PROJECT 3: DETECTION OF CORE-COLLAPSE SUPERNOVAE IN TIME DOMAIN

Sjoerd Seinhorst, Jaffar Husain, Oscar Klock, and Helena Kirchner Sala
Computational Aspects of Machine Learning
Utrecht University
(Dated: June 29, 2022)

Abstract

The mechanism of Core Collapse Supernovae (CCSN) explosions is still under intense study in the astrophysical community. The observation of neutrino and gravitational wave emission from CCSN would give direct information about it. However, its detection faces big challenges, starting with the generation of numerical simulations of this physical process, that make it impossible to detect these sources with the current state-of-the-art. In this project, we propose the implementation of Machine Learning (ML) techniques that take advantage of the peculiarities of the CCSN GW signal to increase the detection capability. In particular, a convolutional neural network (CNN) trained with a set of phenomenological waveforms designed to mimic numerical simulations is presented, with the goal of detecting CCSN phenomenological waveforms in time or frequency domain to save some computation time

I. INTRODUCTION

In recent years, the advanced generation of gravitational wave detectors have allowed the scientific community to detect gravitational waves (GWs) from different astrophysical processes, one of them being the core collapse supernovae (CCSN). The detection of GW is an essential step in the development of multi-messenger astronomy, a recently emerged discipline that provides unique and valuable insights into the properties and processes of the universe. Until the mid 20th century, the only information astronomers could use to study the distant universe was provided by messengers of the electromagnetic force. However, in the last few decades the messengers of the other three forces, namely gravitational waves (GWs), neutrinos, and cosmic rays (CRs), began to be used [1].

These new non-photonic messengers are generally more challenging to detect, but they are associated with extremely high mass or high energy density configurations such as stellar explosions, neutron stars or black holes. The association with this astrophysical phenomena means that the interpretation of multi-messenger observations can have implications for the theories of fundamental physics. Some of the achievements of multi-messenger astronomy include the measurement of the spectrum of ultra- high energy cosmic rays out to the highest observable energies, the discovery of the diffuse high energy neutrino background and the first direct detections of gravitational waves and the use of gravitational waves to characterize merging black holes and neutron stars in strong-field gravity [1].

In this paper, we focus on the detection of GW coming from Core Collapse Supernovae explosions, whose mechanism is still unknown and they are under intense study [1].

A. CCSN process

A CCSN is a powerful and luminous stellar explosion that occurs during the last evolutionary stages of a massive star. In the first stages of a massive star life hydrogen is converted into helium. Fusion reactions in the core decrease resulting in the release of less energy, and gravity causes the core to contract, raising the temperature to the point where helium fusion can begin. The same happens in this stage, leading to the burning of carbon. After the carbon burning stage comes the neon burning, oxygen burning and silicon burning stages, each lasting shorter. Finally, iron is produced, and since it is the most stable element, in order to fuse into heavier elements it must absorb energy instead of realizing it. Therefore, the formation of iron in the core concludes fusion processes and the star begins to collapse in on itself since there is no energy to support it against gravity [2] [3].

During the collapse the temperature in the core increases and protons and electrons combine to form neutrons releasing vast quantities of neutrinos carrying large amounts of energy, causing the core to cool and contract even further. When the density of the inner core exceeds the density at which neutrons and protons are packed together inside atomic nuclei, it becomes incompressible and rebounds at supernuclear densities, creating a huge shock wave. Under the extreme densities of the collapsing core, a small fraction of neutrinos become trapped behind the expanding shock wave and its energy increases the temperature and pressure behind the shock wave, which in turn gives it strength as it moves out through the star initiating the SN explosion. This is the so-called neutrino driven mechanism and it is the dominant theory to explain CCSN explosions in slowly rotating progenitors [2].

The observation of neutrino and gravitational wave emission from CCSN would give direct information about the inner mechanism of the explosion. However, the amplitude of the gravitational waves impinging on a detector on the Earth is extremely faint despite the amount of energy released in this phenomenon, making it very hard to detect. Moreover, due to the complexity and stochasticity of the waveform, generating numerical simulations of this process is challenging and computationally intensive. Therefore, with the current state-of-art technologies it is almost impossible to detect these sources.

