Clustering of electromagnetic showers and particle interactions with graph neural networks in liquid argon time projection chambers

François Drielsma,[1,*] Qing Lin,[1] Pierre Côte de Soux,[2] Laura Dominé,[3] Ran Itay,[1] Dae Heun Koh,[3] Bradley J. Nelson,[2] Kazuhiro Terao,[1] Ka Vang Tsang,[1] and Tracy L. Usher[1]

(DeepLearnPhysics Collaboration)

[1]*SLAC National Accelerator Laboratory, Menlo Park, California, 94025, USA*
[2]*ICME, Stanford University, Stanford, California, 94305, USA*
[3]*Stanford University, Stanford, California, 94305, USA*

(Received 5 January 2021; accepted 26 July 2021; published 11 October 2021)

Liquid argon time projection chambers (LArTPCs) are a class of detectors that produce high resolution images of charged particles within their sensitive volume. In these images, the clustering of distinct particles into superstructures is of central importance to the current and future neutrino physics program. Electromagnetic (EM) activity typically exhibits spatially detached fragments of varying morphology and orientation that are challenging to efficiently assemble using traditional algorithms. Similarly, particles that are spatially removed from each other in the detector may originate from a common interaction. Graph neural networks (GNNs) were developed in recent years to find correlations between objects embedded in an arbitrary space. The graph particle aggregator (GrapPA) first leverages GNNs to predict the adjacency matrix of EM shower fragments and to identify the origin of showers, i.e., primary fragments. On the PILArNet public LArTPC simulation dataset, the algorithm achieves a shower clustering accuracy characterized by a mean purity of 99.4%, a mean efficiency of 99.6% and a primary identification accuracy of 99.8%. It yields a relative shower energy uncertainty of $(4.1 + 1.4/\sqrt{E(\mathrm{GeV})})\%$ and a shower direction uncertainty of $(2.1/\sqrt{E(\mathrm{GeV})})^\circ$. The optimized algorithm is then applied to the related task of clustering particle instances into interactions and yields a mean purity of 99.8% and a mean efficiency of 99.5% for an interaction density of $\mathcal{O}(1)\ \mathrm{m}^{-3}$.

DOI: 10.1103/PhysRevD.104.072004

I. INTRODUCTION

In recent years, accelerator-based neutrino oscillation experiments in the United States have been designed to use liquid argon time projection chambers (LArTPCs) as their central neutrino detection technology [1]. Charged particles that traverse these detectors ionize the noble liquid. The electrons so produced are drifted in a uniform electric field toward a readout plane. The location of the electrons collected on the anode, combined with their arrival time, offers mm-scale resolution images of charged particle interactions [2]. This level of tracking precision—coupled to the detailed calorimetric information that a totally active detector provides—is believed to be the key to resolving some of the ambiguities observed in previous experiments and to extending their energy reach to probe the MeV-scale physics sector [3,4].

The short baseline neutrino program (SBN) [5] aims to clarify an anomalous signal observed by the MiniBooNE experiment [6]. It will eventually make use of three LArTPCs of varying sizes: the short baseline near detector (SBND, 112 t), MicroBooNE (90 t) [7], and ICARUS (600 t) [8]. The DUNE experiment [9] will use the LArTPC technology to measure long-baseline neutrino oscillations with unprecedented precision. It will consist of a near detector (105 t) and a far detector (40 kt). The main signal for these physics endeavors is the unambiguous appearance of electron neutrinos—which manifest themselves as electron showers—in a beam of muon neutrinos. Their success thus centrally depends upon the accurate reconstruction of showers and specifically of their initial positions, directions, and energies. Both experiments also face the substantial challenge of assembling particles into complete neutrino interaction events, which are often accompanied by unrelated activity. Detectors located close to the surface, such as those of the SBN program, suffer from a high rate of cosmic rays, while the future DUNE near detector will observe a high rate of pileup events, with up to twenty neutrino interactions per beam pulse.

[*]drielsma@stanford.edu

Electromagnetic (EM) showers exhibit an incoherent branching tree structure in LArTPCs. As an electron propagates through the dense detector medium, it loses energy through ionization and stochastically emits photons until it comes to a stop. The emitted photons propagate through the noble liquid with a mean free path of 15–30 cm, for energies in the range 10–1000 MeV, before they either produce an electron-positron pair or emit a Compton electron [10]. A single electron or photon creates a cascade of spatially distinct EM particles—referred to as *fragments* in this study—that may be far removed from one another in the image. Assembling these fragments into coherent shower objects has been a persistent challenge in LArTPCs that has not yet been fully resolved using traditional analysis techniques.

Early LArTPC experiments employed time-intensive and nonscalable methods which consisted in physicists scanning images and identifying individual shower pixels by eye [11]. More recent studies have made use of the PANDORA multialgorithm software package [12], which relies on the successful identification of an interaction vertex to reconstruct showers. The vertex is leveraged as a reference point to build a system of polar coordinates around it and to merge shower fragments that fit inside a common cone. The PANDORA paper provides distributions of shower clustering efficiency (referred to as *completeness*) and purity which can be readily compared to those provided later in this article. The MicroBooNE neutral pion reconstruction paper [10]—the most recent analysis making use of this method—shows that clustering inefficiencies dominate the energy smearing, currently in the range of 15%–20%.

Graph neural networks (GNNs) became popular in recent years as a way to leverage the concept of receptive field developed in the context of convolutional neural networks and generalize it to arbitrary objects [13]. The receptive field is no longer exclusively determined by a square neighborhood of pixels in an image but rather defined by an adjacency matrix whose elements indicate whether objects are connected by an edge in a graph. This development is ideally suited to the clustering of EM showers in LArTPCs as they may each be represented by a collection of shower fragments (the graph nodes) that are connected by invisible photons (the graph edges). The task is then to identify the edges that connect fragments within a shower and to tag the fragments that initiate showers, i.e., the primary fragments. Similar methods have been used for other particle aggregation tasks, mostly in collider experiments [14].

In the case of interaction clustering, distinct sources of activity in the detector can be clustered into separate groups using a GNN by building a graph in which particles are nodes and edges represent correlations between particles. The task in then is to identify the edges that connect particles that belong to the same interaction.

The study presented in this paper is reproducible using a `singularity` [15] software container,[1] implementations available in the `lartpc_mlreco`[2] repository and a public simulation sample [16] made available by the DeepLearnPhysics collaboration.[3]

Section II presents the architecture of the reconstruction chain, as applied to the shower clustering task, from the LArTPC image input to the inference stage. Section III outlines the studies that were conducted to optimize the chain. Section IV shows a detailed analysis of the shower reconstruction performance on this sample. Section V describes how the algorithm was adapted to the particle interaction clustering task and provides some performance metrics.

II. RECONSTRUCTION CHAIN

A. Data

The graph particle aggregator (GrapPA) is schematically illustrated for the shower clustering task in Fig. 1. The input LArTPC dataset, PILArNet [16], consists of rasterized 3D energy deposition images of simulated ionizing particle interactions in liquid argon. An image corresponds to a $\sim 12\ \mathrm{m}^3$ cubic volume of liquid argon with an edge length of 768 voxels (1 voxel $= 3^3\ \mathrm{mm}^3$). Each image includes a multiple-particle vertex, i.e., a set of particles originating from a common vertex, overlayed with randomly located tracklike cosmic muon trajectories and showerlike instances. An example image is shown in Fig. 2. This image contains three tracks and two showers originating from the vertex in addition to two muon tracks and a detached shower.

Figure 3 shows the distributions of the number of particles coming from a common vertex in each image, the number of overlayed particles and their respective type compositions. These data emulate a particle density above that expected in an SBN program LArTPC detector. The data is split in two data samples: a training set of 125480 images and a validation set of 22439 images. This dataset is an idealized simulation with no detector-related effects, but it preserves the realistic interaction geometries necessary to demonstrate the viability of a GNN as a particle aggregator. The exact influence of electron recombination, electron lifetime and digitization on reconstruction performance, particularly in real data, will be studied in future publications.

For the shower clustering task, the set of input voxels is constrained to those associated with electromagnetic (EM) activity. In the scope of this paper, the particle type of each voxel is assumed to be known, as it has been demonstrated

[1]https://singularity-hub.org/containers/11757
[2]https://github.com/DeepLearnPhysics/lartpc_mlreco3d
[3]https://osf.io/6gvf4/

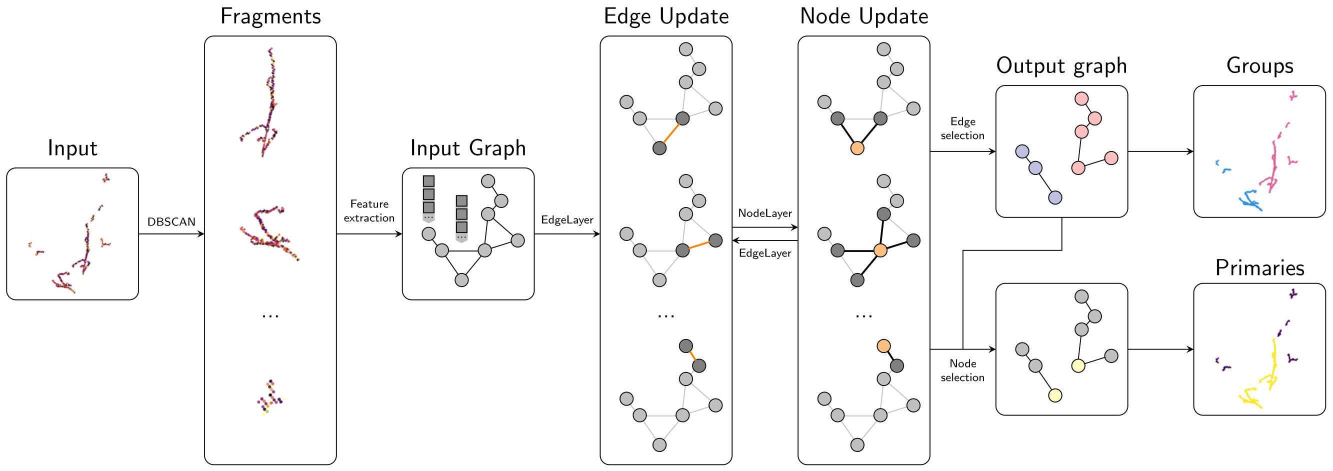

FIG. 1. Architecture of the graph particle aggregator (GrapPA) for shower clustering and primary identification. The input set of voxels associated with electromagnetic showers is passed through a density-based clustering algorithm that forms dense shower fragments. Each fragment is encoded into a set of node features in a graph connected by arbitrary edges carrying edge features. Edge and node features are updated through a series of message passing composed of edge and node updaters. The updated edge features are used to constrain the connectivity graph and the updated node features to identify primaries.

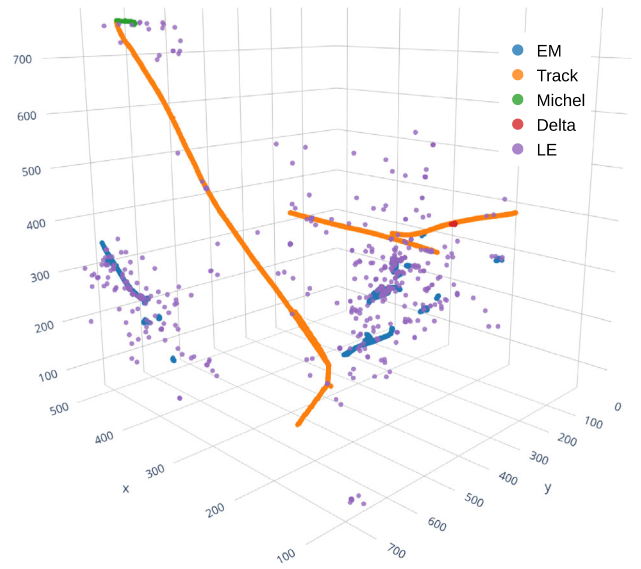

FIG. 2. Example image from the simulated LArTPC input dataset. The colors correspond to the semantic type of the particle that deposited the energy: "EM" stands for electromagnetic, "track" for protons, pions and muons, "Michel" for muon decay electrons, "Delta" for delta electrons and "LE" for low energy scatters (low energy EM and nuclear activity). Axes values represent voxel coordinates.

that semantic segmentation neural network can identify EM voxels with a 99.4% voxel-wise accuracy [17]. A similarly designed neural network has been shown to work on real data from the MicroBooNE LArTPC detector [18]. The exact effect of using the aforementioned algorithm on the shower clustering accuracy will be addressed in a separate paper studying the performance of the end-to-end reconstruction chain.

