FERMILAB-PUB-24-0872-STUDENT

Training NuGraph2 for ICARUS

Eric Novello
Supervisor: Sunny Seo, Co-Supervisor: Giuseppe Cerati
December 2, 2024

Abstract

This paper describes the process of training NuGraph2, a Graphical Neural Network for particle identification and event reconstruction, on simulated ICARUS data and initial performance and results of the training are discussed. This began with an investigation into filtering ICARUS spacepoint data, where it was found that filtering spacepoints by both their χ^2 value and by the number of associated hits was most effective. NuGraph2 was repeatedly trained on three event samples, which were used for optimizing hyperparameters and for debugging preprocessing and training scripts as well as fixing unexpected crashes that rarely happened during training. This work will make a new branch on the NuGraph2 GitHub repository for ICARUS.

1. ICARUS Experiment

ICARUS is the largest of the three Liquid Argon Time Projection Chamber (LArTPC) experiments in the Short-Baseline Neutrino program (SBN) at Fermi National Accelerator Laboratory (Fermilab) that aims to confirm or refute the existence of a fourth generation of neutrino oscillation, a possible explanation for the excess of events in the low energy region seen in MiniBooNE [1].

The SBN program uses the Booster Neutrino Beam (BNB), which provides it with neutrinos produced from 8 GeV protons on target. ICARUS is the farthest of the detectors, located 600 m along the BNB beamline compared to MicroBooNE at 470 m and the Short Baseline Near Detector (SBND) at 110 m [2].

ICARUS contains a total of 760 tons of cryogenically cooled ultra-pure liquid argon split between 2 identical cryostats as shown in Figure 1. In each cryostat, there are two LArTPCs divided by a cathode parallel to the BNB. On the outer edges of each LArTPC there are three anode wire planes. The first two are induction wire planes angled at 0 and 60 degrees from horizontal and the third is a collection wire plane angled at -60 degrees from horizontal. The angles of the three wire planes give each of them a unique 2D representation of the detector. The cathode and anode have an electric potential...which accelerates ionized electrons to anode, where the ionized electrons are produced by interaction of charged particles originating from the initial neutrino interactions or cosmic ray [2][3].

Gaussian pulses on LArTPC wires from ionizing electrons form hits. Spacepoints are formed by grouping hits from across wire planes, where each spacepoint corresponds to three hits, one from each wire plane. Spacepoints are cartesian representations of the locations of the hits on the wires.

