Applying deep neural networks to the detection and space parameter estimation of compact binary coalescence with a network of gravitational wave detectors

Xilong Fan
Department of Physics and Astronomy, Hubei University of Education, Wuhan 430205, China

Jin Li*
Department of Physics, Chongqing University, Chongqing 401331, China

Xin Li
Department of Physics, Chongqing University, chongqing 401331, China

Yuanhong Zhong
College of communication, Chongqing University, Chongqing 400044, China

Junwei Cao
Research Institute of Information Technology, Tsinghua University, Beijing 100084, China
(Dated: November 6, 2018)

In this paper, we study an application of deep learning to the advanced LIGO and advanced Virgo coincident detection of gravitational waves (GWs) from compact binary star mergers. This deep learning method is an extension of the Deep Filtering method used by George and Huerta (2017) for multi-inputs of network detectors. Simulated coincident time series data sets in advanced LIGO and advanced Virgo detectors are analyzed for estimating source luminosity distance and sky location. As a classifier, our deep neural network (DNN) can effectively recognize the presence of GW signals when the optimal signal-to-noise ratio (SNR) of network detectors ≥ 9. As a predictor, it can also effectively estimate the corresponding source space parameters, including the luminosity distance D, right ascension α, and declination δ of the compact binary star mergers. When the SNR of the network detectors is greater than 8, their relative errors are all less than 23%. Our results demonstrate that Deep Filtering can process coincident GW time series inputs and perform effective classification and multiple space parameter estimation. Furthermore, we compare the results obtained from one, two, and three network detectors; these results reveal that a larger number of network detectors results in a better source location.

Keywords: deep neural networks; advanced LIGO and advanced Virgo coincident detection of gravitational waves; multiple space parameter estimation

PACS numbers: 04.30.Db; 07.05.Mh

I. INTRODUCTION

As the authority of gravitational wave (GW) detection, last year, the advanced Laser Interferometer Gravitational-Wave Observatory (advanced LIGO) and advanced Virgo performed the first three-detector detection of a GW signal from a binary neutron star coalescence [1]. This is the fifth direct detection of GWs from compact binary coalescences by advanced LIGO [2–6]. With the continuous improvement of advanced LIGO and advanced Virgo [7–10], numerous GW triggers are expected to appear in the time series. Since, the current data analysis of advanced LIGO and advanced Virgo is computationally expensive for detecting the matched GW signals in noisy data and for distinguishing signals from glitches, new methodologies for signal processing are required that can process data rapidly and with high accuracy. In recent years, the application of machine learning to GW detection has been widely proposed [11, 12, 14–16]. Machine learning enables a computer to use specific learning algorithms to program itself based on large supplies of data to solve particular problems [17]. Among them, a deep learning architecture that is usually in the form of deep neural networks (DNNs) simulates the learning process of a human brain. By processing input information and sharing it with relevant neurons with sufficient neural connections, deep learning enables autonomous learning from

*Electronic address: cqujinli1983@cqu.edu.cn (corresponding-author)

data, yielding simpler data analysis [18, 19]. Most importantly, its ability to directly process raw noisy time series has been demonstrated, which can enable real-time GW observation [12]. Deep learning is expected to be a popular approach for processing big data and analyzing problems in the field of astrophysics.

Traditional NNs composed of three layers—the input, hidden, and output layers—have a limited ability to solve problems. However, researchers have discovered that if the hidden layer is extended with additional layers, its solving ability can be greatly enhanced [20]. NNs with long, interconnected layers between the input and output layers are called deep neural networks (DNN). Depending on the properties of the task, many categories of DNNs can be used, such as a recurrent neural network (RNN) [12, 20, 21] for exhibiting temporal behavior in data with variable input lengths, and a convolutional neural network (CNN) [22–24] for extracting features from data. Because a CNN is much easier to train due to its local receptive fields and weight sharing, and its effectiveness in automatically recognizing input, we have selected CNN as an integral part of our DNN system. Comparing the DNN to the matched filtering, widely used in the advanced LIGO's and advanced Virgo's pipelines, indicates that the DNN can significantly quicken GW searches [12].

There are a variety of GW sources in the advanced LIGO's and advanced Virgo's frequency band, such as the mergers of compact binary stars, which have been researched intensively. In particular, binary neutron stars have proven to be associated with abundant electromagnetic counterparts [25, 26]. As in [12], we first train the DNN as a classifier to distinguish signal from noise. Second, the DNN structure is adjusted and trained as a predictor to estimate a source's space parameters, which can constrain the locations of compact binary stars and provide insight for future multi-messenger observations [27–31].

