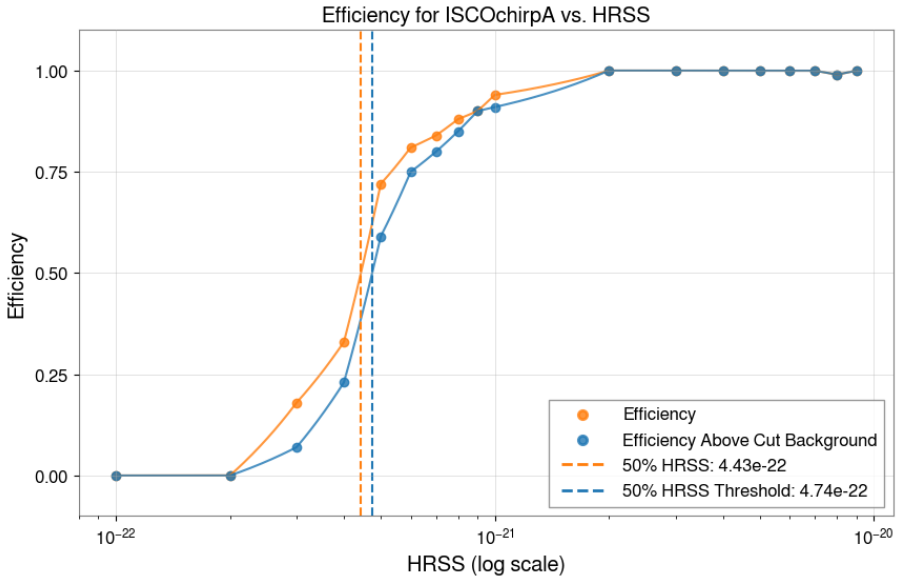


Figure 3.15: The efficiency for the ISCOchirpA waveform at different h_{rss}. The orange curve counts every trigger as detection and the blue curve counts only the triggers having an anomaly score above the loudest cut background event. At low h_{rss} values, no signal is detected, and as the h_{rss} increases, ALBUS detects more of them, eventually detecting all of them. The 50% h_{rss} are represented as dashed lines. These efficiency curves resemble sigmoid functions, the slope of the sigmoid should be the greatest, meaning that every signal is detected above the same h_{rss}. The difference between the two curves should be as low as possible, meaning that the loudest cut background trigger is not negatively impacting the detection sensitivity. We notice a slight dip at 8e-21 h_{rss}. As each injection is associated with a unique identifier, we can go back and look at the reason for this dip. In our case, the background noise was too high at the times chosen for one of the hundred injections at this h_{rss}.

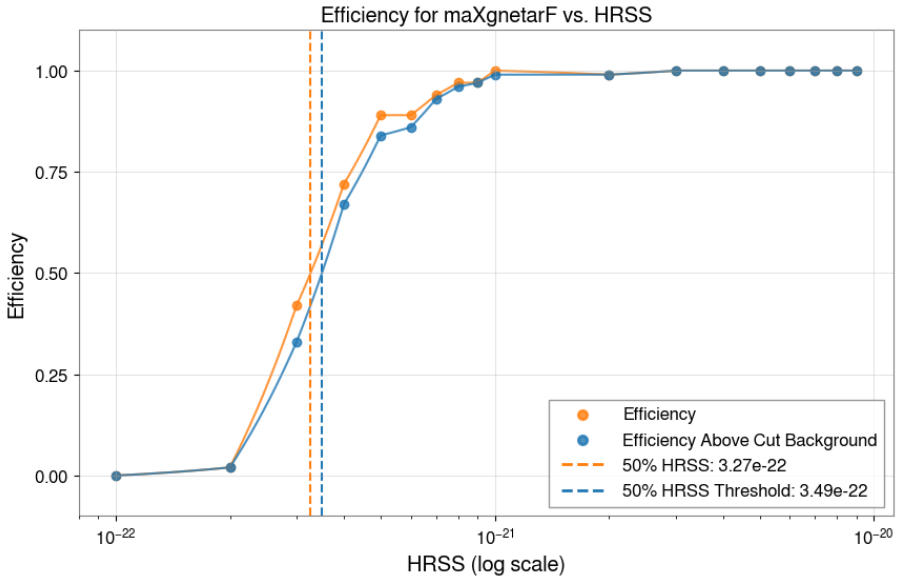


Figure 3.16: The efficiency for the maXgnetarF waveform at different h_{rss}. This waveform is well recovered by the network, the difference between the two curve is small and the slope is high.

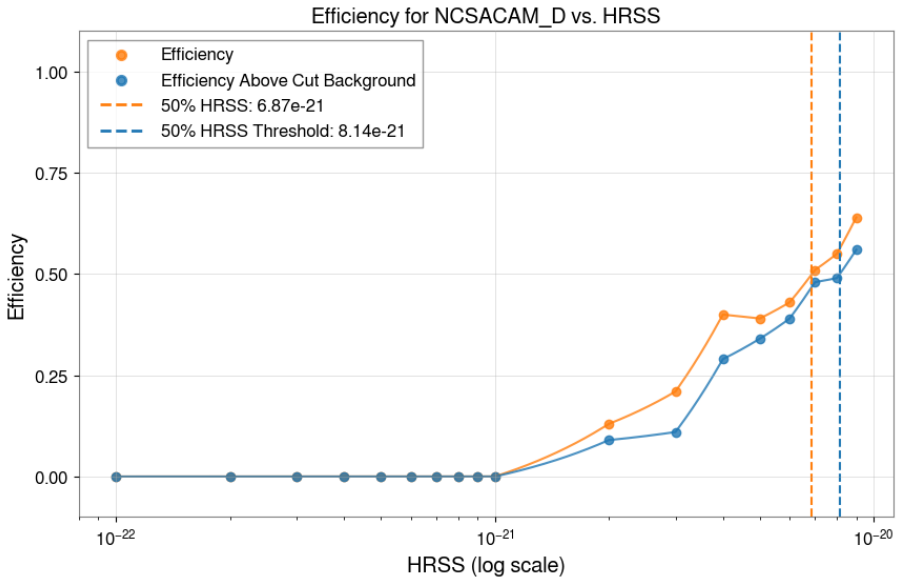


Figure 3.17: The efficiency for the NCSACAM_D waveform at different h_{rss}. This waveform is particularly not well recovered by the network as the efficiency stays at 0 for high h_{rss} values, and does not reach 1. This limitation regarding NCSACAM waveforms is further detailed in 3.4, it is due to the network misclassifying it as glitches due to the morphology of the signal in the spectrogram.

