Technote: Neutrino Event Reconstruction with Deep Learning

Micah Groh
Indiana University

January 2021

Abstract

A machine learning approach for creating particle clusters has been developed using instance segmentation based on Mask R-CNN called Neutrino Event Reconstruction with Deep learning (NERD). NERD is an end-to-end algorithm which simultaneously clusters particles and identifies them. NERD reconstructs, on average, three additional true particles in each event and shows a 21% improvement in the average purity of clusters.

1 Introduction

One of the advantages of ML algorithms is that they can be designed to depend on only the most basic of reconstructed objects, the slice in the case of CVN. However, Prong CVN identifies the particle contribution to reconstructed prongs. Thus, its performance is dependent on the quality of those prongs, which is itself dependent on the reconstructed vertex.

An end-to-end algorithm which clusters particle contributions and identifies them, starting from the slice with no other dependencies, could show significant improvements. The field of computer vision calls such an algorithm instance aware semantic segmentation or simply instance segmentation, which combines two computer vision methods into a single algorithm. The first is object detection [1], an algorithm for finding the locations of objects within an image. The second is semantic segmentation [2], a pixel-wise classification technique.

2 Network

The Mask R-CNN [3] algorithm for instance segmentation was adapted for use by NOvA. A diagram of the network architecture is shown in Fig. 1.

The network has two inputs: an input pixel map for the slice and a set of anchors to act as candidate bounding boxes. Only one view of the event is evaluated by the network at a time. The pixel map used is similar to those used for event CVN. However, the pixel map is upscaled by a factor of three in width and height so that each hit in the event translates to nine pixels in the input array. Upscaling greatly improves the networks ability to reconstruct particles with only one or two hits. A second channel is added to the input array which is identical, but is fixed at a value of 255 for each hit. This channel enhances the network's ability to locate hits with low energies and improves the completeness of the final particle clusters. Finally, it is beneficial in the feature extractor, described below, for the input to have dimensions with powers of two. The downstream most 15 planes are removed and the input is padded with zeros. The resulting final shape is $256 \times 256 \times 2$.

The first element of the architecture is a CNN based on a residual network with 50 layers [5]. Whenever the input shape is reduced, either by pooling or a strided convolution, the feature map is saved to act as one of five levels of the feature map. The five levels are used in a feature pyramid network [6] which upsamples one level of the feature map to be element-wise added with the level above. The result is a set of five feature maps which capture features of different sizes.

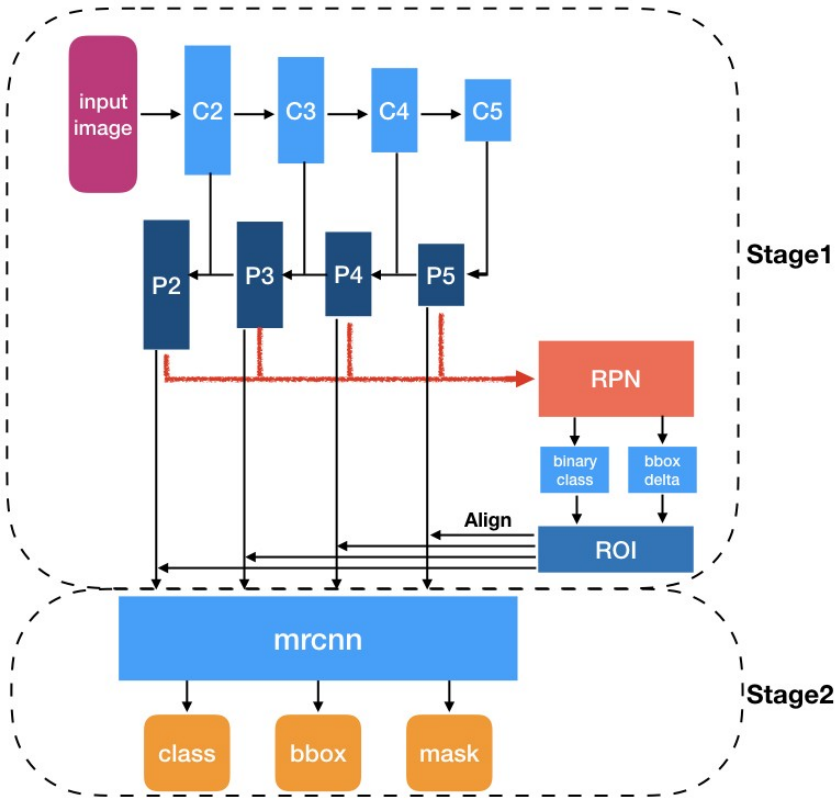

Figure 1: Mask R-CNN Diagram. The architecture proceeds in two stages. The first stage extracts a set of features from the input image. Then, an object detection network locates bounding boxes which, ideally, each contain all pixels from a single object. The second stage applies a position and size correction to the bounding box, classifies the object within the bounding boxes, and identifies which pixels belong to the object. Figure reproduced from [4].