In this paper, we take into account the coincident detection of three detectors, H, L, and V (HLV) referring to the advanced LIGO-Hanford, advanced LIGO-Livingston, and advanced Virgo detectors, respectively, and perform multiple space parameters estimation directly from noisy time series inputs after our DNN is trained by simulated signals. In the first application of the Deep Filtering method to advanced LIGO and advanced Virgo GW detection, George and Huerta have detected GW150914 from noisy time series in one detector and effectively measured their masses [13]. We, then, study the DNN as a predictor to estimate the luminosity distance and sky location of binary black holes. This paper is organized as follows. In section II, we describe the theoretical model for building our training and test data sets. In section III, we introduce the principle of DNN algorithms and investigate the performance of our designed DNN as a classifier and predictor, respectively. Conclusions and remarks are presented in section IV.

II. OBTAINING TRAINING AND TEST DATA

The GW signal from a compact binary coalescence can be divided into three phases: the inspiral, merger, and ringdown of the final object. The waveform h_+, $h_\times$ of all the phases has been successfully simulated using the effective-one-body model. Herein, we adopt the "EOBNRv4" to generate training and test data sets through the PyCBC software package (https://ligo-cbc.github.io). The GW strain $h(t)$ of an interferometer is a linear combination of h_+ and $h_\times$:

$$h(t) = F^+(t)h_+(t) + F^\times(t)h_\times(t), \tag{1}$$

where, the antenna functions $F^+(t)$ and $F^\times(t)$ for a specific detector geometry are mainly related to the angles describing GW polarization ψ and source location at a celestial sphere frame coordinate (right ascension α, declination δ), and the positions and orientations of the detectors. The detailed expressions of these functions, as well as the specific latitude of the detector's location along with the orientation of each of the detector's arms with respect to the local geographical directions, can be found in [32, 33].

GW signals are submerged in Gaussian noise with the advanced LIGO's and advanced Virgo's noise power spectral density (PSD) at designated sensitivities, respectively [34, 35]. Fig. 1 shows one sample of a coincident GW signal in the three detectors. The prior distributions on α, δ, ψ and the orbital inclination ι in $h_+, h_\times$ should be uniformly distributed in their respective ranges. The priors on m_1, m_2 are assumed to be uniform in the range from 10 to 40 $M_\odot$, and the distribution of luminosity distance D are assumed to be uniform on the sky spherical surface from 10 to 4000 Mpc (see Fig. 2). In each time series, all of our simulated binary black hole mergers are included in a 2-second window with cutoff frequency $f_{\rm cutoff} = 40$Hz, and the arrival time of the GW at the Earth's center is selected from the uniform distribution on the interval [1.7, 1.75]s. For each detector, the time strain is

$$s_i(t_0 + \tau_i + t) = F_i^+(t)h_+(t) + F_i^\times(t)h_\times(t) + n_i(t_0 + \tau_i + t), \quad 0 < t < T, \tag{2}$$

where, t_0 is the time at which the GW reaches the origin of any fixed coordinate, τ_i is the time at which the GW propagates from the origin to ith detector, and T is the duration of the GW. The time delay and antenna functions for each detector are based on the space locations of the advanced LIGO and the advanced Virgo (cf. Table 1 of [36]).

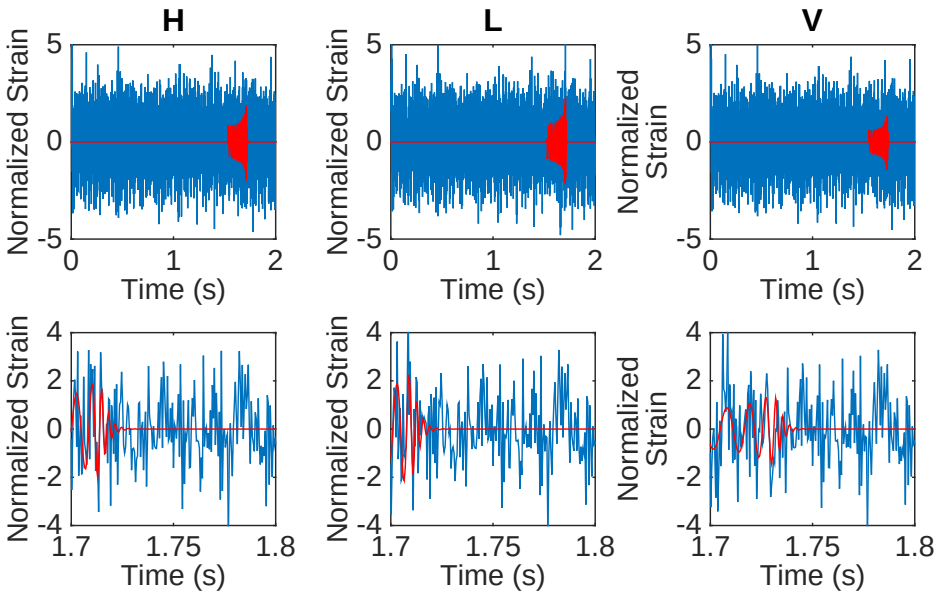

FIG. 1: Top panel: Blue curves are simulated time series in the advanced LIGO and advanced Virgo detectors, respectively, corresponding to the similar GW event and inputted to the DNN together. Bottom panel: Zoom plots corresponding to the top plots around injection time, which varies among detectors owing to their different locations [32, 36]. The sample frequency is $f_s = 2048$Hz.

