Deep learning for core-collapse supernova detection

M. López,[1,2,†] I. Di Palma,[3,*] M. Drago,[3,4] P. Cerdá-Durán,[5] and F. Ricci[3]

[1]*Institute for Gravitational and Subatomic Physics (GRASP), Department of Physics, Utrecht University, Princetonplein 1, 3584 CC Utrecht, Netherlands*
[2]*Nikhef, Science Park 105, 1098 XG Amsterdam, Netherlands*
[3]*Università di Roma La Sapienza, I-00185 Roma, Italy and INFN, Sezione di Roma, I-00185 Roma, Italy*
[4]*Gran Sasso Science Institute (GSSI), I-67100 L'Aquila, Italy and INFN, Laboratori Nazionali del Gran Sasso, I-67100 Assergi, Italy*
[5]*Departamento de Astronomía y Astrofísica, Universitat de València, Dr. Moliner 50, 46100 Burjassot (Valencia), Spain*

(Received 2 December 2020; accepted 9 February 2021; published 10 March 2021)

The detection of gravitational waves from core-collapse supernova (CCSN) explosions is a challenging task, yet to be achieved, in which it is key the connection between multiple messengers, including neutrinos and electromagnetic signals. In this work, we present a method for detecting these kind of signals based on machine learning techniques. We tested its robustness by injecting signals in the real noise data taken by the Advanced LIGO-Virgo network during the second observing run, O2. We trained a newly developed Mini-Inception Resnet neural network using time-frequency images corresponding to injections of simulated phenomenological signals, which mimic the waveforms obtained in 3D numerical simulations of CCSNe. With this algorithm we were able to identify signals from both our phenomenological template bank and from actual numerical 3D simulations of CCSNe. We computed the detection efficiency versus the source distance, obtaining that, for signal to noise ratio higher than 15, the detection efficiency is 70% at a false alarm rate lower than 5%. We notice also that, in the case of the O2 run, it would have been possible to detect signals emitted at 1 kpc of distance, while lowering down the efficiency to 60%, the event distance reaches values up to 14 kpc.

DOI: 10.1103/PhysRevD.103.063011

I. INTRODUCTION

The recent discovery of gravitational waves and high-energy cosmic neutrinos marked the beginning of a new era of the multimessenger astronomy. These new messengers, along with electromagnetic radiation and cosmic rays, give new insights into the most extreme energetic cosmic events. Among them supernovae explosion is one of the challenging targets of this new astronomical approach.

The advanced generation of gravitational wave detectors has proved their capacity of detecting gravitational waves (GWs) from astrophysical processes. While gravitational waves have been detected from mergers of binary black holes and binary neutron stars, core collapse supernovae (CCSN) have not been detected yet and they still represent a puzzle to solve. We had confirmation of the basic CCSN theory through the detection of MeV neutrinos from the SN1987A [1]: the collapse of a massive star's core is driven by the release of gravitational energy and the vast majority of this energy is realized in neutrinos. However, the details of the mechanism of the explosion are still an open question and the astronomical community is trying to disentangle the supernova explosion mechanism with intense studies.

Massive stars ($M > 8\ M_{\odot}$) spend most of their lives burning hydrogen into helium, which settles in the core and, when temperatures increase sufficiently, burns into heavier nuclei until iron is reached. The resulting iron core is supported by electron degeneracy pressure. When the core reaches the Chandrasekhar mass, pressure cannot support its own weight anymore and collapses. The collapse of the inner core is stopped abruptly when nuclear saturation density is reached at the center and the inner core bounces back. Shortly after the core bounce neutrino emission carries away energy from the postshock layer. In the present models of the neutrino driven supernovae explosions, the intense hydrodynamic mass motion has to play a significant role in the energy transfer by the neutrino flux to the medium behind the stagnating core-bounce shock, reviving the outward shock motion and thus initiates the SN explosion. Because of the weak coupling of neutrinos in the region of this energy deposition, in the hydrodynamic models of the explosions a large variety of physical ingredients are needed [2,3]. This so-called

[*] Corresponding author.
Irene.DiPalma@roma1.infn.it
[†] This research was conducted during my studies at Maastricht University.

neutrino driven mechanism [4] is the dominant theory to explain CCSN explosions in slowly rotating progenitors. Observationally only ~1% of the events shows signatures of fast rotation (broad-lined type Ic SNe [5] or long GRBs [6]), therefore neutrino-driven explosions are likely the most common type of CCSN and we will focus this work on those.

In a supernova explosion, GWs are generated in the inner core of the source, so that this messenger carries direct information of the inner mechanism. The feasibility of this scenario will be supported by the joint observation of neutrino and gravitational wave emission from CCSN, by assessing the correlation between neutrino emission and collapsed core motion. Although the phenomenon is among of the most energetic in the Universe, the amplitude of the gravitational wave impinging on a detector on the Earth is extremely faint. For a CCSN in the center of the Milky way, a rare event, we could expect amplitudes of the metric tensor perturbations ranging between 10^{-21}–10^{-23}. To increase the detection probability we should increase the volume of the universe to be explored and this can be achieved both by decreasing the detector noise and using better performing statistical algorithms.

The impossibility of using template-matching techniques in this case, due to the complexity and stochasticity of the waveform, makes it necessary to find new ways to improve the detection statistics. Current efforts to search for gravitational waves from CCSN include targeted searches for observed nearby SNe [7,8] and all-sky generic searches for bursts [9,10]. For the latter, two independent pipelines are used: coherent Waveburst (cWB) [11] and omicron-LIB (oLiB) [12], while BayesWave [13] is a followup of cWB GW candidate events. These searches use algorithms based on excess power to identify signals buried in the detector's noise without taking advantage of any specific feature of CCSN waveform.

In Ref. [14] it was proposed to use machine learning techniques to take advantage of the peculiarities of the CCSN GW signal with the goal of increasing our detection capability with respect to current methods. In particular, the focus was on the monotonic raise of the GW signal in the time-frequency plane due to the g-mode excitation, which is the dominant feature present in the GW spectrum. A similar approach has been followed recently by Refs. [15–17] and in general there has been an increasing interest in the GW community for the use of machine learning methods (see Ref. [18] for a review). Additionally, there has been an attempt of using machine learning techniques for inference in this context [19].

In this paper we follow a similar approach as in Ref. [14]. The main differences are

(i) the use of a more sophisticated convolutional neural network (CNN);

(ii) the injection of simulated CCSN signals in real noise of the three advanced detectors of the LIGO-Virgo network, as measured during August 2017 (Ref. [14] only considered Gaussian noise);

(iii) the improvement of the phenomenological templates used during the training of the CNN network to better match results from numerical simulations.

This paper is structured as follows. In Sec. II we describe our newly improved phenomenological waveform templates that are used to train the CNN networks presented in Sec. III. In Sec. IV we describe the detector noise data used for the injections. Section V is devoted to the procedure of the training of the CNN network and its behavior. In Sec. VI we report the results, showing the detection performance in terms of signal to noise ratio and event distance. Results are discussed in Sec. VII and then we conclude.

II. WAVEFORMS

To implement our search method, we have pursued an approach similar to Ref. [14]. We consider a parametric phenomenological waveform designed to match the most common features observed in the numerical models of CCSN. We focus our attention on the g-modes excitation, the most common feature of all models developed so far to describe the CCSN phenomena, responsible for the bulk of the GW signal in the postbounce evolution of the proto-neutron star. The measurement of the g-mode has been proposed as way of inferring the properties of the proto-neutron star; see, e.g., Refs. [20–23]. The aim of our phenomenological template is to mimic the raising arch observed in core-collapse simulations. To this end we will consider a damped harmonic oscillator with a random forcing, in which the frequency varies with time. The phenomenological templates used in this work differ with respect to the ones in Ref. [14] in two aspects: we use a new and more flexible parametrization for the frequency evolution and we use the distance as a parameter. The phenomenological templates are calibrated to mimic the features in the numerical simulations for nonrotating progenitor stars by Refs. [20,24–31], named the *waveform calibration set*, hereafter.

The new parametrization describes the evolution of the frequency of the g-modes $\nu(t)$ as a cubic spline interpolation to a series of discrete points (t_i, ν_i), where t_i corresponds to postbounce times. Given the relatively simple behavior of $\nu(t)$ observed in numerical simulations, it is sufficient to use three points with $t_i = (0, 1, 1.5)$ s. ν_0, ν_1, and ν_2 are then three new parameters of the template.

In Ref. [14] the amplitude of the generated waveforms has been chosen according to the signal to noise ratio (SNR). In this work we want to go one step further and use distance as a parameter for the waveform generator. In order to do that we relate the amplitude of the waveform with its distance using the data in the waveform calibration set. First, we have to measure for each simulation the typical strain of the component of the signal related to

g-modes. To this aim we apply a high pass filter at 200 Hz, and then we use the section of the waveform containing 99% of the waveform energy to compute the root mean square (rms) value. This procedure filters out signal power at lower frequencies related to other effects different from g modes (standing-shock accretion instabilities, prompt convection, and large scale asymmetries due to shock propagation) that are not considered for this work. Figure 1 shows distribution of the logarithm of the rms strain for 2D and 3D simulations at 10 kpc. There are significant differences between different simulations depending on the dimensionality. The mean and standard deviation for each distribution is $\log_{10} h_{\mathrm{rms,2D}} = -22.4 \pm 0.42$ and $\log_{10} h_{\mathrm{rms,3D}} = -23.1 \pm 0.29$, for the 2D and 3D cases, respectively. Given that 3D simulations are more realistic we use this normalization to generate our phenomenological waveforms. As a consequence, we end up with template amplitudes about a factor 5 smaller than typical 2D simulations. In our waveform generator, the strain of each of the waveforms is scaled to have a rms strain corresponding to a random value following a normal distribution with the mean and standard deviation of our normalization, and scaled to the corresponging distance.