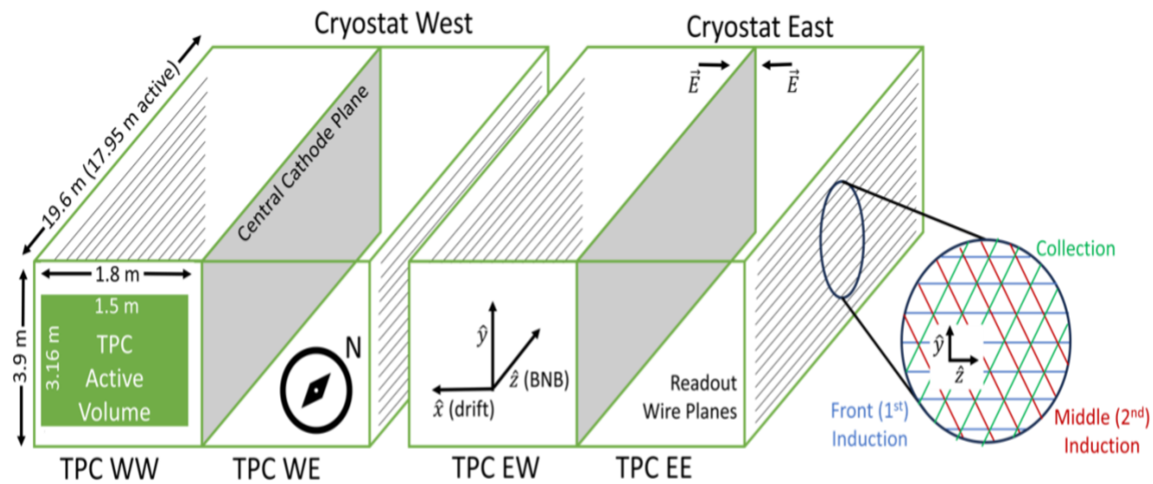


FIG. 1: The geometry of the ICARUS detector

2. NuGraph2 Neural Network

NuGraph2 is a machine learning Graph Neural Network (GNN) developed by the ExaTrkX collaboration for particle reconstruction in neutrino physics [4]. It has been implemented in MicroBooNE and will be implemented in ICARUS. Hits form planar nodes which have input features in the form of the hit's wire index, time, and the integral and Root Mean Square (RMS) width of its Gaussian pulse. Nexus nodes are made from spacepoints and have no input features. Planar nodes within each wire plane are connected to each other and to nexus nodes via graph edges created by a Delauney triangulation algorithm.

