Deep learning techniques for a real-time neutrino classifier

Astrid Anker[1], Pierre Baldi[2], Steven W. Barwick[1], Jakob Beise[3], Hans Bernhoff[4], Dave Z. Besson[5,6], Nils Bingefors[3], Maddalena Cataldo[7], Pisin Chen[8], Daniel García Fernández[9,7], Geoffrey Gaswint[1], Christian Glaser[3], Allan Hallgren[3], Steffen Hallmann[9], Jordan C. Hanson[10], Spencer R. Klein[11], Stuart A. Kleinfelder[12], Robert Lahmann[7,1], Jiayi Liu[1], Mitchell Magnuson[5], Stephen McAleer[2], Zach Meyers[9], Jiwoo Nam[8], Anna Nelles[9,7], Alexander Novikov[5,6], Manuel P. Paul[1], Christopher Persichilli[1], Ilse Plaisier[9,7], Lilly Pyras[9], Ryan Rice-Smith[1], Joulien Tatar[13], Shih-Hao Wang[8], Christoph Welling[9,7], Leshan Zhao[1]

[1]Department of Physics and Astronomy, University of California, Irvine, CA, USA.
[2]Department of Information and Computer Science, University of California, Irvine, CA, USA.
[3]Uppsala University Department of Physics and Astronomy, Uppsala, Sweden.
[4]Uppsala University Department of Engineering Sciences, Division of Electricity, Uppsala, Sweden.
[5]Department of Physics and Astronomy, University of Kansas, Lawrence, KS, USA.
[6]National Research Nuclear University MEPhI (Moscow Engineering Physics Institute), Moscow, Russia.
[7]ECAP, Friedrich-Alexander Universität Erlangen-Nürnberg, Erlangen, Germany.
[8]Department of Physics and Leung Center for Cosmology and Particle Astrophysics, National Taiwan University, Taipei 10617, Taiwan.
[9]DESY, 15738 Zeuthen, Germany.
[10]Whittier College Department of Physics, Whittier, CA, USA.
[11]Lawrence Berkeley National Laboratory, Berkeley, CA, USA.
[12]Department of Electrical Engineering and Computer Science, University of California, Irvine, CA, USA.
[13]Research Cyberinfrastructure Center, University of California, Irvine, CA, USA.

Abstract

The ARIANNA experiment is a detector designed to record radio signals created by high-energy neutrino interactions in the Antarctic ice. Because of the low neutrino rate at high energies, the physics output is limited by statistics. Hence, an increase in detector sensitivity significantly improves the interpretation of data and offers the ability to probe new physics. The trigger thresholds of the detector are limited by the rate of triggering on unavoidable noise. A real-time noise rejection algorithm enables the thresholds to be lowered substantially and increases the sensitivity of the detector by up to a factor of two compared to the current ARIANNA capabilities. Deep learning discriminators based on Fully Connected Neural Networks (FCNN) and Convolutional Neural Networks (CNN) are evaluated for their ability to reject a high percentage of noise events (while retaining most of the neutrino signal) and to classify events quickly. In particular, we describe a CNN trained on Monte Carlo data that runs on the current ARIANNA microcontroller and retains 95% of the neutrino signal at a noise rejection factor of 10^5.

1 Introduction

Multi-messenger astrophysics is a field focused on observing not just light, but other messengers coming from outer space to learn more about the universe. This research is focused on measuring

Fourth Workshop on Machine Learning and the Physical Sciences (NeurIPS 2021).