In synthesis, we have a waveform template that depends on a set of 8 free parameters as reported in Table I. In this table the quality factor Q refers to the damping of the harmonic oscillator and the driver frequency, ν_{driver}, to the mean frequency of the random impulsive force acting on the oscillator; see Ref. [14] for details. Additionally, for any combination of those parameters we can generate multiple realizations due to the random component in the excitation of the harmonic oscillator and on the random value of the rms strain. In order to be able to represent the variety of g-mode features observed in the waveform calibration set, we provide ranges covering all the possibilities (see Table I). To this parameter space one has to add additional restrictions to ensure the monotonicity ($\nu_2 > \nu_1 > \nu_0$) and convexity $((\nu_1 - \nu_0)/(t_1 - t_0) \geq (\nu_2 - \nu_1)/(t_2 - t_1))$ of $\nu(t)$, as seen in the numerical simulations. We have created the waveform template bank that contains 504 different realisations of this parameter set, for each distance, resulting of applying the restrictions above to the 9072 possible combinations of the parameters in Table I. In this way we obtain a reasonably dense covering of the parameter space, an example of a phenomenological waveform is shown in Fig. 2. The computational time of the generation of one phenomenological waveform is about 6 ms. This makes it possible to generate the large template banks necessary to train the CNN, something which would be prohibitively expensive using multidimensional numerical simulations.

TABLE I. Parameter space of the phenomenological templates. The second, third, and fourth columns indicate the range minimum and maximum, respectively) for each parameter and the spacing used in the sampling of the parameter space. For Q and D we show the actual values instead. All times are postbounce times.

Parameter	min.	max.	Δ	Description
t_{ini} [s]	0	0.2	0.1	Beginning of the waveform
t_{end} [s]	0.2	1.5	0.1	End of the waveform
ν_0 [Hz]	50	150	50	Frequency at bounce
ν_1 [Hz]	1000	2000	500	Frequency at 1 s
ν_2 [Hz]	1500	4500	1000	Frequency at 1.5 s
ν_{driver} [Hz]	100	200	100	Driver frequency
Q	(1,5,10)			Quality factor
D [kpc]	(1,2,5,10,15)			Distance to source

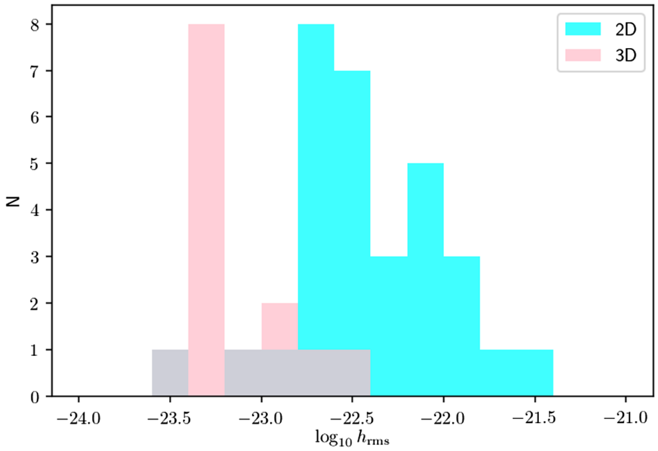

FIG. 1. Number of simulations with a given g-mode root mean square strain at 10 kpc (per logarithmic interval) for 2D (blue bars) and 3D (red bars) simulations in the waveform calibration set.

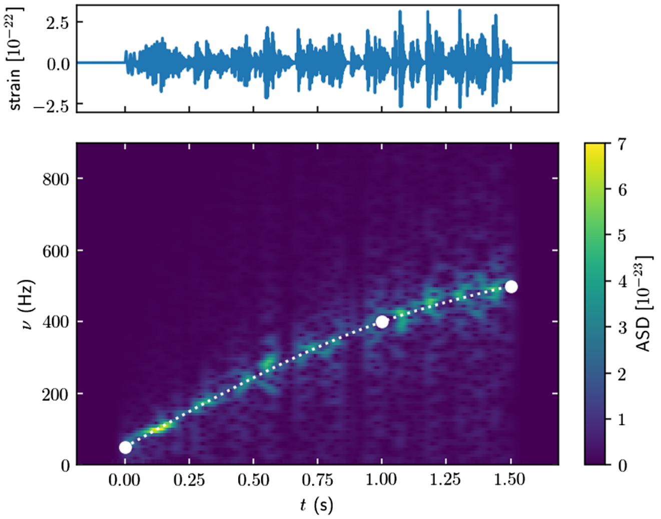

FIG. 2. Example of a phenomenological waveform generated by the waveform generator. The upper panel shows the strain as a function of time and the lower panel the corresponding spectrogram. The white-dotted curve shows the time-frequency dependency used to generate the waveform. White circles represent the pairs (t_i, ν_i) used to generate this curve.

III. METHODOLOGY

A. Challenges and milestones of deep learning

The application of deep learning (DL) across science domains is a booming enterprise. DL algorithms have been very successful in a variety of tasks and in recent times it has emerged as a new tool in the GW field. These methods are able to perform analysis rapidly since all the intensive computation is diverted to the one-time training stage, which could make them orders of magnitude faster than a conventional matched filtering technique. In addition, there are no limitations in the size of the templates bank of GW signals, and even more, it is preferable to use large datasets to cover as deep a parameter space as possible. Because of this fact they sparked the interest of several authors, who have built deep-learning algorithms to demonstrate their power on specific examples, including CCSN [15–17] among others [18,32–34].

A CNN is a specialized kind of DL algorithm to process data that has a known gridlike topology and can learn to differentiate a variety of input types due to its ability for pattern recognition [35]. In a CNN, the input is convolved with a filter, which varies according to the characteristics of the data since it can be *learned* by the network. The computations are performed at each step, as the filter is slid onto the input to compute the corresponding value in the output feature map. Despite the automatic learning of the filter, some parameters need to be tuned by hand.

The input of two-dimensional CNN are images, which have 3 dimensions: width w_{in}, height h_{in}, and depth d_{in}. The width and the height are the image dimensions expressed in pixels, while the depth represents the number of input channels. Commonly, color images have 3 input channels that correspond to colors red, green, and blue. Hence, assume that an image with dimensions $(w_{\mathrm{in}}, h_{\mathrm{in}}, d_{\mathrm{in}})$ is convolved with f filters of size $k \times k$. The amount of pixels that the filter slides at each step is the stride s, while the border of zeros added has a width p, called convolutional padding. The result of the convolution yields the following output:

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

We can also calculate the number of parameters that we need to train for each layer (or level) as $(k \times k \times d_{\mathrm{in}} + 1) \times d_{\mathrm{out}}$. It is interesting to note that each layer of the CNN looks at different patterns since they can *learn* different filters, depending on the information provided by the previous layers. Thus, these layers learn to recognize visual patterns by first extracting local features and subsequently combining them to obtain higher-level representations.

With these ideas in mind, Ref. [14] provided clear evidence that, under relatively simplified conditions, deep CNN algorithms could be more efficient to extract GW signals from CCSNe than the current methodology. Therefore, the aim of this work is to improve the neural network developed in Ref. [14], going deeper with convolutions to increase accuracy while keeping computational complexity at a reasonable cost.

The most straightforward way of improving the performance of deep neural networks is by increasing their size, which includes the number of layers and the number of neurons per layer. Nonetheless, enlarging a network implies training a larger amount of parameters and overcomplicating the model, which increases dramatically the computational cost and reduces the generalization ability of the network, i.e., the network would be prone to overfitting. A fundamental way of solving these issues would be to move from fully connected to sparsely connected architectures, as is discussed in Ref. [36]. This work presents a sophisticated network topology construction, the so-called inception network, that tries to approximate a sparse structure. The architecture is composed by blocks of convolutions, known as inception modules. The input of each block is convolved in parallel by separate CNN layers with different kernels, while the outputs of all the convolutions are stacked, as we can observe in Fig. 3. In such a way, a sparse network is built without the necessity of choosing a particular kernel size, but computational complexity increases drastically. To prevent a high computational cost the authors introduce *dimensionality reduction*, i.e., 1×1 convolutions that reduce the depth of the output. If we convolve our input $(w_{\mathrm{in}}, h_{\mathrm{in}}, d_{\mathrm{in}})$ with f filter 1×1, stride $s = 1$ and padding $p = 0$, according to Eq. (1) the output will be $(w_{\mathrm{in}}, h_{\mathrm{in}}, f)$. Therefore, if $f < d_{\mathrm{in}}$ the depth and the number of parameters will be greatly reduced. In later releases of the inception network, the authors explore further the idea of dimensionality reduction.

