The Astrophysical Journal, 977:222 (15pp), 2024 December 20 https://doi.org/10.3847/1538-4357/ad901e

PyMerger: Detecting Binary Black Hole Mergers from the Einstein Telescope Using Deep Learning

Wathela Alhassan[1], T. Bulik[2], and M. Suchenek[1]

[1] Particle Astrophysics Science and Technology Centre, Nicolaus Copernicus Astronomical Center, Rektorska 4, 00-614 Warsaw, Poland
[2] Astronomical Observatory, University of Warsaw, Aleje Ujazdowskie 4, 00-478 Warsaw, Poland

Received 2024 August 7; revised 2024 October 15; accepted 2024 November 4; published 2024 December 16

Abstract

We present *PyMerger*, a Python tool for detecting binary black hole (BBH) mergers from the Einstein Telescope (ET), based on a deep residual neural network (ResNet) model. ResNet was trained on data combined from all three proposed subdetectors of ET (TSDCD) to detect BBH mergers. Five different lower-frequency cutoffs (F_{low})—5 Hz, 10 Hz, 15 Hz, 20 Hz, and 30 Hz—with the match-filter signal-to-noise ratio (MSNR) ranges 4–5, 5–6, 6–7, 7–8, and >8 were employed in the data simulation. Compared to previous work that utilized data from a single subdetector, the detection accuracy from TSDCD has shown substantial improvements, increasing from 60%, 60.5%, 84.5%, 94.5%, and 98.5% to 78.5%, 84%, 99.5%, 100%, and 100% for sources with MSNRs of 4–5, 5–6, 6–7, 7–8, and >8, respectively. The ResNet model is evaluated on the first ET mock data challenge (ET-MDC1) data set, where the model demonstrates strong performance in detecting BBH mergers, identifying 5566 out of 6578 BBH events, with optimal SNRs starting from 1.2 and a minimum and maximum D_L of 0.5 Gpc and 148.95 Gpc, respectively. Despite being trained only on BBH mergers without overlapping sources, the model achieves high BBH detection rates. Notably, even though the model was not trained on binary neutron star (BNS) and black hole-neutron star (BHNS) mergers, it successfully detected 11,477 BNS and 323BHNS mergers in ET-MDC1, with optimal SNRs starting from 0.2 and 1, respectively, indicating its potential for broader applicability.

Unified Astronomy Thesaurus concepts: Gravitational wave detectors (676); Gravitational waves (678); Convolutional neural networks (1938)

1. Introduction

This is a continuation of our previous work (W. Alhassan et al. 2023; hereafter referred to as WTM1) on the detection of binary black hole (BBH) gravitational-wave (GW) signals from the Einstein Telescope (ET) using deep learning (DL). ET (M. Punturo et al. 2010; M. Maggiore et al. 2020) is designed[3] to be an underground GW detector, where the seismic noise is much lower and hence also the level of Newtonian noise. ET, as shown in Figure 1, will consist of three nested arms (hereafter referred to as subdetectors) 10 km long each, in an equilateral triangle shape, with arm-opening angles of 60° (S. Hild et al. 2010; M. Branchesi et al. 2023). Compared to a 3 km length for advanced Virgo (F. Acernese et al. 2015) and KAGRA (Y. Aso et al. 2013; T. Akutsu et al. 2020), and 4 km for advanced LIGO (LIGO Collaboration et al. 2015), the ET 10 km V-shaped subdetectors will significantly improve its sensitivity, allowing the observation of GWs at lower frequencies that reach ≈2 Hz (S. Hild et al. 2011). Each subdetector of ET will consist of two interferometers, one optimized for low-frequency and one for high-frequency sensitivity. In this work, it is assumed that each subdetector has a single interferometer.

The current standard method for detecting GW signals is match filtering, which is considered the most sensitive algorithm targeting compact binary sources, such as BBHs with spin-aligned components on quasi-circular orbits (D. Gerosa et al. 2013; C. L. Rodriguez et al. 2015, 2016). One of the main drawbacks of such algorithms is their computational complexity, especially when targeting eccentric BBH systems (I. Harry et al. 2016). In addition, match-filtering algorithms might miss signals generated by compact binary populations in dense stellar environments (E. A. Huerta et al. 2014, 2017; S. Klimenko et al. 2016). These limitations open the door to potential applications of DL algorithms, which do not require waveform templates for detections.

Application examples of DL in astrophysics, and more specifically in GWs, were shown in WTM1. Notably, more work has been presented since then, highlighting the significant interest in this promising technology for addressing the considerable challenges posed by the vast amount of data expected from current and upcoming detectors. The Spy project (J. Glanzer et al. 2023) applied machine learning (ML) for glitch classification using data up to the end of the third observing run of LIGO. In this analysis, 233,981 glitches from LIGO Hanford and 379,805 glitches from LIGO Livingston were classified into morphological classes. A related study employed generative adversarial networks to simulate hundreds of synthetic images representing the 22 most common types of glitches observed in the LIGO, KAGRA, and Virgo detectors. A neural network model was then used to classify the simulated glitches, achieving an average accuracy of 99.0% (J. Powell et al. 2023). Accelerating artificial intelligence models for rapid GW detection was recently conducted by P. Chaturvedi et al. (2022). In their work, the ThetaGPU supercomputer was leveraged with the NVIDIA TensorRT–optimized AI ensemble to process a month of advanced LIGO data; the process took 50 s. In R.-Q. Yan et al. (2022), a novel model called a GW squeeze-and-excitation shrinkage network (GW-SESNet) was inspired by attention mechanisms, a mechanism in DL used to highlight important features for the classification of real

[3] See https://www.et-gw.eu/index.php/relevant-et-documents for relevant documents on the ET design.

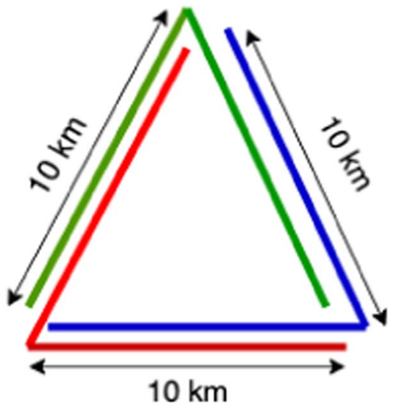


Figure 1. Simplified layout of the three core subdetectors of ET.

signals, noise, and glitches. GW-SESNet utilized the coherent signal-to-noise ratio (SNR) from multiple detectors and achieved recall values of 0.84, 0.98, and 0.98 for noise, real signals, and glitches, respectively. The role of advanced ML tools in the coming decades, either independently or in tandem with traditional data analysis techniques, may become crucial, considering the vast amount of the data expected (R. Shah et al. 2023).

The detection of GW sources represents the first step in GW data analysis; hence, quick and fast detection is required for the subsequent steps, such as source parameter estimation or alerting other telescopes for multimessenger follow-up and studies of the detected source. Fast detection can also be employed for data marking, facilitating a detailed Bayesian parameter estimation analysis of the relevant part of the detector output at a later stage. Coherent multidetector observation, in which multiple detectors are coupled together, is beneficial for improving the coherent SNR (D. Macleod et al. 2016), making it a current trend in GW signal detection.

In WTM1, our work focused on the detection of BBH sources that were generated using only one single subdetector of ET, with a low-frequency cutoff (the starting frequency of the binary systems, which increases as the binary system evolves and the components get closer to each other) of 30 Hz and with five match-filter SNR (hereafter referred to as MSNR) ranges: 4–5, 5–6, 6–7, 7–8, and >8. Expanding upon our previous work, we explore the detection efficiency of BBHs using data combined from all the three proposed subdetectors of ET (hereafter referred to as three-subdetector coherent data or TSDCD), with five different lower-frequency cutoffs ($F_{\rm low}$): 5 Hz, 10 Hz, 15 Hz, 20 Hz, and 30 Hz, employing identical MSNR ranges for each frequency. In addition, we use the same luminosity distance (D_L) range of 145 Mpc–120 Gpc as in WTM1 for our BBH sources. The main objective is to compare the previously obtained results from single-subdetector data (hereafter referred to as SSDD) to the results from TSDCD.

The structure of this work is as follows. In Section 2, we describe the method and tools used for the generation of TSDCD and the preprocessing pipeline employed for spectrogram preparation. Section 3 discusses deep residual neural networks (ResNets) and the training and evaluation processes. Evaluations of the ResNet model on noncontinuous and continuous TSDCD are presented in Sections 4 and 5. An evaluation of the first ET mock data challenge (ET-MDC1) is described in Section 6. A description of the PyMerger software is presented in Section 7. Finally, a summary of this work and future work plans are presented in Section 8.

2. ET Data Simulation

The strain in each subdetector of ET $h_{E\alpha}(t)$ is given as

$$h_{E\alpha}(t) = F_\alpha^{+}(\theta,\phi,\psi)h_+(t) + F_\alpha^{\times}(\theta,\phi,\psi)h_\times(t), \tag{1}$$

where α is the index of the subdetector; $\alpha = (1, 2, 3)$. F_α^+ and $F_\alpha^\times$ are antenna response functions for the two GW polarizations of each subdetector α, which depend on the sky localization (R.A. θ and decl. ϕ) and polarization angle ψ and take into account the motion of the Earth (P. Jaranowski et al. 1998). For a single subdetector, F^+ and $F^\times$ at time t and angle η between the interferometer arms are given as:

$$F_+(t) = \sin\eta[a(t)\cos 2\psi + b(t)\sin 2\psi], \tag{2}$$

$$F_\times(t) = \sin\eta[b(t)\cos 2\psi - a(t)\sin 2\psi], \tag{3}$$

where

$$\begin{aligned} a(t) = {}& \frac{1}{16}\sin 2\gamma(3-\cos 2\lambda)(3-\cos 2\delta)\\ &\times\cos[2(\alpha-\phi_r-\Omega_r t)]\\ &-\frac{1}{4}\cos 2\gamma\sin\lambda(3-\cos 2\delta)\sin[2(\alpha-\phi_r-\Omega_r t)]\\ &+\frac{1}{4}\sin 2\gamma\sin 2\lambda\sin 2\delta\cos[\alpha-\phi_r-\Omega_r t]\\ &-\frac{1}{2}\cos 2\gamma\cos\lambda\sin 2\delta\sin[\alpha-\phi_r-\Omega_r t]\\ &+\frac{3}{4}\sin 2\gamma\cos 2\lambda\cos 2\delta, \end{aligned} \tag{4}$$

$$\begin{aligned} b(t) = {}& \cos 2\gamma\sin\lambda\sin\delta\cos[2(\alpha-\phi_r-\Omega_r t)]\\ &+\frac{1}{4}\sin 2\gamma(3-\cos 2\lambda)\sin\delta\sin[2(\alpha-\phi_r-\Omega_r t)]\\ &+\cos 2\gamma\cos\lambda\cos\delta\cos[\alpha-\phi_r-\Omega_r t]\\ &+\frac{1}{2}\sin 2\gamma\sin 2\lambda\cos\delta\sin[\alpha-\phi_r-\Omega_r t], \end{aligned} \tag{5}$$

where λ denotes the latitude of ET's location, which we define the same as VIRGO. Ω_r is Earth's rotational angular velocity, and ϕ_r is the phase defining the position of the Earth in its diurnal motion at $t = 0$.

γ determines the orientation of the detector arms and is measured counterclockwise from the east to the bisector of the interferometer arms. For ET in the triangular configuration, $\eta = 60°$.

