Graph neural network for 3D classification of ambiguities and optical crosstalk in scintillator-based neutrino detectors

Saúl Alonso-Monsalve,[1,2,*] Dana Douqa,[3,†] César Jesús-Valls,[4] Thorsten Lux,[4] Sebastian Pina-Otey,[4,5] Federico Sánchez,[3] Davide Sgalaberna,[6] and Leigh H. Whitehead[7]

[1] *CERN, The European Organization for Nuclear Research, 1211 Meyrin, Switzerland*
[2] *Universidad Carlos III de Madrid, Av. de la Universidad, 30, 28911 Madrid, Spain*
[3] *University of Geneva, Section de Physique, DPNC, 1205 Genève, Switzerland*
[4] *IFAE, Institut de Física d'Altes Energies, Carrer de Can Magrans, 08193 Barcelona, Spain*
[5] *Aplicaciones en Informática Avanzada (AIA), 08172 Sant Cugat del Vallès Barcelona, Spain*
[6] *ETH Zurich, Institute for Particle Physics and Astrophysics, CH-8093 Zurich, Switzerland*
[7] *Cavendish Laboratory, University of Cambridge, Cambridge, CB3 0HE, United Kingdom*

Deep-learning tools are being used extensively in high energy physics and are becoming central in the reconstruction of neutrino interactions in particle detectors. In this work, we report on the performance of a graph neural network in assisting with particle set event reconstruction. The three-dimensional reconstruction of particle tracks produced in neutrino interactions can be subject to ambiguities due to high multiplicity signatures in the detector or leakage of signal between neighboring active detector volumes. Graph neural networks potentially have the capability of identifying all these features to boost the reconstruction performance. As an example case study, we tested a graph neural network, inspired by the GraphSAGE algorithm, on a novel 3D-granular plastic-scintillator detector, that will be used to upgrade the near detector of the T2K experiment. The developed neural network has been trained and tested on diverse neutrino interaction samples, showing very promising results: the classification of particle track voxels produced in the detector can be done with efficiencies and purities of 94-96% per event and most of the ambiguities can be identified and rejected, while being robust against systematic effects.

I. INTRODUCTION

Since 1999, a series of neutrino oscillation experiments have provided deep insight into the nature of neutrinos [1–8]. A number of these experiments are long-baseline neutrino oscillation experiments that use two detectors to characterize a beam of (anti-)neutrinos: a near detector, located a few hundred meters away from the target that measures the original beam composition, and a far detector, located several hundred kilometres away, that allows for the determination of the beam composition after neutrino flavor oscillations.

The energy of these beam neutrinos ranges from a few hundred MeV up to several GeV. Charged particles can be produced in neutrino interactions, and the energy that they deposit as they traverse the detector can be used to reconstruct the events. In general, the larger the energy transferred from the neutrino to the nucleus, the larger the number of particles and particle types produced in the final state. Modeling nuclear interactions in the target nuclei is highly complex, particularly for high energy transfers where the hadronic component of the interaction is more important. As a result, current long-baseline neutrino oscillation experiments mostly analyze interactions with low particle multiplicity. This situation, however, is expected to change in the coming years. On one hand, the statistical and systematic uncertainties of current experiments have decreased significantly over recent years such that neutrino-nucleus modeling is becoming a dominant source of uncertainty [8, 9]. On the other hand, future experiments like DUNE [10] will use a broad-band energy neutrino beam, expecting a significant fraction of the neutrino interactions to have a high energy transfer to the nucleus.

As a result, in recent years, the neutrino physics community has turned its attention to measuring neutrino-nucleus interaction cross-sections for different ranges of energies and target materials [11] as a way to constrain the oscillation uncertainties while providing new measurements to further develop the interaction models. In parallel, a new generation of neutrino detectors are under development that aim to resolve and reliably identify short particle tracks even in very complex interactions. To achieve this, two main detector technologies stand out: one is based on Liquid Argon Time-Projection-Chambers (LArTPCs) [12] and the other is based on finely segmented plastic scintillators with three readout views [13] that will form part of the near detectors for T2K [14] and, possibly, DUNE [15].

For the latter, the detector response to a charged particle is read out into three orthogonal 2D projections. When reconstructing the 3D neutrino event, different types of hits are rebuilt, introducing non-physical entities that can hinder the reconstruction process. Due to the spatial disposition of such hits, an approach of utilizing Graph Neural Networks (GNNs) [16] is proposed to perform the classification of 3D hits to provide clean tracks for event reconstruction.

The article proceeds in the following way: Sec. II describes properly the motivation behind the methodology

* E-mail: saul.alonso.monsalve@cern.ch
† E-mail: dana.douqa@unige.ch

given the details of the detector technology. Section II introduces deep-learning techniques and explains the specific GNN algorithm used. The simulated data samples and GNN training are discussed in Sec. IV. Results and a study of systematic uncertainties are given in Secs. IV D and V, respectively, followed by concluding remarks in Sec. VI.

II. MOTIVATION

A finely segmented scintillator detector consists of a 3D matrix of plastic scintillator cubes. The scintillation light produced by charged particles traversing the cubes is read out by three orthogonal wavelength-shifting (WLS) fibers that transport the scintillation light out of the detector where silicon photomultipliers (SiPMs) convert it into a certain number of photoelectrons (p.e.), as illustrated in Figs. 1 and 2.

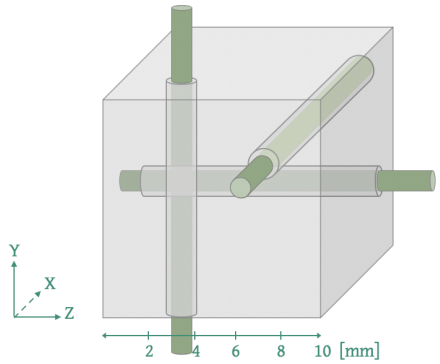


FIG. 1: Geometry of a single SuperFGD element. Each cube (gray) is intersected by three WLS fibers (green). The whole SuperFGD will be an array of $56 \times 184 \times 192$ of these elements ($H \times L \times W$).

Here, we consider the Super Fine-Grained Detector (SuperFGD) [14], which will be used in T2K, as a specific case-study. The detector will have 2 million plastic scintillator cubes, each $1 \times 1 \times 1\,\mathrm{cm}^3$ in size, and provides three orthogonal 2D projections of particle tracks produced by a neutrino interaction, as depicted in Fig. 4a.

To reconstruct neutrino interactions in three dimensions, the light yield measurements in the three 2D views are matched together, as shown in Fig. 4b. The 3D objects, corresponding to the cubes where the energy deposition is reconstructed, are referred to as *voxels*. In addition to the cubes where a particle has passed and deposited energy, light-leakage between neighboring cubes can create additional *crosstalk* signals [17, 18], as depicted in Fig. 2. Moreover, ambiguities in the matching process can give rise to *ghost* voxels, shown in Fig. 3.

To accurately reconstruct neutrino interactions in these detectors, it is crucial to be able to classify each voxel as one of the three types:

- **Track**: a voxel whose energy deposit comes, partially or totally, from scintillation light generated in that same cube.
- **Crosstalk**: a voxel whose energy deposit comes exclusively from light-leakage from neighboring cubes.
- **Ghost**: a voxel with no physical energy deposit with an apparent signal arising from ambiguities when matching the three 2D views into 3D.

Figure 4c shows the three types of voxels using truth information after 3D matching has been performed for an example neutrino interaction. Once these voxels are properly labeled (by a classification algorithm), the ghost voxels can be removed before the full event reconstruction proceeds, while simultaneously cleaning the particle tracks of crosstalk.

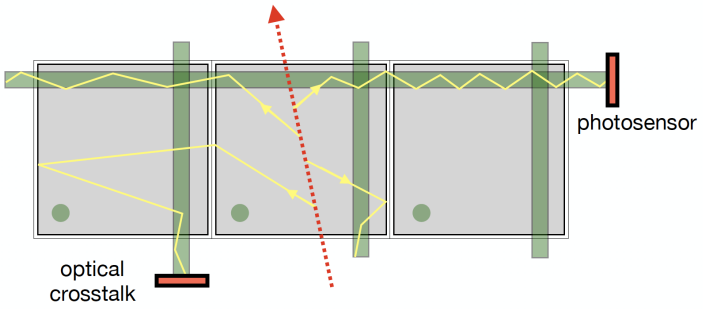


FIG. 2: Sketch of the signal generation, fiber transport, and signal detection processes highlighting the production of optical crosstalk signals. The cubes are depicted in gray, the WLS fibers in green, the dashed red line is a charged track and photons are illustrated in yellow.

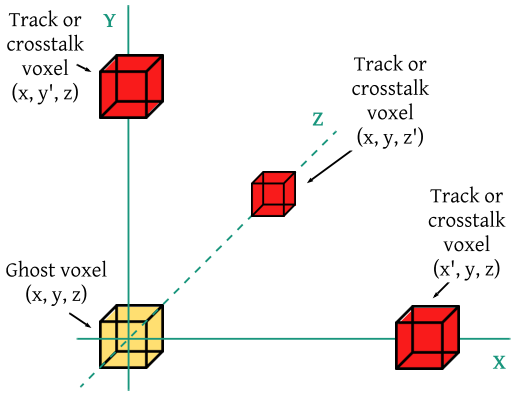
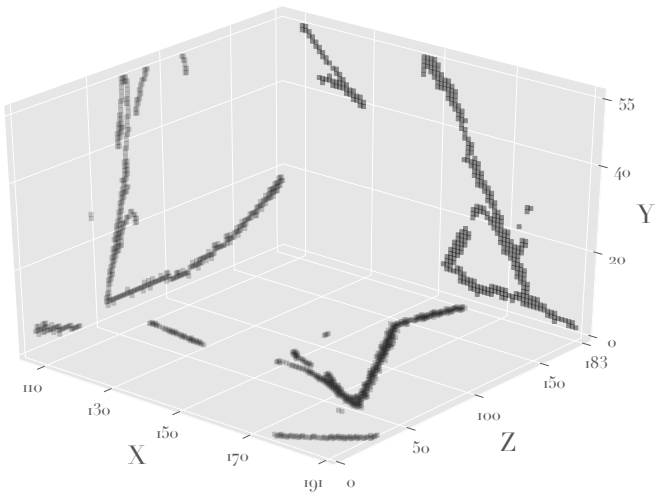
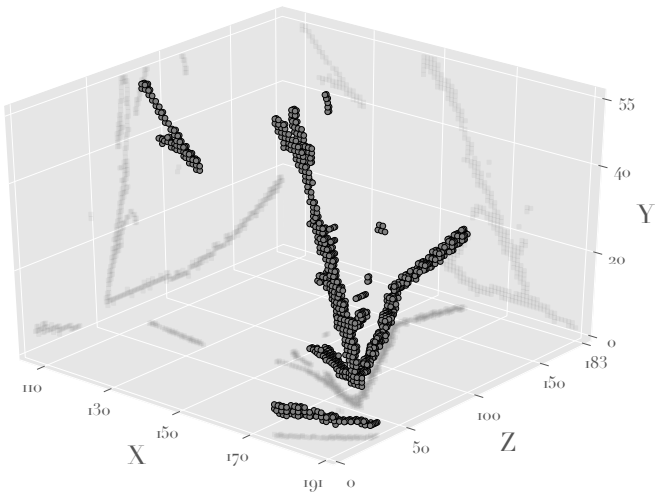


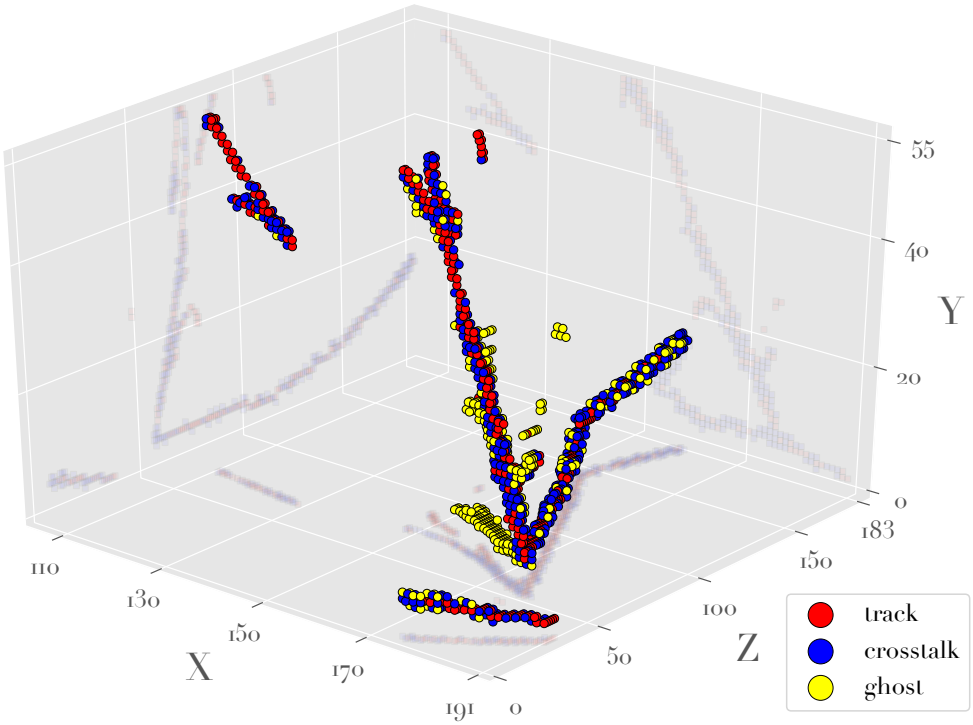
FIG. 3: Example of a ghost voxel arising from a 2D to 3D matching ambiguity. A 2D hit from each of the three track or crosstalk voxels (red) intersect generating a ghost voxel (yellow).



(a) Projections of the observed neutrino interaction onto the three 2D detector views (XY, XZ, and YZ).