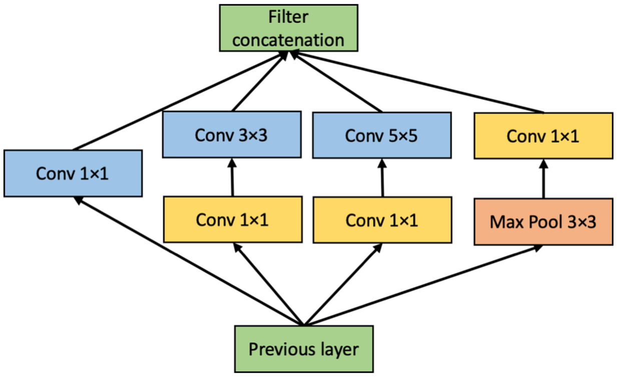

FIG. 3. Inception module with dimensionality reduction, adapted from Ref. [36].

Reference [37] explores other ways of factorizing convolutions in various settings, especially in order to increase the computational efficiency of the solution without reducing the expressiveness of the block. First, the authors examine the factorization into smaller convolutions, where they claim that 5×5 convolution can be *factorized* into two 3×3 convolutions since the final output has the same dimensions. Nonetheless, the main difference between both processes are the number of parameters. A 5×5 convolution needs $(5^2 \times d_{\mathrm{in}} + 1) \times d_{\mathrm{out}}$ parameters to train, while for two 3×3 convolutions it is necessary to train $2 \times (3^2 \times d_{\mathrm{in}} + 1) \times d_{\mathrm{out}}$ parameters, which is less computationally expensive. Second, they analyze the factorization into asymmetric convolutions, such that a convolution $c \times c \to c \times 1$ and $1 \times c$ convolutions. Again, the outputs of both processes have the same dimensionality but different amount of trainable parameters, $(c^2 \times d_{\mathrm{in}} + 1) \times d_{\mathrm{out}} > 2 \times (c \times 1 \times d_{\mathrm{in}} + 1) \times d_{\mathrm{out}}$. Therefore, in Ref. [37] the authors factorize 5×5 convolutions into 3×3, which in turn are factorized by 3×1 and 1×3 convolutions, to lighten the computations.

Another obstacle of deeper networks is the degradation problem, where with increasing depth, accuracy gets saturated and then degrades rapidly. In Ref. [38] this problem is approached by introducing a deep neural network, called the residual network or ResNet. This network is able to learn the identity function using shortcut connections that skip one or more layers, which are also known as “skip connections.” Therefore, the network is reminded every few layers how the input was a few layers before, which can be translated in learning the identity function with a simple demonstration. Furthermore, in Ref. [38] different empirical results show that the degradation problem is well addressed since accuracy gains are obtained from increasing depth.

Because of the improvements in accuracy obtained with the inception network and Resnet, in Ref. [39] they explored the combination of these two brilliant architectures, while factorization is discussed in Ref. [37]. As a result, they developed, among others, an architecture called Inception-Resnet v1 which is ~90 layers deep. It was demonstrated that the introduction of residual connections lead to a dramatic improvement in the computational speed, while it was shown that inception-Resnet algorithms were able to achieve higher accuracies with less iterations of the training phase.

Our problem is simpler than the task performed in Ref. [39], since we only need to discriminate between two classes: templates that contain a GW CCSN signal with noise (event class) and templates that contain only noise (noise class). Hence, the need to increase the complexity of the CNN in our case is due to the loudness of the noise power in the data, rather than the number of the different classes. As a consequence, we have developed reduced (“mini”) versions of Inception v3, Resnet, and Inception-Resnet v1, using the original building blocks of those networks, but adapting them to our needs and limiting the number of layers to ≤30 to avoid overfitting.

During our investigation it was observed that inception blocks did improve the performance with respect to a vanilla CNN with few skip connections, i.e., Mini Inception v1 obtained a better performance than Mini Resnet. From our results we concluded that skip connections enhance the learning for very deep networks (>30 layers). Furthermore, preliminary results of Mini Inception-Resnet v1 showed that the use of factorized inception blocks enlightened the computations while increasing the complexity and the generalization ability of the network, whereas skip connections enhanced the learning of the network by reminding of the previous input. Because of its high performance this last network was optimized and is presented in this work.

B. Architecture of Mini Inception-Resnet

For the development of our Mini Inception-Resnet network, including the model definition, the training and the validation phases, we have used the Keras frameworks [40], based on the TensorFlow backend [41]. We employ the Adam optimizer [42] with a learning rate $lr = 0.001$ and $\epsilon = 10^{-6}$ to avoid divisions by zero when computing backpropagation. The activation functions of all the convolutional layers is the $relu$ activation function, $ReLU(x) = \max(0, x)$. We employ a batch size of 64 because, for our particular task, it is a good trade-off between computational complexity and performance.

Despite facing a classification problem with two classes, the approach used in Ref. [14] is to employ the categorical cross-entropy loss function with a softmax activation function in the last layer, i.e., the problem is treated as a multiclass classification problem with two classes. In this work we simplify this approach by using a *binary cross-entropy* instead and a sigmoid activation function for the output, i.e., we address the problem as a classification problem with a positive class (event class) and a negative class (noise class). Therefore, the output of the network is a probability vector θ, which contains the probabilities of the template belonging to one class or another. The classification task is then performed according to a predefined threshold θ^*, i.e., the template will be classified as *event class* only if this probability overcomes θ^*. It is important to note that *categorical cross-entropy* and softmax activation function are the generalizations of *binary cross-entropy* and a sigmoid activation function, respectively.

In Ref. [39], the authors build 5 different types of blocks, namely, Inception-ResNet-A, Inception-ResNet-B, Inception-ResNet-C, Reduction-A, and Reduction-B. The modules Inception-ResNet-B and Reduction-B are the most expensive blocks, since the convolutions inside them are 1×7, 7×1, and 7×7. Hence, we discard these modules to implement the reduced version of this algorithm. At the same time, we shrink the amount of parameters of our

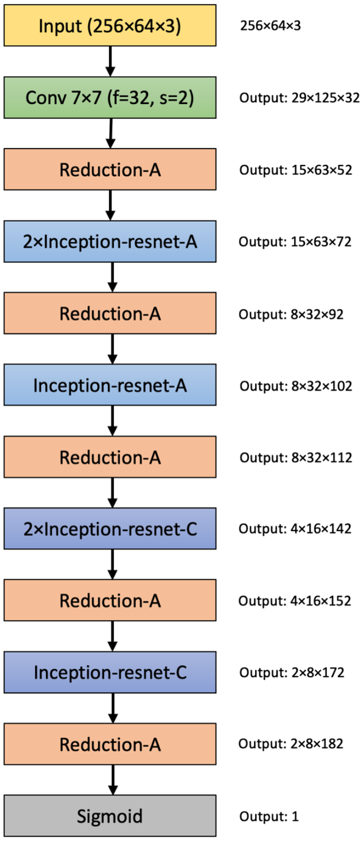

FIG. 4. The overall schema of the Mini Inception-Resnet network. For the detailed modules, please refer to Figs. 5, 6, and 7.

network by interspersing Inception-Resnet modules with Reduction-A blocks (Fig. 4).

The Inception-ResNet-A block (see Fig. 5) is equivalent to the inception module shown in Fig. 3. It is interesting to note that the max pooling layer is substituted by the "shortcut connection," and the 5×5 convolution is *factorized* by two 3×3 convolution layers.

Moreover, Inception-ResNet-C block (see Fig. 6) is the equivalent to the inception module without the 5×5 convolution layer. Note that the max pooling layer is again replaced by the shortcut connection, and the 3×3 convolution is factorized by 1×3 and 3×1 convolution layers. The module Reduction-A (see Fig. 7) shrinks the number of parameters thanks a 3×3 max pooling layer.

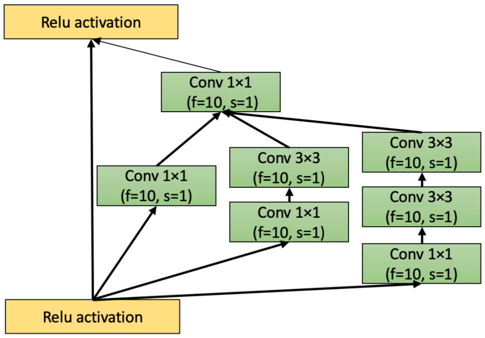

FIG. 5. The schema for Inception-ResNet-A, adapted from Ref. [39].

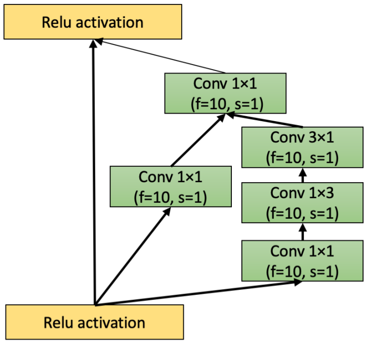

FIG. 6. The schema for Inception-ResNet-C, adapted from Ref. [39].

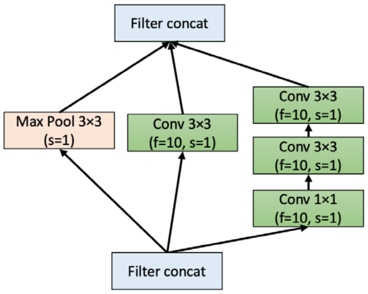

FIG. 7. The schema for Reduction-A, adapted from Ref. [39].