The polarizations of the GW $h_+(t)$ and $h_\times(t)$ in Equation (1), from an inspiraling compact binary system such as BBH, are given as:

$$\begin{aligned} h_+(t) = {}& -\frac{1+\cos^2\iota}{2}\left(\frac{GMc^2D_L}{2(tc-t)}\right)^{1/4}\\ &\times\cos[2\Phi_c + 2\Phi(t-tc; M,\mu)], \end{aligned} \tag{6}$$

$$h_\times(t) = -\cos\iota \left(\frac{GMc^2 D_L}{2(tc - t)}\right)^{1/4} \times \sin[2\Phi_c + 2\Phi(t - tc; M, \mu)], \tag{7}$$

where ι denotes the inclination angle of the orbital plane of the binary system relative to the observer. c and G represent the speed of light and the gravitational constant, respectively. The quantity μ denotes the binary system's reduced mass. t_c and Φ_c represent, respectively, the time and the phase at which the waveform terminates. $t - t_c$ represents the time difference between the current time t and the coalescence time t_c. The function $\Phi(t - t_c; M, \mu)$ represents the binary system's orbital phase. M is the source chirp mass, which depends on the mass components of the binary system m_1 and m_2, and is given as

$$M = (m_1 m_2)^{3/5}/(m_1 + m_2)^{1/5}. \tag{8}$$

In WTM1, our emphasis was on simulating BBH sources using a single subdetector of ET. In this work, we replicated the exact BBH source parameters, data generation tools, and methods employed in WTM1 (for details, please see Section 2 of WTM1) for each subdetector of ET to obtain TSDCD.

2.1. Simulation

We obtained the mass components and redshift (z) for simulating BBH systems from K. Belczynski et al. (2020). A broad set of compact binary models was developed by K. Belczynski et al. (2020) using the population synthesis code StarTrack (K. Belczynski et al. 2002a, 2002b, 2002c), which has been significantly upgraded in K. Belczynski et al. (2008) to focus on processes leading to the formation and further evolution of compact objects.

Restricting the BBH masses to 15–56 $M_\odot$ and redshift z values from 0.033 to 11 allows us to focus on low and medium masses within our selected MSNR ranges. In addition, we have considered five $F_{\rm low}$ bounds, namely 5 Hz, 10 Hz, 15 Hz, 20 Hz, and 30 Hz. Other parameters, such as the sky position angles (θ and ϕ), ι, ψ, and coalescence phases, were sampled from a uniform distribution, as in WTM1.

Assuming the standard ΛCDM cosmology (Planck Collaboration et al. 2016)—where the Hubble constant $H_0 = 67.74\ {\rm km\ s^{-1}\ Mpc^{-1}}$, the energy density parameter $\Omega_\Lambda = 0.6910$, and the matter density parameter $\Omega_m = 0.3075$—we convert the redshift z to D_L, using the analytical approximation provided by M. Adachi & M. Kasai (2012) for the D_L between the BBH source and the detector, given as:

$$D_L = (1 + z)cH_0 \int_0^z \frac{dz}{\sqrt{\Omega_\Lambda + \Omega_m(1 + z)^3}}, \tag{9}$$

where c is the speed of light in the vacuum.

We utilized the frequency-domain phenomenological nonspinning BBH model IMRPhenomD (S. Husa et al. 2016), which is part of the LALSuite (LIGO Scientific Collaboration et al. 2018), employing its Python interface (K. Wette 2020). IMRPhenomD is designed to generate GW signals from merging BBHs with nonprecessing spins. For each subdetector, using the parameters above, we individually generated a BBH GW signal ($h(t)_{E1}$, $h(t)_{E2}$, $h(t)_{E3}$), then added them into the simulated Gaussian detector noise n, assuming an additive, Gaussian, and stationary noise model. This process results in two data sets: the data $d_{1,2,3}$ (if the signal h is present) and d_n (if the data contain noise only), which can be written as follows:

$$d_1(t) = h_{E1}(t) + n_1(t), \tag{10}$$

$$d_2(t) = h_{E2}(t) + n_2(t), \tag{11}$$

$$d_3(t) = h_{E3}(t) + n_3(t), \tag{12}$$

$$d_n(t) = n_1(t) + n_2(t) + n_3(t). \tag{13}$$

To illustrate how the GW signal would look and to compare it to the detector's noise, Figure 2 shows the difference in phase between signals from the three subdetectors and the noise from a single subdetector in the background. The improvement in SNR through the utilization of data from all subdetectors of ET combined can be assessed by calculating the coherent SNR. The coherent SNR SNR$_{\rm coh}$, shown in Figure 3, enhances the observed signal's SNR by incorporating the coherent combination of multiple detectors (D. Macleod et al. 2016). To quantify the improvement in MSNR resulting from the combination of each subdetector, we calculated SNR$_{\rm coh}$ using the following formula:

$$\mathrm{SNR_{coh}} = \sqrt{\sum_{\alpha=1}^{3} \mathrm{MSNR}_\alpha^2}. \tag{14}$$

2.2. Spectrograms

The short-time Fourier transform (STFT; Q. Yin et al. 2013) was used to generate our TSDCD spectrogram samples (S_{E123}), as shown in Figure 4. Three spectrograms (S_{E1}, S_{E2}, and S_{E3}) were produced for each injected segment, $d_{1,2,3}(t)$, from each subdetector ($E1$, $E2$, and $E3$), respectively, and then stacked to form an RGB image, S_{E123}, represented as

$$S_{E123} = \begin{bmatrix} R(i, j) \\ G(i, j) \\ B(i, j) \end{bmatrix}, \tag{15}$$

where $R(i, j)$, $G(i, j)$, and $B(i, j)$ are the normalized intensity values for each channel at pixel i, j in $S_{E1}(i, j)$, $S_{E2}(i, j)$, and $S_{E3}(i, j)$, calculated as:

$$R(i, j) = \frac{S_{E1}(i, j)}{\max\_\mathrm{intensity}}, \tag{16}$$

$$G(i, j) = \frac{S_{E2}(i, j)}{\max\_\mathrm{intensity}}, \tag{17}$$

$$B(i, j) = \frac{S_{E3}(i, j)}{\max\_\mathrm{intensity}}, \tag{18}$$

where max _intensity represents the maximum intensity value across all S_{E1}, S_{E2}, and S_{E3}. A total of 125,000 sources were generated, with each of the five defined $F_{\rm low}$ groups containing 25,000 sources. These sources were evenly distributed among our five MSNR intervals. It is important to note that MSNR values vary between the three detectors using the same parameters, due to orientation and polarization. For each source, at least one detector must have the correct MSNR value to be accepted into the MSNR interval.

Similarly, to generate only-noise spectrograms, the following steps were taken: (1) a Gaussian noise with a color matching the power spectrum density of ET, sampled at ET's frequency, was generated for each subdetector; (2) three random segments of 5 s each were selected from each subdetector's noise, $n_1(t)$, $n_2(t)$, and $n_3(t)$; (3) STFT was

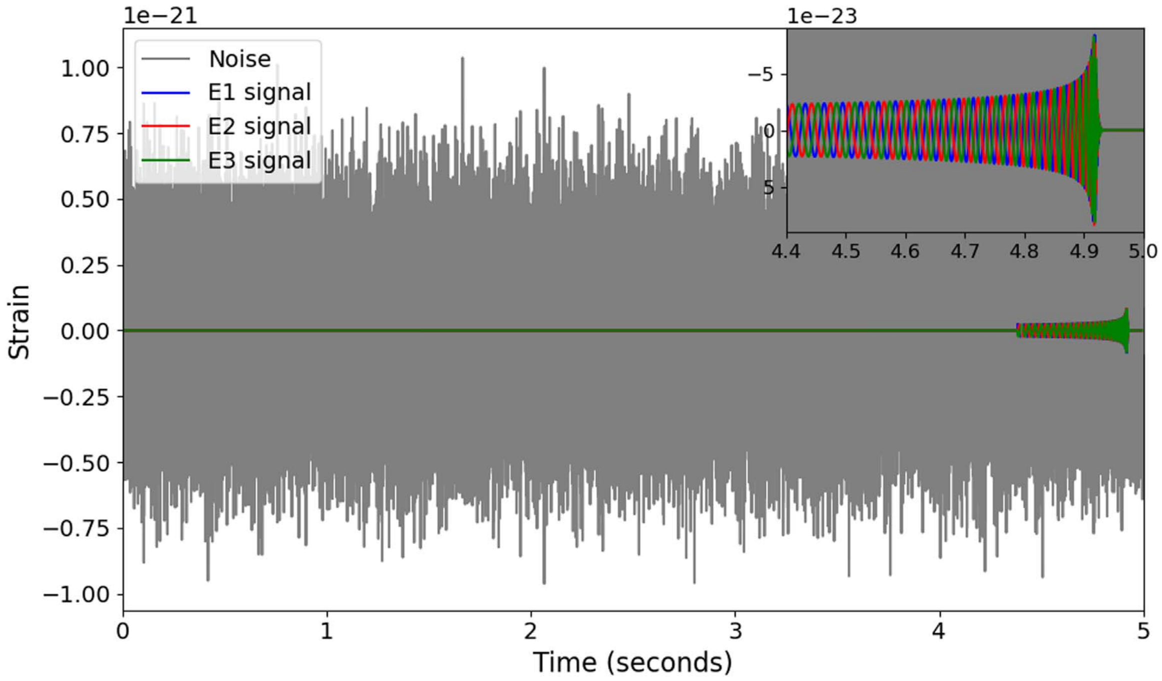


Figure 2. A 5 s segment of simulated noise from a single subdetector is overlapped with BBH GW signals. The gray lines represents noise from a single subdetector's noise $n_1(t)$. The blue, green, and red lines represent the BBH signals from E1, E2, and E3, respectively. The simulated BBH signal has an MSNR in the range 7–8, with component masses $m_1 = 20\,M_\odot$ and $m_2 = 25\,M_\odot$ as well as $F_{\rm low}$ of 30 Hz.

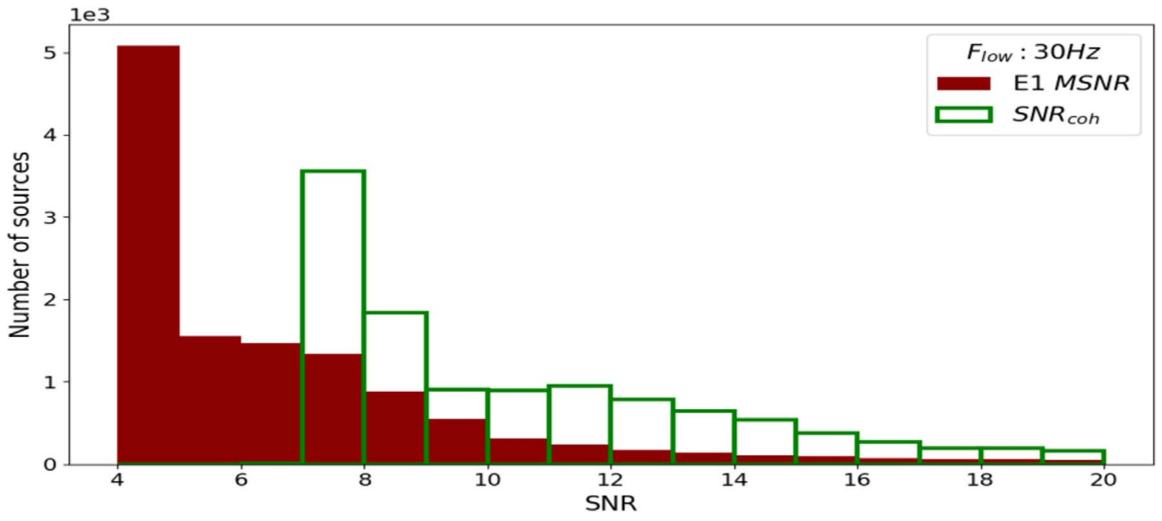


Figure 3. $\mathrm{SNR_{coh}}$ from the combination of each subdetector.

performed on each segment to produce a spectrogram; and (4) all three only-noise spectrograms were stacked together.