The second input to the network are the anchors, boxes in the image space which will be used to make bounding boxes. Three anchors are generated for each pixel in each level of the feature map. These anchors come from three different width/height ratios such that there is a square anchor, an anchor with half the width and double the height, and an anchor with double the width and half the height. The anchor shapes for each feature level are 8×8, 16×16, 32×32, 64×64, or 128×128. In total, 16368 anchors are used.

The feature maps are then used to assign a score between 0 and 1 for each anchor using a region proposal network (RPN) [7]. A score of 1 signifies that that anchor contains a particle which will be reconstructed. The RPN also applies a correction to the size and position of each anchor to make bounding boxes for each particle.

Given the number of anchors analyzed, it is likely that multiple anchors are found for each particle. Any pair of anchors with large overlap[1] will have the anchor with lower score removed from consideration for the remainder of the algorithm. At this step, there is ideally one, and only one, anchor for each particle in the event which will make the bounding boxes used in the second stage of the architecture.

The second stage does three steps in parallel. The first is to classify the particle contained within each bounding box. The second is to correct the size and shape of the bounding once again based on the identified

[1]The overlap is defined by the ratio of the interesting area, the area common to both anchors, to the union, the total area of both anchors. This is often called the intersection over union (IOU).

class. The third is to assign a score to each pixel within the bounding box based on how likely it is to have been produced by a particle of that type. The optimization of the network while learning is to improve all three of these tasks.

There are seven possible classes to be identified in this algorithm. Five are the same used in the particle classifier: electron, muon, photon, proton, or charged pion. A bounding box can also be classified as background, usually the result of clustering only detector noise. The final category consists of any other particles. Since this algorithm is capable of making clusters out very few hits, it is probable that some are primarily from delta rays or kaons which will make up most of this category.

To conserve memory, the bounding boxes are reshaped to a 28x28 pixel mask within the network. The use of masks greatly reduces the computational load of the network when the bounding boxes are expected to be large. It also provides a fixed shape as the intermediate input to the component of the network which computes the mask scores.

Thus, the network has three outputs: the set of scores for each class, a set of bounding boxes, and a mask for each bounding box. The bounding boxes give coordinates for each particle in the event and the pixel scores identify which hits should be included into a cluster. The masks must be reshaped to the size of the bounding box and then placed on the appropriate location in the event to act as the initial set of clusters. These initial 2D clusters must go through post-processing to create the final set of prongs. First, hits outside of the pixel map boundary must be included into the particle clusters. It is assumed that any hits outside the pixel map will not belong to any new particles. Thus, Prim's algorithm, can be used to iteratively add any unclustered hits to the nearest cluster up to 80 cm away. In addition, the step which removes overlapping anchors will sometimes let through an anchor that does not contain a unique particle, particularly when it identifies a different class for the spare anchor. Electromagnetic showers with a large gap will often produce two bounding boxes, one box containing the entire shower and a second containing just the subset downstream of the gap. Whenever this is seen, the smaller of the two bounding boxes is merged with the larger bounding box. The final step is to merge the results for each view using the Kuiper test, identically to the view matching used by FuzzyK [8]. The prong directions are defined from the reconstructed vertex by elastic arms.

3 Performance

Instance segmentation offers a number of clear advantages over FuzzyK and Prong CVN. It is an end-to-end algorithm with no reconstruction dependencies which would affect its performance. It is capable of reconstructing particles resulting from secondary activity in an event, such as a muon decaying into a michel electron. The traditional reconstruction is limited to reconstructing particles with some minimum energy deposited in the detector, but instance segmentation can reconstruct particles even with only one hit. An example event display depicting prongs produced by the algorithm is shown in Fig. 2.

The most interesting improvement in the particle clustering performance is in the number of reconstructed clusters. As mentioned, instance segmentation is capable of producing clusters even with only one hit, so many additional true particles make it into their own cluster. The number of additional true particles in a cluster compared to the traditional reconstruction is shown in Fig. 3. As shown in the figure, instance segmentation reconstructs, on average, almost three additional true particles in each event.