(b) 3D view of the neutrino interaction after the 3D matching of the three 2D views in Figure 4a. The 3D voxels are shown as dark points. Projections of the observed neutrino interaction onto the three 2D detector views (XY, XZ, and YZ) are shown as shadow.



(c) 3D view of the neutrino interaction after the 3D matching of the three 2D views in Fig. 4a. The 3D voxels labelled as track (red), crosstalk (blue), and ghost (yellow) according to the truth information from the simulation. Projections of the observed neutrino interaction onto the three 2D detector views (XY, XZ, and YZ) are shown as shadows.

FIG. 4: Visualization of a neutrino interaction in a finely segmented 3D scintillator detector, demonstrating the relationship between the observed 2D projections onto the three orthogonal 2D views (Fig. 4a), the reconstructed 3D voxels (Fig. 4b), and the true classification of the voxels (Fig. 4c). The energy of the incoming neutrino is 4.754 GeV. The axes are in cm.

In this article, we represent the voxels as nodes in a graph and classify the signals using a deep-learning tech-

nique based on a GNN. The abstract data representation provided by graphs makes this method very versatile and applicable to any experiment where the output data from the detector elements can be represented as a list of features with arbitrary dimensionality.

In the case study presented here, focused on the SuperFGD detector of the T2K experiment, this method shows great potential to assist reconstruction by assigning a probability to each voxel as being track, crosstalk or ghost. Detector response simulations show that about half of the reconstructed voxels in the SuperFGD will not be of track type. For neutrino physics studies, both in terms of cross-sections and oscillation measurements, correct event topology identification and kinematic reconstruction of outgoing tracks are paramount. Ghost and crosstalk voxels, if not dealt with, will smear the track range estimations, and hence the reconstructed momentum and angle. Similarly, more densely populated events in term of voxels will merge highly colinear tracks and will make it more difficult to identify short tracks key for correct topology assignment. The method presented here is therefore expected to benefit future physics measurements in T2K and in any other experiments with similar conceptual challenges as the ones here described.

III. DEEP-LEARNING METHODS

A. Convolutional neural networks and data sparsity

Deep-learning techniques are now commonly applied within the field of neutrino physics. In particular, Convolutional Neural Network (CNN) [19] algorithms that operate on two-dimensional images of the neutrino interactions have been very successful in a number of tasks, such as event classification [10, 20–25], hit-level identification of track-like (linear) and shower-like (locally dense) energy deposits [26, 27], or energy reconstruction [28–30]. Despite the success of CNNs in the neutrino world, images of neutrino interactions are typically very sparse as only those readout channels with a detected signal contribute non-zero values to the images, and in the case of the detector presented in Sec. II the average occupancy of the detector for a neutrino interaction is less than 0.02%. Thus, much of the computation time is spent unnecessarily applying convolutions to empty regions of the images.

The goal of this work is to classify 3D voxels as one of three categories (track, crosstalk or ghost), which is natively a three dimensional problem. To apply a 3D CNN-based algorithm to this detector would require two million voxels to avoid any downsampling or cropping of the input data, which is computationally prohibitive. A popular approach to deal with the sparsity of neutrino interactions is the submanifold sparse convolutional network (SSCN) [31]. Standard "dense" CNNs are very inefficient when applied on images of neutrino interactions, whereas SSCNs require considerably less computation and report almost identical (or even better) results in terms of accuracy [32]. Some neutrino experiments have improved their reconstruction deep-learning algorithms by moving to SSCNs. For example, MicroBooNE recently updated the implementation of their semantic segmentation CNN [27] to an SCNN-based model [33], reporting improvements at inference by a factor of 354 and 33 in memory and wall-time, respectively. The NEXT collaboration are also exploring the idea of using an SSCN for track classification [34].

B. Graph neural networks

An alternative approach for handling with sparse data is to represent hits (or voxels) as nodes in a graph. In computer science, a graph $\mathcal{G}$ is a data structure that represents a mathematical concept consisting of nodes $\mathcal{V}$ and edges $\mathcal{E}$:

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}). \tag{1}$$

A graph can be directed, where each edge has a starting and an ending node that define a direction, or undirected, where the edge simply connects two nodes without inducing a sense of direction. In our case, we use an undirected graph, since we are only interested in the spatial connections between nodes. Figure 5 shows a comparison of the 3D CNN and graph data structures, as well as the radial search method used for defining edges between nodes.

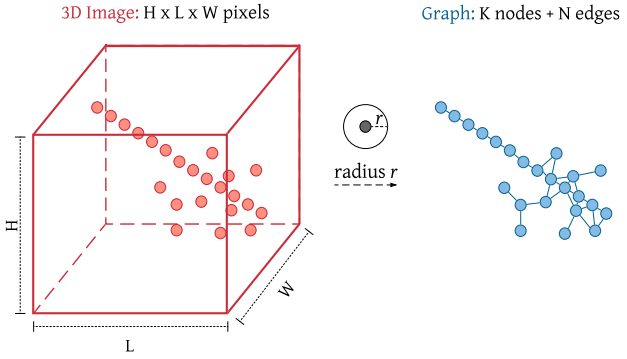


FIG. 5: Data and computation size comparison between a 3D image and a graph. The size of the 3D image on the left is fixed ($H \times L \times W$) regardless of the number of hits as CNNs require fixed image sizes (in most cases). The connected graph shown on the right is a much more efficient representation of the data. Each hit is represented as a graph node and connections, called edges, are made between neighboring hits within a sphere of radius r.

As mentioned above, each detector voxel cube is represented as a node in a graph, and each node consists of a list of input variables called features that describe the physical properties of the detected signal (see

Section IV and Appendix B). The deep-learning algorithm that operates on graphs is the Graph Neural Network (GNN) [16, 35]. GNNs are used in many different fields [36, 37] and can be applied for graph classification [38, 39] or node classification [40–42]. In this article, a GNN inspired by the GraphSAGE algorithm [42] is used to classify individual voxels in SuperFGD events. The application of GNNs to data from neutrino experiments has been recently demonstrated by the IceCube experiment in order to identify entire events as atmospheric neutrino interactions, outperforming a 3D CNN [43]. The work in Ref. [44] also shows an application of GNNs for both node and edge classification for a neutrino detector, where a GNN-based reconstruction chain is used for clustering both electromagnetic showers and particle interactions. Other GNN-based studies have been performed for particle reconstruction in high energy physics detectors [45–47]. The main drawback of GNNs with respect to SSCNs is that the former needs to pre-process the events to perform the neighborhood computation (defining edges) whilst no pre-processing is needed for the SSCN images. However, the advantage of GNNs in this field is that they can use a strong node representation, where a large number of features can define each node without reducing the scalability of the model. To the best of our knowledge, the approach we present in this paper is one of the first attempts of using GNNs for node classification in neutrino experiments.

C. GraphSAGE

GraphSAGE [42] is a technique that leverages the features of graph nodes $\mathcal{V}$ - which can range from physical information to text attributes - to generate efficient representations on previously unseen samples by learning aggregator functions from training nodes. These aggregators can be simple functions (e.g., mean or maximum) or more complex ones, such as Long short-term memory (LSTM) cells [48], and must be functions that take an arbitrary number of inputs without any given order. The model learns not only K aggregator functions that combine information from neighboring nodes but also a set of weight matrices $\mathbf{W}^k, \forall k \in \{1, ..., K\}$, which are used to propagate information through the K layers of the model and combines local information of the node with the aggregator information of its neighbors into an encoding vector (see Algorithm 1). The number of aggregator functions is also used to define the depth of the model, meaning that a GraphSAGE model has a depth of K. In each layer of the aggregator information, a new representation of the node v is computed, denoted by $\mathbf{h}_v^k$ (with $\mathbf{h}_v^0$ being the initial node features $\mathbf{x}_v$). Once trained, it can produce the embedding of a new node given its input features and neighborhood, in the form of the vector of the last layer $\mathbf{h}_v^K$; this embedding is then used as the input of a multilayer perceptron (MLP) [49] that is responsible for predicting the label.

Algorithm 1: GraphSAGE embedding generation (i.e., forward propagation) algorithm (from [42])

Input : Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$; input features $\{\mathbf{x}_v, \forall v \in \mathcal{V}\}$; depth K; weight matrices $\mathbf{W}^k, \forall k \in \{1, ..., K\}$; non-linearity σ; differentiable aggregator functions $\text{AGGREGATE}_k, \forall k \in \{1, ..., K\}$; neighborhood function $\mathcal{N} : v \rightarrow 2^{\mathcal{V}}$

Output: Vector representations $\mathbf{z}_v$ for all $v \in \mathcal{V}$

$\mathbf{h}_v^0 \leftarrow \mathbf{x}_v, \forall v \in \mathcal{V}$;
for $k = 1...K$ **do**
  for $v \in \mathcal{V}$ **do**
    $\mathbf{h}_{\mathcal{N}(v)}^k \leftarrow \text{AGGREGATE}_k(\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\})$;
    $\mathbf{h}_v^k \leftarrow \sigma\left(\mathbf{W}^k \cdot \text{CONCAT}(\mathbf{h}_v^{k-1}, \mathbf{h}_{\mathcal{N}(v)}^k)\right)$
  end
  $\mathbf{h}_v^k \leftarrow \mathbf{h}_v^k / \|\mathbf{h}_v^k\|_2, \forall v \in \mathcal{V}$
end
$\mathbf{z}_v \leftarrow \mathbf{h}_v^K, \forall v \in \mathcal{V}$

Since GraphSAGE learns from node features, it allows us to decide which physical information to use for each voxel. This means that the model can follow the particle set, i.e., by predicting the label for each voxel based on the physical attributes of the target voxel as well as the features of its neighbors.

IV. METHODOLOGY

A. Data sample generation

In order to generate data sets of neutrino interactions with true labels that allow to train and benchmark the classification algorithm, the steps below are followed. For each neutrino interaction:

1. Initial particle types and initial kinematics are specified for all final-state particles produced in the interaction.

2. Initial particles are propagated through the detector geometry producing further particles and leaving signals in the form of energy deposits.

3. Using particle energy deposits, the detector response is simulated.

4. The information is stored as a list of voxels with a unique integer known true label: track, crosstalk or ghost.

Initial particle types and kinematics
The initial particle types and their associated kinematics were simulated following two approaches. Firstly, GENIE datasets were created using `GENIE-G18.10b` neu-

trino interaction software [50]. For a given neutrino flux[i] and target geometry specification, it generates a list of realistic neutrino event interactions both in the number and type of outgoing particles, often referred to as event topologies, and in their individual initial kinematics. Secondly, Particle bomb (P-Bomb) datasets have been constructed as a complementary group of data not affected by the specific tunings provided by a neutrino interaction generator, such as GENIE. The motivation underlying this is to show in the later sections that the reported algorithm performance is not highly dependent on the neutrino interaction modelling. Moreover, given that neutrino generators do not perfectly model real interactions, these two datasets (GENIE and P-Bomb) are also used in the following sections to discuss the reliability of training in GENIE (or other generators) and classifying real data. Hence the purpose of P-Bomb datasets is to provide events similar to those found in neutrino interactions in terms of the outgoing particles but with no realistic kinematic modeling and without considering any kinematic correlations among the outgoing particles. To achieve this the P-Bomb dataset is constructed adding equal numbers of events with the following particle gun combinations, each of which has random flat solid angle and momentum [10-1000 MeV/c] distributions: 1 μ^-; 1 μ^- and 1 proton; 1 μ^- and 1 π^-; 1 μ^- and 1 π^+; 1 μ^- and 2 protons; and 1 μ^-, 1 π^+ and 3 protons. An illustrative comparison between GENIE and P-Bomb neutrino interaction modelling can be found in Appendix A. A summary regarding the number of events and voxels in the two datasets, as well as of the class distribution is presented in Tab. I.

GENIE dataset		**Training**	**Validation**	**Testing**
	# Events	6k	2k	11.5k
	# Voxels	1.83M	606.7k	3.58M
		Track	**Crosstalk**	**Ghost**
	Fraction	43%	37%	20%
P-Bomb dataset		**Training**	**Validation**	**Testing**
	# Events	6k	2k	39.5k
	# Voxels	1.84M	618k	12.3M
		Track	**Crosstalk**	**Ghost**
	Fraction	49%	38%	13%

TABLE I: Descriptions of both GENIE and P-Bomb datasets, displaying the number of events and number of voxels used for training, validating and testing the models. Additionally the fractions of the different classes of voxels are shown, which are conserved through the training, validating, and testing sets.

Particle propagation simulation in the detector

The SuperFGD detector geometry was simulated as described in Ref. [14]. The particle propagation and physics simulation is done by means of `GEANT v4-10.6.1` [52]. GEANT is a Monte Carlo based toolkit that provides realistic propagation of particles through matter. It outputs a list of energy deposits.

All energy deposits[ii] occurring in the same detector cube, including the effect of Birks' quenching [53], are summed to form the list of *track voxels*. To simulate imperfect cube light-tightness, the 3D voxelized energy is then shared with the neighboring cubes, creating a new set of voxels that originally had no energy deposits, the *crosstalk voxels* (see Figure 2). For the energy sharing, a fraction of the energy in the original cube is leaked into each of its six neighbors. The fraction that is shared is sampled from a Poisson distribution, with $\mu = 2.7\%$. Given that the probability for the energy to leak twice is $O(\mu^2)$, only leakage to immediate neighbors is considered. The 3D voxelized energy of both track and crosstalk voxels is projected onto its three orthogonal planes where the detector 2D signals are simulated, converting the continuous energy deposit into discretized photons[iii], weighted by distance-dependent attenuation factors, which are detected with 35% probability. To mimic a minimum threshold detection sensitivity, only 2D hits with three or more detected photons are kept. SuperFGD thresholds at this level are expected to remove virtually all dark rate hits [17], henceforth we have not included noise hits in our simulation. Then, the 2D hits are matched into 3D reconstructed voxels only if the same XYZ coordinate combination can be made using two different combinations of 2D planes. In this process, due to ambiguities some extra voxels are created, the *ghost voxels* (see Fig. 3). Finally, those track and crosstalk voxels not reconstructed after the 3D matching are discarded from the original lists. An example of the 2D to 3D reconstruction is shown in Figures 4a and 4b.