3. ResNet: Deep Residual Neural Networks

In WTM1, the performances of the four most popular computer vision neural network models were evaluated for the detection of BBH GW signals, namely VGG-16 and VGG-19 (K. Simonyan & A. Zisserman 2014), DenseNet (G. Huang et al. 2016), and ResNet (K. He et al. 2016; S. Wu et al. 2018). The ResNet model showed the best overall performance, and thus our goal in this study is to assess its performance on TSDCD compared to SSDD.

One of the great advantages of the ResNet network is its ability to overcome the vanishing gradient problem, which is considered as one of the challenging issues in training deep and complex neural networks models (Y. Hu et al. 2018; M. Roodschild et al. 2020). This was achieved by introducing residual blocks, which enable the training of hundreds of layers (E. Goceri 2019; N. Hasan et al. 2021). These blocks use an additive attribute to merge previous and future layers. ResNet is composed of several residual blocks. Each residual block is formed by adding a residual connection every two layers of conventional convolution. The key concept behind the residual connection, considered a pivotal breakthrough in DL, is to learn a residual function capable of representing the difference between the input X and the target $\mathcal{F}(X)$ (the desired mapping learned by the layer). A residual mapping $\mathcal{R}(X)$ is learned as follows:

$$\mathcal{R}(X) = \mathcal{F}(X) - X. \tag{19}$$

Then the sum of the residual mapping and the input is the output of the layer (Y) and can be calculated as:

$$Y = X + \mathcal{R}(X). \tag{20}$$

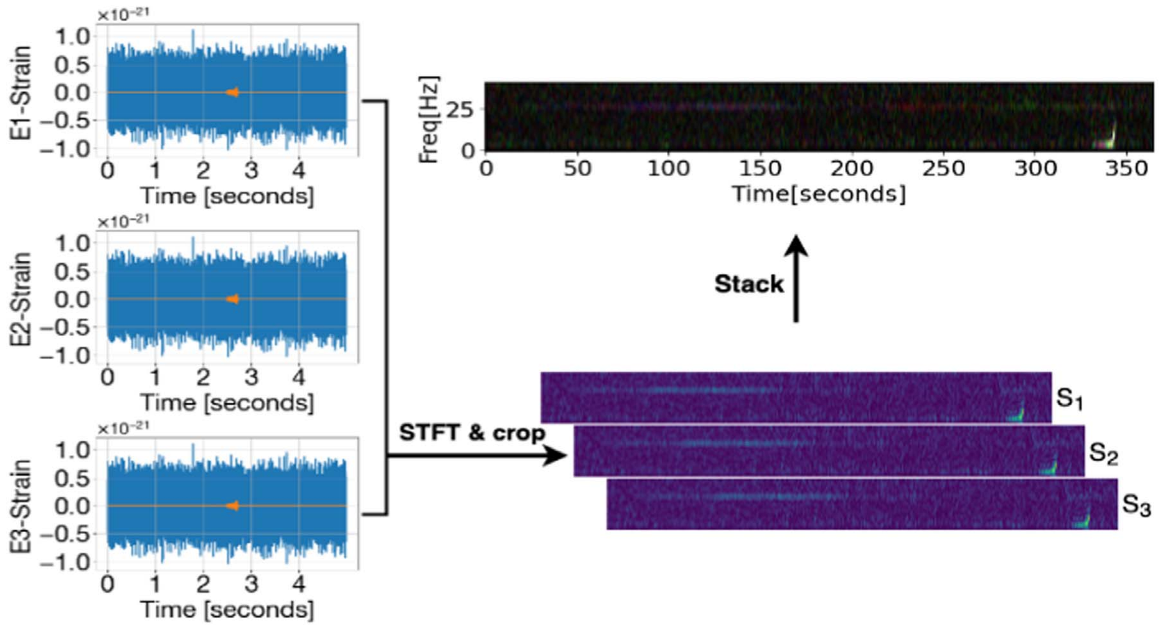


Figure 4. Generation and processing of the spectrograms from TSDCD.

In the ResNet architecture, Y is equivalent to $2\mathcal{F}(X) - X$, reflecting the residual connection $Y = f(x) + f(x) - x$, where $f(x)$ represents the output of a series of trainable layers.

Due to this connection, the original output will be superimposed before sending it to a future layer; hence, this mitigates and prevents any gradient explosion or disappearance when using backpropagation—for details about backpropagation, please see P. Munro (2010)—and allows for the training of very deep networks. More details about ResNet can be found in the Appendix. As previously mentioned in WTM1, ResNet is available in various architectures, distinguished by the number of layers. This includes ResNet-50 and ResNet-101, which comprise 50 and 101 convolutional layers, respectively. We adopt the ResNet-101 architecture (hereafter referred to as ResNet) in this work, as was previously done in WTM1.

3.1. ResNet Training

We keep the same settings for training the ResNet model as in our previous work (WTM1). However, the data set size has increased fivefold in TSDCD compared to the SSDD we used before. This expansion is a result of incorporating five different F_{low} values. In ML, traditionally the data are split into three different sets, for training, testing, and evaluating the model (K. P. Murphy 2012; V. C. Raykar & A. Saha 2015; R. Medar et al. 2017; Q. H. Nguyen et al. 2021; J. Tan et al. 2021).

Table 1 displays the total number of TSDCD samples utilized for the ResNet model training, testing, and validation. All F_{low} values and MSNR ranges are evenly represented in the randomly chosen testing and validation sources, to ensure an unbiased evaluation of the model's performance. The three-subdetector combined spectrograms consist of three layers, where each layer belongs to one subdetector. Hence, the input layer of ResNet was adjusted to a shape size of $365 \times 42 \times 3$, which describes the width, height, and number of layers of the input image, respectively. A batch size of 256, a learning rate of 0.0001, and an RMSprop optimizer (D. Vijendra Babu et al. 2020; D. Xu et al. 2021; R. Elshamy et al. 2023) were used for training the ResNet model for 200 epochs.

Table 1
Total Number of TSDCD Spectrograms: Injected and Only-noise Spectrograms for Training, Testing, and Validation

Type	Number of Sample	Train	Test	Val
Injected	125,000	85,000	20,000	20,000
Only noise	125,000	85,000	20,000	20,000
Total	250,000	170,000	40,000	40,000

Table 2
Classification Report Showing the Precision, Recall, and F1-score Metrics on the Testing Data Set

Type	Precision	Recall	F1-score	Support
Injected	0.993	0.818	0.897	20,000
Only noise	0.845	0.994	0.914	20,000
Avg / total	0.919	0.906	0.905	40,000

4. TSDCD Evaluation

To evaluate the performance of ResNet on the TSDCD, similar to WTM1, we generated a classification report displaying precision, recall, and F1-score values (C. M. Bishop 2006) for our testing data set, as shown in Table 2. Recall, also known as sensitivity (G.-J. Wang et al. 2020), represents the quantity of correctly classified and misclassified sources, where ResNet achieved 80% and 99% on injected and only-noise spectrograms. This result demonstrates a notable trend, indicating that only-noise spectrograms are highly unlikely to be misclassified as injected ones across all the different F_{low} and MSNR ranges. Precision, assessing the quality of classification, achieved an average of 90% with ResNet. The overall performance of ResNet, measured by the F1-score—a harmonic mean of precision and recall—also attained an average of 90%.

To compare the TSDCD results with those obtained in our prior investigations in WTM1, we generated a TSDCD with an F_{low} of 30 Hz only with all five MSNR ranges. This data set

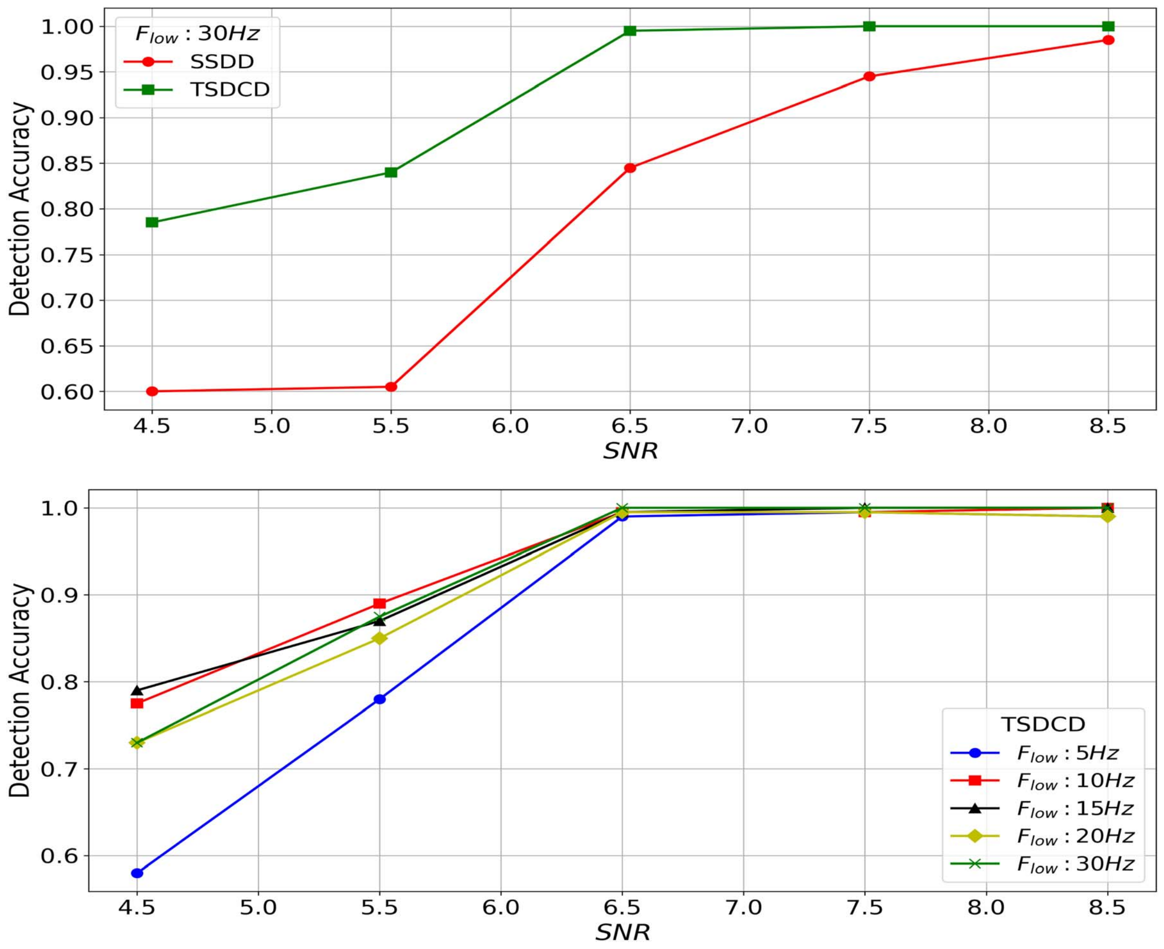


Figure 5. MSNR vs. detection accuracy, Top: SSDD (from WTM1) vs. TSDCD for sources with F_{low} of 30 Hz. Bottom: TSDCD for sources with an F_{low} of 5 Hz, 10 Hz, 15 Hz, 20 Hz, and 30 Hz.

consists of 4000 sources, created using identical parameters to the testing data set in WTM1. In Figure 5, the top panel displays the accuracy of the detection acquired for the TSDCD, alongside the accuracy obtained by the SSDD in WTM1. The plot clearly demonstrates a significant improvement across all five MSNR ranges, particularly at lower MSNR values. The detection accuracy has improved from 60%, 60.5%, 84.5%, 94.5%, and 98.5% to 78.5%, 84%, 99.5%, 100%, and 100% for sources with MSNRs of 4–5, 5–6, 6–7, 7–8, and greater than 8, respectively. This shows a significant improvement of approximately 30.83%, 39.00%, 17.75%, 5.81%, and 1.52% for sources with MSNRs of 4–5, 5–6, 6–7, 7–8, and greater than 8, respectively. It is essential to note that achieving 100% accuracy for sources with MSNRs greater than 8 means successful detections of all sources from our samples. However, it does not guarantee that all sources within this specific range will be detected at all times. This applies to all other MSNR ranges.

