

# Ameliorating transient noise bursts in gravitational-wave searches for intermediate-mass black holes

Melissa Lopez<sup>1,2,†</sup>, Giada Caneva Santoro<sup>3</sup>, Ana Martins<sup>2</sup>, Stefano Schmidt<sup>1,2</sup>,  
Jonno Schoppink<sup>2</sup>, Wouter van Straalen<sup>2</sup>, Collin Capano<sup>4,5</sup>, and Sarah Caudill<sup>5,6</sup>

<sup>1</sup>*Nikhef Science Park 105, 1098 XG, Amsterdam, The Netherlands.*

<sup>2</sup>*Institute for Gravitational and Subatomic Physics (GRASP) Utrecht University,  
Princetonplein 1, 3584 CC, Utrecht, The Netherlands.*

<sup>3</sup>*Institut de Fisica d'Altes Energies (IFAE), Barcelona Institute of Science and Technology, E-08193 Barcelona, Spain*

<sup>4</sup>*Department of Physics, Syracuse University, Syracuse, NY 13244, USA*

<sup>5</sup>*Department of Physics, University of Massachusetts, Dartmouth, MA 02747, USA and*

<sup>6</sup>*Center for Scientific Computing and Data Science Research,  
University of Massachusetts, Dartmouth, MA 02747, USA*

(Dated: November 13, 2024)

The direct observation of intermediate-mass black holes (IMBH) populations would not only strengthen the possible evolutionary link between stellar and supermassive black holes, but unveil the details of the pair-instability mechanism and elucidate their influence in galaxy formation. Conclusive observation of IMBHs remained elusive until the detection of gravitational-wave (GW) signal GW190521, which lies with high confidence in the mass gap predicted by the pair-instability mechanism. Despite falling in the sensitivity band of current GW detectors, IMBH searches are challenging due to their similarity to transient bursts of detector noise, known as glitches. In this proof-of-concept work, we combine a matched-filter algorithm with a Machine Learning method to differentiate IMBH signals from non-transient burst noise, known as glitches. In particular, we build a multi-layer perceptron network to perform a multi-class classification of the output triggers of matched-filter. In this way we are able to distinguish simulated GW IMBH signals from different classes of glitches occurred during the third observing run (O3). We correctly classify over 90% of simulated IMBH signals in O3a and over 70% in O3b in single detector data. Furthermore, we search for a set of simulated IMBH signals in real detector noise.

## I. INTRODUCTION

The origin of supermassive black holes (SMBHs) remains a fundamental mystery, despite their presence in nearly every galaxy, including the Milky Way [1, 2]. Recent observational surveys with the James Webb Telescope [3] have identified several hundreds of luminous quasars at redshifts  $z > 6 - 7$ , indicating the existence of SMBHs with masses on the order of  $10^9 - 10^{10} M_\odot$  at such early cosmic epochs [4]. These findings challenge models of SMBH formation and growth, raising questions about the origin of seed black holes and the mechanisms enabling their rapid growth to supermassive scales.

As stellar evolution models predict, stars with helium core masses in the range  $\sim 32 - 64 M_\odot$  are subject to pulsation pair instability, while stars with helium core masses in the range  $\gtrsim 50 - 130 M_\odot$  leave no remnant due to pair-instability supernovae (PISN) [5, 6]. This mass gap implies that stellar collapse alone may not account for SMBH formation. A plausible scenario is the hierarchical mergers of intermediate-mass black holes (IMBHs), with masses between  $10^2 - 10^5 M_\odot$  [7, 8]. Therefore, direct observation of IMBH populations would strengthen the possible evolutionary link between stellar-mass black

holes and SMBHs and unveil the PISN mechanism and serve as a probe for testing General Relativity.

The Advanced Laser Interferometer Gravitational-Wave Observatory (LIGO) [9] and Advanced Virgo [10], detected 11 candidates during the first observing run (O1) and the second observing run (O2), none of which were IMBH candidates [11–13]. Indeed, the detection of IMBHs in gravitational wave (GW) searches remained elusive until the detection of GW190521 during the third observing run (O3) [8, 14]. The estimate of the individual mass components of GW190521 was  $(m_1, m_2) = (85^{+11}_{-14}, 66^{+17}_{-18}) M_\odot$ , with a final remnant mass of  $M_f = 142^{+28}_{-16} M_\odot$ , and making it the first conclusive direct observation of an IMBH [15]. GW190426\_190642 [13] supersedes GW190521 with a final mass of  $M_f = 172.9^{+37.7}_{-33.6} M_\odot$ , while GW200220\_061928 [16] has probably the highest final mass of the second half of O3 (O3b)  $M_f = 141^{+51}_{-31} M_\odot$ .

Despite falling in the sensitivity band of current GW interferometers, IMBH signals are challenging as few cycles of the signal can be observed with current ground-based detectors. State-of-the-art searches employ weakly modelled algorithms, such as coherent Wave Burst (cWB) in its IMBH configuration [17, 18], or modelled algorithms using matched-filter-based techniques [19–25]. Nonetheless, IMBH searches are also hampered by non-Gaussian transient burst noise, known as glitches, that can mask or mimic GW candidates, reduce the

<sup>†</sup> Corresponding author: [melissa.lopez@ligo.org](mailto:melissa.lopez@ligo.org)

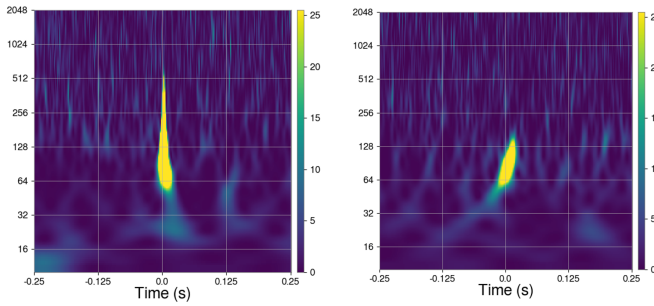


FIG. 1: (Left) Q-transform of a Blip glitch retrieved from *Gravity Spy* [29]. (Right) Q-transform of an event with total mass  $106.6^{+13.5}_{-14.8} M_{\odot}$ .

amount of analyzable data increasing the noise floor, and affect the estimation of the power spectral density (PSD) [26–28]. In Fig. 1 we show the similarities between a Blip glitch and an IMBH signal.

To enhance the sensitivity of current searches, a novel approach is to combine them with Machine Learning (ML) algorithms, fostering a synergistic relationship. With this idea in mind, it has been proposed to enhance matched-filter with ML techniques, modelling the matched-filter process using artificial neural networks, and then optimizing them via gradient descent [30, 31]. It has also been proposed to enhance the performance of cWB employing classification algorithms, such as boosted decision trees and Gaussian mixtures, to discriminate between GW signals and background noise utilizing cWB *trigger* information [32–35]. A novel approach to create a synergistic relationship between ML methods and modeled search algorithms is to understand the behavior of GW signals and glitches across sets of modeled GW signals, or *templates*, formally known as *template banks*. This information can then be fed into a ML algorithm.

The key idea is that when we “match” the unknown detector data  $s(t)$  with the template bank,  $N$  templates “trigger” within a finite time window  $\Delta t$  because they are sufficiently similar to  $s(t)$ . We refer to this inherent cluster structure as *cluster track*, or simply *tracks*, inspired by the tracking of heavy-ion collisions produced at ALICE experiment [36]. Given the morphological time-frequency differences between glitches and GW signals, we expect these “clusters of triggers” to exhibit a meaningful structure to differentiate them with ML methods. We do so by introducing a new statistics: the multi-class probability vector, which measures the probability that each track is generated by either an astrophysical signal or by one of the glitch classes considered.

In this work we use the tracks generated in single-detector, right after the matched-filtering step. Hence, to maintain as much information as possible, we reproduce the IMBH search of GstLAL during O3 using LIGO Hanford (H1), LIGO Livingston (L1) and Virgo (V1) [8]. With this information, the ML algorithm learns to distinguish glitches from simulated IMBH signals in a con-

trolled environment. Afterwards, we use the algorithm to detect a set of simulated IMBH, performing a background estimation. This proof-of-concept study demonstrates that even with a restricted template bank, it is possible to effectively separate signals from glitches, making a potential impact on the accuracy of GW analysis. It is relevant to note that while this particular investigation focuses on IMBH, this method can be extended to other compact binary coalescence (CBC) signals.

This paper is structured as follows. In section II we introduce the current state-of-the-art matched filtering technique and its generated tracks. In section III we provide an overview of the different types of signals employed and the details regarding the construction of the data sets. In section IV we describe the ML model, its input and the learning procedures employed to enhance its performance. In section V we show the training and evaluation metrics of the model, as well as the search of simulated IMBH signals. Finally, in section VII we conclude and propose avenues for future research.

## II. MATCHED FILTERING PIPELINE

As the typical amplitude of a GW signal  $h(t)$  is orders of magnitude smaller than the detector background noise  $n(t)$ , sophisticated detection algorithms, or pipelines, are needed to identify CBC signals. Most of them are based on matched-filter, which in turn relies on the precise models of the sources, for instance, employing Post-Newtonian approaches [37] or the effective-one-body formalism [38]. Some examples are the GstLAL-based inspiral [22, 25, 39–41], PyCBC [19, 21, 42–46], MBTA [47, 48] and SPIIR [49] pipelines. In this work, we use the GstLAL-based inspiral pipeline, hereafter referred to as GstLAL, so in the following sections we provide a high-level overview of some of its particularities.

### A. Time-domain matched filtering

The idea of matched-filter-based searches is to find a target signal  $h(t)$  given the detector output  $s(t) = n(t) + h(t)$ . This is achieved by cross-correlating  $s(t)$  with a modelled waveform  $h_m(t)$ , known as a template. We can define a time-dependent complex scalar product as [19],

$$\langle s|h \rangle(t) = 2 \int_{-\infty}^{\infty} \frac{\tilde{h}_m^*(f) \tilde{s}(f)}{S_n(f)} e^{i2\pi f t} df, \quad (1)$$

where  $S_n(f)$  represents the one-sided PSD, and  $\tilde{\cdot}$  and  $*$  denote the Fourier transform and the complex conjugate, respectively. We can also define the real and imaginary part of the scalar product as  $\langle s|h \rangle(t) = (s|h)(t) + i(s|h)(t)$ . With this definition and given  $h(t)$ , its normalized time series is  $\hat{h} = h/(h|h)$ .

The GstLAL analysis performs matched filtering directly in the time domain to reduce the latency [22, 39].

In accordance with the notation of [39], and given a time series  $s(t)$  and a normalized complex template  $h_c(t)$ , the output of the matched-filter technique is a complex time series,

$$z(t) = (s|h_{\mathcal{R}})(t) + i(s|h_{\mathcal{I}})(t), \quad (2)$$

where  $h_{\mathcal{R}}$  and  $h_{\mathcal{I}}$  are the real and imaginary part of  $h_c(t)$ . The absolute value  $|z(t)|$  of the complex matched filtering as the detection statistics is usually referred to as signal-to-noise ratio (SNR),

$$\rho(t) = |z(t)| = \sqrt{(s|h_{\mathcal{R}})^2(t) + i(s|h_{\mathcal{I}})^2(t)}. \quad (3)$$

The SNR quantifies the “agreement” of the data with a template and as such, it constitutes the main detection statistics. A detection can be claimed only if the SNR exceeds a certain threshold.

### B. Template banks

CBC template waveforms depend on the intrinsic parameters of the source  $\lambda_{int}$ , such as the masses ( $m_1, m_2$ ) and z-spins ( $s_{1z}, s_{2z}$ ) of the binary components, and on extrinsic parameters  $\lambda_{ext}$ , which are related to the position of the source concerning the observer, such as the luminosity distance  $D$ . Thus, the goal of matched filtering is to maximize the detection statistic, SNR, over all these parameters. Luckily, for aligned spin signals, the effect of the extrinsic parameters can be absorbed into a constant scale factor and a phase [50]. For this reason, the SNR (Eq. 3) maximizes over the extrinsic parameters. The maximisation over the intrinsic parameters is usually performed by a brute force approach, where the SNR time-series is computed for a dense grid of modelled GW signals, called *templates*, characterized by  $\lambda_{int}$ . Templates are gathered in large template banks, which are usually generated [51] to achieve a minimal loss of SNR due to the discreteness of the grid, while keeping a manageable computational cost.

Different pipelines employ different strategies to cross-correlate template banks with the output data from the detector while mitigating glitches. In this work we are interested in utilizing the product of this cross-correlation targeting IMBH signals, which are heavily harmed by glitches due to their short duration and similar frequency range. For this aim, we employ GstLAL to reproduce the IMBH search of O3 [8]. We use the IMBH template bank which uses 44902 templates to cover aligned-spin systems with total mass  $M = m_1 + m_2$  in range  $[50, 600] M_{\odot}$  and mass ratio  $q = \frac{m_1}{m_2}$  between  $[1, 10]$ , with both spins in the range  $[-0.98, 0.98]$ . The starting frequency of the analysis is at 10 Hz. For further details the interested reader can refer to [8].

### C. $\xi^2$ -statistic

The detector strain contains glitches that can mask or mimic GW signals, producing large peaks in the SNR time series. As glitches in the detector could yield large SNRs, to mitigate them GstLAL also calculates a signal consistency check, known as  $\xi^2$ -statistic, whenever it records an SNR above a certain threshold. This check is performed by determining the similarity between the SNR time series of the data and the expected SNR time series from the real signal within a  $\delta t$  time window around the peak [39]. Mathematically, the  $\xi^2$ -statistic is defined as,

$$\xi^2 = \frac{\int_{-\delta t}^{\delta t} |z(t) - z(0)R(t)|^2 dt}{\int_{-\delta t}^{\delta t} (2 - 2|R(t)|^2) dt}, \quad (4)$$

where  $z(t)$  is the complex SNR time series,  $z(0)$  is its peak and  $R(t)$  is the auto-correlation series between the complex template waveform and itself. GstLAL weights the SNR with  $\xi^2$  defining

$$\gamma_2 = \frac{\rho}{\left(1 + \max(1, \xi^2)^3/2\right)^{1/5}}, \quad (5)$$

which is utilized to generate triggers

### D. Triggers

When the weighted SNR,  $\gamma_2$ , associated with a template exceeds a certain threshold, the GstLAL pipeline records a feature vector known as a “trigger,” which contains the maximum SNR,  $\xi^2$ ,  $\gamma_2$ , masses, spins, and other parameters.

It often happens that multiple triggers from different templates match the same signal in the data (either of terrestrial or astrophysical origin). They differ in the trigger time, as well as SNR and  $\xi^2$  values. All such triggers must be associated with the same candidate and for this reason, they need to be clustered together to form a single candidate.

