Proceedings of CTD 2020
PROC-CTD2020-44
September 11, 2020

Data Reconstruction Using Deep Neural Networks for Particle Imaging Neutrino Detectors

François Drielsma*, Laura Dominé, Dae Heun Koh, Kazuhiro Terao

On behalf of the DeepLearnPhysics collaboration,
Elementary Particle Physics division
SLAC National Accelerator Laboratory, Menlo Park, CA, 94025, USA

ABSTRACT

A Liquid Argon Time Projection Chamber (LArTPC) is a type of particle imaging detector that can record images of charged particle trajectories with high ($\sim$ mm/pixel) spatial resolution and calorimetric information. At the intensity frontier of high energy physics, LArTPC is the detector technology of choice for a number of experiments including the Short Baseline Neutrino program and the Deep Underground Neutrino Experiment which require high precision neutrino oscillation measurements to answer some fundamental questions about the universe. However, the analysis of detailed particle images can be difficult and building a high quality data reconstruction chain for large scale (over 100 tonnes) LArTPC detectors remains challenging. The research team at SLAC leads the R&D of a full Machine Learning (ML) based data reconstruction chain for LArTPC detectors. Our chain is a multi-task network cascade that performs pixel feature extraction (semantic segmentation using Sparse U-Net with ResNet modules), particle start and end point prediction (Point Proposal Network), pixel clustering for particle instance identification (custom convolution and instance attention layers), and particle flow analysis using Graph Neural Networks (GNNs). The output of the chain is a fully reconstructed event that can be used by physicists to infer neutrino oscillation physics. This R&D takes a significant step forward from the current state of the art in experimental neutrino physics. In this talk, we present the development of our reconstruction chain using an open data set. Our software is made publicly available to improve the reproducibility and transparency of our research work.

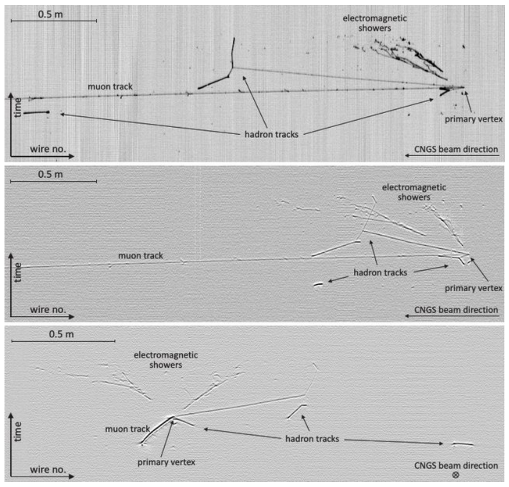
PRESENTED AT

Connecting the Dots Workshop (CTD 2020)
April 20-30, 2020

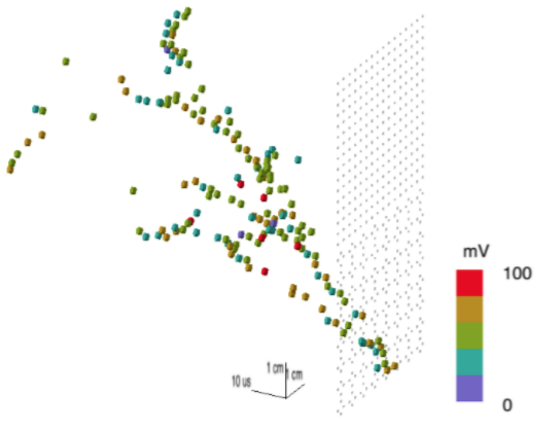
*drielsma@stanford.edu

1 Introduction

In recent years, the accelerator-based neutrino physics community in the United States has moved to employ Liquid Argon Time Projection Chambers (LArTPCs) as a central neutrino detection technology [1]. Charged particles that traverse the detector ionize the noble liquid. The electrons so produced are drifted in a uniform electric field towards a readout plane. The location of electrons collected on the anode combined with their arrival time offers mm-scale resolution images of charged particle interactions, as illustrated in figure 1. Wire-based and pixel-based readout technologies produce either three 2D projections – which can be combined to form a 3D image using tomographic reconstruction – or natively record a single 3D image, respectively.



(a)



(b)

Figure 1: (a) Three 2D projections of a neutrino interaction recorded by the ICARUS detector in the CNGS beam [2]. (b) Single 3D image of a electromagnetic shower recorded by the LArPIX detector [3].

The Short Baseline Neutrino Program (SBN) aims to clarify an anomalous signal observed by the Mini-BOONE experiment by using three $\mathcal{O}(100)$ t scale LArTPCs placed at varying short distances from a neutrino source [4]. The DUNE experiment is a future project that will use the LArTPC technology to measure electron neutrino appearance at a long baseline with unprecedented accuracy [5]. It will consist of a small pixel-readout near detector (105 t) and an enormous wire-readout far detector (40 kt).