The bottom panel in Figure 5 displays the detection accuracy from our testing data set, considering each F_{low} individually. The results indicate a generally similar performance across all frequencies and MSNR ranges, with relatively inferior performance observed for lower MSNR values (4–6) at F_{low} of 5 Hz, as expected.

In a quantitative evaluation, we compared the number of misclassified injected and only-noise spectrograms using confusion matrices (S. V. Stehman 1997). We use 200 randomly selected samples (100 samples from each class) from each MSNR range, with an F_{low} of 30 Hz, obtained from both SSDD and TSDCD, to generate the confusion matrices. As shown in Figure 6, a significant decrease can be observed in the number of misclassified injected spectrograms, from 66, 54, 8, and 1 to 50, 25, 0, and 0 for MSNRs of 4–5, 5–6, 6–7, and 7–8, respectively, on TSDCD, in contrast to the results from SSDD. For only-noise spectrograms, in TSDCD, only two samples were misclassified as injected, in contrast to the 19 total misclassifications observed in SSDD. In Figure 7, the confusion matrices for sources with MSNRs greater than 8 illustrate cases where no only-noise spectrogram was identified as injected in TSDCD, in contrast to three in SSDD.

5. Evaluation on ET’s Continuous Data

As previously done in WTM1, three subdetectors' combined mock data were generated to assess the efficiency of utilizing TSDCD for near-real-time detection and to compare it with previously obtained results from SSDD. 5 hr and and 50 s of time-series data, with no overlapping sources and only BBH injections, encompassing all MSNR ranges, were generated for each F_{low} (5 Hz, 10 Hz, 15 Hz, 20 Hz, and 30 Hz) for each detector. At every 5 s interval in this time-series data, a signal is injected at a random position, resulting in a total of 3610 injected sources, with each MSNR range containing 722 sources. The total duration of the data for each detector is 25 hr,

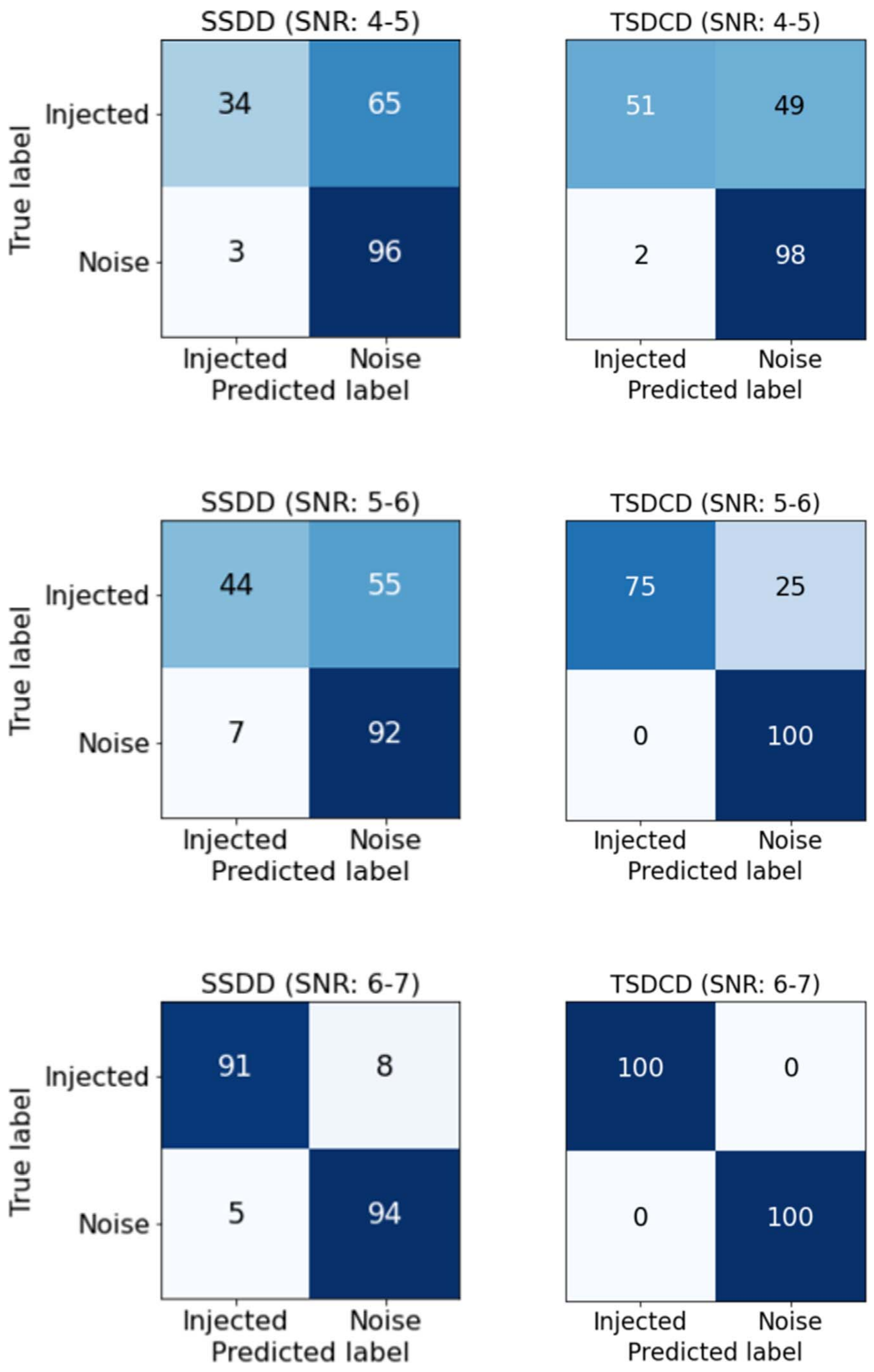


Figure 6. Confusion matrix comparison for MSNRs in the ranges 4–7 between SSDD and TSDCD, with an F_{low} of 30 Hz.

4 minutes, and 16 s. To compare with our previously obtained results, we utilize the data with an F_{low} of 30 Hz.

To perform detection on this time-series data, we follow four main steps, as illustrated in Figure 8: (1) slide a 5 s window simultaneously across the data sets of the three detectors; (2) generate a spectrogram for each window; (3) crop and stack the spectrograms; and (4) feed the stacked spectrogram into the trained ResNet model for prediction.

The false-positive rate (FPR; D. S. Burke et al. 1988) quantifies how often the model incorrectly classifies only-noise samples as injected ones, defined as follows:

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}, \tag{21}$$

where FP is the number of false positives and TN is the number of true negatives.

The FPR was computed for each F_{low} for the entire mock data. Figure 9 depicts FPR as a function of MSNR, where in the top panel, TSDCD results are compared to SSDD results from WTM1. The general trend is consistent with the previous results in WTM1, where FPR decreases as MSNR increases. As shown in the figure, FPR values have significantly improved, especially for sources with lower MSNRs, transitioning from 0.712, 0.535, 0.050, 0.014, and 0.001 to 0.526, 0.296, 0.006, 0.004, and 0.0 for the MSNR ranges 4–5, 5–6, 6–7, 7–8, and >8, respectively. In the bottom panel, FPR values are depicted for each F_{low}. The overall performance is nearly identical across different F_{low} settings, except for the

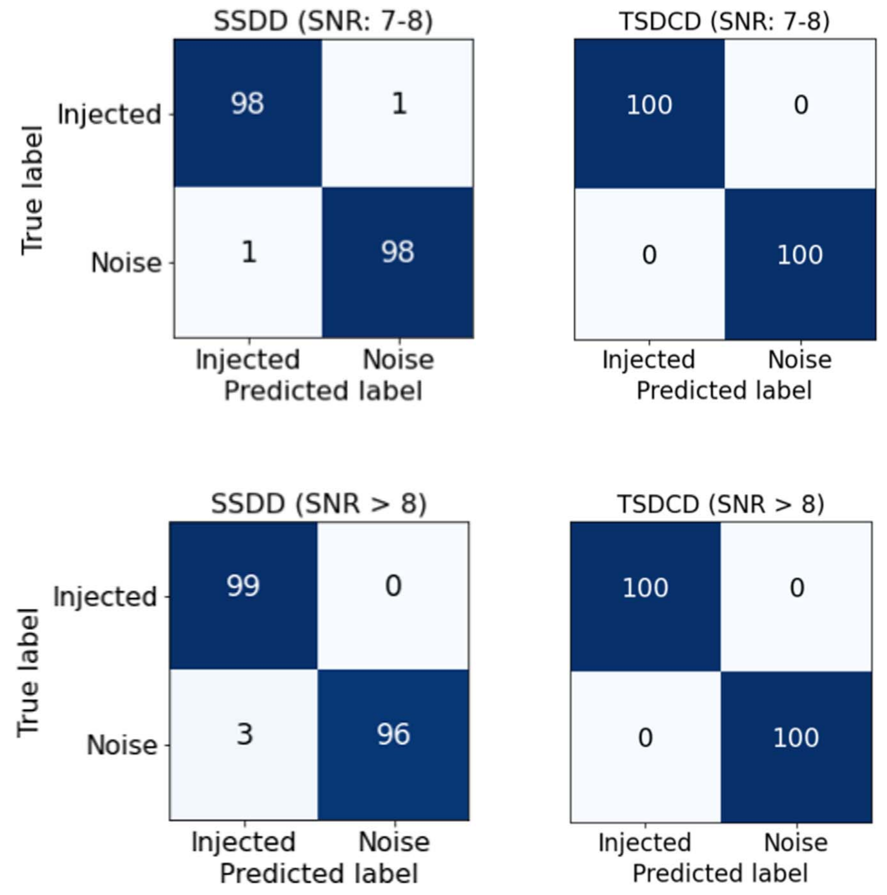


Figure 7. Confusion matrix comparison for MSNRs 7–8 and greater between SSDD and TSDCD, with an F_{low} of 30 Hz.

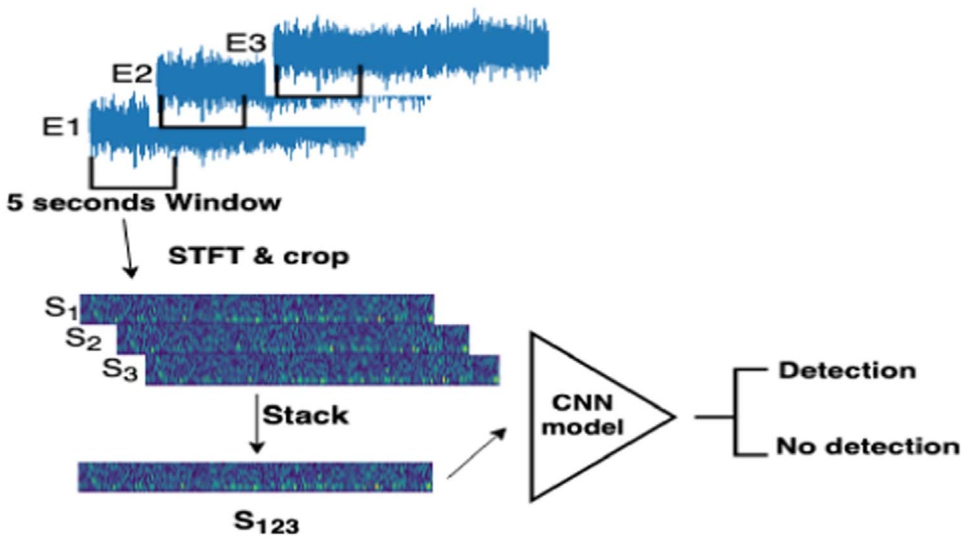


Figure 8. Inferencing on ET's continuous data.

case of F_{low} at 5 Hz. This discrepancy aligns with the accuracy results presented earlier.

