PAPER • OPEN ACCESS

An autoencoder neural network integrated into gravitational-wave burst searches to improve the rejection of noise transients

To cite this article: Sophie Bini *et al* 2023 *Class. Quantum Grav.* **40** 135008

OPEN ACCESS
IOP Publishing

Classical and Quantum Gravity

Class. Quantum Grav. 40 (2023) 135008 (18pp)

https://doi.org/10.1088/1361-6382/acd981

An autoencoder neural network integrated into gravitational-wave burst searches to improve the rejection of noise transients

Sophie Bini[1,2,*], **Gabriele Vedovato**[3], **Marco Drago**[4,5], **Francesco Salemi**[1,2] **and Giovanni A Prodi**[2,6]

[1] Dipartimento di Fisica, Università di Trento, I-38123 Povo, Trento, Italy
[2] INFN, Trento Institute for Fundamental Physics and Applications, I-38123 Povo, Trento, Italy
[3] Dipartimento di Fisica e Astronomia, Università di Padova, I-35131 Padova, Italy
[4] Università di Roma La Sapienza, I-00185 Roma, Italy
[5] INFN, Sezione di Roma, I-00185 Roma, Italy
[6] Dipartimento di Matematica, Università di Trento, I-38123 Povo, Trento, Italy

E-mail: sophie.bini@unitn.it

Received 10 March 2023; revised 5 May 2023
Accepted for publication 26 May 2023
Published 7 June 2023

Abstract

The gravitational-wave (GW) detector data are affected by short-lived instrumental or terrestrial transients, called 'glitches', which can simulate GW signals. Mitigation of glitches is particularly difficult for algorithms which target generic sources of short-duration GW transients (GWT), and do not rely on GW waveform models to distinguish astrophysical signals from noise, such as coherent WaveBurst (cWB). This work is part of the long-term effort to mitigate transient noises in cWB, which led to the introduction of specific estimators, and a machine-learning based signal-noise classification algorithm. Here, we propose an autoencoder neural network, integrated into cWB, that learns transient noises morphologies from GW time-series. We test its performance on the glitch family known as 'blip'. The resulting sensitivity to generic GWT and binary black hole mergers significantly improves when tested on LIGO detectors data from the last observation period (O3b). At false alarm rate of one event per 50 years the sensitivity volume increases up to 30% for

[*] Author to whom any correspondence should be addressed.

signal morphologies similar to blip glitches. In perspective, this tool can adapt to classify different transient noise classes that may affect future observing runs, enhancing GWT searches.

Keywords: gravitational-wave bursts, autoencoder neural network, transient noises

(Some figures may appear in colour only in the online journal)

1. Introduction

Gravitational-wave (GW) interferometers detected 90 astrophysical signals originating from the coalescence of compact objects, mainly black holes (BH) but also neutron stars [1–3]. These exceptional observations have been achieved thanks to Advanced LIGO detectors [4] and Advanced Virgo [5], that together with KAGRA [6] and GEO600 [7] constitute the global network of GW detectors of the LIGO–Virgo–KAGRA (LVK) collaboration.

One of the major challenges for both detector and data-analysis experts is represented by short-duration disturbances, usually referred to as glitches, that are present in GW data with both high signal-to-noise ratio (SNR) and high rate. Short-duration noises are particularly concerning because they can mimic GWTs originated for example by the coalescence of compact binaries, especially in the case of high mass binary BH (BBH) mergers, or other sources still not-detected as supernovae [8], isolated neutron stars [9, 10], cosmic strings [11] GW nonlinear memory [12] and radiation-driven BBH capture events [13]. Moreover, transient noises might overlap with true astrophysical signals and affect the estimation of the parameters of the GW signals [14], their sky localization [15] and the studies performed to test the General Relativity [16]. These transients noises are caused by the instrument itself or by its interaction with the environment [17, 18]. Ideally, the best strategy to reduce their impact is to track back their origin and remove the causes [19]. When it is not possible to resolve the root cause, but the coupling between the noise source and the detector output is known and reproducible, periods of data on the order of seconds or hours can be excluded. On the other side, if the coupling has not completely been identified, the transient noises cannot be vetoed safely, and data analysis algorithms design specific methodologies to minimize their impact (see section 2 and appendix A).

In this work we focus on model-agnostic algorithms, which do not assume any GW templates and are open-wide to generic GWTs in multiple detectors data, and so are more affected by transient noises. We introduce a deep-learning algorithm to classify specific glitch morphologies and mitigate their impact in generic GWT searches. The algorithm proposed consists of a neural network architecture, called *autoencoder*, that learns the morphology of a specific class of glitch from the time-series. When it is applied to a generic detected event, it estimates the similarity between the tested event and the noise class learned. We test this methodology on blip glitches, the family which most affected the GWT search for advanced detectors [1, 17]. A crucial aspect of this methodology is that the network is trained only on examples of transient noises present in the detector data, and no GW signal model needs to be considered. This requirement is determined by the desire to apply the proposed method to generic GWT searches.

Here, we implement the autoencoder neural network on coherent WaveBurst (cWB) [20, 21], a weakly–modelled algorithm used also by LVK collaboration for generic GWT detection and reconstruction [22–29]. Over the years, different strategies to mitigate the impact of transient noises have been integrated into cWB: two estimators, called $Qveto_0$ and $Qveto_1$, have been designed to pinpoint short-duration glitches, and recently cWB has been enhanced by a signal-noise classification with the decision-tree learning algorithm XGBoost [30]. This latter methodology exploits a set of summary statistics computed by cWB, and has shown to increase the search sensitivity for compact binary coalescence and generic GW searches [30–32]. Here, we propose an additional estimator, computed using the autoencoder neural network, that can be straightforwardly included in the statistics used to build the XGBoost model, further enhancing the discrimination of glitches.

The content of the following sections is the following: section 2 describes in more detail short-duration glitches and the investigation performed to find their origins, section 3 introduces cWB and reviews the strategy adopted so far to mitigate transient noises. Section 4 outlines the autoencoder neural network, its architecture, and the training and test datasets. The results obtained with the introduction of this methodology on ad-hoc waveforms and on BBH simulations are reported in section 5.

2. Transient noises in gravitational-wave data

During the third observing run (O3), which took place from April 2019 to March 2020, the median rate of glitches with SNR > 6.5 was about 0.3 min^{-1} in LIGO Hanford, 1 min^{-1} in LIGO Livingston and 0.8 min^{-1} in Virgo, with an increase from November 2019 to January 2020 due to adverse weather conditions [2, 3].

Several methods based on machine-learning techniques have been developed in the latest years to characterize and mitigate transient noises [33]. Many studies propose glitch classification into families according to their morphology in time-series [34, 35] or in time-frequency representations [36, 37]. Classification is crucial to characterize transient noises and identify their root causes: it allows ranking by quantity and characteristics, improving the production of specific vetoes. A successful citizen-science project for supervised classification of GW transient (GWT) noises is GravitySpy [38], which couples a neural network together with human classification performed by citizen scientists. Recently, it has been joined by the project GWitchHunter, more oriented to Virgo glitches [39].