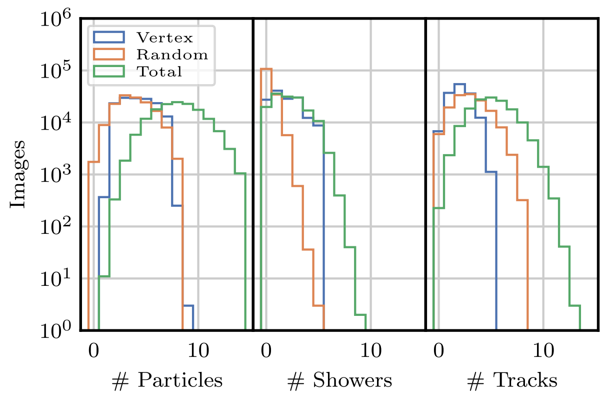

FIG. 3. Distribution of the number of particles, showers and tracks in each image, originating from a common vertex (blue), randomly scattered (orange) and combined (green).

B. Fragment clustering

A shower object encompasses all energy deposition associated with a single primary[4] electron or photon and all its subsequent EM daughters. A shower fragment is defined as a spatially dense subset of voxels of a shower instance such that each voxel is in the Moore neighborhood of at least one other voxel in the fragment, i.e., at least "touches" it diagonally. As the ground truth[5] fragments are not known *a priori*, EM voxels are clustered using the density based spatial clustering of applications with noise (DBSCAN) algorithm [19] with a Euclidean distance

[4]A primary electron or photon does not have an EM parent and is neither a delta ray, a Michel electron nor a deexcitation photon.

[5]In the field of computer vision, ground truth refers to the data labels, i.e., a predefined target for the reconstruction.

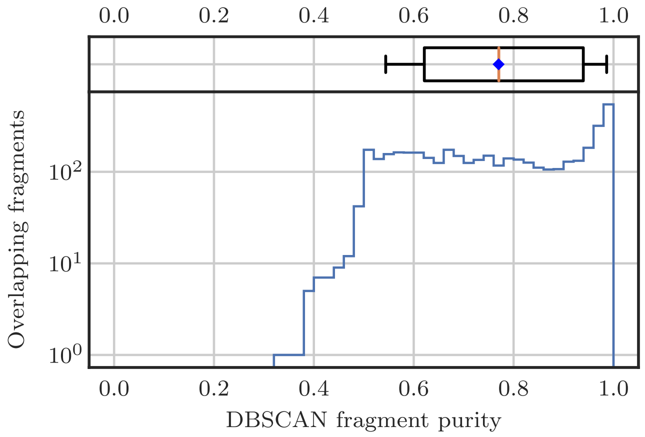

FIG. 4. Distribution of the purity of overlapping shower fragments built using DBSCAN with a distance scale of 1.9. In the top box plot, the blue diamond represents the mean, the orange line the median, the box the IQR and the whiskers span from the 10th to the 90th percentiles.

scale set to 1.9.[6] This ensures that DBSCAN does not break ground-truth fragments, by definition, as any two touching voxels are merged by this algorithm. It can, however, merge two or more fragments that belong to separate shower instances into one. Purity is defined as the maximum fraction of voxels in a predicted fragment that belongs to a single instance. In this dataset, fragments reconstructed with DBSCAN contain more than one ground-truth label in 0.2% of cases and appear in ~2% of images. Figure 4 shows the purity distribution of the small fraction of fragments that do contain an overlap. It is mostly uniform between 0.5 and 1, with a peak close to 1. DBSCAN fragments with a purity < 1 reduce the overall purity of the shower reconstructed downstream of this process.

Fragments strictly smaller than 10 voxels are not included in the input to the clustering task. This selection

(i) emulates detector inefficiencies at low energy;
(ii) removes most spherical fragments, including those from nuclear activity, which lack directionality;
(iii) reduces the overall number of fragments to cluster.

The exact value of this cut might be revisited or possibly removed entirely in the future if the detector under study allows it and if the trade-off between threshold effects and clustering accuracy is deemed advantageous.

To characterize the impact of the above selection on the shower energy resolution, primary showers that are at least 95% contained in the image volume are selected. The fraction of the total shower energy deposited in fragments of size 10 and above is represented as a function of shower energy in Fig. 5. The shower completeness is on average ~82.6%, for showers of 100 MeV and above. For lower energy showers, the fraction of energy deposited in small fragments decreases slightly. This selection introduces an average relative uncertainty of 4.2% on the final energy reconstruction.

[6]This is equivalent to a Chebyshev distance of 1.

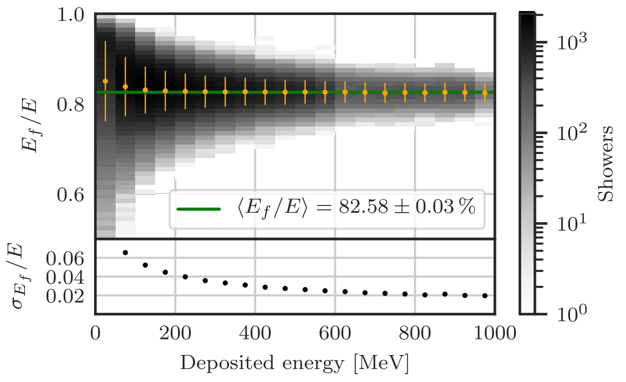

FIG. 5. Fraction of the energy deposited by a shower in fragments of size 10 voxels and above. The orange markers on the top pad represent the mean and their error bars the RMS; the latter is shown on its own in the bottom pad. The green line is a constant fit to the markers above 100 MeV.

Figure 6 shows the fragments constructed upon the shower voxels of an image using the DBSCAN algorithm. The goal of the clustering algorithm is to cluster these fragments together into shower objects.

C. Graph representation

A graph $G(V,E)$ is a collection of nodes V and edges $E \subseteq V \times V$. Each shower fragment represents a node in a graph. There is an arbitrary number of ways to build a graph between the nodes. The optimal choice of input edges is discussed in Sec. III B. Each node is encoded as a vector of F_v *features* and each edge as a vector of F_e

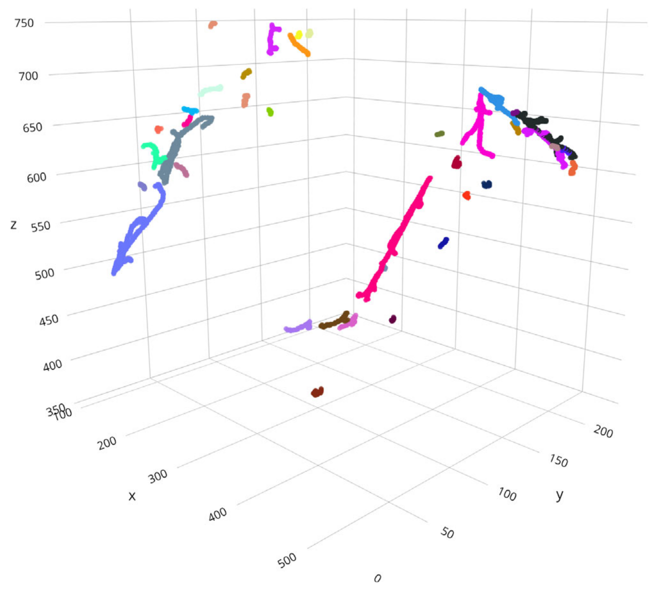

FIG. 6. Image of the EM shower voxels with a color scale that represents the DBSCAN cluster ID. Axes values represent voxel coordinates.

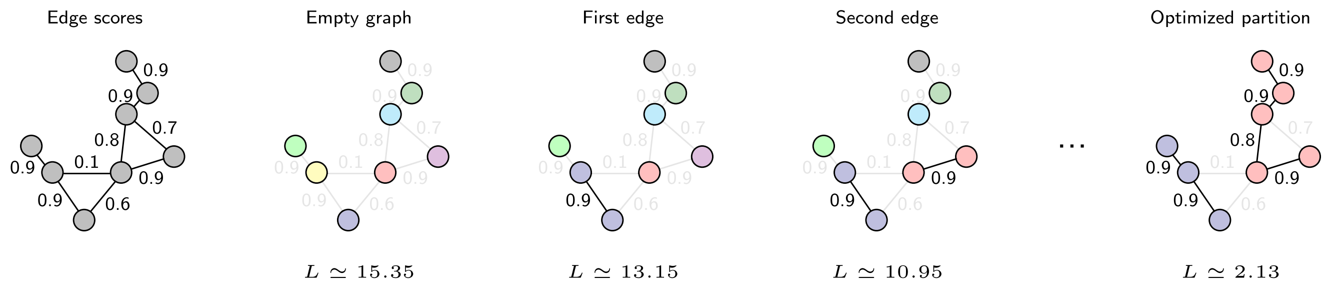

FIG. 7. Schematics of the edge selection mechanism at the inference stage. The partition loss defined in equation (6) is first calculated for an empty graph in which each node forms its own group. Edges are sequentially added in order of decreasing score only if the new partition they form decreases the partition loss. The edge with score 0.6 is not added to the graph because it would put the nodes connected by the edge with score 0.1 in the same group and increase the partition loss.

features. Multiple ways of extracting these features are studied and optimized in Sec. III B.

D. Message passing

Message passing is used to communicate information within a graph [13,20]. During the information propagation process, at step $s+1$, edge attributes are updated by combing the features coming from the nodes it connects together with its own through

$$e_{ij}^{s+1} = \psi_\Theta(\boldsymbol{x}_i^s, \boldsymbol{x}_j^s, \boldsymbol{e}_{ij}^s), \quad (1)$$

with $\boldsymbol{x}_i, \boldsymbol{x}_j$ the node feature vectors associated with nodes i and j, respectively, e_{ij} the features of the edge connecting i to j and ψ a differentiable function such as a multilayer perceptron (MLP) [21]. In order to update the node features, the message coming from node j communicated to node i at step $s+1$ is defined as

$$\boldsymbol{m}_{ji}^{s+1} = \phi_\Theta(\boldsymbol{x}_j^s, \boldsymbol{e}_{ji}^{s+1}), \quad (2)$$

with ϕ_Θ a differentiable function such as an MLP. The messages coming from the neighborhood $\mathcal{N}(i)$[7] of node i are then aggregated with its own at each step to update its features following

$$\boldsymbol{x}_i^{s+1} = \chi_\Theta(\boldsymbol{x}_i^s, \square_{\mathcal{N}(i)}\boldsymbol{m}_{ji}^{s+1}), \quad (3)$$

with χ_Θ a differentiable function such as an MLP and $\square$ an aggregation function such as sum, mean, or max. The specific implementation of the differentiable functions and the number of message passing steps are studied and optimized in Sec. III D.

[7]The neighborhood of node i, $\mathcal{N}(i)$, is the set of nodes which are adjacent to node i in the input graph, i.e., that share an edge with node i.