In terms of the MSNR, chirp mass M, and luminosity distance D_L for undetected sources, Figures 10, 11, and 12 compare the total number of undetected sources between SSDD and TSDCD. Additionally, TSDCD is examined individually for all five different F_{low} values. The right panel in Figure 10 demonstrates a significant improvement across all MSNR ranges, particularly noticeable at relatively higher MSNRs 5–8 and greater, and a noticeable decrease within MSNRs 4–5, where TSDCD exhibits a reduction of 100 undetected sources compared to SSDD. The left panel illustrates variations in the total number of undetected sources across our five F_{low} settings, primarily within the MSNR range of 4–5. Specifically, a total of 548, 531, 549, 562, and 594 undetected sources at F_{low} settings of 5 Hz, 10 Hz, 15 Hz, 20 Hz, and 30 Hz are observed within the MSNR range 4–5, respectively.

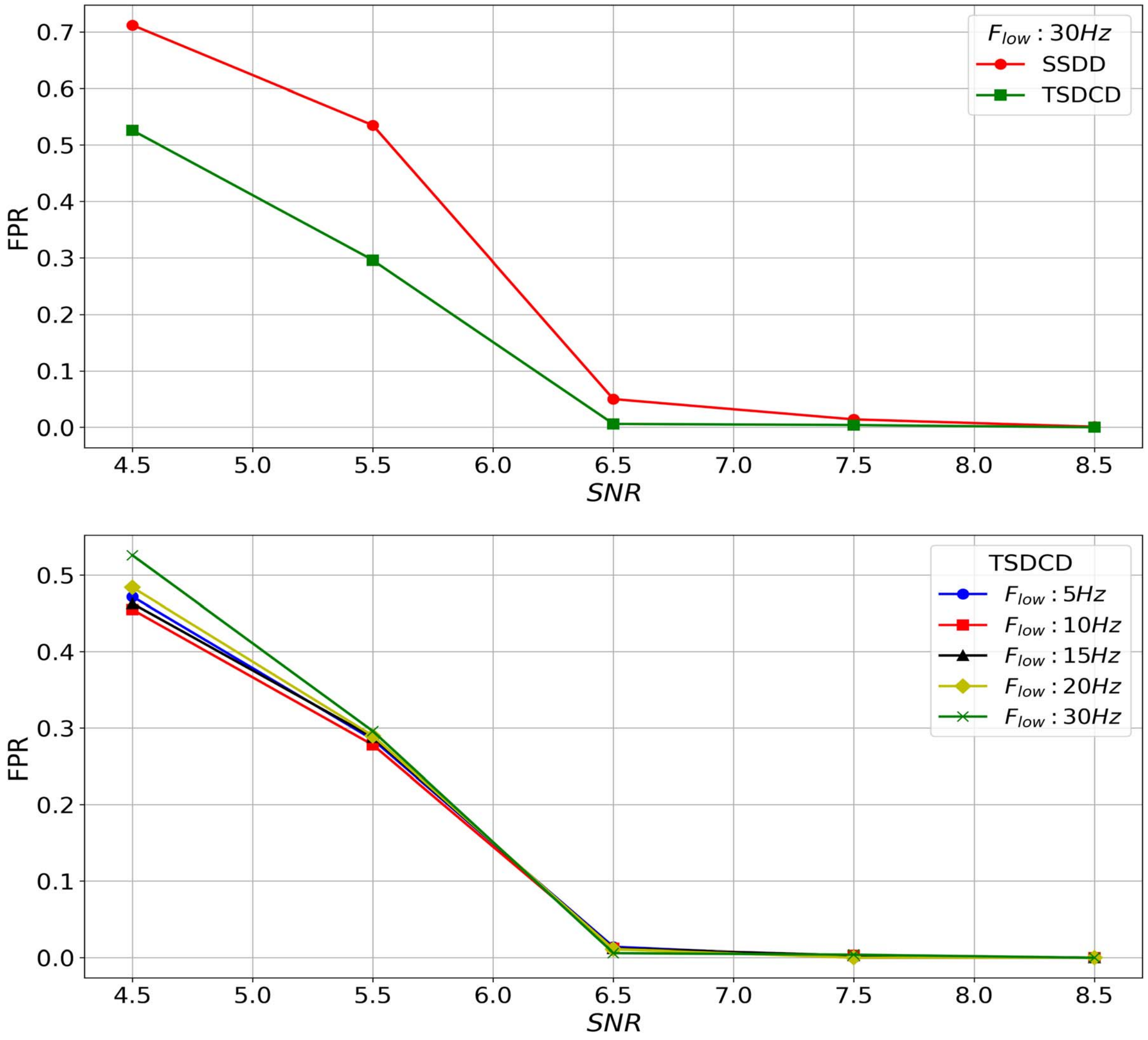


Figure 9. Top: a comparison of FPR values between SSDD (from WTM1) and TSDCD for sources with an $F_{\rm low}$ set at 30 Hz. Bottom: FPR values exclusively from TSDCD for sources with an $F_{\rm low}$ at 5 Hz, 10 Hz, 15 Hz, 20 Hz, and 30 Hz.

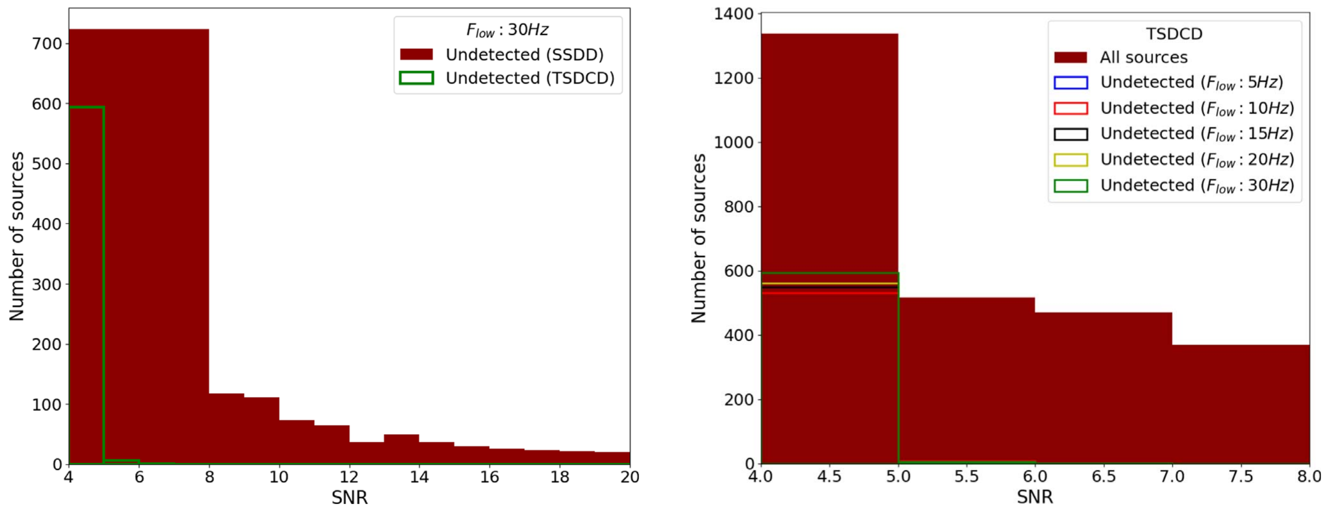


Figure 10. Right: MSNRs of all undetected BBH sources from SSDD (from WTM1) and TSDCD. Left: MSNRs of undetected sources exclusively from TSDCD for sources with an $F_{\rm low}$ at 5 Hz, 10 Hz, 15 Hz, 20 Hz, and 30 Hz.

Similarly, in terms of the chirp mass M, the right panel in Figure 11 demonstrates a significant decrease in the total number of undetected sources in TSDCD, particularly among sources with chirp masses M in the range of 15–20 $M_\odot$. In TSDCD alone, the count of undetected sources of different $F_{\rm low}$ varies among all chirp-mass M values. For an $F_{\rm low}$ at 10 Hz, the minimum counts

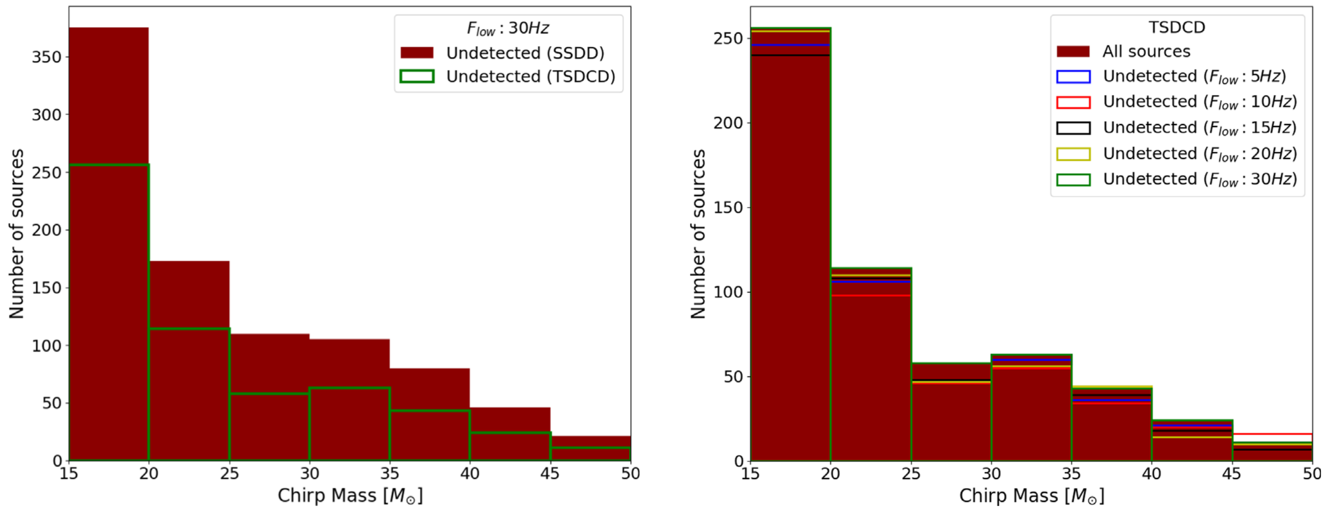


Figure 11. Right: chirp mass M of all undetected BBH sources from SSDD (from WTM1) and TSDCD. Left: chirp mass M of undetected sources exclusively from TSDCD for sources with an $F_{\rm low}$ at 5 Hz, 10 Hz, 15 Hz, 20 Hz, and 30 Hz.

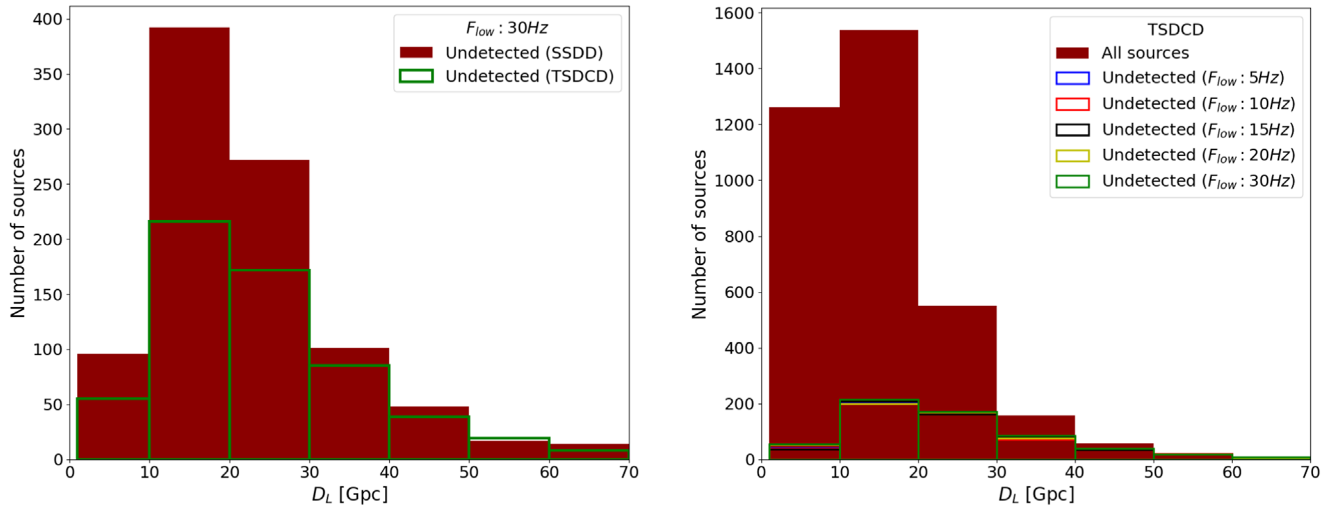


Figure 12. Right: D_L of all undetected BBH sources from SSDD (from WTM1) and TSDCD limited to 70 Gpc. Left: D_L of undetected sources exclusively from TSDCD for sources with an $F_{\rm low}$ at 5 Hz, 10 Hz, 15 Hz, 20 Hz, and 30 Hz.