These studies indicate blip glitches as one of the most concerning classes [40–43]: typically, they have a sub-second duration $\mathcal{O}(10)$ ms and a large frequency bandwidth $\mathcal{O}(100)$ Hz. Their rate during the second observing run (O2) was typically of 2 per hour in LIGO, increased to 4 per hour in LIGO Livingston during the third observing run (O3) [17]. Blip glitches appear to have multiple subclasses that might have different origins. Despite automated algorithms that correlate the detector output with the auxiliary channels, used to monitor the state of the instruments and their environment [44], their origin is still largely unknown. Four subsets of blip have been found correlated respectively with humidity, laser intensity stabilization, computer errors and power recycling cavity controls, but these correlations regard a minority of the total number of classified blip ($\sim$8% in LIGO Hanford and $\sim$2% in LIGO Livingston during the first and second observing runs) [40]. Possible correlations with cosmic rays or errors in the data acquisition system have also been investigated, but no evidence was found [17].

Blip glitches are responsible for a major fraction of the unvetoed high SNR triggers, reducing the effectiveness of GW searches. Thus, specific techniques have to be adopted by each GW data analysis algorithm. For example, the template-based algorithm PyCBC [45] has implemented a specific methodology to mitigate the impact of blip glitches on the search for high mass BBH mergers: while the GW models for these sources might not be distinguishable from blip at first sight, blips actually have an excess of power at middle to high frequencies, which is not belonging to the GW template. To highlight this discrepancy, a consistency test between the high mass BBH waveforms and the transient noises has been introduced in PyCBC [46], leading to a significant improvement in the algorithm performance.

In the next sections, we discuss the strategies implemented in cWB to mitigate the impact of transient noises, and we propose a deep-learning algorithm which targets blip glitches.

3. Transient noises mitigation in coherent WaveBurst

CWB is an algorithm widely used by the LVK collaboration for detection and reconstruction of generic GWT, including signals from compact binary coalescences. cWB combines coherently the detector network time-frequency maps, computed with multi-resolution Wilson–Daubechies–Meyer wavelet transform [47], and maximizes a likelihood ratio statistic over all sky directions. To characterize each detected event, several summary statistics are estimated, as the coherence across the detector network, the signal strength, the peak frequency and the duration. cWB also provides a reconstructed waveform of the detected events for each detector.

To reject the events arising from non-stationary detector noise, cWB computes the coherent energy E_C in the detector network and the residual noise energy E_N, estimated subtracting the reconstructed waveform from the data [20]. The events with network correlation coefficient $cc = E_\mathrm{C}/(E_\mathrm{C}+E_\mathrm{N})$ below a certain threshold are rejected. However, due to the high rate of single detector glitches, there is a non-negligible probability of having accidental coincidences in multiple detectors between independent transient noises. Moreover, glitches which occur in a single detector could match part of the noise in the other detectors, especially in the case of glitches with a simple morphology, as blips. An example of the reconstruction provided by cWB of a noise event is reported in figure 1.

For these reasons, model-agnostic methods are strongly affected by short-duration disturbances and the requirement on the correlation coefficient cc is not enough to mitigate transient noises. To reduce the false alarm rate an extra step is necessary. In cWB a separation between GW signal and noise was implemented based on two statistics, referred to as $Qveto_0$ and $Qveto_1$, computed from the reconstructed waveforms (appendix A).

Recently, to automate the signal-noise separation and avoid the application of hard thresholds, a procedure based on a decision tree learning algorithm, called XGBoost [48], has been implemented. XGBoost performs a binary classification between GW signals and noise learning the differences between the population of the signal and of the noise from a list of eight cWB summary statistics that do not depend on the waveform morphology [30–32]. The signal population is modeled using generic white noise burst (WNB) waveforms, which are basically random noise constrained in a certain time-frequency range sampled from a random distribution. To represent the noise population we employ the time-shift analysis: the data of one detector is shifted with respect to the other detector so that the coincident events by construction do not have an astrophysical origin, but they are solely due to non-stationary detector

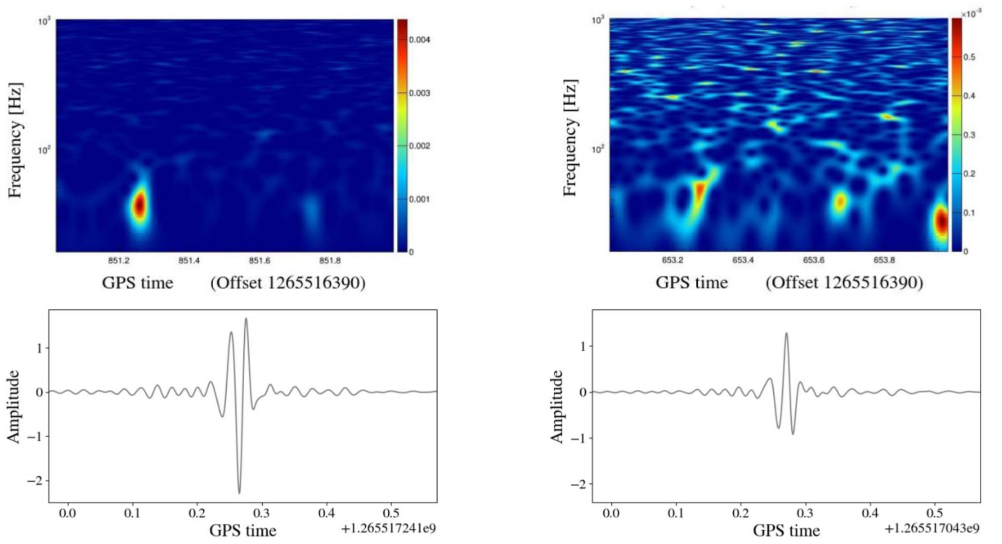

Figure 1. Example of the spectrograms (top) and respective time-series (bottom) of a noise event detected by cWB in LIGO time-shifted data (see section 3). The spectrogram on the left (LIGO Livingston) shows clearly a blip glitch with reconstructed SNR = 12. On the right (LIGO Hanford) the spectrogram shows a weaker disturbance (SNR = 6) with multiple low frequency spots (note that the colour scales are different for the two spectrograms). In both cases, the reconstructed time-series show a blip-like morphology.

noises. The cWB events detected on time-shifted data are referred to as *background* events. The XGBoost model computes a penalty factor which ranges from 0 for noise and 1 for signal, and it updates the cWB ranking statistic ρ [31].

4. Further discrimination of transient noises by an autoencoder neural network

We implement an autoencoder neural network to classify transient noises from cWB reconstructed waveforms. An autoencoder is an unsupervised learning neural network that compresses the input data into a lower dimensional space, called *latent space*, and then reconstructs an output with the original dimension. Performing the compression, the autoencoder highlights the presence of structures in the input data, and disregards redundancies in the data. The autoencoder architecture is used in GW physics for features extraction, data denoising and anomaly detection [49–53].