Previous work has focused on finding new ways to improve the detection statistics. Some research lines 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, while others have started making use of machine learning techniques to take advantage of the peculiarities of the CCSN GW signal to increase our detection capability. The dominant feature used to detect the signal present in the GW spectrum is the monotonic rise of the GW signal in the time-frequency plane due to the g-mode excitation [2].

In this project, we follow the line of implementing machine learning algorithms to detect CCSN GW, specifically focusing on the implementation of Convolutional Neural Networks (CNN). Before diving deep into the machine learning techniques used for the detection of supernovae, we will give a brief introductory explanation on what machine learning is and the current state of the art in the use of these techniques for the study of CCSN.

B. Machine learning and state of the art

Machine learning is a branch of artificial intelligence which focuses on the use of data and algorithms to imitate the way that humans learn. It first introduced in 1959 as the "ability of computers to learn without being explicitly programmed". Years later, a mathematical and relational definition was stated to describe this concept: "A computer program is said to learn from experience E concerning some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E".
Algorithms used in machine learning build models that use what is called "training data" to make predictions or decisions without being explicitly programmed to do so. The main objective is to give the learning machine the ability to perform accurately on new examples after having experienced a learning data set [4][5].
Depending on the nature of the learning signal or response available to a learning system, machine learning approaches are divided into three categories. "Supervised learning", where the computer is given example inputs and their desired outputs and it derives the general rules that maps inputs to outputs from these examples, "Unsupervised learning", where the computer learns from plain examples without any associated response, leaving the algorithm to determine the data structure and patterns on its own, or "Reinforcement learning", where the computer is also presented with examples that lack an associated response but that are accompanied with positive or negative feedback according to the solution the algorithm proposes [4][5].

As Ref. [6] describes, machine Learning (ML) algorithms are a promising tool that has recently started to be used for tackling the issues in the detection and signal characterization of CCSN. Due to the capability of machine learning algorithms to recognise patterns in data, machine learning techniques could make the search of CCSN GW signals more sensitive and robust.
Machine learning algorithms have been used to better different aspects of the CCSN search:

1. To improve the quality of LIGO-Virgo data characterizing and reducing the detector noise is essential to GW searches.

2. To improve the modeling of GW signals preding the GW waveforms in areas of the signal parameter space not covered by full numerical relativity.

3. To improve the sensitivity of GW searches where the exact signal morphology is unknown.

4. To do parameter estimation of GW signals and source population inference.

In Ref. [7] 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 this project we implemented an adaption of Google DeepMinds deep neural network: WaveNet. We trained the network on phenomenological waveforms designed to mimic numerical data simulations of GW signals to make the binary classification of GW signal or no GW signal.

C. WaveNet

WaveNet was created in 2016 with the main objective of generating more realistic sounding text to speech audio. The network was able to train on waveform data with tens of thousands of samples per second and at the time of publishing, WaveNets performance was unmatched. Besides being able to generate realistic sounding speech, it also showed promising results as a discriminative model, being able to discriminate between phonemes [8].

The architecture of WaveNet enables it to act as a series of wavelets of varying widths. The varying of the dilation rates serves as the varying widths of the wavelet forms and preserves both the time-domain and frequency-domain information. The authors stacked 1D convolutional layers, doubling the dilation rate at every layer. The dilation rates, in turn, enable the network to fit progressively wider receptive fields to the dataset starting with two time steps then four and so on; the first convolutional layer sees a window with two time steps at a time, while the next layer sees four time steps at a time and so on. Thus the WaveNet architecture mimics a series of wavelets with varying window sizes (receptive fields). The shorter receptive fields pick up higher frequencies while the longer receptive fields pick up the longer-term patterns of the signals. In the paper, the authors stacked three identical groups of 10-convolutional layers each with dilation rates of 1, 2, 4, ..., 512. The authors also used a causal convolution to preserve the sequence length throughout the training. Thus every convolutional layer would output a sequence of the same length as the original input sequences. An example of a four-layer dilated convolutional network is shown in Figure 1.