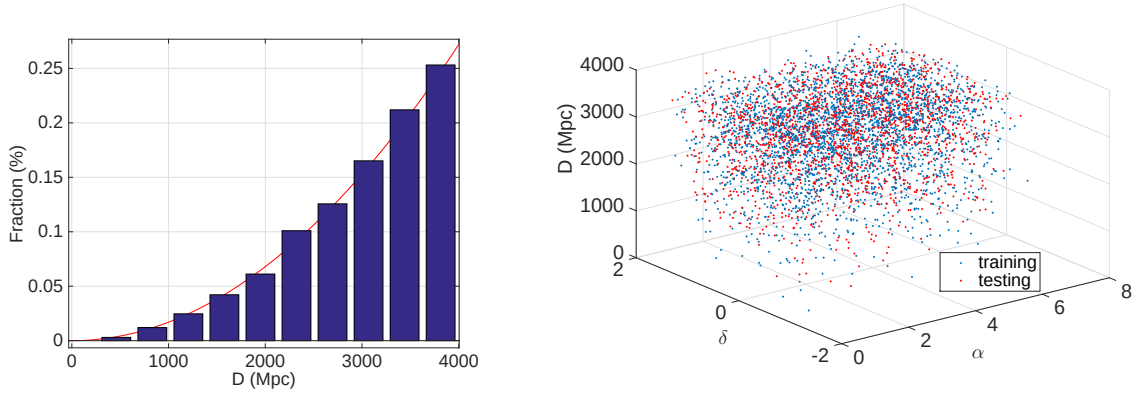

FIG. 2: Left panel: Distribution of the source luminosity distance D. Right panel: Source luminosity distance D and angles at the celestial sphere frame coordinate (α, δ) chosen for the training and test data sets. Here, the optimal SNRis9.

In this paper, we adopt the optimal SNR $\rho = 2[\int_0^\infty |\tilde{h}(f)|^2 df / S_h(f)]^{1/2}$ to describe the SNR [37], where, $S_h(f)$ is the noise PSD. In the following calculations, all of the SNRs are calculated from network detectors.

Splitting data into separate sets for training and testing is necessary for supervised learning. We randomly generate 3,000 training sets and 2,000 test sets with a given SNR (see right panel of Fig. 2). In addition to these, 2,000 validation sets are separately generated for each SNR in order to adjust the hyperparameters of the DNN. We, then, standardize all of the sets. Herein, we select a sampling frequency of 2048Hz (cf. Fig. 6 of [38]). For coincident observation, the data from H, L, and V should be inputted together, leading to a 3×4096 dimensional tensor for each input data set. In total, we generate 100,000 sets for training by adding different noise to 10,000 templates, and 2,000 sets for testing with each specific SNR.

III. DNN FOR GW DETECTION AND SPACE PARAMETER ESTIMATION

Structure of a DNN: An NN composed of many neurons can simulate the learning process of a human brain and

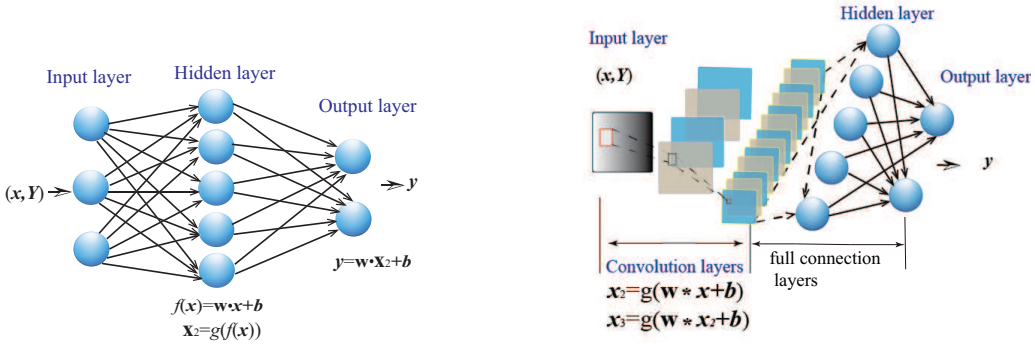

FIG. 3: Left panel: Traditional neural network (NN), in which the neurons (blue spheres) are connected to neurons in other layers by synapses (arrows). In the training process, the NN gets input $\vec{x}$ and $\vec{Y}$ and produces output $\vec{y}$. Right panel: Illustration of a DNN structure with convolution layers and fully connected layers, where $*$ is a convolution operator.