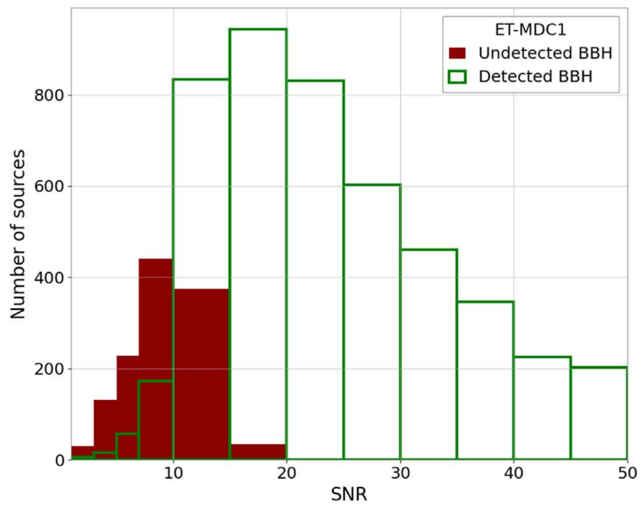
observed are 240, 98, 46, 55, and 34 in the chirp-mass M ranges of 15–20, 20–25, 25–30, 30–35, and 35–40 $M_\odot$, respectively. For the chirp-mass M ranges 40–45 and 45–50, the minimum counts observed are 14 and 7 for an $F_{\rm low}$ of 20 Hz and 15 Hz, respectively. Regarding the D_L, the right panel of Figure 12 shows a substantial reduction in the count of undetected sources in TSDCD, especially for sources with D_L in the range of 10–30 Gpc. In the left panel, no significant variation is observed across different $F_{\rm low}$ settings. In summary, both SSDD and TSDCD exhibit a higher detection rate for sources with higher MSNR, chirp mass, and shorter D_L, as indicated by the smaller FPR values. Notably, TSDCD demonstrates a significant improvement over SSDD. It is worth noting that the same source may exhibit different MSNR values on each subdetector, yet TSDCD consistently outperforms SSDD across these variations.

For a more qualitative check, Table 3 shows the maximum D_L ($D_{L_{\rm Max}}$) in gigaparsecs, the minimum MSNR (MSNR$_{\rm Min}$), and the minimum chirp mass $M_{\rm Min}$ in $M_\odot$, along with the counts of detected sources for all $F_{\rm low}$ out of the total 3610 sources. The ResNet model successfully detected sources at a distance of 86.601 Gpc, with an average MSNR of 3.9 (the averaged MSNR over the three detectors) and a chirp mass of 13.632 at 5 Hz. At 10 Hz, 15 Hz, 20 Hz, and 30 Hz, ResNet successfully identified sources with average MSNR values of 4.031, 4.033, 4.023, and 4.001, along with chirp masses of 20.320, 15.630, 14.201, and 17.532, respectively.

Table 3
Maximum D_L ($D_{L_{\rm Max}}$) in Gigaparsecs, Minimum MSNR (MSNR$_{\rm Min}$), Minimum Chirp Mass ($M_{\rm Min}$) in $M_\odot$, and Counts of Detected Sources for All $F_{\rm low}$ out of 3610 Total Sources

$F_{\rm low}$	$D_{L_{\rm Max}}$	MSNR$_{\rm Min}$	$M_{\rm Min}$	Total
5 Hz	86.601	3.900	13.632	3053
10 Hz	90.255	4.031	20.320	3069
15 Hz	67.553	4.033	15.630	3056
20 Hz	67.553	4.023	14.201	3042
30 Hz	67.553	4.001	17.532	3009

In terms of computational efficiency, we concurrently processed a total of 25 hr, four minutes, and 16 s of data from



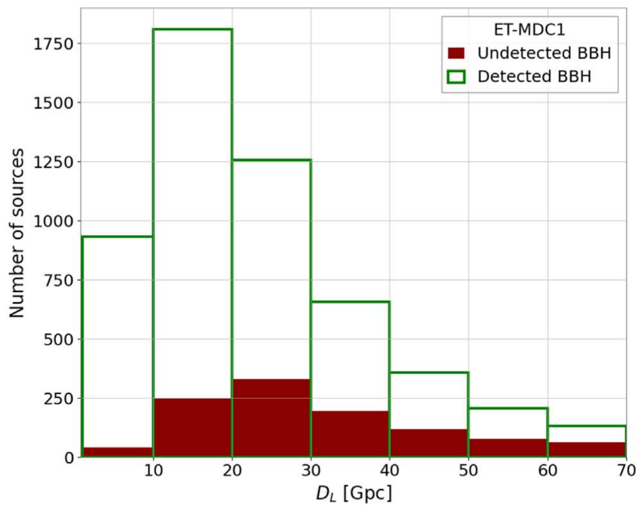


Figure 13. Optimal SNR (left) and D_L (right) of detected and undetected BBH sources from ET-MDC1.

each detector in parallel. Utilizing the same configuration as in WTM1 (a Core i7 MacBook Pro with 16 GB memory, 2667 MHz DDR4, and a 2.6 GHz processor), the entire data scan was completed in 15 minutes, averaging 1.9 minutes for each hour. The processing time has been significantly reduced compared with that achieved on SSDD of 4.7 minutes for each hour. The key difference lies in simultaneously reading data from all three subdetectors. Hence, utilizing TSDCD is suitable for near-real-time detection and could be further enhanced with a more powerful setup.

6. ET-MDC1: ET Mock Data Challenge

A mock data challenge for ET by T. Regimbau et al. (2012) was first released in 2012 and updated in early 2024 to contain 1.3 terabytes of data for ET only, in addition to Cosmic Explorer. The recent release, called ET-MDC1, contains a continuous GW signal plus noise and noise only of about one month (30.8 days), split into 1300 segments of 2048 s, sampled at 8192 Hz. The use of the ET V-shaped detectors E1, E2, E3, the GW signal, and noise (colored Gaussian noise with no noise artifacts in this version) were simulated, in addition to the null stream E0. The data contain the parameters of the injected sources, such as the optimal SNR, component masses (M1 and M2), and D_L. The GW signal contains 59,540 binary neutron star (BNS), 6578 BBH, and 1977black hole-neutron star (BHNS) events, with optimal SNR ranges between 0.13 and 586.12. For BNS, BBH, and BHNS, IMRPhenomPNRv2 (which contains tidal effects), IMRPhenomXPHM, and IMRPhenomXPHM approximants were used, respectively. In ET-MDC1, BBH has a wide range of optimal SNRs, spanning from 0.8 to 586. The component masses (M1 and M2) vary significantly, with minimum values of 7 and 6 $M_{\odot}$, respectively, and maximum values reaching 793 and 617 $M_{\odot}$. The D_L of these BBH systems ranges from 0.5 Gpc to 154.37 Gpc.

The great challenge in evaluating our ResNet model on ET-MDC1 is that it was only trained on BBH mergers and with no overlapping sources. In addition to that, the waveforms were not continuous in the training data set, and BNS has a long inspiral phase that can last for days, which the model has not seen during training. We evaluated ResNet on the entire ET-MDC1 E1, E2, and E3 injected data (signal plus noise) to assess its performance. Additionally, we tested the model on one week of noise-only data and null-stream data to check the FPR with three different thresholds of 0.5, 0.3, and 0.1 (equivalent to 50%, 70%, and 90% confidence, respectively). As illustrated in Table 4, when accepting detections with at least 90% confidence, no FPs were recorded. Only one false detection was observed using the 0.3 threshold and 10 when using 0.5.

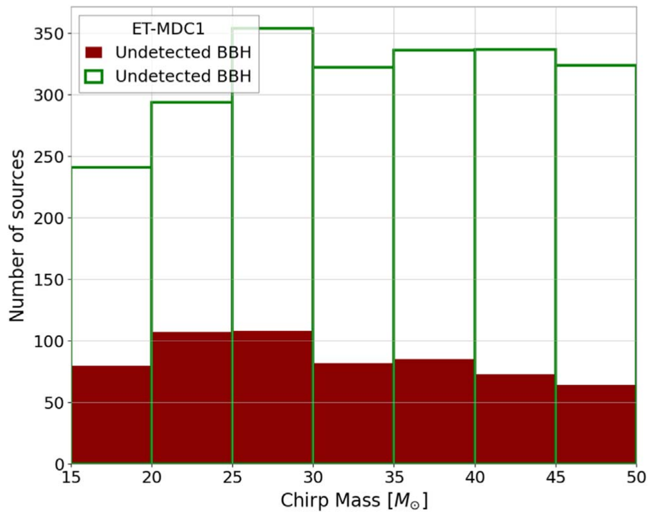


Figure 14. Chirp mass M of detected and undetected BBH sources from ET-MDC1.

Table 4
FPR from One Week of Noise and Null Data Only

Threshold	Null	Noise
0.5	0	10
0.3	0	1
0.1	0	0

As shown in Figures 13 and 14, ResNet successfully detected 5,566 BBH mergers out of a total of 6578. 75% of these sources have an average optimal SNR of 38.3, an average D_L of 32 Gpc, and an average chirp mass of 96 $M_{\odot}$. The detected BBH sources have a minimum and maximum D_L of 0.5 Gpc and 148.95 Gpc, respectively. The minimum and

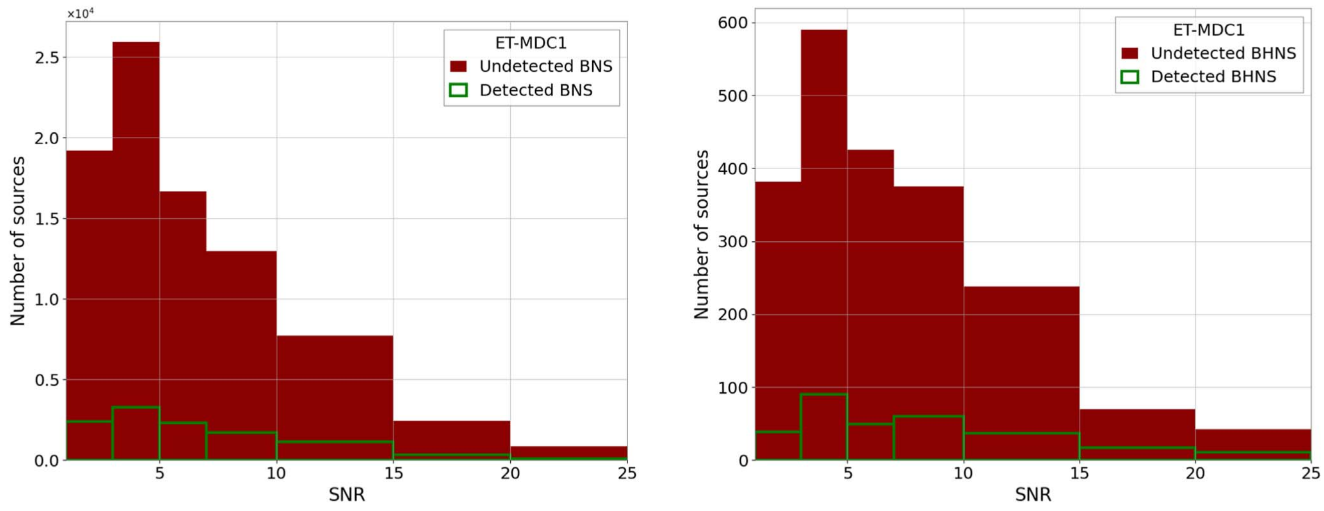


Figure 15. Optimal SNR of detected and undetected BNS (right) and BHNS (left) sources from ET-MDC1.

maximum optimal SNR are 1.2 and 586, respectively, and the minimum and maximum chirp mass are 6 and 596 $M_\odot$, respectively.