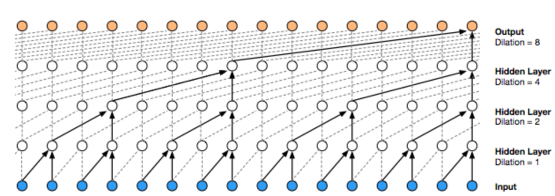

Figure 1: Visualization of a stack of dilated causal convolutional layers

II. METHODOLOGY

A. Data

The model inputs are computer generated and preprocessed parametric phenomenological time series of GWs produced by CC-

SNe and measured by the advanced LIGO–VIRGO detector network. First, phenomenological models are used to generate GWs as produced by CCSNe. These are then projected from the source location onto the three detectors (LIGO Hanford (H1), LIGO Livingston (L1) and VIRGO (V1)) and the measurement is modeled. These measured signals are injected into simulated characteristic noise patterns of the different detectors. An equal amount of waveforms are created consisting of only detector noise. Finally all waveforms are whitened. The GW generation, propagation and projection are all parameterized. We received the waveforms resulting from this process from [2].
The data was divided into subsets depending on the distance to the source D. Each subset contained 1080 samples of injections with varying parameter settings, mimicking GW signals injected in background noise, and another 1080 samples of only noise. Each sample contained waveforms from the three detectors. Figures 2 and 3 shows an example of a loud injected GW signal.

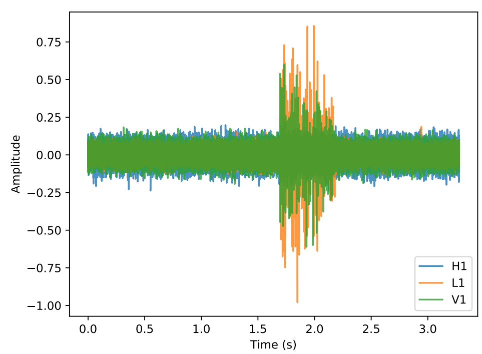

Figure 2: Injected GW signal in noise for all three detectors. The sudden increase in amplitude is clearly visible in this example, especially in the detectors; LIGO Livingston (L1) and Virgo (V1).

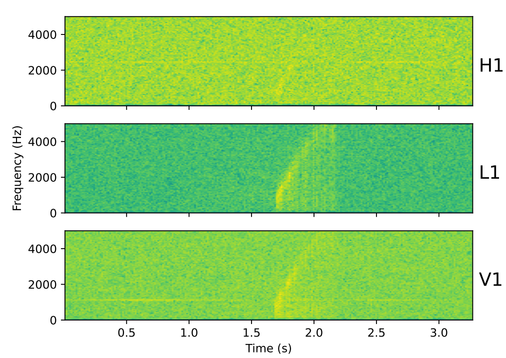

Figure 3: Spectrogram showing the frequencies of the same signal for each detector with brighter yellow indicating a higher amplitude of that frequency.

We randomly divided the data into a training set validation set and a blind set (80 %, 10 %, 10 %). All sets contained an equal number of samples for each class (injection and noise). We also used a test set consisting of 374 samples per class. Instead of applying curriculum learning, where the network is trained on the closer distances first with more distant signals added gradually over the training process, we trained it on combinations of distances. The combinations were (1 kpc and 5 kpc), (1 kpc and 10 kpc) or (1 kpc and 15 kpc). We found that we got better performance with this approach compared to using curriculum learning.

B. Practical procedure

We tried different ANNs and compared their performance in order to settle on which direction to pursue. The first networks were simple conventional CNNs, Inception-style networks and ResNet-style networks. [9][11][12][13][14]. Inception networks are characterized by multiple consecutive convolutional layers, where each layer contains multiple different receptive fields. ResNet-style networks contain residual connections, also called skip connections, that bridge one or more convolutional layers and are joined later in the network, commonly by addition. Another attempt was an implementation of a network used to detect low-magnitude earthquake signals called GroningenNet [10]. Inspired by the success of these networks—up to 85% validation accuracy on nearby signals—we turned our attention to WaveNet. WaveNet combines the multiple receptive fields of Inception and the skip connections of ResNet in an efficient design optimized for time series data.