is one of the most popular methods in machine learning. The standard structure of an NN includes an input layer, a hidden layer, and an output layer (see left panel of Fig. 3). Information enters the input layer and is transferred to neurons in the next layer (i.e., the hidden layer) via synapses. Using vector $\vec{x}$ as an example, herein, the input information is expressed by linear combinations as $f(\vec{x}) = \vec{w} \cdot \vec{x} + b$, where $\vec{w}$ and b are weights and bias, respectively [12], and $\vec{Y}$ is the real answer to $\vec{x}$. The weights and bias are learned through training. In the hidden layer, the information is split; thus, it becomes simpler to analyze. To obtain a desired result, a regression analysis is applied on the synapses between the hidden and output layers; for instance, a logistic regression analysis is used for classification. Based on the features of the learning problem, certain activation functions $g(x)$, such as the logistic sigmoid, hyperbolic tan, and rectified linear unit (ReLU or Ramp) [12], are applied to the output of the hidden layer, i.e., $\vec{y} = g(f(\vec{x}))$. The output layer is a logistic regression layer containing two neurons as a classifier corresponding to two classes (i.e., true or false) and a number of neurons as a predictor corresponding to the number of estimated space parameters. Regarding the mathematical operations, some references provide detailed explanations [39]. Occasionally, there is more than one hidden layer. Because each neuron branches out to connect with all the neurons in the adjacent layer, such an NN is called a fully connected neural network and correspondingly the layers are called fully connected layers.

Commonly, some practical learning problems need more than one hidden layer. If a fully connected network is used, a large number of hyperparameters must be determined including the number of hidden layers and neurons in each layer, the choice of activation functions, the learning rate, and the iterations, and it is very difficult to train such a network. Convolutional neural network (CNNs) have been explored for the neocognitron [12, 22], a neural network that is not fully connected, since each neuron is connected to only a few neurons in the following layer, and that shares weights among the neurons. By performing a convolution calculation, a CNN can learn signal features by rotating and scaling signals. Due to their effective feature extraction and avoidance of redundant hyperparameter calculations, CNNs have obtained remarkable success in the fields of computer vision [40] and natural language processing [41]. Currently, the application of CNNs to time series data processing is being developed [12, 42]. In fact, DNNs, which combine fully connected layers with convolutional and modification layers, have been proposed to be more effective for both signal detection and parameter estimation from noisy time series data [12]. The right panel of Fig. 3 provides an illustration of the DNN structure. In the convolution layers, output from the previous layer, $\vec{x}_k$, is filtered by neurons through convolution operations and transferred to the output of this layer, $\vec{x}_{k+1}$, (i.e., $\vec{x}_{k+1} = g(\vec{w} * \vec{x}_k + b)$, where, $*$ is the convolution operator). Therefore, we plan to use this type of DNN to learn how to recognize GW signals from raw time series data sets and estimate the corresponding parameters of the GW sources. Determining the optimal hyperparameters of the DNN, including the number of convolutional and fully connected layers, size of pooling layers, number of neurons in each layer, learning rate, iterations, and activation functions, remains a challenging problem [20]. Currently, several approaches for determining optimal hyperparameters have been proposed, such as the randomized trial-and-error-based methods adopted in this paper(http://neuralnetworksanddeeplearning.com), Bayesian optimization [43], and genetic algorithms [44].

Training process: Once the structure of a DNN is determined, then, for each attempted hyperparameter, the DNN should be trained. As in human learning, the training process is the most important step for DNN learning, and the learning algorithm plays a key role in this process. The goal of training is to determine the optimal weights and bias of the DNN. Initially, the computer provides random weights, $\vec{w}$, and bias, b, to the DNN, which then undergoes several rounds of training using the training input data and obtains output $\vec{y}$. Finally, the optimal $\vec{w}$ and b are determined when the loss function $l(\vec{w}, b)$, which describes the difference between the DNN result $\vec{y}$ and

TABLE I: Modified version of the DNN in [12] used for classification; it is also used for prediction by replacing the 15^{th} layer with a ramp (ReLU) function (same method as [12]) and changing the size of layer 14 according to the dimension of predicted parameters. The number of neurons in the consecutive layers is 8, 16, and 32, respectively. The kernel sizes for the convolutional layers are 3×16, 1×16, 1×16, and 1 for all the pooling layers. The stride is set to 1 for all of the convolutional layers and 4 for all of the pooling layers. The dilation factor of the convolutional layers are set to 1, and the padding size of the pooling layer is zero. The function of pooling is max.