E. Loss definition

Downstream of the message passing steps, two fully connected linear layers reduce the edge and node features separately to two channels each. The outputs are passed through the softmax function and the second channel is used to create a vector of N_v primary scores for nodes, $\boldsymbol{s}^v$, and a vector of N_e adjacency scores for edges, $\boldsymbol{s}^e$. The binary cross-entropy loss is then applied to node and edge scores alike as follows:

$$\mathcal{L}_v = -\frac{1}{N_v}\sum_i y_i \ln(s_i^v) + (1-y_j)\ln(1-s_i^v), \quad (4)$$

$$\mathcal{L}_e = -\frac{1}{N_e}\sum_{(i,j)\in E} a_{ij}\ln(s_{ij}^e) + (1-a_{ij})\ln(1-s_{ij}^e), \quad (5)$$

with y_i the primary label of node i and a_{ij} the adjacency label of the edge connecting node i to node j. The primary label, y_i, is 1 if the fragment initiated the shower, 0 otherwise. The definition of the target adjacency matrix, $\boldsymbol{A}$, which determines the adjacency labels, is discussed in Sec. III E. The total loss is defined as $\mathcal{L} = \mathcal{L}_v + \mathcal{L}_e$.

F. Inference

The network predicts an edge score matrix, $\boldsymbol{S}^e$, which tries to replicate the predefined ground-truth adjacency matrix, $\boldsymbol{A}$. In a graph partition problem, $\boldsymbol{A}$ should be designed such that, if $a_{ij} = 1$, then nodes i and j belong to the same group. The converse statement does not have to hold, as nodes i and j may not be connected directly as long as they are linked through an indirect path.

Figure 7 schematically illustrates how the score matrix is converted into a node partition prediction. At the inference stage, one has to find the optimal node partition, $\hat{\boldsymbol{g}}$, such that, if s_{ij}^e is close to 1, nodes i and j are encouraged to be put in the same true group. Mathematically, this corresponds to minimizing the partition cross-entropy loss defined as

$$L(\boldsymbol{S}^e|\boldsymbol{g}) = -\frac{1}{N_e}\sum_{(i,j)\in E}\delta_{g_i,g_j}\ln(s^e_{ij}) + (1-\delta_{g_i,g_j})\ln(1-s^e_{ij}), \tag{6}$$

for $\boldsymbol{g}$, i.e., $\hat{\boldsymbol{g}} = \min_{\boldsymbol{g}} L(\boldsymbol{S}^e|\boldsymbol{g})$, with δ the Kronecker delta. If an edge is not in the input graph, it does not contribute to the grouping optimization loss.

The cardinality of the set of all possible partitions of a set of n nodes, G, corresponds to the Bell number B_n. This number grows quickly with the number of nodes to prohibitively large values that rule out brute-force optimization. Instead, edges are considered sequentially to be added to a predicted adjacency matrix, $\hat{\boldsymbol{A}}$, in order of decreasing edge score. At each step, the partition score is evaluated by running the Union-Find algorithm [22] on the predicted matrix. The edge is permanently added to $\hat{\boldsymbol{A}}$ if the new partition improves the loss defined in equation (6). The optimizer stops when the next available edge has a score below 0.5.

Given the predicted partition of the graph nodes, $\hat{\boldsymbol{g}}$, the primary nodes are identified by picking those with the highest primary score in each group.

G. Metrics

Three clustering metrics are used to systematically characterize the performance of the clustering algorithms in this paper: efficiency, purity and adjusted Rand index (ARI) [23]. The efficiency and purity are defined as:

$$\text{Efficiency} = \frac{1}{N}\sum_{i=1}^{N_t}\max_j |c_j \cap t_i|, \tag{7}$$

$$\text{Purity} = \frac{1}{N}\sum_{i=1}^{N_p}\max_j |c_i \cap t_j|, \tag{8}$$

with N_t the true number of showers, N_p the predicted number of showers, N the total number of voxels in the image, t_k the kth true cluster and c_k the kth predicted cluster. The Rand index (RI) is defined as the accuracy on binary edge classification between any two pair of voxels; the ARI ajdusts for random chance by shifting this measure with respect to the average RI obtained for all possible permutations of the predicted labels, i.e.,

$$\text{ARI} = \frac{\text{RI} - E(\text{RI})}{\max(\text{RI}) - E(\text{RI})}, \tag{9}$$

with E the expectation value. Note that if one of the partitions contains a single cluster, a single voxel mistake yields an ARI of 0, as permutations do not affect RI.

III. OPTIMIZATION

A. Training regiment

In this section, the reconstruction steps are optimized to maximize the clustering accuracy in a model that uses ground-truth shower fragments as an input and does not attempt to predict shower primaries. Variations are studied with respect to a baseline model in which

(i) the input node and edge features are geometric;
(ii) the input graph is a complete graph;
(iii) the number of message passings is 3;
(iv) the ground-truth adjacency matrix corresponds to a cluster graph built upon groups;
(v) the batch size is 128;
(vi) the Adam optimizer [24] is used with a learning rate is 0.0025.

The reconstruction chain is trained for 25 epochs of the training set for each configuration under study. The edge classification accuracy and cross-entropy loss are cross-validated with 10000 events from the validation set every ~1 epoch to check for overtraining.

B. Feature extraction

Each shower fragment has to be encoded into a set of node features and each edge in the input graph can be provided with a set of features of its own. Two methods of feature extraction have been considered in the context of this paper and are presented and compared in this subsection: Geometric and CNN.

Geometric features are a list of summary statistics of the distribution of fragment voxels in Euclidean space. It includes the following 22 features:

(i) normalized covariance matrix (9 features);
(ii) normalized principal axis (3 features);
(iii) centroid (3 features);
(iv) number of voxels (1 feature);
(v) initial point (3 features);
(vi) normalized initial direction (3 features).

In this study, the initial point of a fragment is acquired by picking the center of the voxel, $\boldsymbol{v}_s$, closest the true simulated particle first energy deposition. In a realistic setting, it will be reconstructed using the point proposal network (PPN) [25]. Figure 8 shows the distribution of the distance between the point with the highest PPN score in a given fragment and the true initial fragment point. The point is closer than 3 voxels from the true point in 96.3% of cases.

The initial direction, $\hat{\boldsymbol{d}}$, is estimated by calculating the normalized mean direction from the initial point, $\boldsymbol{v}_s$, to all the other fragment voxels within a neighborhood distance R_n of the initial point, i.e.,

$$\hat{\boldsymbol{d}} = \langle\boldsymbol{v}\rangle/|\langle\boldsymbol{v}\rangle|, \qquad \langle\boldsymbol{v}\rangle = \frac{1}{N_n}\sum_{\{i|d_{si}\le R_n\}}(\boldsymbol{v}_i - \boldsymbol{v}_s), \tag{10}$$

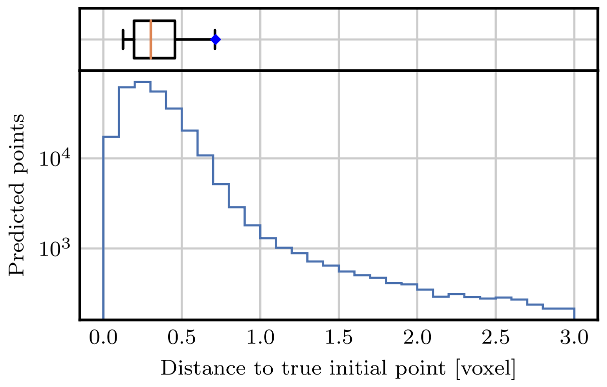

FIG. 8. Distance between the point with the highest PPN score within a fragment and the true initial point of the fragment. In the top box plot, the blue diamond represents the mean, the orange line the median, the box the IQR and the whiskers span from the 10th to the 90th percentiles.

with d_{si} the Euclidean distance between $\boldsymbol{v}_i$ and $\boldsymbol{v}_s$ and $N_n = \#\{i | d_{si} \leq R_n\}$ the number of voxels in the neighborhood. The radius was optimized to $R_n = 5$ by minimizing the spread of the angle between this direction estimate and the true normalized particle momentum, $\boldsymbol{d} = \boldsymbol{p}/|\boldsymbol{p}|$, as shown for multiple radial cut values in Fig. 9. In the following, the importance of the initial point and direction for the clustering task is studied by training the model without them (referred to as NI).

The geometric edge features include 19 components:
(i) closest points of approach (CPAs) (6 features);
(ii) displacement between CPAs (3 features);
(iii) outer product of displacement (9 features);
(iv) length of displacement (1 feature).

The CNN feature extractor treats each fragment as an individual node image—by masking out voxels that are not associated with it in the image—and each pair of fragments

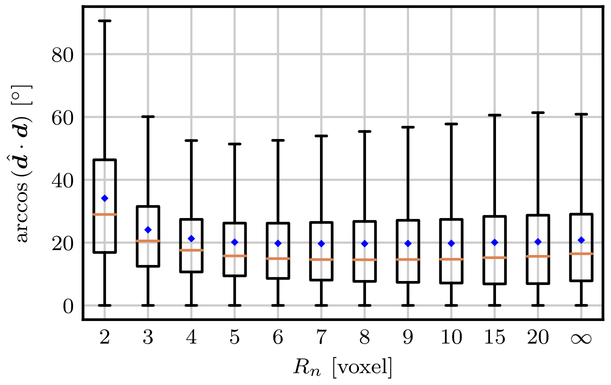

FIG. 9. Box plot of the angle between the reconstructed direction of a shower fragment, $\hat{\boldsymbol{d}}$, and the normalized true particle momentum, $\boldsymbol{d}$, as a function of the neighborhood cut, R_n. The blue diamonds represent the means, the orange lines the medians, the boxes the IQRs and the whiskers extend at most 150% of the IQR on either side of the box.

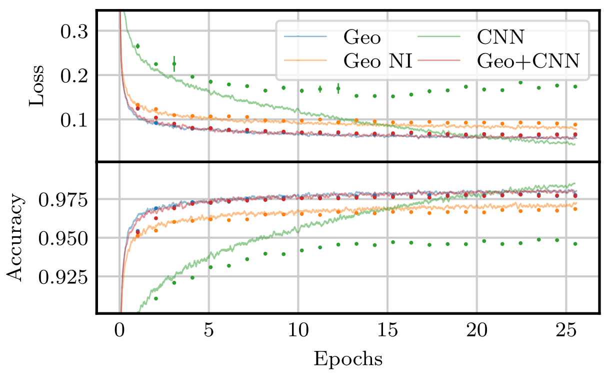

FIG. 10. Edge score loss and edge prediction accuracy for the different encoders under considerations. The training curves are represented as lines and the validation points as round markers with statistical error bars.

as an edge image. Images are passed through a convolutional neural network (CNN) which consists of alternating ResNet blocks [26] and strided convolutions which progressively reduce the spatial size of each image while increasing the number of features in each channel. Sparse convolutions are used to efficiently handle mostly empty images [27]. In this study, the kernel size is set to 5, the number of strided convolutions to 8 and the input number of filters to 32. The features of the most spatially compressed 3^3 voxels image are average pooled to form a vector of 64 features per node and 64 features per edge.

Figure 10 shows the training and validation curves for each type of feature extractor, produced using the training and validation datasets, respectively. Removing the initial point and initial direction (NI) from the geometric features reduces the edge prediction accuracy by ~1%. The CNN as a stand-alone encoder quickly and dramatically overfits the training set. The addition of the CNN features to the baseline geometric features does not measurably improve the global edge classification loss.

C. Input graph

The input set of fragments is partitioned into groups based on an adjacency score matrix. An edge is given a score only if it appears in the input graph. Several graph construction methods were studied to find the optimal receptive field for nodes:
(i) complete graph (all possible pairwise edges);
(ii) Delaunay graph (edges in the spatial Delaunay triangulation of the input voxels only);
(iii) MST graph (edges in the spatial minimum spanning tree of the input voxels only);
(iv) 5NN graph (edges connecting each node with its 5 nearest neighbors only).

These graphs are all defined undirected, so that if a message path exists from node i to node j, its reciprocal path exists as well. The network is trained to activate both reciprocal paths if two nodes belong to the same group.

