Passing Neutrinos

Azimuth and Zenith prediction based on Neural Networks

Maria Grazia Miccoli[1]

University of Bari - Aldo Moro, e-mail: m.miccoli45@studenti.uniba.it

ABSTRACT

Context. This report explores the study of subatomic particles known as neutrinos. These particles are nearly massless, carry no electric charge, and, despite being the most abundant particles in the universe, are incredibly difficult to detect. However, the IceCube Neutrino Observatory, a telescope installed in the ice of the South Pole, is able to detect these elusive particles, enabling scientists to study their trajectories through the Earth.
Aims. A recently closed challenge on Kaggle tasked participants with predicting the direction of these particles, specifically their azimuth and zenith angles.
Methods. The top-performing solutions demonstrated that this prediction is feasible by representing each neutrino event as a graph and utilizing transformer-based models. My objective was to tackle this task by employing simpler neural networks and a matrix representation of the data, while also addressing the computational limitations of the available resources.
Results. Simpler neural networks achieved good results using only activation timestamps or both timestamps and charge values.The most performing has been LSTM Layer-based neural network, although at the expense of time.

Key words. Machine learning, Deep Learning, Neural Network, Neutrinos, Azimuth, Zenith, event reconstruction, direction prediction

1. Introduction

In physics, neutrinos are elementary subatomic particles first discovered experimentally by Cowan and Reines' team in 1956. Their conceptual origin, however, traces back to Wolfgang Pauli, who hypothesized their existence in 1930 to explain the missing momentum and energy in beta decay (Wikipedia, Neutrino elettronico – Wikipedia, L'enciclopedia libera (2024)).

Neutrinos are peculiar in that they have zero charge and a very small mass. Despite being the most abundant particles in the universe, they are notoriously difficult to detect due to these properties. Nevertheless, studying neutrino behavior is crucial for scientists. Consequently, a specialized telescope called the IceCube Neutrino Observatory has been constructed. Extending kilometers into the ice at the South Pole and equipped with 5160 sensors, it can detect neutrinos as they interact with the ice, allowing for the study of their passage through the Earth (IceCube – Neutrinos in Deep Ice The Top 3 Solutions from the Public Kaggle Competition (2023)).

1.1. Neutrino and Ice

A question that might arise after reading about both neutrinos and the telescope descriptions is: "How does the telescope detect neutrinos if these particles have zero charge?" The answer lies in the way neutrinos interact with ice. When a neutrino encounters ice, it interacts with the ice particles, transferring energy to them and causing them to become excited or break apart. The energy released can be sufficient to produce secondary charged particles, such as muons or electrons, which emit light as they pass through the ice. This light is recorded by the sensors. There are two types of neutrino-ice interactions:

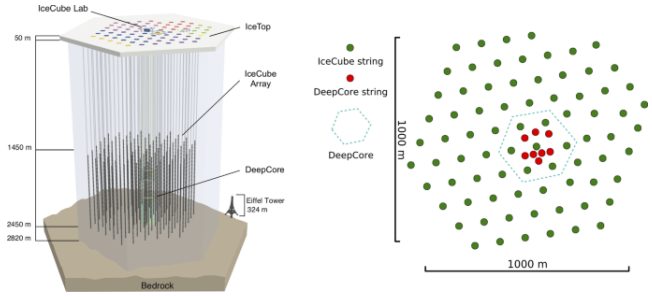


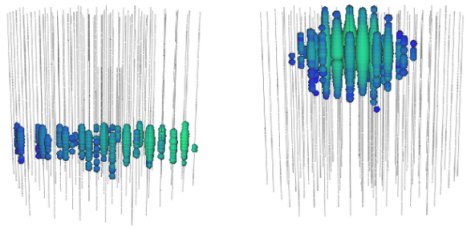
Fig. 1. The figure illustrates the structure of the telescope within the polar terrain. It consists of 86 rows of sensors, known as Digital Optical Modules (DOMs), which are distributed at depths ranging from 1450 meters to 2450 meters. For more detailed information on its components, please refer to the reference article. IceCube – Neutrinos in Deep Ice The Top 3 Solutions from the Public Kaggle Competition (2023)

- Tracks: Neutrinos that produce a linear trail of photons, which can travel long distances within the ice.
- Cascade: Neutrinos that produce a shower of light particles, which tend not to travel long distances.

(IceCube – Neutrinos in Deep Ice The Top 3 Solutions from the Public Kaggle Competition (2023))

2. Task: From Pulses to Directions

The task undertaken in this project is the prediction of a neutrino's direction, which was proposed as a challenge on the Kag-



(a) Track (b) Cascade

Fig. 2. The two types of interaction between neutrino and ice.