The efficiency and purity of the prongs is shown in Fig. 4. Instance segmentation shows an improvement in the average prong efficiency from 81% to 84% and an improvement in the average purity from 66% to 80%, a 21% improvement in purity. In particular, the fraction of prongs with 100% purity has been greatly improved at a comparable efficiency. In addition, improvement is seen in every particle type. The purity plots in Fig. 4 show a discontinuity at 0.5. Prongs labels are identified by the highest energy contributor, so a purity less than 0.5 is only possible with three particles contributing to a prong. The steeper discontinuity at this point shows that fewer particle are usually being clustered together, this is particularly evident in the photon purity distribution.

It is important to consider the performance of the algorithm under a variety of event topologies. As an event becomes more "complex", the difficulty in reconstructing particles increases. Figure 5 shows the

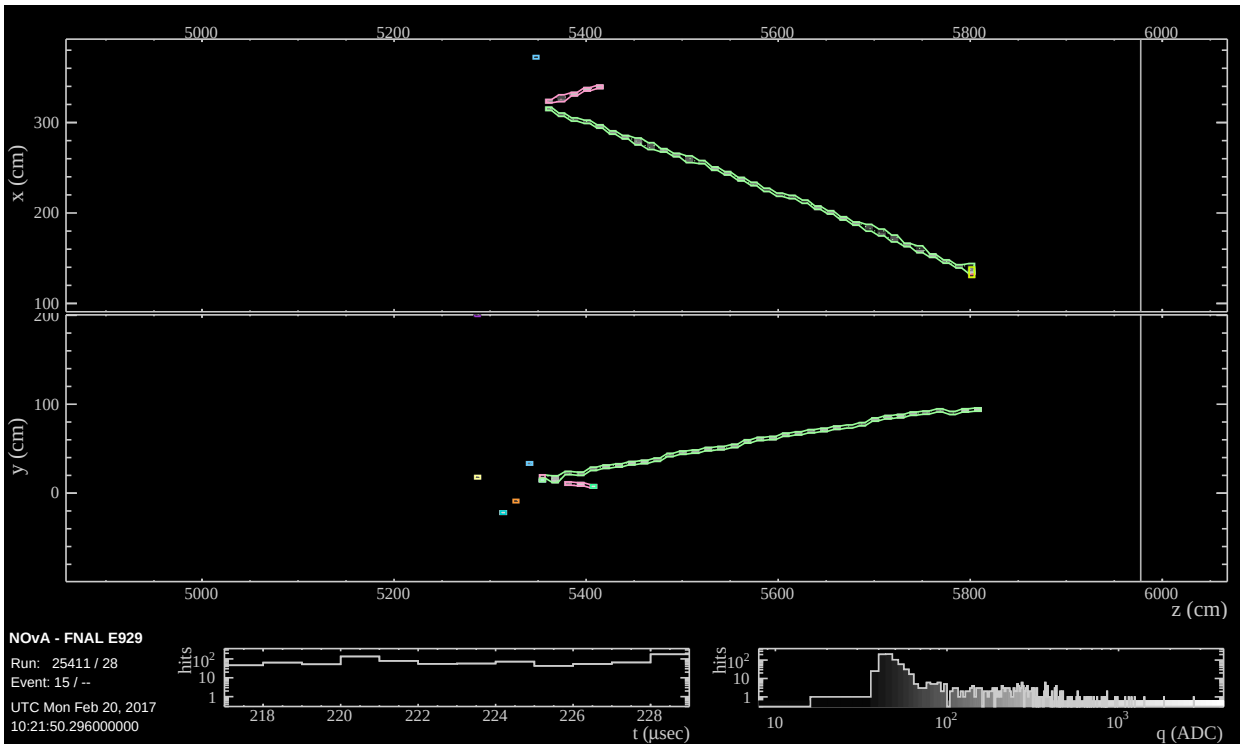

Figure 2: Instance Segmentation Event Display. A ν_μ CC event with the prongs produced by instance segmentation overlaid. All hits make it into an appropriate cluster. The yellow cluster at the end of the muon track in the X-view is a clustered michel electron.

average purity of clusters separated by the number of primaries produced in the neutrino interaction. Here, the number of primaries is being used as a proxy for the complexity of the event. Not only does instance segmentation produce higher quality prongs for all topologies, but the improvements are more significant for more complex events.