Both of these physics endeavors centrally depend upon the efficient identification and accurate reconstruction of neutrino interactions. Traditional pattern recognition techniques are heavily challenged by the extraordinary level of detail LArTPCs images exhibit. Machine Learning (ML) is at the forefront of computer vision and offers a way to cover the large phase space necessary for this task. This paper presents a novel end-to-end ML-based reconstruction chain for LArTPCs. The following sections detail the architectures of the network modules developed for the hierarchical extraction of physics features from an image. Each stage of the reconstruction is trained and tested independently on a data set of 125280/22439 (train/test) images of size 768^3 voxels capturing a realistic density of particle interactions in a $12\,\mathrm{m}^3$ volume of LAr.

2 Semantic Segmentation and Point Proposal

2.1 Architecture

The first module in the reconstruction chain is designed to extract voxel-level information in 3D images: the particle type of each voxel [6] and the location of important points [7]. These two tasks share a common backbone architecture called “U-ResNet”, a hybrid of two popular architectures: U-Net [8] and ResNet [9]. U-ResNet is schematically represented on the top of figure 2; it is an autoencoder-type network which

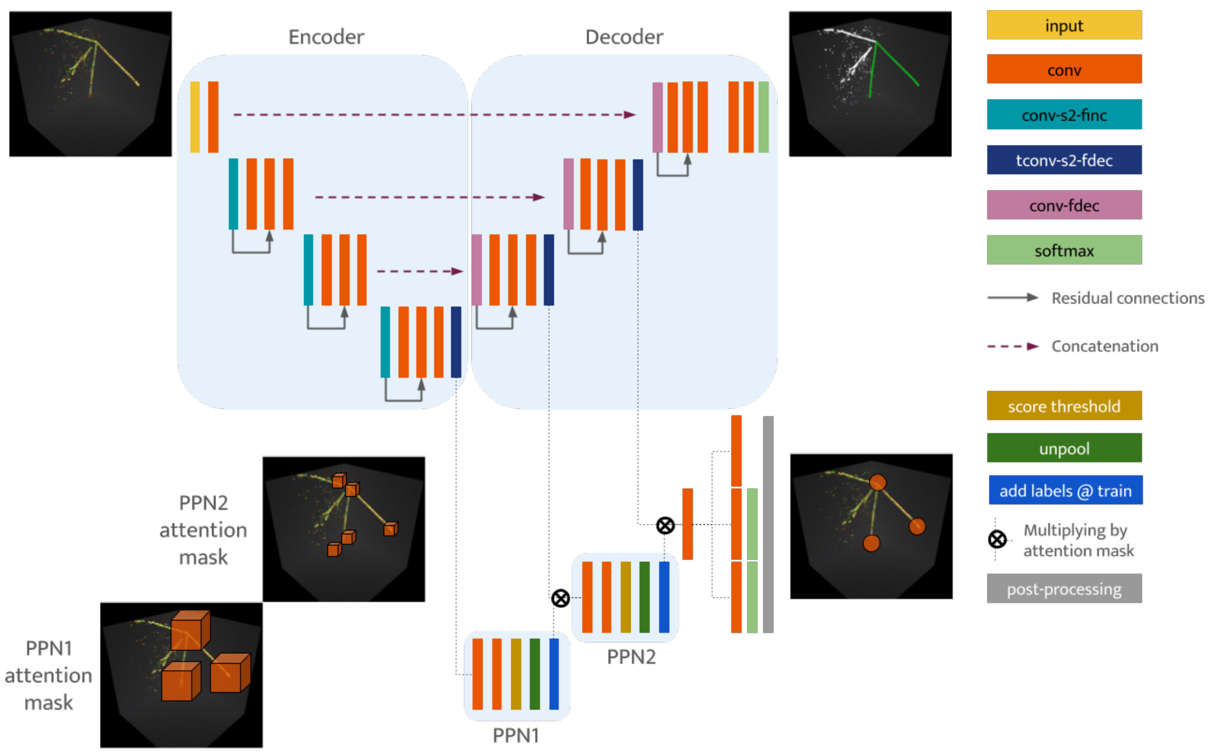


Figure 2: U-ResNet+PPN architecture for semantic segmentation and point proposal. Turquoise boxes are convolutions with stride 2 that increase the number of filters. Dark blue boxes are transpose convolutions with stride 2 that decrease the number of filters. Purple boxes are convolutions with stride 1 that decrease the number of filters. The spatial size of feature maps is constant across the horizontal dimension.

consists of two components. The first part is the encoding path: the image resolution is downsampled using strided convolutions while increasing the features dimension, thus learning from the input data at different scales. The number of down-sampling operations is referred to as *depth*. The second part is called the decoding path: it takes the low-spatial size, highly compressed features and up-samples them back to the original image resolution while decreasing the number of features at each step back to the initial number of features. The output layer predicts a score for each of the target particle classes: heavy ioninizing particle (HIP), minimum ionizing particle (MIP), electromagnetic shower, delta rays and Michel electrons. In this section, the results are shown for a U-ResNet of depth 6 with 16 features at the highest resolution and a linear increase in the number of features at each downsampling step.