The GstLAL pipeline adopted a simple approach to solve this problem: the triggers with the largest SNR peak within the vicinity ( $\pm 1$  s) will be defined as the cluster centroid and utilized for further analysis<sup>2</sup>. While this approach has the benefits of being simple, it discards many pieces of information, such as the position of the various triggers within the template bank and their time ordering. For illustration, Fig. 2 shows the tracks that “matched” GW190521 during the IMBH search in O3 for

<sup>2</sup> Another clustering of triggers is performed at a later stage of the pipeline. However, this has no importance for our purposes.

LIGO Livingston (L1). We represent the IMBH template bank (grey) as a function of the progenitor masses, and the track is coloured as a function of the maximum SNR  $\rho$ .

Afterwards, each trigger is ranked [25, 52, 53] according to its probability of originating from an actual signal. The goal of the pipeline is then to obtain a list of triggers, ordered by their likelihood  $\Lambda$  to be of astrophysical origin.

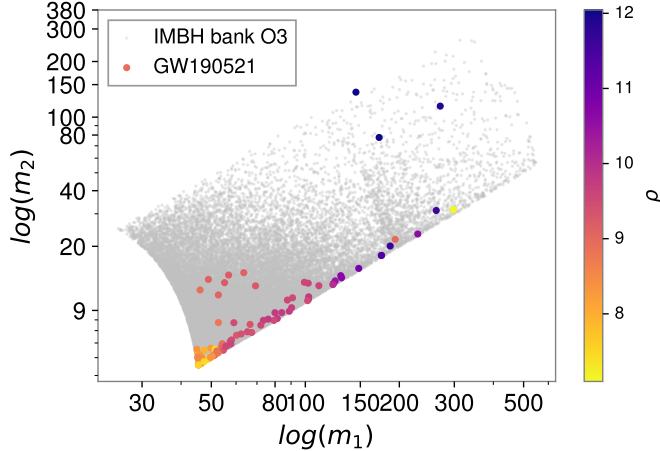


FIG. 2: IMBH template bank from O3 (grey) as a function of the progenitors’ masses measured in  $M_{\odot}$  and logarithmic scale. In colour we show the maximum SNR  $\rho$  of the “matching” templates of GW190521 in LIGO Livingston (L1).

### III. DATA

Once we have collected the tracks data we aim to utilize a supervised ML method to differentiate GW from glitches. Thus, we construct two different data sets: a *controlled* data set which contains well-known glitches and simulated IMBH signals, referred to as *known data set*; and a second data set which contains real GW signals and other *a priori* unknown signals, which we use to construct an accidental background of time shifts (see Section III D for details), referred to as *unknown data set*. While the first data set is employed to assess the performance of the method, the second data set is used to understand the significance of the ML statistic.

In Section III A, we describe the simulated IMBH signals, and the different glitch classes present in the *known data set*. In Section III B we describe the clustering procedure utilized for defining a track. In Section III C we show the patterns generated by an IMBH simulation and a glitch through the template bank, and we define the feature vector associated with the triggers that will be the input to the ML model. Lastly, in Section III D we discuss how we construct the accidental background.

#### A. Injections and glitches

Here, we describe the simulated IMBH waveforms, and the six different glitch populations from the data set of *Gravity Spy* during O3 in H1, L1 and V1 [28, 29]. As *Gravity Spy* is a classification algorithm, it returns a classification for each glitch. Glitches have been selected with a *Gravity Spy* confidence level greater than 90%, indicating that the model is highly confident that the glitches belong to the assigned class.

The chosen glitch classes encompass both short-duration (microsecond) and long-duration (second) signals, which are commonly observed in current ground-based interferometers and cover a broad frequency range. We provide a detailed description of each class below and illustrate the various glitch morphologies in Fig. 3.

- **Injections:** For this aim we use IMRPhenomD approximant [54] with masses  $m_1 \in [50, 400] M_{\odot}$  and  $m_2 \in [10, 250] M_{\odot}$ , and dimensionless spins  $\chi_1, \chi_2 \in [0, 0.99]$ , uniformly sampled within the specified range with aligned spins. The distances are uniformly sampled from the range  $d \in [10, 250]$  Mpc. The inclination is sampled from a uniform distribution, and the sky-localization is randomized.
- **Blip:** these glitches have a characteristic morphology of a symmetric “teardrop” shape in time-frequency in the range  $[30, 250]$  Hz with short-durations,  $\sim 0.04$  s [55]. Due to their abundance and form, these glitches hinder both the unmodeled burst and modelled CBC can searches with particular emphasis on compact binaries with large total mass due to the short duration of their waveforms, among others [56, 57]. Moreover, as their source is still unknown, they cannot be removed from astrophysical searches.
- **Fast\_Scattering:** they appear as short-duration arches ( $\sim 0.2 - 0.3$  s) in the frequency range  $[20 - 60]$  Hz. These glitches are strongly correlated with ground motion in range  $[0.1 - 0.3]$  Hz and  $[1 - 6]$  Hz, which in turn is associated with thunderstorms and human activity near the detector. Note that this class was added during O3, and it is more abundant in L1 than H1, but not present in Virgo, due to differences in ground motion and detector sensitivity [58].
- **Koi\_Fish:** their naming comes from their imaginative frontal resemblance to koi fishes. They are also similar to **Blip** but typically feature high-SNR than the latter, spanning the frequency range of  $\sim 20 - 1000$  Hz.
- **Low\_Frequency\_Burst:** they are short-duration glitches ( $\sim 0.25$  s) in the frequency range  $[10 - 20]$  Hz with a distinctive hump shape at the bottom of



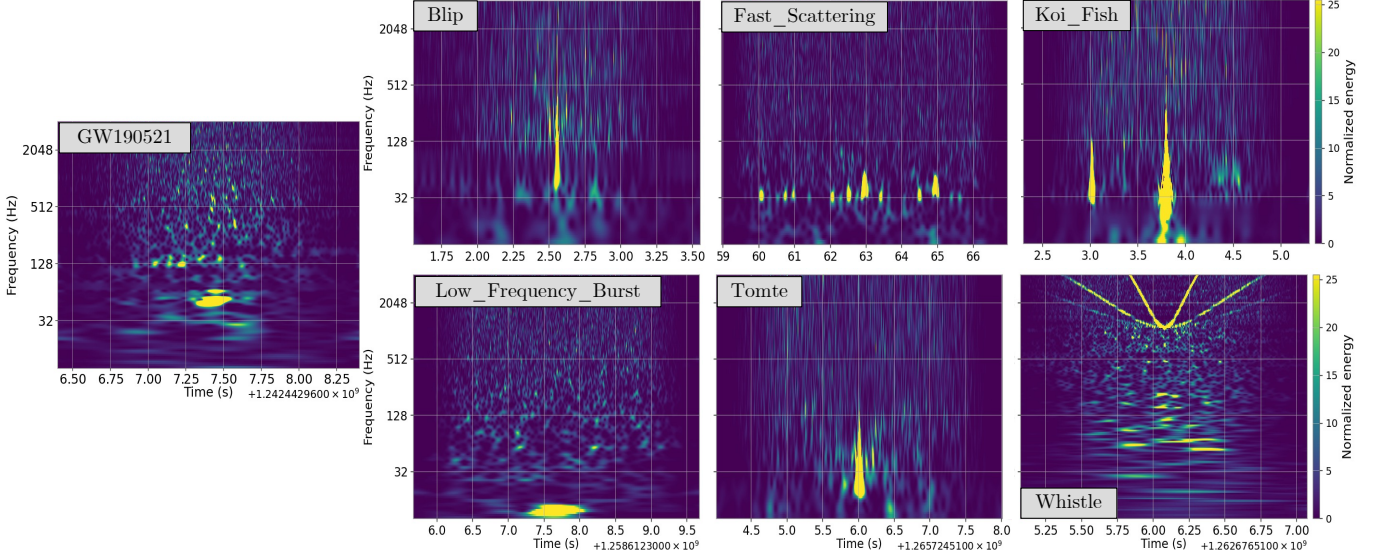


FIG. 3: Q-transform of GW190521 and example of different glitch classes selected from *Gravity Spy* at  $> 90\%$  confidence in LIGO Livingston (L1) [29].

spectrograms. These occurrences were prevalent in L1 data during O1 and H1 data in O3a.

- **Tomte:** these glitches are also short-duration ( $\sim 0.25$  s) with a characteristic triangular morphology. As Blip glitches, their source is unknown, so they cannot be removed from astrophysical searches.
- **Whistle:** these glitches have a characteristic V, U or W shape at higher frequencies ( $\gtrsim 128$  Hz) with typical durations  $\sim 0.25$  s. They are caused when radio-frequency signals beat with the voltage-controlled oscillators [59].

### B. Clustering

In this study, we are interested in exploring the generalization power between the first and second half of the third observing run, namely O3a and O3b, respectively. For this aim, we use the *known data set* from O3a for training, validation and testing, while the *known data set* of O3b is used to test this generalization power. Afterwards, we use the unknown data set to construct an accidental background (see III D) and assess the significance of our ML statistic.

In the following, we describe how the cluster tracks of these data sets were defined.

- **Known data set:** Since we know the GPS time when an Injection or a glitch occurred, we use it as the centroid of the cluster,  $t_{cluster}$ , and we select a window  $t_w = \pm \Delta t$  around it. We experimented with  $\Delta t \in \{0.05, 0.1, 0.2, 0.5, 1\}$  s to find the optimal  $t_w$  of the cluster since IMBH signals are short. It is relevant to note that while we experiment with

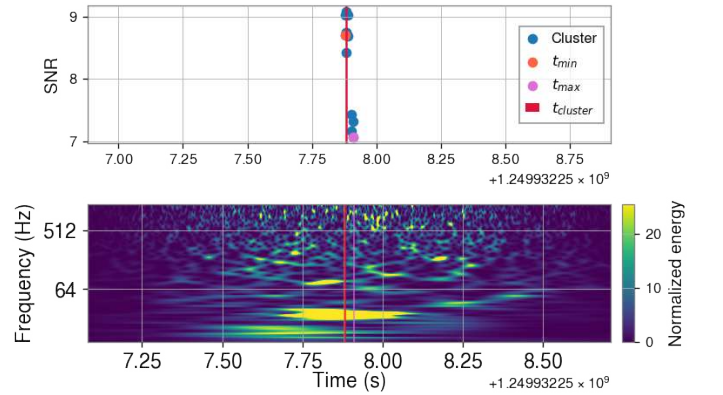


FIG. 4: A **Fast Scattering** glitch labelled by *Gravity Spy*. (Top) SNR of GstLAL tracks as a function of time, where each point represents a template.  $t_{min}$  (orange) and  $t_{max}$  (pink) mark the beginning and end of the cluster, respectively, while  $t_{cluster}$  (red) indicates its centroid given by *Gravity Spy*. (Bottom) Q-transform of the time series containing the **Whistle** labelled by *Gravity Spy*.

different  $t_w$ , once the algorithm has trained with a given  $t_w$ , this value is fixed for validation and testing. We provide a thorough discussion on the selection of optimal  $t_w$  in Section V A.

- **Unknown data set:** Usually clusters have a centroid with the highest  $\rho$ , and a set of neighbours with smaller  $\rho$ . For illustration, in the top panel of Fig. 4 we can observe the SNR of a cluster with this behavior. In a realistic setting, we do not know the GPS time of the centroid of the cluster  $t_{cluster}$

*a priori*, so we define them following a procedure similar to GstLAL. We order all the triggers according to their GPS time. Afterwards, we divide the whole observing run in windows of  $\pm 1$  s, starting from the beginning of the run. Within each window, we select the trigger with maximum  $\rho$  as the centroid, so that  $t_{cluster} = t_{\max \rho}$ . As before, we select  $t_w = \pm \Delta t$  around  $t_{cluster}$  to define the cluster.

While the methodology outlined above provides a systematic approach to identifying and analyzing matched-filtering clusters, it is a simplified approach with several limitations. One significant caveat is the assumption that the trigger with the highest SNR within a given window accurately represents the true centroid of a cluster in the *unknown data set*. This approach may lead to inaccuracies if multiple signals or glitches overlap within the window, potentially causing the algorithm to misidentify the centroid or miss other relevant triggers with slightly lower SNR. Additionally, the fixed window size, although optimized for specific signal characteristics like IMBH signals, might not be ideal for unexpected GW signals of unknown length, leading to suboptimal clustering in some cases. These factors could introduce biases or reduce the accuracy of the clustering, highlighting the need for further refinement and validation of the clustering algorithm in future work.

### C. Trigger tracks

As we mentioned in Section IID when  $\gamma_2$  (see Eq. 5) associated with a template is over a certain threshold, it will produce a trigger with intrinsic parameters  $\lambda_{int}$  associated. In this particular work, we reduce the single-detector  $\lambda_{int}$  associated with the  $i$ th template forming the feature vector  $\phi_i$ ,

$$\phi_i = \{\rho_i, \xi_i, m_{1,i}, m_{2,i}, s_{1z,i}, s_{1z,i}\}, \quad (6)$$

containing the SNR  $\rho$ , the consistency check  $\xi$  (see Eq. 4), the masses of the progenitors ( $m_1, m_2$ ) and the  $z$ -component of their spins ( $\xi_1, \xi_2$ ), respectively. As we mentioned before, an unknown signal  $s(t)$  might have an associated *track*, so to illustrate this scenario we show in Fig. 5 the tracks generated by an **Injection** (Fig. 5a) and a *Blip* glitch (Fig. 5b) projected in the mass and SNR space. We show the O3 IMBH template bank as a function of the progenitor masses  $m_1$  and  $m_2$ , where every dot represents a template. We also colour the tracks, where their colour is related to the maximum SNR,  $\rho$ . As we can see in Fig. 5a, the **Injection** matches a concrete space in the low-mass region at  $\rho \sim 10$ , and a slightly sparse space in the higher-mass region at  $\rho \sim 20$ . On the other hand, in Fig. 5b the templates matching the *Blip* glitch are at  $\rho \sim 12$ , being sparse in the progenitor mass dimension. Assuming that the distribution of GW and

glitch tracks possess distinct underlying features, we can employ ML methods to differentiate them.

### D. Time shift and false alarm rate

Our model will compute a certain statistic  $\Lambda^*$  for the tracks obtained from matched filtering the detector data. We refer to this resulting distribution as *foreground*. It is challenging to assess if single-detector tracks have astronomical or terrestrial origin, but time-coincident signals in multiple detectors thousands of kilometers apart are more likely to be astronomical.

To assess the significance of our foreground, we need to compare it with an noise-only distribution, referred to as *background*. The background tracks time-coincident in multiple detectors must be uncorrelated, so we can use the time-sliding technique, which time-shifts the detectors with respect each other. Subsequently, the coincident triggers observed in the time-shifted data will serve as the background triggers, which are accidental coincidences and not due to actual astrophysical events.

