PROCEEDINGS OF SCIENCE

Deep Learning for the Classification and Recovery of Cosmic-Ray Radio Signals against Background Measured at the South Pole

Abdul Rehman[a,*], **Alan Coleman**[a], **Frank G. Schröder**[a,b] **for the IceCube Collaboration**[c]

[a] *Bartol Research Institute, Department of Physics and Astronomy, University of Delaware, Newark, DE, 19716, USA*

[b] *Karlsruhe Institute of Technology, Institute for Astroparticle Physics, 76021 Karlsruhe, Germany*

[c] *Full author list at* https://icecube.wisc.edu/collaboration/authors/

E-mail: arehman@udel.edu, acoleman@udel.edu, fgs@udel.edu

A major hurdle in radio detection of cosmic-ray air showers is the continuous background that contaminates the signals. In this work, we use deep learning techniques to mitigate the effect of background by training two convolutional neural networks (CNNs). One network is used to distinguish the radio traces containing air-shower signals from those traces containing only background. The other network is trained to extract the underlying radio signals by removing the noise from the contaminated traces. In order to produce the required dataset for the training of the CNNs, we used CoREAS to simulate radio signals from cosmic-ray air showers for the geomagnetic field and observation height of the South Pole. As noise samples, we used radio background recorded by SKALA v2 antennas of a prototype station of the IceCube-Gen2 surface array at the South Pole. The frequency band used in the analysis ranges from 100 MHz to 350 MHz. The results show that the trained networks can indeed improve, on the one hand, the detection threshold of an externally triggered radio array and, on the other hand, the accuracy of the pulse parameters, such as the arrival time and amplitude of the radio pulses, which are subsequently used to reconstruct the properties of cosmic-ray air showers.

9th International Workshop on Acoustic and Radio EeV Neutrino Detection Activities - ARENA2022
7-10 June 2022
Santiago de Compostela, Spain

*Speaker

PoS(ARENA2022)012

1. Introduction

The energy threshold of IceTop [1], the surface component of IceCube Neutrino Observatory at the South Pole [2], increases over the years due to the snow accumulation on top of the detectors [3]. In order to address this issue, an enhancement of IceTop is planned in the upcoming years. The enhancement will be furnished on the footprint of IceTop and will be comprised of a total of 32 stations [4–6]. Each station itself will consist of 8 scintillator panels and 3 SKALA v2 radio antennas [7, 8]. The enhancement will improve the energy threshold of IceTop by mitigating the effect of snow. Moreover, it will also improve the accuracy of cosmic-ray measurements by detecting different components of air showers simultaneously.

To test the performance of the new stations, in January of 2020, a complete prototype station was successfully deployed at the South Pole. The station has been continuously taking data since then and has successfully measured air showers [9, 10]. In the prototype mode we currently record two sets of radio data from the station. The first set consist of scintillator-triggered events, and the second set of data is a software initiated readout of the antenna signals, also referred to as *soft triggered* data [11]. For this data, the antennas are read out at a constant rate of ~ 1 event per min. The soft-triggered data is used to study the radio background at the South Pole. For both sets of data, the waveforms of all prototype antennas are recorded.

In this work we are using one-dimensional CNNs to construct models for classifying the radio signals emanating from cosmic-ray air showers against the pure radio background. Moreover, we also construct a CNN model that is trained to extract the underlying radio signals from contaminated traces. The background data recorded by the prototype station is used to train the CNNs. This work is a continuation of past research that developed CNNs trained on modeled background [12] and built on previous studies by Tunka-Rex [13] and others [14].