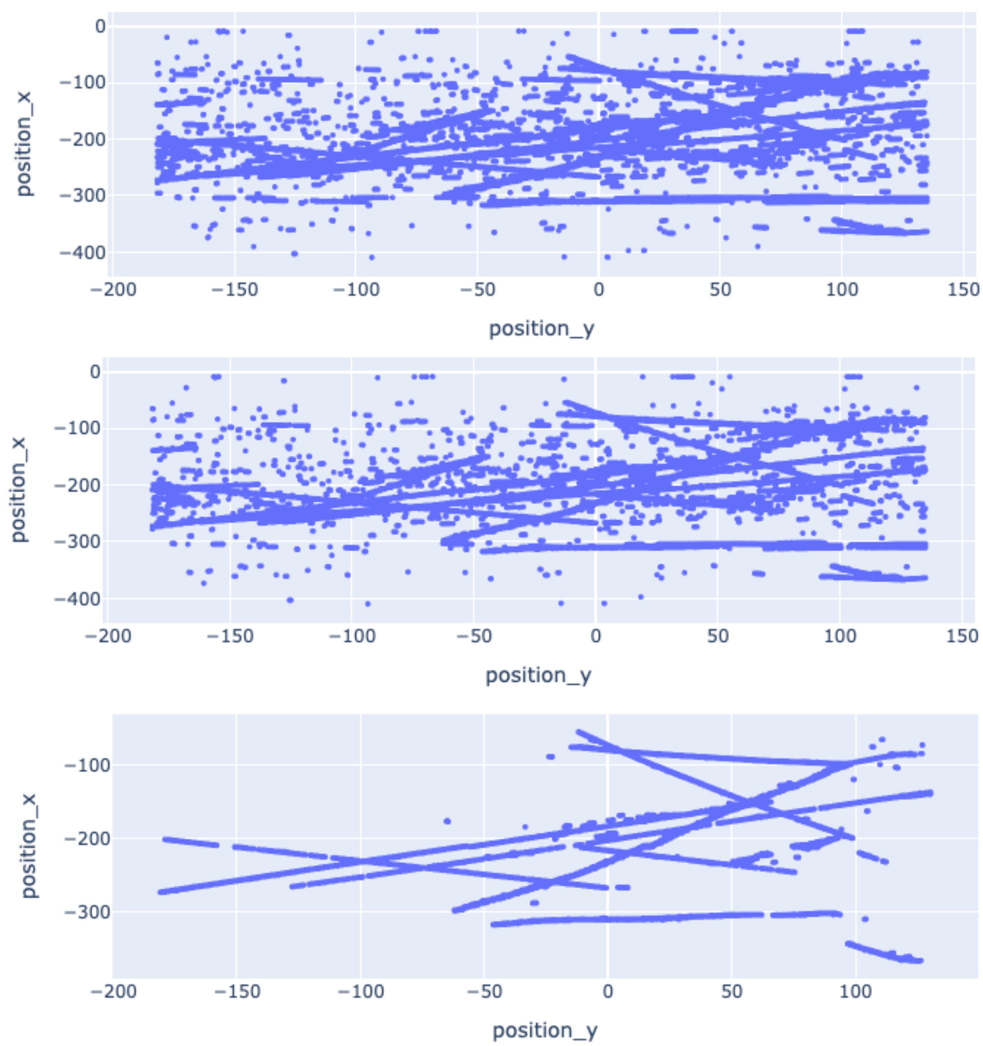
3. Spacepoint Filtering Techniques

The spacepoints from any event are cluttered with noise that obscure the ionization tracks and showers (signal feature) of the events. To address this, spacepoint filtering is adopted and two filtering techniques were investigated.

The first filtering technique tried was filtering spacepoints by their χ^2 value, which was already included in the spacepoint data produced by the HDF5Maker module of LArSoft [5]. A variety of χ^2 threshold values were investigated with lower threshold values eliminating more noise, however lowering the threshold value too much, such as $\chi^2 < 0.1$, resulted with removing spacepoints from the ionization tracks and showers without noticeable improvement in filtering noise compared to more modest value of χ^2 filters, suggesting there was a better technique.

The second technique investigated was filtering by the number of associated hits used to make spacepoints where there should be one hit from each of the three wire planes. Many spacepoints were only associated with two hits. Removing these spacepoints eliminated far more noise than sharp χ^2 filters without loss of data in the ionization tracks or showers. Because of these findings, filtering spacepoints by number of associated hits was added to the HDF5Maker module of LArSoft by an expert.

Using both filtering techniques together was also investigated with a variety of moderate χ^2 thresholds (0.5-1.5). Figure 2 below shows a comparison of the various filtering techniques. This was not added to the HDF5Maker module of LArSoft as it showed little improvement over filtering by associated hits, however χ^2 filtering was added to a later stage of the NuGraph2 workflow.



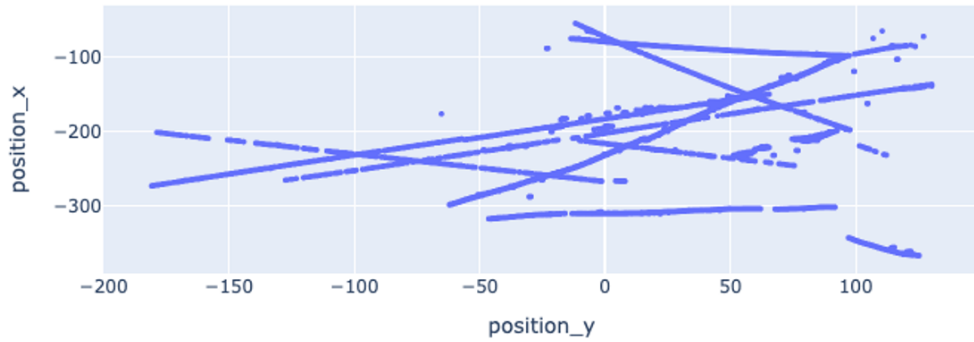


FIG. 2: 2D projection of spacepoints without filtering (top), with a χ^2 filtering of 0.5 (middle-top), with associated hit filtering (middle-bottom), with the two filtering techniques combined (bottom).

4. Event Samples

Three samples of simulated neutrino (Multi-Particle Vertex) and cosmic muon background (Multi-Particle Rain) events were used to train NuGraph2 for ICARUS. They had 112 runs (~3,500 events), 1183 runs (~31,000 events), and 10363 runs (~272,000 events) respectively. These will be referred to as the small, medium, and large samples for the rest of this paper.

Within a sample, all runs were made separately as their own files and had their run number equal to 1. A python script was used to give each run a unique run number.