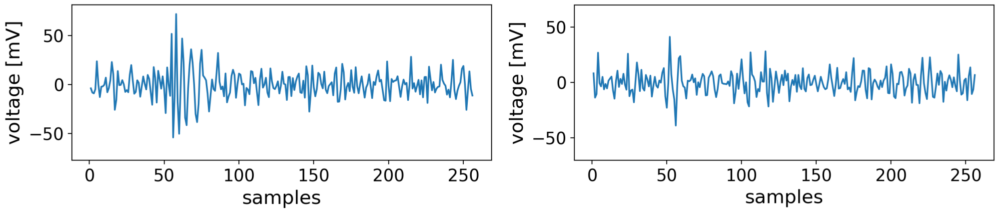

Figure 1: Example waveforms for neutrino signal (left) and noise (right). Each plot contains 256 samples, which is the data from one antenna.

extreme-high-energy neutrinos to better understand the fierce processes of astronomical objects that create them [3]. In many ways, neutrinos are ideal cosmic messengers, but their expected rates and interaction cross-section are extremely small [1, 7]. If the sensitivity of the detector is increased, there is a higher chance of measuring these elusive neutrinos. One way to increase the sensitivity is to lower the trigger threshold so that smaller neutrino signals are recorded by the detector. The problem with this is that the trigger rate is already dominated by unavoidable noise, and the detector has a limited data transmission rate since it's located in a remote region of Antarctica. However, if noise is identified and rejected in real time, the trigger thresholds can be lowered while maintaining the same data rate, thus increasing the sensitivity of the detector.

The ARIANNA detector is an array of radio autonomous stations located in Antarctica [4]. Each station consist of four log periodic dipole antennas (LPDA's), but only two antennas need to measure a signal to trigger the station. Once a station has triggered, the digitized waveforms of every antenna channel contain 256 samples. The waveform data from all channels are piped into an Xilinx Spartan 4 FPGA, and then further processed and stored to an internal $32\,\mathrm{GB}$ memory card by an MBED LPC 1768 microcontroller. Once a triggered event is saved, it is transferred to UC Irvine via Iridium Satellite, which has an expected operation trigger rates of $0.3\,\mathrm{mHz}$. In this study, neural networks are used to classify incoming data in real time into noise and signal. The classification problem is highly asymmetric with noise being many order or magnitude more common than signal. The efficiency and processing time of the network is determined by implementing it onto the current MBED (high level control of the board) and separately a possible upgrade to the MBED, a Raspberry Pi. Any future improvements to the ARIANNA hardware need to be low powered and able to withstand the harsh conditions of Antarctica.

2 Noise rejection using neural networks

To implement a deep learning filter, the network structure needs to be optimized for fast and accurate classification. For accuracy, the two metrics are neutrino signal efficiency (defined here as the ratio of correctly identified signal events to the total number of signal events) and noise rejection factor (defined here as $\frac{1}{1-N}$, where N is the noise efficiency, i.e., the ratio of correctly identified noise events to the total number of noise events). The goal is to reject several orders-of-magnitude of noise while retaining most of the neutrino signals. At trigger level, the signal purity is only of secondary concern. In the following, the target is 5 orders-of-magnitude noise rejection while providing a high signal efficiency at or above 95%. This would enable the trigger threshold to be lowered significantly – thus increasing the sensitivity to extreme-high-energy neutrinos – while keeping a low event save rate of a few mHz. Typically using a more complex network structure would yield more accurate results, but this would also create a slower network. These two constraints need to be optimized as the deep learning architecture is developed.

2.1 Data

NuRadioMC [6, 5] is a Monte Carlo simulation package for radio neutrino detectors, and it is used to simulate a representative set of the expected neutrino events and thermal noise events for the ARIANNA detector. In total 121,597 events are generated for the neutrino signal data set, and one million events are generated for the noise data set. In Fig. 1, one antenna's data from one event is plotted for both neutrino signal and noise separately.