Here, the autoencoder performs an anomaly detection task: the training dataset contains time-series examples belonging to a single class, so that the network learns that specific morphology. Once the autoencoder is applied to a time-series with a different signature, the *anomaly*, the network struggles to reconstruct it properly. The goodness of the reconstruction is evaluated through the mean square error (MSE) between the input time-series ($X_{i,\mathrm{input}}$) and the one reconstructed by the autoencoder ($X_{i,ae}$):

$$\mathrm{MSE} = \sum_{i=0}^{n} (X_{i,\mathrm{input}} - X_{i,ae})^2 \tag{1}$$

where n is the time-series length. The MSE is computed for each detector, and when searching for GWTs it is weighted according to the SNR square of the event in each detector. In the following, the weighted MSE estimator will be referred to as the autoencoder statistics.

4.1. The architecture

An autoencoder network learns the key features of the input data, which in this case is a time-series x_i with n data points. The network consists of two parts: an encoder $f_{\mathrm{E}}(\cdot)$ that compresses the input representation into a lower dimension latent space, and a decoder $g_{\mathrm{D}}(\cdot)$ that converts the latent representation to the original format. The encoder and decoder weights, E and D, are found by minimizing the difference between the input x_i and the autoencoder output $g_{\mathrm{D}}(f_{\mathrm{E}}(x_i))$. The error function J is the mean square errors between the input and the output:

$$J(E,D;x_i) = \frac{1}{n} \sum_{i=1}^{n} ||g_{\mathrm{D}}(f_{\mathrm{E}}(x_i)) - x_i||^2. \tag{2}$$

At the beginning of the training procedure the weights are set randomly, then they are updated minimizing equation (2) for each x_i present in the training dataset. In other words, the weights are updated so that $g_{\mathrm{D}}(f_{\mathrm{E}}(x_i)) \sim x_i$: the network searches for an approximation to the identity function but, as the latent space has a lower dimension than the input data, the algorithm is forced to learn a compressed representation. The autoencoder performs a dimensional reduction, but differently from common tools as principal component analysis [54], can learn non-linear and complex features.

Once trained, the weights E and D are fixed and the network is able to reconstruct properly a time-series only if it has a morphology similar to the examples present in the training dataset. The goodness of the reconstruction is estimated with the MSE (equation (1)): GW time-series with low MSE values are similar to Blip glitches, while higher MSE values indicate different morphologies. Other anomaly detection methodologies, that can be considered for future developments, exploit directly the latent space distribution learned by the autoencoder. This could be accomplished either by clustering the latent space [55] or by studying the latent space distribution with normalizing flows [56].

We build the network using the deep-learning application programming interface Keras [57]. The autoencoder is made of multiple convolutional layers, one of the most recognized machine-learning algorithm to extrapolate relevant features in time-series and images. More details on the architecture of the network are reported in the appendix B.

A crucial aspect of the architecture is the latent space dimension: the lower the dimension, the stronger the compression of the input, and the less information is retained by the network. Test at different compression factors showed that for higher compression the network learns only loud blip-like morphology, and it struggles when the combination of the glitch with the detector noise leads to slightly different morphology, resulting in a poor ability to mitigate the background events.

4.2. The autoencoder input data

The inputs x_i of the autoencoder are cWB reconstructed time-series which, thanks to the whitening procedure[7] and the selection of the most energetic wavelets in the time-frequency maps, appear much cleaner than the raw GW detector data. Before being processed by the autoencoder, each cWB reconstructed time-series, sampled at 2048 Hz, is windowed to 416 data points (corresponding to $\sim$0.2 s), and centred around the absolute maximum value. Several window lengths have been tested from about 200 to 800 data points, but this choice is a good compromise between containing the entire blip-like evolution and minimizing the information to be learnt. Next, the time-series is normalized in amplitude between $[0, 1]$, as suitable for neural networks. Some examples of cWB reconstructed waveforms processed to be input to the autoencoder are shown in figure 2.

4.3. The training dataset

The training dataset is made of blip glitches according to the GravitySpy [38, 43] classification. To retrieve the blip time-series as observed by cWB, we run the pipeline in each single detector, obtaining a list of detected events, glitches and eventually GW signals, in each detector. Next, we select blip glitches comparing the GPS time of cWB single detector events and GravitySpy blip GPS time. We employ two similar glitch families classified by GravitySpy, nominally *blip* and *tomte*.

Deep-learning methods benefit from the largest possible training dataset, and in order to collect more training samples several strategies have been developed, usually referred to as *data augmentation* techniques. In this work, we double the amount of the training dataset including also the vertical flip of each time-series. Moreover, we add gaussian noise to each input data to improve the capability of the autoencoder network to pinpoint also low SNR glitches. The resulting training dataset is composed of 4608 glitches occurred during the second period of the third observing run (O3b). Additional 512 classified samples constitute the validation dataset, which tests that the neural network is not over-fitting. We retrieve blip glitches from the two LIGO detectors, but there is evidence for the presence of such glitches also in Virgo detector [40], whose transient noises are also classified by GravitySpy. Figure 2 shows two examples of time-series present in the validation dataset and the reconstruction achieved by the autoencoder.

4.4. Signal models for testing

We test the autoencoder neural network on three sets of waveforms. The first is made of ad-hoc signals, generally used in generic GWT searches to evaluate the sensitivity of a weakly modelled pipeline over a wide parameter space [29]. They include Sine-Gaussian (SG), Gaussian pulse (GA) and WNB. The SG waveforms are characterized by the central frequency, and the quality factor Q. SG refers to circularly polarized Sine-Gaussian, while SGE are elliptically polarized. The GA signals are characterized by the duration, and the WNB by their bandwidth, duration and lower frequency bound.

[7] https://gwburst.gitlab.io/documentation/latest/html/faq.html#the-whitening.

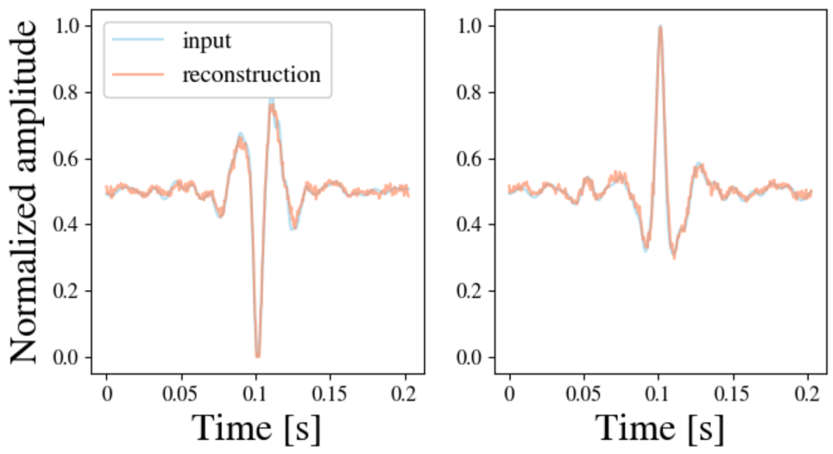

Figure 2. Two examples of blip time-series according to the GravitySpy classification detected by cWB in LIGO Hanford. In blue the autoencoder inputs x_i, that are cWB reconstructed waveforms windowed and normalized as described in section 4.2). In orange the autoencoder reconstructions $g_{\mathrm{D}}(f_{\mathrm{E}}(x_i))$.