2. Data Set Distribution

Radiowave pulses from the extensive air showers (EASs) were simulated using the CORSIKA v7.7401 (COsmic Ray SImulations for KAscade) [15] software, a Monte-Carlo code which propagates EASs generated by cosmic rays. In CORSIKA, the radio emission from an EAS is calculated using CoREAS [16]. The IceTop atmospheric model [15] is used as the shower media for particle interactions and Sibyll 2.3d is used as the high-energy hadronic interaction model for the produced simulations [17]. Proton and iron are used as primary particles with energies between 10^{17} and $10^{17.1}$ eV. Zenith angles (θ) ranging from 0 to 0.8 in $\sin^2\theta$ with random azimuth angles (ϕ) are used as arrival directions of the primary particles. Before feeding the data to the networks for training, the data are pre-processed similarly to what is described in [12], but with a few differences. Instead of modeled noise we are adding noise measured by the prototype station at the South Pole to the CoREAS simulations. Also, the lower limit of the frequency band is 100 MHz instead of 50 MHz. The design band of the SKALA antennas is 50 to 350 MHz and the study to search for the optimal sub-band to be used for the radio analysis at the South Pole is still ongoing [18].

The signal-to-noise ratio (SNR), used to quantify the radio signal strength within a given voltage waveform of amplitudes, Y_i in each bin i is calculated as follows,

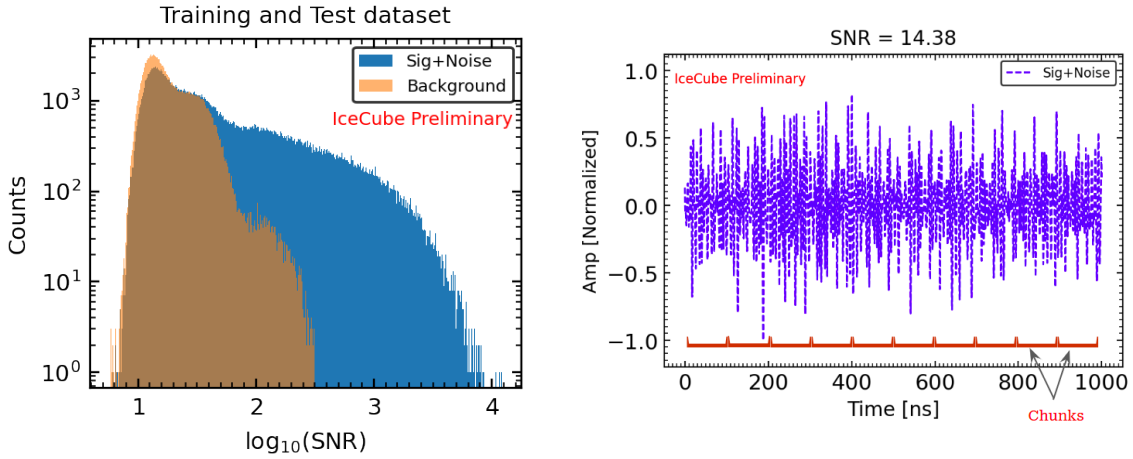

Figure 1: Left: Signal-to-noise ratio (SNR) distribution of the data set used for training and testing. Shown in blue is the SNR distribution of waveforms containing simulated radio signals added to background whereas the orange one shows the distribution of the background alone. Right: An example waveform containing radio signal plus measured background. The total length of the waveform is 4000 bins which is equivalent to 1000 ns in the time domain, it is normalized in the range -1 and 1 before being used for training. The chunks for calculating the SNR (eq. (1)) are highlighted in red. The SNR value of this example waveform is 14.38.

$$\text{SNR} = \left(\frac{\max[|Y_i|]}{\text{median}[\text{RMS}(n_j)]} \right)^2 . \tag{1}$$

Here, the numerator is the maximum magnitude of the waveform. The denominator, which is used as a noise estimator, is calculated by dividing the waveform into consecutive, non-overlapping chunks of equal length, n_j (see fig. 1, right). The RMS of each of these chunks is calculated and then the median RMS value is taken. Different values of n (number of chunks) are tested to compute the RMS values and the results converged at $n = 10$, where increasing n further did no alter the results. By estimating the noise in this way we can minimize the possible bias due to the presence of radio frequency interference (RFI) and air-shower pulses.

The SNR distribution of our dataset using the above described definition is depicted in fig. 1. The blue distribution shows the SNR values of waveforms containing signal and background whereas the orange distribution is for background only waveforms. There are different features in the background distribution, the sources of which are under investigation.