To increase the significance of our foreground trigger, we need to compare it with an extensive population of background triggers. Hence, we make the process of time-sliding the data  $\Delta t_l$  iterative by defining the time shift  $T_l \equiv l\Delta t_l$ , where  $l$  is known as *lag*. When  $l = 0$  we recover the foreground, which is also known as *zero-lag*. A relevant quantity is the total observing time of the foreground or *search time*  $T_f$ , and the total observing time of the background  $T_b$ . In our setup, for the  $l^{\text{th}}$  time-shift  $T_f > T_{b,l}$ , since the overlap between detectors will decrease with increasing offsets. Nonetheless, it is possible to perform time shifts on a *ring*, where data that slides past the end of the segment is placed at the beginning. Note that even in this scenario, time shifts will result in different durations due to gaps in the data. To achieve a significant estimation of the background we would perform  $L$  time shifts, such that  $T_f \ll T_b = \sum_{l=1}^L T_{b,l}$ .

To assess the significance of our track with statistic  $\Lambda^*$  from the foreground, we compare it with tracks produced within the background, known as *false alarms*. We can define the *false alarm probability*  $P_F$ , as the probability of obtaining  $j \geq 1$  triggers with a statistic  $\Lambda \geq \Lambda^*$  within the background, during the time of the search  $T_f$ .

$$P_F(j \geq 1 | \mathcal{F}(\Lambda^*), T_f) = 1 - e^{-\mathcal{F}(\Lambda^*)T_f}. \quad (7)$$

Here, we define  $\mathcal{F}(\Lambda^*)$  as the *false alarm rate* of a trigger with statistic  $\Lambda^*$ , measured as

$$\tilde{\mathcal{F}}(\Lambda^*) = \frac{\sum_{l=1}^L \hat{N}_l(\Lambda \geq \Lambda^*)}{T_b}, \quad (8)$$

where  $\hat{N}_l$  represents the false alarms of the  $l^{\text{th}}$  time-shift, i.e. the number of background triggers within the  $l^{\text{th}}$  time-shift with statistic  $\Lambda \geq \Lambda^*$ . As before,  $T_b$  is the

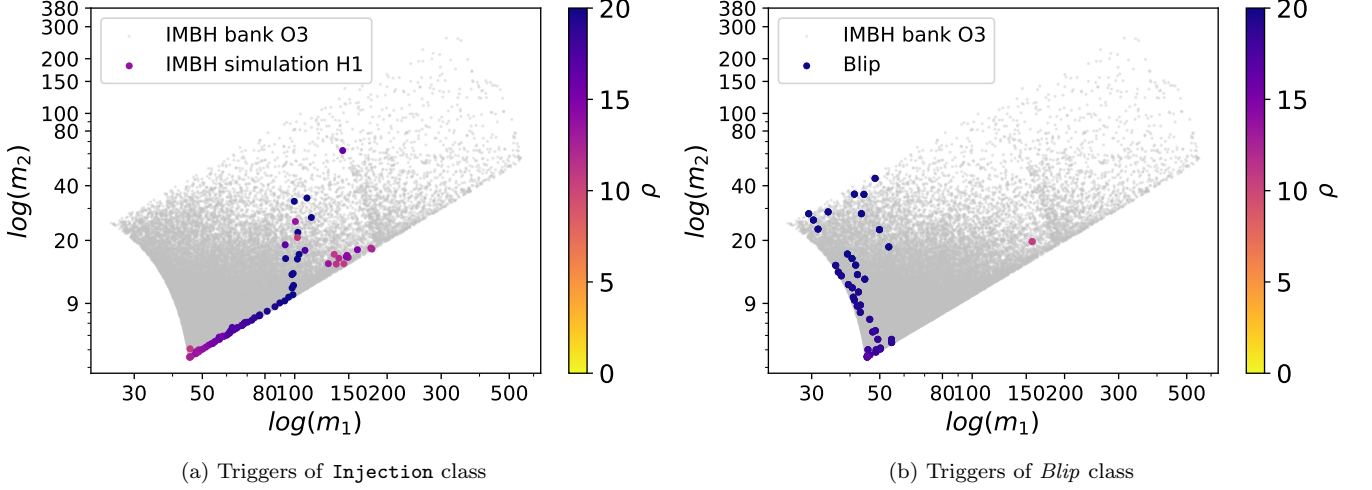


FIG. 5: O3 IMBH template bank as a function of the progenitor masses in logarithmic scale, where every grey point represents a template from the bank. Triggers associated with an *Injection* or a *Blip* glitch are coloured according to the maximum SNR,  $\rho$ .

total background time. It is relevant to note that for each slide the duration of  $T_{b,l}$  will vary, so we define the *effective number of slides*  $\tilde{N} \equiv T_b/T_f$ , so that

$$\begin{aligned} \tilde{F}(\Lambda^*) &= \frac{\sum_{l=i}^L \hat{N}_l(\Lambda \geq \Lambda^*)}{T_b T_f / T_f} \\ &= \frac{\hat{N}(\Lambda \geq \Lambda^*)}{T_f \tilde{N}} = \frac{\bar{N}(\Lambda \geq \Lambda^*)}{T_f}. \end{aligned} \quad (9)$$

We define  $\hat{N}(\Lambda \geq \Lambda^*)$  as the total number of background triggers with  $\Lambda \geq \Lambda^*$  in all time shifts, while  $\bar{N}(\Lambda \geq \Lambda^*)$  is the average number of false alarms in a single experiment with a  $T_f$  duration.

Since the offsets of the time shifts should be large enough that each slide can be thought of as an independent experiment, we chose to slide L1 in steps of 3 s, and V1 in steps of 6 s with respect to H1, which is fixed. We performed enough time shifts to produce 1,000 yr of data in triple detector coincidence. Afterwards, the time shifts are binned according to their coincidence type and their time of background.

To minimize the number of false positives, the types of detector time coincidences analyzed in this work are either double or triple. Using Eq. 9, it is necessary to properly measure the time of the search  $T_s$  and the time of the background  $T_b$ . If we are measuring the time in double-time coincidence, for example, between H1 and L1, it would imply that V1 was down at that time (no V1). Otherwise, if all detectors record data simultaneously, it is considered a triple-time coincidence. In Table I we can see an overview of the time of the search  $T_s$ . Since H1L1 (no V1) is extremely small, it is computationally intensive to generate 1000 yr of time shifts. Thus, in the

TABLE I: Total time of search ( $T_s$ ) for different detector combinations, measured in years.

Time of Search ( $T_s$ )	
<b>H1L1 (no V1)</b>	0.0000382
<b>H1V1 (no L1)</b>	0.1117765
<b>L1V1 (no H1)</b>	0.1338800
<b>H1L1V1</b>	0.5619618

interest of time and computational resources we focus on triple-coincident time as we expect a better performance.

As in a standard search, to assess if two tracks are coincident, we only consider tracks within the time coincidence window. For example, using the H1L1V1 time-coincidence, we can have tracks that occur in all detectors simultaneously and tracks that occur only in H1 and L1. Thus, we will have double coincident tracks, such as H1L1, during H1L1V1 time, as well as triple coincident tracks in H1L1V1 during H1L1V1 time.

#### IV. METHODOLOGY

The challenges in GW research require innovative solutions, and ML, has emerged as a crucial tool for addressing them due to its adaptability and transversality. The main advantage of ML techniques is their rapidity because most of the computations are made during the training stage. In the past few years, researchers have explored different ML applications to GW data analysis, such as detection of CBC detection [60–66], burst identification [67–75], glitch characterization [28, 29, 76, 77] and synthetic data generation [78–83], among others. We refer the interested reader to [84] for a review.

In [30] the authors seek to enhance current state-of-the-

TABLE II: Original data set size before sampling

	Hanford (H1)	Livingston (L1)	Virgo (V1)
Injections	85107	101307	54436
Blip	2717	1701	1534
Fast_Scattering	114	18589	-
Koi_Fish	5147	4061	731
Low_Frequency_Burst	1523	109	3044
Tomte	537	18619	674
Whistle	1946	95	246

art search algorithms with ML applications. Following a similar line of reasoning, and as previously mentioned, we use the triggers generated by the IMBH search of GstLAL during O3 to train a classification algorithm for multiple classes. In this way, we output a statistic to differentiate between IMBH signals and glitches, providing direct information about the nature of these populations.

### A. Feature vector

Under the assumption that different signals have different track structures, it is possible to learn them with ML. Tracks have data structures known in ML as *multi-instance representation*, where  $N$  feature vectors  $\phi_i$ , also known as triggers, represent a given example. This implies that each track may contain a different number of triggers for each example. Consequently, the input length to our ML model changes from one example to another. This poses a limitation because standard ML techniques only accept fixed-length inputs. As a proof-of-concept to overcome this obstacle, we average all the feature vectors  $\phi_i$  associated with a given signal by weighting them with  $\rho$ , resulting in the following input feature

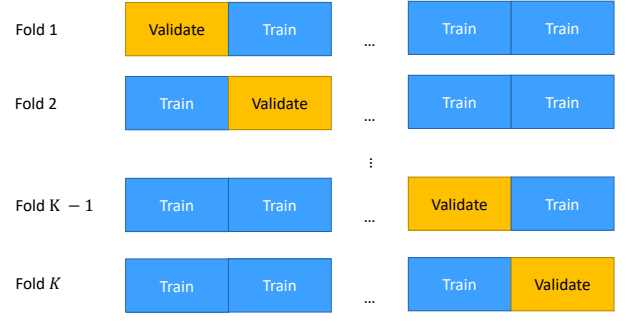
$$\Phi = \{\bar{\rho}, \bar{\xi}, \bar{m}_1, \bar{m}_2, \bar{s}_{1z}, \bar{s}_{1z}\}$$

$$\text{where } \bar{x} = \frac{\sum_{i=1}^N x_i \rho_i}{\sum_{i=1}^N \rho_i}, \quad (10)$$

for  $i \in 1, \dots, N$  the triggers associated with each signal. In this way, we give more weight to the template that best “matches” a given signal. This weighted average enhanced the performance of our ML model with respect to a standard average.

### B. Tackling class imbalance

While we can simulate a large number of GW signals, glitches have a finite nature. Despite their occurrence rate being approximately  $\sim 1 \text{ min}^{-1}$ , some classes of glitches are more common than others [14]. For example, in L1 there are  $> 10^4$  **Tomtes**, while there are  $\sim 10^3$  **Whistles**. In Table II we show the original size of our training and validation set, which is highly imbalanced.

FIG. 6: Illustration of  $K$ -fold cross-validation.

This is problematic for classification ML algorithms, as heavily imbalanced datasets tend to be biased towards the majority class to minimize their loss function. This may wrongly be interpreted as a “good” performance, while in reality, the model is only learning a single class: the largest one. Similarly to [80], to circumvent this issue we use undersampling or oversampling techniques:

- **Undersampling:** In our data set, the number of **Injections**, i.e. simulated IMBH signals, performed is significantly higher than the largest glitch class. Thus, we randomly sample this class to match the size of the oversampled glitches.
- **Oversampling:** To oversample we use bootstrapping with replacement, which stochastically resamples the existing data set allowing the same example to appear more than once [85]. Using this method, we oversample all glitch classes to equal the size of the largest glitch class.

While this method improved the performance of our ML method, it is important to realize that bootstrapping with replacement does not generate new data nor gives new information about the classes. If the original data set does not represent the class population, it will bias the ML algorithm producing overfitting.

To avoid overfitting we use *K-fold cross-validation* [86]. This method partitions the data set in  $K$  subsets or folds (see Fig. 6). The model is evaluated  $K$  times, using a different fold as validation in each iteration, while training on the rest of the folds. This allows for a more robust estimate of the model’s performance, as it accounts for the variability in the data and limits the potential for overfitting. In particular, we chose  $K = 9$  as it is a trade-off between computational complexity and performance.

As we mentioned in Section III B we want to test the generalization ability of the MLP for O3a and O3b. Therefore, we use the *known data set* of O3a for the standard ML procedure of training, validation and testing, splitting the data set into 80% for training, 10% for validation and 10% for testing with the  $K$ -fold cross-validation method. Once we have tested in *known data set* of O3a, we use the whole *known data set* of O3b to test the generalization ability of the MLP.



### C. Training model

In this work, we use a multi-layer perceptron (MLP), also known as a fully connected or dense network. This model implements a mapping between the input space  $\mathbb{R}^n$  to an output space  $\mathbb{R}^d$ ,  $f_\theta : \mathbb{R}^n \rightarrow \mathbb{R}^d$ , where  $\theta$  represents its trainable parameters.

The unit of an MLP is the perceptron, a mathematical simplification of human neurons [87]. In this unit, the information is given by a  $n$ -vector  $x = [x_1, \dots, x_n]$ , that travels in the neuron and interacts multiplicatively with the weights,  $w = [w_1, \dots, w_n]$ . The goal of the perceptron is to learn the value of the weights, which controls the strength of the influence of the previous neuron. In this simplistic model, the information is summed, such that if the sum is above a certain threshold, the neuron creates or *fires* a signal. This *firing rate* is modelled as the activation function  $f : \mathbb{R} \rightarrow \mathbb{R}$ , which is non-linear. Thus, we can mathematically define the  $a$  output, known as activation as,

$$a = f\left(\sum_i^n w_i x_i + b\right), \quad (11)$$

where  $b$  is the bias term, which can be interpreted as the error of the model. The MLP model is a stacking of neurons, where neurons are arranged in layers. In this setup, for a given layer  $l$  we can define the matrix of weights  $W_{ij}^{(l)}$  and the vector of biases  $b_i^l$ , where the weights are the connection between the unit  $j$  of layer  $l$  and the unit  $i$  of layer  $l + 1$ , and the bias is associated with the unit  $i$  of layer  $l + 1$ . Then, we can express the MLP compactly as

$$z^{(l+1)} = W^{(l)} a^{(l)} + b^{(l)}, \quad a^{(l+1)} = f(z^{(l+1)}),$$

$$\text{where } z_i^{l+1} = \sum_{j=1}^2 W_{ij}^{(1)} x_j + b_i^l. \quad (12)$$

To differentiate IMBH signals from glitch classes (see Sections III A), we construct an MLP (discussed in Section IV C) for each detector, that inputs the feature vector  $\Phi$  (see Eq. 10) and outputs a probability vector that indicates the likelihood for each class. The size of this probability vector depends on the detector: we distinguish 7 classes for H1 and L1, but 6 classes for V1, as `Fast_Scattering` class is not present in that detector.