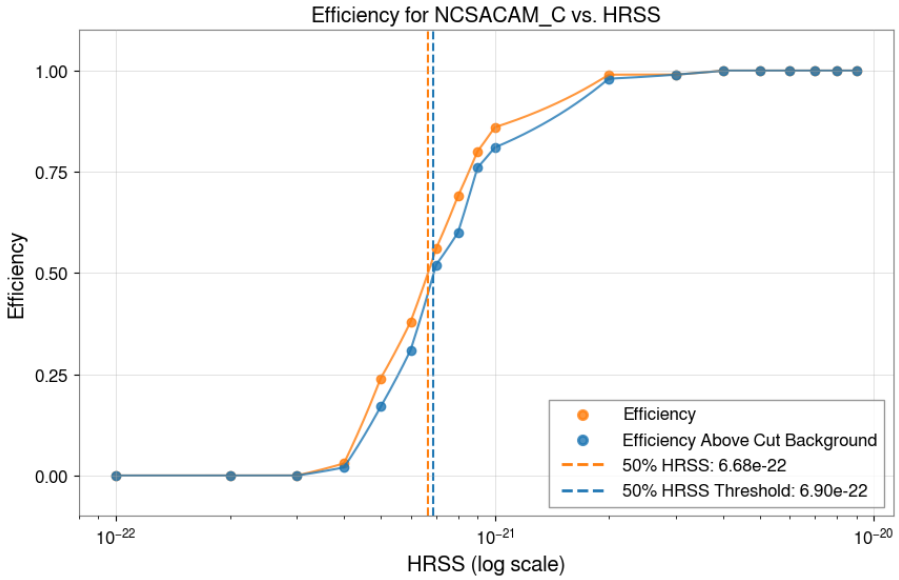


Figure 3.18: The efficiency for the NCSACAM_C waveform at different h_{rss}. Contrary to the NCSACAM_D waveform from the same family, this waveform is relatively well recovered by the network. This is due to the morphology of the signal, especially the harmonics which reduce the verticality of the signal and help the network in not classifying it as a glitch.

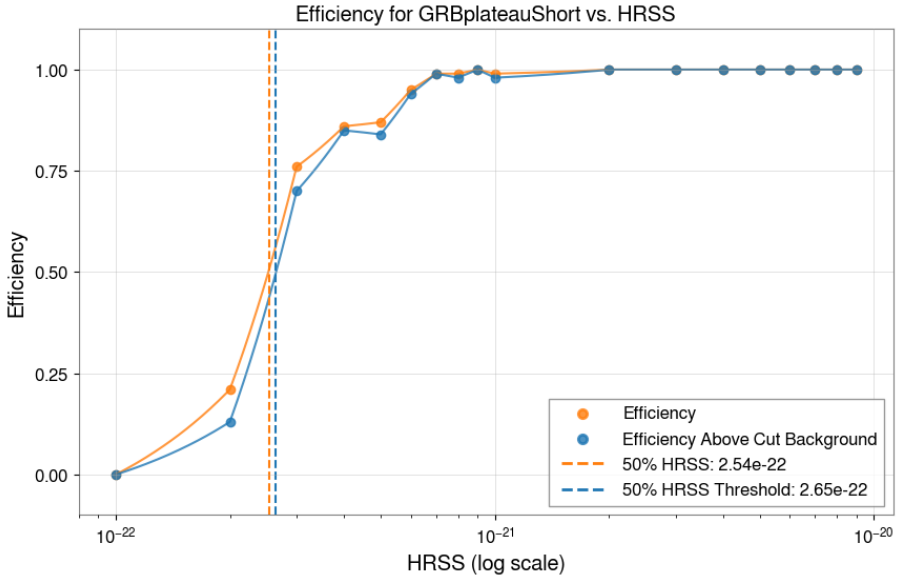


Figure 3.19: The efficiency for the GRBplateauShort waveform at different h_{rss}. This waveform is well recovered by the network. We notice a dip at 5e-21 h_{hrss} which might indicate a change of regime in the inner workings of the network for a waveform of this amplitude.

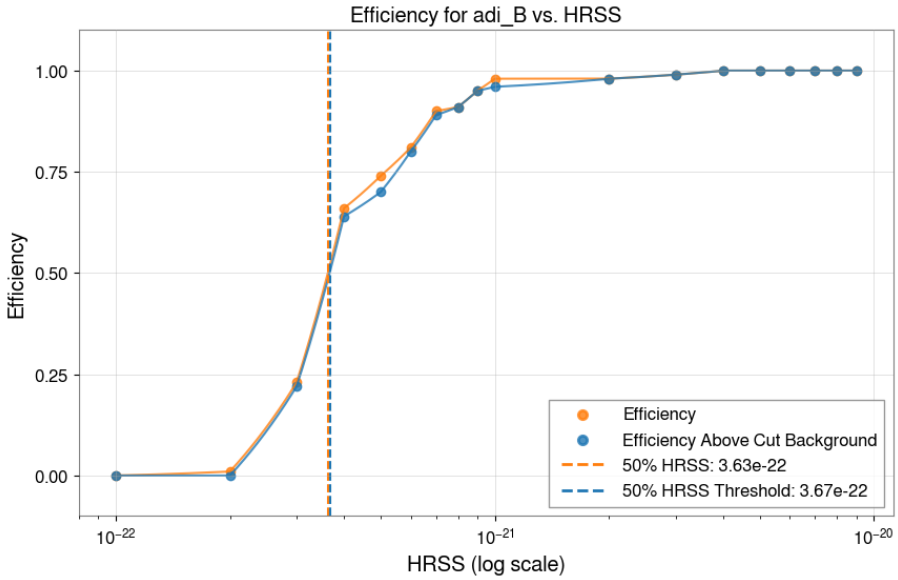


Figure 3.20: The efficiency for the adi_B waveform at different h_{rss}. This waveform is relatively well recovered by the network. However, an efficiency of 1 is only reached a high h_{rss} values. This waveform, as NCSACAM suffers from a drop in efficiency due to its high verticality and resemblance to glitches.