Our dataset contains a total of ~300k background waveforms and ~150k signal-plus-background (noisy) waveforms. This is then divided into two subsets, 80% of which goes in to the training of networks and 20% for the testing. The splitting of dataset into smaller subsets is achieved using the scikit-learn [19] python library.

3. Classification and De-noising

The Tensorflow [20] and Keras [21] libraries are used to construct the architecture of our *classifier* and *de-noiser* networks. Both networks are based on the auto-encoder technique where the data is first compressed into lower dimensional space using encoding layers and then mapped

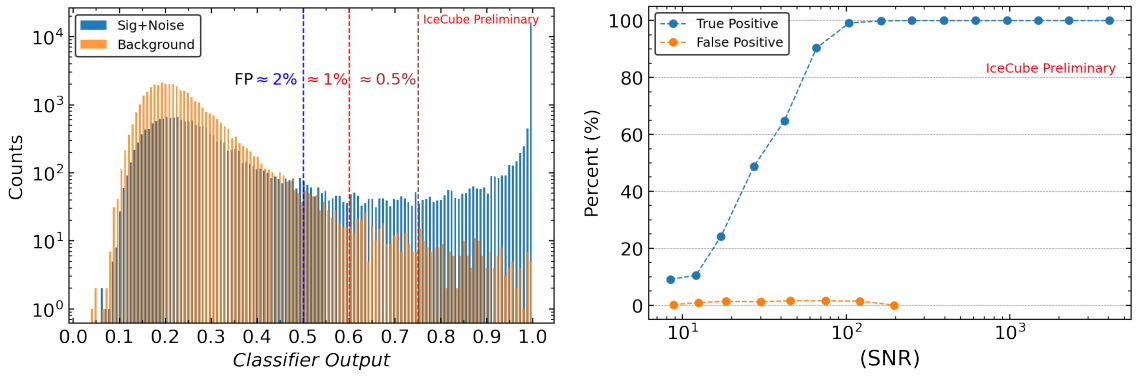

Figure 2: Left: Classifier output values for noisy waveforms (blue) and pure-background waveforms (orange). False positive (FP) rates are indicated by the vertical dashed lines for certain thresholds of the classifier output. Right: TP and FP rates of events for a classifier score above 0.6 as function of the SNR.

back to the original dimensions using decoding layers. The layer-wise structure of the networks is explained in [12]. The mean squared error is used as the loss function and to regularize the training process we used the early stopping criterion of Keras. This monitors the training process and terminates the training when the value of the loss function stops improving after a certain number of epochs. This also gives a way to restrict the networks from over-fitting the data.

3.1 Classifier Results

The *classifier* is constructed with the aim to distinguish the radio signal waveforms from the pure background waveforms. It is trained using supervised learning on the dataset described in section 2, along with the labels 0 and 1 for noisy signals and background-only waveforms, respectively. The network architecture consists of 2 pairs of encoding and decoding layers. Each convolutional layer in both aforementioned layers has 8 filters and a kernel size of 140 bins which in the time domain is equivalent to 35 ns. The network has a final dense layer with only one neuron and a sigmoid layer as an activation function. This final layer normalizes the input from the previous layer in the [0;1] range. The output results for our classifier are shown in fig. 2, left. We allow for a small false positive rate of about 1% and hence choose a threshold of 0.6, which means, waveforms with classifier output value ≥ 0.6 are considered as signal-like and < 0.6 are considered as background-like. The correctly classified waveforms or true positive (TP) and misidentified waveforms or false positive (FP) rate calculated with the threshold of 0.6 is shown as a function of SNR values in fig. 2, right. For later applications, different thresholds could be used depending on the false-positive rate that is tolerable for a specific cosmic-ray analysis.