Simulation output

The resulting output from the simulation is a list of voxels and their associated energy deposits in the three planes, each with one of the following three labels that we want to classify, as described in Sec. II: track, crosstalk or ghost voxel. Using the list of voxels of each event, further features are computed for each voxel as described in Appendix B. The correlation matrices of the features for the GENIE and P-Bomb datasets are presented in Fig. 18 in Appendix C. The graph's adjacency matrix is built utilizing the position of each voxel, as will be detailed below. Both the new list of expanded voxel features plus the corresponding adjacency matrix of the event are fed into the GNN algorithm.

[i] We used the T2K flux, which peaks at 600 MeV/c, see Ref. [51].

[ii] Only signals in the first 100 ns are considered. Further delayed signals, such as decays, can be treated as independent graphs.

[iii] No waveform processing is simulated. A single conversion factor is used from energy deposit to number of photons in the WLS fiber, based on laboratory data [17].

B. Network architecture

Each graph in GraphSAGE is constructed using the proximity of two voxels in that graph. If both voxels are spatially located within a radius of 1.75 cm[iv], then we consider them to be connected in the graph by an edge; we repeat the same procedure for each pair of voxels[v]. Additionally, we consider a neighborhood depth of three, i.e., to produce the embedding of a voxel, we use the voxel features together with its first neighbors' features, the features of the neighbors of its neighbors, i.e, second neighbors' features, and the features of the neighbors of the neighbors of the neighbors of its neighbors, i.e., third neighbors' features. The aggregator used to combine the feature of the neighbors is the mean aggregator, which produces the average of the neighbors' values. This final embedding is then passed to an MLP consisting of two fully connected layers - each followed by a LeakyReLU activation function - and a final output layer followed by a softmax activation function. Figure 6 illustrates the GraphSAGE-based approach used, while Tab. II shows the architectural parameters chosen. Categorical cross-entropy is chosen as the loss function to minimize during training, as it is considered the standard one for multi-class classification problems, where each training example corresponds to a voxel:

$$J = -\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{c} y_j^{(i)} \log \hat{y}_j^{(i)}, \tag{2}$$

where:

- $\boldsymbol{y}^{(k)}$: true values corresponding to the k^{th} training example. $\boldsymbol{y}^{(k)}$ is a vector with all components equal to zero except for the class j, which is equal to one.
- $\hat{\boldsymbol{y}}^{(k)}$: predicted values corresponding to the k^{th} training example. $\hat{\boldsymbol{y}}^{(k)}$ is a vector with each component $\hat{y}_j^{(k)}$ denoting the score (continuous value from 0 to 1) of being of class j.
- m: number of training examples, equal to the total number of voxels in the training sample.
- c: number of classes/neurons corresponding to the output. In this case, the three classes are: track, crosstalk, and ghost.

The output layer of the model consists of three neurons, one for each of the three classes, with values v_i for $i = 1, 2, 3$. The sum of neuron values is given by $\sum_{i=1}^{3} v_i = 1$ such that each neuron value gives a fractional score that can be used to classify voxels. In other words, the model returns scores for each voxel to be one of the three desired outputs, which can be interpreted as the probability: track-like, crosstalk-like, or ghost-like.

Parameter	value
Encoding size	128
Depth	3
Aggregator	mean
Fully Connected Layer 1	128 neurons
Fully Connected Layer 2	128 neurons
Fully Connected Layer 3 (output)	3 neurons

TABLE II: Architectural parameters; for more information about the meaning of the parameters, see Sec. III C.

C. Training

The network was trained for 50 epochs[vi] using Python 3.6.9 and PyTorch 1.3.0 [54] as the deep-learning framework, on an NVIDIA RTX 2080 Ti GPU. Adam [55] is used as the optimizer, with a mini-batch size of 32, and an initial learning rate of 0.001 (divided by 10 when the error plateaus, as suggested in [56]). The model has a total of 105,347 parameters. As is standard in machine learning, the dataset was split into three disjoint sets: the training set, to optimize the model's parameters; the validation set, to avoid overfitting and perform model selection; the test set, to verify the integrity of the model for new data. Figure 7 shows the validation results during the training process, measured by the F_1-score metric:

$$F_1 = 2\,\frac{\text{precision}\cdot\text{recall}}{\text{precision}+\text{recall}}. \tag{3}$$

The precision and recall are defined as:

$$\text{precision} = \frac{\text{true}_{\text{positives}}}{\text{true}_{\text{positives}} + \text{false}_{\text{positives}}}, \tag{4}$$

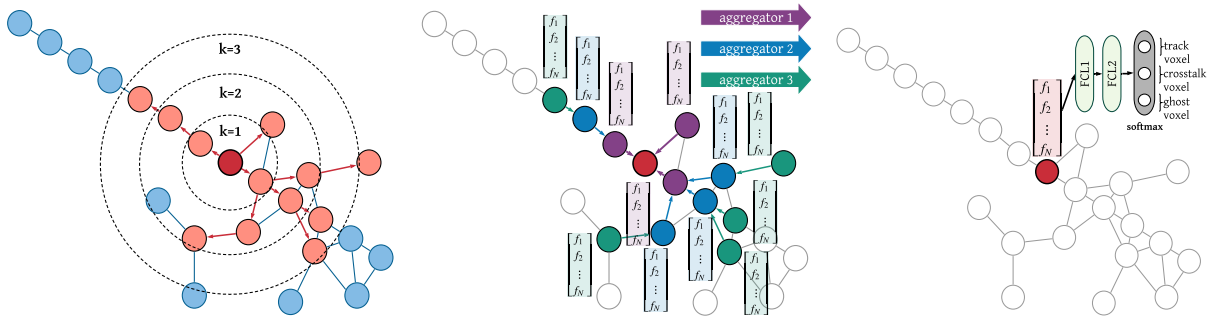
$$\text{recall} = \frac{\text{true}_{\text{positives}}}{\text{true}_{\text{positives}} + \text{true}_{\text{negatives}}}, \tag{5}$$

where the labels are compared as one class (denoted as positive) vs. all the others (denoted as negative). The model used later for inference on new data is the one that maximizes the F_1-score for the validation set, as it has the best generalization for unseen data.

[iv] To link only those voxels within the 3×3×3 cube of voxels centred on the target voxel (the maximum diagonal distance from the center of this cube is $\sqrt{1^2+1^2+1^2} \approx 1.75$).

[v] If a voxel has no neighbors, it is discarded from the graph and cannot be classified; this happens for less than 0.6% of the total number of voxels.

[vi] Epoch: one forward pass and one backward pass of all the training examples. In other words, an epoch is one pass over the entire dataset.



(a) Sample neighborhood (orange) for a depth $K = 3$. The red node indicates the target node to be classified; orange nodes are nodes taken into account for the classification given a depth of three; blue nodes are not taken into account. Each depth k represents a distance level from the target to the corresponding neighbor set.

(b) Aggregate feature information from neighbors. Each node has features $f_1, \ldots, f_N$, with purple, blue, and green nodes information being aggregated through aggregators 1, 2, and 3, respectively.

(c) Use aggregated information as input for the fully connected layers and predict the label.

FIG. 6: Visual illustration of the GraphSAGE sample and aggregate approach with a depth of three [42].

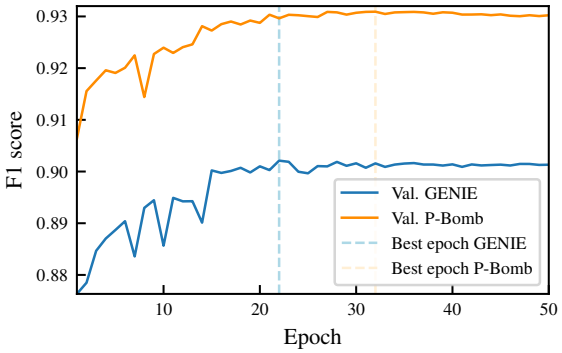


FIG. 7: Validation F1 results on GENIE and P-Bomb samples.

D. Results

The GNN voxel-type predictions are compared against the true labels to evaluate the network performance and identify possible areas of improvement. Here, we choose the output class with the highest score as the predicted class of each voxel although, depending on the type of analysis, different selection criteria could be applied in the future.

The efficiencies and purities of these predictions are calculated by two methods: per voxel and per event, the former are given by the following formulas for each type of voxels:

$$\text{efficiency}_\text{i} = \frac{\#\text{ voxels with label}_\text{true}\text{= label}_\text{pred}\text{=i}}{\#\text{ voxels with label}_\text{true}\text{=i}}, \quad (6)$$

$$\text{purity}_\text{i} = \frac{\#\text{ voxels with label}_\text{true}\text{= label}_\text{pred}\text{=i}}{\#\text{ voxels with label}_\text{pred}\text{=i}}. \quad (7)$$

The efficiencies and purities per event are defined as the mean of the efficiencies and purities of individual voxels for each event. The results of both methods for four sets of training/testing samples are shown in Tab. III, giving nearly identical performance that is independent of the dataset used to train and test the GNN.

As an example, Fig. 8 shows the voxel prediction results from the GNN when applied to the event shown in Fig. 4, a GENIE event that features a track almost completely composed of ghost voxels. Figure 8a shows the class predicted for each voxel, while Fig. 8b displays which voxels were correctly/incorrectly classified.

A more in-depth analysis of the GNN performance can be carried out by studying the effects of different event properties on the efficiencies and purities of the predictions. For these studies, the results of the GNN trained and tested on the GENIE dataset are used.

One of the factors expected to affect these predictions is the number of voxels in the event. Figure 9 shows the relationship between the mean efficiency and purity per event for each type of voxel as a function of the total number of voxels in the event. The figure also shows the mean number of events in each bin (in light blue). It is clear that both the efficiencies and purities of the three types of voxels decrease as the number of voxels in the event increases. This decrease is coupled with an increase of the fraction of ghost voxels as the total number of

		GENIE Training				P-Bomb Training			
GENIE Testing	**Per Voxel**		Track	Crosstalk	Ghost		Track	Crosstalk	Ghost
		Efficiency	93%	90%	84%	Efficiency	93%	89%	80%
		Purity	93%	87%	91%	Purity	91%	86%	89%
	Per Event		Track	Crosstalk	Ghost		Track	Crosstalk	Ghost
		Efficiency	94%	94%	88%	Efficiency	94%	93%	88%
		Purity	96%	91%	92%	Purity	95%	91%	91%
P-Bomb Testing	**Per Voxel**		Track	Crosstalk	Ghost		Track	Crosstalk	Ghost
		Efficiency	94%	93%	87%	Efficiency	95%	93%	88%
		Purity	95%	90%	92%	Purity	95%	91%	92%
	Per Event		Track	Crosstalk	Ghost		Track	Crosstalk	Ghost
		Efficiency	94%	94%	87%	Efficiency	95%	93%	88%
		Purity	96%	90%	92%	Purity	96%	91%	92%

TABLE III: Mean efficiencies and purities of voxel classification, calculated for the whole sample (per voxel) and as a mean of the event-by-event efficiencies and purities (per event).

voxels increases, which are the hardest for the GNN to classify.

The number of tracks in the event is an estimate of the complexity of its topology. According to Fig. 10, the classification efficiencies and purities drop as the number of tracks increases. This behaviour is also correlated with the increasing fraction of ghost voxels in the events.

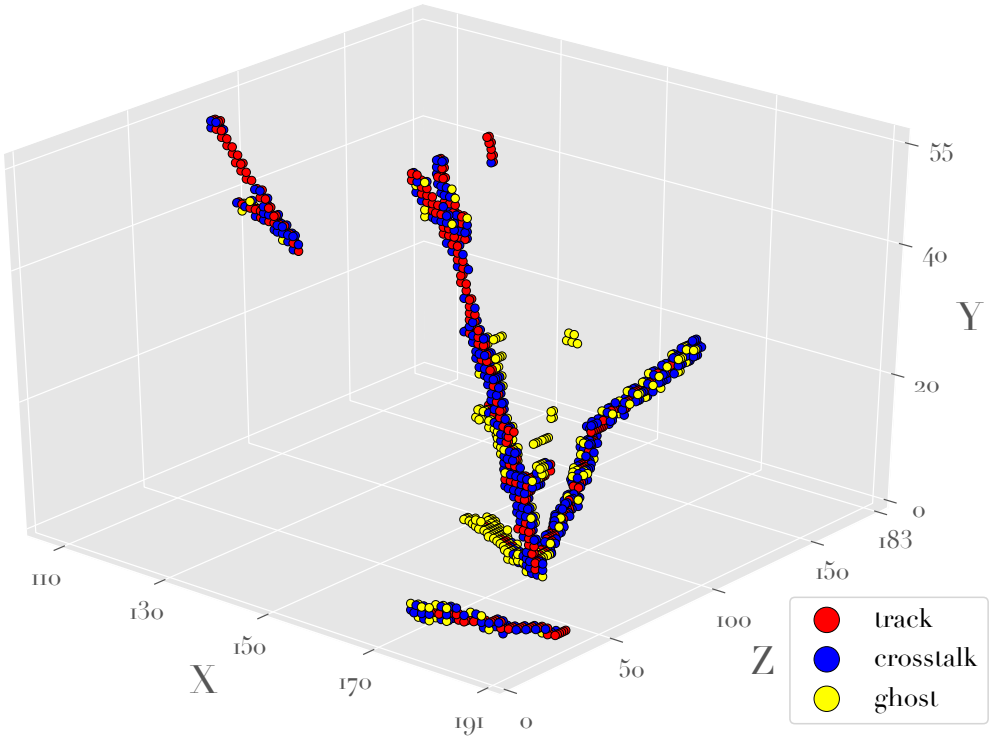
The region around the interaction vertex is of particular interest in the event. It is expected that a high spatial density of voxels within a certain volume of the detector may pose a challenge for the GNN to correctly identify the voxel type. This can be observed by studying the efficiencies and purities as a function of the distance to the interaction vertex, as shown in Fig. 11. At the interaction vertex itself, it is clear that there are only track voxels and the GNN can identify them with over 96% efficiency and 100% purity. The following 2 cm exhibit only a small fraction of ghost voxels, mainly due to the high spatial density of voxels with real signals in that volume, which is mainly occupied by track and crosstalk voxels. As we go further from the vertex, the spatial density of voxels decreases and the tracks emerging from the vertex diverge allowing for easier voxel classification. However, this trend is reversed around 10 cm from the vertex where the protons emerging from the CCQE (Charged-Current Quasi-Elastic) interaction vertex would have reached their range and only the low-ionizing muon tracks remain. The lower average voxel charge at these distances complicates the process of classification as most of the variables used as an input to the GNN are based on charge, which can be observed in the dropping efficiencies and purities.