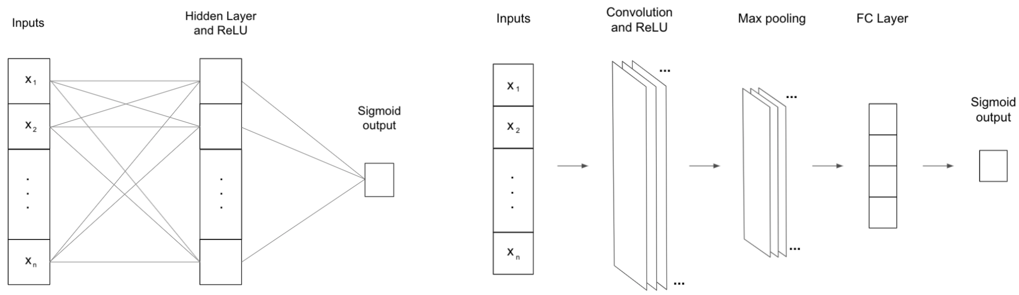

Figure 2: Baseline architecture of a fully connected neural network (left) and a convolutional neural network (right).

2.2 Network structures and training

All of the networks are created with Keras, a high-level interface to the machine-learning library TensorFlow [2]. Because a fast execution time of the network is one of the primary objectives, it is important to keep the network size small. While the number of trainable parameters can give an indication of network size, the number of Floating Point Operations (FLOPs) is the chosen metric for network size in this paper. An efficient way to improve the network speed is to reduce the input data size. Instead of feeding the signal waveform from all four antennas into the network, one way to cut down on the size of input data is to use only the two antennas that caused the trigger. As each signal waveform consists of 256 samples, the total input size to the network is 512 samples. In addition, a further reduced input data set is studied that contains only 100 samples; selecting the antenna with the highest signal amplitude, only the 100 samples around the maximum absolute value are used. The reasoning for this is that the dominant neutrino signal does not span over the whole record length and typically spans over less than 50 samples.

The two network architectures studied in the following are a fully connected neural network (FCNN) and a convolutional neural network (CNN), depicted in Fig. 2. The FCNN used in this baseline test is a fully connected single hidden layer with a node size of 64 for the 100 input samples and 128 for the 512 input samples, a ReLU activation, and then a sigmoid activation in the output layer. The CNN structure consists of 5 filters with 10x1 kernels each, a ReLU activation, a dropout of 0.5, a max pooling with size 10x1, a flattening step to reshape the data, and a sigmoid activation in the output layer. Both the CNN and FCNN are trained using the Adam optimizer with varying learning rates from 0.0005-0.001 depending on which value works best for each individual model. The training data set contains a total of 100,000 signal events and 600,000 noise events, where 80% is for training and 20% is to validate the model during training. Once the network is trained, the test data is used which contains 21,597 signal events and 500,000 noise events.

2.3 Neural network performance

The signal and noise classification distributions are significantly distinct. With the sigmoid activation in the output layer, the classification distribution falls between 0 and 1, where close to 0 is noise-like and close to 1 is signal-like. Once trained, a similar threshold cut value distribution to the left plot in Fig. 3 is obtained for the signal and noise data of all models. From each distribution, the amount of signal efficiency vs. noise rejection can be varied by choosing different threshold cut values. Training and testing these networks with each input data size yields the signal efficiency vs. noise rejection plot on the right hand side of Fig. 3. Each data point corresponds to a different threshold cut value, and the final threshold cut value is chosen by optimizing the noise rejection for the desired signal efficiency. Since all of the networks have efficiencies above our target of 95% for signal at 10^5 noise rejection, the main consideration for which network is best is the amount of FLOPs required for each network because this directly impacts the processing time. Typically, CNN's have less parameters overall due to their convolutional nature, which focuses on smaller features within a waveform; comparatively, the FCNN considers the whole waveform to make its prediction, so it requires more node connections. The next step is to investigate the FLOPs for each network, and determine the processing time on a given device.