We implemented the MLP structure in *PyTorch* [88]. After several experiments, we selected the best-performing architecture from our results: it consists of 3 hidden layers of 350 each, using the ReLU activation function [89], except the output layer that uses Softmax activation function, since it is a multi-class task. This MLP architecture is common for each GW detector, but they have been trained separately, i.e. it is a single-detector classifier. To optimize the networks we use cross-entropy loss function, and Adam optimizer [90]. Moreover, to adjust the number of epochs and avoid overfitting

we implemented an *early stopping* algorithm [91]. We define an epoch as the number of times the network has passed through the whole training and validation data set. Thus, the early stopping algorithm calculates the difference in validation accuracy  $\mathcal{A}_{val}$  between the current epoch,  $e$ , and the best epoch,  $e^{\text{best}}$ . The training process finishes if

$$|\mathcal{A}_{val}^e - \mathcal{A}_{val}^{e^{\text{best}}}| \leq \epsilon, \quad (13)$$

during 150 epochs, with  $\epsilon = 0.0001$ . For the learning rate, we use an *adaptive learning rate* built-in the *PyTorch* function `ReduceLROnPlateau`. Setting the initial learning rate to  $10^{-3}$ , if the validation accuracy remains constant after 100 epochs, the learning rate decreases 10%. The combination of these methods proved to increase the performance while decreasing the time needed for fine-tuning.

### D. Time coincident tracks

Even though the MLP receives direct information from the `Injection` class and various glitch classes, misclassifications are still common due to the simplicity of the feature vector (see Section IV A). Nonetheless, if a potential GW signal is detected in multiple detectors, the likelihood of it being of astronomical origin increases significantly. Hence, it is standard in GW searches to evaluate triggers that happened within the light time travel between detectors: 10 ms between H1 and L1, 27 ms between H1 and V1, and 26 ms between L1 and V1. An additional 5 ms is considered for statistical fluctuations [22].

For the `Injection` class of the *known data set*, we can identify coincident tracks because we have access to the ground truth. However, for the glitches of the *known data set* and the *unknown data set* itself, this information is not available *a priori*, so we need to define criteria for when two tracks are time coincident. In a standard matched-filtering search, a time coincident trigger occurs when the same template triggers in different detectors within the light travel time between them.

In this work, the approach is inherently different, as we do not deal with individual triggers but with averaged tracks. Thus, two tracks from different detectors are considered time coincident if the average time of the triggers within each track falls within the light travel time and time fluctuations. Nevertheless, even if two tracks are time coincident, there is no guarantee that they originate from the same signal. Therefore, two tracks are only considered time coincident if they have at least one common trigger. If tracks from different detectors coincide in time, we assign to the coincidence a probability equal to the harmonic mean, one of Pythagorean means [92, 93], of each trigger, relabelling our statistic for  $N$  detectors as

$$\bar{P}_{inj} = N / \left( \sum_k^N P_{inj,k}^{-1} \right) \quad (14)$$

where  $P_{inj}$  is the probability of being an **Injection** (see Section III A for a description). It is important to note that the MLP performs a single-detector inference, and the time coincident step is computed independently afterwards. In future works, it would be interesting to provide the full information of the time coincident tracks to a ML algorithm. This integration could enhance the model's ability to distinguish GW signals from glitches.

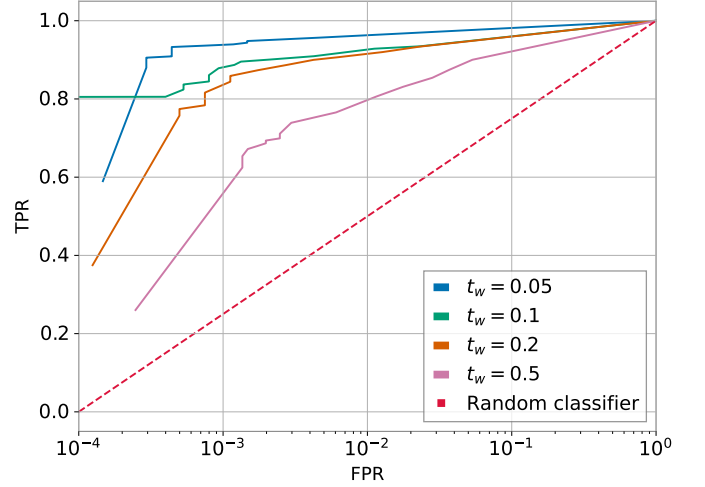
## V. RESULTS

In the present Section, we show the results of the performance of the MLP model with H1 data. The results for L1 and V1 can be found in the Appendix VIII. Firstly, in Section V A, we describe the selection of the time window  $t_w$  for the *known data set* of O3a, i.e. the controlled data set of O3a composed of simulated GW signals and well-known glitches. Afterwards, in Section V B, we show the results of the training and validation with the previous data set. In Section V C we test the model with the *known data set* of O3a and O3b. In Section V D we assess the significance of the ML inference with the accidental background of time shifts.

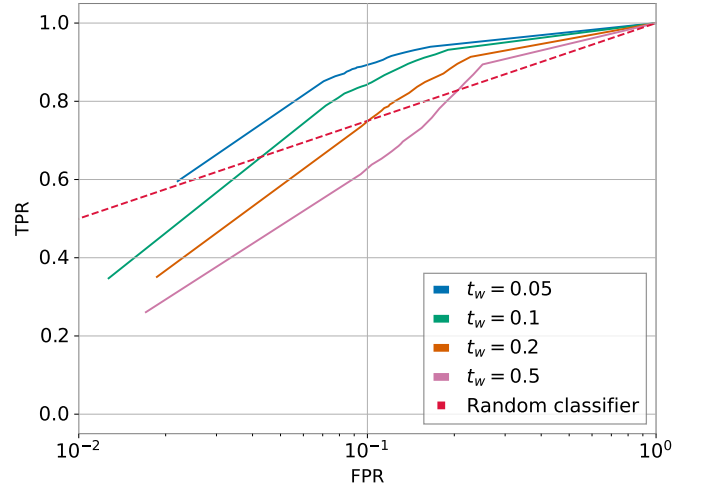
### A. Selecting a time window

As mentioned in Section III B, we explored different time windows  $t_w$  for each GW detector, namely H1, L1 and V1. To evaluate the performance of the algorithms without fine-tuning them we use the ROC curve, which is represented as the TPR as a function of FPR [94]. A relevant point is that our task is a multi-class classification, so to compute the ROC curve, which is usually employed in binary classification, we need to reduce our problem to a pairwise comparison, i.e. **Injection** class (positive class) against all other classes (negative class).

Furthermore, it is custom in ML to calculate the area under the ROC curve as an evaluation metric, since models with a larger area under the curve have a better performance. However, in the field of GW, it is required to know the performance of the model at low FPR as we want to minimize the number of FPs when claiming detection, i.e. the number of glitches incorrectly classified as GW signals. For this aim, we select a grid of decision thresholds  $\theta^*$  in range  $[0.1, 0.9]$  with a spacing of 0.1 and in range  $[0.9, 1.0]$  with a spacing of 0.01. When a given probability exceeds this value, we classify the input as a positive, i.e. **Injection** class, otherwise, it is classified as negative, i.e. any other class. As this grid of decision threshold is not usual in ML, the area under the curve has a different magnitude, so instead we select the  $t_w$



(a) Testing on O3a with *known data set*.



(b) Testing on O3b with *known data set*.

FIG. 7: Receiver operator curve (ROC) for different time windows  $t_w$  in H1, i.e. true positive rate (TPR) as a function of false positive rate (FPR) in logarithmic scale. The positive class is **Injection** class, while the negative class is any of the other classes. (Top) Testing in the *known data set* of O3a. (Bottom) Testing in the *known data set* of O3b. Note that the dashed line indicates a random guess.

with the best performance in *known data set* of O3a and O3b.

In Fig. 7 we present the ROC curves of H1 for the *known data set* of O3a (Fig. 7a) the *known data set* of O3b (Fig. 7b). In Fig. 7a we can observe that the TPR degrades as we increase the size of  $t_w$ . In Fig. 7b we can see a similar behavior but a higher FPR range.

However, at  $\text{FPR} < 10^{-1}$  the performance drops dramatically below the random classifier line (dashed line), which implies that the model is making random guesses. As we increase the size of  $t_w$ , the performance of the al-

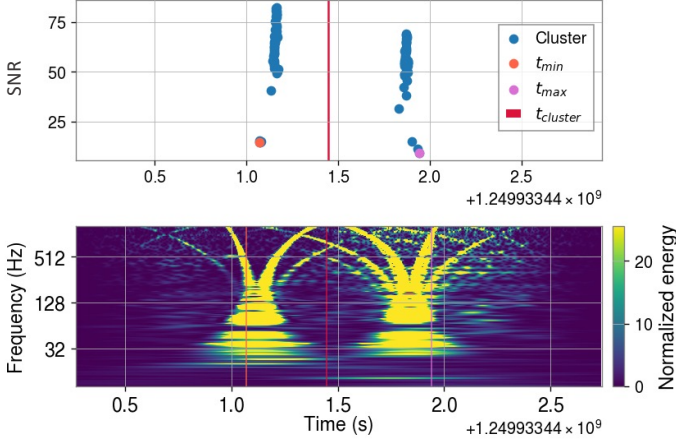


FIG. 8: A Whistle glitch labelled by *Gravity Spy*. (Top) SNR of GstLAL tracks as a function of time, where each point represents a template.  $t_{min}$  (orange) and  $t_{max}$  (pink) mark the beginning and end of the cluster, respectively, while  $t_{cluster}$  (red) indicates its centroid given by *Gravity Spy*. (Bottom) Q-transform of the time series containing the Whistle labelled by *Gravity Spy*. While *Gravity Spy* labels a single glitch, GstLAL identifies both.

gorithm decreases. This behavior could be caused by the short duration of IMBH signals, meaning that if we define  $t_w$  to be large, the clusters will contain random triggers that will bias the classification task. Furthermore, this behavior is also observed in L1 and V1 (see Appendix A 1 for details), so we conclude that  $t_w = 0.05$  s is the best time window for our task.

It is relevant to note that one limitation of this method is the lack of ground truth when defining the centroid  $t_{cluster}$  of a glitch. As an example, in Fig. 8 we show a Whistle labelled by *Gravity Spy*. The time centre of the cluster,  $t_{cluster}$ , is marked in red, and according to *Gravity Spy*, it is associated with a single Whistle. GstLAL identifies two clusters that correspond to two Whistle instead, as can be seen from the bottom panel. In future works, it would be relevant to study the effect of this offset on the behavior of the model.

### B. Training with known data set of O3a

Once we have selected  $t_w$ , and after several fine-tuning experiments, we can train and validate our model with the *known data set* of O3a. As we mentioned in Section IV B, we use 9-fold cross-validation to enhance the performance of the algorithm. Thus, in Fig. 9 we show, for training and validation, the mean accuracy (top panel) and loss (bottom panel) of the 9-fold cross-validation as a function of the epochs, where the shaded region represents  $\pm 3$  standard deviations. Note that the loss is plotted in logarithmic scale so we can appreciate the dif-

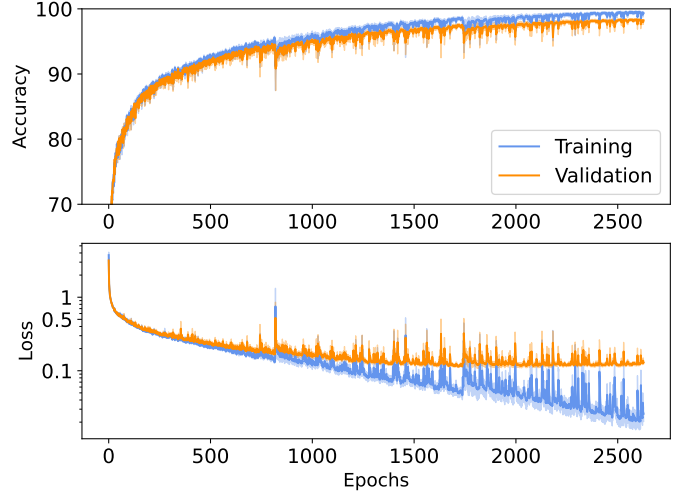


FIG. 9: Comparison between training and validating with 9-fold cross-validation for H1 during O3a. (Top) Mean accuracy at 3 standard deviations (shaded region) as a function of the epochs during training and testing. (Bottom) Average 9-fold cross-validation loss as a function of the epochs during training and testing.

ference between training and validation loss is minimal, which implies that the model is not overfitting. Another relevant point is that both mean accuracy and loss seem to have sharp peaks around epochs 900 and 1700, which could imply that the learning is unstable. However, this is a known effect of the adaptive learning rate that we described in Section IV C. The described behavior is also present in L1 and V1, whose results can be found in Appendix A 2.

### C. Diving into the known data: machine learning performance

Employing GPU Tesla V100 with a memory of 16 Gb allowed us to train our model in  $\approx 17.40$  h for H1 data,  $\approx 15.55$  h for L1 data,  $\approx 31.20$  h for V1 data. Such a large amount of time is mainly due to the 9-fold cross-validation procedure. Nonetheless, once the models are trained we can predict a single input in  $2.9 \times 10^{-6}$  s. As we also want to test the generalization power of the model between O3a and O3b *known data set*, we present their confusion matrix for H1 in Fig. 10 (see results for L1 and V1 in Appendix A 3). While in the y-axis we represent the ground truth, in the x-axis we represent the predictions of the model. Furthermore, the percentages in the rows sum 100%, and the elements in the diagonal were successfully classified.