3.2 De-noiser Results

The *de-noiser* is trained independently from the classifier. However, the architecture of the best performing network again consists of 2 pairs of encoding and decoding layers. Each convolutional layer has 8 filters with a kernel size of 800 bins (corresponding to 200 ns). Simulated CoREAS waveforms processed without adding noise are used as labels for the 'true' signal during training. Examples of de-noised waveforms, using the network, in comparison with the noisy and true

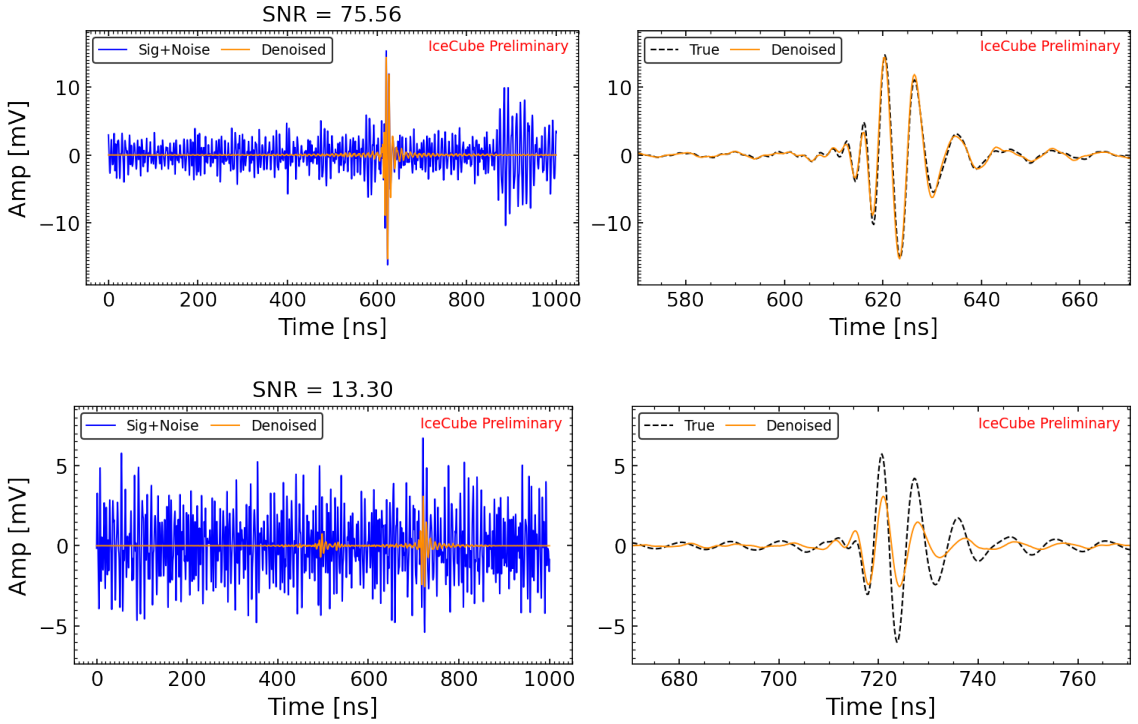

Figure 3: Examples waveforms that are de-noised using the network. Shown on left, in orange, are the outputs from the *de-noiser* overlaid on the noisy waveforms (blue). Shown on the right are the same de-noised waveforms in comparison with the true simulated waveforms. The x-axis is zoomed in to show structures. The 1st row represents an example of good de-noising where the 2nd row represent the case where the network was unable to reconstruct the amplitude of the radio pulse within the waveform.

waveforms are shown in fig. 3. The first row in the figure includes an example where the network was able to completely remove the noise from the waveform without distorting the shape of the signal. The second row, however, shows an example where, although the noise was mostly removed from noisy waveform, but the signal amplitude was not recovered properly after de-noising. Additionally, a second pulse has been erroneously added by the network to the left of the true pulse. These kind of false pulses were frequently observed when initially using small kernels in the convolutional layers, but we were able to largely reduced them by selecting a large kernel size.