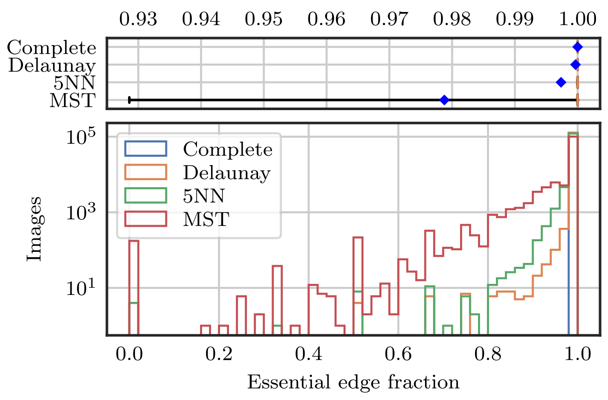

FIG. 11. Fraction of the edges necessary to make perfect clustering predictions that appears in an image for the different input graphs under study. In the top box plot, the blue diamonds represent the means, the orange lines the medians and the whiskers span from the 10th to the 90th percentiles.

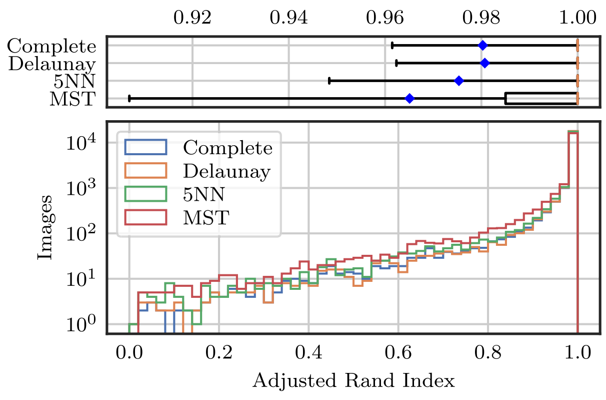

FIG. 13. Adjusted Rand index (ARI) distributions on the test dataset for the four input graphs under consideration. In the top box plot, the blue diamonds represent the means, the orange lines the medians, the boxes the IQRs and the whiskers span from the 10th to the 90th percentiles.

Restricting the number of edges in the input graph can potentially simplify the clustering task by allowing for the nodes to only focus on messages coming from nodes that are adjacent in the input graph. To be suitable, an input graph must include at least one *essential* path from each node to at least one other node that belongs to the same group, without passing through a node that does not. Figure 11 shows the fraction of essential edges that appear in the input graph, i.e., the fraction of nodes that are reachable through an essential path. It shows that the MST graph is the most inefficient proposed network, on average missing a prohibitively large ∼2.3% of the edges necessary to make correct predictions. All the other graphs are viable options, although their accuracy will be limited in images missing essential edges.

Figure 12 shows the training and validation curves for the aforementioned input graph structures. Figure 13 shows the adjusted Rand index clustering metric on the test set for each configuration. The complete graph, which is the one that includes all possible message passing routes, performs best. Delaunay graphs perform similarly—at a much greater computational cost—while other input graphs fail to yield a similar precision. This demonstrates the ability of the network to prioritize messages purely based on the features that it is provided with.

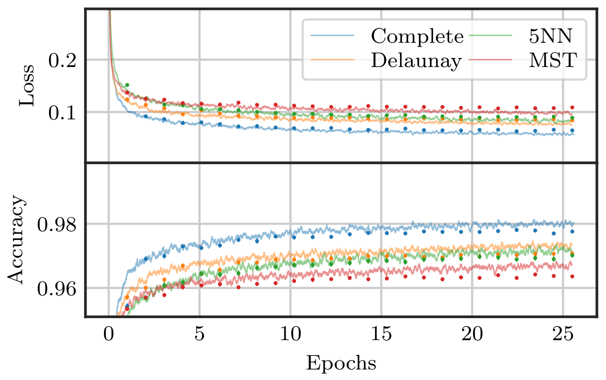

FIG. 12. Edge score loss and edge prediction accuracy for the different input graphs under consideration. The training curves are represented as lines and the validation points as round markers with statistical error bars.

D. Message passing

At a message passing step $s+1$, the edge features are updated by an edge updater which maps $2F_v^s + F_e^s$ features coming from the nodes it connects and its own features to F_e^{s+1} features, with F_v^s the number of node features at step s and F_e^s the number of edge features at step s. In this study, regardless of the step number, F_e^{s+1} is set to 64. The edge updater MLP consists of three linear layers, each preceded by a 1D batch normalization layer and followed by a LeakyRELU layer of leakiness 0.1. The first linear layer brings the number of features to 64, the other two maintain that number.

The nodes are updated from F_v^s features to F_v^{s+1} features by a function of neighboring nodes and the edge attributes that connect them, as summarized by equations (2) and (3). Five `Pytorch Geometric` [28] *layers* were studied in the context of this paper: MetaLayer, NNConv, EdgeConv, GATConv and AGNNConv.

The MetaLayer [20] uses two successive MLPs to combine the edge features, the node features and their neighbor features. The first MLP combines the F_v^s source node features together with the F_e^{s+1} edge features using three linear layers, each preceded by a 1D batch normalization and followed by a LeakyRELU layer of leakiness 0.1. This produces one message per edge e_{ij}, m_{ij}^{s+1}, each containing F_v^{s+1} features. The second MLP combines the F_v^s target node features with the averaged F_v^{s+1} message features using the same architecture as the first MLP. At each message passing step, F_v^{s+1} is set to 64.

The NNConv [29] layer is defined as

$$\boldsymbol{x}_i^{s+1} = \boldsymbol{\Theta}\boldsymbol{x}_i^s + \sum_{j\in\mathcal{N}(i)} \boldsymbol{x}_j^s \cdot h_\Theta(\boldsymbol{e}_{j,i}^{s+1}), \qquad (11)$$

with $\boldsymbol{\Theta}$ an $F_v^{s+1} \times F_v^s$ matrix of weights and h_Θ an MLP that maps F_e^{s+1} edge features to an $F_v^{s+1} \times F_v^s$ matrix. In this study, the MLP is composed of three layers of a batch normalization, a linear layer and a LeakyReLU function of leakiness 0.1. The first layer increases the number of features from F_e to $F_v^{s+1} \times F_v^s$ and the following two keep it constant.

The EdgeConv [30] layer is defined as

$$\boldsymbol{x}_i^{s+1} = \sum_{j\in\mathcal{N}(i)} h_\Theta(\boldsymbol{x}_i^s \| \boldsymbol{x}_j^s - \boldsymbol{x}_i^s), \qquad (12)$$

with $\|$ the concatenation operator and h_Θ an MLP that maps $2F_v^s$ concatenated features to F_v^{s+1} features. The implementation of the MLP uses an identical implementation to that of the NNConv layer.

The GATConv [31] layer uses the concept of attention:

$$\boldsymbol{x}_i^{s+1} = \alpha_{ii}\Theta\boldsymbol{x}_i^s + \sum_{j\in\mathcal{N}(i)} \alpha_{ij}\Theta\boldsymbol{x}_j^s,$$
$$\alpha_{ij}^s = \frac{\exp(LR(\boldsymbol{a}^T[\boldsymbol{\Theta}\boldsymbol{x}_i^s \| \boldsymbol{\Theta}\boldsymbol{x}_j^s]))}{\sum_{j\in\mathcal{N}(i)\cup\{i\}} \exp(LR(\boldsymbol{a}^T[\boldsymbol{\Theta}\boldsymbol{x}_i^s \| \boldsymbol{\Theta}\boldsymbol{x}_j^s]))}, \qquad (13)$$

with $LR = \text{LeakyReLU}$ and $\boldsymbol{a}$ the attention weight vector of size $2F_v^s$. The weight of the message coming from each node in the neighborhood of i is explicitly learned as a function of the node features.

The AGNNConv [32] node updater uses a slightly different attention mechanism:

$$\boldsymbol{X}^{s+1} = \boldsymbol{P}^s\boldsymbol{X}^s,$$
$$P_{ij}^s = \frac{\exp(\beta\cos(\boldsymbol{x}_i^s, \boldsymbol{x}_j^s))}{\sum_{j\in\mathcal{N}(i)\cup\{i\}} \exp(\beta\cos(\boldsymbol{x}_i^s, \boldsymbol{x}_j^s))}, \qquad (14)$$

with β a learnable parameter. Note that for both of the attention-based layers and the EConv layer, the node features do not take into account the edge features explicitly.

Figure 14 shows the training and validation curves for the functions under study and for three iterations of message passing. The MetaLayer node updater performs best, NNConv is comparable but overtrains faster at a large number of epochs. Note that, for this task, the added complexity through the use of an MLP is evidently useful. All layers that explicitly include the edge features—or the difference between node features as a substitute in the EdgeConv function—perform similarly. Figure 15 shows the training and validation curves using the MetaLayer node updater with a number of message passing varying between 1 and 5. Adding more than three message passing steps does not measurably improve the edge classification accuracy.

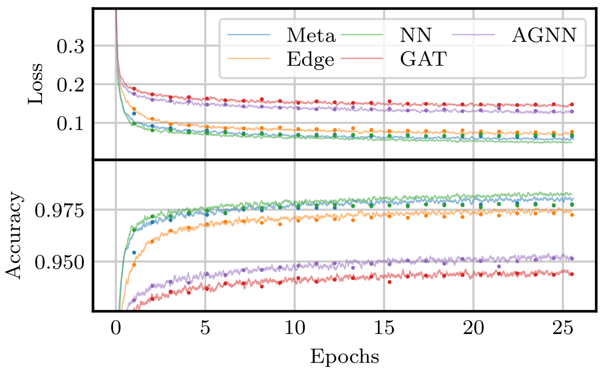

FIG. 14. Edge score loss and edge prediction accuracy for the different node updater architecture. The training curves are represented as lines and the validation points as round markers with statistical error bars.

E. Objective

The target adjacency matrix may be defined in different ways. The only requirement is that it forms at least a tree within each group of nodes, so that the true partition can be predicted on an edge basis. One possibility is to set the value of edges that connect two nodes within the same group to 1 and all others to 0. This encourages the network to build a cluster graph, i.e., a disjoint union of complete graphs.

The second possibility uses the score predictions to define an objective. A tree of $n - 1$ edges is built on each true group of n nodes so as to maximize the sum of adjacency scores of the tree edges. The CE loss is then only applied to those edges in the trees and those separating different node groups. This allows the network freedom as to which edge to turn on, as long as it builds a forest, i.e., a disjoint union of trees. In the following discussion, the first objective is referred to as the *cluster* target and the second as the *forest* target.

Figure 16 shows the training edge accuracy and edge loss for the two objectives defined above. Figure 17 shows their adjusted Rand index (ARI) [23] distribution on the test set. Both targets yield very similar results with a small lead for

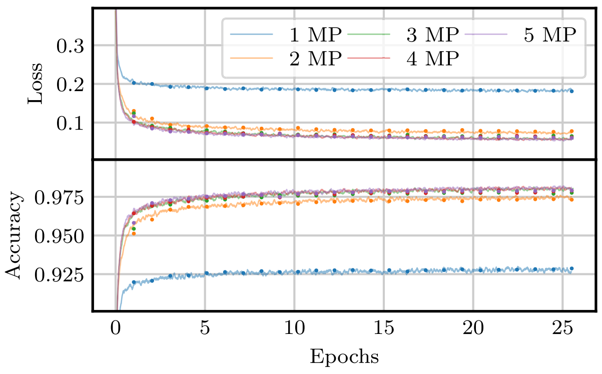

FIG. 15. Edge score loss and edge prediction accuracy for the different number of message passing steps. The training curves are represented as lines and the validation points as round markers with statistical error bars.