In Fig. 10a we show the confusion matrix of the test using the O3a *known data set* for H1, where the accuracy is 98.8%. We can see that 93.2% of Injections were correctly classified with little amount of misclassifications. However, for the O3b *known data set* the accuracy de-

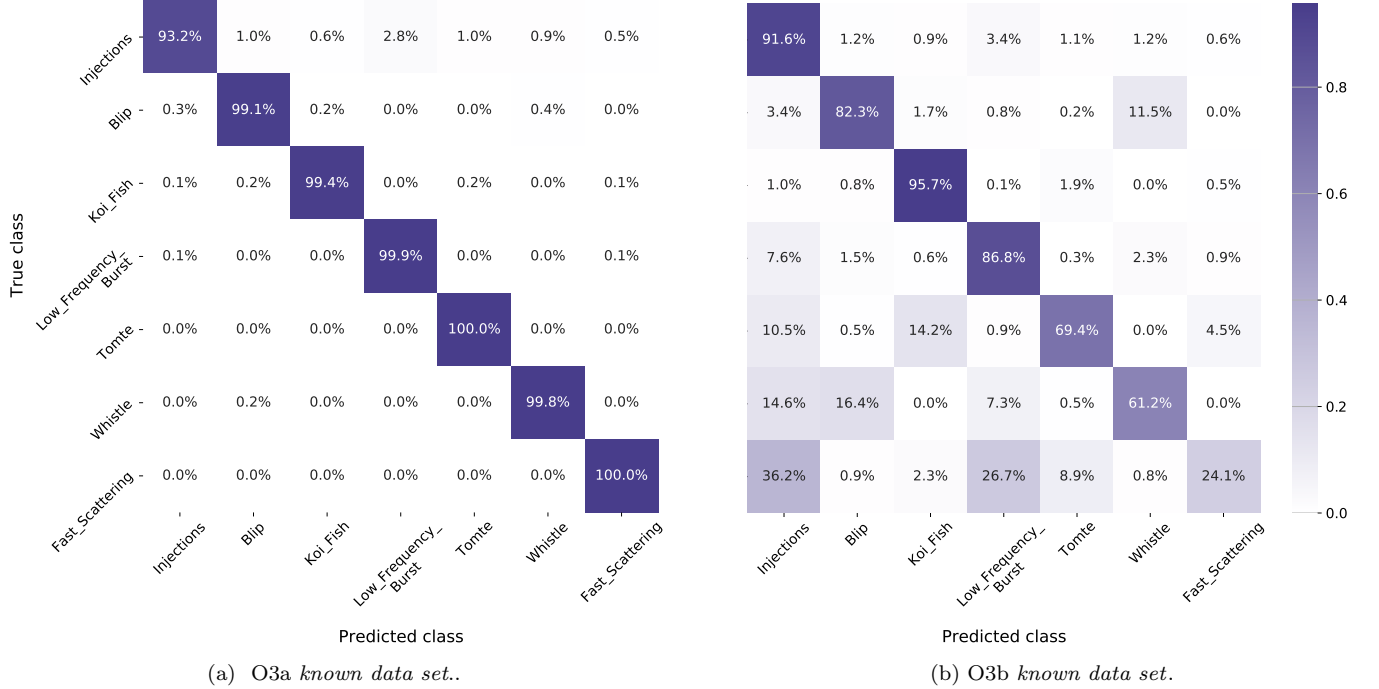


FIG. 10: Relative values of the confusion matrix for the test set for H1. The y-axis represents the ground-truth, i.e. the true class, and the x-axis represents the prediction of the model, i.e. the predicted class.

creases sharply to 73.0%. We can see in Fig. 10b that some of the glitches are wrongly classified as **Injections** or other classes. In particular, the **Fast\_Scattering** class seems to be the most problematic, as 36.2% of them seem to be classified as **Injections** and 26.7% as **Low\_Frequency\_Burst**. This lack of generalization between O3a and O3b is common to L1 and V1, where the accuracy in O3a is 95.8% and 99.3%, and the accuracy in O3b decreases to 67.5% and 75.9%, respectively. The misclassification of **Fast\_Scattering** is also present in L1, but the misclassification of **Low\_Frequency\_Burst** as **Fast\_Scattering** is more acute. Interestingly, in V1, the most problematic is the **Whistle** class (see Fig. 19 and Fig. 20 in Appendix A 3).

To assess the degree of misclassification during O3b with respect to the SNR distribution we compute the TPR as a function of the average SNR  $\bar{\rho}$  (Fig. 11a). For this aim, we arrange the **Injections** in SNR bins and compute their TPR setting the decision threshold to be  $P_{inj}^* = 0.9$ . Interestingly, we can observe that the TPR is high for low  $\bar{\rho}$ , showing dip at  $\bar{\rho} \sim 8$  of  $\text{TPR} \sim 0.85$  for H1, and  $\text{TPR} \sim 0.7$  for L1 and V1. Moreover, the TPR becomes unstable at  $\bar{\rho} > 15$ .

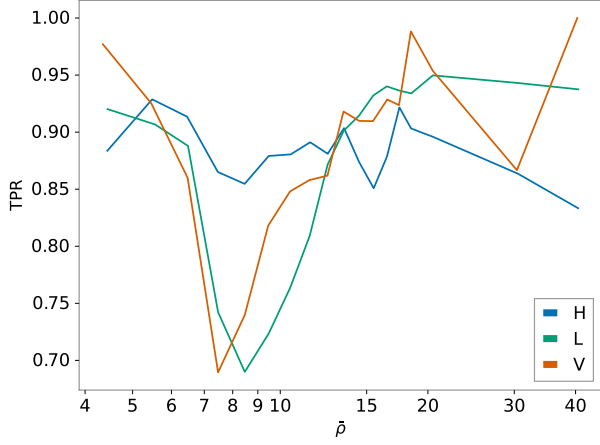
To explain this behavior we need to observe the SNR  $\rho$  distributions, computed by *Omicron* [95], of the different classes for the *known data set* of O3b (Fig. 11b). Firstly, it is important to note that *Gravity Spy* classifies glitches with *Omicron* SNR  $> 7.5$ . In Fig. 11b we show the distribution of  $\rho$ , the resulting SNR from *Omicron*, where we can still observe that most glitch

populations are centred around  $\rho \sim 8$ , which explains the dip in Fig. 11a. Our method understands that most classes of problematic glitches, such as **Fast\_Scattering**, **Tomte** and **Low\_Frequency\_Burst** are around this value, so it does not have trouble distinguishing **Injections** at lower  $\rho$ . Note that at  $\rho < 5$  the model simply differentiates between **Whistle**, **Fast\_Scattering** and **Injection** classes. Furthermore, as  $\rho$  increases, the population of **Injections** decreases, so due to the low statistic at  $\bar{\rho} > 15$  the TPR seems unstable. We note that a different SNR distribution for the **Injection** class might lead to a different behavior of the TPR as a function of the SNR. We will leave this exploration to future work.

A possible cause for this lack of generalization between O3a and O3b can be the fact that the interferometers are evolving systems, so the glitches produced at the beginning of the observing run can be different from the ones produced at the end. Furthermore, we remind the reader that while the **Injections** class is the actual ground truth, there is some bias in the definition of the glitch classes (e.g. Fig. 8). Another reason is the limitation of the model itself, as we are performing a multi-classification task with information on 6 variables (see Eq. 10). To lower the number of FPs, i.e. glitches that are incorrectly classified as **Injections**, in this single-detector task we can use time coincidence between detectors (see Section IV D for a definition), since it is less common for glitches to appear in several interferometers at the same time.

In Fig 12 we present the probability of being an





(a) True positive rate (TPR)

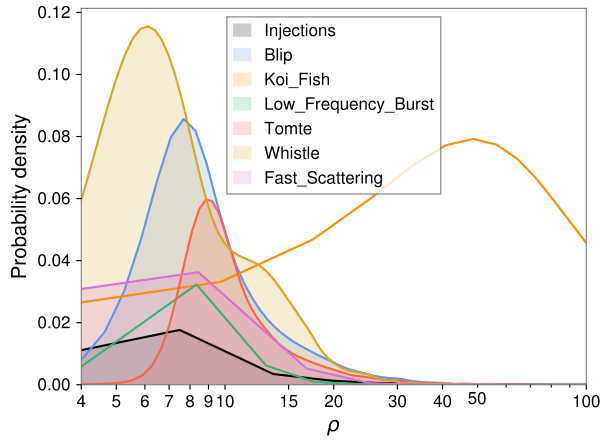
(b) SNR  $\rho$  probability density of different classes.

FIG. 11: (Top panel) Single detector TPR as a function of the average SNR  $\bar{\rho}$  in the *known data set* of O3b for H1, L1 and V1. (Bottom panel) SNR  $\rho$  probability density of different classes in the *known data set* of O3b.

**Injection** ( $P_{inj}$ ) for the *known test set* of H1, in blue, L1, in red, and their time coincidence, in purple. In the top panel, we present the probability density of the **Injection** class, while in the bottom panel, we show the probability density of the glitches. In the top panel, we can observe that, for both detectors, while the MLP model classifies many **Injections** as such with large  $P_{inj}$ , there are still many **Injections** that have a low  $P_{inj}$ , which can be misclassified as glitches. Conversely, the MLP model gives many glitches a low  $P_{inj}$ , but there are still many glitches with a high  $P_{inj}$ . To increase  $P_{inj}$  of **Injections** and lower the  $P_{inj}$  of glitches, we use time coincidence. Hence, enforcing time coincidence increases  $P_{inj}$  of **Injections** (top panel), while completely discarding the glitches (bottom panel). Moreover, under the assumption that tracks with a low number of triggers are produced by detector noise we limit our analysis to tracks with 10 triggers or more.

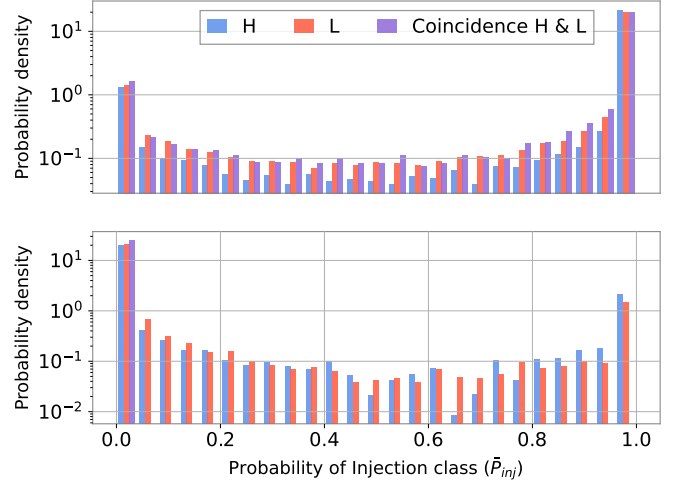


FIG. 12: Probability density of being **Injection**, using the *known test set* of O3b for H1, in blue, L1, in red, and their time coincidence, in purple. (Top) Probability density of elements in the **Injection** class in logarithmic scale. (Bottom) Probability density of elements in all the other classes, i.e. glitches, in logarithmic scale. Given the counts of the  $i$ th bin  $c_i$  and its width  $b_i$ , we define the probability density as  $c_i/(\sum_i^N c_i \times b_i)$ , where  $N$  is the total number of bins of the histogram.

Time coincidence, together with track reduction, discards many **Injection** tracks: while we have 21900 tracks in H1 and 43100 tracks in L1, their coincidence yields only 8967 tracks. As before, in Appendix A3 we present the coincident results of the pairs L1 and V1, and H1 and V1, where an identical behavior can be observed.

#### D. Diving into the *known data*: significance of $P_{inj}$ statistic

Up until now, we performed the standard statistical tests in the field of ML with  $P_{inj}$  as our probability. Nonetheless, a ML based detection algorithm requires a proper understanding of the significance of its statistics considering the accidental background. With the time-shifted background, we intend to find extreme values of  $\bar{P}_{inj}$  (see Eq. 14), but due to the probabilistic nature of ML algorithms, our statistic saturates at 1 as we can see in Fig. 12. Since we are interested in the region with the highest probability we redefine our statistics as

$$-\log(1 - \bar{P}_{inj}) + \epsilon, \quad (15)$$

where  $\bar{\cdot}$  represents again the harmonic mean (see Eq. 14 for a definition), and  $\epsilon = 10^{-20}$  is added to prevent undefined behavior when  $\bar{P}_{inj} \approx 1$ , ensuring numerical stability. The main advantage of using the harmonic mean in this context is that it places more emphasis on smaller

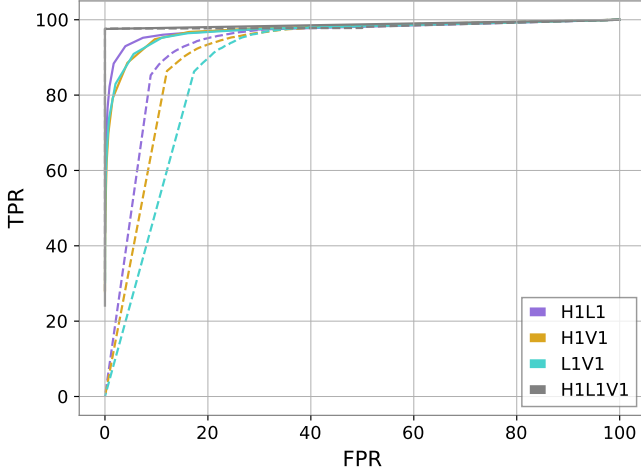


FIG. 13: ROC curve of double and triple detector time (H1L1V1) for Eq. 15 (dashed lines) and Eq. 16. Here, TPR is the rate of correctly classified GW injections from the zero lag, while FPR is the rate of incorrectly classified time slides background tracks.

TABLE III: Area under the curve (AUC) from ROC curves in Fig. 13.

	Time	
	Detector	H1L1V1
$-\log(1 - \bar{P}_{inj})$	H1L1	9281.54
	H1V1	9113.69
	L1V1	8843.81
	H1L1V1	9838.81
$\overline{P_{inj}/(\xi^2/SNR^2)}$	H1L1	9752.80
	H1V1	9716.73
	L1V1	9712.19
	H1L1V1	9877.87

values, which helps to down-rank candidates with lower statistics. To assess the performance of this statistic we compute the ROC curve, but with a different definition of TP and FP. TP are coincident GW injections of O3a present in the zero lag that has been identified by Gst-LAL, i.e. they produce a track. On the other hand, FP are the coincident tracks of the time shifts of O3. With this definition we can construct the ROC curve stepping on different values of our statistic  $-\log(1 - \bar{P}_{inj}) + \epsilon$ . Note that better performing ROC would be the ones that maximize the area under the curve (AUC). It is relevant to note that, as we are working with triple time if one detector does not observe the signal, then its contribution  $P_{inj,k}^{-1} = 0$  (see Eq. 14).

In Fig. 13 we present the ROC curve of  $-\log(1 - \bar{P}_{inj}) + \epsilon$  (dashed lines) in triple detector time (H1L1V1) for different detector combinations. We can observe that H1L1 and H1V1 have a better performance than L1V1. This implies that the L1V1 combination has

more background tracks, possibly due to a larger glitch population in both detectors for different reasons. On the other hand, H1L1V1 has a better performance than double-coincidences since we only have two background tracks. We summarize their AUC in Table III.

As we have seen for L1V1, it is possible that the behavior of MLP is overly optimistic, caused by a combination of its training in a completely controlled population-GW simulations and well-known glitches-and an overly simplified track input. Because of this, we explored a combination of  $P_{inj}$  with variables used in traditional GW searches, such as SNR and  $\xi^2$ . The best-performing statistic was

$$\overline{P_{inj}/(\xi^2/SNR^2)}, \quad (16)$$

where again  $\bar{\cdot}$  represents the harmonic mean. We present in Fig. 13 (solid lines) its ROC curve in triple detector time (H1L1V1). We can observe that all double coincidences have improved greatly with respect to the statistic  $-\log(1 - \bar{P}_{inj}) + \epsilon$ . Moreover, in triple coincidence, the AUC is 9877.87, while before it was 9838.81 (see Table III).

In GW searches, a common magnitude to assess the performance is to use the false alarm rate (FAR) measured in  $\text{yr}^{-1}$  (see Eq. 9), constructed by employing the time shifts. In particular, it is custom to represent the inverse FAR (iFAR) as a function of the ranking statistic,  $\overline{P_{inj}/(\xi^2/SNR^2)}$ , which we show in Fig. 14. This plot relates the ranking statistic with the iFAR significance, i.e. the number of false alarms per the number of analysis times measured in years. As an example, if we have  $\overline{P_{inj}/(\xi^2/SNR^2)} \approx 150$  for H1L1 coincidence, then we will have an iFAR  $\approx 0.01$ , or equivalently, one false alarm per 100 years. Note that H1L1V1 is in the bottom left corner as it has two samples.