As the main goal of this GNN is to identify ghost voxels in order to eliminate them from the events, it is important to make sure that true track and crosstalk voxels are not lost in the process. According to the GENIE sample results, only 1.1% of all true track voxels and 3.3% of crosstalk voxels are incorrectly classified as ghost voxels by the GNN. In addition, it is important not to miss ghost voxels: the GNN correctly identified 84.5% of all ghost voxels, where 72.1% of those classified incorrectly were predicted as crosstalk. Therefore, although not ideal, this issue is not critical as crosstalk voxels have a smaller influence on future studies than track voxels.

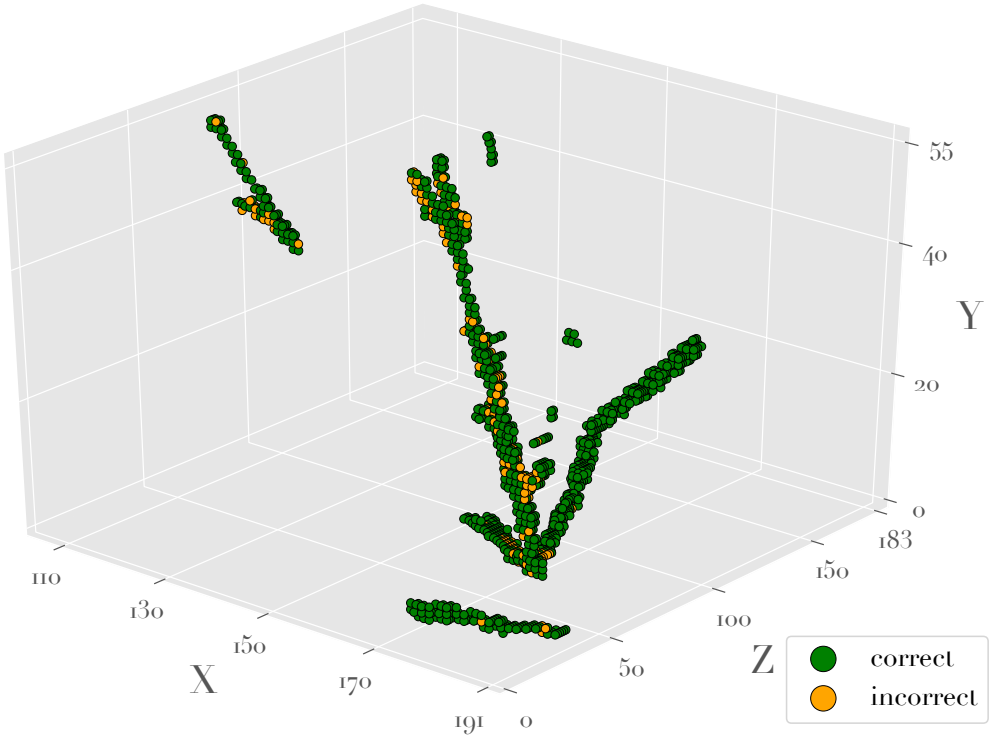
Lastly, we compare the results of the GNN against a conventional method of voxel classification which relies on a charge cut. As described in Appendix B, each voxel has three charges that correspond to the signals from the three fibers passing through it. Since other voxels along the same fiber may have signals causing a larger amplitude to be recorded, we consider the smallest of these three charges to be the most accurate estimation of the true voxel charge. Hence, this minimum charge is used for the purposes of this charge cut. Since, by definition, we expect higher energy deposition in track voxels compared to crosstalk and ghost voxels, we set a lower limit for the minimum charge in a voxel such that any voxels with a higher minimum charge than the threshold are classified as track voxels. Figure 12 shows the distribution of the minimum voxel charge for the three types of voxels. From this figure, it is clear that it is not possible to separate ghost from crosstalk voxels. Thus, this classification is only binary such that we have two categories: track or other. We decide to place this cut at 12 p.e., where the track and non-track voxel curves intersect.

To compare the results of this cut with those of our GNN, we combine the predictions of the crosstalk and ghost categories. Table IV shows the efficiency and purity of the classifications for the two methods. It is evident that using only a charge cut can still yield a comparable track voxel classification efficiency to the GNN. However, it struggles to correctly classify non-track voxels which, in turn, reduces the purity of the predicted track voxels.

Another advantage of the GNN over the charge cut is the improved capability of reducing the number of "fake" tracks, i.e. a cluster of ghost voxels that closely resembles the structure of a real particle track. Since fake tracks are usually produced by the shadowing of real tracks, the corresponding number of p.e. measured in the three readout views is higher than 12 p.e., hence the charge

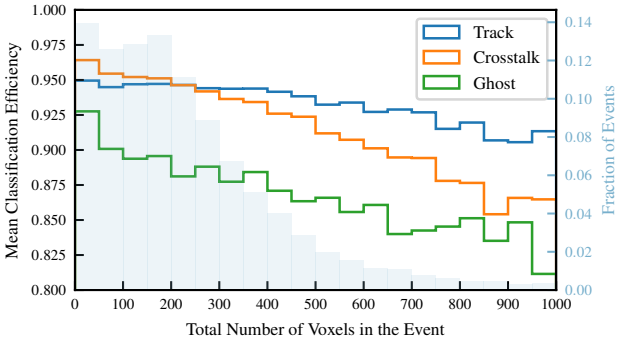


(a) Prediction: voxels are colored based on the GNN predictions.

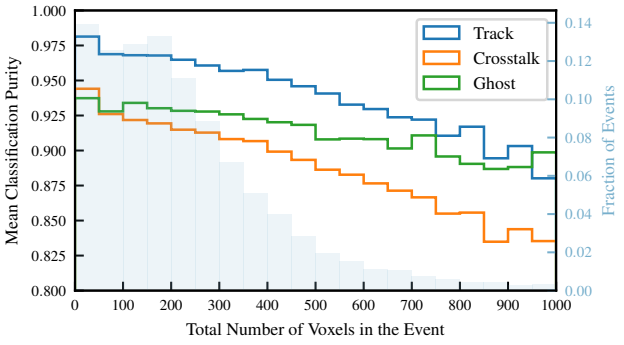


(b) Accuracy: voxels correctly classified by the GNN are shown in green.

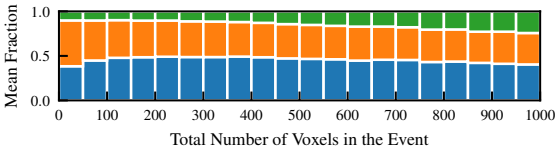
FIG. 8: Example GNN prediction results for the interaction shown in Fig. 4. The axes are in cm.



(a) Efficiency.

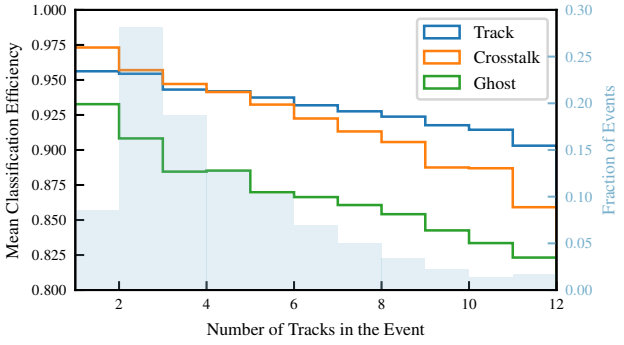


(b) Purity.

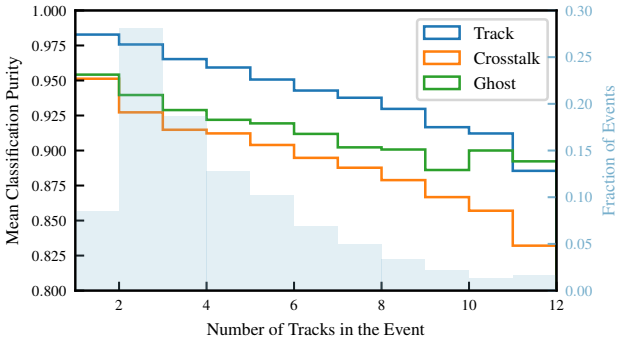


(c) Mean fraction of each type of voxel as a function of the number of voxels in the event (blue = track, orange = crosstalk, green = ghost).

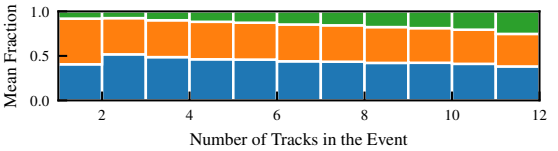
FIG. 9: Efficiency and purity as a function of the number of voxels in the event for a sample trained and tested on GENIE simulated data.



(a) Efficiency.



(b) Purity.



(c) Mean fraction of each type of voxel as a function of the number of tracks in the event (blue = track, orange = crosstalk, green = ghost).

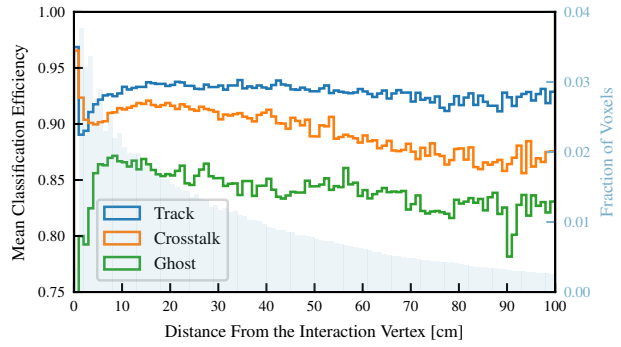
FIG. 10: Efficiency and purity as a function of the number of tracks in the event for a sample trained and tested on GENIE simulated data.

cut cannot reject them easily. The superiority of the GNN in reducing ghost tracks is shown in Appendix D for a number of neutrino interactions and compared to the charge cut method.

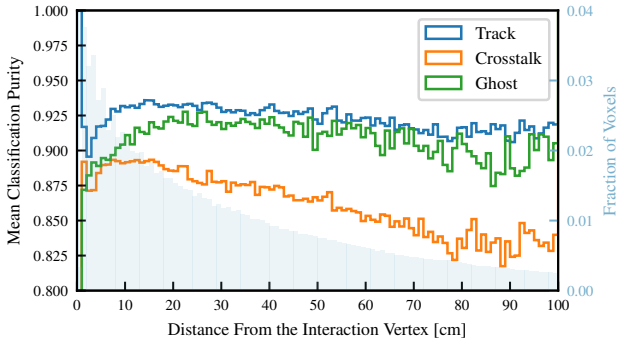
	GNN		**Charge Cut**		
	Track	Other		Track	Other
Efficiency	94%	96%	Efficiency	93%	80%
Purity	96%	95%	Purity	80%	91%

TABLE IV: Mean efficiencies and purities of voxel classification for the GNN and a simple charge cut.

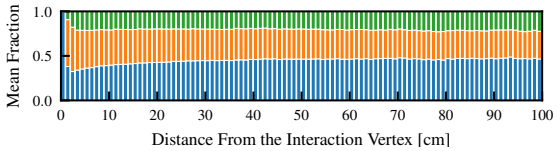
Figure 13 shows the advantage of the three-fold classification of the GNN over the binary classification of the charge cut when comparing the fraction of true total deposited energy obtained using each method. In the case of the GNN, the total deposited energy in an event is the sum of the true energy deposited in all non-ghost voxels. For the charge cut, only the energy deposited in track voxels is used. This causes an average energy loss of 5% per event when using a method that also excludes the crosstalk voxels, compared to less than 1% when using the GNN that can isolate ghost voxels.



(a) Efficiency.



(b) Purity.



(c) Mean fraction of each type of voxel as a function of the distance to the vertex (blue = track, orange = crosstalk, green = ghost).

FIG. 11: Efficiency and purity as a function of the distance to the neutrino interaction vertex for a sample trained and tested on GENIE data.

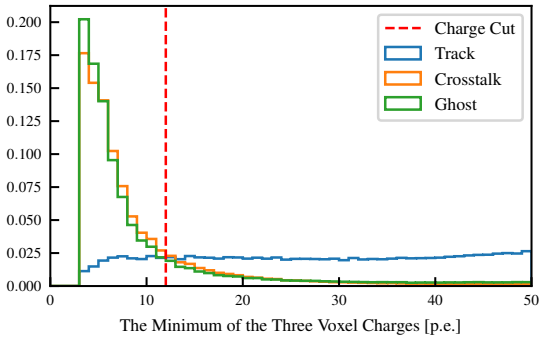


FIG. 12: The distribution of the minimum charge among the three voxel charges for the GENIE sample.