Input	Matrix(3×4096)
1 ReshapeLayer	3-tensor(size:1×3×4096)
2 ConvolutionLayer	3-tensor(size:8×1×4081)
3 PoolingLayer	3-tensor(size:8×1×1021)
4 Ramp	3-tensor(size:8×1×1021)
5 ConvolutionLayer	3-tensor(size:16×1×1006)
6 PoolingLayer	3-tensor(size:16×1×252)
7 Ramp	3-tensor(size:16×1×252)
8 ConvolutionLayer	3-tensor(size:32×1×237)
9 PoolingLayer	3-tensor(size:32×1×60)
10 Ramp	3-tensor(size:32×1×60)
11 FlattenLayer	vector(size: 1920)
12 LinearLayer	vector(size: 64)
13 Ramp	vector(size: 64)
14 LinearLayer	vector(size: 2)
15 SoftmaxLayer	vector(size: 2)
Output	class

the real answer $\vec{Y}$, reaches the minimum value. The loss function differs for classification and parameter estimation, which is discussed further in the following subsections. Applying each trained DNN to the validation sets, the optimal hyperparameters can be determined according to accuracy. Our optimal hyperparameters are illustrated in Table I.

In this paper, we adopt the ADAM method, which uses stochastic gradient descent with an adaptive rate, to train the DNN [45]. Gradient descent with an adaptive rate is invariant to the diagonal rescaling of the gradients of the loss function. When the local minimum is reached, the first-order partial differential equations equals to zero: $\partial l/\partial \vec{w} = \partial l/\partial b = 0$. This applies only to neurons from the input layer to the output layer; for adjusting the weights of neurons in the preceding layers, a back-propagation algorithm should be used [46]. From the updated weights in the output layer and by differentiating the activation function, back-propagation can be used to determine the updated weights in all preceding layers [47]. On the Mathematica 11.1 platform, we use the $NetChain$ command to design the DNN, whose hyperparameters are selected by random trials. Then, for each designed DNN, we apply the $NetInitialize$ command to initialize the network with a random distribution of weights and bias before training it.

A. DNN as a classifier

The main task of the DNN as a classifier is to detect GW signals in numerous time series data sets. The aim of the classifier is to correctly distinguish the class corresponding to $\vec{x}$ with maximum likelihood; therefore, the loss function is defined as the negative mean log-likelihood:

$$l(N, \vec{Y}, \vec{x}) = -\frac{1}{N}\left[\sum_{i=1}^{N} \log(P(\vec{Y} = \vec{y}|x_i))\right], \tag{3}$$

where, x_i represents the i^{th} training input data set and N is the size of the training data sets, which in our case is 100,000. $\vec{Y}$ and $\vec{y}$ are the real and predicted classes for x_i, respectively. $\log P(\vec{Y} = \vec{y}|x_i)$ represents the log probability of the predicted class of the DNN, $\vec{y}$, is the real class $\vec{Y}$ for a given x_i.

The input of the classifier comprises several 3×4096 time series from three detectors. Each sample is recorded as three time series, which are simultaneously measured by three separate detectors, H, L, and V. Based on the structure in [12], we have made some modifications. First, the three time series inputs from H, L, and V are joined into a 1×12288 matrix using a reshape layer, since the dimension of each neuron can then be simplified without any loss in

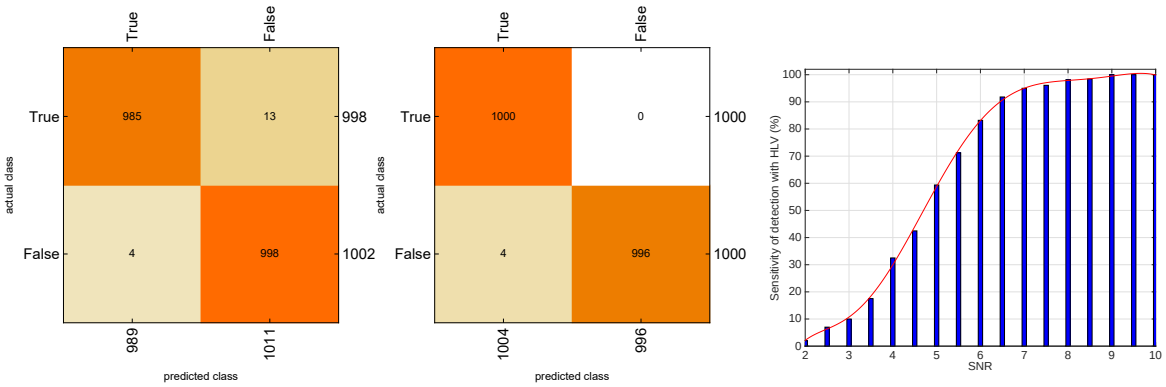

FIG. 4: Left panel: confusion matrix of a classifier on a test set with SNR = 8 (sensitivity is 98.70%). Middle panel: confusion matrix of a classifier on a test set with SNR = 9 (sensitivity is 100% for all signals with a higher SNR). Right panel: sensitivity of detection varying with the SNR, where, the false alarm rate = 0.4%.

accuracy in feature extraction. Second, specific values for kernel size, stride, dilation, and other DNN hyperparameters are set. Third, the DNN is trained. Then, validation sets are used to check the effectiveness of the trained DNN. After selecting a wide variety of hyperparameters for the DNN, the structure of our classifier is finally determined (shown in Table I) based on the accuracy and operation time of the hyperparameters. Later on, we use NVIDIA GPU (Gtx1080ti, 11G ram) to implement the deep learning process.