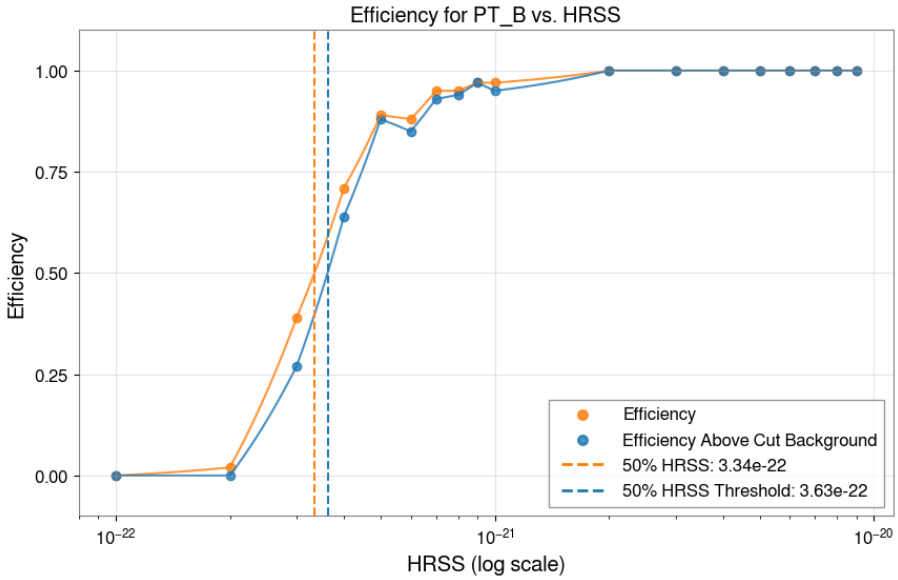


Figure 3.21: The efficiency for the PT_B waveform at different h_{rss}. This waveform is well recovered by the network. We notice a dip a 6e-21 h_{hrss} which might indicate a change of regime in the inner workings of the network for a waveform of this amplitude.

We chose to not focus the analysis on the ECBC (NCSACAM) waveforms since they are not adapted for our network due to their numerous harmonics on which ALBUS is not trained and due to their low frequency and verticality which tricks ALBUS into classifying them as glitches. The poor performance of ALBUS on these waveforms is discussed in more detail in section 3.4.

A comparison of the 50% h_{hrss} obtained using the previous and new version of ALBUS is shown in Table 3.1. The results are different for each waveform but the average change in sensitivity is a factor of 1.00, the new version of ALBUS has a similar performance to the previous one while having a better clustering process. This version performs better on signals which resemble the training set of ALBUS (signals without harmonics that are not too steep, like maXgnetar of PT), and it performs less well on signals that are more different than its training set (like ISCOchirp). A discussion of those results is presented in section 3.4.

Waveform	% 50 h_{rss} (10^{-22})		Ratio
	Previous version	Current version	
GRBplateauShort	2.78	2.65	1.05
ISCOchirpA	4.12	4.74	0.87
ISCOchirpB	2.34	2.69	0.87
ISCOchirpC	2.12	2.29	0.92
PT_A	1.67	2.02	0.82
PT_B	5.68	3.63	1.57
adi_B	2.12	3.67	0.58
maXgnetarD	5.68	5.32	1.07
maXgnetarE	6.13	5.32	1.15
maXgnetarF	3.68	3.49	1.05
Average Ratio			1.00

Table 3.1: Comparison of the 50% h_{rss} of the previous and current versions of ALBUS on astrophysical waveforms.

3.4 Discussion

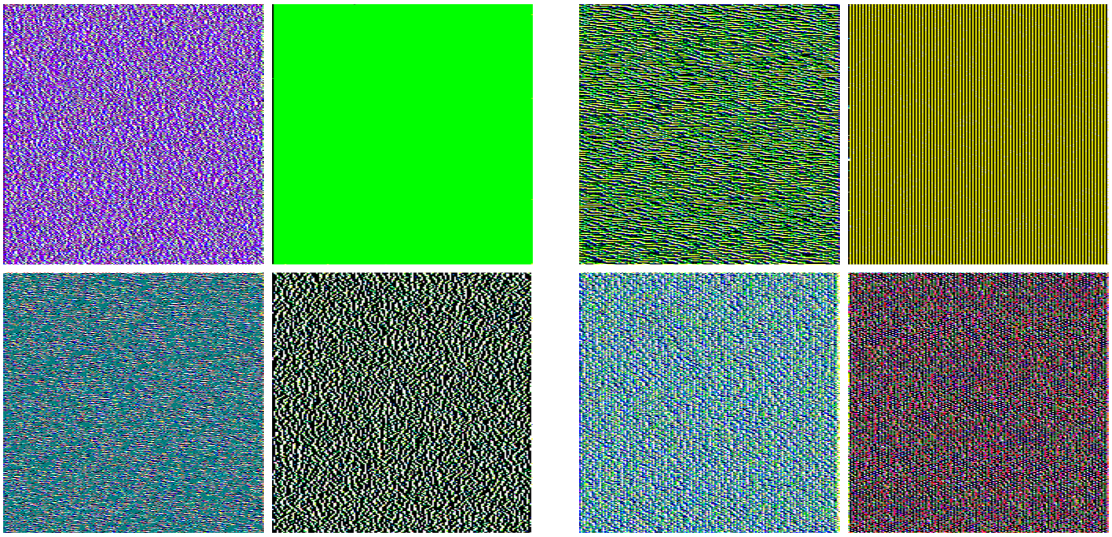
3.4.1 Under the Hood

Neural networks are often labelled as black-box algorithms, assuming that their behaviour is unexplainable. This is partially true due to their large number of parameters. However, it is still possible to grasp some aspects of the inner workings of CNNs. Understanding the behaviour of the algorithm is useful for discussing its results. We will first present the maximum response samples and then intermediate probabilities outputs of the CNN.

Maximum Response Samples

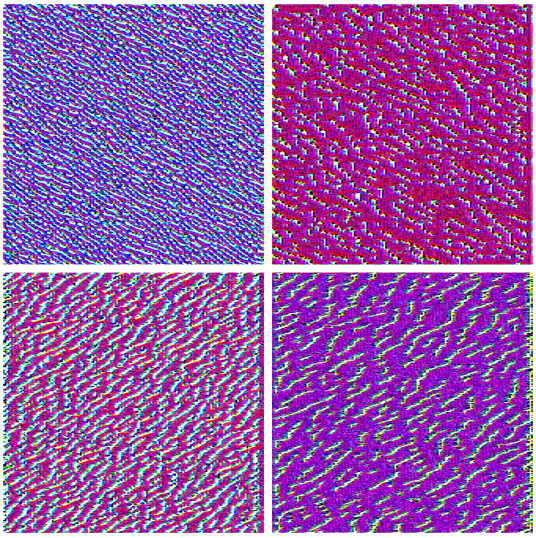
Having at our disposal a trained neural network with fixed weights that takes as input images, we can generate synthetic images that maximize the activation of a certain filter at a certain layer. We can perform gradient ascent of this activation with the parameters being the values of the pixels in the input image. This produces synthetic images that highlight what information a filter of the network is extracting from the image.

Fig. 3.22 shows the maximum response sample for the first four filters of the first six layers of the network. The first layers extract low-level information like the colour of the image or its texture. For example, the second filter in the first layer is maximally activated when the input image is green. The second filter in the second layer is looking for bright, yellow, vertical lines, which is reminiscent of what our glitches look like in our spectrograms, this filter is probably one of the many filters looking for glitches. As we go deeper into the network, the patterns become increasingly complex because they are the result of combinations of the previous patterns. Some of these deeper filters look for continuous ridges, reminiscent of our chirping signals. This figure only shows 24 maximum response samples each one corresponding to a filter, the networks contain about two thousand filters and two million parameters.