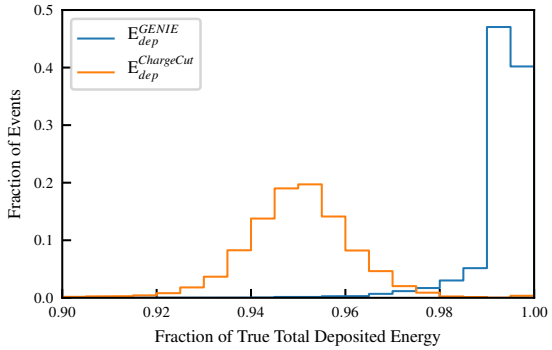


FIG. 13: The fraction of the true total deposited energy obtained when using the GNN (trained on GENIE) or the charge cut as a classification method.

V. SYSTEMATIC UNCERTAINTY CONSIDERATIONS

The results presented in Sec. IV D show that the GNN is a very powerful technique for removing ghost voxels and identifying optical crosstalk in 3D-reconstructed neutrino interactions. In this section, we investigate potential sources of systematic uncertainty and test the robustness of this technique.

One of the main limitations in the measurement of the neutrino oscillation parameters in long-baseline experiments comes from uncertainties in the modeling of neutrino interactions, not yet fully constrained by data and partially incomplete for describing all the details of the interaction final state. For example, the modeling of hadron multiplicity and kinematics may considerably change the image of the neutrino interaction, particularly near the neutrino vertex, or the total energy deposited by all the particles produced by the neutrino interaction. Hence, it is hard to obtain a data-driven control sample to train a neural network without making any prior assumptions. Since the GNN is trained only on a subset of the parameter space, the results could be biased if the detected neutrino interactions belong to a region of the parameter space not well covered by the MC generator. To account for a potentially incomplete sampling of the parameter space, different training samples (GENIE and P-Bomb) were generated, as described in Sec. IV D. The difference in terms of neutrino interaction modeling between these two datasets, by construction, is expected to be much larger than the difference between GENIE and real neutrino interactions, see Appendix A. As presented in Tab. III the performance is still very good even when the samples used for training and testing were largely different in terms of modeling, supporting the safeness of training using MC events to classify real data.

The robustness of the GNN against model dependencies can be verified by training different neural networks

on different event samples and applying them to the same set of neutrino interactions. A difference in the observables used in the physics measurement, such as particle momenta, energy deposit, etc., obtained by the different training can be assigned as a systematic uncertainty introduced by the method.

A study was performed to evaluate the impact of the method on the total true energy deposited in the detector. The difference between the total energy deposit computed after rejecting the voxels classified as ghosts for both network trainings was computed. Figure 14 shows the distribution of the total true deposited energy before and after discarding the voxels classified as ghosts. Both GENIE- and P-Bomb- trained GNNs give very similar results over the full range of total deposited energy. The total true deposited energy computed with and without ghost rejection differ on average by less than 1 MeV with a standard deviation of approximately 5.5 MeV, mainly due to a few outlier entries, and 68% of the events with a difference better than 0.192 MeV, as shown in Fig. 15. Hence, it is expected to be improved by increasing the statistics of the training samples.

This corresponds to less than 2% of the mean total deposited energy per event. In Fig. 16 the impact of the different training sample is shown as a function of the total deposited energy. The fractional standard deviation, defined as the standard deviation of the difference between deposited energy computed from different GNN trainings and divided by the true deposited energy, shown in the bottom panel, is less than 2% and almost constant as a function of the deposited energy. This means that the performance of the method is about the same irrespective of the total deposited energy. This study confirms that GNN can be used for classifying 3D voxels potentially with limited systematic uncertainties in the deposited energy, while drastically improving the tracking capability.

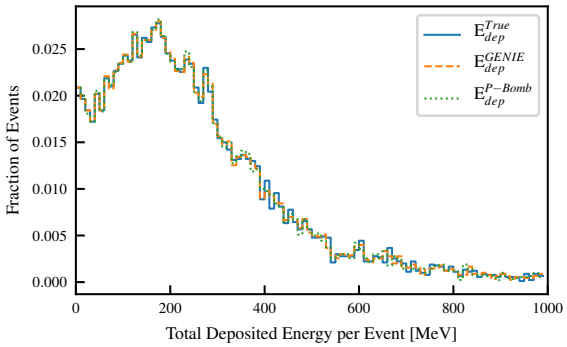


FIG. 14: Distribution of the total true deposited energy after rejecting the ghost voxels classified either with GENIE- (dashed orange) or P-Bomb- (dotted green) trained GNNs and without any ghost rejection (solid blue). The mean total deposited energy per event is about 288 MeV.

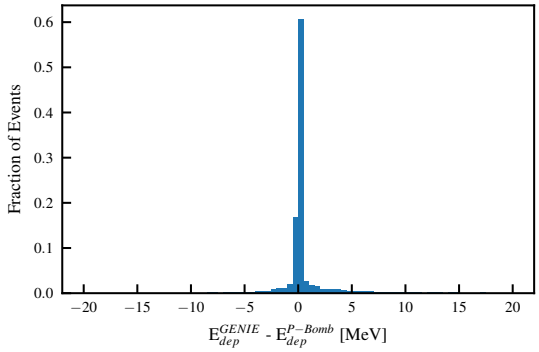


FIG. 15: Difference between the total true deposited energy computed after rejecting the ghost voxels classified with GENIE- and P-Bomb- trained GNNs. The mean is 0.78 MeV while the standard deviation is 5.5 MeV. About 40% of events show no difference between P-Bomb and GENIE, 68% have a difference within ±0.192 MeV, while only 5% of the events have a difference outside the range ±6.35 MeV.

Another potential issue could be given by a mismodeling of the amount of crosstalk. In addition to the nominal optical crosstalk (2.7%), two further datasets were simulated using 2% and 5% crosstalk and the voxel classification was performed using the GNN trained with nominal crosstalk. As shown in Tab. V, the efficiency and the purity is relatively stable even in the case where the crosstalk model is wrong, in particular for identifying track voxels. Whilst a drop in purity for track voxels is observed with 5% crosstalk, such a large mismodeling is highly unlikely given that crosstalk can be measured even with small prototypes to sub-percent precision [17]. Hence, crosstalk mismodeling is not considered to be a source of additional systematic uncertainty given that the GNN method is robust to small crosstalk variations.

Nominal Crosstalk 2.7%		Track	Crosstalk	Ghost
	Efficiency	93%	90%	84%
	Purity	92%	87%	91%
Crosstalk 2%		Track	Crosstalk	Ghost
	Efficiency	92%	89%	81%
	Purity	94%	83%	89%
Crosstalk 5%		Track	Crosstalk	Ghost
	Efficiency	94%	89%	88%
	Purity	86%	91%	93%

TABLE V: Mean efficiencies and purities of voxel classification, per voxel, for different crosstalk values, i.e. 2.7% (nominal), 2%, and 5%. The GNN was trained with GENIE training samples with nominal crosstalk and tested on the same GENIE sample with different crosstalk values to study its robustness.

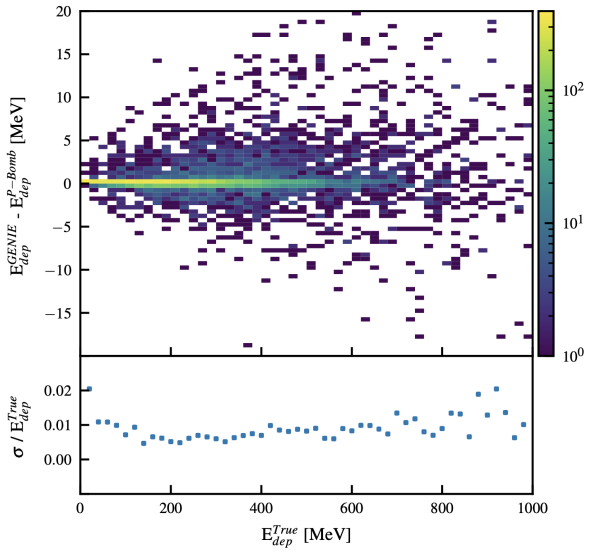


FIG. 16: Top: difference between the total true deposited energy computed after rejecting the ghost voxels classified with GENIE- and P-Bomb- trained GNNs as a function of the total true deposited energy. Bottom: fractional standard deviation of the difference of the total true deposited energy computed after rejecting the ghost voxels classified with GENIE- and P-Bomb- trained GNNs as a function of the total true deposited energy.

VI. CONCLUSIONS

A graph neural network inspired by GraphSAGE was developed and tested on simulated neutrino interactions in a 3D voxelized fine-granularity plastic-scintillator detector with three 2D readout views with the same geometry as SuperFGD, a detector that will be installed in the near detector (ND280) of T2K. The advantage of this neural network is that the graph data structures provide a natural representation of the neutrino interactions.

The neural network was able to identify ambiguities and scintillation light leakage between neighboring active scintillator detector volumes as well as real signatures left by particles with efficiencies and purities in the range of 94-96% per event, with a clear improvement with respect to less sophisticated methods. In particular, it can reduce the number of fake tracks produced by the shadowing of real tracks observed in the 2D readout views. The performance was tested for neutrino events with different number of voxels, number of tracks and voxels at different distances from the vertex, variables that could hint to interaction model dependencies of the method. Efficiencies and purities were found to be relatively stable and the trends were consistent with the expectation. The robustness of the neural network against possible systematic uncertainties introduced by the method was tested. The results were obtained using neural networks trained on different samples, produced either with the GENIE event generator or by randomizing the number of final state particles and relative momentum to obtain a more generic sample that does not belong to any particular theoretical model. It was found that the bias introduced on the total deposited energy of the event by arbitrarily choosing a different training sample is, on average, less than 1 MeV, or less than 2% for true deposited energies in the range 0.0-1.0 GeV. The impact of potential mismodeling of the light leakage between neighboring scintillator volumes was tested. Results show that the performance of the neural network is robust to expected changes in the crosstalk modeling.

To conclude, we showed that a graph neural network has great potential in assisting a 3D particle-set reconstruction of neutrino interactions. Similar results may be expected for other types of detectors that aim to a 3D reconstruction of the neutrino event from 2D projections and that share analogous features like ambiguities and leakage of signal between detector voxels, such as the very similar detector proposed as part of the DUNE near detector.

VII. ACKNOWLEDGMENTS

D. Douqa and F. Sánchez acknowledge the Swiss National Foundation Grant No. 200021_85012. C. Jesús-Valls and T. Lux acknowledge funding from the Spanish Ministerio de Economía y Competitividad (SEIDI-MINECO) under Grants No. FPA2016-77347-C2-2-P and SEV-2016-0588. S. Pina-Otey acknowledges the support of the Industrial Doctorates Plan of the Secretariat of Universities and Research of the Department of Business and Knowledge of the Generalitat of Catalonia. IFAE is partially funded by the CERCA program of the Generalitat de Catalunya. The authors also thank T. Jiang, T. Zhao, and D. Wang for their implementation of GraphSAGE[vii], on which the software of this paper was based.

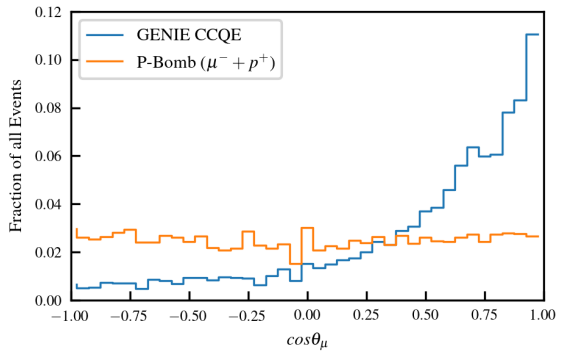
This work was initiated in the framework of the T2K Near Detector upgrade project, fruitful discussions in this context with our colleagues are gratefully acknowledged. The authors acknowledge the T2K Collaboration for providing the neutrino interaction and detector simulation software.

Appendix A: Comparison of GENIE and P-Bomb

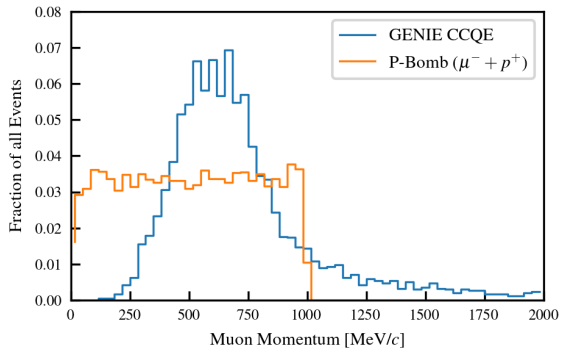
Two different types of neutrino interactions have been studied, as described in Section IV A. The neutrino modelling differences can be easily visualized by comparing two of the simplest subsets of data from each dataset. GENIE charge current quasi elastic interactions (CCQE)

[vii] https://github.com/twjiang/graphSAGE-pytorch

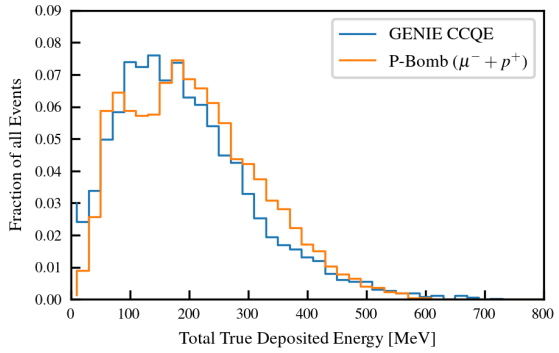
typically produce an outgoing muon and proton in the final state. For illustration, we compare this sub-sample of the GENIE dataset with the $\mu^- + p^+$ sub-sample in the P-Bomb dataset in Fig. 17.



(a) Angular distribution of muons.



(b) Momentum distribution of muons.



(c) Total energy deposit.

FIG. 17: Distributions of CCQE GENIE interactions compared to $\mu^- + p^+$ interactions in P-Bomb.

Appendix B: Input variables

The list of variables used as features for the graph nodes is given below. Each node is placed at XYZ coordinates matching the center of a cube, however, these center coordinates are not node variables by themselves since the detector response is isotropic. The numbers in front of each variable match those in Fig. 18.