In parallel to the U-shaped network, additional layers are introduced at three spatial resolutions to form the so-called Point Proposal Network (PPN), as shown at the bottom of figure 2. These layers (PPN1–3) are attached to the decoding path, as it contains all the information extracted at each resolution it is connected to. At each spatial resolution and starting at the deepest level, PPN layers attempt to predict a positive score for voxels that contain a ground-truth PPN point. Positive voxels form a mask that is then applied to the following PPN layers. At the training stage, true-positives in the attention masks are added to both the PPN1 and PPN2 layers to be able to back-propagate through them. For each voxel that has been selected through these successive attention masks, the final layer, PPN3, uses 3×3 convolutions which predict:

- a set of coordinates with respect to the voxel center;
- a proximity score representing the likelihood of being less than 5 voxels from a ground-truth point;
- a type score related to the probability of belonging to a certain particle class.

2.2 Loss definition

The semantic segmentation part of the loss, $\mathcal{L}_{seg}$, is a simple cross-entropy classification loss on the class scores predicted by the network. For the PPN component, at any spatial resolution strictly lower than the original image resolution, *positive* voxels are defined as voxels containing at least one true label point. *Negative* voxels do not contain any true label points. At the original image resolution, among all voxels $\{\boldsymbol{x}_i\}_{i=1}^{n}$, the subset of positive voxels, A, is defined as voxels within a certain distance threshold $d_{positive} = 5$ voxels from the true label points $\{\boldsymbol{q}_j\}_{j=1}^{n_t}$; other voxels are labeled as negative. The PPN loss consists of

- a classification loss at each PPN layer averaged over all voxels. With $y_{k,i} \in \{0; 1\}$ the classification label and $p_{k,i}$ the predicted probability for voxel i in layer k to be positive, the loss reads

$$\mathcal{L}_{class} = - \sum_{k=1,2,3} \frac{1}{n_k} \sum_{i=1}^{n_k} y_{k,i} \log(p_{k,i}) + (1 - y_{k,i}) \log(1 - p_{k,i}); \tag{1}$$

- an L_1 distance loss on the predicted positions of true positive voxels at the last layer. The raw point predictions of the network, $\{\boldsymbol{r}_i\}_{i=1}^{n_3}$, representing shifts with respect to the center of the voxels, $\{(0.5 + \boldsymbol{x}_i)\}_{i=1}^{n_3}$, are compared with the closest ground truth points, $\{\boldsymbol{q}_j\}_{j=1}^{n_t}$, in a loss of the form:

$$\mathcal{L}_{dist} = \frac{1}{n_3} \sum_{i=1}^{n_3} \mathbb{1}_A \min_{1 \le j \le n_t} \|\boldsymbol{r}_i + 0.5 + \boldsymbol{x}_i - \boldsymbol{q}_j\|_1; \tag{2}$$

- a type prediction cross-entropy loss at the last layer. The predicted point type is compared with the semantic type of the closest ground truth point. With n_c the number of semantic classes, $\{\boldsymbol{y}_i\}_{i=1}^{n_3}$ one-hot encoded n_c-vectors indicating to which class the point belongs, and $\{\boldsymbol{p}_i\}_{i=1}^{n_3}$ the n_c-vectors of predicted probabilities that the point belongs to each class, the loss is defined as

$$\mathcal{L}_{type} = -\frac{1}{n_3} \sum_{i=1}^{n_3} \mathbb{1}_A \sum_{c=1}^{n_c} y_{i,c} \log(p_{i,c}). \tag{3}$$

The loss used for optimization is the sum of all the aforementioned losses, i.e. $\mathcal{L} = \mathcal{L}_{seg} + \mathcal{L}_{class} + \mathcal{L}_{type} + \mathcal{L}_{dist}$.

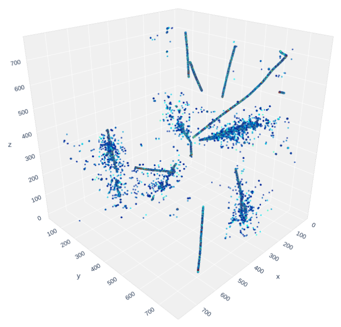
2.3 Performance

The first two panels of figure 3 show an example set of predictions made on an event in the test dataset. The semantic predictions are obtained by taking the argmax of the segmentation scores. Point proposals are reconstructed by applying the following post-processing to the set of predictions given by the network:

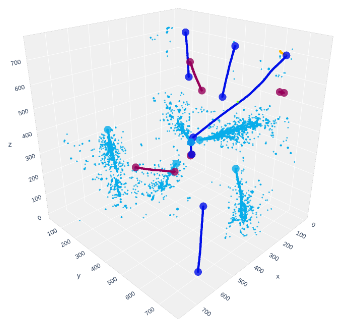
1. remove points with a ‘positive’ score below 0.5;
2. run DBSCAN [10] with a distance scale of $\epsilon = 1.999$ voxels on the remaining point predictions;
3. pool the points that belong to a common cluster by taking their mean position and max class;
4. remove points of class c that are further than 2 voxels away from any voxel of semantic class c.

The rightmost panel of figure 3 shows the semantic segmentation confusion matrix. All the classes are identified with a high level of precision, with HIPs, MIPs and Shower being classified with a voxel-wise accuracy above 98 %. The most significant sources of confusion are delta rays and Michel voxels identified as MIP. This is understandable considering that delta rays and Michel always either overlap or touch MIPs.