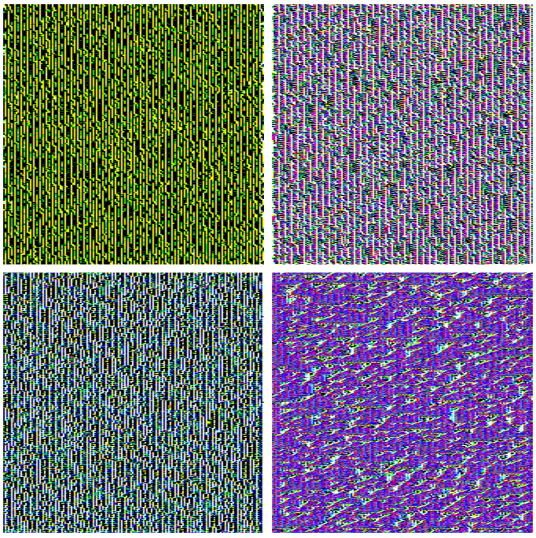


Layer 1

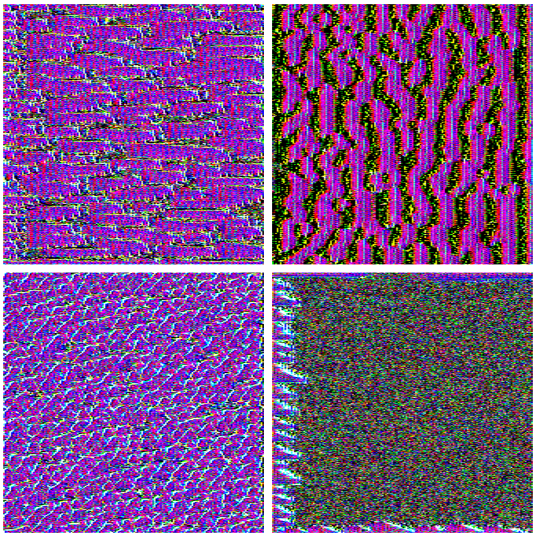
Layer 2



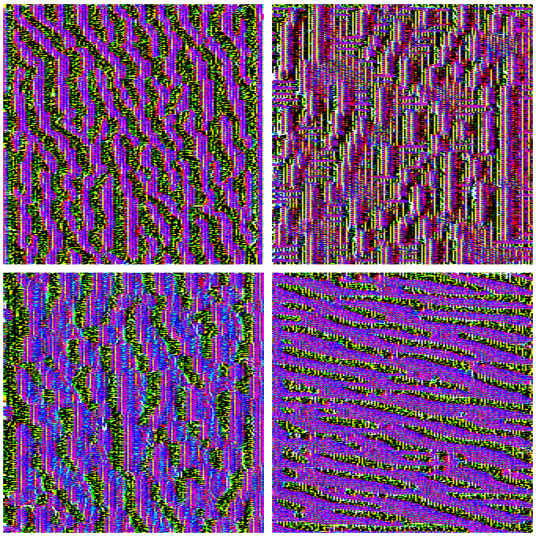
Layer 3



Layer 4



Layer 5



Layer 6

Figure 3.22: Maximum response samples of 4 filters for different layers

Intermediate Probabilites

This new version of ALBUS performs image segmentation, which assigns a label to each pixel in the spectrogram. The chosen label is the one which has the highest probability. We can access these probabilities that are produced by ALBUS just after the SoftMax layer, visualizing these intermediate probabilities reveals information that is lost when converted to a segmented spectrogram. Such visualisation of the intermediate probabilities and their logarithm is shown in Fig. 3.23.