C. WaveNet implementation

We use an architecture similar to [8] with three identical blocks of four layers of dilated convolutions where the dilation rates are increased by a factor of 2 at each layer. As described above, the dilated convolutions allow the network to access a wider history of the input sequences. The input sequences are three-dimensional: number of samples, length of each sample, and the number of channels. The kernels are of size two for the causal convolutional layers. In the last convolutional layer, we reduce the kernel size to one. A batch-normalization layer is added after the first convolutional layer and before the activation function to normalize the input sequences. Batch normalization enables the neural net to optimize the scale and mean of input sequences which, in turn, has been shown to reduce the vanishing gradient problem. Average pooling is used after the last convolutional layer to reduce the dimensions of the output maps, followed by a dense classification layer with one output and a sigmoid activation.

As in [8] we use the gated activation units, residual block, and skip connections. Gated activation units have been found to perform better than ReLU for modeling wave-like signals. Residual learning and skip connections are added to enable training deeper layers while minimizing the training error caused by the failure of the standard back-propagation to find optimal weights. Residual learning uses shortcut connections to skip one or more layers and get added unmodified to the output from the skipped layers. A residual connection is added after each dilated convolution. In our implementation of WaveNet, we use three blocks of layers and four layers in each block. Using a higher number of layers in each block would widen the receptive field of the network. We therefore would have liked to use more layers; however, we couldn't use a higher number of layers due to memory constraints. The architecture of the neural net is depicted below:

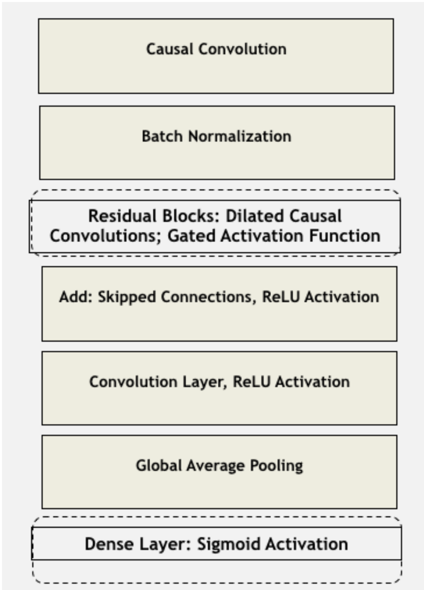

Figure 4: Network Architecture

III. RESULTS

In order to validate the strength of the neural architecture and identify its potential limitations, we trained and validated it on various combinations of the data as mentioned above. Before presenting these combinations, we will show the results for 1 kpc to set a benchmark. The same set of results are shown for all other combinations. 80 percent of each dataset was used for training and 10 percent for validation. The last 10 percent of each dataset was reserved for testing.

In the following table, the results obtained in every training are collected:

	Training accuracy	Validation accuracy
1 kpc	95 %	95 %
1–5 kpc	80 %	80 %
1–10 kpc	95 %	74%
1–15 kpc	70 %	70 %

A. 1 kpc

The model was trained with 20 epochs and we obtained 95 percent accuracy on the training and validation datasets. In the following figures we present the model training and validation accuracy and loss rates and the model predictions for the training dataset. The model predictions for the validation dataset are similar. In addition, we present samples of the model-generated probability signals and compare them to the original signals. The results confirm the wavelet-like structure of WaveNet; the neural net is able to identify the regions where the injections are located. The model-generated probability signals show distinct patterns in the regions where the original samples contain injections. In the 1 kpc case, the model-generated probability signals are distinct and heavily clustered. For higher kpc (5, 10, and 15) where the injections are fainter, the patterns are still distinct but much less heavily clustered. Figure 5 and 6 present the accuracy and loss rates of the training and validations respectively. Figure 7 presents the generated signals.