The second set of waveforms contains cosmic strings simulations, potential burst sources supposed to originate after a spontaneous phase transition in the early Universe [11]. Here, we inject cosmic strings from cusps [58], which are characterized by the amplitude, and frequency range of [1 Hz, 1500 Hz]. These waveforms have been included to test the robustness of the proposed methodology on potential GW signals with a morphology similar to blip glitches. The results achieved cannot be compared directly to the LVK search [11], because the detection efficiency is parametrized differently.

The third set of waveforms is composed of BBH coalescences with quasi circular orbits, as they represent the GW signals mainly observed so far. Their waveforms include both precession and higher order modes, and they are computed using `SEOBNRv4PHM` model [59].

5. Results

We compute the autoencoder statistic, defined in equation (1) as the MSE between the cWB reconstructed waveform of each detected event and the corresponding time-series reconstructed by the autoencoder, for the background events and the injected signals. A low MSE suggests that the event considered has a blip-like signature, while a high MSE indicates a different morphology. We include the obtained MSE in the list of summary statistics used by XGBoost to perform the signal-noise separation in cWB. This configuration will be referred as `XGBoost` $+$ `AE` model, while the configuration without the autoencoder statistic will be referred simply as `XGBoost` model. The cWB pipeline's set up and the hyperparameters employed for the XGBoost tuning are equal for the two configurations, and the same used for the generic GWT search performed with cWB enhanced by XGBoost, which provides the most stringent constraints on the isotropic emission of GW energy from burst sources to date [32]. Using these two configurations, we analyse 40 days, between February and March 2020, of coincident data between the LIGO detectors. We accumulate about 380 years of background using the time-shift analysis (section 3), 70% of which is used to train the XGBoost model and the remaining

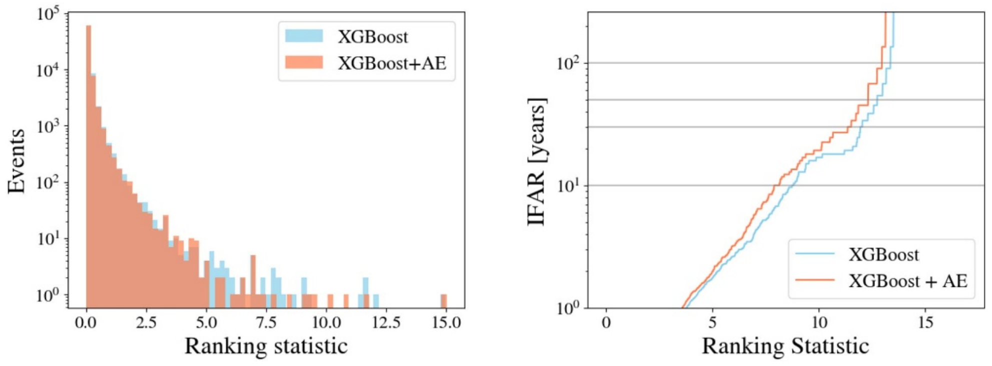

Figure 3. (Left) Background events versus the ranking statistic ρ for the `XGBoost` model (blue) and the `XGBoost + AE` one (orange). (Right) IFAR versus the ranking statistic for the background events for the two models. The `XGBoost + AE` model reduces the number of background events at $\rho > 5$, so that at a fixed IFAR threshold the corresponding ρ is lower. The gray lines mark the IFAR thresholds for which the search sensitivity is reported in figures 4 and 5.

30% for testing. Similar results are obtained also with different percentages dedicated to the training and the testing.

First, we discuss the impact of the autoencoder statistic on the background events distribution (section 5.1), then we report the search sensitivity achieved on ad-hoc waveforms, cosmic strings, and BBH simulations (section 5.2).

5.1. *Background events distribution*

The background events distribution is crucial to assess the significance of the detected events in terms of the inverse false alarm rate (IFAR), where IFAR = 1/FAR. The FAR of a detected event with ranking statistic ρ_k is:

$$\mathrm{FAR} = \frac{N(\rho_k)}{T} \tag{3}$$

where $N(\rho_k)$ is the number of background events with $\rho > \rho_k$ and T the accumulated background time. So, the fewer the background events and the lower their ρ, the more sensitive the search for GWT will be. In figure 3 (left) the background events are shown versus the cWB ranking statistic ρ. The autoencoder further mitigates the background distribution, with 28 events at $\rho > 5$ using the `XGBoost + AE` model, rather than 47 events with the `XGBoost` model. This enhancement can be appreciated in figure 3 (right), which shows the IFAR versus the ranking statistics for the background distribution for the two models considered. At a fixed ρ, the inclusion of the autoencoder's statistic increases the corresponding IFAR, meaning that a potential GW signal detected by cWB with a certain ρ can have a higher significance with the `XGBoost + AE` model, or in other words that using the autoencoder we can assing the same IFAR to weaker GW signals.

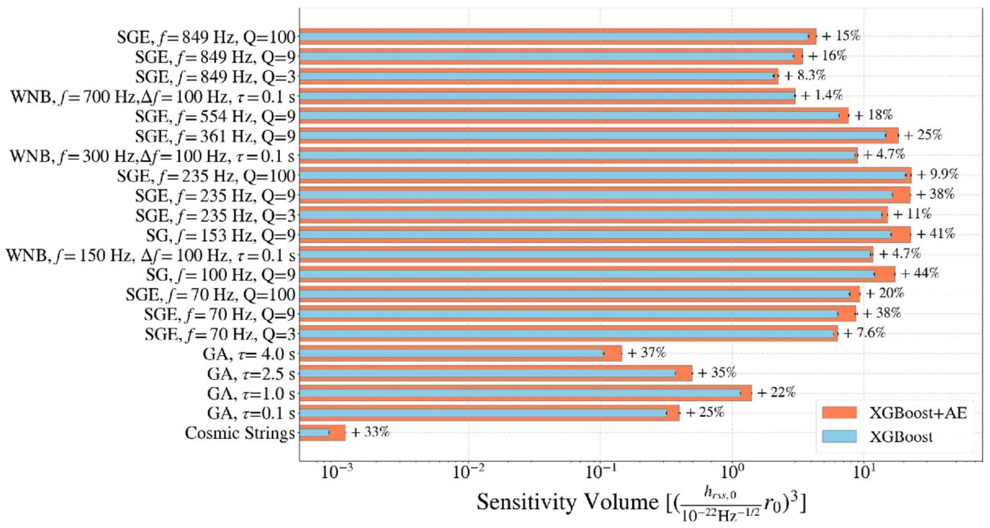

Figure 4. Sensitivity volume obtained with cWB with autoencoder feature included in the XGBoost model (`XGBoost + AE`) in orange, and without it (`XGBoost`) in blue, at IFAR = 50 years. The values next to each bar show the percentage improvement w.r.t to the volume obtained with `XGBoost` model. From bottom to top, the waveforms are: cosmic strings, Gaussian Pulse (GA) characterized by the duration τ, then ordered according to frequency Sine Gaussian (SG) and SG elliptically polarized (SGE) characterized by central frequency f, and the quality factor Q and white noise burst (WNB) with bandwidth Δf, duration τ and lower frequency bound f.