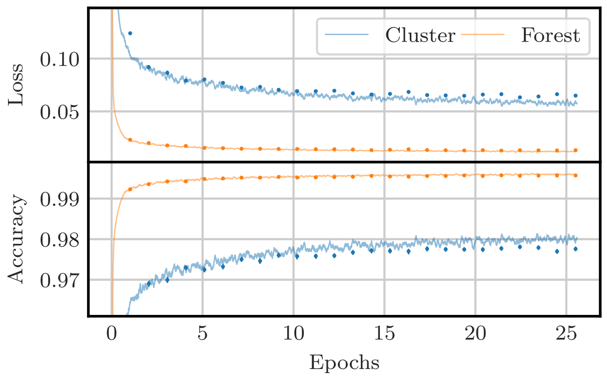

FIG. 16. Edge score loss and edge prediction accuracy for the different objectives under consideration. The training curves are represented as lines and the validation points as round markers with statistical error bars.

the baseline cluster target. Note that the forest target does not require the post-processing described in section II F, as not all edges connecting nodes within a group are trained to be activated. This also means that a single edge incorrectly turned on in a forest introduces mistakes at the inference stage.

IV. RESULTS

A. Training

The baseline model is trained for 25 epochs[8] on the edge and node prediction tasks using the DBSCAN-formed shower fragments and achieves a loss and accuracy summarized in Fig. 18. The edge classification accuracy is not affected by the addition of the node classification task nor the use of DBSCAN. The primary classification accuracy is close to 1 when initial points are known. The overall scale difference between the node and edge losses does not affect the training: if more weight is given to the node loss, the network trains slower, as the edge classification is essential to the node classification. The results presented in this section are produced using the baseline model.

B. Shower grouping

Figure 19 shows example outputs of the shower reconstruction algorithm for the four events that contain the most fragments in the test set. All four events have a high clustering accuracy, only missing or merging small fragments incorrectly. The bottom event highlights the importance of the partition loss optimization at the inference stage. Some edges are connecting two separate showers together but the loss minimization allows for the recovery of an almost perfect partition. The bottom left group of the fourth event originates from the same shower

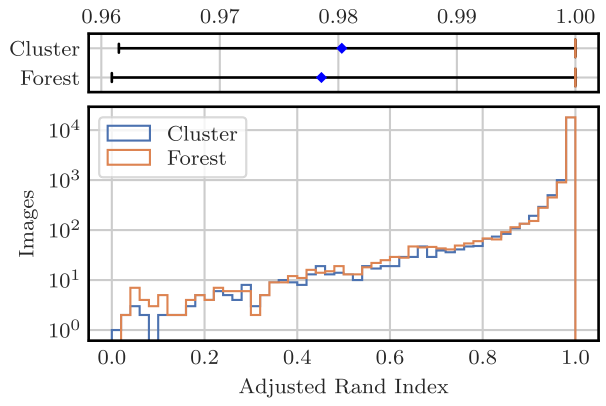

FIG. 17. Adjusted Rand Index (ARI) distributions for the different objectives under consideration. In the top box plot, the blue diamonds represent the means, the orange lines the medians and the whiskers span from the 10th to the 90th percentiles.

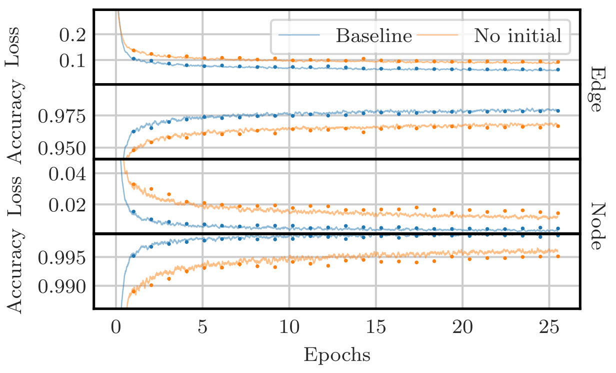

FIG. 18. Cross-entropy loss and classification accuracy of edges and nodes for the full shower reconstruction model, with and without initial points. The training curves are represented as lines and the validation points as round markers with statistical error bars.

but the primary is out of volume. The network correctly associated them together but would not be penalized for making a mistake there. Also note that there is no primary identification mistake in these examples.

Figure 20 shows the clustering metrics associated with the baseline model applied to the test set. The cases with ARI $= 0$ represent $\sim 1\%$ of all events in this test set and are omitted from the ARI distribution. Figure 21 shows the number of reconstructed showers as a function of the number of true showers in a single image. To prevent small fragments that are either omitted or merged to affect the histogram content, only shower instances with more than 100 voxels (~ 60 MeV) are included. In 95% of events, the estimated shower count is exact.

C. Primary identification

Figure 22 shows the distributions of primary scores for ground-truth primary and secondary nodes in the test set.

[8]The validation set accuracy does not improve beyond this mark.

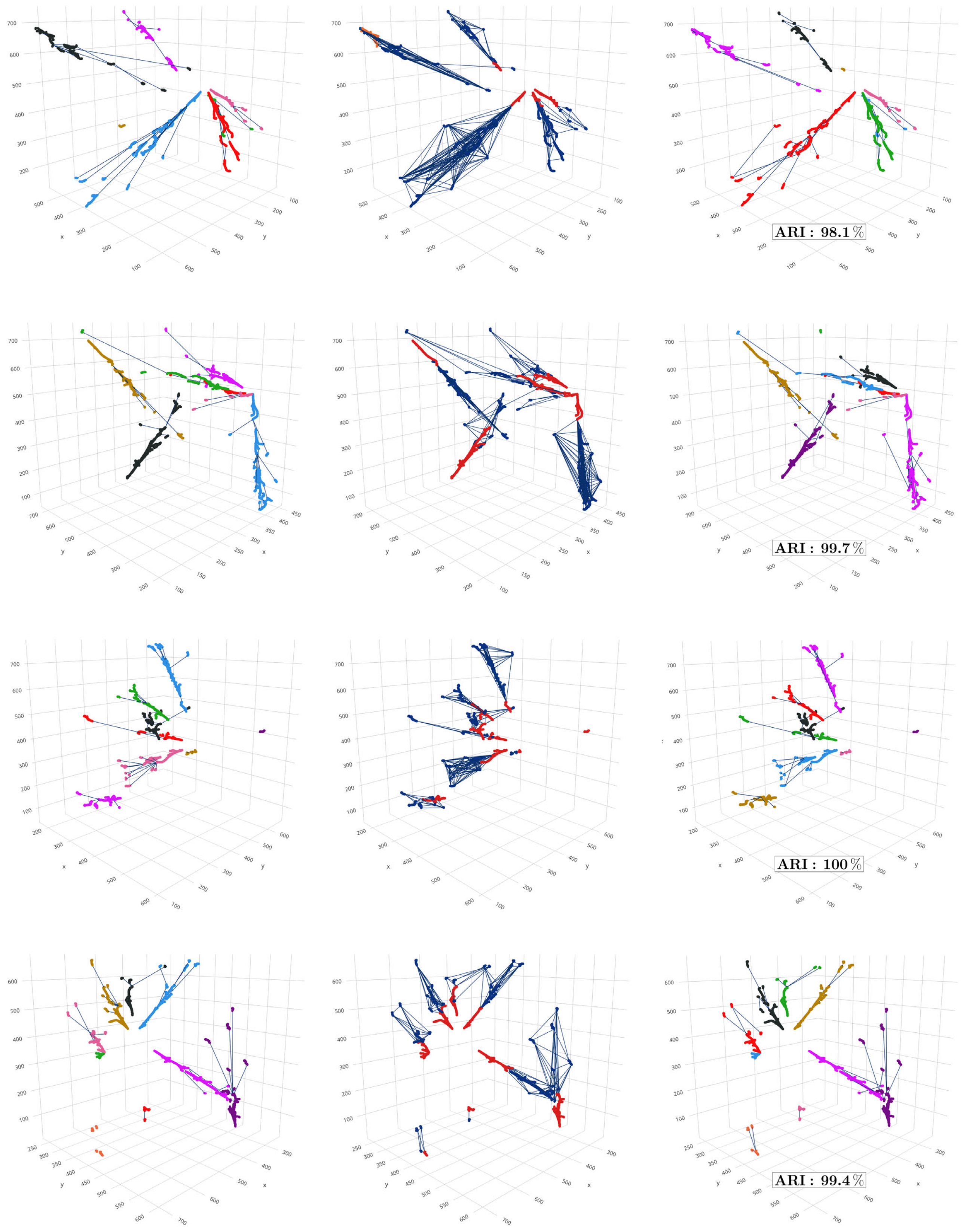

FIG. 19. Shower clustering predictions for the four events with the highest number of shower fragments in the test dataset (one event per row). Left: ground-truth shower labels (color) and edges representing the true fragment parentage. Middle: primary node scores represented as a node color ranging from 0 (blue) to 1 (red) and edges with an adjacency score > 0.5 (the closer to 1, the darker the edge). Right: inferred shower labels (color) and selected edges. Axes values represent voxel coordinates.

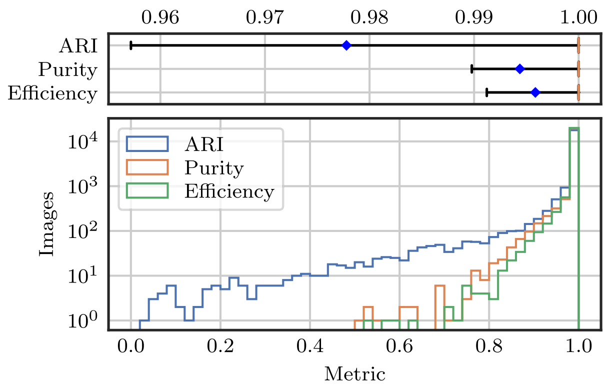

FIG. 20. Distributions of the three clustering metrics for the shower clustering task. In the top box plot, the blue diamonds represent the means, the orange lines the medians and the whiskers span from the 10th to the 90th percentiles.

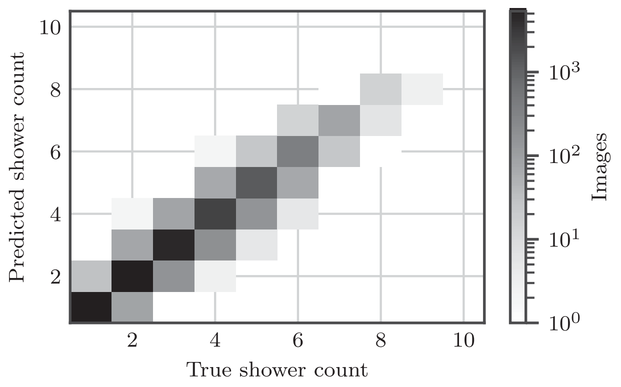

FIG. 21. Number of predicted showers as a function of the number of true showers in an event. Only shower instances with over 100 voxels are included.

This study shows that 0.08% of secondary fragments have a score larger than 0.5 and 0.84% of primary fragments have a score below 0.5.

Given the partition predicted by the network, a single primary fragment is assigned to each shower group by selecting the one with highest score. This scheme yields a group-wise primary identification accuracy of 99.77% for showers consisting of two or more fragments. This task is relatively trivial given an understanding of the direction of travel of the shower. The prior knowledge of fragment initial points helps, but even without them, the primary identification accuracy is still at 98.94%.