After training for nearly nine hours, the classifier obtains 98.70% sensitivity of detection for SNR = 8 and 100% sensitivity for SNR $\geq$9, with a false alarm rate of 0.4%. The confusion matrices are shown in the left and middle panels of Fig. 4. The right panel of Fig. 4 illustrates the sensitivity varying with the SNR in the three network detectors, where, sensitivity increases for a higher SNR.

B. DNN as a predictor

With a structure similar to that of our classifier, the predictor uses a ReLU function instead of the final softmax layer [12] and a different output size. When trained with a template bank of expected signals, a DNN is able to estimate multiple parameters directly from noisy time series data [12]. Herein, we focus on the parameters D, α, and δ for multiple parameter estimation, which directly provide the location of sources. With respect to the DNN as a predictor, we also compare its performance on one, two, and three network detectors, which can reveal meaningful insights into the coincident observation of GWs.

For parameter estimation, the loss function is generally the mean squared error:

$$l(N,\vec{Y},\vec{y}) = \frac{1}{N}\left(\sum_{i=1}^{N}(Y_i - y_i)\right)^2, \tag{4}$$

where, Y_i and y_i are the real and predicted values of parameters D_i, α_i, δ_i, respectively, corresponding to the i^{th} training data set, and N is the number of training data sets.

Results: Our predictor is capable of estimating multiple parameters from simulated time series data in network detectors. By comparing the estimation errors for one, two, and three network detectors (Fig. 5), we have found that luminosity distance errors $\epsilon_D = y_D - Y_D$ decrease for a higher number of network detectors. In addition, the areas of angular uncertainty of celestial coordinates are constrained from several hundred deg^2 for one detector to 100 deg^2 for three network detectors. Fig. 5 (e) indicates that the gradient direction of the declination error, ϵ_δ, on the right ascension error, ϵ_α, tends to be vertical, which is in accordance with existing error ellipses of angular parameters of all-sky maps [48]. Furthermore, when SNR $\geq$ 8, the relative errors of D,α,and δ in a multi-parameter predictor for advanced LIGO and advanced Virgo are all less than 23% for three network detectors.

IV. CONCLUSION

We have investigated the robustness of DNNs in coincident GW searches and multiple space parameter estimation in advanced LIGO and advanced Virgo, and have also shown that Deep Filtering can be extended to the network

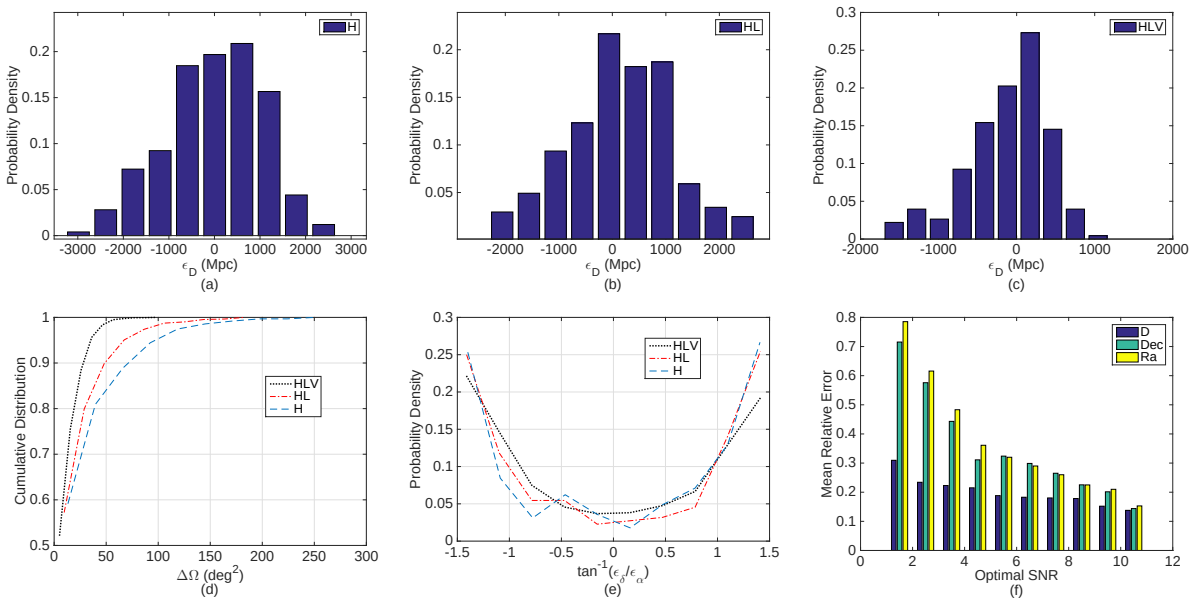

FIG. 5: (a), (b), (c) Distribution of errors $\epsilon_D = y_D - Y_D$ in a multi-parameter predictor for advanced H, HL, and HLV, respectively. (d) Cumulative distribution of areas of angular parameter (α, δ) uncertainty. (e) Distribution probability of the gradient angle for ϵ_δ on ϵ_α. Here, SNR = 8. (f) Mean relative errors of D, α, δ in the multi-parameter estimation varying with optimal SNR.