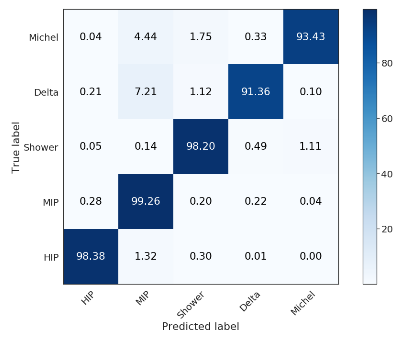
The left panel of figure 4 shows the distribution of distance from a true label point to the closest predicted point; 97.8 % of the true label points are within 10 voxels of a predicted point. The right panel of figure 4 shows the distribution of distances from a predicted point to the closest true label point; 97.8 % of the predicted points are within 10 voxels of a true label point. For both of these figures, true label points of type delta rays and predicted label points of type delta rays – defined as having a delta class score ≥ 0.1 – are not included. The fraction of true points that are more than 5 voxels away from any predicted point is 2.6 %, 1.4 %, 5.6 % and 1.2 % for the HIP, MIP, Shower and Michel true point types, respectively.



(a)

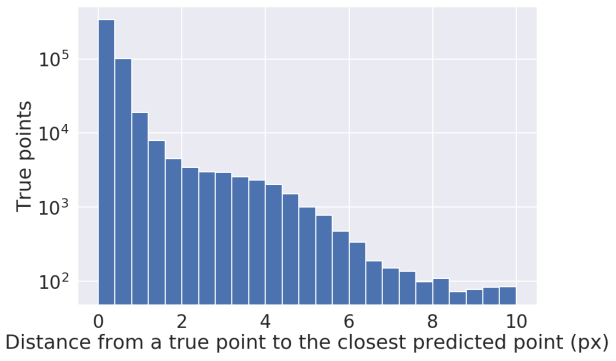


(b)

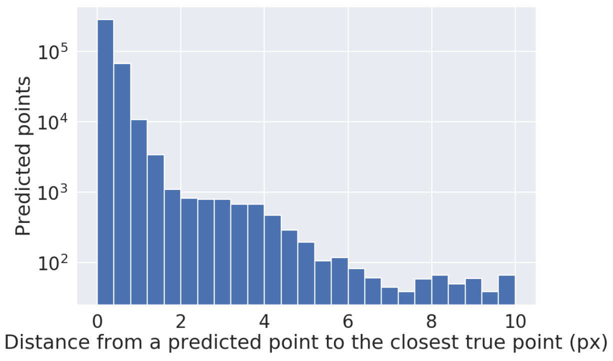


(c)

Figure 3: (a) Energy deposits from charged particle trajectories, whose color corresponds to an energy scale. (b) Semantic segmentation of the image into five classes: HIP in purple, MIP in dark blue, showers in light blue, delta rays in green and Michel electrons in orange. The large points represent PPN predictions. (c) UResNet semantic segmentation confusion matrix. Each row sums to 100 %.



(a)



(b)

Figure 4: (a) Distance from a ground-truth point to the closest predicted point. (b) Distance from a predicted point to the closest ground truth point. Points of type delta are excluded from (b) and (c).

3 Dense Particle Clustering

3.1 Architecture

The third task in the reconstruction chain involves clustering voxels into different dense particle instances using a CNN. The method described in this section follows the construction in [11] and uses a U-ResNet with two decoders as shown in figure 5. The input 3D image is passed through a shared encoder and then expanded into two output feature planes. The first feature plane is referred to as the *embedding* layer and the second as the *seediness* layer. The embedding layer learns a coordinate transformation of the input image voxel coordinates such that points that belong to the same particle are embedded close to one another. The seediness layer quantifies how likely a given voxel is to be close to the centroid of embedded points that share the same particle id. Once a reasonable transformation that groups each voxels into localized clusters in embedding space is obtained, information from the two feature maps are combined in a post-processing stage where cluster labels are assigned in a sequential manner, as detailed in the following subsections.