Because of its deepness, the resulting Mini Inception-Resnet architecture is much more flexible than the one presented in Ref. [14]. As we have discussed previously, increasing the number of layers might be counterproductive and would drastically increase the computational complexity of the network. Nonetheless, these two concerns are solved with the incorporation of shortcut connections, which allow the input not to be forgotten, and a factorized gridlike architecture that alleviates the computational complexity of the neural network, on top of avoiding overfitting.

IV. DATA

The Advanced LIGO-Virgo detector network collected data for three runs, during which more than 60 possible GW events have been identified [43,44]. Almost all of them, if confirmed, are associated with the coalescence of binary systems with the peculiar chirp shape of the signal. This feature is used to extract the signal from the detector noise adopting a matched filter approach. In the case of CCSN, data must be selected and processes using different algorithms. To enhance the detection probability and reduce the false alarm rate, the proposed method implies to select data in a time window around trigger times given by the neutrino observatories and take advantage of common GW features predicted by CCSN numerical simulations.

To assess the robustness of our method, we selected data from the second observing run of the Advanced GW detectors, without relying on any neutrino information. In particular, we chose a stretch of real data even containing glitches, taken during August 2017, when Virgo joined the run [45]. The period includes about 15 days of coincidence time among the three detectors and we used this dataset to generate about 2 years of time-shift data to train and test the neural network as noise class.

To build images for our neural network algorithm we use the internal features of cWB algorithm.[1] cWB is used by the LIGO and Virgo Collaborations for the GWs search that uses minimal assumptions of the expected signal. It measures the energy excesses over the detector noise in the time-frequency domain and combines these excesses coherently among the various detectors of the network [9,10]. It is used in both the low latency mode [46] and in triggered searches for CCSN [7,8]. In our work we used this software tool to compute the Wavelet transform, on the base of which the images of 256 × 64 pixels are built covering the frequency band from 0 to 2048 Hz and a time range of 2 s. Because the gravitational-wave signal is embedded in noise and difficult to extract, in Ref. [14] a technique to visually enhance the coincidences among all the interferometers of the network has been developed. The method consists in using primary colors for the spectrograms of each detector: red (R) for LIGO-Hanford, green (G) for LIGO-Livingston, and blue (B) for Virgo.

The main results in this work present some differences with respect to Ref. [14]: we incorporate the information of the source distance; we cover a larger parameter space with our phenomenological waveforms; we consider real data from the second observing run, we do not anymore build images based on cWB selective information. The idea is to use the neural network as a follow-up of multimessenger information. We define the starting time of each image every 2 s, echoing the choice done in Ref. [14]. The images containing the central time of injected signals are considered as the *event class*, instead of the ones without signals that are the *noise class*. The injected signal is expected to be about 600–700 ms in duration, drawn everywhere in the image, with a small probability to be between two consecutive images. Such images are used for the training in any case, therefore the network can recognize also a partial signature of the event.

We first compare the new architecture of the neural network using, for the training and validation, the same data set of Ref. [14] (Sec. IV A). To tune our CNN we train the algorithm with the new phenomenological templates injected in the real noise (Sec. IV B). Finally we test the network with injections of phenomenological waveforms (Sec. IV C) and waveforms from CCSN numerical simulations (Sec. IV D).

A. Previous set

In Ref. [14], phenomenological supernova signals were injected in Gaussian noise simulating the final expected sensitivity of Advanced LIGO and Virgo detectors. Signals were injected at fixed network SNR, and did not include any information about source distance. This set was constructed using the information given by the cWB algorithm and, unlike in the following datasets, only using events passing the first stage of cWB analysis. This set contains about 10 000 images with signals for 11 different SNR ranging from 8 to 40 and the same amount with only noise, 75% of the signals are used to train the network and 25% for validation.

B. Training set

The training set for CCSN signals has been constructed injecting waveforms at fixed distances: 0.2, 0.4, 1, 2, and 3 kpc. For this purpose, we have used the waveform template bank described in Sec. II injecting, for each distance, of the order of 70 000 waveforms, with random sky localization. 75% of the set is used in the actual training while the remaining 25% is used for validation. The upper distance has been chosen according to the requested minimal SNR of the injected signals, in which the trace of the CCSN signal is distinguishable from the noise in the image passed to neural network.

C. Blind set

In the blind set we injected a new ensemble of about 2 60 000 simulated signals, generated by the phenomenological templates described in Sec. II. In this case distance is chosen in a uniform distribution between 0.2 and 15 kpc, position in the sky are randomly chosen. This set is used to quantify the detection efficiency and to test the network. It is not involved in the training or validation procedure. The set is bigger than the training set, to maintain enough statistic for all the distances.

D. Test set

For the final test we perform injections using CCSN waveforms from numerical simulations found in the literature. In particular we focus on 3D simulations of nonrotating progenitors representative of the neutrino driven mechanism. The selection *test set*, hereafter, see Table II, is performed based on the realism of the computed simulations in terms of neutrino transport and equation of state and on the completeness of the GW signal.[2] The selection includes models

[1] cWB home page, https://gwburst.gitlab.io/; public repositories, https://gitlab.com/gwburst/public documentation, https://gwburst.gitlab.io/documentation/latest/html/index.html.

[2] Some of the models in the literature compute less than 100 ms after bounce or have a poor sampling rate.

TABLE II. List of models of the test set used in the injections. M_{ZAMS} corresponds to the progenitor mass at zero-age in the main sequence (ZAMS). Unless commented, all progenitors have solar metallicity, result in explosions and their GW signal do not show signatures of the standing-shock accretion instability (SASI).

Model name	Reference	M_{ZAMS}	Comments
s9	[47]	9 $M_\odot$	Low mass progenitor, low GW amplitude.
s25	[47]	25 $M_\odot$	Develops SASI.
s13	[47]	13 $M_\odot$	Non-exploding model.
s18	[48]	18 $M_\odot$	Higher GW amplitude.
he3.5	[48]	⋯	Ultra-stripped progenitor (3.5 $M_\odot$ He core).
SFHx	[49]	15 $M_\odot$	Non-exploding model. Develops SASI.
mesa20	[50]	20 $M_\odot$	
mesa20_pert	[50]	20 $M_\odot$	Same as mesa20, but including perturbations.
s11.2	[31]	11.2 $M_\odot$	
L15	[28]	15 $M_\odot$	Simplified neutrino treatment.

with a variety of mass progenitors and features in the GW spectrum, and coincides with the choice for ongoing SN searches by the LIGO-Virgo-KAGRA collaboration. Except for model L15, none of the models coincide with the models selected for the waveform calibration set used in Sec. II. With this choice the injected waveforms are in practice completely uncorrelated to any information we have used to train the CNN network. The procedure is similar to the one used for the blind set of the previous test: we injected about 65 000 waveforms uniformly in distance and sky directions, from 100 pc to 15 kpc.

V. TRAINING METHODOLOGY

In this section we describe how we convert training images into categorical data for the identification of CCSN signatures in Gaussian and real noise, to solve our binary image classification task.

As in Ref. [14], we train the network using *curriculum learning*, where we start training with the easiest datasets, and then gradually the task difficulty is increased. We note that, although our new template bank is constructed using a series of fixed distances, the SNR follows, for each of these distances, a statistical distribution resulting from the random process used to generate the waveforms (see Sec. II). In practice, instead of using the distance, we define data as a set of templates that have SNRs in a fixed range. In this way, the difficulty of the datasets increases with decreasing SNR. It is important to note that because we use this methodology to improve the learning of the network, it is key to obtain a high performance when learning easy examples at high SNR, to be able to capture the hard examples later on. The datasets are balanced, so that 50% of the templates belong to the event class and 50% to the noise class. Because the present network is much larger than that in Ref. [14] where we had balanced training and validation sets, here we use 75% of the data for training and 25% for testing.

In Ref. [14] we measured the performance of the neural network in terms of the efficiency η_{CNN} and the false alarm rate FAR_{CNN}, which are equivalent to the true positive rate and the false discovery rate, respectively. Here we will redefine these variables in terms of the confusion matrix (see Table III), but the definitions are completely equivalent.

$$\eta_{CNN} = \frac{\text{correctly classified signals}}{\text{all the signals at CNN input}} = \frac{TP}{TP + FN}, \tag{2}$$

$$FAR_{CNN} = \frac{\text{misclassified noise}}{\text{all classified events}} = \frac{FP}{FP + TP}. \tag{3}$$

In this research we also measure the performance of our network with the *receiver operating characteristic* curve (ROC curve), which is created by plotting the true positive rate (TPR) agains the false positive rate (FPR). Note that the definition of η_{CNN} coincides with TPR, but FPR is defined as

$$FPR = \frac{FP}{FP + TN} \tag{4}$$

VI. RESULTS

A. Waveform injection in Gaussian noise: Comparison with previous results

In this subsection we will describe the experiments performed with injections in Gaussian noise. To train and validate the network, we use the dataset described in Sec. IV A, composed of waveforms ranging in the interval SNR $= [8, 40]$. This choice allows for a direct comparison with the results in Ref. [14] and it helps to improve the present software architecture.