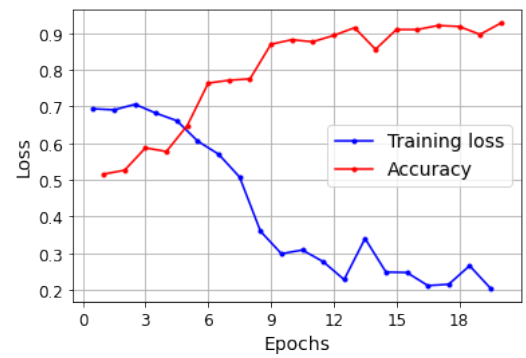

Figure 5: 1 kpc training dataset: accuracy and loss rates

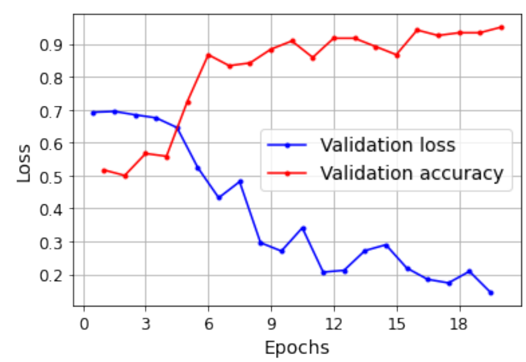

Figure 6: 1 kpc validation dataset: accuracy and loss rates

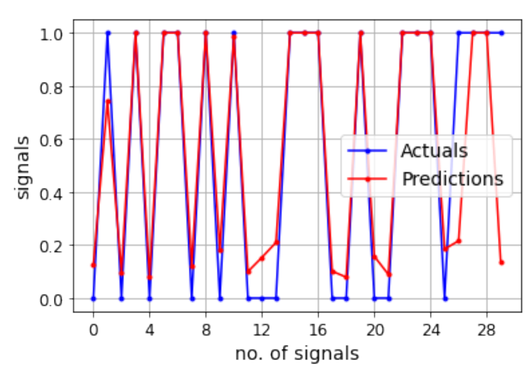

Figure 7: 1 kpc training dataset: samples of actual vs model-predicted results. Actual Injections are classified as 1 and Background as 0. Predictions are in the form of probabilities due to sigmoid activation.

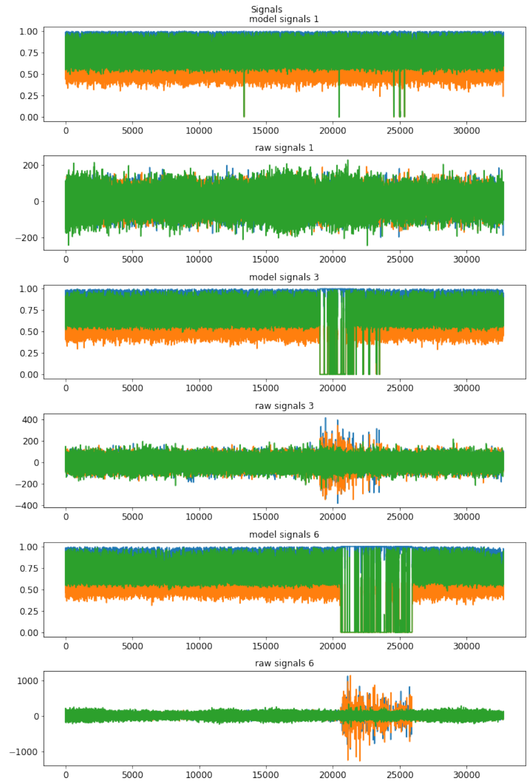

Figure 8: 1 kpc: 3-pairs of samples of model-generated results vs. original signals. All samples are injections. The model-generated signal probabilities show distinct patterns in the regions where the actual samples contain the injections

B. 1 kpc and 5 kpc

The model was trained with 30 epochs on the combined dataset containing both 1 kpc and 5 kpc datasets. We obtained around 80 percent accuracy on the training and validation datasets. The same set of results are presented in the Appendix as for 1 kpc.

C. 1 kpc and 10 kpc

The model was trained with 70 epochs on the combined dataset containing both 1 kpc and 10 kpc datasets. Using a higher number of epochs was designed to identify the points at which the model would start over- or under-fitting the dataset. We obtained around 95 percent accuracy on the training dataset and validation accuracy of around 74 percent. Therefore, while the training accuracy improved markedly with a higher number of epochs, the validation accuracy did not. The over/under fitting started after around 40 epochs. The same set of results are presented in the Appendix as for 1 kpc.

D. 1 kpc and 15 kpc

The model was trained with 50 epochs on the combined dataset containing both 1 kpc and 10 kpc data. We obtained around 70 percent accuracy on the training and validation datasets. Over/under fitting seems to start after about 40 epochs. The same set of results are presented in the Appendix as for 1 kpc.