D. Shower energy resolution

For each ground-truth shower instance in the dataset, the total amount of energy that it deposits inside the image volume is integrated—including the fragments smaller than size 10 that are not in the input to the reconstruction chain—to form the ground-truth shower energy, E. For

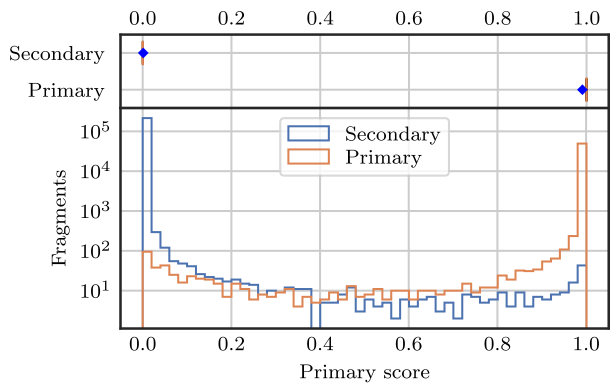

FIG. 22. Fragment primary scores of ground-truth primary nodes and ground-truth secondary nodes. In the top box plot, the blue diamonds represent the mean scores and the orange lines the median scores.

each true shower, the reconstructed cluster with the highest overlap is selected and the energy of its voxels is summed to form an energy estimate, $\hat{E}$. This estimate is multiplied by a fudge factor of 1.211 to compensate for the energy lost in small fragments measured in Fig. 5. In order to assess the importance of the calorimetric information on the shower energy resolution, the energy is also estimated using the voxel count alone divided by a factor 1.69, obtained by fitting the relation between the true number of EM voxels and the energy deposition, as shown in Fig. 23.

Figure 24 shows a distribution of the relative shower energy residuals for the showers that are at least 95% contained inside the images of the test dataset. The residuals are provided with and without leveraging the calorimetric information. These results show that the uncertainty on the shower energy is significantly driven by the prior fragment size selection. Figure 25 shows the 1σ energy uncertainty as a function of the shower energy.

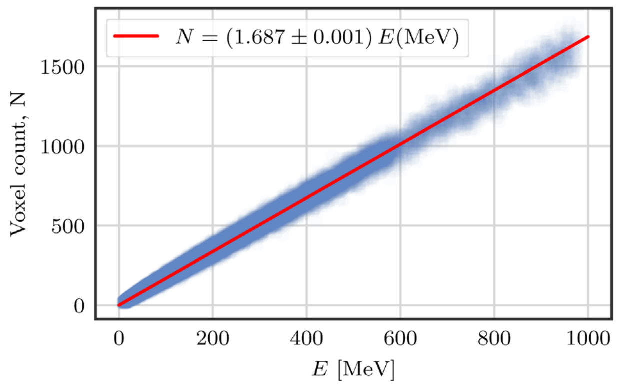

FIG. 23. Number of shower voxels contained in shower fragments of size 10 and above as a function of the total energy deposited by the shower.

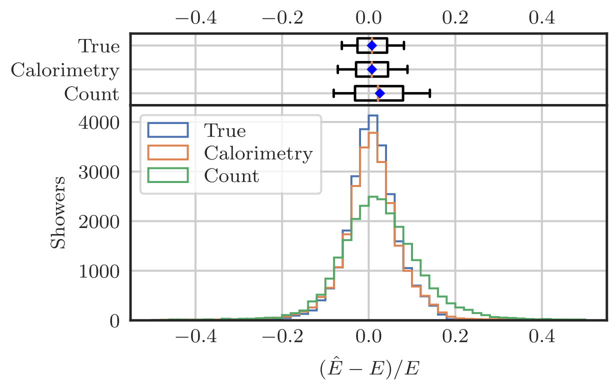

FIG. 24. Reconstructed relative shower energy residual distribution. In the top box plot, the blue diamonds represent the means, orange lines the medians, the boxes the IQRs and the whiskers span from the 10th to the 90th percentiles. "True" uses the true energy of true clusters, "calorimetry" the true energy depositions of reconstructed clusters and "count" a constant factor applied to the reconstructed voxel count.

The uncertainty decreases as the shower energy increases and reaches an accuracy around 5% at 1 GeV.

E. Shower angular resolution

For each ground-truth shower in the dataset, its true direction is obtained by normalizing its primary momentum vector to its norm, $\boldsymbol{d} = \boldsymbol{p}/|\boldsymbol{p}|$. In practice, the fitted direction, $\hat{\boldsymbol{d}}$, is estimated by taking the mean direction from the predicted primary initial point to the primary points with a neighborhood radius R_n, as shown in Eqs. (10). Setting R_n to a constant is suboptimal because the geometry of primary fragments can vary significantly from one shower to another. Smaller radii are preferable for

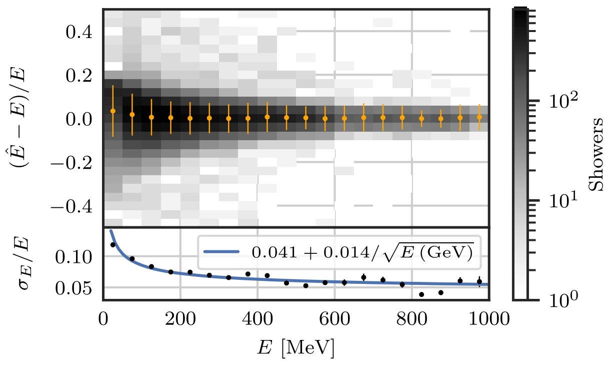

FIG. 25. Reconstructed relative shower energy residual distribution as a function of the shower energy. The orange markers on the top pad represent the means and their error bars the RMS; the latter is shown on its own in the bottom pad.

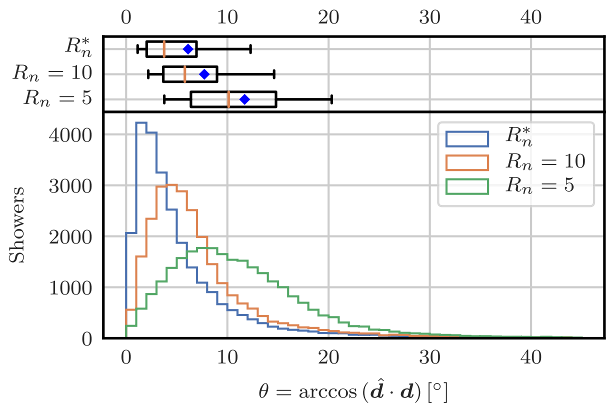

FIG. 26. Reconstructed shower direction residual distribution. In the top box plot, the blue diamonds represents the means, the orange lines the medians, the boxes the IQRs and the whiskers span from the 10th to the 90th percentiles.

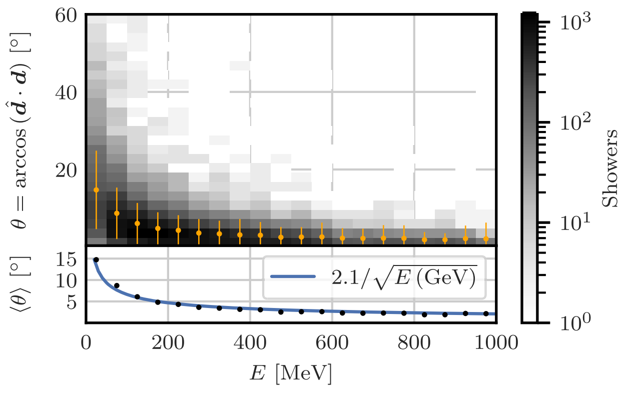

FIG. 27. Reconstructed shower direction residual distribution as a function of the shower energy. The orange markers on the top pad represent the mean and their error bars the RMS; the former is shown on its own in the bottom pad.

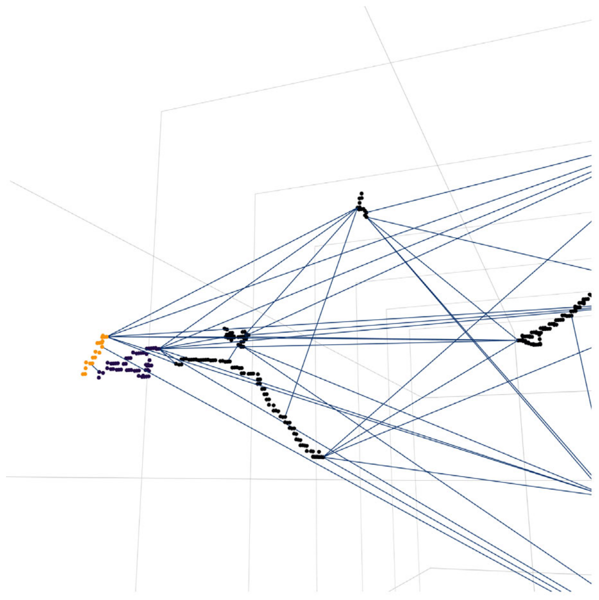

FIG. 28. Primary scores for an ambiguous shower start with a color scale that ranges from 0 (black) to bright yellow (1).

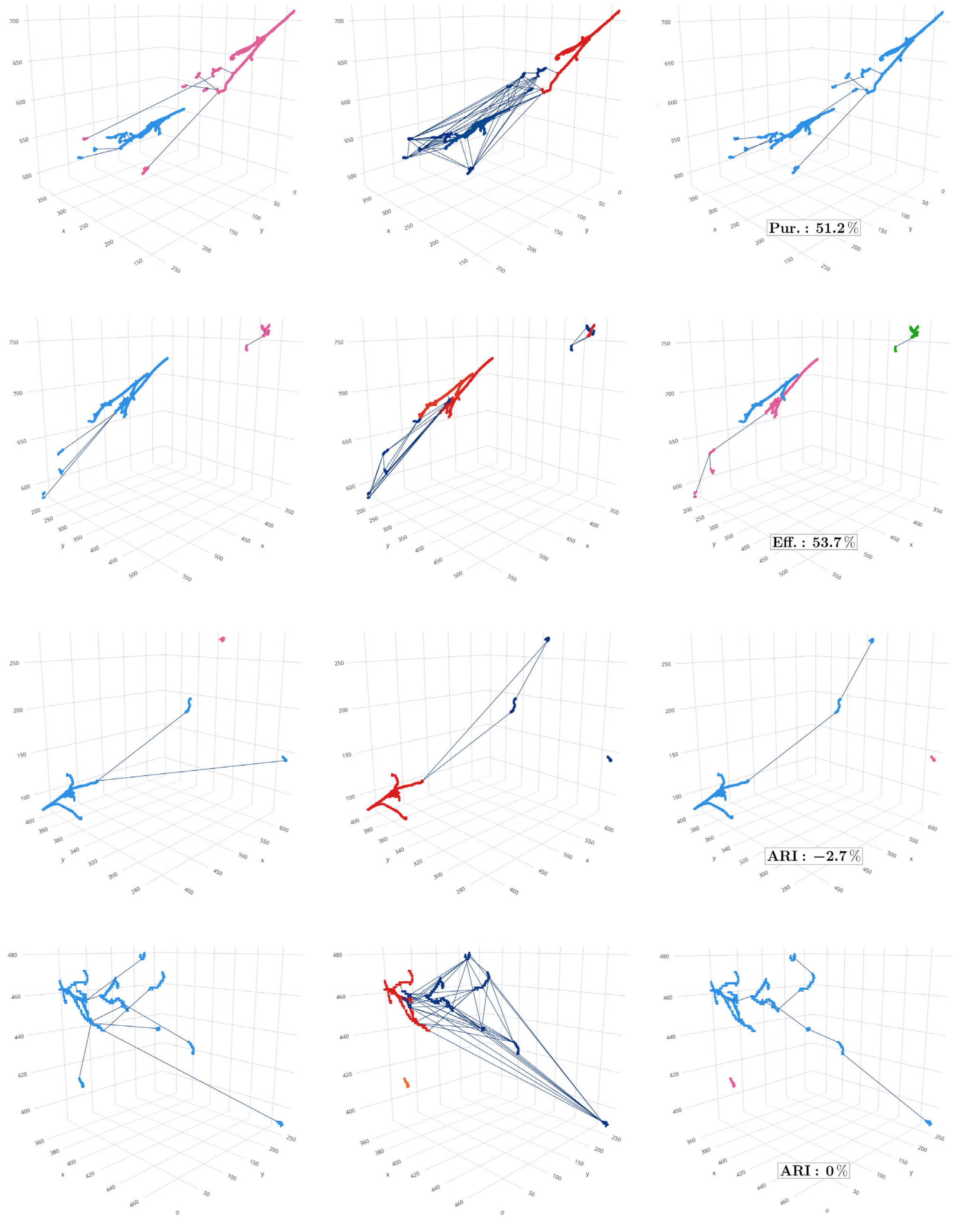

FIG. 29. Shower clustering predictions with the largest mistakes in three categories and one with an ARI of 0 (one event per row). Left: ground-truth shower labels (color) and edges representing the true fragment parentage. Middle: primary node scores represented as a node color ranging from 0 (blue) to 1 (red) and edges with an adjacency score >0.5 (the closer to 1, the darker the edge). Right: inferred shower labels (color) and selected edges. Axes values represent voxel coordinates.

fragments that curve or branch out a lot, but larger radii are advantageous for mostly linear showers.