Once the run numbers were modified, the run files are concatenated into one file. If the run numbers weren't made unique, runs would overwrite each other in the concatenated file

5. Preprocessing

Preprocessing turns the samples into the input for NuGraph2 training by assigning semantic labels to hits and creates planar and nexus nodes, which are made from hits and spacepoints respectively. Graph edges connecting the planar nodes to one another and to nexus nodes are then created via a Delauney triangulation algorithm. Events are then split into training, validation, and testing samples. Regardless of the entire sample size, 90% of events are put into training, 5% in validation, and 5% in testing samples.

Before the graph nodes and edges are made, a χ^2 cut of 0.5 is applied to the spacepoints, empty events are removed, and hits with too many semantic labels are removed. These were applied here instead of in LArSoft for efficiency and ease of implementation as making new samples wasn't required for testing different filter combinations.

It is of note that hits with too many semantic labels were not present in the small and medium samples and were exceptionally rare in the large sample. Furthermore, empty events were not present in the small sample and rarely seen in the medium sample after the χ^2 filter was added.

Ideally, the χ^2 filter and removal of empty events would be applied in LArSoft as it would allow for smaller file sizes and possibly make run-concatenation and preprocessing faster.

6. Training NuGraph2

When training begins, events are shuffled into equal sized batches, which are operated on in a random order. When NuGraph2 training finishes operating on a batch, it compares the output from the training to the truth and adjusts its machine learning parameters to minimize the difference between the output and truth. These updated parameters are used when operating on the next batch. Once all batches have been operated on, the process repeats for a set number of epochs. The batch order is different for each epoch to reduce bias towards any batch.

6.1 Hyperparameter Optimization

There are several parameters that affect the training results, duration, and resource requirements. These parameters are the learning-rate, number of epochs, and the batch-size. Investigations into hyperparameter optimizations discussed in this paper were only done on the small and medium samples as investigations with the large sample would have been inefficient and time consuming.

6.1.1 Learning-rate

NuGraph2 has a default learning-rate of 0.001, however loss plots from trainings on the medium sample with 160 epochs showed signs of possible undertraining, which was evident from the loss plot not saturating. This led to a short investigation for an optimized learning-rate on the medium sample. Training with learning-rate 0.002, 0.0015, and 0.00125 showed signs of overtraining. Eventually, training with learning-rate 0.00115 was done, which showed no signs of overtraining or undertraining, making it the optimized learning-rate, at least for the medium sample. The optimized learning-rate can change depending on the sample size. Figure 3 (below) shows the loss plots from the learning-rate investigation. The large sample was trained with learning-rate 0.001 as the learning-rate investigation wasn't finished at the time and there were concerns about time constraints.