E. Efficiency

The efficiency of the model is the same as its true positive rate, meaning its ability to correctly classify the signals. It is calculated as follows:

$$\eta_{\text{CNN}} = \frac{\text{correctly classified signals}}{\text{all the signals at CNN input}}. \tag{1}$$

The false alarm rate, or false discovery rate is given by

$$FAR_{\text{CNN}} = \frac{\text{misclassified noise}}{\text{all classified events}}. \tag{2}$$

This is the measure we can use to compare our results to previous research. To classify a signal, the model had to predict the probability of a sample being a signal to be greater than 0.5. Everything below would be classified as noise. Figure 9 shows the distributions of probabilities for all samples in the test set. In Figure 10, the efficiency of the blind set and the test set are presented.

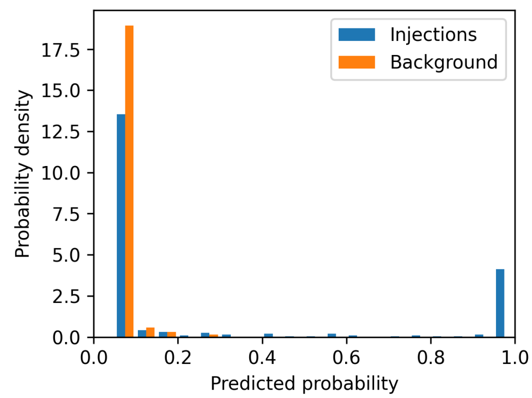

Figure 9: Probability distribution for all samples in the test set. Perfect classification would show all blue samples (signal) to be above 0.5 and all yellow samples (noise) below 0.5.

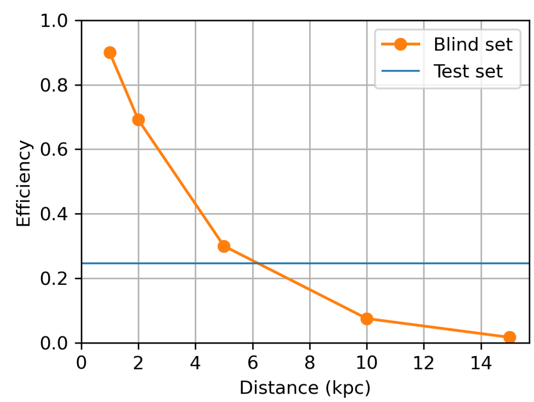

Figure 10: The curved line refers to the efficiency on the testing of the blind set, obtaining a 90 % for 1 kpc, 69.2 % for 2 kpc, 30 % for 5 kpc, and 7.5 % 10 kpc and 1.7 % for 15 kpc. The black horizontal line is the efficiency of the test set, where all data for different kpc are mixed. The efficiency obtained is 24.6 %

IV. DISCUSSION

The WaveNet architecture is a promising approach to classify GW signals on one dimensional data. We could achieve good results on training and validation data, however, our model is performing worse on test data than other models. On the blind set we reach an efficiency of 90 % on kpc 1, which is just below what has been reported in other research. Our models efficiency consistently decreases with higher kpc where previous models are decreasing with distance but stabilizes around 60 %. Compared to previous models, we used a different training strategy and considerably less training data. This approach was enough to get good results during training and validation but failed to deliver on the blind set and test set. However, we should note that due to memory issues which we consistently faced during training, we were only able to train the model on 1 kpc before testing it on the test set. We believe better results could be obtained with a more comprehensive training with all the datasets.

The primary strength of the WaveNet architecture, the one of concern to our project, probably lies in its ability to mimic a series of wavelets of varying widths. However, in this architecture, the WaveNet produces a generalized set of wavelet forms to analyze the datasets. That is to say, these wavelet forms are not specifically designed for our datasets. This suggests a natural way to improve the results by first producing specific wavelets for the datasets and then feeding these wavelets to WaveNet or a fully-connected feed-forward neural net.

We made the assumption that signals further away from detectors would be more difficult to classify correctly and the results support this assumption. However, there are misclassifications made for each distance, which means the accuracy is not only dependent on the distance. In order to improve the network, we propose to first make an analysis of which parameters contribute the most to the misclassified samples. This could give a more informed idea of what is difficult and what is not.

V. CONCLUSION

We implemented a WaveNet-style CNN and trained it to find CCSN-GW signals in noisy data. We achieved 90% efficiency and 95% accuracy on the closest signals of 1 kpc, down to 1.7% efficiency on distant 15 kpc signals. However, as noted earlier, memory constraints limited our ability to train on the full dataset which may have contributed to the lower efficiency results on more distant signals.