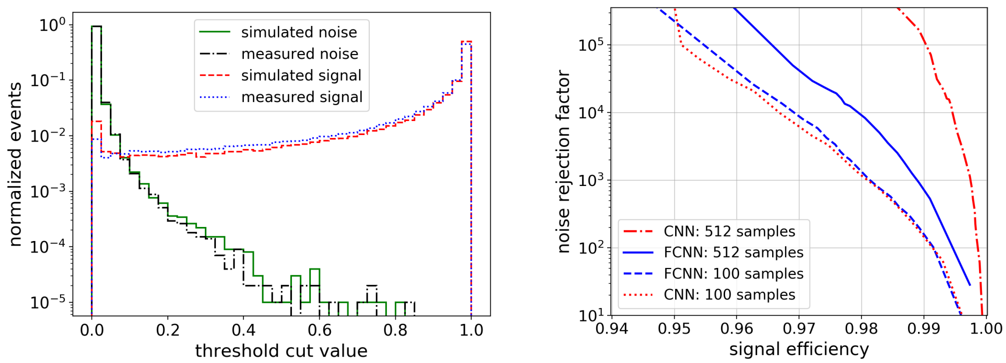

Figure 3: Histogram of threshold cut value for signal/noise classification (left), and signal efficiency vs. noise rejection factor for FCNN's and CNN's with input data of 100 or 512 samples (right).

Table 1: Processing times per event and the number of Floating Point Operations (FLOPs) of various models. *memory limitations prevented this measurement.

model	FLOPs	MBED	Raspberry Pi
FCNN 512 samples	131,457	45 ms	2.5 ms
CNN 512 samples	55,816	*	1.5 ms
FCNN 100 samples	12,993	4.7 ms	0.46 ms
CNN 100 samples	10,096	3.7 ms	0.39 ms

3 Experimental tests

We verify our simulation results in a lab measurement. A neutrino-like signal is generated using an arbitrary waveform generator, attenuated, amplified, and recorded by the ARIANNA data acquisition system (DAQ) and processed by the neural network in real time. The attenuation and amplification steps lead to low signal-to-noise ratios as expected for neutrinos. A noise data set is measured by recording thermal noise fluctuations from the amplifier. As shown on the left side of Fig. 3, the distributions of the neural network output between simulated and measured data agree well, confirming the correctness of the simulated data set and the derived conclusions. Next, the processing time of the deep-learning filter is studied. As the deep learning filter is intended as a real-time trigger, a fast execution time is crucial. The current ARIANNA hardware is used to test and measure the execution time under realistic conditions. Two microprocessors are explored for their processing time and power consumption: a Raspberry Pi compute module 3+ and the MBED microcontroller. The MBED is the current device installed in ARIANNA and the neural network is implemented through custom C code. The Raspberry Pi is a microcomputer with a Raspbian operating system which is based on Debian. As with the MBED, the neural net is implemented with a similar custom C code on the Raspberry Pi. Because the optimal networks found in the previous section are small and shallow, it is fairly simple to write a custom code that implements the trained neural network in C for maximum performance. To test the prediction capabilities and the classification time in both devices, a simulated event is read in and either matrix multiplied by the array of weights and biases in the FCNN case or convolved with the weights and bias filters in the CNN case.

The total event processing time target value is under $10\,\mathrm{ms}$ so that the current trigger thresholds can be lowered substantially. This would increase the detector's sensitivity to neutrino by up to a factor of two. Table 1 gives the processing times for various networks. The 100 input sample CNN has the lowest processing time of $3.7\,\mathrm{ms}$ ($270\,\mathrm{Hz}$) and meets the efficiency requirements. In this test, only the execution time of the neural network is measured. The time to transfer the data from the waveform digitizers to the microcomputer of the current ARIANNA hardware is $7.3\,\mathrm{ms}$, which is significantly longer and restricts the low-level trigger rate. However, this can be solved with a new revision of the ARIANNA DAQ in the future.