To fully evaluate the performance of the *de-noiser* we consider two metrics, the *Power Ratio* as shown in fig. 4 left, and the *Peak Time Difference* (Δt) shown in fig. 4 right, defined as (see [12] for full description of the quantities)

$$\text{Power Ratio} = \frac{\left[P_{\text{sig}} - P_{\text{noise}}\right]_{\text{measured}}}{\left[P_{\text{sig}} - P_{\text{noise}}\right]_{\text{true}}}, \quad \text{and} \quad \Delta t = T_{\text{measured}} - T_{\text{true}}\,. \tag{2}$$

We compute both quantities for the input noisy data and de-noised data and plot them as a function of SNR. A significant improvement can be seen over the entire range of SNR values for both quantities by using the networks. A bias in the power ratio plot, in the SNR $\lesssim$ 100 region, is the result of the waveforms shown in second row of fig. 3 where the amplitude of de-noised pulses are smaller than the true ones.

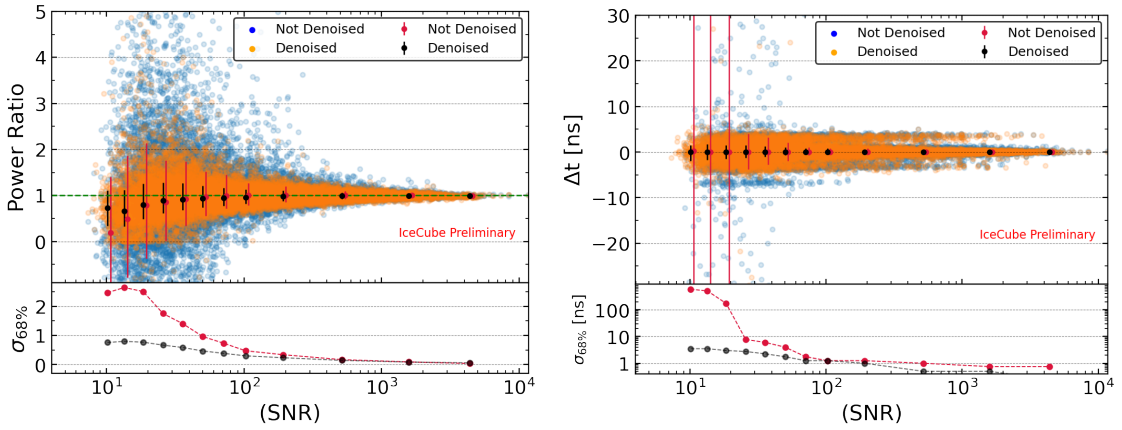

Figure 4: Left: power ratio; right: peak time difference (Δt) over SNR (see [12] for detailed description of both quantities). The subplot at the bottom of both plots represents the 68% containment. A small bias can be seen in the power ratio similar to as in [12].

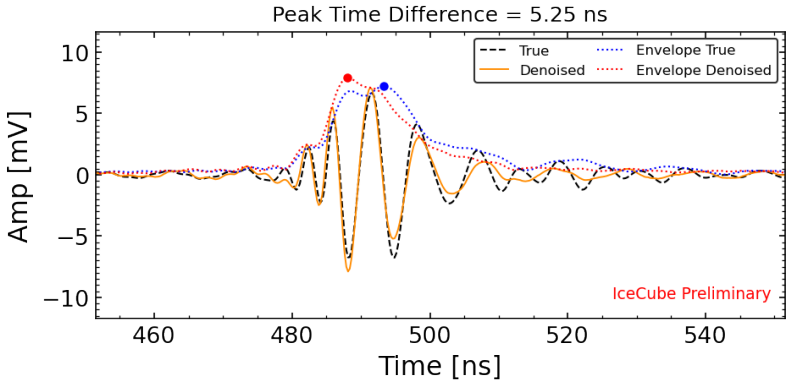

Figure 5: An example plot where the network was able to identify and de-noise the noisy waveform, however, the pulse shape was slightly mis-reconstructed, hence a peak time difference of 5.25 ns between the true and denoised pulse can be seen.

The peak time difference, as shown in fig. 4, shows an improved resolution, especially at low SNR values with typical time differences of a few ns. Additional distributions are observed at approximately ±4 ns along with the central one at ≃ 0 ns. Due to the high frequencies and the inherent duration of air shower pulses, the peak location of the Hilbert envelope can switch between two local extrema when they have similar amplitudes. This is shown in fig. 5. The red and blue dotted lines in the plot represent the Hilbert envelope computed for the de-noised and true waveforms, respectively. The circles on both lines show where the respective peak location of both the waveforms occur. From this we can see that, although, the two signals looks very similar by eye there is, however, a difference of 5.25 ns between the two peak locations.