VI. REFERENCES

[1] Mészáros, P., Fox, D. B., Hanna, C., & Murase, K. (2019). Multi-messenger astrophysics. Nature Reviews Physics, 1(10), 585-599.

[2] López, M., Di Palma, I., Drago, M., Cerdá-Durán, P., & Ricci, F.(2021).Deep learning for core-collapse supernova detection. Physical Review D, 103(6), 063011.

[3] Core-collapse, Swinburn university of Technology `https://astronomy.swin.edu.au/cosmos/C/core-collapse`

[4] Wikipedia contributors. (2022, 22 June). Machine learning. Wikipedia. `https://en.wikipedia.org/wiki/Machine_learning`

[5] GeeksforGeeks. (2022, 25 June). An introduction to Machine Learning. `https://www.geeksforgeeks.org/introduction-machine-learning/`

[6] Cuoco, E., Powell, J., Cavaglià, M., Ackley, K., Bejger, M., Chatterjee, C., … & Williams, D. (2020). Enhancing gravitational-wave science with machine learning. Machine Learning: Science and Technology, 2(1), 011002.

[7] Astone, P., Cerdá-Durán, P., Di Palma, I., Drago, M., Muciaccia, F., Palomba, C., & Ricci, F. (2018). New method to observe gravitational waves emitted by core collapse supernovae. Physical Review D, 98(12), 122002.

[8] Van Den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., ... & Kavukcuoglu, K. (2016). WaveNet: A generative model for raw audio. SSW, 125, 2.

[9] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770–778)

[10] Shaheen, A., Waheed, U. B., Fehler, M., Sokol, L., & Hanafy, S. (2021). GroningenNet: Deep learning for low-magnitude earthquake detection on a multi-level sensor network. Sensors, 21(23), 8080.

[11] Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 1–9).

[12] Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Proceedings of the 32nd International Conference on Machine Learning (pp. 448–456). PMLR.

[13] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the Inception Architecture for Computer Vision. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 2818–2826).

[14] Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Proceedings of the 32nd International Conference on Machine Learning (pp. 448–456). PMLR.

VII. APPENDIX

A. 1 and 5 kpc Figures

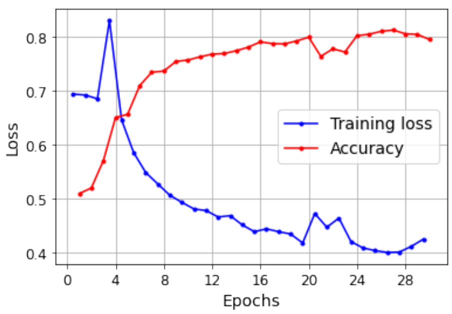

Figure 11: 1 and 5 kpc training dataset: accuracy and loss rates

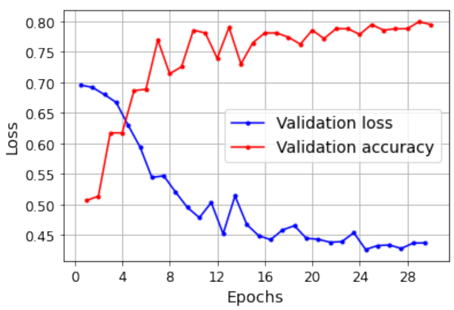

Figure 12: 1 and 5 kpc validation dataset: accuracy and loss rates

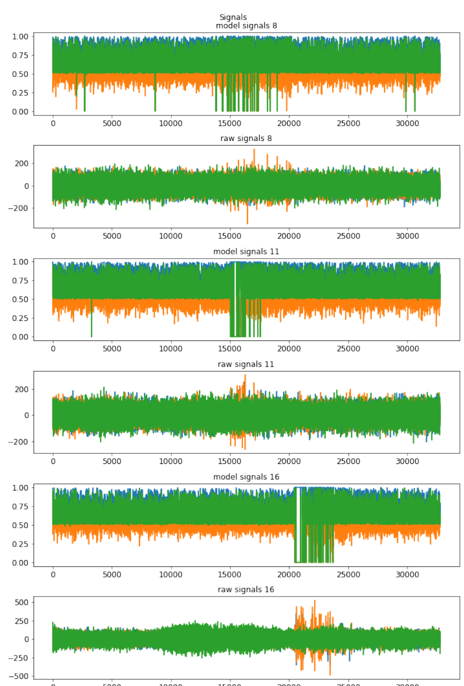

Figure 13: 1 and 5 kpc signals: 3-pairs of samples of model-generated results vs. original signals. All samples are injections.