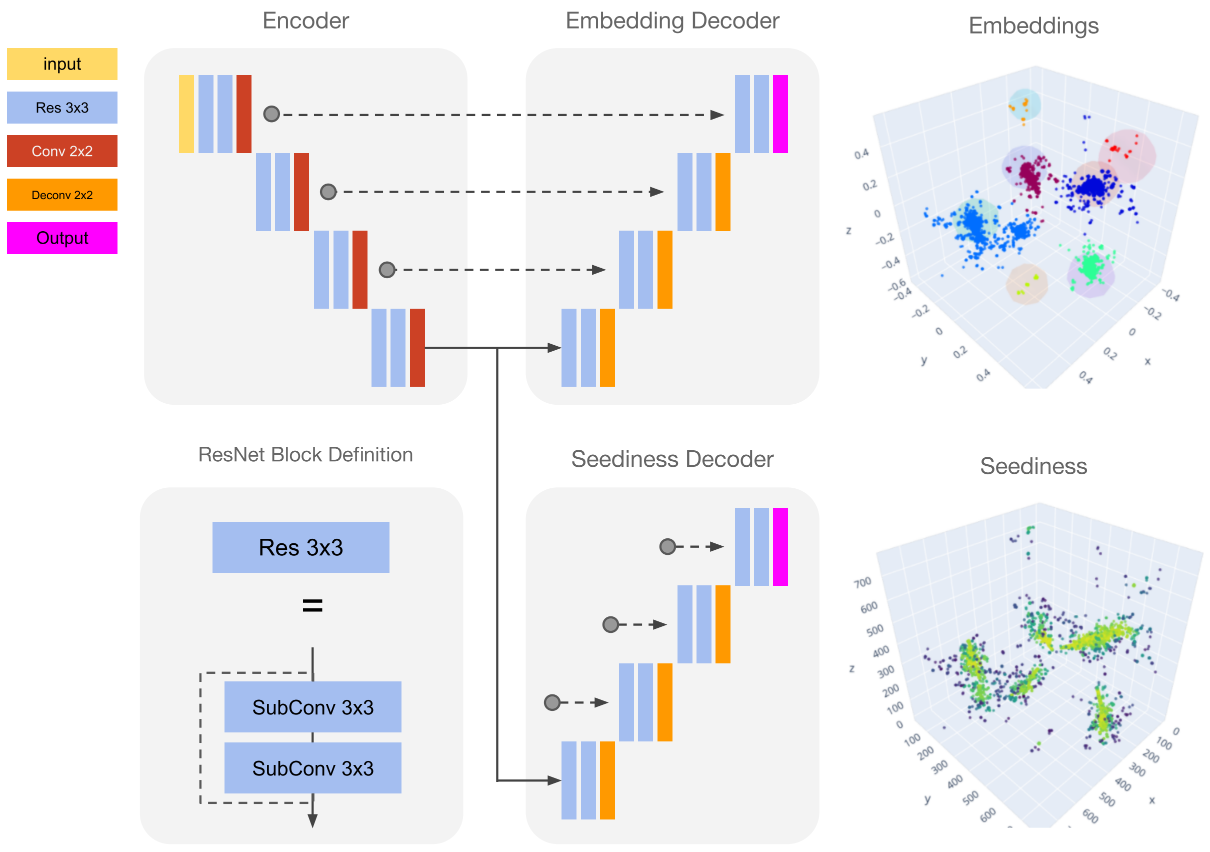


Figure 5: Architecture of the UResNet-based dense particle clustering network.

3.2 Loss Definition

The position of the voxels in the input image are first normalized to a range that spans -1 to 1 in each direction, $\{\boldsymbol{x}_i\}_{i=1}^{n}$. The embedding decoder of the network transforms the original coordinates by learning a perturbation with respect to each voxel, i.e. $\boldsymbol{e}_i = \boldsymbol{x}_i + f(\boldsymbol{x}_i)$. It also outputs a margin value for each voxel, σ_i, which estimates the spatial extent of the cluster the voxel belongs to in embedding space. The seediness decoder predicts a seediness score for each voxel, $\{s_i\}_{i=1}^{n}$. The clustering loss consists of:

- the embedding loss, which penalizes embedded voxel coordinates, $\{\boldsymbol{e}_i\}_{i=1}^{n}$, that are far away from their cluster centroid in embeddings space, $\{\boldsymbol{\mu}_k\}_{k=1}^{K}$, i.e

$$\mathcal{L}_{emb} = -\frac{1}{K}\sum_{k=1}^{K}\frac{1}{n_k}\sum_{i=1}^{n_k}\left[y_{i,k}\log\left(p_{i,k}\right) + (1-y_{i,k})\log(1-p_{i,k})\right], \tag{4}$$

  with K the true number of clusters, n_k the number of voxels in cluster k, $y_{i,k} \in \{0,1\}$ an indicator of whether or not voxel i belongs to cluster k, $p_{i,k} = \exp\left(-||\boldsymbol{e}_i - \boldsymbol{\mu}_k||^2/\sigma_k^2\right)$ the distance between $\boldsymbol{e}_i$ and $\boldsymbol{\mu}_k$ mapped to a value between 0 and 1 and σ_k the mean margin for cluster k;

- the smoothing loss, which enforces all margins within a common cluster to be similar, i.e.

$$\mathcal{L}_{smooth} = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{n_k}\sum_{i=1}^{n_k}|\sigma_i - \sigma_k|; \tag{5}$$

- the seediness loss, which tends to maximize the seediness score of points close to the centroid, i.e

$$\mathcal{L}_{seed} = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{n_k}\sum_{i=1}^{n_k}(s_{i,k} - p_{i,k})^2; \tag{6}$$

- the intercluster loss, which encourages the network to separate clusters in embedding space, i.e.

$$\mathcal{L}_{inter} = \frac{1}{K(K-1)}\sum_{i<j}[2\delta_d - ||\mu_i - \mu_j||]_+^2, \tag{7}$$

  with δ_d an inter-cluster margin parameter and $[\cdot]_+ = \max(0, \cdot)$.

The full loss is a weighted sum of the different components: $\mathcal{L}_{clust} = \gamma_1\mathcal{L}_{emb} + \gamma_2\mathcal{L}_{seed} + \gamma_3\mathcal{L}_{smooth} + \gamma_4\mathcal{L}_{inter}$.