A principal characteristic of a GW ranking statistic is that it increases monotonically as the probability of finding a GW signal increases. Because of this, real GW signal tracks would lay on the far right of the iFAR distribution, increasing the false alarm. To investigate whether our ranking statistic behaves in this way we remove all tracks that correspond to real GW signals, yielding the dashed line distributions. While there is a noticeable discrepancy in the H1L1 distribution, this issue does not occur with H1V1 and L1V1. This may be due to the lower sensitivity of V1 or a limitation of our statistics.

To further understand the performance of our statistic, we measure the recovery of GW injections from O3a. To compute the TPR, we fix iFAR at 0.1, or equivalently, one per 10 years, using the results in Fig. 14. In Fig. 15 we present TPR as a function of the injected SNR (Fig. 15a) and as a function of the luminosity distance  $D_L$  in Mpc (Fig. 15b).

In Fig. 15a we can observe that for all detector combinations  $\text{TPR} \approx 0.95$  for injected  $\text{SNR} \gtrsim 8$ . For  $\text{SNR} \lesssim 8$  there is a drop in performance for double coincidence

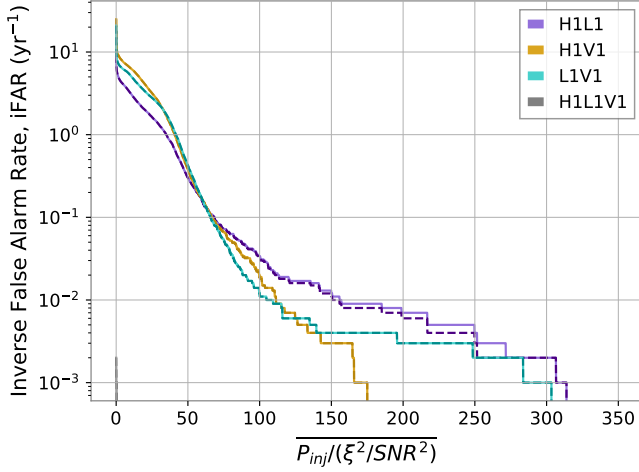


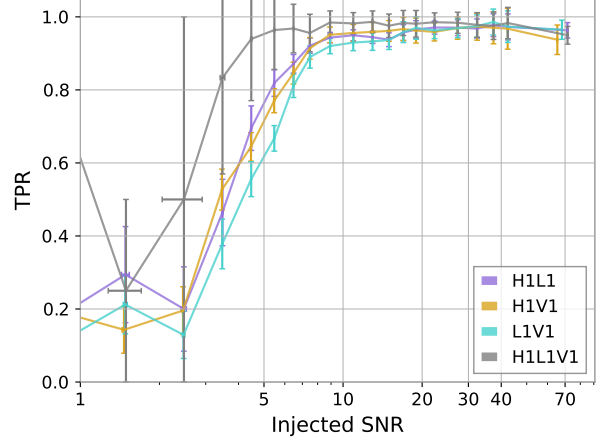
FIG. 14: Inverse false alarm rate (iFAR) as a function of the ranking statistic in triple time (H1L1V1) for different detector combinations. The solid lines show the time-shifted background with real GW signal tracks, while dashed lines show the time-shifted background without them.

tracks, while triple coincident tracks maintain their TPR until  $\approx 5$  injected SNR. However, we must note that for an injected SNR  $\approx 5$  there is less statistic, as we can observe from the increase in size of the vertical error bars. In Fig. 15b we can observe that the TPR for H1V1 and L1V1 decreases steeply for  $D_l \gtrsim 0.6$  Mpc, possibly due to the lower sensitivity of V1. Moreover, H1L1 and H1L1V1 have a similar trend, with a drop in performance at  $D_l \gtrsim 1.2$  Mpc. As in the case of the injected SNR, there is less statistic for  $D_l \gtrsim 1$  Mpc. In future work, we could conduct a broader injection campaign to mitigate these issues.

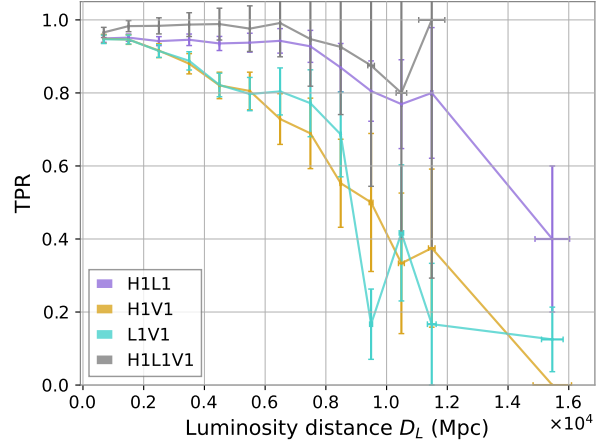
## VI. DISCUSSION

In the previous Sections, we showcased the utilization of matched-filtering triggers to learn the patterns, or tracks, of different types of signals from single-GW detectors. It is relevant to highlight the reliance of the ML model on a mere 6 parameters (see Eq. 10) to perform the multi-classification task. This approach does not only recover successfully the test set of the population we have trained on but also accurately recovers injections of GW signals. In future works, it would be interesting to compare these results with state-of-the-art algorithms to better quantify the performance of our method.

Despite these achievements, certain limitations must be noted. The primary constraint is that we characterize the feature vector of our input with only 6 features. Before this investigation, we have not utilized a feature selection procedure for the definition of the input feature vector, as we have limited ourselves to variables that



(a) TPR as a function of injected SNR.



(b) TPR as a function of luminosity distance.

FIG. 15: TPR, i.e., the rate of correctly classified GW injections, at iFAR = 0.1 (equivalent to one per 10 years) for different detector combinations in triple time (H1L1V1). Error bars represent the standard error at 3 standard deviations.

are well-understood in the GW community. Such an approach could enhance the performance of the model.

Another limitation pertains to the ranking statistic itself. In this work, we have combined the statistic of the ML model with standard measurements such as SNR and  $\xi^2$ . Due to time constraints, this exploration was limited, but it does not imply that our ranking statistic is the optimal one. In future works we will further explore enhancing our statistic with different variable combinations. However, we expect that the largest improvement would come from the inclusion of the time dimension, i.e. the behavior of the signal through the template bank. This will most likely provide valuable insight useful to enhance the distinction between these classes.

## VII. CONCLUSION

In this investigation, we propose a flexible method to detect CBC signals combining the robustness of matched-filtering as an optimal filter, with the generalization power of ML algorithms. Thus, we construct an MLP model to learn from sets of the matched filtering triggers, labelled here as tracks, and perform a multi-classification task in a single detector. Specifically, we tackle the IMBH search of GstLAL during O3, but this method could be extended to other CBC signals and matched-filtering algorithms.

Since multiple templates could potentially match a given signal, meaning that the input to the model would have variable length, in this proof-of-concept work, we have reduced the dimensionality of the problem by averaging the matching templates and weighting them by SNR. Another difficulty of our task is to deal with highly unbalanced data, so to mitigate this problem, we undersample large classes and oversample small classes. Nonetheless, this could bias the model towards certain repetitive features, so to avoid overfitting we employ 9-fold-cross-validation and early-stopping algorithm. In the following, we revisit some of the major results of this method:

- *Controlled data set:* We trained our model in O3a data, and we tested its generalization ability in O3b. We used the standard statistics to test its robustness. We obtained accuracies of 98.8%, 95.8% and 99.3% for H1, L1 and V1 in O3a test set, while showing a sharp decrease in accuracy of 73.0%, 67.5% and 75.0% for H1, L1 and V1 in O3b test set (see Section V C). This drop in performance was caused by the influence of glitches, so we decided to implement time coincidence to lower this background.
- *Novel time coincidence:* To enhance the performance of the model, we enforce time coincidence among detectors, which greatly reduces the background of glitches. As we are dealing with clusters of triggers, instead of single triggers, we take the average time of the cluster and the light time travel between detectors to consider that tracks from different detectors coincide in time (see Section IV D).
- *Computational efficiency:* This method only uses 6 variables to perform this classification (see Eq. 10), and while its training is intensive, we can classify

a given input in  $2.9 \times 10^{-6}$  s. As this process is highly parallelizable, this computation essentially is inexpensive.

- *Construction of accidental background:* We constructed the time shift background to assess the significance of our ranking statistic (see Section V D) and tested its performance on simulated IMBH signals. With an iFAR = 0.1 yr, we have a TPR  $\gtrsim 0.8$  at SNR  $\sim 5$  for all detector combination. Regarding the luminosity distance, H1L1 and H1L1V1 have a TPR  $\gtrsim 0.8$ , with a drop in performance at  $D_L \gtrsim 1.2$  Mpc.

In summary, this method has shown not only to have a robust performance in a controlled environment classifying IMBH injections but also shown its generalization power in finding simulated GW signals, demonstrating that it is possible to form a synergistic relationship between current state-of-the-art matched-filtering techniques and novel ML methods.

In future work, we will explore a different ML method that can process varying length inputs, as the time information might be relevant to enhance the performance of the model. For this aim, several ML algorithms could be employed, such as recurrent neural network [96] or even transformers with an attention mechanism [97]. With such a model we could perform a fair comparison with state-of-the-art pipelines to quantify the potential improvements in IMBH searches that this technology may offer. Furthermore, and as we mentioned before since this methodology is flexible and simply relies on matched filtering computation, we could extend it to other CBC signals and state-of-the-art matched-filtering algorithms.

## VIII. ACKNOWLEDGEMENTS

We would like to thank Harsh Narola for his useful comments. S.C. is supported by the National Science Foundation under Grant No. PHY-2309332. This project was supported by Nikhef Laboratory, and we would like to thank the Nikhef Computing group. The authors are also grateful for computational resources provided by the LIGO Laboratory and supported by the National Science Foundation Grants No. PHY-0757058 and No. PHY-0823459. This material is based upon work supported by NSF's LIGO Laboratory which is a major facility fully funded by the National Science Foundation.

---

[1] D. Richstone et al., *Nature* **395**, A14 (1998), [arXiv:astro-ph/9810378](#).  
[2] E. H. T. Collaboration, *The Astrophysical Journal Letters* **930**, L12 (2022).

[3] J. P. Gardner et al., *Space Sci. Rev.* **123**, 485 (2006), [arXiv:astro-ph/0606175](#).  
[4] H. Suh et al., arXiv pre-print (2024), [arXiv:2405.05333 \[astro-ph.GA\]](#).