The radius is optimized in order to minimize the relative deviation of the points from a straight line. The primary fragment points are ordered from closest to farthest from the initial point, $\{\boldsymbol{v}\}_{i=1}^{n}$. The mean, $\bar{\boldsymbol{v}}_k$, and covariance matrix, $\boldsymbol{\Sigma}_k$, are first evaluated for the closest three points (as the covariance matrix is undefined for $k < 3$). The mean and covariance matrix are then iteratively updated for $k = 4, \ldots, n$ following

$$\bar{\boldsymbol{v}}_k = \frac{1}{k}((k-1)\bar{\boldsymbol{v}}_{k-1} + \boldsymbol{v}_k), \tag{15}$$

$$\boldsymbol{\Sigma}_k = \frac{k-1}{k}\boldsymbol{\Sigma}_{k-1} + \frac{1}{k-1}(\boldsymbol{v}_k - \bar{\boldsymbol{v}}_k)(\boldsymbol{v}_k - \bar{\boldsymbol{v}}_k)^T. \tag{16}$$

For each combination of points, k, the ordered eigenvalues of the covariance matrix, $\{\lambda_{k,i}\}_{i=1}^{3}$, are evaluated. The optimal neighborhood of points minimizes the spread around the principal axis, i.e.,

$$k^* = \min_k \frac{\lambda_{k,1} + \lambda_{k,2}}{\lambda_{k,3}}. \tag{17}$$

Figure 26 shows the angular distribution between the true direction and the estimate, $\theta = \arccos(\hat{\boldsymbol{d}} \cdot \boldsymbol{d})$, for all the true showers in the dataset. The residual angle distribution has a mode of ∼2°, a mean of 6.1° and a median of 3.8°, when using the optimized neighborhood, R_n^*. The distributions for fixed neighborhood radii perform significantly worse. A radius of 5, while on average optimal for all fragments as shown in Fig. 9, carries a large uncertainty when used for primary fragments. Figure 27 shows the angular uncertainty as a function of the shower energy. It shows that the uncertainty decreases significantly with energy to reach a mean as low as ∼2.0°.

F. Mistakes analysis

A study of the events with low clustering purity reveals that the algorithm occasionally merges showers when the direction vector of one of its fragments can clearly be back-propagated to another shower fragment from a distinct instance. This may stem from an inconsistency between the true photon momentum and the local direction estimate of the fragment. The top row of Fig. 28 shows an event with a purity of 0.51 in the test set. Events with a purity < 0.9 represent ∼1% of this dataset. The middle row shows an event with an efficiency of 0.54 in test set. Events with an efficiency < 0.9 represent ∼0.7% of this dataset. This event showcases the difficulty to choose whether to merge large colinear fragments or to separate them. The third row shows the event with the lowest ARI and the fourth with an ARI of 0. The last example highlights that a clustering may have an ARI of 0 but be mostly accurate, if the number of true or predicted showers is one.

There is a total of 119 showers in the whole test set that have a misidentified primary. The majority of those mistakes stem from an incorrect partition of the nodes. The remaining mistakes can be attributed to ambiguous shower starts that do not have a fragment clearly upstream of the others. An example is provided in Fig. 29. The network picks the leftmost fragment but the one directly to its right is the labeled shower start. The network shows its uncertainty by giving scores of 0.76 and 0.13 to the left and right fragments, respectively.

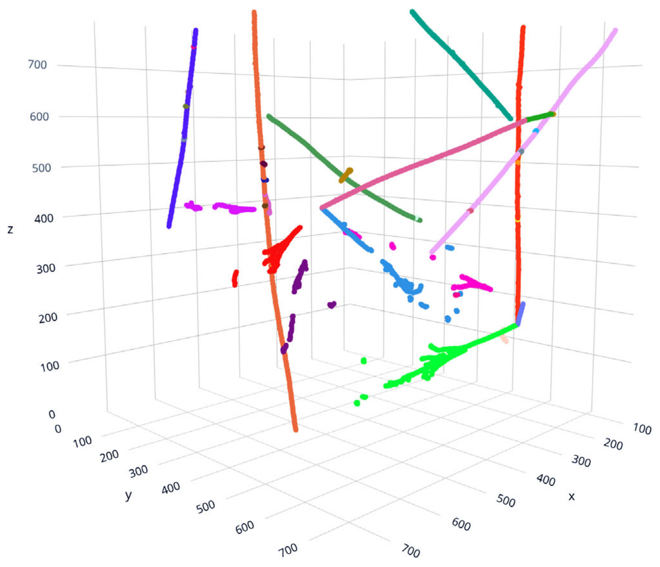

FIG. 30. Particle labels of a superimposition of two events in the test set. Axes values represent voxel coordinates.

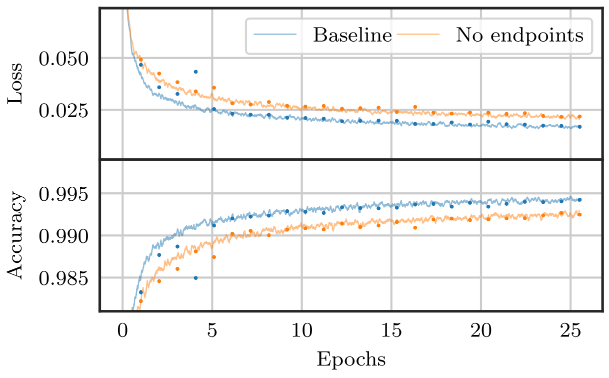

FIG. 31. Edge score loss and edge prediction accuracy for the different interaction clustering models. The training curves are represented as lines and the validation points as round markers with an error bar.

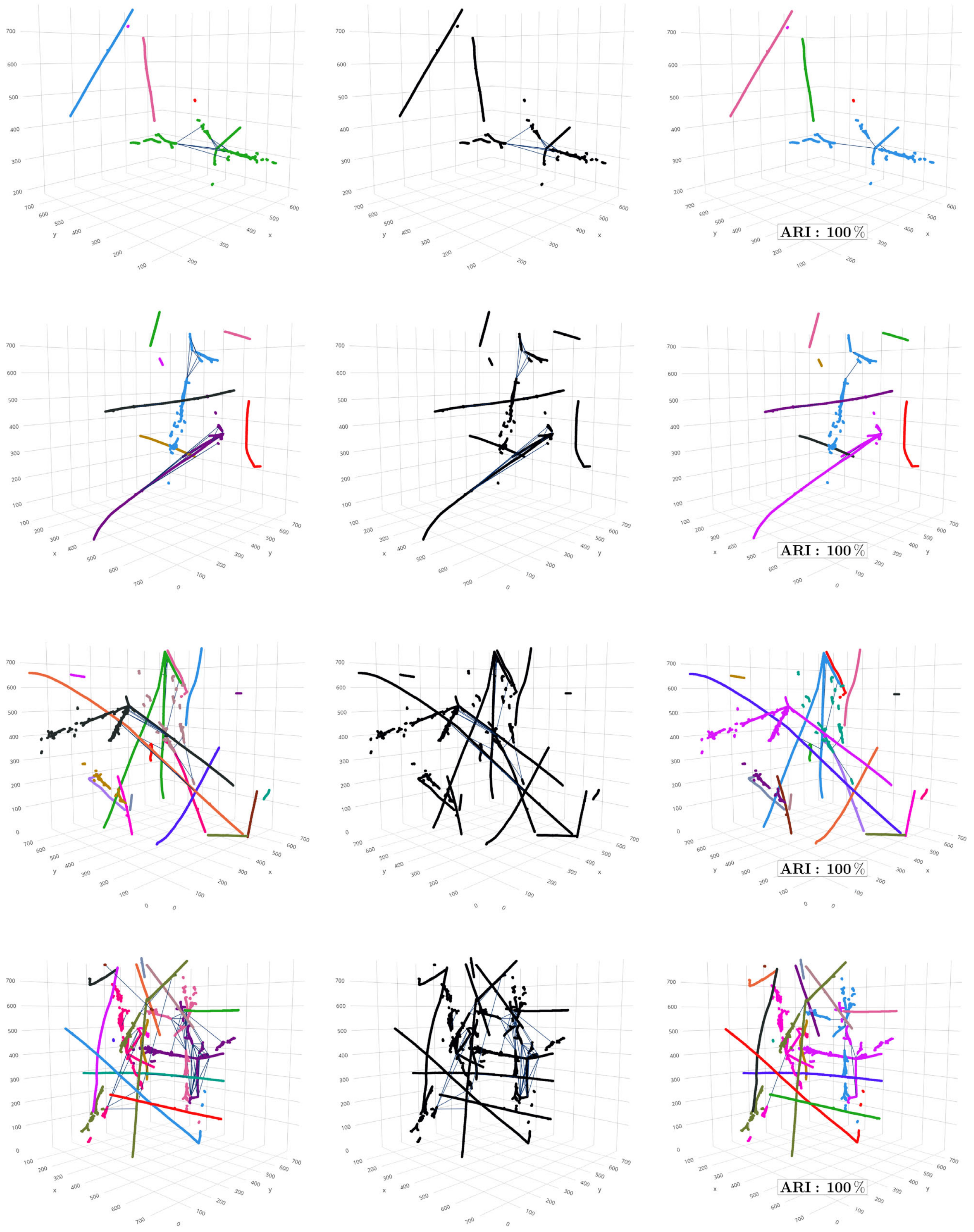

FIG. 32. Interaction clustering predictions on four randomly picked events with 1, 2, 3 and 4 randomly merged images (from top to bottom). Left: ground-truth interaction labels (color) and ground-truth cluster graph edges. Middle: edges with an adjacency score > 0.5 (the closer to 1, the darker the edge). Right: inferred interaction labels (color) and selected edges. Axes values represent voxel coordinates.

V. INTERACTION CLUSTERING

A. Modifications

Interaction clustering is defined as the association of particle instances together into groups that share a common particle ancestor. In this study, the individual particle instances are assumed to be known *a priori* from the previous reconstruction steps. In the full reconstruction chain, the shower instances will be provided by the reconstruction algorithm described in the previous sections while tracks, Michel and Deltas are to be clustered by a separate algorithm such as DBSCAN [19] or a CNN-based dense clustering algorithm [33].

Images simulated for this dataset only contain a single interaction vertex and multiple stray tracks and showers. In order to teach the network to separate multiple interaction vertices, several of these images may be stacked together. At the training stage, the number of images that are stacked together before being fed to the network follows a Poisson distribution of mean 2. Figure 30 shows an example of two stacked images and their particle labels.

This task utilizes an identical reconstruction chain to that used for the shower clustering. The input to the chain consists of particle instances instead of shower fragments and the target is interaction instances. The edge features are identically defined while node features are extended by adding the number-encoded particle class (0–4 corresponding to shower, track, Michel and delta rays, respectively), the mean and RMS energy deposition and the terminal point of tracks (other classes do not have well defined end points and are given a duplicate of the initial point instead).

Downstream of the message passing stage, the updated node features are not explicitly used to make any prediction. The edge features are the basis for an adjacency matrix prediction, while the groups are extracted by using the method described in Sec. II F. Figure 31 shows the training and validation edge classification loss and accuracy for the interaction clustering task. These metrics are evaluated with and without the endpoint information; adding the endpoints to the particle instances increases the edge classification accuracy by ∼0.2%.