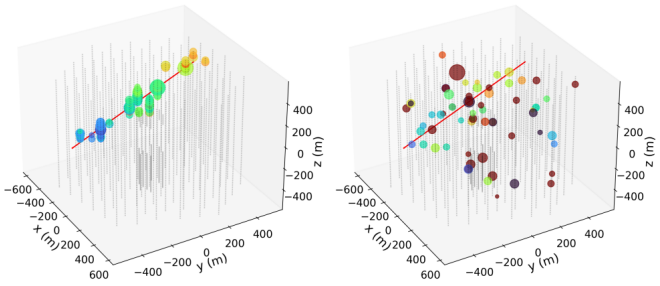


Fig. 3. This is a visual example of an event: the points represent the pulses recorded over time as a neutrino passes. The red line indicates the direction of the neutrino. In the first image, the event is depicted without noise, while in the second one, noise is included.

gle platform. Accurately predicting the direction and trajectory of a neutrino is crucial for scientists because it opens many avenues of research, including:

1. Researching the source from which the particle arrives, so that they can do a better mapping of the known (and also unknown) universe.
2. Searching for dark matter.
3. Probing the physical oscillations through atmospheric neutrinos.
4. Study the absorption capacity of the earth.

And many others. This section describes the details of the task, its dataset and the solutions already developed for the related Kaggle challenge. All this has been described in the articles " IceCube – Neutrinos in Deep Ice The Top 3 Solutions from the Public Kaggle Competition (2023) " and " Public Kaggle Competition "IceCube-Neutrinos in Deep Ice" (2023) " and will also be described how the task has been scaled down to be able to conduct the project and cope with the computational limitations.

2.1. Details of the task

Kaggle's task is to predict the direction of a neutrino passing through the IceCube telescope in terms of Azimuth and Zenith. The Azimuth is the horizontal angle, measured in degrees, indicating the position of an object relative to the observer, while the Zenith is the vertical angle. In other words, Azimuth can be seen as the latitude and Zenith as the longitude of the particle's trajectory. Each passage of a neutrino is considered an event, and each event is characterized by a set of pulses —neutrino-ice interactions— recorded by the sensors affected by the released photons. A sensor is activated only if it detects a certain amount of charge released after the neutrino-ice interaction. However, even if the signal's digitization trigger is activated to record the pulse, it might not be fully digitized, potentially resulting in noise. Additionally, each event is described by a variable number of pulses.

2.2. Detaset

Kaggle provides us with a dataset consisting of two folders:

1. Train: The training dataset consists of 660 files in parquet format, each containing the pulses that describe 200,000 events. In total, Kaggle provides pulses for 132 million events, each described by a variable number of pulses ranging from 0 (as some events might be purely noise) to several hundred. The target variables (azimuth and zenith) are not included within these files; instead, they are stored in a separate file called "train meta". This file contains the target variables associated with each event via an event ID and also includes the indices of the first and last row of pulses corresponding to each specific event.
2. Test: comprises only one file, which contains descriptions of events without associated azimuth and zenith values.

Additionally, a CSV file is provided, containing the x, y, and z positions for each of the 5160 IceCube sensors, along with their corresponding IDs for unique identification.

Making the summary of the situation, Kaggle gives us 132 million events, each event is described by the following features:

1. ID: unique event identifier
2. First pulse: index of the line that in the train file contains the first pulse belonging to the considered event (the others follow)
3. Last impulse: index of the line in the train file containing the last impulse belonging to the event considered.
4. Azimuth: first target features associated with the event.
5. Zenith: second target features associated with the event.

Each pulse, within train and set files, are described by:

1. ID: Identifier of the event they are associated with.
2. Time: nanosecond in which that pulse occurred in a time window.
3. Charge: charge detected at that time.
4. Sensor ID: Identified of the sensor that detected that pulse.
5. Auxiliar: flag indicating whether that pulse is relevant (Aux = False) or is noise (Aux = True)

For this project, the challenge was scaled down by taking only the first 300,000 events and performing extensive preprocessing.

2.3. Proposed Solutions

Before briefly describing the solutions proposed by the three winners of the challenge, we will explain the evaluation metric used by Kaggle: the mean angular error between the predicted and true event origins. This process involves determining the Mean Absolute Error (MAE) of the angular distance between two directions. Initially, the directions are converted into Cartesian coordinate vectors. Then, the scalar product of these vectors is computed, which corresponds to the cosine of the angle between them. By taking the inverse cosine (arccos) of this value, the angle between the two input vectors is obtained.