- [5] S. E. Woosley et al., *Nature* **450**, 390 (2007), [arXiv:0710.3314 \[astro-ph\]](#).
- [6] S. E. Woosley and A. Heger, *Astrophys. J. Lett.* **912**, L31 (2021), [arXiv:2103.07933 \[astro-ph.SR\]](#).
- [7] J. E. Greene, J. Strader, and L. C. Ho, *Annual Review of Astronomy and Astrophysics* **58**, 257 (2020), [arXiv:1911.09678 \[astro-ph.GA\]](#).
- [8] R. Abbott et al. (LIGO Scientific, VIRGO, KAGRA), *Astron. Astrophys.* **659**, A84 (2022), [arXiv:2105.15120 \[astro-ph.HE\]](#).
- [9] J. Aasi et al. (LIGO Scientific), *Class. Quant. Grav.* **32**, 074001 (2015), [arXiv:1411.4547 \[gr-qc\]](#).
- [10] F. Acernese et al. (VIRGO), *Class. Quant. Grav.* **32**, 024001 (2015), [arXiv:1408.3978 \[gr-qc\]](#).
- [11] B. P. Abbott et al. (LIGO Scientific, Virgo), *Phys. Rev. X* **9**, 031040 (2019), [arXiv:1811.12907 \[astro-ph.HE\]](#).
- [12] B. P. Abbott et al. (LIGO Scientific, Virgo), *Phys. Rev. D* **100**, 064064 (2019), [arXiv:1906.08000 \[gr-qc\]](#).
- [13] R. Abbott et al. (LIGO Scientific, VIRGO), *arXiv e-prints* (2021), [arXiv:2108.01045 \[gr-qc\]](#).
- [14] R. Abbott et al. (LIGO Scientific, Virgo), *Phys. Rev. X* **11**, 021053 (2021), [arXiv:2010.14527 \[gr-qc\]](#).
- [15] R. Abbott et al. (LIGO Scientific, Virgo), *Phys. Rev. Lett.* **125**, 101102 (2020).
- [16] R. Abbott et al. (LIGO Scientific Collaboration, Virgo Collaboration, and KAGRA Collaboration), *Phys. Rev. X* **13**, 041039 (2023).
- [17] S. Klimenko et al., *Phys. Rev. D* **93**, 042004 (2016), [arXiv:1511.05999 \[gr-qc\]](#).
- [18] M. Drago et al., *SoftwareX* **14**, 100678 (2021).
- [19] B. Allen et al., *Phys. Rev. D* **85**, 122006 (2012), [arXiv:gr-qc/0509116](#).
- [20] C. Messick et al., *Phys. Rev. D* **95**, 042001 (2017), [arXiv:1604.04324 \[astro-ph.IM\]](#).
- [21] S. A. Usman et al., *Classical and Quantum Gravity* **33**, 215004 (2016).
- [22] S. Sachdev et al., *arXiv pre-print* (2019), [arXiv:1901.08580 \[gr-qc\]](#).
- [23] K. Chandra et al., *Phys. Rev. D* **104**, 042004 (2021), [arXiv:2106.00193 \[gr-qc\]](#).
- [24] K. Chandra et al., “Salient features of the optimised PyCBC IMBH search,” (2021), [arXiv:2110.01879 \[gr-qc\]](#).
- [25] L. Tsukada et al., *Phys. Rev. D* **108**, 043004 (2023), [arXiv:2305.06286 \[astro-ph.IM\]](#).
- [26] B. P. Abbott et al. (LIGO Scientific, Virgo), *Class. Quant. Grav.* **35**, 065010 (2018), [arXiv:1710.02185 \[gr-qc\]](#).
- [27] F. Acernese et al. (Virgo), *Class. Quant. Grav.* **40**, 185006 (2023), [arXiv:2210.15633 \[gr-qc\]](#).
- [28] J. Glanzer et al., *Class. Quant. Grav.* **40**, 065004 (2023), [arXiv:2208.12849 \[gr-qc\]](#).
- [29] M. Zevin et al., *Class. Quant. Grav.* **34**, 064003 (2017), [arXiv:1611.04596 \[gr-qc\]](#).
- [30] J. Yan et al., *Phys. Rev. D* **105**, 043006 (2022).
- [31] S. Ruder, *arXiv preprint* [arXiv:1609.04747](#) (2016).
- [32] T. Mishra et al., *Phys. Rev. D* **104**, 023014 (2021), [arXiv:2105.04739 \[gr-qc\]](#).
- [33] M. J. Szczepańczyk et al., *Phys. Rev. D* **107**, 062002 (2023), [arXiv:2210.01754 \[gr-qc\]](#).
- [34] V. Gayathri et al., *Phys. Rev. D* **102**, 104023 (2020), [arXiv:2008.01262 \[gr-qc\]](#).
- [35] D. Lopez et al., *Phys. Rev. D* **105**, 063024 (2022), [arXiv:2112.06608 \[gr-qc\]](#).
- [36] K. Aamodt et al. (ALICE), *JINST* **3**, S08002 (2008).
- [37] L. Blanchet et al., *Phys. Rev. Lett.* **74**, 3515 (1995).
- [38] A. Buonanno and T. Damour, *Phys. Rev. D* **59**, 084006 (1999), [arXiv:gr-qc/9811091](#).
- [39] C. Messick et al., *Physical Review D* **95**, 042001 (2017), [arXiv:1604.04324 \[astro-ph.IM\]](#).
- [40] C. Hanna et al., *Physical Review D* **101**, 022003 (2020).
- [41] K. Cannon et al., *SoftwareX* **14**, 100680 (2021).
- [42] B. Allen, *Physical Review D* **71**, 062001 (2005).
- [43] T. D. Canton et al., *Physical Review D* **90**, 082004 (2014).
- [44] A. H. Nitz et al., *The Astrophysical Journal* **849**, 118 (2017).
- [45] G. S. Davies et al., *Physical Review D* **102**, 022004 (2020).
- [46] A. H. Nitz et al., *Physical Review D* **98**, 024050 (2018).
- [47] T. Adams et al., *Classical and Quantum Gravity* **33**, 175012 (2016).
- [48] F. Aubin et al., *Classical and Quantum Gravity* **38**, 095004 (2021).
- [49] Q. Chu et al., *Physical Review D* **105**, 024023 (2022).
- [50] J. Creighton and W. Anderson, “Gravitational-wave physics and astronomy: An introduction to theory, experiment and data analysis,” (Wiley, 2011).
- [51] P. Ajith, N. Fotopoulos, S. Privitera, A. Neunzert, and A. J. Weinstein, *Phys. Rev. D* **89**, 084041 (2014), [arXiv:1210.6666 \[gr-qc\]](#).
- [52] K. Cannon, C. Hanna, and J. Peoples, *arXiv pre-print* (2015), [arXiv:1504.04632 \[astro-ph.IM\]](#).
- [53] C. Hanna et al., *Phys. Rev. D* **101**, 022003 (2020), [arXiv:1901.02227 \[gr-qc\]](#).
- [54] S. Khan et al., *Phys. Rev. D* **93**, 044007 (2016), [arXiv:1508.07253 \[gr-qc\]](#).
- [55] M. Cabero et al., *Class. Quant. Grav.* **36**, 15 (2019), [arXiv:1901.05093 \[physics.ins-det\]](#).
- [56] B. P. Abbott et al. (LIGO Scientific, Virgo), *Class. Quant. Grav.* **35**, 065010 (2018), [arXiv:1710.02185 \[gr-qc\]](#).
- [57] B. P. Abbott et al. (LIGO Scientific, Virgo), *Class. Quant. Grav.* **33**, 134001 (2016), [arXiv:1602.03844 \[gr-qc\]](#).
- [58] S. Soni et al., *Class. Quant. Grav.* **38**, 195016 (2021), [arXiv:2103.12104 \[gr-qc\]](#).
- [59] L. Nuttall et al., *Class. Quant. Grav.* **32**, 245005 (2015), [arXiv:1508.07316 \[gr-qc\]](#).
- [60] D. George and E. A. Huerta, *Phys. Rev. D* **97**, 044039 (2018), [arXiv:1701.00008 \[astro-ph.IM\]](#).
- [61] H. Gabbard et al., *Phys. Rev. Lett.* **120**, 141103 (2018), [arXiv:1712.06041 \[astro-ph.IM\]](#).
- [62] K. Sharma, K. Chandra, and A. Pai, *arXiv pre-print* (2022), [arXiv:2208.02545 \[astro-ph.HE\]](#).
- [63] P. G. Krastev., *Phys. Lett. B* **803**, 135330 (2020), [arXiv:1908.03151 \[astro-ph.IM\]](#).
- [64] A. Menéndez-Vázquez et al., *Phys. Rev. D* **103**, 062004 (2021), [arXiv:2012.10702 \[gr-qc\]](#).
- [65] G. Baltus et al., *Phys. Rev. D* **103**, 102003 (2021), [arXiv:2104.00594 \[gr-qc\]](#).
- [66] G. Baltus et al., *Phys. Rev. D* **106**, 042002 (2022), [arXiv:2205.04750 \[gr-qc\]](#).
- [67] M. Cavaglia et al., *arXiv e-prints*, [arXiv:2002.04591](#) (2020), [arXiv:2002.04591 \[astro-ph.IM\]](#).
- [68] J. M. Antelis et al., *Phys. Rev. D* **105**, 084054 (2022).
- [69] A. Iess et al., *arXiv pre-print* (2020), [arXiv:2001.00279 \[gr-qc\]](#).

- [70] A. Iess et al., *Astron. Astrophys.* **669**, A42 (2023), [arXiv:2301.09387 \[astro-ph.IM\]](#).
- [71] M. López Portilla et al., *Phys. Rev. D* **103**, 063011 (2021), [arXiv:2011.13733 \[astro-ph.IM\]](#).
- [72] M. López et al. (2021).
- [73] Q. Meijer et al., arXiv pre-print (2023), [arXiv:2308.12323 \[astro-ph.IM\]](#).
- [74] V. Boudart and M. Fays, *Phys. Rev. D* **105**, 083007 (2022), [arXiv:2201.08727 \[gr-qc\]](#).
- [75] V. Boudart, *Phys. Rev. D* **107**, 024007 (2023), [arXiv:2210.04588 \[gr-qc\]](#).
- [76] S. Bahaadini et al., *Information Sciences* **444**, 172 (2018).
- [77] P. Laguarda et al., arXiv pre-print (2023), [arXiv:2310.03453 \[astro-ph.IM\]](#).
- [78] M. Lopez et al., *Phys. Rev. D* **106**, 023027 (2022), [arXiv:2203.06494 \[astro-ph.IM\]](#).
- [79] M. Lopez et al., arXiv pre-print (2022), [arXiv:2205.09204 \[astro-ph.IM\]](#).
- [80] J. Powell et al., *Class. Quant. Grav.* **40**, 035006 (2023), [arXiv:2207.00207 \[astro-ph.IM\]](#).
- [81] J. Yan, A. P. Leung, and D. C. Y. Hui, *Mon. Not. Roy. Astron. Soc.* **515**, 4606 (2022), [arXiv:2207.04001 \[astro-ph.HE\]](#).
- [82] T. Dooney, S. Bromuri, and L. Curier, in *2022 IEEE International Conference on Big Data (Big Data)* (IEEE Computer Society, Los Alamitos, CA, USA, 2022) pp. 5468–5477.
- [83] T. Dooney et al., *Phys. Rev. D* **110**, 022004 (2024), [arXiv:2401.16356 \[physics.ins-det\]](#).
- [84] E. Cuoco et al., *Mach. Learn. Sci. Tech.* **2**, 011002 (2021), [arXiv:2005.03745 \[astro-ph.HE\]](#).
- [85] G. James et al., “An introduction to statistical learning: with applications in r,” (Springer, 2013).
- [86] Y. Bengio and Y. Grandvalet, *J. Mach. Learn. Res.* **5**, 1089 (2004).
- [87] F. Rosenblatt, *Psychological review* **65**, 386 (1958).
- [88] A. Paszke et al., “Pytorch: An imperative style, high-performance deep learning library,” (2019).
- [89] A. F. Agarap, arXiv preprint [arXiv:1803.08375](#) (2018).
- [90] D. P. Kingma and J. Ba, arXiv pre-print (2014), [arXiv:1412.6980 \[cs.LG\]](#).
- [91] Y. Yao, L. Rosasco, and A. Caponnetto, *Constructive Approximation* **26**, 289 (2007).
- [92] H. Larry and I. Niven, *Mathematics Magazine* **58**, 151 (1985).
- [93] J. Komić, “Harmonic mean,” in *International Encyclopedia of Statistical Science*, edited by M. Lovric (Springer Berlin Heidelberg, Berlin, Heidelberg, 2011) pp. 622–624.
- [94] K. M. Ting, “Confusion matrix,” in *Encyclopedia of Machine Learning*, edited by C. Sammut and G. I. Webb (Springer, 2010) p. 209.
- [95] F. Robinet et al., *SoftwareX* **12**, 100620 (2020), [arXiv:2007.11374 \[astro-ph.IM\]](#).
- [96] R. C. Staudemeyer and E. Rothstein Morris, *arXiv e-prints*, [arXiv:1909.09586](#) (2019).
- [97] A. Vaswani and toehrs, *arXiv e-prints*, [arXiv:1706.03762](#) (2017), [arXiv:1706.03762](#).

## Appendix A: Results of LIGO Livingston and Virgo

### 1. Selecting a time window

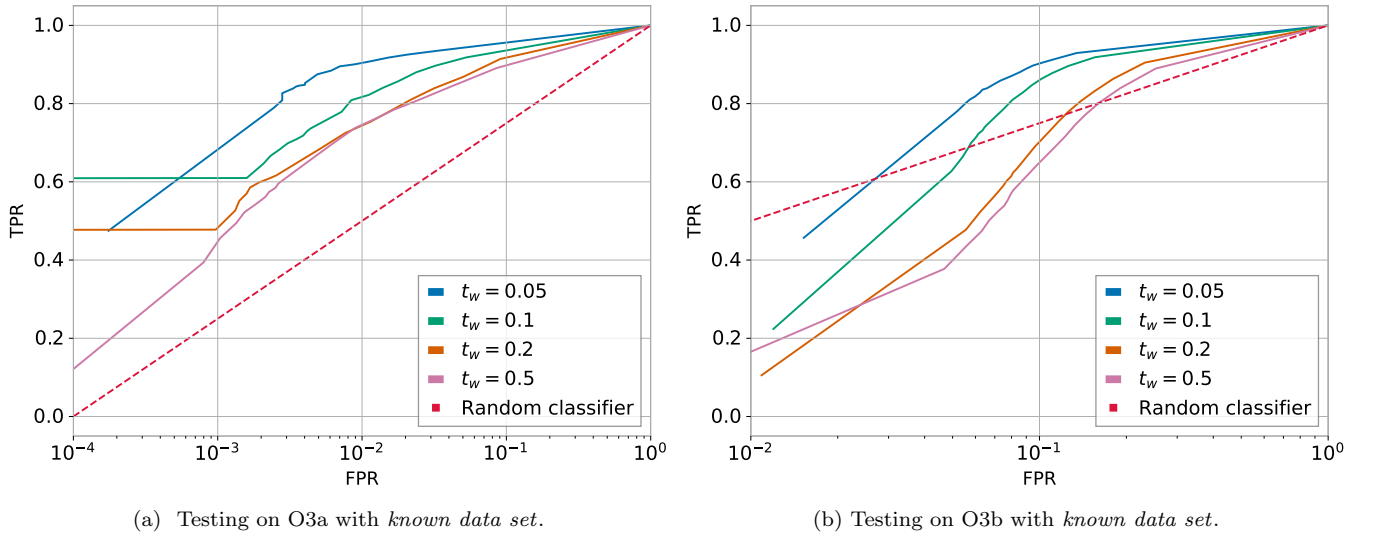


FIG. 16: ROC curve for different time windows  $t_w$  in L1, i.e. TPR as a function of False Positive Rate FPR in logarithmic scale. (Left) Testing in the *known data set* of O3a. (Right) Testing in the *known data set* of O3b. Note that the dashed line indicates a random guess.

Similarly to Section V A, we show in Fig. 16 and Fig. 17 the ROC curves of L1 and V1, respectively. We can observe that the TPR degrades as we increase the size of  $t_w$ , being this decrease sharper for the *known data set* of O3b than for the *known data set* of O3a. Notably, for V1 (see Fig. 17), while the ROC curve of  $t_w = 0.05$  in O3a is almost constant, the ROC curve of O3b decreases even faster than in the case of O3b L1.

As in Fig. 7, in Fig. 16 and Fig. 17  $t_w = 0.05$  s has a better performance than other time windows. A possible explanation for this behavior could be that since IMBH signals are short, larger time windows would add random triggers that are unrelated to the IMBH signal itself, biasing the model. Hence, as in Section V A we conclude that  $t_w = 0.05$  is the best time window for our task.

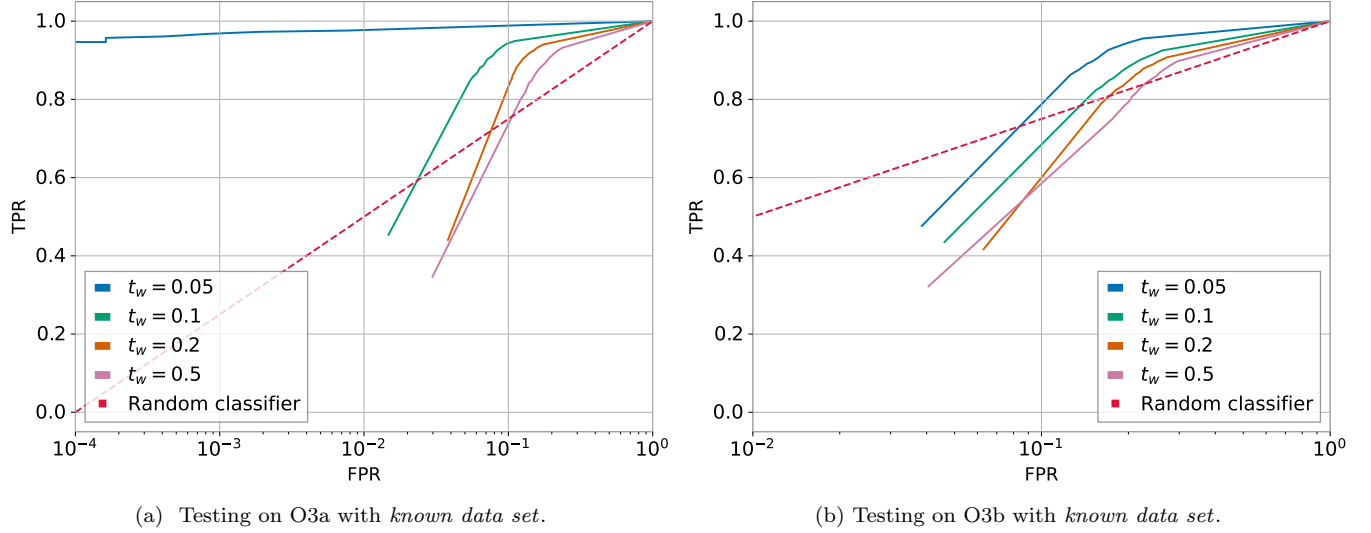


FIG. 17: ROC curve for different time windows  $t_w$  in V1, i.e. TPR as a function of False Positive Rate FPR in logarithmic scale. (Left) Testing in the *known data set* of O3a. (Right) Testing in the *known data set* of O3b. Note that the dashed line indicates a random guess.