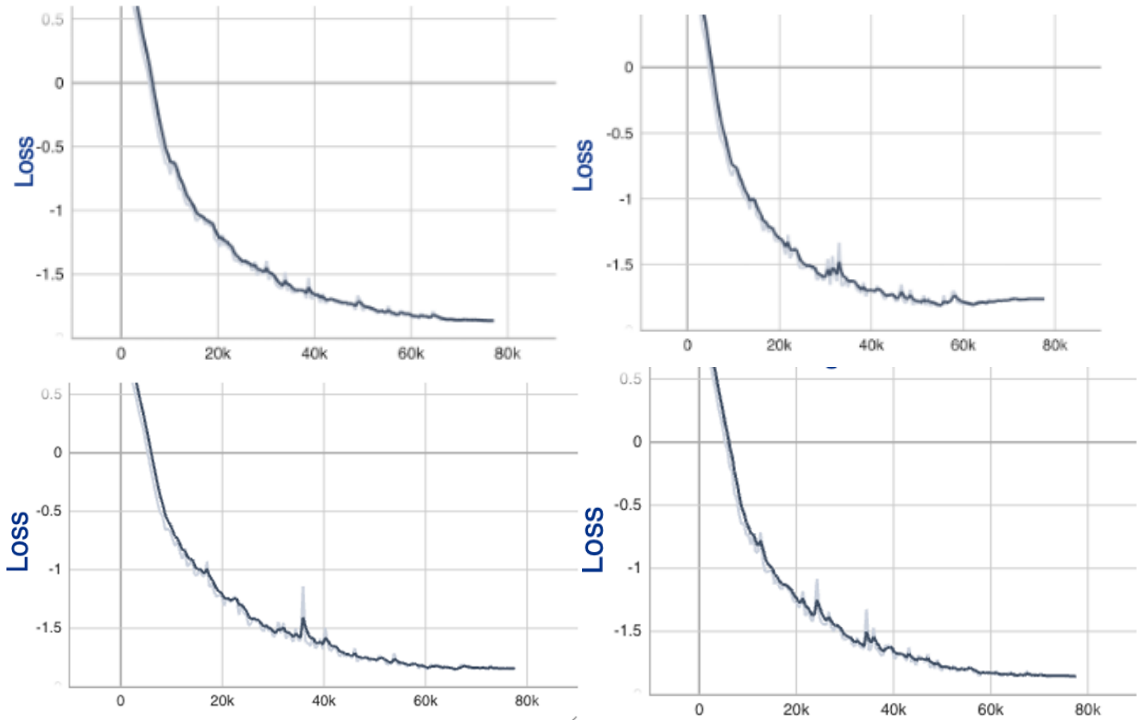


FIG. 3: Loss plots as a function of batches operated on from training the medium sample with learning-rate 0.001 (top-left), 0.0015 (top-right), 0.00125 (bottom-left), 0.00115 (bottom-right)

Table. 1: Tables comparing the values of precision and recall filter and precision and recall semantic for the train, validation, and test categories with different learning rates on the medium sample.

Learning-rate = 0.001					
prec_filter/train	0.9745	prec_filter/val	0.973	prec_filter/test	0.9724
recal_filter/train	0.9763	recall_filter/val	0.9748	recal_filter/test	0.9742
prec_sem/train	0.878	prec_sem/val	0.8755	prec_sem/test	0.8679
recall_sem/train	0.8882	recall_sem/val	0.8755	recal_sem/test	0.8679

Learning- rate = 0.00115					
prec_filter/train	0.9697	prec_filter/val	0.9731	prec_filter/test	0.9724
recal_filter/train	0.9753	recall_filter/val	0.9749	recal_filter/test	0.9743
prec_sem/train	0.8819	prec_sem/val	0.8771	prec_sem/test	0.8698
recall_sem/train	0.8819	recall_sem/val	0.8771	recal_sem/test	0.8698

Learning-rate = 0.00125					
prec_filter/train	0.972	prec_filter/val	0.9724	prec_filter/test	0.9727
recal_filter/train	0.9766	recall_filter/val	0.9748	recal_filter/test	0.9754
prec_sem/train	0.921	prec_sem/val	0.878	prec_sem/test	0.8774
recall_sem/train	0.921	recall_sem/val	0.878	recal_sem/test	0.8774
Learning-rate = 0.0015					
prec_filter/train	0.9745	prec_filter/val	0.9731	prec_filter/test	0.9725
recal_filter/train	0.9761	recall_filter/val	0.9747	recal_filter/test	0.9744
prec_sem/train	0.8925	prec_sem/val	0.8754	prec_sem/test	0.8694
recall_sem/train	0.8925	recall_sem/val	0.8754	recal_sem/test	0.8694

As shown in table 1 (above), training with learning-rate 0.00125 shows clear signs of overtraining as both the precision semantic and recall semantic values for the training sample is much higher than they are for the validation and test samples. The training with learning-rate 0.00115 is the most consistent between the training, validation, and test samples, further showing that it is the optimal learning-rate.

6.1.2 Epochs

Training iterates over many epochs and Nugraph2 uses 80 epochs by default. NuGrpah2 was trained many times using 160 epochs for both the small and medium samples. Doubling the epochs from 80 to 160 marginally improved all training results at the cost of extending the training duration by roughly 50%. The improvements were far weaker in the medium sample than in the small sample as statistical uncertainties were less present.