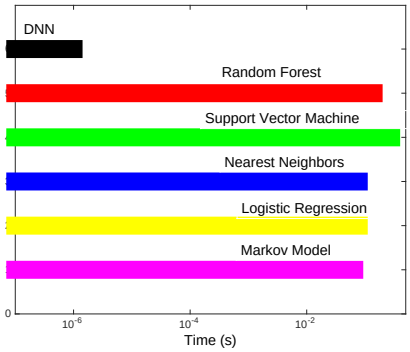

FIG. 6: Data processing times for classification and prediction by our DNN and by other machine learning methods. Note: times refer to the processing of one test set by our DNN (using GPU:NVIDIA GPU (Gtx1080ti, 11G ram)) and the processing of one test set by other machine learning methods (using CPU: 2.7 GHz Intel Core i5)

detection of binary black hole mergers. Based on the features of a compact binary star's GW waveform, 100,000 theoretical inspiral-merger-ringdown signals are simulated to train our DNN. The training process takes approximately nine hours, after which the instant classification or prediction with a specific SNR is achieved (shown in. Fig.6). From these results, we can determine the fundamental relation between the performance of the DNN and the optimal SNR of the data sets. This is meaningful for broadening the scope of current GW searches in advanced LIGO and advanced Virgo and for future real-time multi-messenger astrophysics.

For the above DNN structure, we have discussed signal-to-noise classification and space parameter estimation in one, two, and three detectors. The signal-to-noise classification indicates that the DNN can correctly classify the GW signals of compact binary stars with 100% detection sensitivity for time series with a SNR ≥ 9. From the parameter estimation, the DNN is able to estimate space parameters simultaneously. Our results also confirm that advanced LIGO and advanced Virgo can better estimate luminosity distance and sky location with a greater number of network detectors. Moreover, when a deep learning algorithm is used in advanced LIGO's and advanced Virgo's pipeline, there will be powerful instruments and equipment such as FPGAs for large data processing, and the corresponding sensitivity will be greatly improved. At present, our work mainly provides inspiration for deep learning in the network

detection and source location of advanced LIGO and advanced Virgo.

Furthermore, the main advantage of adding a DNN to GW searches is that retraining a DNN is time-saving once the DNN is trained well at a given PSD of advanced LIGO and advanced Virgo [12]. Using a DNN facilitates real-time coincident GW detection, since a trained DNN is very efficient in performing classification and prediction. Based on the instant time detection and space parameter estimation of binary black hole mergers, the real-time observation of binary neutron star coalescence will likely be determined in the near future, which is highly important for the search for electromagnetic counterparts [49–51].

Acknowledgements

We would like to express our great gratitude to Dr. George and Huerta for their inspiration and patient help. This work is supported by National Natural Science Foundation of China No.11873001, No. 11633001, No.11673008, No.61501069, the Natural Science Foundation of Chongqing No. cstc2018jcyjAX0767, the Strategic Priority Program of the Chinese Academy of Sciences (Grant No. XDB 23040100), Newton International Fellowship Alumni Follow-on Funding and the Fundamental Research Funds for the Central Universities project No. 106112017CDJXFLX0014, 106112016CDJXY300002. Our work is also supported by Chinese State Scholarship Fund and Newton International Fellowship Alumni Follow on Funding.