5.2. *Search sensitivity for ad-hoc waveforms and BBH simulations*

To evaluate the sensitivity of generic searches, simulated signals are injected into the detector data. The root-squared-sum (rss) strain amplitude of the signal is defined as:

$$h_{\mathrm{rss}} = \sqrt{\int_{-\infty}^{+\infty} [h_+^2(t) + h_\times^2(t)]dt} \tag{4}$$

where h_+ and $h_\times$ are the plus and cross polarization of the GW signal. Varying the injection amplitude, it is possible to compute the detection efficiency $\epsilon(h_{\mathrm{rss}})$ as the ratio between the signals detected with an IFAR over a certain threshold and the total amount of injections. From the latter, it is possible to compute the h_{rss50}, i.e. the strain amplitude at which 50% of the injections are recovered, which is a typical metric to express the sensitivity in GW bursts searches.

A cumulative metric is the sensitivity volume $\mathcal{V}$ [22, 32], defined as:

$$\mathcal{V} = 4\pi\,(h_{\mathrm{rss},0} r_0)^3 \int_0^\infty \frac{dh_{\mathrm{rss}}}{h_{\mathrm{rss}}^4}\epsilon(h_{\mathrm{rss}}), \tag{5}$$

where $h_{\mathrm{rss},0}$ is a reference amplitude value at a nominal distance r_0. This metric, having a factor h_{rss}^4 to the denominator, emphasizes the contribution to the sensitivity of the weaker signals.

Figure 4 shows the results for a wide set of ad-hoc waveforms and cosmic strings with an IFAR >50 years. The sensitivity volume obtained with the inclusion of the autoencoder is higher for all the waveforms tested: for cosmic strings the improvement is 33%, for GA waveforms between 22%–37%, for SG is 8%–44% and for WNB is 1.4%–4.7 %. Moreover,

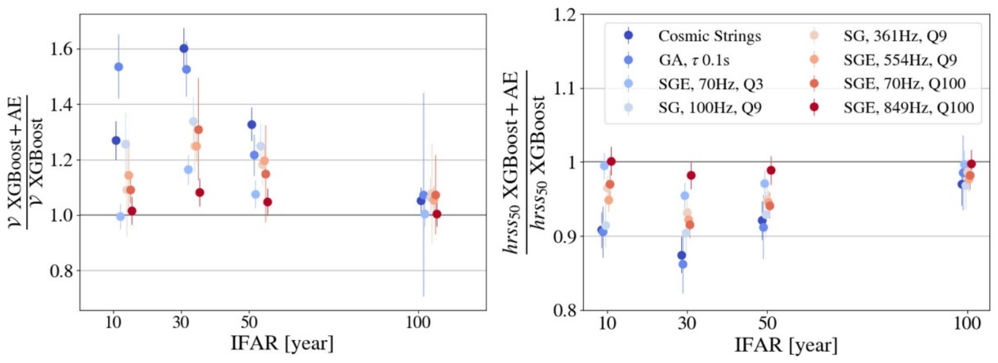

Figure 5. Ratio between the sensitivity volume $\mathcal{V}$ (left) and h_{rss50} (right) obtained including the autoencoder (`XGBoost + AE`) and without using it (`XGBoost`) at different IFAR thresholds (10, 30, 50, 100 years). Data points are slightly shifted around the IFAR thresholds to avoid overlaps. The waveforms are: cosmic strings, Gaussian Pulse (GA) characterized by the duration τ, then Sine Gaussian (SG) characterized by central frequency f, and the quality factor Q and white noise burst (WNB) with bandwidth Δf, duration τ and lower frequency bound f.

we investigate the performance of the two XGBoost models considered over different IFAR thresholds (10, 30, 50, 100 years) in figure 5.[8] The improvement in sensitivity volume and in h_{rss50} obtained with the autoencoder statistic is remarkable also at lower IFAR thresholds, and it is more evident on the waveforms that have a morphology similar to blip glitches, as GA, cosmic strings and low frequency SG. At IFAR >100 years, the search sensitivities obtained by the two models are comparable. This regime corresponds to the background distribution region with $\rho > 13$ (figure 3), where there is a single loud glitch which is not affected by the inclusion of the autoencoder. Such background events are rare, and the methodological scope of this work does not justify the computational cost of accumulate more statistic to discuss further this regime. Instead, we consider relevant the improvement achieved at lower IFAR thresholds, as that is the region where most of the GW signals detected so far lies.

In addition, we report the sensitivities achieved for BBH mergers injections in terms of the observed volume V (figure 6) [2]. Given a local merger rate density R, this metric is computed counting the number of detections above a certain IFAR threshold N_{det}, and considering $N_{\mathrm{det}} = \mathrm{VT}R$ where T is the observing time. The volume obtained including the autoencoder is sightly enhanced for all IFAR thresholds, due to the reduction of the background distribution shown in figure 3.

6. Conclusions

In this work, we propose an autoencoder neural network to mitigate the impact of short-duration transient noises, which constitute a major concern for generic GWT searches. The neural network is integrated in cWB, a weakly-modeled algorithm widely adopted in the GW

[8] The error bars on the h_{rss50} values are computed from the detection efficiencies at each injected amplitude. The error on the ratio $h_{rss50,\ \mathtt{XGBoost\ +\ AE}}/h_{rss50,\ \mathtt{XGBoost}}$ is propagated assuming the errors on the two terms of the ratio independent or, in other words, neglecting their strong positive correlation. This results in a very conservative choice as the majority of the events is detected by both XGBoost models. The error bars on the sensitivity volume are computed from the ones on h_{rss} and by propagating equation (5).