- `0-2: peXY, peXZ, peYZ`
  Number of photons detected in the XY, XZ or YZ-fiber intersecting the cube under consideration corrected by the expected attenuation.

- `3-5: mXY, mXZ, mYZ`
  Number of active voxels intersected by the fiber associated to `peXY, peXZ` or `peYZ`

- `6: pewav`
  Average number of detected photons `peXY, peXZ, peYZ`, each weighted by the fiber multiplicity `mXY, mXZ, mYZ`.

$$\texttt{pewav} = \frac{\frac{\texttt{peXY}}{\texttt{mXY}} + \frac{\texttt{peXZ}}{\texttt{mXZ}} + \frac{\texttt{peYZ}}{\texttt{mYZ}}}{3}$$

- `7-9: pullX, pullY, pullZ`
  Relative difference between the light measured in two different 2D planes.

$$\texttt{pullX} = \frac{\texttt{peXY} - \texttt{peXZ}}{\texttt{peXY} + \texttt{peXZ}}$$

$$\texttt{pullY} = \frac{\texttt{peXY} - \texttt{peYZ}}{\texttt{peXY} + \texttt{peYZ}}$$

$$\texttt{pullZ} = \frac{\texttt{peXZ} - \texttt{peYZ}}{\texttt{peXZ} + \texttt{peYZ}}$$

- `10: residual`
  Similarity of the light yield measured in the three 2D planes, measured as the squared distance from each `peXY, peXZ, peYZ` to the average, weighted by the squared average.

$$\mu = \frac{\texttt{peXY} + \texttt{peXZ} + \texttt{peYZ}}{3}$$

$$\texttt{residual} = \frac{(\texttt{peXY} - \mu)^2 + (\texttt{peXZ} - \mu)^2 + (\texttt{peYZ} - \mu)^2}{\mu^2}$$

- `11: pullXYZ`
  Similarity of the light yield measured in the three 2D planes, measured as a combination of 2D pulls (a_1,a_2,a_3) weighted by `pewav`.

$$a_1 = \frac{\frac{\texttt{peXY}}{\texttt{mXY}} - \frac{\texttt{peXZ}}{\texttt{mXZ}}}{\frac{\texttt{peXY}}{\texttt{mXY}} + \frac{\texttt{peXZ}}{\texttt{mXZ}}}$$

$$a_2 = \frac{\frac{\texttt{peXY}}{\texttt{mXY}} - \frac{\texttt{peYZ}}{\texttt{mYZ}}}{\frac{\texttt{peXY}}{\texttt{mXY}} + \frac{\texttt{peYZ}}{\texttt{mYZ}}}$$

$$a_3 = \frac{\frac{\texttt{peXZ}}{\texttt{mXZ}} - \frac{\texttt{peYZ}}{\texttt{mYZ}}}{\frac{\texttt{peXZ}}{\texttt{mXZ}} + \frac{\texttt{peYZ}}{\texttt{mYZ}}}$$

$$\texttt{pullXYZ} = \frac{a_1a_2 + a_1a_3 + a_2a_3}{\texttt{pewav}}$$

- 12: ratioMQ
  Ratio between the average voxel multiplicity in the three fibers and pewav.

$$\texttt{ratioMQ} = \frac{\frac{\texttt{mXY+mXZ+mYZ}}{3}}{\texttt{pewav}}$$

- 13-14: R1, R3
  Number of active neighbor voxels in a sphere of certain radius.
  ↪ R1, r=1 cm.
  ↪ R2, r=2 cm.
  ↪ R3, r=5 cm. R2 was not used as a variable due to the high correlation with R1, but is used to compute RR.

- 15-20: x+, x-, y+, y-, z+, z-
  Boolean variables representing the existence of immediate neighbors in each of the 6 surrounding cubes

- 21: orthogonal_neighbor
  It is 1 if any of x+, x-, y+, y-, z+, z- is 1.

- 22: RR
  Ratio between the number of close and far voxels. The $\epsilon = 10^{-7}$ prevents numerical problems when R3=0.

$$\texttt{RR} = \frac{\texttt{R2}}{\texttt{R3} + \epsilon}$$

- 23: ratioDQ
  Ratio between the average voxel distance aveDist around the voxel and the weighted average light yield pewav.

$$\texttt{ratioDQ} = \frac{\texttt{aveDist}}{\texttt{pewav}}$$

- 24: aveDist
  Average distance from the voxel center C to all fired voxel centers (C_i) within a sphere of radius 2.5 cm.

$$\texttt{aveDist} = \frac{1}{N}\sum_{i}^{N} \text{EuclidianDist}(C, C_i)$$

A number of these variables are calculated from the same underlying properties of the energy deposits. In theory, an infinitely deep GNN trained on an infinite amount of training data would be able to extract all of the information required for classification from the few underlying properties. In practice, we use a larger number of derived variables to guide the GNN to allow it to more easily extract information from the data and to converge quickly in the training process. Global position was intentionally not used as a variable to avoid the GNN to learn neutrino modelling specific behaviours.

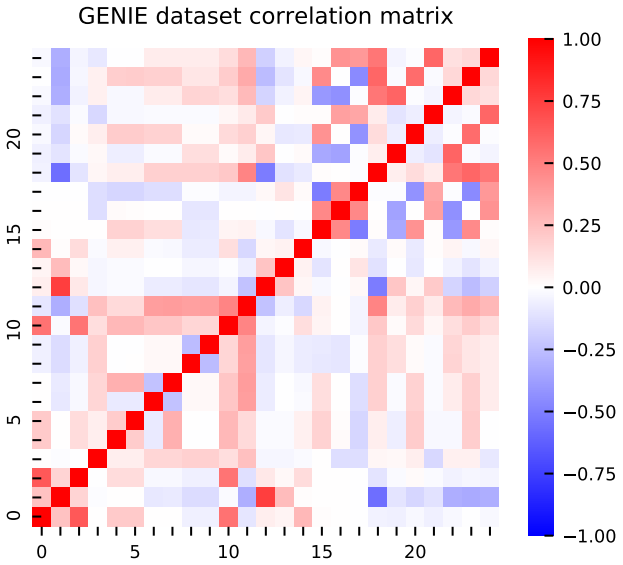
Appendix C: Comparison of GENIE and P-Bomb simulated data samples

Figure 18 shows the correlations of the input variables defined in Appendix B for the GENIE and P-Bomb data samples. Differences between the two matrices arise from the different topologies of interactions produced by the two generator methods.

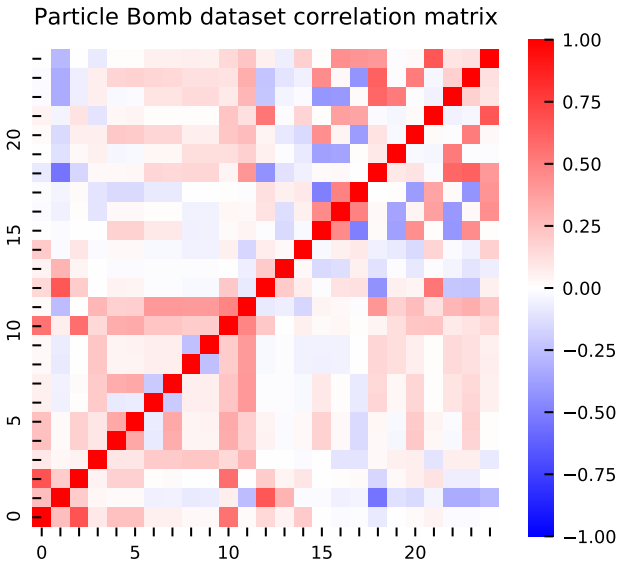
Appendix D: Event Gallery

This section contains a number of visualizations to show the classification performance of the GNN for a number of neutrino interactions with different complexity and topology. Displays are shown for different events in Figs 19 - 24: all voxels with their true classification, only the true track voxels, the classified track voxels using the charge cut method, and the classified track voxels using the GNN. The interactions shown here are examples of interactions containing many ghost voxels in order to showcase the GNN performance.

The track voxel classification ability of the charge cut and GNN methods can be seen by comparing subfigures (c) and (d) with (b), respectively, for each interaction. The GNN is able to reject ghost voxels very well, as shown in Figs 20, 22, 23 and 24 where ghost tracks remain using the charge cut method. In general, the performance improvement from the GNN increases with the complexity of the interactions. For simple interactions with only a single muon in the final state both methods perform similarly.

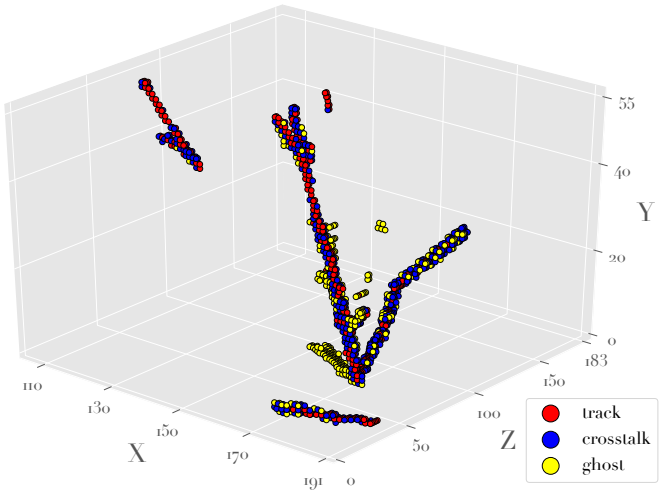


(a) GENIE dataset correlation matrix.

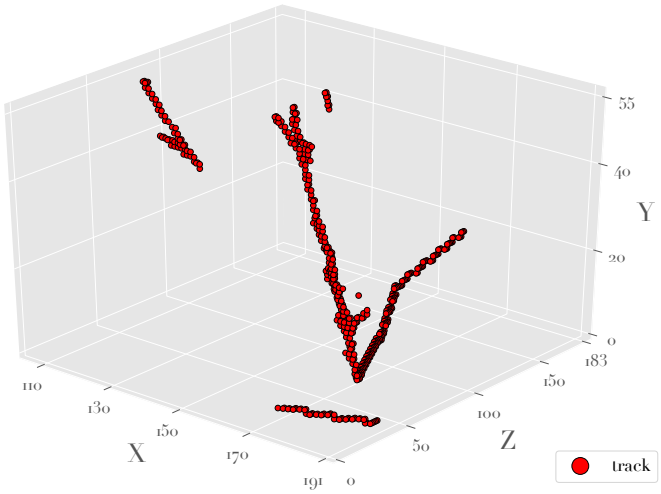


(b) P-Bomb dataset correlation matrix.

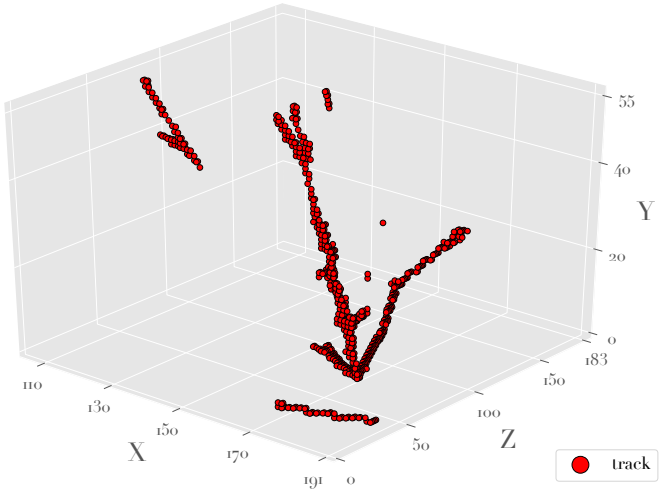
FIG. 18: Correlation matrices for the input variables of the GENIE and P-Bomb datasets used. Appendix B gives the mapping between the numbers on the axes and the variable names.



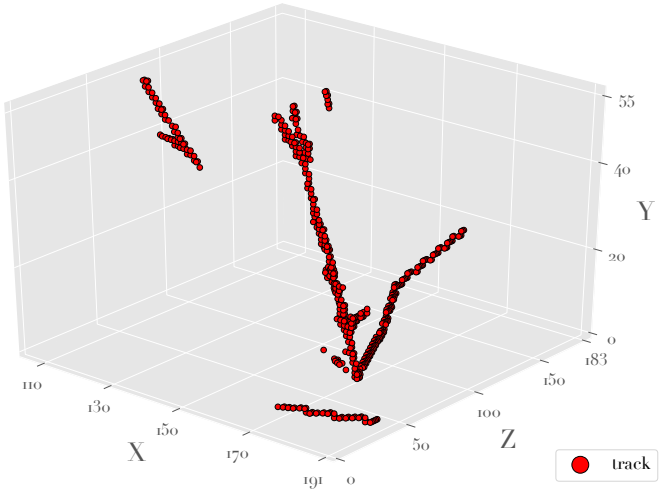
(a) The 3D voxels labelled as track (red), crosstalk (blue) and ghost (yellow) according to the truth information from the simulation are shown.



(b) Only the 3D voxels labelled as track according to the truth information from the simulation are shown.

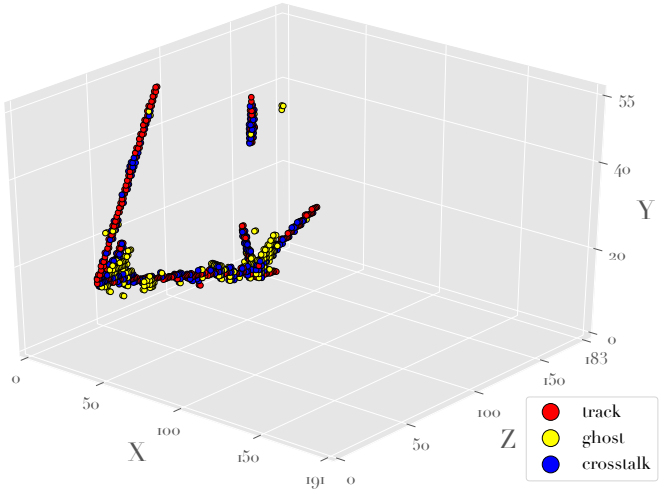


(c) The 3D voxels labelled as track according to the charge cut classification are shown.