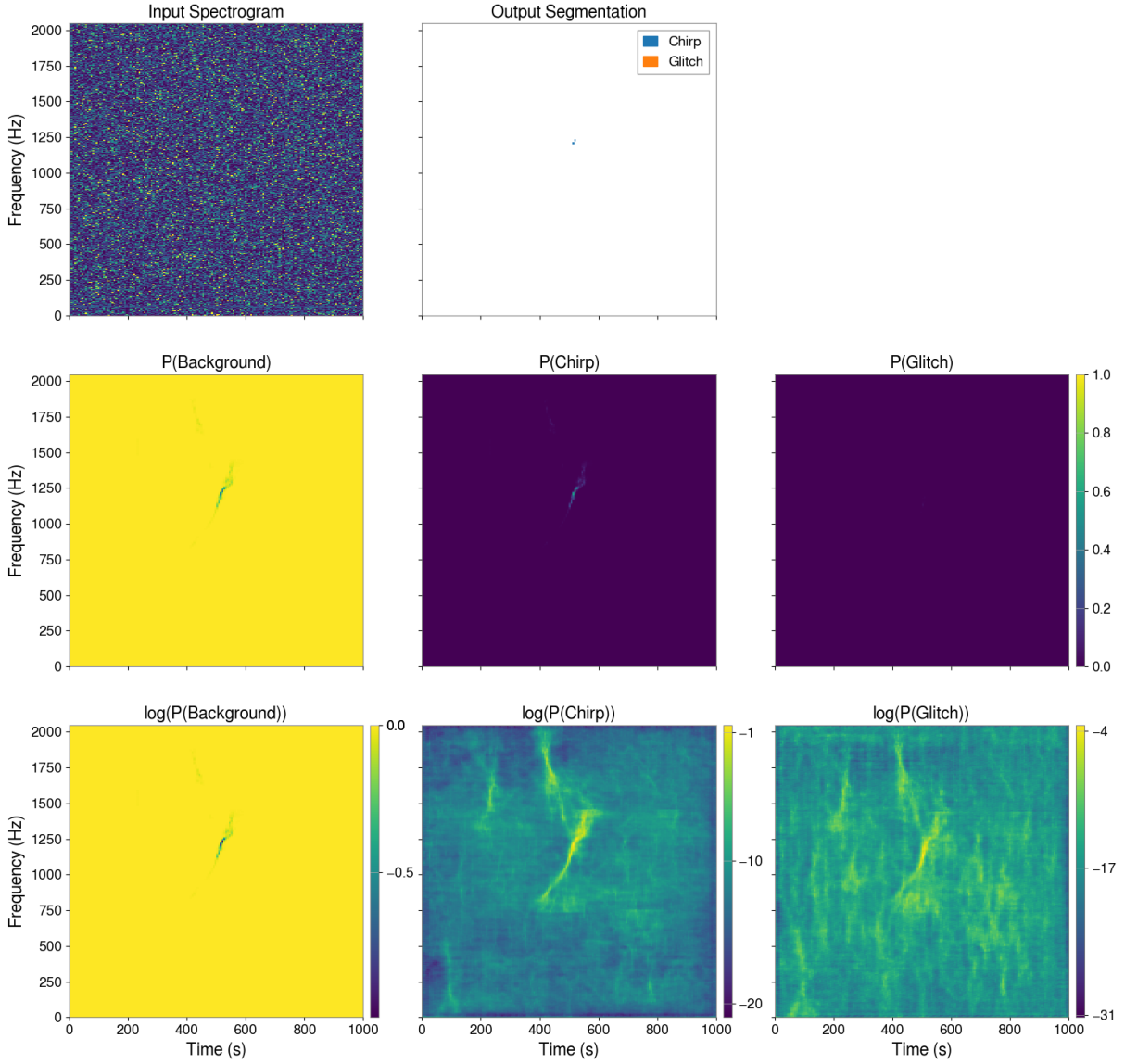


Figure 3.23: A visualisation of the probabilities associated with each class for a spectrogram of background data where ALBUS finds a trigger. The input spectrogram (top left) and its segmented output (top middle). The probabilities for each class (middle row) are obtained after the SoftMax layer. The logarithm of those probabilities (bottom row) highlights the complex patterns captured by the network.

This figure shows a spectrogram of non-coincident data that does not contain any physical signals. However, the random fluctuations of the noise can resemble lines and be

detected as chirps by the CNN. The segmented output of ALBUS contains part of a faint line that the network classified as a chirp. To better grasp what the network was seeing when segmenting those few pixels we can look at the output of the SoftMax layer which contains probabilities for each class. The background probabilities are high throughout the spectrogram. A faint line appears in the chirp probabilities, the part of this line that has a higher $\mathbb{P}$(chirp) than $\mathbb{P}$(background) is classified as a chirp. The probabilities for the glitch class are near 0. The SoftMax layer takes the exponential of the logits that ALBUS outputs, we can access those normalized logits by taking the logarithm of the probabilities, this gives us a better view of what the network truly sees. Many features that were not visible before appear, we first see that the small segmented signal is seen as part of a potentially larger signal. The CNN also highlights a potential chirp at around 400 s and 1800 Hz, which is somewhat visible in the input spectrogram. Other small, and low probability, chirp-like ridges appear in this log-probability spectrogram. The glitch probabilities are lower but resemble those of the chirp in morphology, although having more emphasis on vertical ridges. The background probabilities are high throughout the spectrogram and decrease where the other probabilities are high.

3.4.2 Limitations

With this greater understanding of the workings of ALBUS we can address the limitations that were highlighted by the analysis. The two main limitations were identified by inspecting the injection analysis process using the intermediate probabilities.

The first limitation is that low-frequency and steep vertical signals like the NCSACAM waveforms are confused for glitches by the network. This is illustrated in Fig.3.24 where we see that the signal is labelled as a glitch. The harmonics present in NCSACAM, when they become visible in the spectrogram, aid the network in classifying it as a signal and not a glitch since they decrease the verticality of the signal, this is visualised in Fig. 3.25.

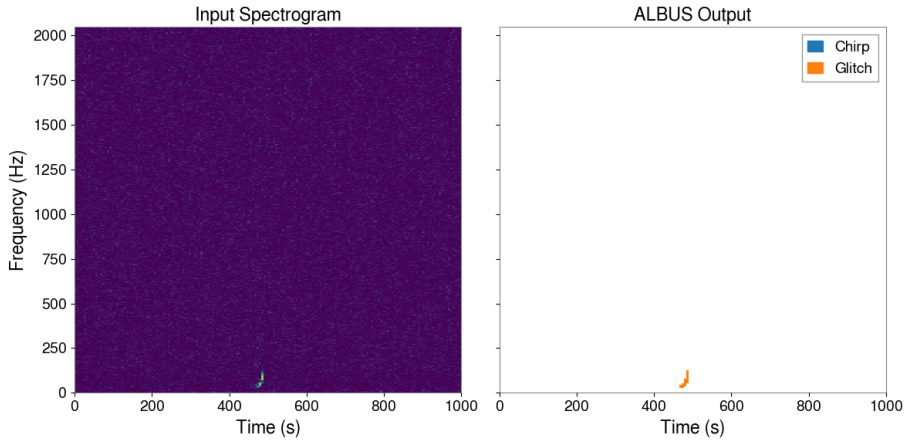


Figure 3.24: An example of an injected NCSACAM_D waveform at 2×10^{-21} h_{rss} that is misclassified as a glitch by the network.

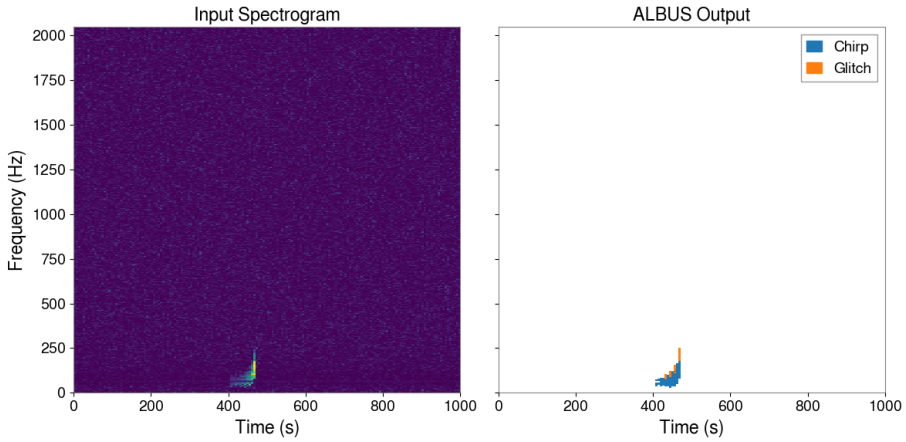


Figure 3.25: An example of an injected NCSACAM_C waveform at 2×10^{-21} h_{rss} that is partially classified as a chirp, in part because of its harmonics. Since this version of ALBUS was not trained on harmonics it still partially misclassified it as a glitch.

The second limitation is that for a pixel to be labelled as chirp it has to have a chirp probability higher than the background and glitch probabilities. It is observed that potential signals are first seen in the probability spectrograms before reaching this threshold. This phenomenon is more pronounced for the waveforms that do not closely resemble those of the training set, like the ISCOchirp waveforms, this decreases the 50% h_{rss} because the waveform has to reach a high h_{rss} to pass this threshold. This is illustrated in Fig.3.26 and Fig.3.27.