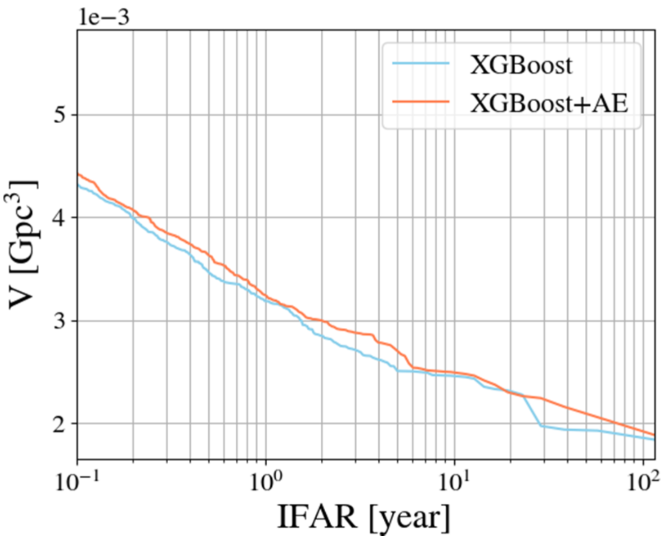

Figure 6. Observed volume at different IFAR for BBH simulations. The volume obtained with the inclusion of the autoencoder (orange) is compared with the model without the autoencoder (blue).

community. Over the years, cWB has designed two estimators to recognize short-duration glitches, and recently it has implemented a more efficient separation of GW signals from noises based on the machine-learning algorithm XGBoost [32]. This work constitutes a further step of this development, and shows to improve generic GWT searches in real operating conditions.

Here, we focus on blip glitches, one of the most common transient noise family in GW data, whose origin is still unknown. We include the autoencoder statistic in the XGBoost model, and we perform injections of ad-hoc waveforms, cosmic strings and BBH simulations in GW data. We report the sensitivity achieved both in terms of sensitivity volume and $h_{\mathrm{rss}50}$. The inclusion of the autoencoder statistic enhances ad-hoc waveforms and cosmic strings at different IFAR thresholds, in particular the most evident enhancement is achieved for the waveforms which have a morphology similar to blip glitches, as short-duration gaussian pulses, sine gaussians and cosmic strings. The search sensitivity for BBH simulations is also slightly enhanced by the addition of the autoencoder statistic.

Here, the methodology is applied to the LIGO detector network, but it could be easily extended to multiple GW detectors. In addition, the autoencoder statistic could be exploited by other signal-noise classification procedures, as the one based on Gaussian Mixture Model recently proposed for cWB [60, 61].

With respect to previous deployed methods, as the *Qveto* parameters, the autoencoder neural network is able to learn also different transient noise classes, if present in the training dataset. This flexibility will be highly valuable as new glitch classes appear in the future GW observing periods.

Data availability statement

The data that support the findings of this study are openly available at the following URL/DOI: https://gwosc.org/archive/O3b_4KHZ_R1/.

Acknowledgments

This material is based upon work supported by NSF's LIGO Laboratory which is a major facility fully funded by the National Science Foundation. The authors are grateful for computational resources provided by the LIGO Laboratory and supported by National Science Foundation Grants PHY-0757058 and PHY-0823459. This research has made use of data, software and/or web tools obtained from the Gravitational Wave Open Science Center, a service of LIGO Laboratory, the LIGO Scientific Collaboration and the Virgo Collaboration.

Appendix A. *Qveto* parameters

cWB calculates several summary statistics to characterize the detected events. Here we focus on two of them, called *Qveto*, introduced to pinpoint short-duration glitches and reviewed here for the first time[9].

The first, $Qveto_0$, estimates the ratio between the energies far and near the maximum peak of the signal. More precisely, once computed the absolute maximum amplitude A_{max}, N_{TH} samples before and after the maximum peak are selected. Their amplitudes are indicated with $A_{i,\mathrm{bef}}$ and $A_{i,\mathrm{aft}}$. Further relative amplitudes $A_{i,\mathrm{rel}}$ are selected if $|A_{i,\mathrm{rel}}| > |A_{\mathrm{max}}|/A_{\mathrm{TH}}$. Figure 7 shows the cWB reconstructed waveform of a transient noise with the relevant amplitudes for the estimation of the $Qveto_0$ highlighted. $Qveto_0$ is then defined as:

$$Qveto_0 = \frac{\sum A_{i,\mathrm{rel}}{}^2}{A^2_{\mathrm{max}} + \sum A^2_{i,\mathrm{bef/aft}}}. \tag{6}$$

The lower the $Qveto_0$, the strongest the peak amplitude compared to the surrounding fluctuations, suggesting a blip-like morphology. N_{TH} and A_{TH} are hard thresholds which are empirically selected looking at glitches reconstructed waveforms. Default values are $N_{\mathrm{TH}} = 1$, and $A_{\mathrm{TH}} = 7.6$. This procedure is applied to each detector independently, then the minimum value is selected. The second parameter, $Qveto_1$, models approximately the reconstructed waveform with a CosGaussian function:

$$\mathrm{CosGaussian}(w, Q) = \cos(\omega t) * \exp\left[\frac{(-\omega t)^2}{2\; Qveto_1{}^2}\right] \tag{7}$$

and the $Qveto_1$ factor is estimated as:

$$Qveto_1 = \sqrt{\frac{-\pi^2}{2\log(R)}}, \text{ with } R = \frac{A_{\mathrm{bef}} + A_{\mathrm{aft}}}{2A_{\mathrm{max}}} \tag{8}$$

where A_{bef} (A_{aft}) is the absolute value of the maximum peak before (after) the main peak. The $Qveto_1$ is computed for each detector and weighted according to the SNR square of the detected event in each detector.

On LVK publications based on the data from the first, second and third observing runs (O1–O2–O3) [62–64], *Qveto* parameters were employed in cWB to split the detected events into multiples bins: a 'clean' bin and one or more 'dirty' bins. Dirty bins were populated by short-duration blip-like events, while the clean bin contains only triggers with *Qveto* parameters above a certain threshold (as an example for the O3 generic GW burst search [29] $Qveto_1 < 3.4$ for the first dirty bin, $Qveto_1 < 3.4$ and $Qveto_0 = 0$ for the second dirty bin, and $Qveto_1 > 3.4$

[9] See also in the cWB documentation: https://gwburst.gitlab.io/documentation/latest/html/faq.html#the-qveto.

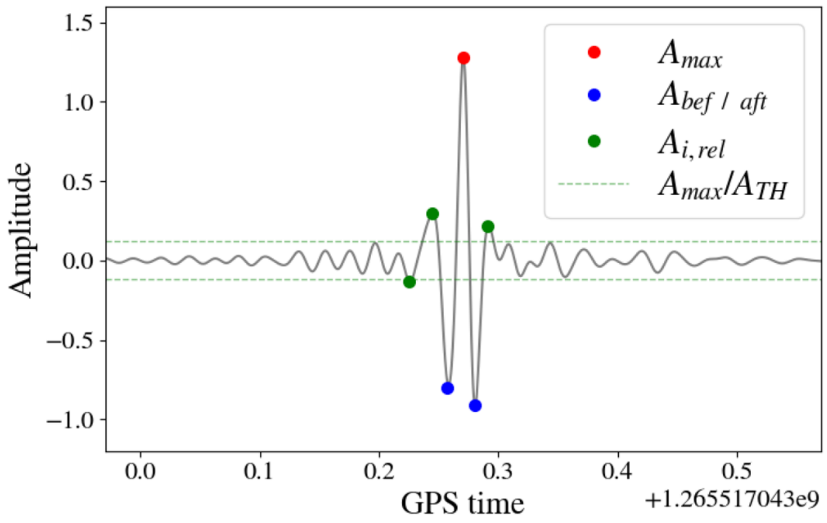

Figure 7. Reconstructed waveform of a transient noise passing in cWB in LIGO Hanford. The coloured dots show the amplitudes used to estimate the $Qveto_0$ according to equation (6). The green lines correspond to the threshold $\pm A_{\mathrm{max}}/A_{\mathrm{TH}}$ over which $A_{i,\mathrm{rel}}$ amplitudes are chosen.

for the clean one). Then, each event was ranked in each bin independently, leading to the introduction of a trial factor equal to the number of bins. Thus, *Qveto* parameters played a key role during the signal-noise separation process.