To evaluate the performance of the particle classification, we once again consider the highest score output by the network and construct classification matrices. These can be seen in Fig. 6. A new feature in these plots is the confusion between protons and photons. While high energy protons and photons are topologically very different, at low energies they are very similar, often producing only a single hit increasing the challenge of the classification. These single hit clusters are particularly common from neutron interactions which will either collide with a nucleus to make a proton or be captured and emit several low energy photons. Also seen in the plot is a large misclassification between charged pions and protons. The training sample was balanced to have approximately equal numbers of ν_e CC, ν_μ CC, and NC, but was not balanced for the individual particle types. Charged pions are produced more rarely in neutrino interactions and as such, the network did not fully learn the topology. Future iterations of this algorithm should either balance or weight the sample to remedy this.

4 Production

Coming soon...

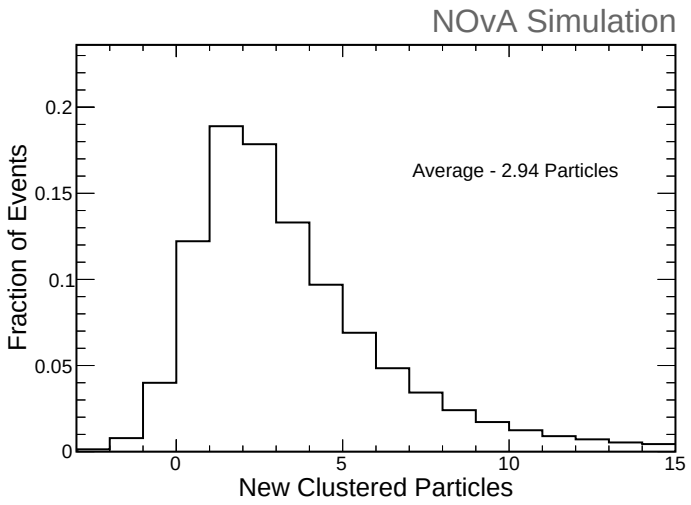

Figure 3: Reconstructed True Particles. The difference in the number of true particles that have their own reconstructed cluster between the k-means algorithm and instance segmentation.

5 Future Improvements

The following thoughts are ordered approximately from easiest to most challenging to implement.

5.1 Class Balancing

As mentioned in the text above, the training sample for the network was balanced to have approximately equal numbers of ν_e CC, ν_μ CC, and NC. However, as has been seen with past variants of prong CVN, this results in a greatly imbalanced sample of the individual particles [9]. When moving to a balanced sample, a significant improvement was seen in the classification performance, particularly for pions, from Prong CVN when retraining for 2020 and similar results would be expected here. The simplest way to address the imbalance would be to weigh each particle class to be approximately equal. This is easy to implement, a configuration setting in the training scripts, but would require completely retraining the network.

Other considerations relating to the class scheme would be to use only three major categories: electromagnetic (photons and electrons), hadronic (protons and pions), or muons. Another would be to add in a neutron daughter category as photons emitted from neutron capture are substantially different from photons in $\pi^0 \to \gamma\gamma$ decays and result in the mis-classification seen between photons and protons.

5.1.1 View Matching

The view matching used is identical to the Kuiper test used for FuzzyK. The algorithm is designed to match one and only one prong from one detector view to one prong in the other view. This strategy works well except in the case where two particles are overlapping in one view leaving an orphaned prong in the other. Cathal Sweeney has done work already to improve the matching [10] which could be incorporated after validation.

A more challenging approach would be to modify the network to propose candidate particles across both views, but the approach for this is not clear.

5.2 Timing

The timing of hits could aid in the particle classification and separating clusters from overlapped hits. This is notable for michel electrons at the end of muon tracks and neutron daughters which tend to be separated