## 2. Training with *known data set* of O3a

As in Section V B, in Fig. 20 we show, for training and validation, the mean accuracy (top row) and loss (bottom row) of the 9-fold cross-validation as a function of the epochs for L1 (left column) and V1 (right column), where the shadowed region represents  $\pm 3$  standard deviation. Note that the loss is plotted in logarithmic scale, so we can appreciate that there is less overfitting in V1 than in L1.

Another relevant point is the peaks in the loss functions, in particular around epoch 500 in L1 and around epoch 1000 in V1. These peaks indicate a change in the learning rate due to the adaptive learning rate scheme (see Section IV C for details).

## 3. Diving in to the *known data*: performance evaluation

Similarly to Section V C, where we want to test the generalization power between the *known data sets* of O3a and O3b for H1, we test the *known data sets* of O3a and O3b for L1 and V1, presenting their confusion matrices in Fig. 19 and Fig. 20, respectively. In Fig. 19a and Fig. 20a we can see that most inputs are correctly classified, yielding an accuracy of 95.81% in L1 and 99.27% in V1. Nonetheless, Fig. 19b and Fig. 20b we can observe an increase of false positives, decreasing the accuracy to 67.45% in L1 and 75.91% in V1.

In L1 (see Fig. 19b), the **Injections** are mostly correctly classified, with 1.2% mislabelled as **Blip**. However, 19.5% of **Low-Frequency-Burst**, 12.6% of **Whistle** and 11.3% of **Fast-Scattering** are incorrectly classified as **Injections**. Interestingly, 54.5% of **Low-Frequency-Burst** are misclassified as **Fast-Scattering** since they share a similar frequency range. In V1 (see Fig. 20b), the **Injections** are also mostly correctly classified, with 1.7% mislabelled as **Blip**. However, 61.5% of **Whistle** and 16.3% of **Blip** are incorrectly classified as **Injections**. It is also interesting to note

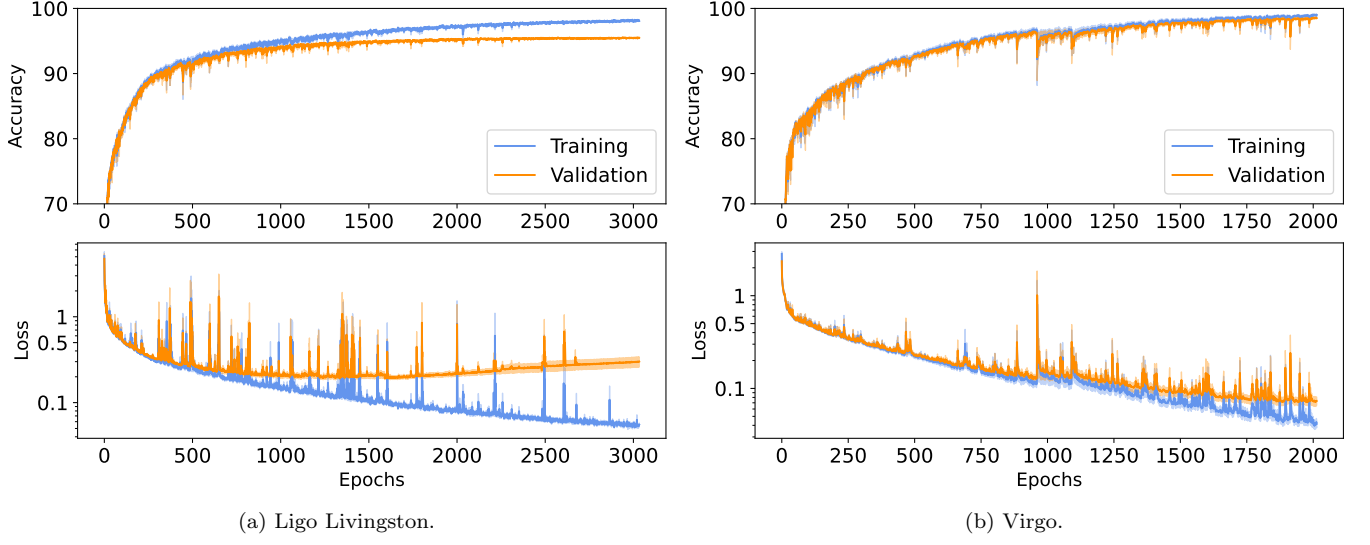


FIG. 18: Comparison between training and validating with 9-fold cross-validation during training and testing. (*Top row*) Mean accuracy at 3 standard deviations as a function of the epochs during. (*Bottom row*) Average 9-fold cross-validation loss as a function of the epochs.

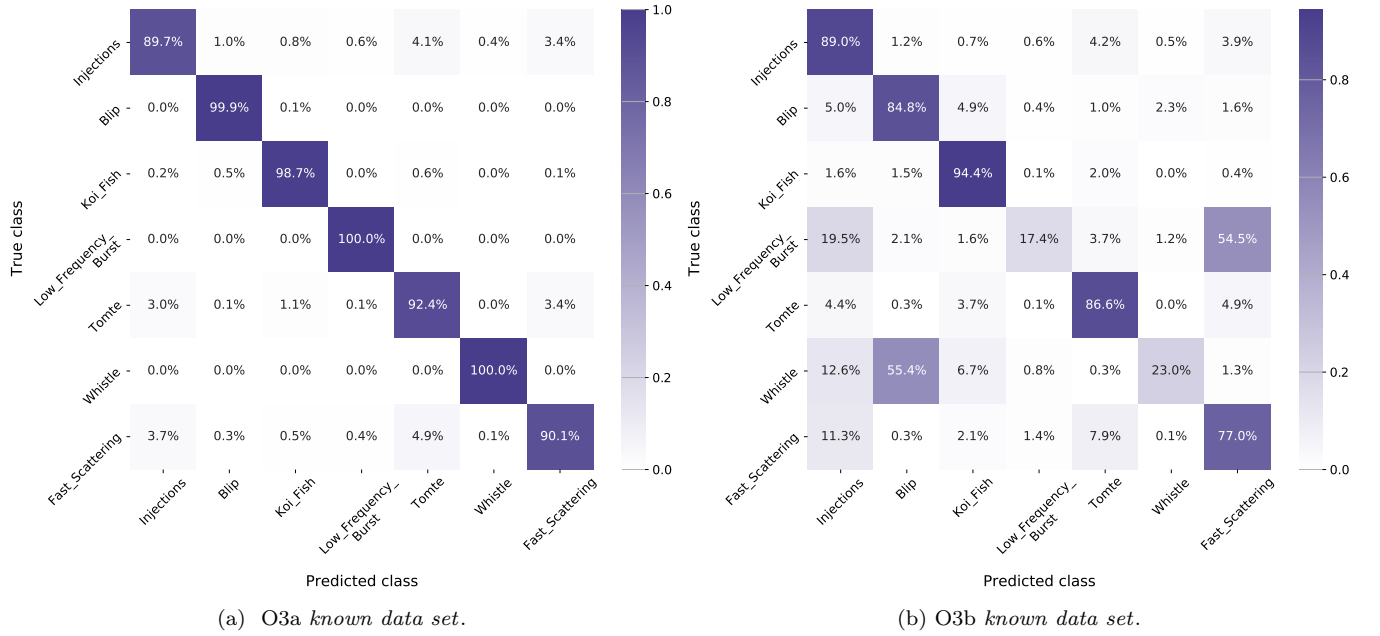


FIG. 19: Relative values of the confusion matrix for the test set for L1.

that in this detector only 4.9% **Low\_Frequency\_Burst** are misclassified as **Injections**, while a 10.2% and 11.5% of **Tomte** are incorrectly labelled as **Blip** or **Low\_Frequency\_Burst**, respectively.

Due to the poor generalization between O3a and O3b, we need to reduce the number of false positives. For this aim, we enforce time coincidence as in Section V C. Similarly to Fig. 12, in Fig. 21 we present the probability of being **Injection** ( $P_{inj}$ ) for L1 (left column) and V1 (right column). In the top row, we show the probability density of  $P_{inj}$  for the **Injection** class, and in the bottom row, we show the probability density of  $P_{inj}$  for any other class. When we enforce time coincidence, represented in yellow for L1 and V1, and in turquoise for H1 and V1,  $P_{inj}$  increases for the **Injection** class, while we completely discard all the other classes. Note that this is at the cost of reducing the number of correctly classified **Injections**.



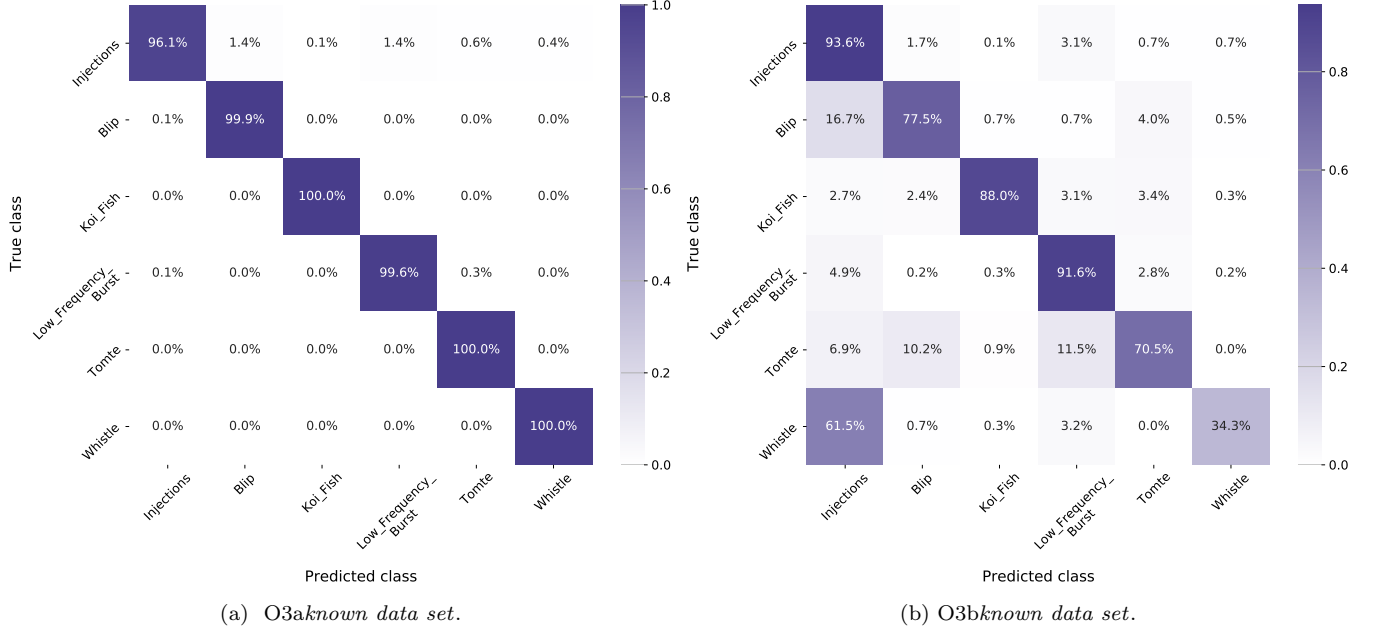


FIG. 20: Relative values of the confusion matrix for the test set for V1.

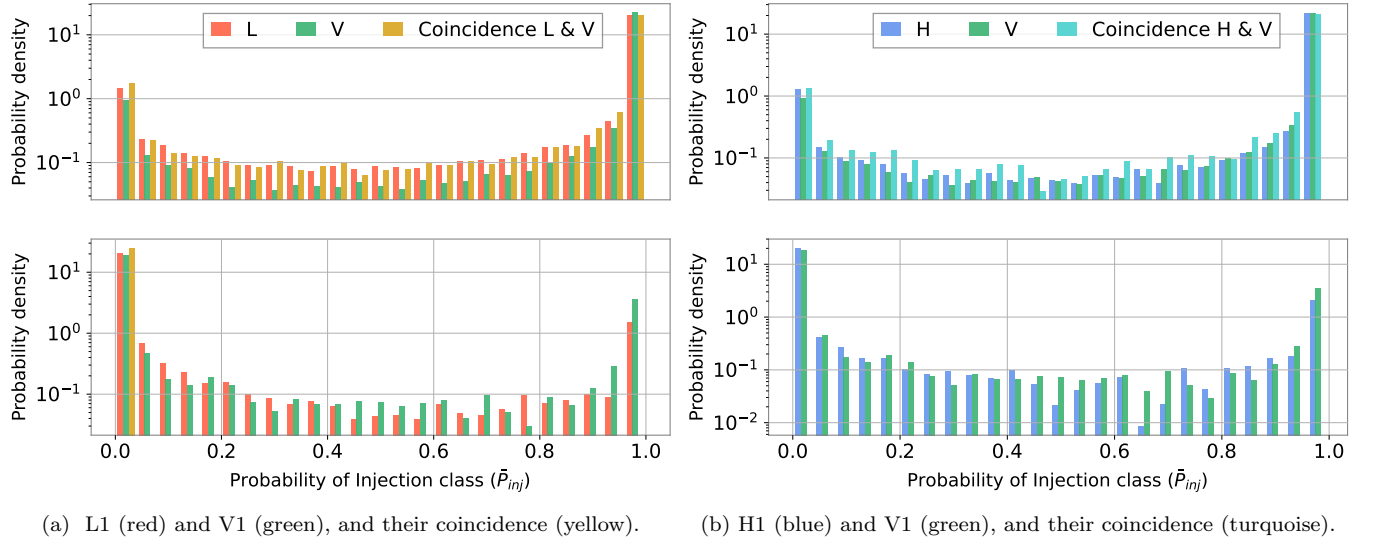


FIG. 21: Probability density of being **Injection** ( $P_{inj}$ ), using the *known* test set of O3b. (*Top row*) Probability density of elements in the **Injection** class in logarithmic scale. (*Bottom row*) Probability density of elements in all the other classes, i.e. glitches, in logarithmic scale. Given the counts of the  $i$ th bin  $c_i$  and its width  $b_i$ , we define the probability density as  $c_i/(\sum_i^N c_i \times b_i)$ , where  $N$  is the total number of bins of the histogram.