3.3 Results

Inference follows an iterative greedy search procedure for detecting instances, as described in [11], and treats the voxels belonging to different semantic classes separately. The post-processing function first picks the voxel with the highest seediness, s_k, queries the corresponding embedding vector, $\boldsymbol{e}_k$, and the margin, σ_k. This defines a Gaussian Kernel of mean e_k and width σ_k, from which one can compute the probability values, $p_{i,k}$, for all the remaining points. The voxels with probability above some threshold p_0 are clustered together. This process is repeated with the next highest seediness voxel until either all the points are attributed to a cluster or the next highest seediness is below some threshold s_0. The values of p_0 and s_0 are selected for each class by doing a grid search on a small validation set and optimizing clustering metrics.

In order to characterize the clustering performance, three metrics are used: efficiency, purity and Adjusted Rand Index (ARI). Efficiency is a measure of undersclustering defined as the average fraction of a ground-truth cluster that is mapped to a single predicted cluster. Conversely, purity us a measure of overclustering defined as the average fraction of a predicted cluster that belongs to a single ground-truth cluster. The ARI is a more stringent metric which measures the similarity between two partitions by looking at the fraction of pair of voxels that are correctly either combined or separated, while accounting for random chance [12].

Figure 6 shows summary statistics of the clustering metric distributions associated with each semantic class. This algorithm achieves high clustering accuracy across all classes; the lowest accuracy is associated with tracks, as they can extend over much longer distances and touch or overlap regularly.

4 Graph Neural Networks Clustering

4.1 Architecture

At this stage of the reconstruction, Graph Neural Networks (GNNs) are used to cluster distant objects together: shower fragments into shower instances and particles into interactions. The reconstruction chain is schematically illustrated for the shower clustering task in figure 7 and detailed in [13].

The shower fragments are each initially encoded into a set of 22 features: the fragment voxels centroid (3), their covariance matrix (9), its eigenvalues (3), the fragment size (1), the initial point (3) and direction (3). Nodes are connected together by a complete graph in which edges are each given 19 features: the two closest points of approach (6), their displacement vector (3), its outer product (9) and its norm (1).

The node and edge features are updated through message passing. At each step, the edge features are updated by a 3-layer perceptron which combines the edge features with the node features of the two nodes that it connects together. The node features are updated by two successive 3-layer percetrons, one that builds messages by combining source node features with edge features and the other that folds target node features with aggregated messages. After three message passing steps, the number of edge and node features are reduced to two each and passed through the softmax function to produce a *primary* score for shower fragment nodes and an *adjacency* score matrix for edges. The ground-truth adjacency matrix is built such

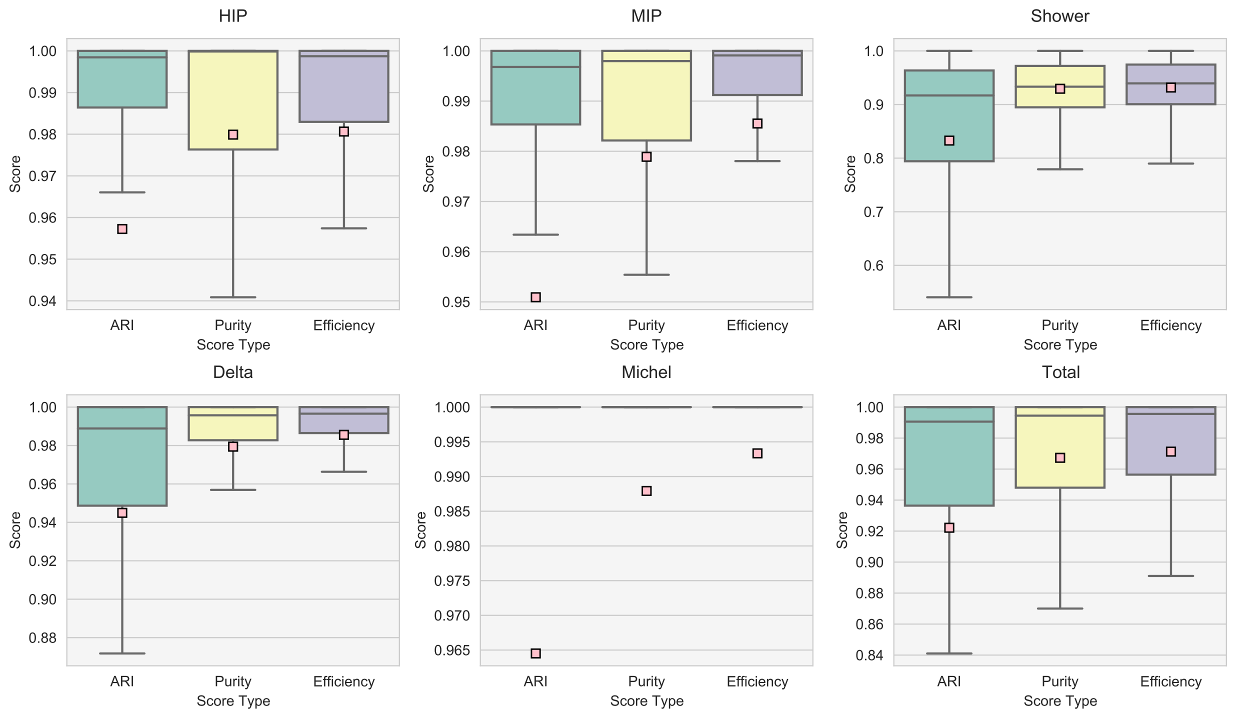


Figure 6: Box plot of the clustering metrics obtained with the UResNet-based dense particle clustering network. The square markers represent the means, the boxes the interquartile range, the lines inside the boxes the medians and the whiskers span at most 150 % of the IQR on either side of the box.