The large sample was trained with 80 epochs as improvements from having more epochs would be insignificant compared to those from the medium sample and training it with 80 epochs roughly takes 14 days, but this depends on the computer resources during training. For comparison, training the small and medium samples with 80 epochs takes 1.5 and 16 hours respectively.

6.1.3 Batch-size

The final hyperparameter is the batch-size, which defaults to 64 events per batch. Changing the batch-size doesn't show any noticeable differences in training performance, however it greatly influences the training time and resource requirements. As the batch-size increases, the training duration decreases, and the resource requirements increases. Training with larger batch-sizes wasn't investigated as using the default batch-size is rather resource intensive (~40 GB GPU). Smaller batch-sizes were only used for debugging purposes and to stay within resource availability when training the small sample before filters were added to preprocessing.

7. Results

At the end of each epoch, four confusion matrices are produced: recall filter, precision filter, recall semantic, and precision semantic. Recall and precision are defined as [4]:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \qquad \text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Where TP is true positives, FP is false positives, and FN is false negatives. Recall is more commonly referred to as efficiency and precision is more commonly referred to as purity. Filter matrices show how well signal and noise are distinguished from one another while semantic matrices show how well particles of different semantic classes are distinguished from one another. The semantic classes are minimum ionizing particle (MIP), highly ionizing particle (HIP), shower, Michel electron, and diffuse.

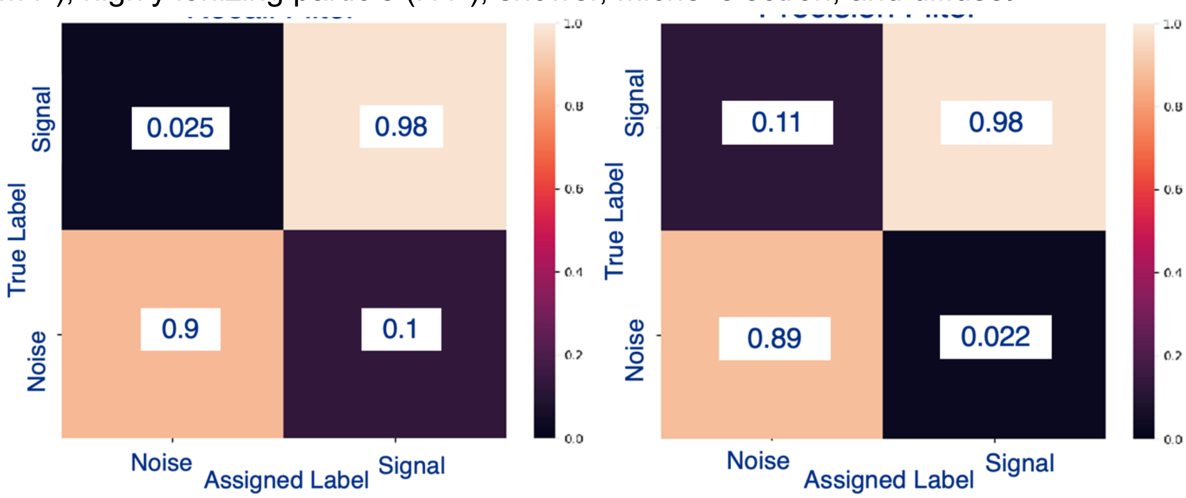


FIG. 4: Recall filter (Left) and Precision Filter (Right) confusion matrices from the large sample

As shown in Figure 4 (above), the filter matrices show reasonable results with 98 percent of the hits identified as signal truly being signal in both matrices and 90 and 89 percent of hits identified as noise truly being noise in the recall and precision filter matrices respectively There is some minor room for improvement with how NuGraph2 identifies noise, but overall, the results are promising.