B. Performance

Figure 32 shows the output of the interaction clustering algorithm for four randomly selected events in the test set, with one to four interaction vertices stacked together. All four events have a high clustering accuracy, only missing or merging small particles incorrectly.

The metrics described in Sec. II G are used to systematically characterize the performance of the reconstruction chain applied to the interaction clustering task. Figure 33 shows the clustering performance as a function of the number of images that are superimposed.

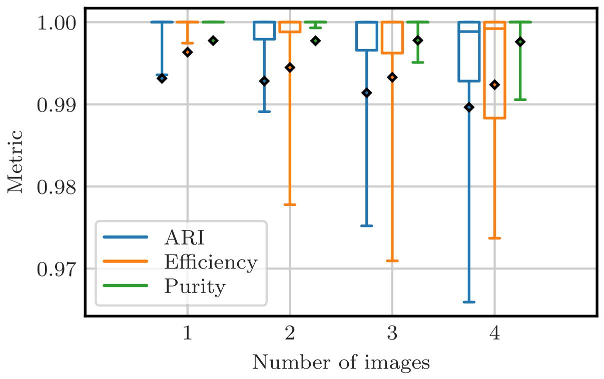

FIG. 33. Interaction clustering metrics as a function of the number of interactions in the image. The diamonds represent the means, the lines the medians, the boxes the IQRs and the whiskers span from the 10th to the 90th percentiles.

As shown in Fig. 3, each image contains one neutrino-like interaction of 4.3 ± 1.6 particles overlayed with 3.8 ± 1.7 randomly scattered cosmic-like interactions. For an image of $\sim 12\ \mathrm{m}^3$, this corresponds to an interaction density of 0.40 ± 0.14 interactions/m^3, which increases linearly with the number of superimposed images. The density observed of a single image, for instance, is equivalent to ∼120 interactions in a single ICARUS image, far above the expected rate [5]. A stack of two images contains two neutrinolike interactions, which corresponds to ∼18 such interactions in the DUNE-ND volume, close to the maximum expected rate [9], overlayed with ∼30 cosmic rays, which is unrealistically large. This demonstrates that this algorithm should easily deal with the expected rate of interactions in the foreseeable future.

C. Mistakes analysis

Figure 34 shows the three events that are reconstructed with the lowest purity, efficiency and ARI on the first three rows, respectively, and an event with an ARI of 0. The first event exhibits a purity of 56.1% due to a cosmic muon crossing and overlapping one of the vertex tracks. Events with purity < 0.9 represent ∼0.6% of this test set. The second event has an efficiency of 49.7% as the correlation between showers is not found sufficient by the network to associate them in an interaction. This may be due to a direction estimate not accurately representing the shower momentum or the vertex not being clearly defined by a track. Events with efficiency < 0.9 represent ∼0.8% of this test set. The bottom two rows show an event with an ARI of −1.3% and one with an ARI of 0. These two examples have a very low ARI but only contain minor mistakes, merging a small fragment it should not while omitting another.

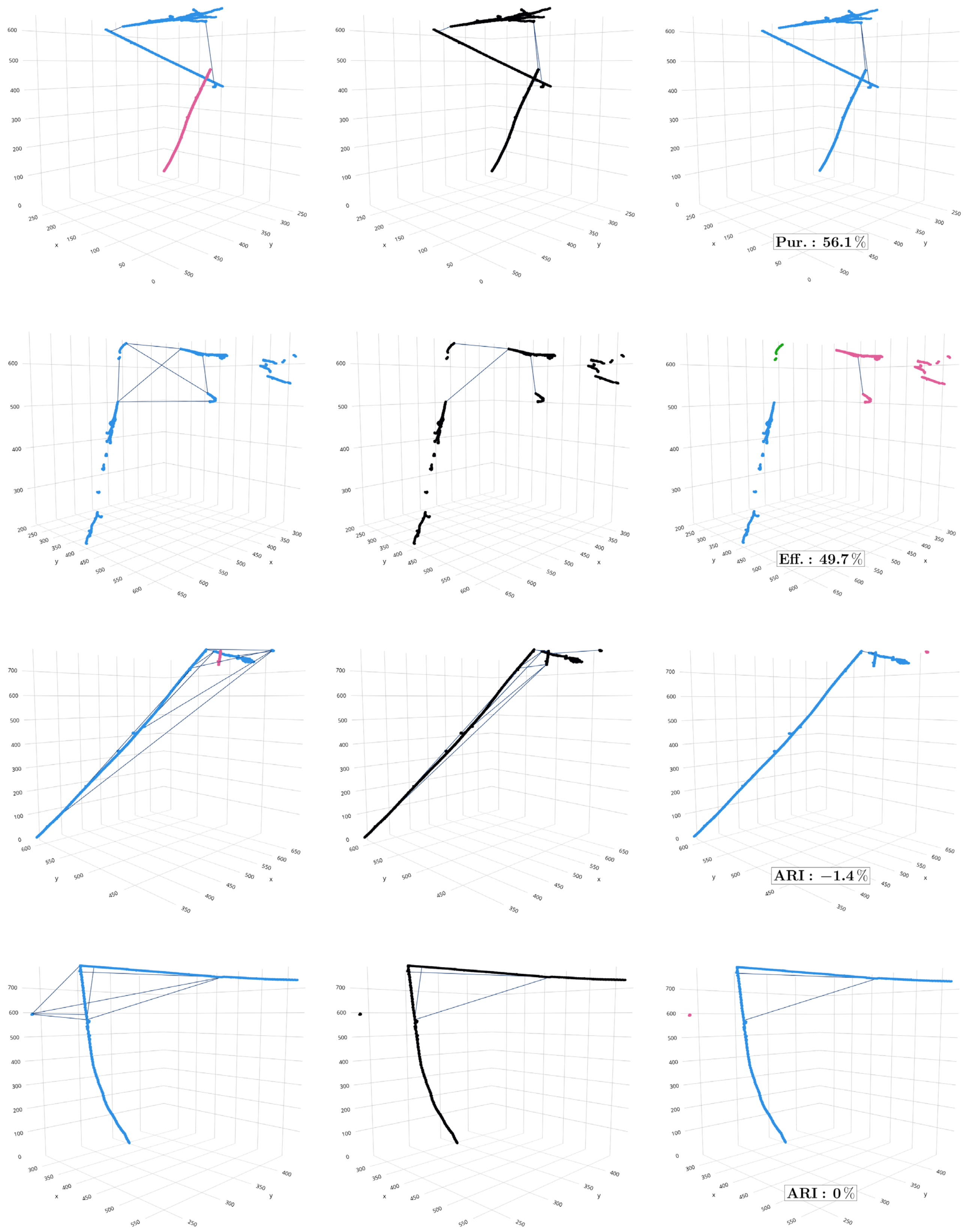

FIG. 34. Interaction clustering predictions with the largest mistakes in three categories and one with an ARI of 0 (one event per row). Left: ground-truth interaction labels (color) and ground-truth cluster graph edges. Middle: edges with an adjacency score > 0.5 (the closer to 1, the darker the edge). Right: inferred interaction labels (color) and selected edges. Axes values represent voxel coordinates.

VI. CONCLUSION

Graph neural networks (GNNs) are an ideally suited method to tackle the clustering of spatially detached objects in liquid argon time projection chambers (LArTPCs). A GNN-based reconstruction chain was developed to cluster electromagnetic showers and particle interactions. This paper studied its performance on a generic 3D sample of particle interactions in liquid argon and demonstrated a clustering efficiency and purity well above 99% for both tasks.

A good shower energy resolution is a core requirement for the upcoming SBN program and DUNE experiment to reach their scientific goals. The reconstruction of the shower direction will be of central importance when matching neutral pion decay showers together or when back-propagating photons to a vertex. The clustering of particle into interactions will become essential for future high-rate LArTPCs and the GNN algorithm developed here shows that this can be achieved. The algorithm described in this paper will be part of an end-to-end, machine-learning-based reconstruction chain developed at SLAC for all LArTPCs.

The accuracy of this algorithm is expected to be affected by that of the upstream reconstruction modules in the chain. The performance of the reconstruction chain as a whole has been evaluated on a the same simulated dataset and will be the subject of an upcoming paper. Discrepancies between data and simulations may also deteriorate the accuracy of the GNN-based clustering algorithm. Methods for mitigating the effect of such differences are actively being researched and will be discussed in future publications.

ACKNOWLEDGMENTS

This work is supported by the U.S. Department of Energy, Office of Science, Office of High Energy Physics, and Early Career Research Program under Contract No. DE-AC02-76SF00515 (SLAC Fundamental Particle Physics) and No. DE-AC02-05CH11231 (CompHEP Exa.TrkX).

[1] B. Baller *et al.*, J. Instrum. **9**, T05005 (2014).

[2] C. Rubbia, Report No. CERN-EP-INT-77-08 (1977).

[3] R. Acciarri *et al.* (ArgoNeuT Collaboration), Phys. Rev. D **95**, 072005 (2017).

[4] R. Acciarri *et al.* (ArgoNeuT Collaboration), Phys. Rev. D **99**, 012002 (2019).

[5] M. Antonello *et al.* (MicroBooNE, LAr1-ND, and ICARUS-WA104 Collaborations), arXiv:1503.01520.

[6] A. A. Aguilar-Arevalo *et al.* (MiniBooNE Collaboration), Phys. Rev. Lett. **121**, 221801 (2018).

[7] R. Acciarri *et al.* (MicroBooNE Collaboration), J. Instrum. **12**, P02017 (2017).

[8] S. Amerio *et al.* (ICARUS Collaboration), Nucl. Instrum. Methods Phys. Res., Sect. A **527**, 329 (2004).

[9] R. Acciarri *et al.* (DUNE Collaboration), arXiv:1601.02984.

[10] C. Adams *et al.*, J. Instrum. **15**, P02007 (2020).

[11] A. Ankowski *et al.* (ICARUS Collaboration), Acta Phys. Pol. B **41**, 103 (2010), https://www.actaphys.uj.edu.pl/R/41/1/103.

[12] R. Acciarri *et al.*, Eur. Phys. J. C **78**, 82 (2018).

[13] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, and M. Sun, arXiv:1812.08434.

[14] J. Duarte and J.-R. Vlimant, arXiv:2012.01249.

[15] V. V. Sochat, C. J. Prybol, and G. M. Kurtzer, PLoS One **12**, 1 (2017).

[16] C. Adams, K. Terao, and T. Wongjirad, arXiv:2006.01993.

[17] L. Dominé and K. Terao (DeepLearnPhysics Collaboration), Phys. Rev. D **102**, 012005 (2020).

[18] C. Adams *et al.* (MicroBooNE Collaboration), Phys. Rev. D **99**, 092001 (2019).

[19] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96 (AAAI Press, California, USA, 1996), pp. 226–231.

[20] P. W. Battaglia *et al.*, arXiv:1806.01261.

[21] C. M. Bishop, *Neural Networks for Pattern Recognition* (Oxford University Press, Inc., New York, 1995).

[22] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, Third Edition*, 3rd ed. (The MIT Press, Cambridge, MA, 2009).

[23] W. M. Rand, J. Am. Stat. Assoc. **66**, 846 (1971).

[24] D. P. Kingma and J. Ba, arXiv:1412.6980.

[25] L. Dominé *et al.*, arXiv:2006.14745.

[26] K. He, X. Zhang, S. Ren, and J. Sun, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 770–778.

[27] B. Graham and L. van der Maaten, arXiv:1706.01307.

[28] M. Fey and J. E. Lenssen, in *ICLR Workshop on Representation Learning on Graphs and Manifolds* (2019).

[29] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, in *Proceedings of the 34th International Conference on Machine Learning*, ICML'17 (2017), pp. 1263–1272.

[30] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, arXiv:1801.07829.

[31] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, arXiv:1710.10903.

[32] K. K. Thekumparampil, C. Wang, S. Oh, and L.-J. Li, arXiv:1803.03735.

[33] D. H. Koh *et al.*, arXiv:2007.03083.