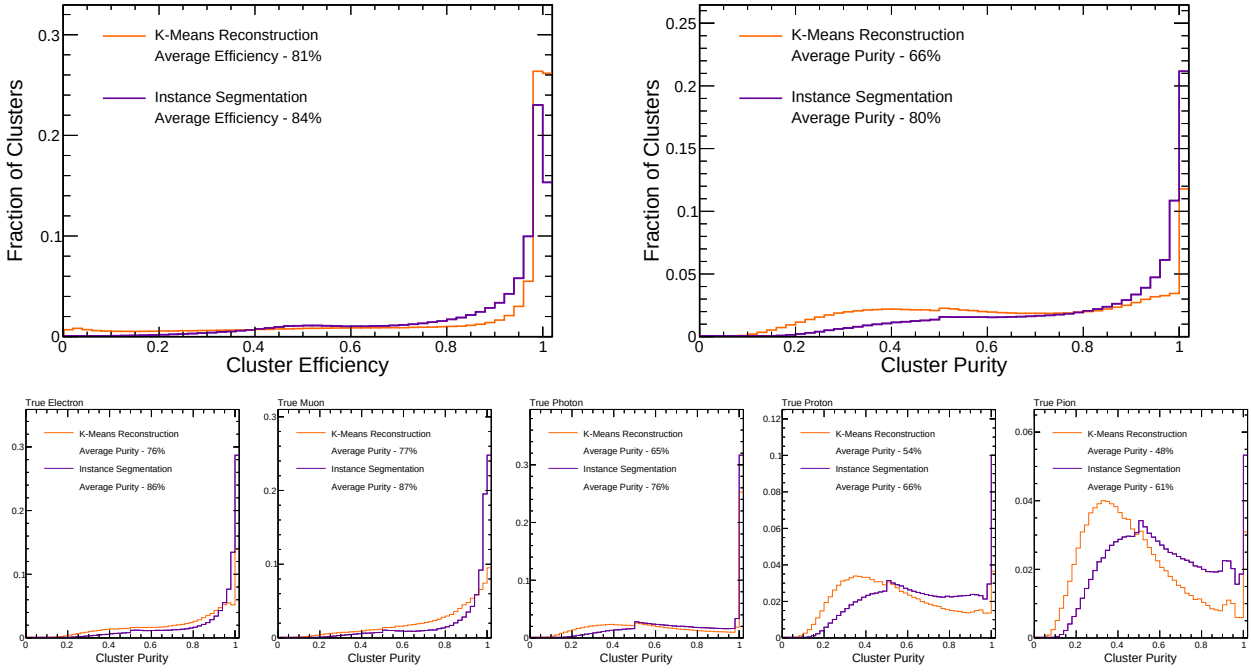

Figure 4: Prong Quality Comparison. Top: The efficiency (left) and purity (right) of reconstructed prongs using instance segmentation and the traditional k-means reconstruction. Bottom: The purity of prongs using instance segmentation for true particles. From left to right: electron, muon, photon, proton, and charged pion.

in time from the rest of the event.

Code already exists in *CVNMapper* to produce a pixelmap with the timing of each hit, it just needs to be turned on and then incorporated into the h5cafs and training files. It is not clear the best way to incorporate the timing in the network's inputs, but the simplest would be as a third channel of the input image.

5.3 Vertexing

The clusters produced by the module are associated with the elastic arms vertex to make prongs. Since the clusters themselves have no relation to the elastic arms vertex, the vertex location is sometimes bizarre. The vertex can sometimes be in the middle of a prong, a common occurrence with ν_μ CC QE events. In other cases, it makes little sense to attach some prongs to the vertex, such as pions which hard scatter into a new direction.

The vertex is needed to compute prong directions and, when combined with a tracking algorithm like break point fitter, to estimate particle energies or other properties. A new module which starts from the clusters to produce a vertex could show improvement over the current method: vertex (elastic arms)→clusters (FuzzyK). Secondary vertices could also be considered at the same time.

5.4 Hit Sharing

The network inputs assume that each hit has a unique identity, the particle identity which is the highest energy contributor to the hit. However, the network is capable of assigning a hit to multiple to clusters and, indeed, does so anyway in some cases such as placing a proton at the vertex of a muon track. Note that the network learned to do this without ever being shown that this was possible.

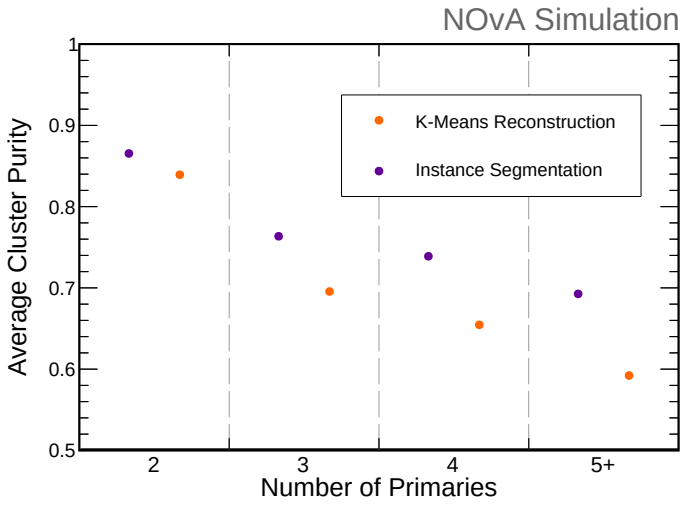

Figure 5: Event Complexity. The average purity of prongs reconstructed by instance segmentation and k-means separated by the number of primaries produced in the neutrino interaction.