4. Conclusion and Outlook

We have presented results of two CNNs that are used for the identification and de-noising of simulated radio signals from cosmic-ray air showers. To train the networks we used samples of

radio background from the prototype surface station at the South Pole. With these networks we have shown that a significant improvement regarding the detection threshold as well as the accuracy of the signal time and amplitude can be achieved over the traditional method of a simple SNR threshold. The results presented here are, however, limited by the quality of background data used for training. Both, the hardware for recording the data and the software for processing the data, will improve in the future. Hence, further improvements in the results may be achieved by investing more efforts in the pre-processing of the background data. Furthermore, with these networks we achieved a similar performance to what was previously shown for modeled or Gaussian noise. This give us additional motivation that these networks can be used to improve the detection threshold of radio experiments. Finally, both the *classifier* and *de-noiser* will be implemented as part of IceCube's radio data processing software [22], to enable the better identification of measured air showers and to improve the accuracy of the radio reconstruction of cosmic rays detected by IceCube.

Acknowledgement: This work was supported by the U.S. National Science Foundation-EPSCoR (RII Track-2 FEC, award # 2019597) and by NASA EPSCoR project, Grant 80NSSC20M01.We thank the Tunka-Rex collaboration for sharing their code, which became the starting point of this work (`https://gitlab.ikp.kit.edu/tunkarex/denoiser`). This research was also supported in part through the use of Data Science Institute (DSI) computational resources at the University of Delaware.

References

[1] R. Abbasi *et al. Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **700** (2013) 188–220.

[2] M. G. Aartsen *et al. Journal of Instrumentation* **12** no. 03, (2017) P03012.

[3] **IceCube** Collaboration, K. Rawlins *ICRC2013* **33** (2013) 2356.

[4] **IceCube** Collaboration, F. G. Schröder *PoS* **ICRC2019** (2020) 418.

[5] A. Haungs *EPJ Web Conf.* **210** (2019) 06009.

[6] T. Huber, J. Kelley, S. Kunwar, and D. Tosi *PoS* **ICRC2017** (2017) 401.

[7] **IceCube** Collaboration, A. Leszczyńska and M. Plum *PoS* **ICRC2019** (2020) 332.

[8] E. de Lera Acedo *et al. Proceedings of ICEAA 2015* (2015) 839–843.

[9] **IceCube** Collaboration, M. Oehler and R. Turcotte *PoS* **ICRC2021** (2021) 225.

[10] **IceCube** Collaboration, H. Dujmović, A. Coleman, and M. Oehler *PoS* **ICRC2021** (2021) 314.

[11] R. Turcotte. PhD thesis, Karlsruhe Institute for Technology, 2022. Submitted.

[12] A. Rehman, A. Coleman, F. G. Schröder, and D. Kostunin *PoS* **ICRC2021** (2021) 417.

[13] D. Shipilov *et al.* *EPJ Web Conf.* **216** (2019) 02003.

[14] P. A. Bezyazeekov *et al.* in *25th European Cosmic Ray Symposium*. 1, 2017. arXiv:1701.05158 [astro-ph.IM].

[15] D. Heck *et al. Report fzka* **6019** no. 11, (1998) .

[16] T. Huege, M. Ludwig, and C. W. James *AIP Conf. Proc.* **1535** no. 1, (2013) 128.

[17] F. Riehn *et al. Physical Review D* **102** no. 6, (2020) 063002.

[18] **IceCube** Collaboration, A. Coleman *PoS* **ICRC2021** (2021) 317.

[19] F. Pedregosa *et al. J. Machine Learning Res.* **12** (2011) 2825–2830.

[20] M. Abadi *et al. arXiv:1603.04467* (2016) .

[21] F. Chollet *et al.*, "Keras." https://keras.io, 2015.

[22] R. Abbasi *et al.* *Journal of Instrumentation* **17** no. 06, (Jun, 2022) P06026.