The first winning solution represented each event as a graph and created a model that combines EdgeConv, a neural network

for graphs that captures local information from each pulse, with transformer architecture to capture global information about an event. They developed and combined six model variants to improve performance. Additionally, they focused on preserving the intrinsic characteristics of the data by minimizing the necessary preprocessing. Azimuth and zenith were predicted separately.

The second solution, like the first, represents each event as a graph. To solve the task, it combines transformer architecture with a Fourier encoder, which embeds the continuous input variables into a multidimensional space. The third solution is similar to the second, also utilizing a transformer-based architecture.

All three models were evaluated with a mean angular error of 0.96. To determine precise rankings, Kaggle applied additional metrics, such as evaluating the models on different types of neutrinos or wakes. However, since no further details were provided to perform such an in-depth evaluation, the assessment for this project will be based solely on the mean angular error. For more details on the three solutions, please refer to the reference article IceCube – Neutrinos in Deep Ice The Top 3 Solutions from the Public Kaggle Competition (2023).

3. My solution

The solution I propose for this task focuses primarily on an initial representation of inputs that is different from graphs and on the use of simpler neural networks. My goal is to solve the same task with a more straightforward approach.

3.1. The idea

In summary, the task involves predicting the direction of a neutrino, represented by azimuth and zenith, with each event characterized by a variable number of pulses. My approach was to use the original number of significant pulses for each event, considering only the activation nanosecond of each sensor activated during each event. Additionally, another experiment was conducted that also considered the charge detected at each nanosecond. This was done to try to improve data generalization by accounting for the different types of wakes caused by neutrinos.

3.2. Pre-processing

To implement my idea, I applied an extensive preprocessing phase. As mentioned earlier, to address limited computational capacities, the first 300,000 events were considered, which, after some trials, proved to be the most significant. Essentially, the following filters were applied:

1. The events contained many pulses that represented only noise and were not significant for prediction, with some even consisting solely of noise pulses. Therefore, all rows (or entire events) of pulses representing only noise were removed from the original dataset to ensure that we were working only with significant pulses.
2. After the first filter, each event was described by the most significant pulses. To further facilitate the prediction of azimuth and zenith, an additional filter was applied to retain only the pulses with a charge greater than the overall average charge of the event to which they belonged. This way, we could better geolocate the area where the neutrino actually passed by hitting the ice particles, giving less importance to pulses recorded from a weaker light trail detection.
3. The third filter focuses on the activation nanoseconds of the sensors. In an event, it can happen that the same sensor records two different pulses at different nanoseconds due to signal propagation. Since we only need to know that the neutrino's passage hit that particular sensor in that specific location, only the first pulse that activated the sensor (i.e., the row with the smallest nanosecond for that sensor) was retained for each duplicated sensor in a particular event.

All these filters were applied sequentially, one after the other, and significantly reduced the size of the initial dataset.

3.3. Data Representation

In the representation of event impulses, a matrix representation was favored over graph-based methods. Consequently, for this task, two matrices were constructed with identical shapes and features. However, within each cell, one matrix contains the nanoseconds of activation while the other records the charge released by the sensor at that nanosecond. The matrices are structured as follows:

1. Features/Columns: Each event is characterized by sensor IDs because we require only the set of sensors activated during the event for each pulse, ensuring flexibility in accommodating events of varying sizes in the model.
2. Rows/Examples: Each event is represented on a row of the matrix and is described by the activated sensors.
3. Cell: Within each sensorID-event cell, one can find the activation nanosecond if considering the matrix that accounts for the temporal sequence of activation; or the detected charge from that sensor for the event, if considering the matrix of detected charges.

Another matrix is dedicated to the Azimuth and Zenith targets, linked to the corresponding event via event ID. Furthermore, after creating the time and charge matrices, it was observed that both matrices suffer from significant sparsity (as there may be events described by the activation of only a few sensors out of 5160). A final normalization phase was applied using the Max Absolute Scaler, which allowed for the preservation of matrix sparsity without information loss.

	Sensor 1	Sensor 2	...	Sensor 5160
event 1	0	0.9566	...	0.8566
event 2	0	0.7226	...	0

Table 1. This is an example of how the matrix appears containing the charges recorded by the sensor in that particular event. A cell with a value of 0 indicates that the sensor did not activate for that event.

4. Implementation

The implementation is divided into two experimental phases:

- **Phase 1:** Models are created that take as input a single matrix, the one related to the sensors' activation times, and predict azimuth and zenith.
- **Phase 2:** Multi-input and multi-output models are developed that work on both matrices previously described to predict azimuth and zenith.