[1] B. P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), Phys. Rev. Lett. 119, 161101 (2017)
[2] B. P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), Phys. Rev. Lett. 119, 141101 (2017).
[3] B. P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), Phys. Rev. Lett. 116, 061102 (2016).
[4] B. P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), Phys. Rev. Lett. 116, 241103 (2016).
[5] B. P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), Phys. Rev. Lett., 118, 221101 (2017).
[6] J. Li, X. L. Fan, Sci. China-Phys. Mech. Astron. 60, 120431 (2017).
[7] B. P. Abbott et al. (LIGO Scientific Collaboration), Class. Quantum Grav. 32 074001 (2015).
[8] F. Acernese et al. (Virgo Collaboration), Class. Quantum Grav. 32, 024001 (2015).
[9] D. Blair, et. al., Sci. China-Phys. Mech. Astron. 58, 120402 (2015).
[10] D. Blair, et. al., Sci. China-Phys. Mech. Astron. 58, 120405 (2015).
[11] R. Biswas et al., Phys. Rev. D 88, 062003 (2013).
[12] D. George and E. A. Huerta, Phys. Rev. D 97, 044039 (2018) arXiv:1701.00008.
[13] D. George and E. A. Huerta, Phys. Lett. B, 778 (2018) arXiv:1711.03121.
[14] A. Mytidis et al., arXiv:1508.02064 (2015).
[15] T. F. Alejandro et al., Phys. Rev. D 94, 124040 (2016).
[16] K. A. Hodge, The Search for Gravitational Waves from the Coalescence of Black Hole Binary Systems in Data from the LIGO and Virgo Detectors Or: A Dark Walk through a Random Forest, PhD thesis, California Institute of Technology, Pasadena, California, USA (2014).
[17] Jaime G Carbonell, Ryszard S Michalski, and Tom M Mitchell. An overview of machine learning. In Machine learning, pages 3−23. Springer Berlin Heidelberg, Heidelberg, Germany (1983).
[18] W. S. McCulloch and W. Pitts, Bull. Math. Biophys., 5(4):115−133 (1943).
[19] G. A Carpenter, Neural networks, 2(4):243−257 (1989).
[20] J. Schmidhuber, Neural Networks 61 85 (2015).
[21] J. Wang, SIAM J. Sci. Comput., 18(5), 1479−1493 (2006).
[22] K. Fukushima, Biological Cybernetics 36, 193 (1980).
[23] Y. LeCun and Y. Bengio (MIT Press, Cambridge, MA, USA) Chap. Convolutional Networks for Images, Speech, and Time Series, pp. 255−258 (1998).
[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, in Advances in Neural Information Processing Systems 25, edited by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Curran Associates, Inc.) pp. 1097−1105 (2012).
[25] L. Li and B. Paczyński, ApJ, 507: 59 (1998).
[26] B. D. Metzger and E. Berger, ApJ, 746: 48 (2012).
[27] B.P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), Astrophys. J. Lett. 848, L12 (2017).
[28] D. A. Coulter et al., GCN 21529, 1 (2017).
[29] D.A. Coulter et al., Science, in press (2017), DOI: 10.1126/science.aap9811.
[30] LIGO Scientific Collaboration and Virgo Collaboration, GCN 21513, 1 (2017).
[31] X. L. Fan, Sci. China-Phys. Mech. Astron. 59, 640001 (2016).
[32] C. Cutler, E. E. Flanagan, Phys. Rev. D 49, 2658 (1994).
[33] P. Jaranowski et al., Phys. Rev. D 58, 063001 (1998). arXiv:gr-qc/9804014v1.
[34] D. Shoemaker, Advanced LIGO anticipated sensitivity curves-LIGO Document (2010).
[35] F. Acernese et al., Class. Quantum Grav. 32 024001 (2015). arXiv:1408.3978.

[36] Bernard F Schutz, Classical Quant. Grav., 28, 12 (2011).
[37] B.S. Sathyaprakash, B.F. Schutz, Living Rev. Relativity 12, 2 (2009). arXiv: 0903.0338v1.
[38] B. P. Abbott et al. (LIGO Scientific Collaboration and Virgo Collaboration), Phys. Rev. Lett. 116, 241102 (2016).
[39] K. O'Shea, R. Nash, arXiv:1511.08458 (2015).
[40] D. Mishkin et al., Comput. Vis. Image Und. 161, 11-19 (2017).
[41] R. Collober et al., J. Mach. Learn. Res.,12, 2493-2537 (2011).
[42] Y. Zheng et al., Time Series Classification Using Multi-Channels Deep Convolutional Neural Networks, Web-Age Information Management pp 298−310, Springer, Cham, Switzerland (2014).
[43] J. Snoek, H. Larochelle, and R. P. Adams, in Advances in Neural Information Processing Systems 25, edited by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Curran Associates, Inc.) pp. 2951−2959 (2012).
[44] L.X. Xie, A. Yuille, arXiv:1703.01513v1(2017).
[45] Diederik P. Kingma, Jimmy Ba, arXiv:1412.6980(2017).
[46] D. E. Rumelhart et al., Learning representations by back-propagating errors, Cognitive modeling 5(3):1 (1988).
[47] J. Li et al., Brief Introduction of Back Propagation (BP) Neural Network Algorithm and Its Improvement, Advances in Computer Science and Information Engineering, pp 553−558 (2012).
[48] L. Q. Wen, Y. B Chen, Geometrical expression for the angular resolution of a network of gravitational-wave detectors, Phys. Rev. D 81, 082001 (2010).
[49] Z. J. Cao, Sci. China-Phys. Mech. Astron. 59, 110431 (2016).
[50] H. Gao, Sci. China-Phys. Mech. Astron. 61, 059531 (2018).
[51] T. P. Li, et al., Sci. China-Phys. Mech. Astron. 61, 031011 (2018).