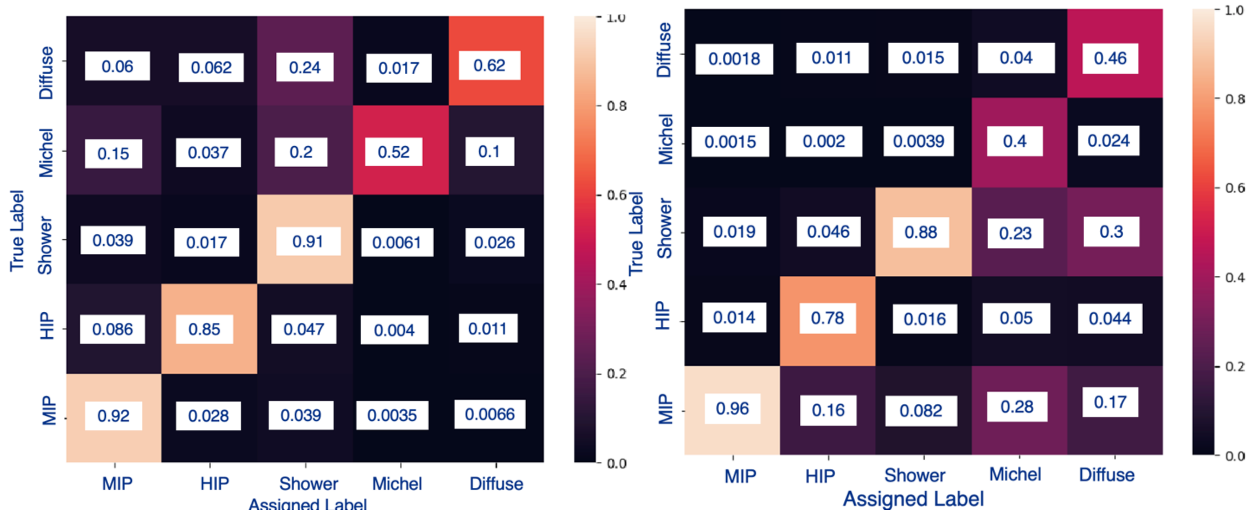


FIG. 5: Recall Semantic (Left) and Precision Semantic (Right) confusion matrices from the large sample

While the semantic matrices overall show promising results in Figure 5 (above), there is much to be desired with the Michel electron and diffuse categories as they should ideally have similar results to the other categories. One of the main reasons for the undesired performances would be because these categories contain far fewer hits than the other categories.

8. Next Steps

The results of the large sample will be used to further search for optimized hyperparameters, which will be the parameters where the Michel electron and diffuse categories have comparable results to the other categories. NuGraph2 will then be trained with these parameters before being integrated into ICARUS Pandora reconstruction.

Acknowledgements

This manuscript has been authored by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics.

This work was supported in part by the U.S. Department of Energy, Office of Science, Office of Workforce Development for Teachers and Scientists (WDTS) under the Science Undergraduate Laboratory Internships Program (SULI).

References

[1] A. A. Aguilar-Arevalo et al (MiniBooNE Collaboration) "Significant Excess of Electronlike Events in the MiniBooNE Short-Baseline Neutrino Experiment", Phys. Rev. Lett. 121, 221801 (2018); arXiv: 1805.12028

[2] M. Antonello et al, (2014), *A Proposal for a Three Detector Short-Baseline Neutrino Oscillation Program in the Fermilab Booster Neutrino Beam*, https://arxiv.org/pdf/1503.01520
[3] P. Abratenko et al, (2023), *ICARUS at the Fermilab Short-Baseline Neutrino Program - Initial Operation*, https://arxiv.org/pdf/2301.08634
[4] A. Aurisano, V. Hewes, G. Cerati, et al., "NuGraph2: A Graph Neural Network for Neutrino Physics Event Reconstruction", Phys. Rev. D 110 (2024) 3, 032008; arXiv:2403.11872
[5] E.L. Snider and G. Petrillo (2017) J. Phys.: Conf. Ser. 898 042057, *LArSoft: toolkit for simulation, reconstruction and analysis of liquid argon TPC neutrino detectors,* https://iopscience.iop.org/article/10.1088/1742-6596/898/4/042057/pdf