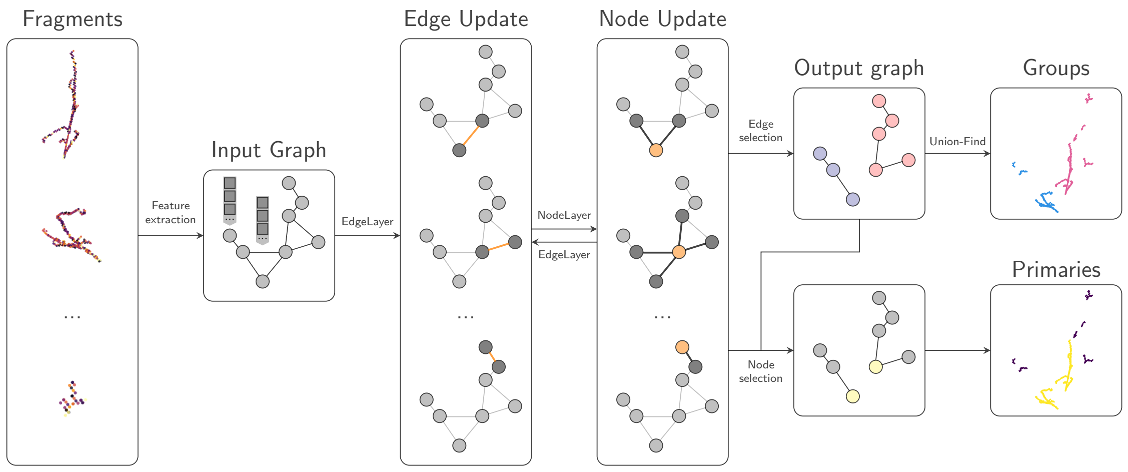


Figure 7: Schematics of the reconstruction chain for shower clustering and primary identification.

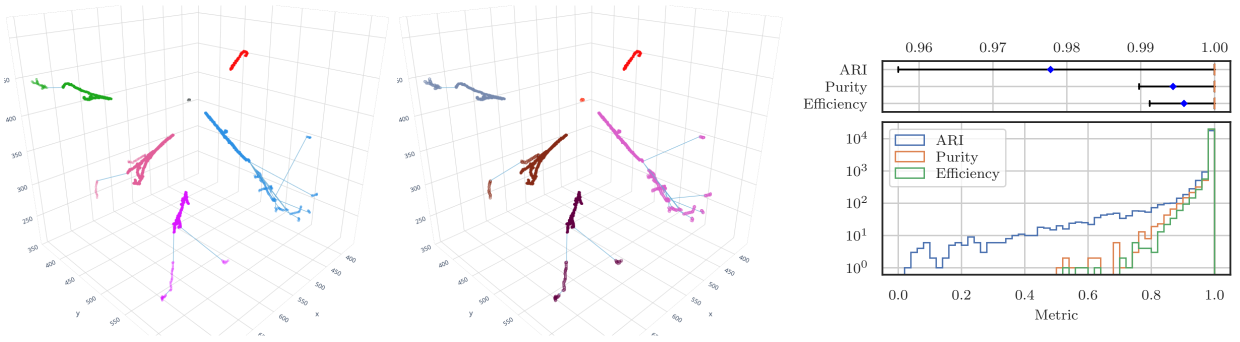
that if two nodes belong to the same group, the edge that connects them is given a label of 1. The edge scores are used to constrain the connectivity graph and the primary scores to identify shower primaries.

The interaction clustering task uses an identical architecture and a very similar feature extractor, with particle instances as an input instead of fragments and interaction groups as an output instead of shower instances. For the latter task, the node features are not explicitly used.

4.2 Results

4.2.1 Shower clustering

The leftmost two panels of figure 8 show an example of shower clustering and primary identification produced by the GNN algorithm. The third panel shows the distribution of clustering metrics on the entire test set. The network achieves an average purity and efficiency of 99.6 % and a mean ARI of 98.1 %. The network identifies primaries by selecting the node with the highest primary score in each predicted group, which yields a 99.6 % primary prediction accuracy for this dataset.