4 Summary and discussion

Due to the low neutrino flux at extreme-high-energies, the physics output of neutrino detectors is limited by statistics. Probing new physics is made possible by implementing deep learning techniques to increase the sensitivity of the ARIANNA detector. It was demonstrated that already a small shallow CNN is capable of rejecting five order of magnitude of noise while retaining 95% of the neutrino signal. In the future, several improvements to the ARIANNA hardware will be considered. First, the ARIANNA hardware that sends the data from the FPGA to the MBED can be parallelized to decrease the event readout time, which would provide the opportunity to trigger the detector at even higher rates. Second, more capable computing on the hardware through improved electronics will be studied. The current generation of ARIANNA hardware is now more than 10 years old, and many recent microcomputer systems offer more performance at comparable power consumption.

Acknowledgement

We are grateful to the U.S. National Science Foundation-Office of Polar Programs, the U.S. National Science Foundation-Physics Division (grant NSF-1607719) for supporting the ARIANNA array at Moore's Bay, and NSF grant NRT 1633631. Without the invaluable contributions of the people at McMurdo Station, the ARIANNA stations would have never been built. We acknowledge funding from the German research foundation (DFG) under grants GL 914/1-1 and NE 2031/2-1m, the Taiwan Ministry of Science and Technology, the Swedish Government strategic program Stand Up for Energy, MEPhI Academic Excellence Project (Contract No. 02.a03.21.0005) and the Megagrant 2013 program of Russia, via agreement 14.12.31.0006 from 24.06.2013.

References

[1] M. Aartsen et al. Differential limit on the extremely-high-energy cosmic neutrino flux in the presence of astrophysical background from nine years of icecube data. *Physical Review D*, 98(6), Sep 2018. ISSN 2470-0029. doi: 10.1103/physrevd.98.062003. URL `http://dx.doi.org/10.1103/PhysRevD.98.062003`.

[2] M. Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

[3] M. Ackermann et al. Astrophysics Uniquely Enabled by Observations of High-Energy Cosmic Neutrinos. *Bull. Am. Astron. Soc.*, 51:185, 2019.

[4] A. Anker et al. Targeting ultra-high energy neutrinos with the arianna experiment. *Advances in Space Research*, 64(12):2595–2609, Dec 2019. ISSN 0273-1177. doi: 10.1016/j.asr.2019.06.016. URL `http://dx.doi.org/10.1016/j.asr.2019.06.016`.

[5] C. Glaser et al. Nuradioreco: a reconstruction framework for radio neutrino detectors. *The European Physical Journal C*, 79(6), Jun 2019. ISSN 1434-6052. doi: 10.1140/epjc/s10052-019-6971-5. URL `http://dx.doi.org/10.1140/epjc/s10052-019-6971-5`.

[6] C. Glaser et al. Nuradiomc: simulating the radio emission of neutrinos from interaction to detector. *The European Physical Journal C*, 80(2), Jan 2020. ISSN 1434-6052. doi: 10.1140/epjc/s10052-020-7612-8. URL `http://dx.doi.org/10.1140/epjc/s10052-020-7612-8`.

[7] A. van Vliet, R. A. Batista, and J. R. Hörandel. Determining the fraction of cosmic-ray protons at ultrahigh energies with cosmogenic neutrinos. *Physical Review D*, 100(2), Jul 2019. ISSN 2470-0029. doi: 10.1103/physrevd.100.021302. URL `http://dx.doi.org/10.1103/PhysRevD.100.021302`.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] See the first paragraph of Section 1.
   (b) Did you describe the limitations of your work? [Yes] This is discussed at the end of Section 3.
   (c) Did you discuss any potential negative societal impacts of your work? [No] Our work has no impact on society.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [No] We use the open-source simulation code NuRadioMC which is documented in the literature. Repeating all assumptions in detail here is impossible given the 4 page limit of the paper.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] The data set is too big to be easily shared but the MC code is open source and the dataset can be reproduced with the information we provide in the paper.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 2.2
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] The error associated with the networks used were less than 2%, so error bars were omitted.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] The networks trained in this study were small so there was no need for extra processing power.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We used the open-source simulation code NuRadioMC and cited it accordingly.
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]