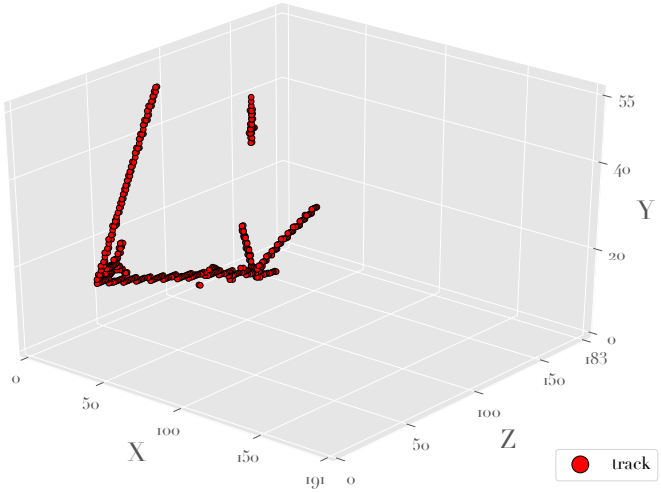


(d) The 3D voxels labelled as track according to the GNN classification are shown.

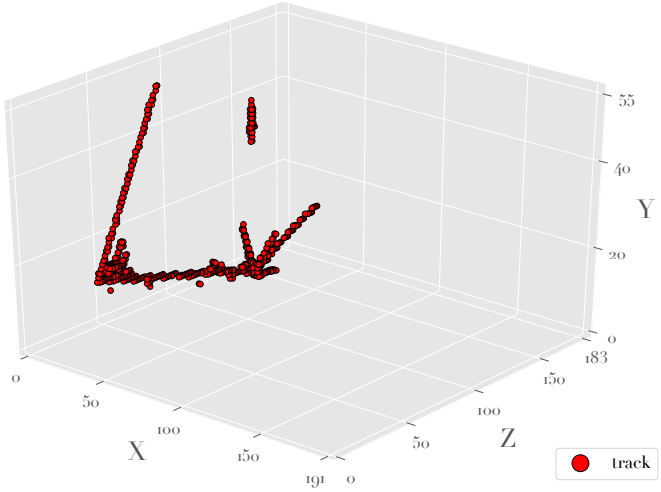
FIG. 19: 3D visualization of a neutrino interaction in a finely segmented 3D scintillator detector after the 3D matching of the three 2D views. The GNN cut is able to almost entirely reject the fake track traveling on the XZ plane and stopping near to the vertex at X∼160 cm and Z∼70 cm, while the charge cut cannot. The energy of the incoming neutrino is 4.754 GeV. The axes are in cm.



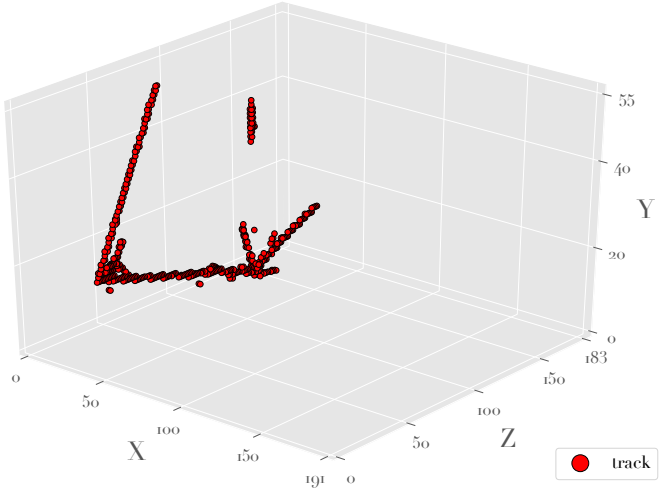
(a) The 3D voxels labelled as track (red), crosstalk (blue) and ghost (yellow) according to the truth information from the simulation are shown.



(b) Only the 3D voxels labelled as track according to the truth information from the simulation are shown.

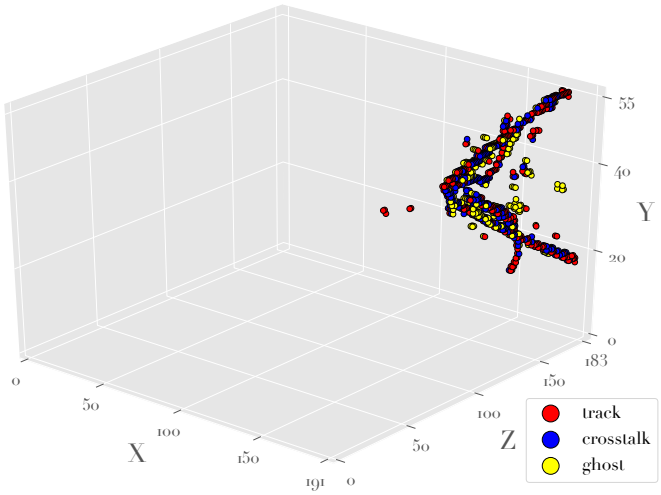


(c) The 3D voxels labelled as track according to the charge cut classification are shown.

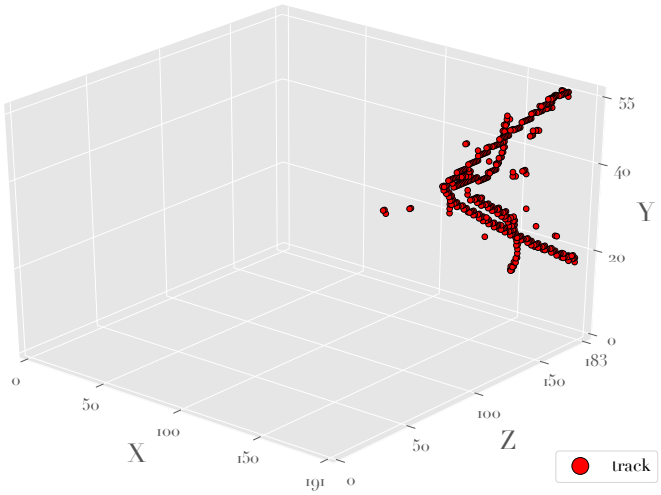


(d) The 3D voxels labelled as track according to the GNN classification are shown.

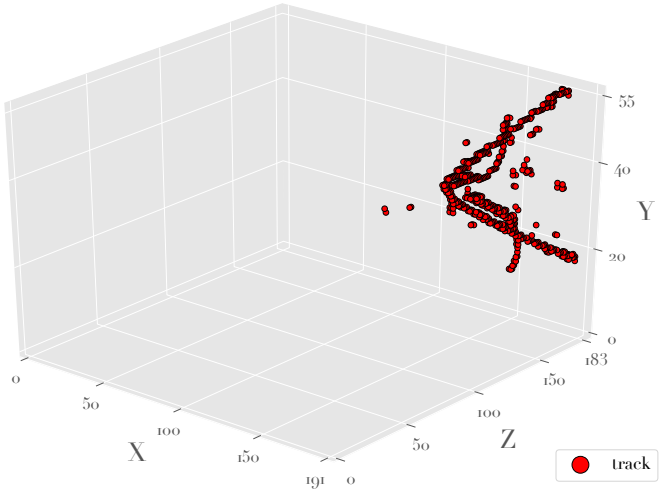
FIG. 20: 3D visualization of a neutrino interaction in a finely segmented 3D scintillator detector after the 3D matching of the three 2D views. The charge cut is not able to reject two fake tracks, one coming from a vertex a X<50 cm Z<50 cm traveling on the XZ plane and stopping near to the vertex at X∼160 cm and Z∼70 cm,. Moreover, the charge cut leave a bump of ghost voxels around the vertex that could mimic the interaction of a few low-energy protons, an effect that could bias the reconstruction of the neutrino energy. The energy of the incoming neutrino is 760 MeV. The axes are in cm.



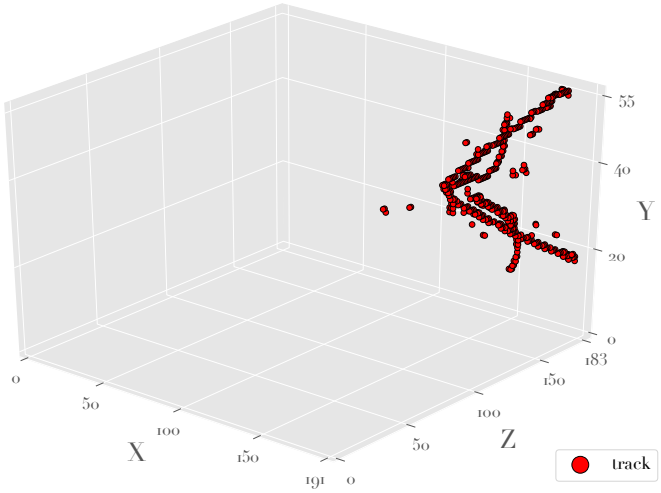
(a) The 3D voxels labelled as track (red), crosstalk (blue) and ghost (yellow) according to the truth information from the simulation are shown.



(b) Only the 3D voxels labelled as track according to the truth information from the simulation are shown.

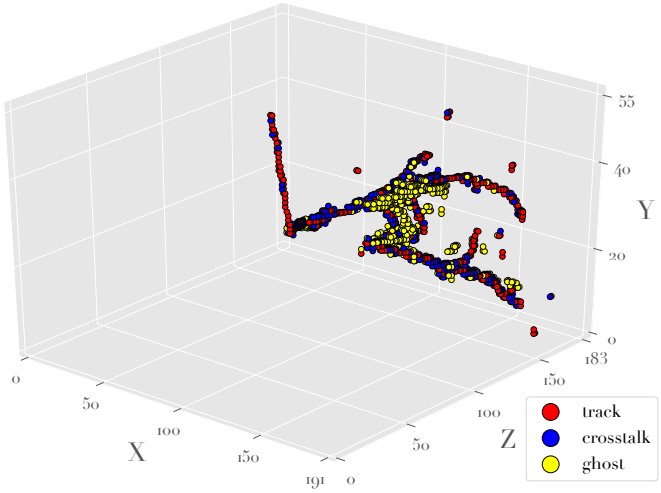


(c) The 3D voxels labelled as track according to the charge cut classification are shown.

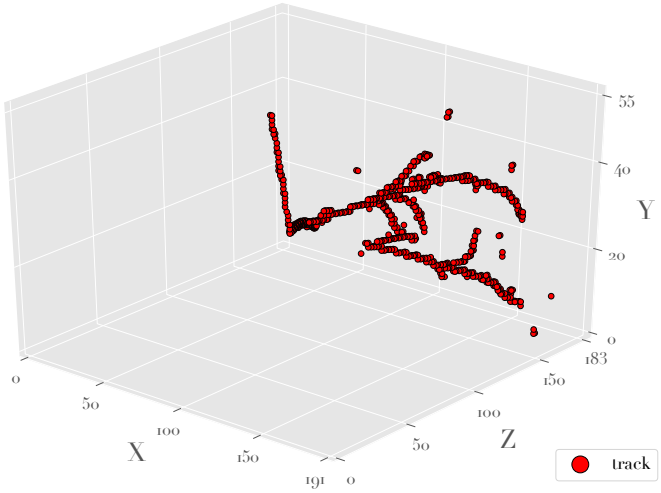


(d) The 3D voxels labelled as track according to the GNN classification are shown.

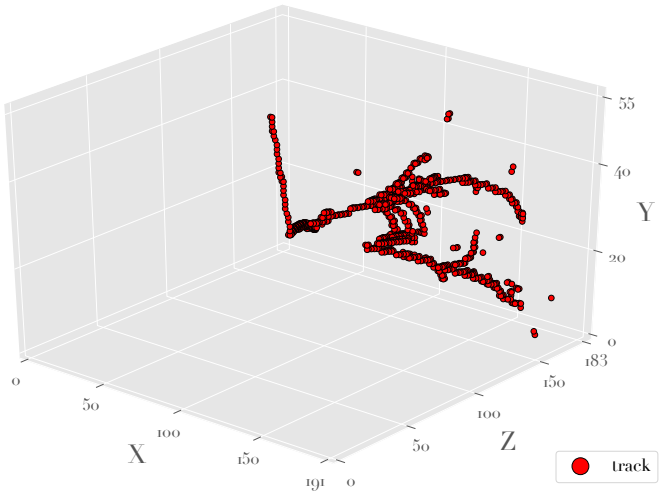
FIG. 21: 3D visualization of a neutrino interaction in a finely segmented 3D scintillator detector after the 3D matching of the three 2D views. In this even the performance of GNN and the charge cut is quite similar because the ghost voxels are mainly given by the overlap of crosstalk hits in the 2D readout views. The energy of the incoming neutrino is 5.076 GeV. The axes are in cm.



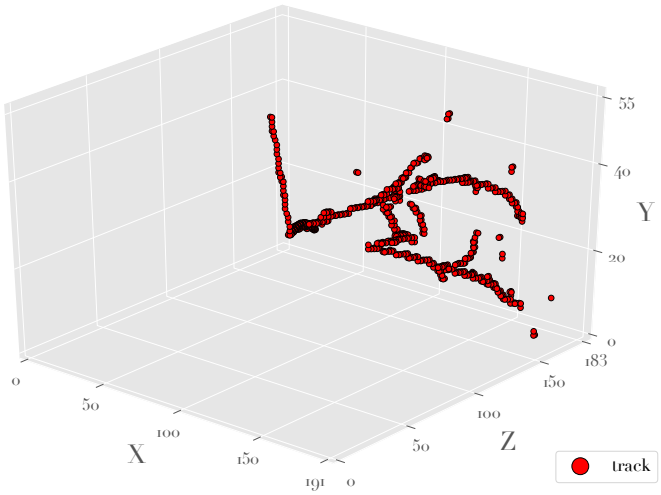
(a) The 3D voxels labelled as track (red), crosstalk (blue) and ghost (yellow) according to the truth information from the simulation are shown.