To improve the performance of Ref. [14] it is necessary to minimize FAR_{CNN} while maximizing η_{CNN}. Therefore, from Eq. (3) we wish to minimize FP instead of FN, i.e., we need to penalize the algorithm when it classifies noise class as event class. To be able to penalize the algorithm we implement *weighted binary cross-entropy*, where we assign weight w to the noise class and weight 1 to the event class. We vary this parameter between $w = [1.0, 3.5]$, where $w = 1$ would be equivalent to a normal binary cross-entropy and $w = 3$ would mean that it is 3 times more important to correctly classify the noise class rather

TABLE III. Confusion matrix for event and noise class.

		Actual class	
		Event	*Noise*
Predicted class	Event	True positive (TP)	False positive (FP)
	Noise	False negative (FN)	True negative (TN)

than the event class. Nonetheless, in this work we only present the results of $w = \{1.0, 2.0\}$ to be able to compare the best working case ($w = 2.0$), with the base case ($w = 1.0$).

Moreover, the algorithm returns the probability θ that a certain template belongs to the event class. We want this probability to be high without dramatically decreasing η_{CNN}. Therefore, we define the decision threshold θ^* in range $[50\%, 85\%]$; when a given probability exceeds this value, we will classify the template as an event, otherwise, it is classified as noise. Therefore, we perform different experiments to tune w and θ. In Figs. 8 and 9, we obtain η_{CNN} and FAR_{CNN} for $w = \{1, 2\}$ and $\theta = \{50\%, 65\%, 85\%\}$.

In Fig. 8, we report the high performance of low θ in terms of η_{CNN}, paying the prize in even relative high FAR_{CNN}. The opposite behavior occurs for high θ. To be able to improve the probability distribution θ, we will penalize the loss function with $w = 2.0$. This means that the impact of correctly classifying noise templates is twice higher than correctly classifying event templates, as we show in Fig. 9 where the FAR_{CNN} is minimized with respect to Fig. 8 with some cost in η_{CNN}.

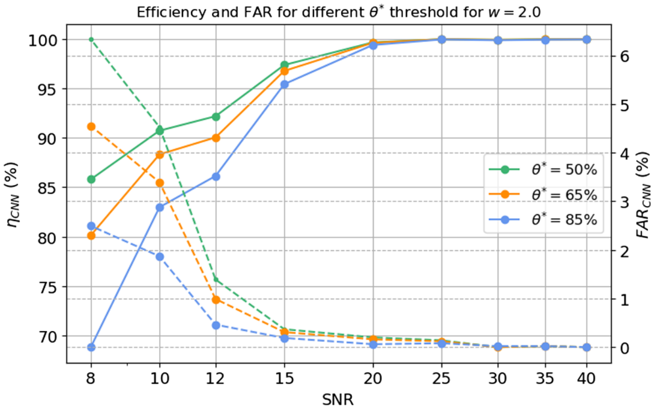

FIG. 9. η_{CNN} (solid lines) and FAR_{CNN} (dashed lines) for different SNRs computed during the validation process for $w = 2.0$ and different θ^* thresholds.

Notice that w will penalize the learning, so if the network is learning correctly the results would be enhanced, but it will lead to poor results otherwise. This is evident when we compare the results shown in Figs. 8 and 9: if we increase w we have less performance in terms of η_{CNN}, with little gains in FAR_{CNN}. To have a clearer comparison between Figs. 8, 9 and the results from Ref. [14], we plot the validation results of Mini Inception Resnet for $w = \{1, 2\}$ in Fig. 10.

Since we want to obtain a trade-off between η_{CNN} and FAR_{CNN}, we settle $w = 2.0$ and $\theta^* = 65\%$.

The main improvement of our network with respect to Ref. [14] is the minimization of FAR_{CNN} towards ~0% for the SNR in range [15, 20], while maintaining the same η_{CNN}. We note also that the poor performance at low SNR is due to the fact that this architecture is susceptible to the strong presence of Gaussian white noise, as is pointed out in Ref. [51]. Hence, the role of the decision threshold $\theta^* = 65\%$ is twofold. On one hand, with this decision threshold we obtain $\max(FAR_{CNN}) \approx 4\%$ for low SNR which is the upper limit obtained by the previous paper [14]. On the other hand, $\theta^* = 65\%$ provides us with a fair trade-off between η_{CNN} and FAR_{CNN} as we have discussed before.

In terms of speed performance, in a GPU *Nvidia* Quadro P5000 it takes 1 h 18 min to train, validate, and test Mini

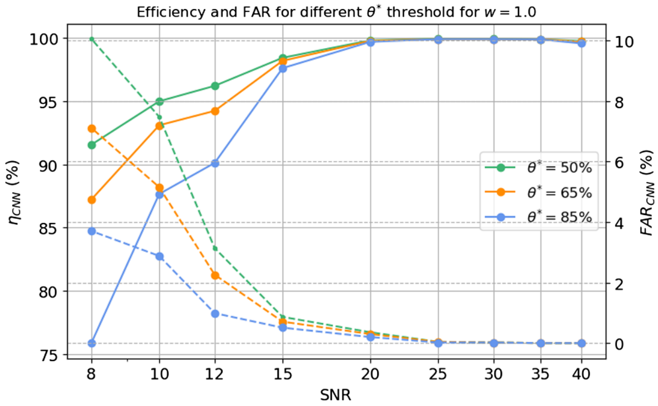

FIG. 8. η_{CNN} (solid lines) and FAR_{CNN} (dashed lines) for different SNRs computed during the validation process for $w = 1.0$ and different θ^* thresholds.

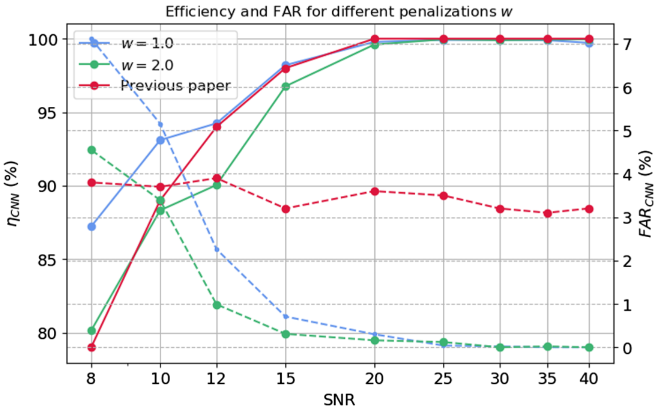

FIG. 10. η_{CNN} (solid lines) and FAR_{CNN} (dashed lines) as functions of SNR computed during the validation process of $w = \{1, 2\}$, with $\theta^* = 65\%$, and [14], where $\theta^* = 50\%$.

Inception Resnet for this particular dataset with 5 epochs for each SNR. A great part of this time is employed in training the neural network, so with bigger datasets the computational time will increase. Nonetheless, once the network is trained, the prediction is performed within minutes.

B. Waveform injections in real detector noise: Training and validation

In this section we describe the experiments performed using the *training set* (Sec. IV B). This set contains injected phenomenological signals in real noise in the interval $\mathrm{SNR} = [1, 232]$. As before, for each dataset at a given SNR we calculate FAR_{CNN} and η_{CNN} during the validation. We also vary the penalization parameter $w \in \{1, 2\}$ and as in the previous section we choose $w = 2$ and the decision threshold $\theta^* = 65\%$.

For the network to learn correctly the input, it is crucial to perform a smooth "curriculum learning." Because of the difficulty of the dataset, we separate the event templates into bins of size N and noise templates are packed accordingly. We performed the training for different N but a better trade-off between η_{CNN} and FAR_{CNN} was observed for $N = 30000$, which provided a smoother transition between SNR bins. Therefore, in Fig. 11 we show the results of the validation process having fixed $N = 30000$, $\theta^* = 65\%$, and $w = 2$.

In Fig. 11 we note that η_{CNN} is around 98% above $\mathrm{SNR} = 32$ and below this value η_{CNN} starts decreasing. Instead, FAR_{CNN} is around 0% but increases for SNR values below 20. For lower SNR values of the network the method tends to show more and more an erratic behavior that we foresee due to the statistical structure of the real noise.

This procedure is rather fast. In terms of speed performance, in a GPU *Nvidia* Quadro P5000 it takes 2h 21 min for Mini Inception-Resnet to train and validate for this particular dataset, but only 10 min to predict the *blind set* and *test set*. The time increase in the training phase is due to the fact that now we set the number of epochs to 10 instead of 5 to guarantee a better convergence of the network's trainable parameters.

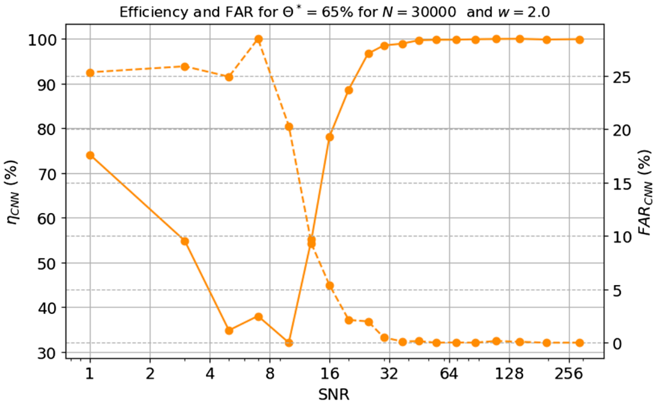

FIG. 11. η_{CNN} (solid line) and FAR_{CNN} (dashed line) for SNR in range [1, 232], for $w = 2.0$ and $\theta^* = 65\%$. These results are obtained from validating on 25% of the data that we have not trained on.