B. 1 and 10 kpc Figures

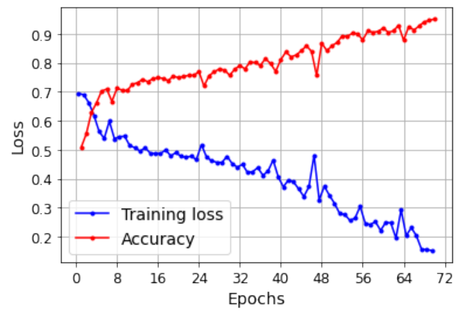

Figure 14: 1 and 10 kpc training dataset: accuracy and loss rates

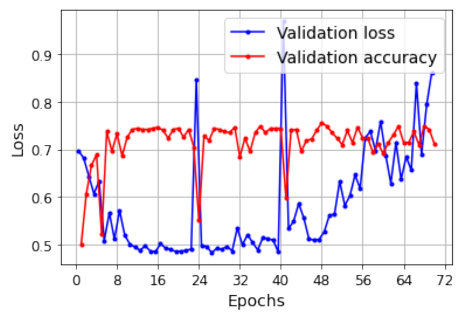

Figure 15: 1 and 10 kpc validation dataset: accuracy and loss rates

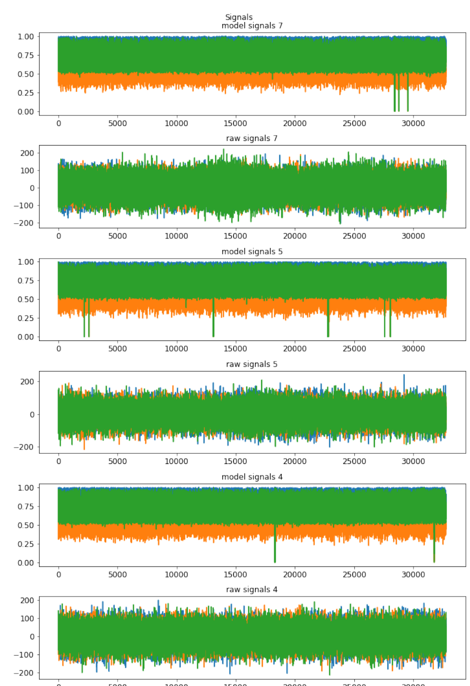

Figure 16: 1 and 10 kpc signals: 3-pairs of samples of model-generated results vs. original signals. All samples are injections.

C. 1 and 15 kpc Figures

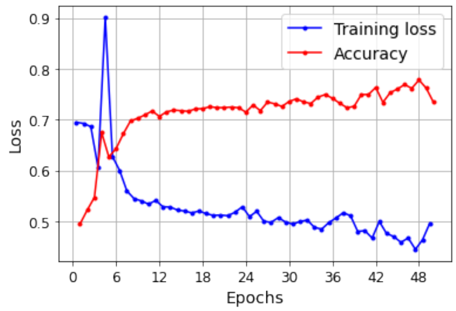

Figure 17: 1 and 15 kpc training dataset: accuracy and loss rates

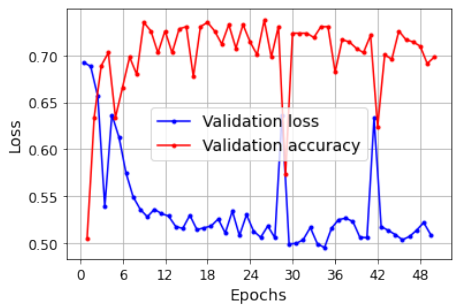

Figure 18: 1 and 15 kpc validation dataset: accuracy and loss rates

Figure 19: 1 and 15 kpc signals: 3-pairs of samples of model-generated results vs. original signals. All samples are injections.