4.1. Phase 1: Predict Azimuth And Zenith Using Only The Nanoseconds Of Activation

To proceed with this phase, I started with a scikit-learn model and then moved on to neural networks built with multiple layers in Keras:

- **SGD regressor**: for the initial model, we utilized the SGD regressor from scikit-learn. This model accepts a single input and produces a single output value. Consequently, two SGD regressors were implemented, one to predict azimuth and the other to predict zenith. Both regressors were trained using the nanosecond matrix as input and employing the same hyperparameters.
- **Dense Layer-Based Neural Network:**A single-input, multi-output neural network was constructed using Keras. The network primarily consists of Dense layers, which apply a linear transformation to the input data followed by a nonlinear activation function to capture the most significant patterns within the data. Additionally, a Masking layer serves as the initial layer, enabling the network to disregard the sparsity of the input matrix (i.e., inactive sensors). Following each Dense layer, a Dropout layer is employed to prevent overfitting by deactivating neurons that do not contribute to a robust representation of the data. Every Dense layer utilizes the Leaky ReLU activation function, effectively addressing the vanishing gradient problem. Furthermore, the network employs the mean squared error as the loss function and utilizes the Loss Scale Optimizer with AdaMax as inner optimizer to prevent underfitting. The network was trained using a 20% validation set and a batch size of 16. Initially designed for 20 epochs, the training process converged effectively within 3 epochs, leveraging the early stopping technique.
- **LSTM Layer-Based Neural Network:**Building upon the foundation of the previous neural network, a variation incorporating LSTM layers was developed. This LSTM-based network was trained using the same loss function, optimizer, and hyperparameters as the previous model. The Early Stopping technique halted training after 5 epochs.

4.2. Phase 2: Predicting Azimuth and Zenith Using The Nanoseconds Of Activation and Detected Charge

To incorporate both nanoseconds of activation and the corresponding detected charge, multi-input and multi-output neural networks were developed. These networks were built upon the foundation of the previously described neural networks, utilizing the same loss function, optimizer, and hyperparameters.

- **1D Convolutional Layer-Based Neural Network**: This neural network builds upon the LSTM Layer-Based Neural Network by incorporating 1D convolutional layers. These layers capture local information within each event using a kernel of size 3. The Early Stopping technique halted training after 5 epochs.
- **1D Convolutional Layer-Only Neural Network**: This network is based on the previous one but eliminates the LSTM layers and operates solely with 1D convolutional layers. It employs kernels of size 5 and 3. The Early Stopping technique terminated training after 5 epochs.
- **LSTM-Only Neural Network**: This neural network expands upon the LSTM Layer-Based Neural Network by making it multi-input. The Early Stopping technique halted training after 8 iterations.

5. Conclusion

Considering the mean angular error of the winning solution of 0.96, the results achieved in terms of mean angular error and training time for my solutions are as follows:

	Mean Angular Error	Fit time
SGD Regressor	1,570549829	1h 43min
Dense Layer-Based	0.2340966	53min
LSTM Layer-Based	0.23334724	5h 34min
1D Convolutional Layer-Based	0.23352784	45min
1D Convolutional Layer-Only	0.23548542	57min
LSTM Layer-Only	0.23589146	17h 5min

Table 2. Results

Disregarding the SGD Regressor, we can observe that neural networks achieve remarkable results with a mean angular error of 0.23. To determine the best model, we can focus on the execution time, where the neural networks with LSTM layers perform the slowest, and the ones with the best time performance are those combining convolutional layers and LSTM.

The first phase of the experiment demonstrated that to obtain good results, it is sufficient to consider the activation time of the sensors and that by introducing the charge as well, in the second phase, the performance worsens very slightly (probably due to the limited number of events considered and their representative significance in terms of charge).

Therefore, we can conclude that simple neural networks were able to solve the task within acceptable time frames. However, a further test for a better comparison with the state-of-the-art would be to use the entire dataset provided by Kaggle.

References

Wikipedia, Neutrino elettronico – Wikipedia, L'enciclopedia libera,2024, http://it.wikipedia.org/w/index.php?"

IceCube – Neutrinos in Deep Ice The Top 3 Solutions from the Public Kaggle Competition, Habib Bukhari and Dipam Chakraborty and Philipp Eller and Takuya Ito and Maxim V. Shugaev and Rasmus Ørsøe,2023, 2310.15674, arXiv, astro-ph.HE

icecube-neutrinos-in-deep-ice,Ashley Chow, Lukas Heinrich, Philipp Eller, Rasmus Ørsøe, Sohier Dane, IceCube - Neutrinos in Deep Ice, Kaggle, 2023 https://kaggle.com/competitions/icecube-neutrinos-in-deep-ice