C. Waveform injections in real detector noise: Final results

In this section we present the results obtained when we used the network trained and optimized in the previous section on the data of the blind set (Sec. IV C) and the test set (Sec. IV D). The network has not been trained by any of the images of these two sets so they can be used for the final test of the performance of the network. The signals injected in the blind set correspond to waveforms generated by the same procedure used to generate the training set, while the injections in the test set correspond to realistic CCSN waveforms.

In Fig. 12, we report the histogram of the injections in the real noise. Such plot shows the robustness of the decision threshold $\theta^* = 65\%$ even in the case of real detector noise.

In Fig. 13 we plot the ROC curve and we calculate the area under the curve (AUC).

We note the high performance of the test set $(\mathrm{AUC} = 0.79)$ compared with that obtained for the blind set $(\mathrm{AUC} = 0.90)$. Even if we only trained our network with phenomenological waveforms from the template bank described in Sec. II, such waveforms mimic the behavior of the test set described in IV D, which is the main reason behind such good results.

Another interesting graph that shows the resemblance between the blind set and test set is Fig. 14. Here we plot η_{CNN} as a function of the distance.

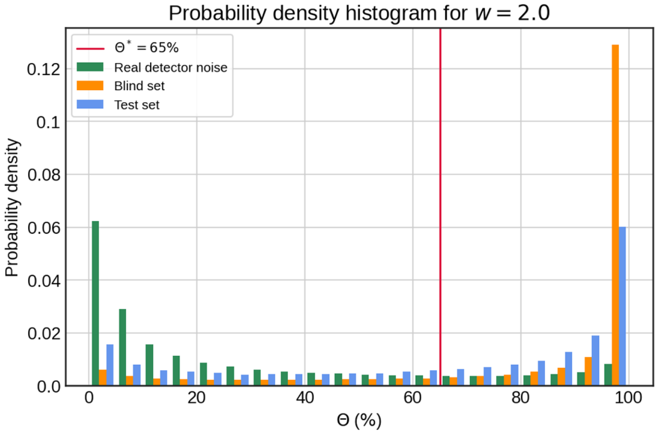

FIG. 12. Histogram of real detector noise and injections in real time as a function of the probabilities predicted by Mini Inception Resnet. The vertical line represents the chosen decision threshold $\theta^* = 65\%$. Given the counts of the ith bin c_i and its width b_i, we define the probability density as $c_i/(\sum_i^N c_i \times b_i)$, where N is the total number of bins of the histogram.

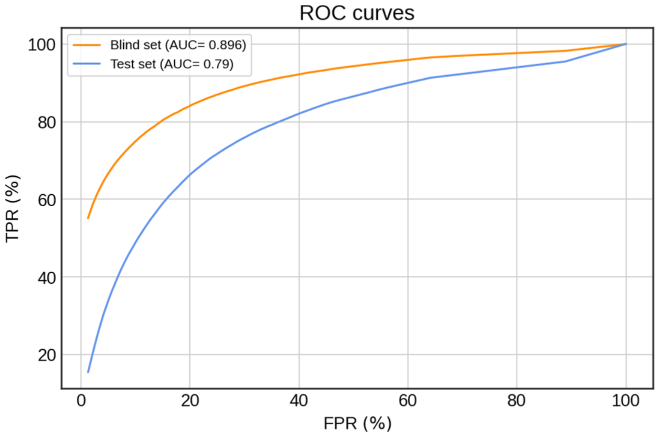

FIG. 13. Performance of our neural network for the blind set and the test set for $\{w, N, \theta^{*}\} = \{2, 30000, 65\%\}$. AUC is presented in the legend of the plot.

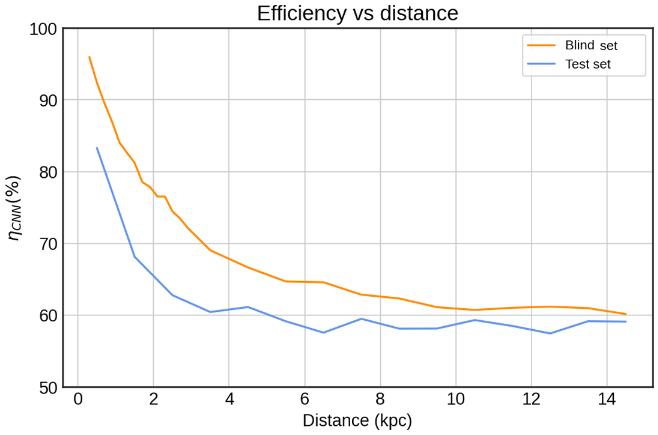

FIG. 14. η_{CNN} as a function of the distance computed during the testing process for $\{w, N, \theta^{*}\} = \{2, 30000, 65\%\}$.

As we can see, at short distances there is a difference in efficiency between blind set and test set of ≈10%, but when we increase the distance, they seem to reach a lower limit at $\eta_{CNN} \approx 60\%$. In Fig. 15 we also plotted η_{CNN} against SNR.

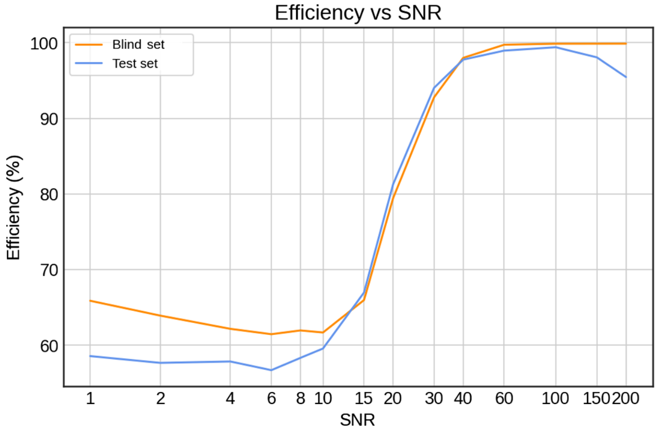

FIG. 15. η_{CNN} as a function of SNR computed during the testing process for $\{w, N, \theta^{*}\} = \{2, 30000, 65\%\}$.

For low SNR, the difference in efficiency η_{CNN} in the two case, blind set and test set, is around 10%, while for SNR >15 we obtain the same efficiency. These final results assess the robustness of this method to detect CCSN signals embedded in the real detector noise.

VII. DISCUSSION

The search of CCSN signals is carried by a software algorithm whose architecture trains 98997 parameters, significantly more than the 3210 parameters taken into account in Ref. [14]. This implies an increase of the network complexity by a factor of 30. We trained the Mini Inception-Resnet using of about 26 000 images corresponding to spectrograms of phenomenological waveforms injected in real noise of the three detector network LIGO-Virgo during the second observation run, and similar number of images without signals. We used the curriculum learning with decreasing value of the SNR for the training. The significant differences with Ref. [14] are

(i) the increase of the training images by a factor ∼10,
(ii) the extended variability of the injected waveforms, to mimic the behavior of the results from the CCSN numerical simulations,
(iii) the novel waveform parametrization for the frequency evolution,
(iv) the use of real detector noise instead of Gaussian one,
(v) images are not anymore built by applying a SNR threshold by cWB.

First, to compare the efficiency of this new method with previous results, we run the Mini Inception-Resnet network with the same setup as in Ref. [14]. The validation step shows that, with the appropriate choice of parameters ($\theta^{*} = 65\%$ and $w = 2$) we minimize the FAR_{CNN} toward ∼0% almost maintaining the same efficiency η_{CNN} for the range of SNR = [15, 20].

Then, we tested this method by injecting signals in the noise data of the LIGO-Virgo network taken during the second observation run. We have applied this analysis method for detecting two classes of signals. The first one is a blind set composed of the same phenomenological templates having the same analytical structure of those signals of the training set. The second one is based on 3D realistic numerical CCSNe simulations available in the literature.

In the validation process, carried on using the dedicated 25% of the training set, where signals are uniformly distributed in distance between 0.2 and 15 kpc, we obtain about 80% efficiency with a false alarm rate of about 5% for SNR = 16, see Fig. 10.

Applying the same method trained with phenomenological templates to the case of realistic GW signals from 3D numerical simulations (test set) we still obtain a reliable performance. Overall, when compared to the case of the blind set, the efficiency at SNR > 15, is very similar while at lower SNR we observe a reduction of less than 10%, see Fig. 14. The satisfying agreement is an indication that our phenomenological template generator is mimicking the main features observed in realistic CCSN and, therefore, it supports the choice of this kind of templates to train CNNs. Moreover, the high performance of the network for both testing datasets implies a high generalization ability, meaning that the network is not prone to overfitting. However, the decrease in efficiency at low SNR could be an indication that some of the features of CCSN are not perfectly captured by the templates, for example, the variability of the waveform amplitude for the duration of the signal (we consider that is in average constant) or the presence of low frequency components associated with SASI. Future work could incorporate these two features to improve the performance of the search method.