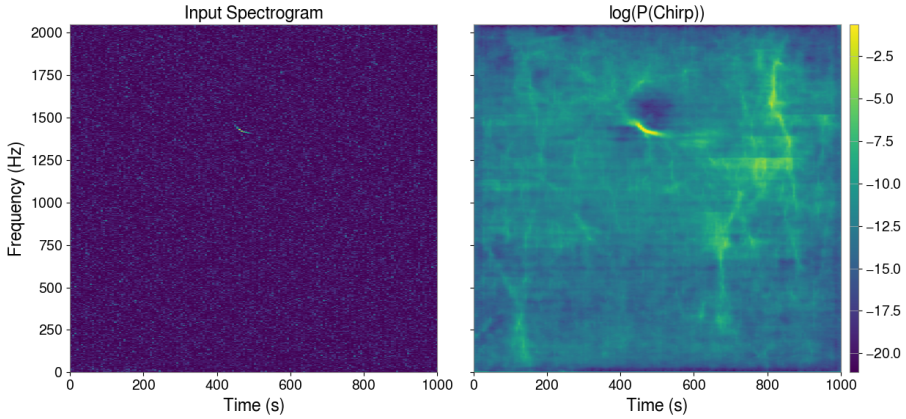


Figure 3.26: An example of an injected ISCOchirpA waveform at 5×10^{-22} h_{rss} that is not classified as chirp by the network even though it is highlighted in the logarithm of the probabilities for the chirp class. The probability of those pixels being background is higher and thus ALBUS does not detect any trigger.

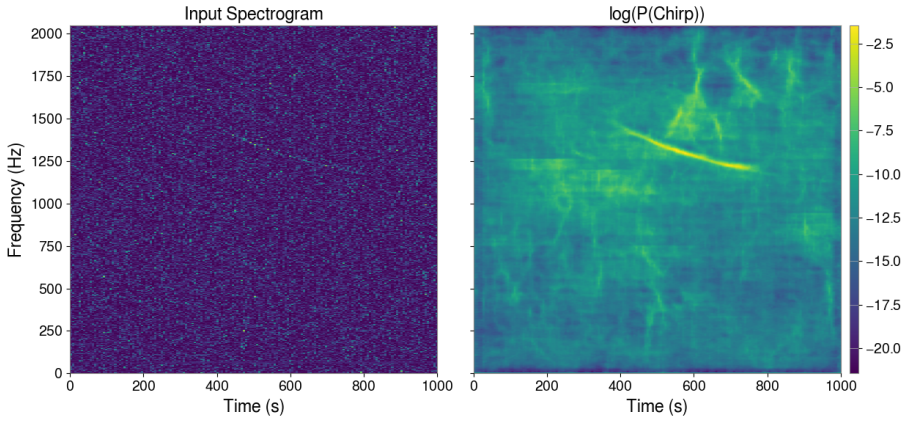


Figure 3.27: An example of an injected maXgnetarE waveform at 4.5×10^{-22} h_{rss} that is not classified as chirp by the network even though it is highlighted in the logarithm of the probabilities for the chirp class. The probability of those pixels being background is higher and thus ALBUS does not detect any trigger.

3.4.3 Prospects

Although the improvement of the clustering process is not a priori supposed to increase the sensitivity of ALBUS, resolving the previously mentioned limitations could make this new version of ALBUS a better detection engine in the GWpyxel pipeline. The problem of the network confusing signals for glitches comes from the fact that glitches and highly vertical chirps have similar morphologies in the spectrogram. A possible solution to this problem could be to use the sky position to discriminate glitches from signals. Since physical signals come from a single direction in the sky, we can shift the two time series to point to a certain direction in the sky.

The problem of pixels belonging to signals that are not segmented as such because they have a higher probability of being background is probably a consequence of class imbalance. Class imbalance happens in classification problems when some labels are over-represented and others are not. The average size of our signals in the training data is 167 pixels, while the rest of the image of size $513 \times 166 - 167 = 84991$ pixels is labelled as background. The chirp/background ratio is 0.2%. It could be the case that the network has been biased by this disparity and learned that assigning large probabilities to the background class decreases its loss during training. Class imbalance is a well-documented problem and many solutions exist to mitigate its effects in our case of semantic segmentation like modifying the cross-entropy loss function [58][59].

A possible direct solution to this problem could be to assign the label "chirp" to a pixel if its probability of being a chirp is above a certain fixed threshold instead of above the background probability. This introduces an empirical value at which to threshold these chirp probabilities, and it is reminiscent of the need to apply Yen's threshold in the previous version of ALBUS. It could still provide better results than the previous method since the probabilities outputted by the new ALBUS are more continuous than the previous outputs of ALBUS which were discontinuous. In practice, one could decrease this probability threshold to an arbitrarily low value, this would have the effect that any

chirp-like ridge inside a spectrogram would be marked as a trigger. Provided that a good detection statistic is used to rank the multiple triggers and enough computing power is available to process them, having more low-probability triggers is not a problem.

Other possible improvements could be to address the issue of the clustering of two intersecting signals. Even though this is not bound to happen in the next years, it could become relevant for future detectors such as the Einstein Telescope which could detect several intersecting signals inside a spectrogram of 1000s. Possible ways to solve the separation of intersecting lines are already used with neural networks in robotics [60] and in other spectrogram segmentation work like [48]. One could also transition to instance segmentation which combines object detection with semantic segmentation to directly distinguish different triggers. Signals with harmonics on which the previous version of ALBUS was trained could be added to the training set of this new version by modifying the target maps. Since the new generation of the target maps uses an edge detection filter, harmonics will decrease the edge strength of the signal, making the direct application of this new algorithm ineffective. This can be easily solved by creating the target maps for each harmonic independently and taking their union as a final target map. The inputs of the network are .png images of the spectrograms, this means that three channels are used as input, but they all carry the same information, this can be changed by using only one channel for the amplitude in the coherence spectrogram. Other input channels can be added to use more information such as spectrograms from the Virgo detector. Increasing the parameter space covered by the chirps in the training set of ALBUS could be beneficial, for example, we could add exponentially decaying chirps to mimic the ISCOchirp waveforms.

3.5 Conclusion

After seeing the emergence of gravitational waves for the theory of General Relativity, we explained how laser interferometers were capable of detecting such waves. We then presented various astrophysical sources which are often poorly understood and our scientific knowledge would benefit from measurements of their gravitational radiation.

The data analysis tools used to extract the physical information from the strain of the detectors were presented, notably the ALBUS model which is a novel way of analysing spectrograms to detect potential signals using deep learning methods.

This work started as an attempt to improve the clustering process of ALBUS. We first tried to add another clustering algorithm on top of the existing processes. We then reverted to a more elegant change in the network which required regenerating the dataset and retraining the model with some modifications.

The direct clustering obtained with this new version is of better quality and the sensitivity was not hindered. These changes open the door for many possible exciting improvements. The field of deep learning does not cease to provide better and better models. Exploring the intersection of the state of the art in gravitational wave astronomy and deep learning seems deeply promising.