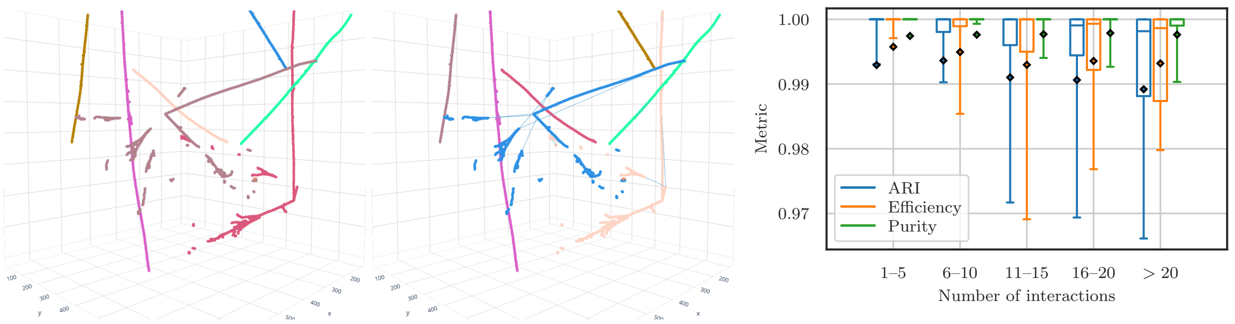


(a) (b) (c)

Figure 8: (a) Ground-truth shower group labels and particle edges, (b) predicted shower group labels and edges. The filled circle correspond to voxels that belong to the primary fragment. (c) Distribution of shower clustering metrics. In the top box plot, the blue diamonds represent the means, the orange lines the medians and the whiskers span from the 10th to the 90th percentile.

4.2.2 Interaction grouping

The leftmost two panels of figure 9 show an example of interaction clustering produced by the GNN algorithm. The third panel shows the distribution of clustering metrics of the entire test set for different number of interactions in the image. For a realistic density of 5–10 interactions per image, the network achieves an average purity of 99.8 %, an average efficiency of 99.3 % and a mean ARI of 99.1 %.



(a) (b) (c)

Figure 9: (a) Ground-truth interaction labels, (b) predicted interaction labels and edges. (c) Box plot distribution of interaction clustering metrics as a function of the number of interactions per image. The diamonds represent the means, the lines the medians, the boxes the interquartile ranges and the whiskers extend from 10th to the 90th percentile.

5 Conclusions

Machine Learning (ML) algorithms are the state-of-the-art in Computer Vision and offer the most promising path forward in maximizing the physics output of highly detailed Liquid Argon Time Projection Chambers images. This paper demonstrates the success of a modular end-to-end ML-based reconstruction chain which takes 3D particle interaction images as an input and hierarchically extracts increasingly high-level information at each stage by building upon the previous steps. Convolutional Neural Networks are used to extract voxel-level features while Graph Neural Networks are employed to tackle the clustering of spatially detached objects. The algorithm provides a list of interactions per image, a list of particle instances per interaction, their particle types, start and end points. The performance of each module has been evaluated independently and shown to perform up to the highest standards.

References

[1] C. Rubbia, "The Liquid Argon Time Projection Chamber: A New Concept for Neutrino Detectors," CERN-EP-77-08 (1977).

[2] M. Antonello *et al*, "Precise 3D Track Reconstruction Algorithm for the ICARUS T600 Liquid Argon Time Projection Chamber Detector," Advances in High Energy Physics, 1–16 (2013) [arXiv:1210.5089].

[3] Daniel A. Dwyer *et al*, "LArPix: Demonstration of low-power 3D voxelated charge readout for liquid argon time projection chambers," [arXiv:1808.02969].

[4] M. Antonello *et al*, "A Proposal for a Three Detector Short-Baseline Neutrino Oscillation Program in the Fermilab Booster Neutrino Beam," [arXiv:1503.01520].

[5] R. Acciarri *et al.*, "Long-Baseline Neutrino Facility (LBNF) and Deep Underground Neutrino Experiment (DUNE)," FERMILAB-DESIGN-2016-04 (2016) [arXiv:1601.02984].

[6] L. Dominé and K. Terao, "Scalable deep convolutional neural networks for sparse, locally dense liquid argon time projection chamber data", Phys. Rev. D 102 (2020), 012005

[7] L. Dominé *et al.*, "Point Proposal Network for Reconstructing 3D Particle Positions with Sub-Pixel Precision in Liquid Argon Time Projection Chambers", [arXiv:2006.14745].

[8] Olaf Ronneberger, Philipp Fischer and Thomas Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," [arXiv:1505.04597].

[9] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," CPVR, 770–778 (2016) [arXiv:1512.03385].

[10] M. Ester, H-P. Kriegel, J. Sander and X. Xu, "A Density-based Algorithm for Discovering Clusters a Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," KDD, 226–231 (1996).

[11] D. Koh *et al.*, "Scalable, Proposal-free Instance Segmentation Network for 3D Pixel Clustering and Particle Trajectory Reconstruction in Liquid Argon Time Projection Chambers", [arXiv:2007.03083].

[12] W. M. Rand, "Objective Criteria for the Evaluation of Clustering Methods," JASA **66**, 846-850 (1971).

[13] F. Drielsma *et al.*, "Clustering of Electromagnetic Showers and Particle Interactions with Graph Neural Networks in Liquid Argon Time Projection Chambers Data", [arXiv:2007.01335].