One of the advantages of the newly developed phenomenological templates is that they contain information about the distance to the source, which allows us to study the performance of the blind set with respect to the distance and to compare directly with the results of the test set. With respect to the distance, the efficiency shows in general a quick drop at ∼2 kpc followed by a gentle decline, falling down to about 60% at 15 kpc. This contrasts very much with the behavior with respect to SNR that shows a step decline at $\mathrm{SNR} \sim 15$. The reason for this difference is that, at a given distance, there is some variability in the amplitude of the possible waveforms, which tends to smooth out the results over a range of distances. We expect that at larger distances the efficiency will keep decreasing towards zero, but we did not see this effect within the limited set of distances used in this work. The performance with the *realistic* test set is somewhat worse than with the blind set, but the difference in efficiency is never larger than 10%, in agreement with the results obtained as a function of SNR.

We note that this results have been obtained using realistic waveforms from 3D models, which are, in general, about a factor 5 weaker than those of 2D simulations. It is also important to notice that we have used real O2 noise, so the results are expected to be better for the current detector configuration, which recently ended O3, and will improve further once the final sensitivity of LIGO, Virgo, and KAGRA detectors will be achieved. These two factors make it difficult to compare our results with those obtained in other papers using injections based on 2D simulations, simulated Gaussian noise and/or ultimate detector sensitivity [15–17].

In fact, for the case of neutrino driven explosions in Ref. [15] they use a set of waveforms from 55 numerical simulations (mixed 2D and 3D) to perform about 10^5 injections with random orientations in the sky in the range 0.2–200 kpc. Using a LIGO-Virgo-KAGRA network with optimal sensitivity, they obtain an efficiency of 50% at 4 kpc with false alarm probability of 0.1%. These results are similar to our work, however, it is difficult to do a closer comparison since they are using an interferometer network with ultimate sensitivity.

The work of Ref. [16] focused on using genetic programming algorithms to improve the significance of a single interferometer detection. For that purpose they trained the algorithm making injections of CCSN waveforms in real detector noise from the LIGO-Virgo first observing run (O1). For the case of neutrino driven explosions the algorithm is trained using waveforms from 2D and 3D CCSN simulations (8 in total) injected at different locations in the sky and distances in the range 1–7.5 kpc (about 15 000 injections in total). Similarly to Ref. [14], they employed the cWB pipeline. For waveforms from 3D simulations (not the same as ours) they get an efficiency of 86% at 3.16 kpc with 12% of false negatives. Again, the results are in the bulk of our numbers but it is difficult to compare, since they are using a network with lower sensitivity than ours and the injections that are comparable to ours amount only to 4 different signals. Their results show that it is possible to have a detection with high significance (3σ) for signals with an SNR as low as 10. However, it should be noted that, in their case, the same waveforms were used for training and for testing.

Finally, Ref. [17] utilized a CNN trained using 5 waveforms from neutrino-driven CCSN 3D simulations injected in Gaussian noise considering the spectral sensitivity curve of Virgo during the third observing run (O3). Training was performed with about 25 000 random injections in the sky at distances between 0.01 and 10 kpc. To test the robustness of the method they also accounted for short duration detector noise transients, known as glitches, in simulated data. When using different waveforms for training and testing, they obtain an efficiency of ∼90% of all triggers with a ∼10% false alarms (all distances in the range). When using the same waveforms for testing and training they observe a drop in the efficiency, below 50%, for values of the SNR in the range 11–16, depending on the waveform.

Despite the differences with earlier works, overall our results seem consistent with other machine learning approaches. The drop of the efficiency at $\mathrm{SNR} \sim 10$–15 is common for all algorithms (except for Ref. [15] that does not show this metric), which makes one wonder if there is some intrinsic limitation of machine learning algorithms that prevents getting closer to $\mathrm{SNR} \sim 8$, the typical value for optimal template-matching algorithms. It could also be possible that more complex architecture or training sets with different pixel resolutions might improve the efficiency of this method. These are aspects that we would like to explore in the future.

VIII. CONCLUSION

We developed a new machine learning algorithm to further improve the detectability of a GW signal from CCSN, following the path traced in Ref. [14]. Regarding the applicability of our method for the GW detection, we have considered a detection threshold, $\theta^* = 65\%$, that results in a FAR of about 5% at SNR ~ 15 (or a FPR of $\sim 10\%$ at TPR $= 50\%$). These values could be appropriate for an observation with high confidence of an event in coincidence with a neutrino signal. In those cases the neutrino signal is expected to be bounded within 20 s during the initial SNEWS alert [52] and very likely well within 1 s in the detailed analysis of high sensitivity neutrino detectors such as Super-K [53]. If the method were to be used in all-sky nontriggered searches, the range of values of FAR needed to make a detection with high confidence could be achieved by using values of θ very close to 100%. The efficiency of the algorithm in this regime is something that could be explored in future work.

These results are very promising for future detections of GWs from CCSN, because the network allows us to observe more than half of the events within 15 kpc.

This work has multiple possible extensions. At present the entire data processing is rather fast: The training and validation phase, performed in the real detector noise, is done in 2 h and 21 min using a GPU *Nvidia* Quadro P5000, while predicting the test set takes 3 ms for each 2 s long image. Given that we take advantage of the Keras/TensorFlow framework, widely used within the machine learning community, it should be easy to increase the complexity of our current CNN or to incorporate the latest developments in machine learning algorithms, with a reasonable increase of the computational cost of the signal search. Furthermore, we could increase the number of classes to be able to detect other GW sources with the same architecture. In the future, the new algorithm presented here should be compared under realistic conditions with the methods currently in use within the LIGO-Virgo Collaboration to evaluate the real advantages of the method. In particular, CNNs have the advantage that, once the training phase is performed, they have a very low computational cost, which could provide an advantage in the design of new low-latency detection pipelines for CCSN.

ACKNOWLEDGMENTS

The authors would like to thank M. Razzano for the critical reading of the manuscript and for his constructive input. This research has made use of data, software, and/or web tools obtained from the Gravitational Wave Open Science Center [54], a service of LIGO Laboratory, the LIGO Scientific Collaboration and the Virgo Collaboration. LIGO is funded by the U.S. National Science Foundation. Virgo is funded, through the European Gravitational Observatory (EGO), by the French Centre National de Recherche Scientifique (CNRS), the Italian Istituto Nazionale della Fisica Nucleare (INFN) and the Dutch Nikhef, with contributions by institutions from Belgium, Germany, Greece, Hungary, Ireland, Japan, Monaco, Poland, Portugal, Spain. P. C. D. acknowledges support from the Grants PGC2018-095984-B-I00, PROMETEU/2019/071 and the *Ramon y Cajal* funding (RYC-2015-19074) supporting his research. M. L. is supported by the research program of the Netherlands Organisation for Scientific Research (NWO). In addition, I. D. P. and F. R. acknowledge support from the Amaldi Research Center funded by the MIUR program “Dipartimento di Eccellenza” (CUP:B81I18001170001), the Sapienza School for Advanced Studies (SSAS) and the support of the Sapienza Grant No. RM120172AEF49A82.

[1] K. Sato and H. Suzuki, Analysis of Neutrino Burst from the Supernova 1987a in the Large Magellanic Cloud, Phys. Rev. Lett. **58**, 2722 (1987).

[2] H. Th. Janka, Neutrino-driven explosions, arXiv:1702.08825.

[3] H. Th. Janka, in *Neutrino Emission from Supernovae* (Springer International Publishing, Cham, 2017), pp. 1575–1604.

[4] H. A. Bethe, Supernova mechanisms, Rev. Mod. Phys. **62**, 801 (1990).

[5] W. Li, J. Leaman, R. Chornock, A. V. Filippenko, D. Poznanski, M. Ganeshalingam, X. Wang, M. Modjaz, S. Jha, R. J. Foley, and N. Smith, Nearby supernova rates from the Lick Observatory Supernova Search—II. The observed luminosity functions and fractions of supernovae in a complete sample, Mon. Not. R. Astron. Soc. **412**, 1441 (2011).

[6] R. Chapman, N. R. Tanvir, R. S. Priddey, and A. J. Levan, How common are long gamma-ray bursts in the local Universe?, Mon. Not. R. Astron. Soc. **382**, L21 (2007).

[7] B. P. Abbott *et al.*, First targeted search for gravitational-wave bursts from core-collapse supernovae in data of first-generation laser interferometer detectors, Phys. Rev. D **94**, 102001 (2016).

[8] B. P. Abbott *et al.*, Optically targeted search for gravitational waves emitted by core-collapse supernovae during the first and second observing runs of Advanced LIGO and Advanced Virgo, Phys. Rev. D **101**, 084002 (2020).

[9] B. P. Abbott *et al.*, All-sky search for short gravitational-wave bursts in the first Advanced LIGO run, Phys. Rev. D **95**, 042003 (2017).

[10] B. P. Abbott *et al.*, All-sky search for short gravitational-wave bursts in the second Advanced LIGO and Advanced Virgo run, Phys. Rev. D **100**, 024017 (2019).