Bibliography

[1] A. Einstein, "Die Feldgleichungen der Gravitation," *Sitzungsberichte der Königlich Preussischen Akademie der Wissenschaften*, pp. 844–847, Jan. 1915.

[2] S. M. Carroll, *Spacetime and Geometry: An Introduction to General Relativity*. Cambridge University Press, 2019.

[3] J. Weber, "Evidence for discovery of gravitational radiation," *Phys. Rev. Lett.*, vol. 22, pp. 1320–1324, Jun. 1969. DOI: 10.1103/PhysRevLett.22.1320.

[4] J. L. Levine, "Early gravity-wave detection experiments,1960-1975," *Physics in Perspective*, vol. 6, no. 1, pp. 42–75, Apr. 2004. DOI: 10.1007/s00016-003-0179-6.

[5] A. Cho. "Remembering joseph weber, controversial pioneer of gravitational waves." Accessed: 2024-04-26. (2016), [Online]. Available: https://www.science.org/content/article/remembering-joseph-weber-controversial-pioneer-gravitational-waves.

[6] M. A. Bizouard, "Gravitational wave burst searches," in *Gravitational Waves*, ser. Ecole de Physique des Houches, Session CX, Jul. 2018.

[7] C. F. Da Silva Costa and O. D. Aguiar, "Spherical gravitational wave detectors: Minigrail and mario schenberg," *Journal of Physics: Conference Series*, vol. 484, no. 1, Mar. 2014. DOI: 10.1088/1742-6596/484/1/012012.

[8] V. Faraoni, "A common misconception about LIGO detectors of gravitational waves," *General Relativity and Gravitation*, vol. 39, no. 5, pp. 677–684, May 2007. DOI: 10.1007/s10714-007-0415-5.

[9] The LIGO Scientific Collaboration, "Advanced ligo," *Classical and Quantum Gravity*, vol. 32, no. 7, Mar. 2015. DOI: 10.1088/0264-9381/32/7/074001.

[10] M. Rakhmanov, J. Romano, and J. Whelan, "High-frequency corrections to the detector response and their effect on searches for gravitational waves," *Classical and Quantum Gravity*, vol. 25, Sep. 2008. DOI: 10.1088/0264-9381/25/18/184017.

[11] LIGO Scientific Collaboration and Virgo Collaboration, "Observation of gravitational waves from a binary black hole merger," *Phys. Rev. Lett.*, vol. 116, 6 Feb. 2016. DOI: 10.1103/PhysRevLett.116.061102.

[12] T. Creighton, "Tumbleweeds and airborne gravitational noise sources for LIGO," *Classical and Quantum Gravity*, vol. 25, no. 12, Jun. 2008. DOI: 10.1088/0264-9381/25/12/125011. arXiv: 0007050 [gr-qc].

[13] ET Steering Committee Editorial Team, "ET Design Update 2020," GWIC, Tech. Rep., 2020.

[14] N. Andersson, *Gravitational-Wave Astronomy: Exploring the Dark Side of the Universe*, 1st ed. Oxford: Oxford University Press, Nov. 2019.

[15] LIGO Scientific Collaboration and Virgo Collaboration, "GW170814: a three-detector observation of gravitational waves from a binary black hole coalescence," *Phys. Rev. Lett.*, vol. 119, 14 Oct. 2017. DOI: 10.1103/PhysRevLett.119.141101.

[16] The Virgo Collaboration. "Virgo collaboration observatory gallery." Accessed: 2024-04-26. (2021), [Online]. Available: http://public.virgo-gw.eu/index.php?gmedia=5vp4v&t=g.

[17] P. Amaro-Seoane, H. Audley, S. Babak, *et al.*, "Laser Interferometer Space Antenna," 2017. arXiv: 1702.00786 [astro-ph.IM].

[18] 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*, vol. 11, 2 Jun. 2021. DOI: 10.1103/PhysRevX.11.021053.

[19] LIGO Scientific Collaboration, Virgo Collaboration, and KAGRA Collaboration, "Observation of gravitational waves from two neutron star–black hole coalescences," *The Astrophysical Journal Letters*, vol. 915, no. 1, p. L5, Jun. 2021. DOI: 10.3847/2041-8213/ac082e.

[20] LIGO Scientific Collaboration and Virgo Collaboration, "GW170817: observation of gravitational waves from a binary neutron star inspiral," *Phys. Rev. Lett.*, vol. 119, 16 Oct. 2017. DOI: 10.1103/PhysRevLett.119.161101.

[21] M. Sieniawska and M. Bejger, "Continuous gravitational waves from neutron stars: Current status and prospects," *Universe*, vol. 5, no. 11, 2019. DOI: 10.3390/universe5110217.

[22] J. W. T. Hessels, S. M. Ransom, I. H. Stairs, P. C. C. Freire, V. M. Kaspi, and F. Camilo, "A radio pulsar spinning at 716 hz," *Science*, vol. 311, no. 5769, pp. 1901–1904, 2006. DOI: 10.1126/science.1123430.

[23] N. Christensen, "Stochastic gravitational wave backgrounds," *Reports on Progress in Physics*, vol. 82, no. 1, 2018.

[24] The LIGO Scientific Collaboration, the Virgo Collaboration, and the KAGRA Collaboration, "All-sky search for short gravitational-wave bursts in the third advanced ligo and advanced virgo run," *Phys. Rev. D*, vol. 104, 12 Dec. 2021. DOI: 10.1103/PhysRevD.104.122004.

[25] LIGO Scientific Collaboration, Virgo Collaboration, and KAGRA Collaboration, "All-sky search for long-duration gravitational-wave bursts in the third advanced ligo and advanced virgo run," *Phys. Rev. D*, vol. 104, 10 Nov. 2021. DOI: 10.1103/PhysRevD.104.102001.

[26] M. H. P. M. van Putten, "Directed searches for broadband extended gravitational wave emission in nearby energetic core-collapse supernovae," *The Astrophysical Journal*, vol. 819, no. 2, p. 169, Mar. 2016. DOI: 10.3847/0004-637X/819/2/169.

[27] S. Dall'Osso, B. Giacomazzo, R. Perna, and L. Stella, "Gravitational waves from massive magnetars formed in binary neutron star mergers," *The Astrophysical Journal*, vol. 798, no. 1, p. 25, Dec. 2014. DOI: 10.1088/0004-637X/798/1/25.

[28] E. A. Huerta, C. J. Moore, P. Kumar, *et al.*, "Eccentric, nonspinning, inspiral, gaussian-process merger approximant for the detection and characterization of eccentric binary black hole mergers," *Phys. Rev. D*, vol. 97, 2 Jan. 2018. DOI: 10.1103/PhysRevD.97.024031.