For undetected BBH sources, the optimal SNR ranges from 0.8 to 51.7, with 75% having an average optimal SNR of 10.7. The D_L ranges between 2.4 Gpc and 154.4 Gpc. The minimum and maximum chirp mass of the undetected sources are 7.8 $M_\odot$ and 484.6 $M_\odot$, respectively.

These results demonstrate the great performance of our model. The performance can be significantly improved by considering parameters outside the current range for BBH masses and distances. Additionally, incorporating samples of overlapping sources into the training data set will positively impact the results.

Although ResNet was not trained on BNS and BHNS mergers, the model was able to detect 11,477 BNS mergers and 323 BHNS mergers. The optimal SNRs of the detected and undetected BNS and BHNS sources are shown in the right and left panels of Figure 15. BNS and BHNS detected sources have optimal SNR ranges from 0.2 to 383 and from 1 to 50, respectively.

The observation that the ResNet model is capable of detecting BNS and BHNS mergers, despite being trained only on BBH mergers, is due to the fact that inspiral GW signals share a common functional structure, consisting of three phases: inspiral, merger, and ringdown. However, the exact form of the signal depends on several physical parameters, such as the masses of the binary objects, their spins, the orbital eccentricity, and the distance to the source. When using matched filtering, a template generated with a specific set of parameters may perform poorly in detecting a signal with a different set of parameters, leading to a reduction in the cross-correlation between the template and the observed signal (D. Macleod et al. 2016). In contrast to matched filtering, DL models have the capacity to generalize beyond their training data. This characteristic allows convolutional neural networks (CNNs) to capture shared features across different types of compact binary systems, including BBH, BNS, and BHNS. All of these systems produce signals that include inspiral, merger, and ringdown phases, making the generalizations ideal candidates for CNN-based classification. When training our model, we specifically focused on the merger phase, which is common across all compact binary systems. Our model was trained on thousands of merger waveforms with component masses ranging from 10 to 56 solar masses. This training allowed the network to learn features that generalize well across systems, irrespective of the specific mass or spin configurations. However, the performance could be significantly improved if this model should be trained on diverse and well-representative samples of all three different systems, in addition to overlapping and nonoverlapping sources.

7. PyMerger

PyMerger (W. Alhassan et al. 2024) is a Python tool for detecting BBH mergers from ET, built based on the trained ResNet model described above. PyMerger is our first step toward developing a full AI-based pipeline for detection and, soon, parameter estimation of BBH from ET. We have developed and shared the first version to openly source the trained model and allow the results' reproducibility.

The current version handles only the GW frame file format (.gwf), with the intention to support HDF5 in the upcoming version. As shown in Figure 16, the software takes six arguments: (1) a directory containing the input .gwf files; (2) a directory to store the results; (3) the sampling rate—the sampling rate of the input data (either 8192 or 4096), with the default being 8192; (4) the number of data segments to be processed for each detector (i.e., the number of .gwf files to be processed for each subdetector), with the default being 1; (5) a list of the three channel names to be processed, the default being [“E1:STRAIN,” “E2:STRAIN,” “E3:STRAIN”]; and (6) the threshold value for merger detection, the default being 0.1 (accepting detections with at least 90% confidence). The software processes each file by sliding a window (the window size is based on the sampling rate) to scan the data and look for BBH mergers. The output is a text file (.txt) with three columns: (1) the starting GPS time of the sliding window; (2) the end GPS time of the sliding window; and (3) the detection probability of all the detected mergers.

Regarding computing efficiency, PyMerger scans 1 hr of TSDCD in 1.9 minutes. This can be significantly improved, as DL models can be substantially accelerated based on the hardware used. For instance, using the NVIDIA TensorRT–optimized AI ensemble, P. Chaturvedi et al. (2022) processed 1 month of advanced LIGO data in 50 s. Future work will focus

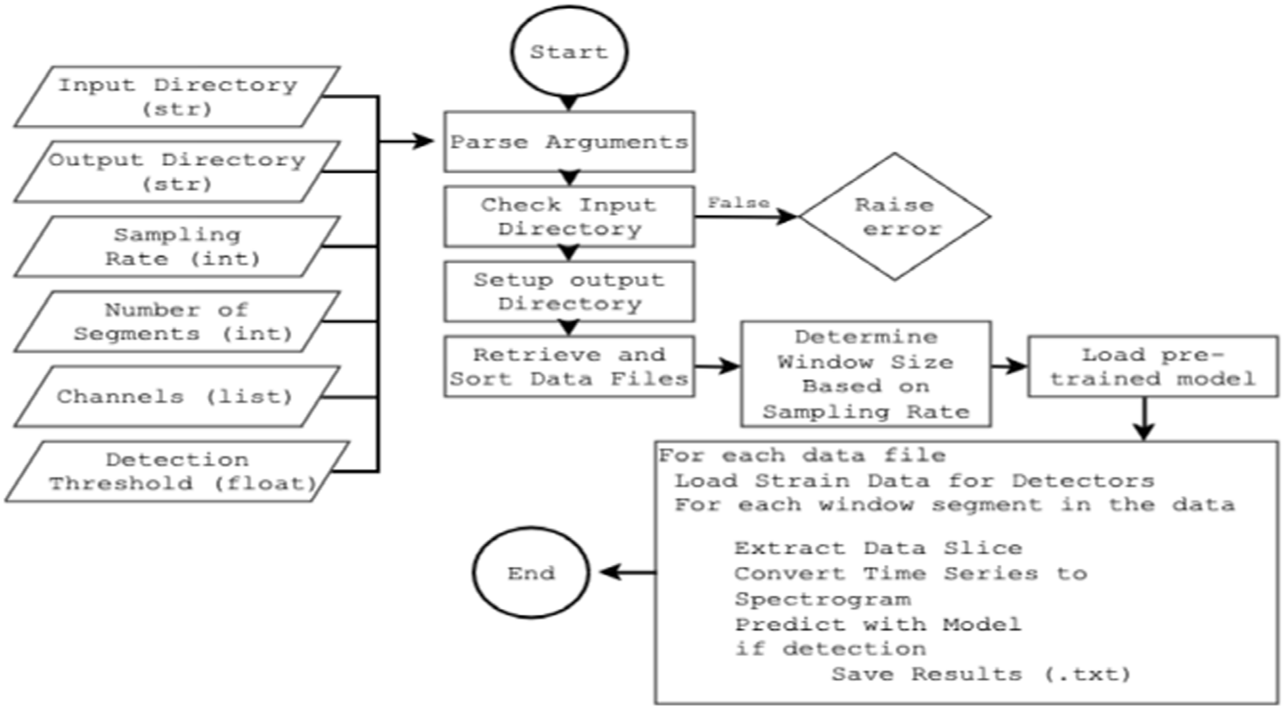


Figure 16. A flowchart outlines the high-level process of PyMerger, starting from argument parsing to data processing, model prediction, and result logging.

on incorporating BNS and BHNS events along with glitchy noise.

The software is publicly available at https://github.com/wathela/PyMerger. Researchers in the field are encouraged to make use of the software and kindly provide us with their feedback, comments, and suggestions.

8. Conclusion

In this study, we have conducted a comparative analysis of the efficiency in detecting BBH sources using data from all three subdetectors of ET simultaneously versus using data from a single subdetector. The comparison has specifically focused on cases with an F_{low} set to 30 Hz. Furthermore, we have explored TSDCD's performance across five different F_{low} settings (5 Hz, 10 Hz, 15 Hz, 20 Hz, and 30 Hz), employing the same MSNR ranges as in our previous study (4–5, 5–6, 6–7, 7–8, and >8). Utilizing the ResNet model, which exhibited superior performance in our previous study, TSDCD demonstrated a significant enhancement in detection accuracy compared to SSDD. The accuracy improved from 60%, 60.5%, 84.5%, 94.5%, and 98.5% to 78.5%, 84%, 99.5%, 100%, and 100% for sources with MSNR ranges of 4–5, 5–6, 6–7, 7–8, and >8, respectively. The results indicate a substantial accuracy improvement for lower MSNR ranges: 4–5, 5–6, and 6–7, with gains of 18.5%, 24.5%, and 13%, respectively. Additionally, there is a significant improvement of 5.5% and 1.5% for higher MSNR ranges: 7–8 and >8.

For more rigorous evaluation, the ResNet model was evaluated on the ET-MDC1 data set, where the model demonstrated strong performance in detecting BBH mergers, identifying 5566 out of 6578 BBH events, with optimal SNRs starting from 1.2 and a minimum and maximum D_L of 0.5 Gpc and 148.95 Gpc, respectively. Despite being trained only on BBH mergers without overlapping sources, the model achieved high detection rates. However, to further enhance its performance, it is essential to include a broader range of parameters for BBH masses and distances and to incorporate overlapping sources in the training data. Notably, even though the model was not trained on BNS and BHNS mergers, it successfully detected 11,477 BNS and 323 BHNS mergers, with optimal SNRs starting from 0.2 to 1, respectively. This indicates the model's potential for broader applicability. Future work will focus on improving detection rates by including both overlapping and nonoverlapping BNS and BHNS events in the training process. Built on ResNet, PyMerger is a Python tool designed for detecting BBH mergers from ET data. PyMerger operates without the need for data whitening or bandpassing and can scan 1 hr of TSDCD in 1.9 minutes on an average laptop. The current version supports GW frame format (.gwf) files.

Acknowledgments

This research was funded in part by the Preludium 2023 grants (UMO-2023/49/N/ST9/01287), Maestro (2018/30/A/ST9/00050) and Opus (2023/49/B/ST9/02777) from the National Science Centre, Poland. Additionally, this work has been supported in part by the European Union's Horizon 2020 research and innovation program under grant agreement No. 952480 (DarkWave project) and by the International Research Agenda Programme AstroCeNT (MAB/2018/7), funded by the Foundation for Polish Science (FNP) from the European Regional Development Fund. The authors thank Professor Piotr Gawron for helpful discussions and reviews.

Software: LALSuite (LIGO Scientific Collaboration et al. 2018), pyCBC (A. Nitz et al. 2023).