However, this procedure depends on hard thresholds that had to be tuned manually according to the performance on some set of simulations, and it cannot be generalized to different transient noise morphology that may arise in next observing runs. For these reasons, it has been substituted with a machine-learning based algorithm [30–32] which learns the population of the signal and of the noise from a list of cWB summary statistics. Among the others, this list includes the *Qveto* parameters through the following definitions:

$$Q_a = \sqrt{Qveto_0}\,, \qquad Q_p = \frac{Qveto_1}{2\sqrt{\log_{10}(\min(200, E_{\mathrm{c}}))}} \tag{9}$$

where E_{c} is the coherent energy in the detector network (section 3).

Appendix B. More details on the autoencoder neural network

We report here more details on the methodology implemented. The algorithm is based on deep-learning, a subset of machine-learning, in which a task is learned from a large amount of data. Deep-learning methods consist of a network, i.e. a sequence of layers of algorithms, each of which extrapolate information of the data it is applied to. These networks are usually referred as *neural network* because they are inspired by the human brain processing. The neural network proposed in this works consists of two parts, an encoder that compress the input into a lower dimensional space and a decoder that returns the compressed representation into the original shape. Both the encoder and the decoder are made of multiple layers. The neural networks with this structure are called *autoencoders*. The first layer learns a representation of the input data, and the subsequent layers receive the representation learned by the preceding layers,

Table 1. Autoencoder neural network architecture. Each line represents a layer of algorithm. On the right, there is the output shape of each layer, which is also the input shape for the subsequent layer. For example: the input data is a time-series with 416 data points. Then, a convolutional layer with $k = 128$ filters is applied. The time-series sample rate is 2048 Hz.

Layer	Output shape (length, dimension)
Encoder	
Input	(416, 1)
Convolutional	(416, 128)
Max pooling	(208, 128)
Convolutional	(20, 816)
Max pooling	(104, 16)
Convolutional	(52, 16)
Flatten	(832)
Dense	(200)
Decoder	
Dense	(832)
Reshape	(52, 16)
Convolutional	(52, 16)
Up pooling	(10, 416)
Convolutional	(10, 416)
Up pooling	(208, 128)
Convolutional	(208, 128)
Up pooling	(416, 128)
Convolutional	(416, 1)

revealing more and more complex features. The main layers used are briefly introduced below and summarized in table 1. A review can be found in reference [65].

- *Convolutional layer*: it computes the convolution between the input x and the filters, or *kernels*. Multiple kernels are present in each convolutional layers, and have dimension equal to $(m \times k)$, being m the length of the stride and k the number of kernels applied. The convolutional layer calculates the dot product between the input and the weight W^k and transmits the result of the multiplication to an activation function f. The output of the convolutional layer is:
$$h_{W,b} = f(W^k x + b) \tag{10}$$
where b is a bias term, typically equal to 1. Here, f is a ReLU function [66], which converts the input to positive numbers:
$$f(z) = \max(0, z). \tag{11}$$
The kernels are assigned randomly at the beginning of the training process and are updated minimizing the error function (equation (2)). The filter length m is equal to 3 and slides over the entire input. Usually, multiple filters are used to acquire more complex kinds of features.

- *Max pooling layer*: after a convolutional layer typically there is a pooling layer, that downsamples the convolutional output $h_{W,b}$ picking the maximum value over a spatial window considered. In this case, we have a window equal to 2, meaning that we take the maximum values between two adjacent values. The combination of convolutional layers and max pooling layers is repeated multiple times to extract the most relevant features.

- *Dense layer*: it consists of several basic units, called *neurons*, in which a weight is applied as in equation (10). Each neuron is fully connected to all neurons of the previous layers. In this autoencoder architecture a dense layer is used to compress the output of the decoder layers to the desired latent space dimension, equal to 200 in this case.
- *Up sampling layer*: opposite to max pooling layer up-sample the representation repeating the data by 2.
- *Flatten* and *reshape*: the first flatten the inputs from a shape $x^{a,b}$ to $x^{a\times b}$, the second reshapes the inputs into the given shape.

The main settings used to train the network are: the epochs, i.e. the number of iterations over the entire training dataset, sets to 75, the bach size, i.e. the number of training samples per weights update, that is 16, and the optimization algorithm which updates the network weights minimizing the loss function that is ADAM [67]. The total number of network parameter is 349 513. The training time is about 22 min and the execution time in case of single time-series evaluation is about 0.0032 s using 16-core AMD opteron 6376 CPU.

ORCID iDs

Sophie Bini https://orcid.org/0000-0002-0267-3562
Marco Drago https://orcid.org/0000-0002-3738-2431

References

[1] Abbott B P *et al* (LIGO Scientific Collaboration, Virgo Collaboration) 2019 *Phys. Rev.* X **9** 031040
[2] Abbott R *et al* (LIGO Scientific Collaboration, Virgo Collaboration) 2021 *Phys. Rev.* X **11** 021053
[3] Abbott R *et al* (LIGO Scientific, VIRGO, KAGRA) 2021 GWTC-3: compact binary coalescences observed by LIGO and Virgo during the second part of the third observing run (arXiv: 2111.03606)
[4] Aasi J *et al* (LIGO Scientific Collaboration) 2015 *Class. Quantum Grav.* **32** 074001
[5] Acernese F *et al* (Virgo Collaboration) 2015 *Class. Quantum Grav.* **32** 024001
[6] Aso Y *et al* 2013 *Phys. Rev.* D **88** 043007
[7] Willke B *et al* 2002 *Class. Quantum Grav.* **19** 1377
[8] Abbott B *et al* 2020 *Phys. Rev.* D **101** 084002
[9] Lopez D (The LIGO, Virgo and KAGRA collaborations) 2022 *Ann. Phys., Lpz.* 2200142
[10] Abbott R *et al* 2022 arXiv:2210.10931
[11] Abbott R *et al* 2021 *Phys. Rev. Lett.* **126** 241102
[12] Ebersold M and Tiwari S 2020 *Phys. Rev.* D **101** 104041
[13] Ebersold M, Tiwari S, Smith L, Bae Y B, Kang G, Williams D, Gopakumar A, Heng I S and Haney M 2022 *Phys. Rev.* D **106** 104014
[14] Pankow C *et al* 2018 *Phys. Rev.* D **98** 084016
[15] Macas R, Pooley J, Nuttall L K, Davis D, Dyer M J, Lecoeuche Y, Lyman J D, McIver J and Rink K 2022 *Phys. Rev.* D **105** 103021
[16] Kwok J Y, Lo R K, Weinstein A J and Li T G 2022 *Phys. Rev.* D **105** 024066
[17] Davis D *et al* 2021 *Class. Quantum Grav.* **38** 135014
[18] Acernese F *et al* 2022 arXiv:2205.01555
[19] Schofield R Logbook 2020 (Ligo Laboratory) 52184
[20] Klimenko S, Yakushin I, Mercer A and Mitselmakher G 2008 *Class. Quantum Grav.* **25** 114029
[21] Drago M *et al* 2021 *SoftwareX* **14** 100678
[22] Abadie J *et al* (The LIGO Scientific Collaboration and The Virgo Collaboration) 2012 *Phys. Rev.* D **85** 122007