[11] S. Klimenko, G. Vedovato, M. Drago, F. Salemi, V. Tiwari, G. A. Prodi, C. Lazzaro, K. Ackley, S. Tiwari, C. F. Da Silva, and G. Mitselmakher, Method for detection and reconstruction of gravitational wave transients with networks of advanced detectors, Phys. Rev. D **93**, 042004 (2016).

[12] R. Lynch, S. Vitale, R. Essick, E. Katsavounidis, and F. Robinet, An information-theoretic approach to the gravitational-wave burst detection problem, Phys. Rev. D **95**, 104046 (2017).

[13] N. J. Cornish and T. B. Littenberg, BayesWave: Bayesian inference for gravitational wave bursts and instrument glitches, Classical Quantum Gravity **32**, 135012 (2015).

[14] P. Astone, P. Cerdá-Durán, I. Di Palma, M. Drago, F. Muciaccia, C. Palomba, and F. Ricci, A new method to observe gravitational waves emitted by core collapse supernovae, Phys. Rev. D **98**, 122002 (2018).

[15] M. L. Chan, I. S. Heng, and C. Messenger, Detection and classification of supernova gravitational waves signals: A deep learning approach, Phys. Rev. D **102**, 043022 (2020).

[16] M. Cavaglia, S. Gaudio, T. Hansen, K. Staats, M. Szczepanczyk, and M. Zanolin, Improving the background of gravitational-wave searches for core collapse supernovae: A machine learning approach, Mach. Learn. Sci. Technol. **1**, 015005 (2020).

[17] A. Iess, E. Cuoco, F. Morawski, and J. Powell, Core-collapse supernova gravitational-wave search and deep learning classification, arXiv:2001.00279.

[18] E. Cuoco *et al.*, Enhancing gravitational-wave science with machine learning, arXiv:2005.03745.

[19] M. C. Edwards, Classifying the equation of state from rotating core collapse gravitational waves with deep learning, Phys. Rev. D **103**, 024025 (2021).

[20] B. Müller, H.-T. Janka, and A. Marek, A new multi-dimensional general relativistic neutrino hydrodynamics code of core-collapse supernovae. iii. Gravitational wave signals from supernova explosion models, Astrophys. J. **766**, 43 (2013).

[21] A. Torres-Forné, P. Cerdá-Durán, M. Obergaulinger, B. Müller, and J. A. Font, Universal Relations for Gravitational-Wave Asteroseismology of Protoneutron Stars, Phys. Rev. Lett. **123**, 051102 (2019).

[22] H. Sotani and T. Takiwaki, Dimension dependence of numerical simulations on gravitational waves from proto-neutron stars, Phys. Rev. D **102**, 023028 (2020).

[23] M.-A. Bizouard, P. Maturana-Russel, A. Torres-Forné, M. Obergaulinger, P. Cerdá-Durán, N. Christensen, J. A. Font, and R. Meyer, Inference of proto-neutron star properties from gravitational-wave data in core-collapse supernovae, arXiv:2012.00846 [Phys. Rev. D (to be published)].

[24] J. W. Murphy, C. D. Ott, and A. Burrows, A model for gravitational wave emission from neutrino-driven core-collapse supernovae, Astrophys. J. **707**, 1173 (2009).

[25] A. Marek, H.-T. Janka, and E. Müller, Equation-of-state dependent features in shock-oscillation modulated neutrino and gravitational-wave signals from supernovae, Astron. Astrophys. **496**, 475 (2009).

[26] K. N. Yakunin, P. Marronetti, A. Mezzacappa, S. W. Bruenn, C.-T. Lee, M. A. Chertkow, W. R. Hix, J. M. Blondin, E. J. Lentz, O. E. B. Messer, and S. Yoshida, Gravitational waves from core collapse supernovae, Classical Quantum Gravity **27**, 194005 (2010).

[27] S. Scheidegger, R. Käppeli, S. C. Whitehouse, T. Fischer, and M. Liebendörfer, The influence of model parameters on the prediction of gravitational wave signals from stellar core collapse, Astron. Astrophys. **514**, A51 (2010).

[28] E. Müller, H.-Th. Janka, and A. Wongwathanarat, Parametrized 3d models of neutrino-driven supernova explosions—Neutrino emission asymmetries and gravitational-wave signals, Astron. Astrophys. **537**, A63 (2012).

[29] K. N. Yakunin, A. Mezzacappa, P. Marronetti, S. Yoshida, S. W. Bruenn, W. R. Hix, E. J. Lentz, O. E. Bronson Messer, J. A. Harris, E. Endeve, J. M. Blondin, and E. J. Lingerfelt, Gravitational wave signatures of *ab initio* two-dimensional core collapse supernova explosion models for 12–25 $M_\odot$ stars, Phys. Rev. D **92**, 084040 (2015).

[30] T. Kuroda, K. Kotake, K. Hayama, and T. Takiwaki, Correlated signatures of gravitational-wave and neutrino emission in three-dimensional general-relativistic core-collapse supernova simulations, Astrophys. J. **851**, 62 (2017).

[31] H. Andresen, B. Müller, E. Müller, and H.-Th. Janka, Gravitational wave signals from 3D neutrino hydrodynamics simulations of core-collapse supernovae, Mon. Not. R. Astron. Soc. **468**, 2032 (2017).

[32] M. Carrillo, M. Gracia-Linares, J. A. González, and F. S. Guzmán, Parameter estimates in binary black hole collisions using neural networks, Gen. Relativ. Gravit. **48**, 141 (2016).

[33] D. George and E. Huerta, Deep neural networks to enable real-time multimessenger astrophysics, Phys. Rev. D **97**, 02 (2018).

[34] P. G. Krastev, Real-time detection of gravitational waves from binary neutron stars using artificial neural networks, Phys. Lett. B **803**, 135330 2019.

[35] Y. Bengio, I. Goodfellowand A. Courville, *Deep Learning* (MIT Press, Cambridge, MA, 2016), http://www.deeplearningbook.org.

[36] C. Szegedy *et al.*, Going deeper with convolutions, in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, Boston, MA, USA, 2015), pp. 1–9, https://doi.org/10.1109/CVPR.2015.7298594.

[37] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, Rethinking the Inception Architecture for Computer Vision, arXiv:1512.00567.

[38] K. He *et al.*, *Deep Residual Learning for Image Recognition* (IEEE, Las Vegas, NV, USA, 2016), pp. 770–778, https://doi.org/10.1109/CVPR.2016.90.

[39] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, arXiv:1602.07261.

[40] F. Chollet *et al.*, Keras (2015), https://github.com/fchollet/keras.

[41] A. Martín *et al.*, TensorFlow: Large-scale machine learning on heterogeneous systems, 2015, arXiv:1603.04467, Software available from https://www.tensorflow.org/.

[42] D. P. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization* (2014), arXiv:1412.6980.

[43] R. Abbott *et al.*, GWTC-2: Compact binary coalescences observed by LIGO and Virgo during the first half of the third observing run (2020), arXiv:2010.14527.

[44] B. P. Abbott *et al.*, GWTC-1: A Gravitational-Wave Transient Catalog of Compact Binary Mergers Observed by LIGO and Virgo during the First and Second Observing Runs, Phys. Rev. X **9**, 031040 (2019).

[45] R. Abbott *et al.*, Open data from the first and second observing runs of Advanced LIGO and Advanced Virgo, SoftwareX **13**, 100658 (2021).

[46] B. P. Abbott *et al.*, Low-latency gravitational wave alerts for multi-messenger astronomy during the second Advanced LIGO and Virgo Observing Run, Astrophys. J. **875**, 161 (2019).

[47] D. Radice, V. Morozova, A. Burrows, D. Vartanyan, and H. Nagakura, Characterizing the gravitational wave signal from core-collapse supernovae, Astrophys. J. **876**, L9 (2019).

[48] J. Powell and B. Müller, Gravitational wave emission from 3D explosion models of core-collapse supernovae with low and normal explosion energies, Mon. Not. R. Astron. Soc. **487**, 1178 (2019).

[49] T. Kuroda, K. Kotake, and T. Takiwaki, A new gravitational-wave signature from standing accretion shock instability in supernovae, Astrophys. J. **829**, L14 (2016).

[50] E. P. O'Connor and S. M. Couch, Exploring fundamentally three-dimensional phenomena in high-fidelity simulations of core-collapse supernovae, Astrophys. J. **865**, 81 (2018).

[51] P. Roy, S. Ghosh, S. Bhattacharya, and U. Pal, Effects of degradations on deep neural network architectures (2018), arXiv:1807.10108.

[52] P. Antonioli, R. Tresch Fienberg, R. Fleurot, Y. Fukuda, W. Fulgione, A. Habig, J. Heise, A. B. McDonald, C. Mills, T. Namba, L. J. Robinson, K. Scholberg, M. Schwendener, R. W. Sinnott, B. Stacey, Y. Suzuki, R. Tafirout, C. Vigorito, B. Viren, C. Virtue, and A. Zichichi, SNEWS: The supernova early warning system, New J. Phys. **6**, 114 (2004).

[53] S. Fukuda *et al.* (Super-Kamiokande Collaboration), The Super-Kamiokande detector, Nucl. Instrum. Methods Phys. Res., Sect. A **501**, 418 (2003).

[54] https://www.gw-openscience.org/