Appendix
ResNet: Deep Residual Neural Network

CNNs (K. Fukushima 1980) are DL models designed to process data with a 2D structure, like images or spectrograms,

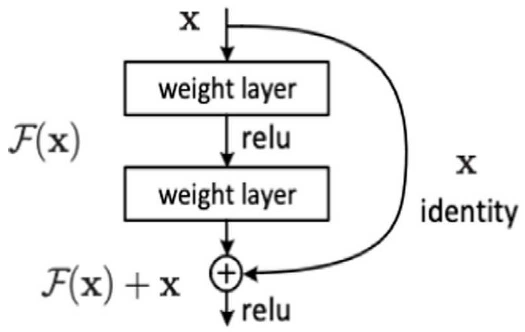

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 17. Residual learning: a residual block.

for tasks such as pattern recognition and object detection (K. O'Shea & R. Nash 2015). A CNN typically consists of convolutional layers that extract features using filters applied through convolution operations (I. Goodfellow et al. 2016), followed by activation functions (e.g., ReLU, *tanh*, and *softmax*; J. S. Bridle 1990; S. Haykin 1998). Pooling layers reduce the spatial dimensions by aggregating local regions, and fully connected layers act as classifiers, similar to traditional artificial neural networks (A. F. Agarap 2017). In essence, CNNs perform nonlinear mappings from input images to class scores. CNN models suffer from the vanishing gradient problem (also called the degradation problem), where increasing the depth of the network leads to a higher training error. When deep neural networks are trained, an increase in network depth often leads to a saturation and subsequent degradation in performance. The degradation is not caused by overfitting but by difficulties in optimization. This is counterintuitive, because deeper models are expected to, theoretically, learn better representations.

ResNets were introduced by K. He et al. (2016) as a solution to the degradation problem in DL models. ResNets solve this problem by introducing shortcut connections that bypass certain layers, allowing for easier optimization of deep models. ResNet introduces the concept of residual learning. The core component of ResNet is the residual block, shown in Figure 17, where the network learns a residual function instead of directly learning the desired mapping. Let X represent the input to a residual block and $\mathcal{F}(X)$ be the desired transformation. In a traditional network, the output of a layer would be $H(X) = \mathcal{F}(X)$. However, in ResNet, the network learns a residual function $\mathcal{R}(X)$ defined as:

$$\mathcal{R}(X) = \mathcal{F}(X) - X, \tag{A1}$$

which can be rearranged as

$$\mathcal{F}(X) = \mathcal{R}(X) + X. \tag{A2}$$

This residual function $\mathcal{R}(X)$ is learned by a series of layers, and the output of the residual block is

$$Y = X + \mathcal{R}(X). \tag{A3}$$

The addition of the input X to the learned residual mapping allows the network to bypass transformations, helping to preserve gradients during backpropagation.

By stacking multiple residual blocks, ResNet enables the training of very deep networks without the performance degradation seen in traditional deep architectures. The general form for the output of the lth residual block in a deep ResNet can be written as

$$Y^{[l+1]} = Y^{[l]} + \mathcal{F}(Y^{[l]}, W^{[l]}), \tag{A4}$$

where $Y^{[l]}$ is the input to the lth block, $\mathcal{F}(Y^{[l]}, W^{[l]})$ is the residual function learned by the lth block, and $W^{[l]}$ are the trainable weights of the block. For deeper networks like ResNet-50 or ResNet-101, the bottleneck design is used to reduce the number of parameters. In this design, three layers are used instead of two. These layers use a combination of three convolutions: (1) a 1×1 convolution to reduce the dimensionality; (2) a 3×3 convolution for feature extraction; and (3) a final 1×1 convolution to restore the original dimensionality. The residual function for a bottleneck block is

$$\mathcal{R}(X) = W_3 \cdot (W_2 \cdot (W_1 \cdot X)), \tag{A5}$$

where W_1, W_2, and W_3 are the weights for the three convolutional layers. The bottleneck design reduces computational complexity, by reducing the dimensionality of the feature maps before applying expensive convolutions. When the dimensions of the input X and the output $\mathcal{F}(X)$ do not match, ResNet employs a projection shortcut to match dimensions using a linear projection:

$$Y = W_s \cdot X + \mathcal{F}(X), \tag{A6}$$

where W_s is the projection matrix (typically a 1×1 convolution) used to adjust the dimensions of X before adding it to the residual function. For the training, ResNet uses standard backpropagation (P. Munro 2010), with the total loss $\mathcal{L}$ minimized using stochastic gradient descent. For classification tasks, the loss function is typically cross-entropy:

$$\mathcal{L} = -\sum_i y_i \log(\hat{y}_i), \tag{A7}$$

where y_i is the true label and $\hat{y}_i$ is the predicted probability for class i. The introduction of residual connections significantly improved the performance of deep neural networks. ResNet achieved state-of-the-art results in the ImageNet competition (O. Russakovsky et al. 2015) and has been widely adopted in various applications, including for image classification, object detection, and segmentation.

ORCID iDs

Wathela Alhassan https://orcid.org/0000-0002-8266-3005
T. Bulik https://orcid.org/0000-0003-2045-4803

References

Acernese, F., Agathos, M., Agatsuma, K., et al. 2015, CQGra, 32, 024001
Adachi, M., & Kasai, M. 2012, PThPh, 127, 145
Agarap, A. F. 2017, arXiv:1712.03541
Akutsu, T., Ando, M., Arai, K., et al. 2020, PTEP, 2021, 05A101
Alhassan, W., Bulik, T., & Suchenek, M. 2023, MNRAS, 519, 3843
Alhassan, W., Bulik, T., & Suchenek, M. 2024, PyMerger: Detecting Binary Black Hole Mergers from Einstein Telescope Using Deep Learning, v1.0, Zenodo, doi:10.5281/zenodo.13931800
Aso, Y., Michimura, Y., Somiya, K., et al. 2013, PhRvD, 88, 043007
Belczynski, K., Bulik, T., & Kalogera, V. 2002a, ApJL, 571, L147
Belczynski, K., Bulik, T., & Rudak, B. 2002b, ApJ, 571, 394
Belczynski, K., Kalogera, V., & Bulik, T. 2002c, ApJ, 572, 407
Belczynski, K., Kalogera, V., Rasio, F. A., et al. 2008, ApJS, 174, 223
Belczynski, K., Klencki, J., Fields, C. E., et al. 2020, A&A, 636, A104
Bishop, C. M. 2006, Pattern Recognition and Machine Learning (Information Science and Statistics) (Berlin: Springer)
Branchesi, M., Maggiore, M., Alonso, D., et al. 2023, JCAP, 2023, 068
Bridle, J. S. 1990, Neurocomputing (Berlin: Springer), 227
Burke, D. S., Brundage, J. F., Redfield, R. R., et al. 1988, NEJM, 319, 961

Chaturvedi, P., Khan, A., Tian, M., Huerta, E. A., & Zheng, H. 2022, Front. Artif. Intell., 5, 828672
Elshamy, R., Abu-Elnasr, O., Elhoseny, M., & Elmougy, S. 2023, NatSR, 13, 8814
Fukushima, K. 1980, Biol. Cybern., 36, 193
Gerosa, D., Kesden, M., Berti, E., OShaughnessy, R., & Sperhake, U. 2013, PhRvD, 87, 104028
Glanzer, J., Banagiri, S., Coughlin, S. B., et al. 2023, CQGra, 40, 065004
Goceri, E. 2019, in IEEE International Topic Meeting on Image Processing Theory, Tools and Applications (IEEE), 1
Goodfellow, I., Bengio, Y., & Courville, A. 2016, Deep Learning (Cambridge, MA: MIT Press)
Harry, I., Privitera, S., Bohe, A., & Buonanno, A. 2016, PhRvD, 94, 024012
Hasan, N., Bao, Y., Shawon, A., & Huang, Y. 2021, SN Comput. Sci., 2, 1
Haykin, S. 1998, Neural Networks: A Comprehensive Foundation (2; Englewood Cliffs, NJ: Prentice- Hall)
He, K., Zhang, X., Ren, S., & Sun, J. 2016, in Proc. IEEE Conf. on Computer Vision and Pattern Recognition (New York: IEEE)
Hild, S., Abernathy, M., Acernese, F., et al. 2011, CQGra, 28, 094013
Hild, S., Chelkowski, S., Freise, A., et al. 2010, CQGra, 27, 015003
Hu, Y., Huber, A., Anumula, J., & Liu, S.-C. 2018, arXiv:1801.06105
Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. 2016, arXiv:1608.06993
Huerta, E. A., Kumar, P., Agarwal, B., et al. 2017, PhRvD, 95, 024038
Huerta, E. A., Kumar, P., McWilliams, S. T., OShaughnessy, R., & Yunes, N. 2014, PhRvD, 90, 084016
Husa, S., Khan, S., Hannam, M., et al. 2016, PhRvD, 93, 044006
Jaranowski, P., Królak, A., & Schutz, B. F. 1998, PhRvD, 58, 063001
Klimenko, S., Vedovato, G., Drago, M., et al. 2016, PhRvD, 93, 042004
LIGO Scientific Collaboration, Aasi, J., Abbott, B. P., et al. 2015, CQGra, 32, 074001
LIGO Scientific Collaboration,Virgo Collaboration, KAGRA Collaboration 2018, LVK Algorithm Library—LALSuite, Free software (GPL)
Macleod, D., Harry, I. W., & Fairhurst, S. 2016, PhRvD, 93, 064004
Maggiore, M., Van Den Broeck, C., Bartolo, N., et al. 2020, JCAP, 2020, 050
Medar, R., Rajpurohit, V. S., & Rashmi, B. 2017, in IEEE International Conference on Computing, Communication, Control and Automation (Piscataway, NJ: IEEE), 8
Munro, P. 2010, in Backpropagation, ed. C. Sammut & G. I. Webb (Boston, MA: Springer), 73
Murphy, K. P. 2012, Machine Learning: A Probabilistic Perspective (Cambridge, MA: MIT Press), 80
Nguyen, Q. H., Ly, H.-B., Ho, L. S., et al. 2021, Math. Probl. Eng., 2021, 4832864
Nitz, A., Harry, I., Brown, D., et al. 2023, gwastro/pycbc: v2.3.3 release of PyCBC, Zenodo, doi:10.5281/ZENODO.10473621
O'Shea, K., & Nash, R. 2015, arXiv:1511.08458
Planck Collaboration, Ade, P. A. R., Aghanim, N., et al. 2016, A&A, 594, A13
Powell, J., Sun, L., Gereb, K., Lasky, P. D., & Dollmann, M. 2023, CQGra, 40, 035006
Punturo, M., Abernathy, M., Acernese, F., et al. 2010, CQGra, 27, 194002
Raykar, V. C., & Saha, A. 2015, in Machine Learning and Knowledge Discovery in Databases (Berlin: Springer), 3
Regimbau, T., Dent, T., Del Pozzo, W., et al. 2012, PhRvD, 86, 122001
Rodriguez, C. L., Chatterjee, S., & Rasio, F. A. 2016, PhRvD, 93, 084029
Rodriguez, C. L., Morscher, M., Pattabiraman, B., et al. 2015, PhRvL, 115, 051101
Roodschild, M., Gotay Sardinas, J., & Will, A. 2020, Prog Artif Intell., 9, 351
Russakovsky, O., Deng, J., Su, H., et al. 2015, Int. J Comput. Vis., 115, 211
Shah, R., Bhaumik, A., Mukherjee, P., & Pal, S. 2023, JCAP, 2023, 038
Simonyan, K., & Zisserman, A. 2014, arXiv:1409.1556
Stehman, S. V. 1997, RSEnv, 62, 77
Tan, J., Yang, J., Wu, S., Chen, G., & Zhao, J. 2021, arXiv:2106.04525
Vijendra Babu, D., Karthikeyan, C., Shreya, & Kumar, A. 2020, in IOP Conf. Series: Materials Science and Engineering 993 (Bristol, UK: IOP Publishing), 012080
Wang, G.-J., Li, S.-Y., & Xia, J.-Q. 2020, ApJS, 249, 25
Wette, K. 2020, SoftX, 12, 100634
Wu, S., Zhong, S., & Liu, Y. 2018, Multimed. Tools Appl., 77, 10437
Xu, D., Zhang, S., Zhang, H., & Mandic, D. P. 2021, NN, 139, 17
Yan, R.-Q., Liu, W., Yin, Z.-Y., et al. 2022, RAA, 22, 115008
Yin, Q., Shen, L., Lu, M., Wang, X., & Liu, Z. 2013, JSEE, 24, 26