[29] S. Woosley and J. Bloom, "The supernova–gamma-ray burst connection," *Annual Review of Astronomy and Astrophysics*, vol. 44, no. Volume 44, 2006, pp. 507–556, 2006. DOI: https://doi.org/10.1146/annurev.astro.43.072103.150558.

[30] A. Corsi and P. Mészáros, "Gamma-ray burst afterglow plateaus and gravitational waves: Multi-messenger signature of a millisecond magnetar?" *The Astrophysical Journal*, vol. 702, no. 2, Aug. 2009. DOI: 10.1088/0004-637X/702/2/1171.

[31] M. H. P. M. van Putten, "Proposed source of gravitational radiation from a torus around a black hole," *Phys. Rev. Lett.*, vol. 87, 9 Aug. 2001. DOI: 10.1103/PhysRevLett.87.091101.

[32] P. Geurts, L. Wehenkel, and F. d'Alché-Buc, "Gradient boosting for kernelized output spaces," *ACM International Conference Proceeding Series*, vol. 227, Jun. 2007. DOI: 10.1145/1273496.1273533.

[33] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96, Portland, Oregon: AAAI Press, 1996, pp. 226–231.

[34] R. J. G. B. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Advances in Knowledge Discovery and Data Mining*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 160–172.

[35] L. McInnes, J. Healy, and S. Astels, "Hdbscan: Hierarchical density based clustering," *The Journal of Open Source Software*, vol. 2, no. 11, p. 205, 2017.

[36] S. J. Prince, *Understanding Deep Learning*. The MIT Press, 2023. [Online]. Available: http://udlbook.com.

[37] V. Dumoulin and F. Visin, *A guide to convolution arithmetic for deep learning*, 2018. arXiv: 1603.07285 [stat.ML].

[38] V. Dumoulin and F. Visin, *A guide to convolution arithmetic for deep learning*, https://github.com/vdumoulin/conv_arithmetic/tree/master, Accessed: 2024-05-15, 2016.

[39] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. arXiv: 1512.03385 [cs.CV].

[40] K. He, G. Gkioxari, P. Dollár, and R. Girshick, *Mask R-CNN*, 2018. arXiv: 1703.06870 [cs.CV].

[41] O. Ronneberger, P. Fischer, and T. Brox, *U-Net: Convolutional networks for biomedical image segmentation*, 2015. arXiv: 1505.04597 [cs.CV].

[42] G. W. O. S. Center, *Audio samples of GWTC-1*, https://gwosc.org/audiogwtc1/, Accessed: 2024-05-15, 2024.

[43] M. Maggiore, *Gravitational Waves: Volume 1: Theory and Experiments*. Oxford University Press, Oct. 2007. DOI: 10.1093/acprof:oso/9780198570745.001.0001.

[44] M. Zevin, S. Coughlin, S. Bahaadini, *et al.*, "Gravity spy: Integrating advanced ligo detector characterization, machine learning, and citizen science," *Classical and Quantum Gravity*, vol. 34, no. 6, Feb. 2017. DOI: 10.1088/1361-6382/aa5cea.

[45] J. Glanzer, S. Banagiri, S. B. Coughlin, *et al.*, "Data quality up to the third observing run of advanced ligo: Gravity spy glitch classifications," *Classical and Quantum Gravity*, vol. 40, no. 6, Feb. 2023. DOI: 10.1088/1361-6382/acb633.

[46] O. Serra, F. Martins, and L. Padovese, "Active contour-based detection of estuarine dolphin whistles in spectrogram images," *Ecological Informatics*, vol. 55, 2020. DOI: https://doi.org/10.1016/j.ecoinf.2019.101036.

[47] Y. Sato, S. Nakajima, N. Shiraga, *et al.*, "Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images," *Medical Image Analysis*, vol. 2, no. 2, pp. 143–168, 1998. DOI: 10.1016/s1361-8415(98)80009-1.

[48] C. Jin, M. Kim, S. Jang, and D.-G. Paeng, "Semantic segmentation-based whistle extraction of indo-pacific bottlenose dolphin residing at the coast of jeju island," *Ecological Indicators*, vol. 137, 2022. DOI: https://doi.org/10.1016/j.ecolind.2022.108792.

[49] P. Virtanen, R. Gommers, T. E. Oliphant, *et al.*, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.

[50] V. Boudart and M. Fays, "Machine learning algorithm for minute-long burst searches," *Phys. Rev. D*, vol. 105, 8 Apr. 2022. DOI: 10.1103/PhysRevD.105.083007.

[51] F. Xing, Y. Xie, X. Shi, P. Chen, Z. Zhang, and L. Yang, "Towards pixel-to-pixel deep nucleus detection in microscopy images," *BMC Bioinformatics*, vol. 20, no. 1, p. 472, Sep. 2019. DOI: 10.1186/s12859-019-3037-5.

[52] J.-C. Yen, F.-J. Chang, and S. Chang, "A new criterion for automatic multilevel thresholding," *IEEE Transactions on Image Processing*, vol. 4, no. 3, pp. 370–378, 1995. DOI: 10.1109/83.366472.

[53] S. van der Walt, J. Nunez-Iglesias, J. L. Schönberger, *et al.*, "Scikit-image: Image processing in python," *PeerJ*, vol. 2, e453, 2014. DOI: 10.7717/peerj.453.

[54] A. Paszke, S. Gross, F. Massa, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, Curran Associates, Inc., 2019, pp. 8024–8035.

[55] L. N. Smith, *Cyclical learning rates for training neural networks*, 2017. arXiv: 1506.01186 [cs.CV].

[56] K. Krishnamoorthy, "A more powerful test for comparing two poisson means," *Journal of Statistical Planning and Inference*, vol. 119, pp. 23–35, Jan. 2004. DOI: 10.1016/S0378-3758(02)00408-1.

[57] A. L. Piro and E. Thrane, "Gravitational waves from fallback accretion onto neutron stars," *The Astrophysical Journal*, vol. 761, no. 1, p. 63, Nov. 2012. DOI: 10.1088/0004-637x/761/1/63.

[58] M. Yeung, E. Sala, C.-B. Schönlieb, and L. Rundo, "Unified focal loss: Generalising dice and cross entropy-based losses to handle class imbalanced medical image segmentation," *Computerized Medical Imaging and Graphics*, vol. 95, 2022. DOI: https://doi.org/10.1016/j.compmedimag.2021.102026.

[59] M. S. Hossain, J. M. Betts, and A. P. Paplinski, "Dual focal loss to address class imbalance in semantic segmentation," *Neurocomputing*, vol. 462, pp. 69–87, 2021. DOI: https://doi.org/10.1016/j.neucom.2021.07.055.

[60] A. Caporali, K. Galassi, R. Zanella, and G. Palli, "Fastdlo: Fast deformable linear objects instance segmentation," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9075–9082, 2022. DOI: 10.1109/LRA.2022.3189791.