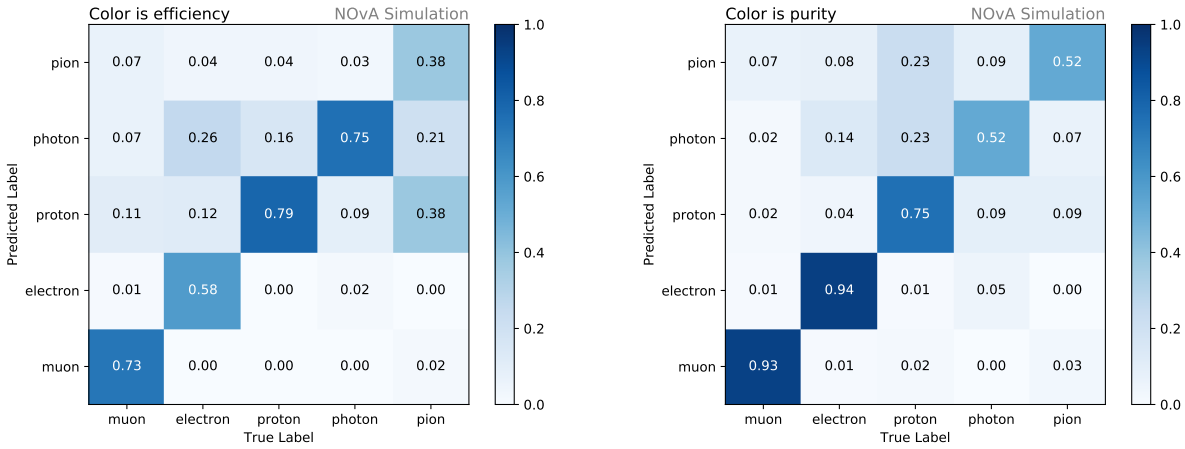

Figure 6: Instance Segmentation Classification Matrix. Prongs reconstructed by instance segmentation are used to fill a matrix with the X-axis showing the true identity and the Y-axis show the largest scoring category output from the network. Left: The diagonal shows the efficiency for selecting each particle and the off-diagonal values shows how particles are misclassified. Right: The diagonal shows the purity of each selected sample and the off-diagonal shows the backgrounds to each selection.

The construction of the network training targets could be modified to teach the network properties of hit sharing. Each hit would have a vector of particle contributors as its true identity.

References

1. Jiao, L. *et al.* A Survey of Deep Learning-Based Object Detection. *IEEE Access* **7,** 128837–128868. ISSN: 2169-3536. `http://dx.doi.org/10.1109/ACCESS.2019.2939201` (2019).
2. Ulku, I. & Akagunduz, E. *A Survey on Deep Learning-based Architectures for Semantic Segmentation on 2D images* 2020. arXiv: `1912.10230 [cs.CV]`.

3. He, K., Gkioxari, G., Dollár, P. & Girshick, R. *Mask R-CNN* 2018. arXiv: 1703.06870 [cs.CV].
4. Zhang, X. *Simple Understanding of Mask RCNN* Apr. 2018. %5Curl%7Bhttps://alittlepain833.medium.com/simple-understanding-of-mask-rcnn-134b5b330e95%7D.
5. He, K., Zhang, X., Ren, S. & Sun, J. *Deep Residual Learning for Image Recognition* 2015. arXiv: 1512.03385 [cs.CV].
6. Lin, T.-Y. *et al. Feature Pyramid Networks for Object Detection* 2017. arXiv: 1612.03144 [cs.CV].
7. Ren, S., He, K., Girshick, R. & Sun, J. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks* 2016. arXiv: 1506.01497 [cs.CV].
8. Niner, E. D. *Observation of Electron Neutrino Appearance in the NuMI Beam with the NOvA Experiment* PhD thesis (Indiana U., 2015).
9. Psihas, F. *et al.* Context-Enriched Identification of Particles with a Convolutional Network for Neutrino Events. *Phys. Rev. D* **100,** 073005. arXiv: 1906.00713 [physics.ins-det] (2019).
10. Sweeney, C. *Weighting hits with new view matching* NOvA Internal Document, DocDb 48495.