[23] Abadie J *et al* (The LIGO Scientific Collaboration and The Virgo Collaboration) 2010 *Phys. Rev.* D **81** 102001
[24] Abbott B *et al* (LIGO Scientific Collaboration) 2009 *Phys. Rev.* D **80** 102001
[25] Abbott B *et al* (LIGO Scientific Collaboration) 2006 *Class. Quantum Grav.* **23** S29–S39
[26] Abbott B *et al* (LIGO Scientific Collaboration and TAMA Collaboration) 2005 *Phys. Rev.* D **72** 122004
[27] Abbott B *et al* (LIGO Scientific Collaboration) 2005 *Phys. Rev.* D **72** 062001
[28] Abbott B *et al* (LIGO Scientific Collaboration) 2004 *Phys. Rev.* D **69** 102001
[29] Abbott R *et al* (The LIGO Scientific Collaboration, The Virgo Collaboration and The KAGRA Collaboration) 2021 *Phys. Rev.* D **104** 122004
[30] Mishra T, O'Brien B, Gayathri V, Szczepańczyk M, Bhaumik S, Bartos I and Klimenko S 2021 *Phys. Rev.* D **104** 023014
[31] Mishra T *et al* 2022 *Phys. Rev.* D **105** 083018
[32] Szczepańczyk M J *et al* 2023 *Phys. Rev.* D **107** 062002
[33] Cuoco E *et al* 2020 *Mach. Learn.: Sci. Technol.* **2** 011002
[34] Mukund N, Abraham S, Kandhasamy S, Mitra S and Philip N S 2017 *Phys. Rev.* D **95** 104059
[35] Powell J, Trifiro D, Cuoco E, Heng I S and Cavaglià M 2015 *Class. Quantum Grav.* **32** 215012
[36] George D, Shen H and Huerta E 2018 *Phys. Rev.* D **97** 101501
[37] Razzano M and Cuoco E 2018 *Class. Quantum Grav.* **35** 095016
[38] Zevin M *et al* 2017 *Class. Quantum Grav.* **34** 064003
[39] Razzano M, Di Renzo F, Fidecaro F, Hemming G and Katsanevas S 2023 *Nucl. Instrum. Methods Phys. Res.* A **1048** 167959
[40] Cabero M *et al* 2019 *Class. Quantum Grav.* **36** 155010
[41] Abbott B P *et al* 2016 *Class. Quantum Grav.* **33** 134001
[42] Abbott B P *et al* 2018 *Class. Quantum Grav.* **35** 065010
[43] Glanzer J *et al* 2023 *Class. Quantum Grav.* **40** 065004
[44] Smith J R, Abbott T, Hirose E, Leroy N, Macleod D, McIver J, Saulson P and Shawhan P 2011 *Class. Quantum Grav.* **28** 235005
[45] Usman S A *et al* 2016 *Class. Quantum Grav.* **33** 215004
[46] Nitz A H 2018 *Class. Quantum Grav.* **35** 035016
[47] Necula V, Klimenko S and Mitselmakher G 2012 *J. Phys.: Conf. Ser.* **363** 012032
[48] Chen T and Guestrin C 2016 A scalable tree boosting system *Proc. 22nd ACM Sigkdd Int. Conf. on Knowledge Discovery and Data Mining* pp 785–94
[49] Gabbard H, Messenger C, Heng I S, Tonolini F and Murray-Smith R 2022 *Nat. Phys.* **18** 112–7
[50] Ormiston R, Nguyen T, Coughlin M, Adhikari R X and Katsavounidis E 2020 *Phys. Rev. Res.* **2** 033066
[51] Corizzo R, Ceci M, Zdravevski E and Japkowicz N 2020 *Expert Syst. Appl.* **151** 113378
[52] Morawski F, Bejger M, Cuoco E and Petre L 2021 *Mach. Learn.: Sci. Technol.* **2** 045014
[53] Shen H, George D, Huerta E A and Zhao Z 2019 Denoising gravitational waves with enhanced deep recurrent denoising auto-encoders *2019 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE) pp 3237–41
[54] Bro R and Smilde A K 2014 *Anal. Methods* **6** 2812–31
[55] Bini S 2020 Unsupervised classification of short transient noise to improve gravitational wave detection *Master's Thesis* Università di Pisa
[56] Rezende D and Mohamed S 2015 Variational inference with normalizing flows *Int. Conf. on Machine Learning* (PMLR) pp 1530–8
[57] Chollet F 2015 Keras, version 2.11.0 (available at https://github.com/fchollet/keras)
[58] Damour T and Vilenkin A 2005 *Phys. Rev.* D **71** 063510
[59] Ossokine S *et al* 2020 *Phys. Rev.* D **102** 044055
[60] Gayathri V, Lopez D, Pranjal R S, Heng I S, Pai A and Messenger C 2020 *Phys. Rev.* D **102** 104023
[61] Lopez D, Gayathri V, Pai A, Heng I S, Messenger C and Gupta S K 2022 *Phys. Rev.* D **105** 063024
[62] Abbott B P *et al* (LIGO Scientific Collaboration and Virgo Collaboration) 2017 *Phys. Rev.* D **95** 042003
[63] Abbott B *et al* (LIGO Scientific Collaboration and Virgo Collaboration) 2019 *Phys. Rev.* D **100** 024017
[64] Abbott R *et al* (The LIGO Scientific Collaboration, The Virgo Collaboration and The KAGRA Collaboration) 2021 *Phys. Rev.* D **104** 122004

[65] Alzubaidi L, Zhang J, Humaidi A J, Al-Dujaili A, Duan Y, Al-Shamma O, Santamaría J, Fadhel M A, Al-Amidie M and Farhan L 2021 *J. Big Data* **8** 1–74
[66] Agarap A F 2018 arXiv:1803.08375
[67] Kingma D P and Ba J 2014 arXiv:1412.6980