(b) Only the 3D voxels labelled as track according to the truth information from the simulation are shown.

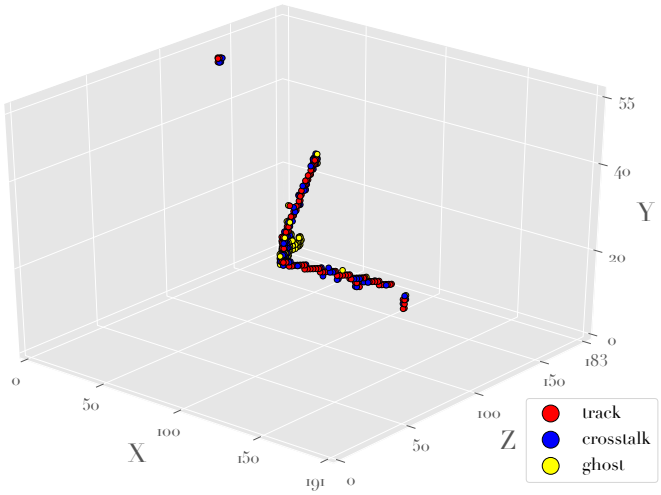


(c) The 3D voxels labelled as track according to the charge cut classification are shown.

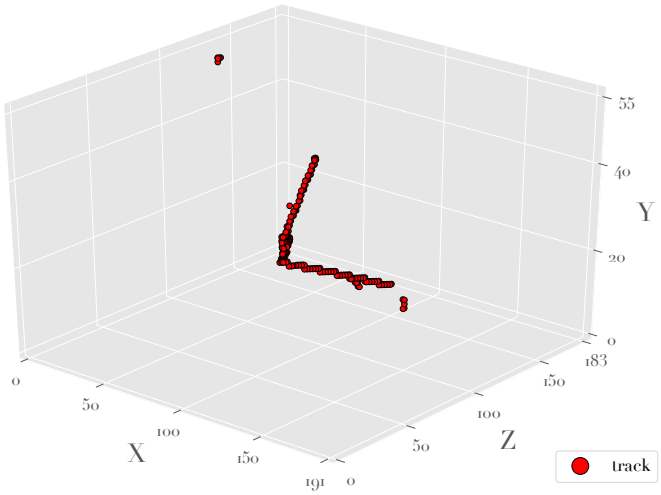


(d) The 3D voxels labelled as track according to the GNN classification are shown.

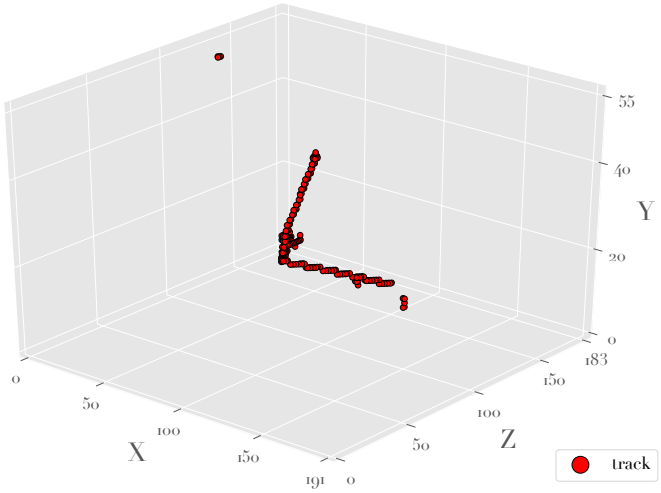
FIG. 22: 3D visualization of a neutrino interaction in a finely segmented 3D scintillator detector after the 3D matching of the three 2D views. This neutrino event has a quite high multiplicity and tracks are quite close each other. This produce relatively big clusters of ghost voxels that produce at least two fake tracks even after the charge cut. Instead GNN allows to classify ghosts more precisely and correctly visualize the correct number of tracks. Moreover, the charge cut makes true tracks more fat making their separation harder and, potentially, less precise the particle momentum reconstruction. The energy of the incoming neutrino is 1.064 GeV. The axes are in cm.



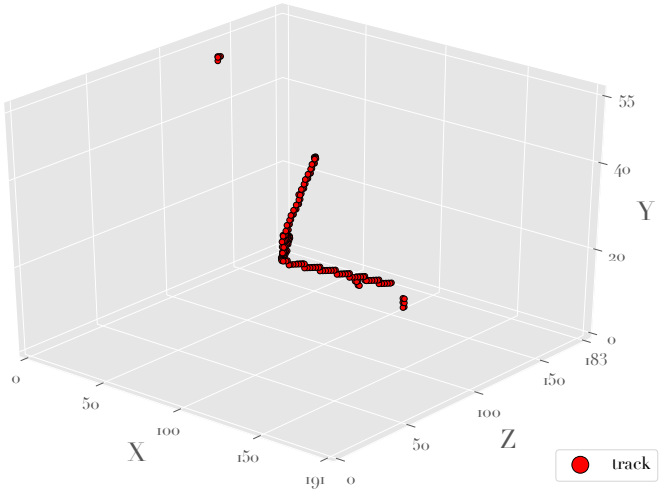
(a) The 3D voxels labelled as track (red), crosstalk (blue) and ghost (yellow) according to the truth information from the simulation are shown.



(b) Only the 3D voxels labelled as track according to the truth information from the simulation are shown.

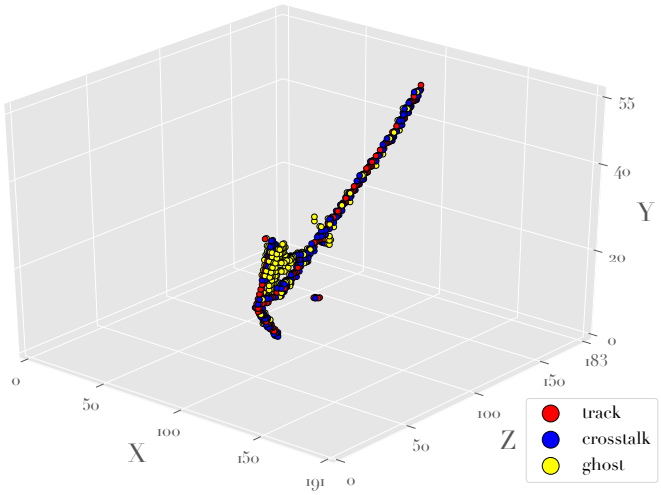


(c) The 3D voxels labelled as track according to the charge cut classification are shown.

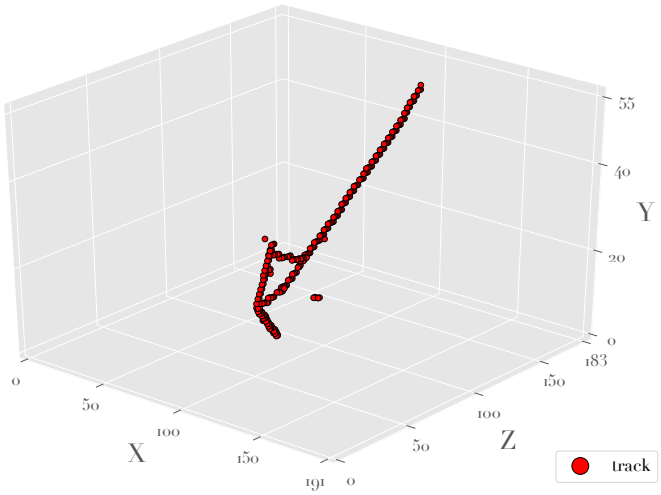


(d) The 3D voxels labelled as track according to the GNN classification are shown.

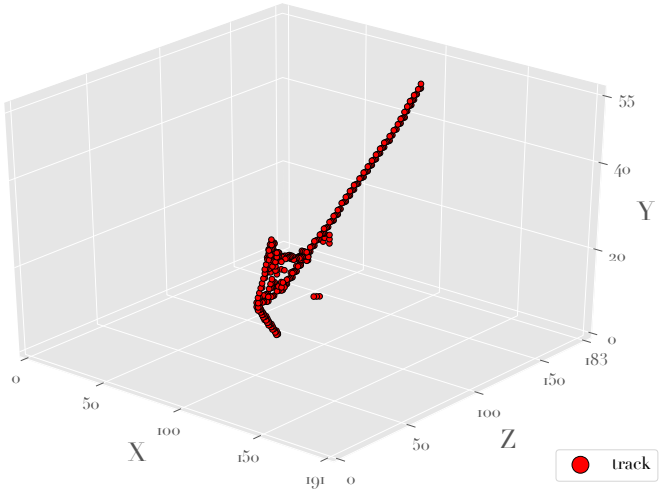
FIG. 23: 3D visualization of a neutrino interaction in a finely segmented 3D scintillator detector after the 3D matching of the three 2D views. Although this is a relatively simple neutrino event, the charge cut is not able to reject a fake track stopping near the neutrino interaction vertex while GNN can provide a much cleaner reconstruction. The energy of the incoming neutrino is 1.132 GeV. The axes are in cm.



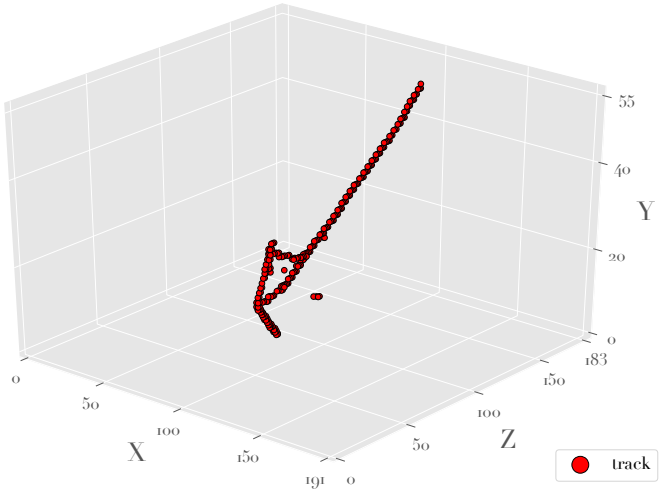
(a) The 3D voxels labelled as track (red), crosstalk (blue) and ghost (yellow) according to the truth information from the simulation are shown.



(b) Only the 3D voxels labelled as track according to the truth information from the simulation are shown.



(c) The 3D voxels labelled as track according to the charge cut classification are shown.



(d) The 3D voxels labelled as track according to the GNN classification are shown.

FIG. 24: 3D visualization of a neutrino interaction in a finely segmented 3D scintillator detector after the 3D matching of the three 2D views. In the neutrino event GNN can easily reject the relatively big cluster of ghost voxels that would make difficult a proper reconstruction of the number of tracks and corresponding energy, in particular near to the interaction vertex. The energy of the incoming neutrino is 1.897 GeV. The axes are in cm.

[1] Y. Fukuda *et al.* (Super-Kamiokande), Phys. Rev. Lett. **81**, 1562 (1998), arXiv:hep-ex/9807003 [hep-ex].
[2] B. Aharmim *et al.* (SNO), Phys. Rev. C **72**, 055502 (2005), arXiv:nucl-ex/0502021 [nucl-ex].
[3] T. Araki *et al.* (KamLAND), Phys. Rev. Lett. **94**, 081801 (2005), arXiv:hep-ex/0406035 [hep-ex].
[4] M. H. Ahn *et al.* (K2K), Phys. Rev. D **74**, 072003 (2006), arXiv:hep-ex/0606032 [hep-ex].
[5] F. P. An *et al.* (Daya Bay), Phys. Rev. Lett. **108**, 171803 (2012), arXiv:1203.1669 [hep-ex].
[6] P. Adamson *et al.* (MINOS), Phys. Rev. Lett. **112**, 191801 (2014), arXiv:1403.0867.
[7] M. Acero *et al.* (NOvA), Phys. Rev. Lett. **123**, 151803 (2019), arXiv:1906.04907 [hep-ex].
[8] K. Abe *et al.* (T2K), Nature **580**, 339 (2020), arXiv:1910.03887 [hep-ex].
[9] M. A. Acero *et al.* (NOvA), Phys. Rev. **D98**, 032012 (2018), arXiv:1806.00096 [hep-ex].
[10] B. Abi *et al.* (DUNE), arXiv:2002.03005 [hep-ex] (2020).
[11] M. Tanabashi *et al.* (Particle Data Group), Phys. Rev. D **98**, 030001 (2018).
[12] C. Rubbia (1977).
[13] A. Blondel, F. Cadoux, S. Fedotov, M. Khabibullin, A. Khotjantsev, A. Korzenev, A. Kostin, Y. Kudenko, A. Longhin, A. Mefodiev, P. Mermod, O. Mineev, E. Noah, D. Sgalaberna, A. Smirnov, and N. Yershov, Journal of Instrumentation **13** (02), P02006.
[14] K. Abe *et al.*, T2K ND280 Upgrade - Technical Design Report (2019), arXiv:1901.03750 [physics.ins-det].
[15] G. Yang (DUNE), PoS **ICHEP2018**, 868 (2019).
[16] A. Sperduti and A. Starita, IEEE Transactions on Neural Networks , 714 (1997).
[17] A. Blondel *et al.*, The SuperFGD Prototype Charged Particle Beam Tests (2020), arXiv:2008.08861 [physics.ins-det].
[18] O. Mineev *et al.*, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **923**, 134 (2019).
[19] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, Neural Computation **1**, 541 (1989).
[20] A. Aurisano *et al.*, Journal of Instrumentation **11** (09), P09001.
[21] R. Acciarri *et al.* (MicroBooNE), JINST **12** (03), P03011, arXiv:1611.05531 [physics.ins-det].
[22] B. Abi, R. Acciarri, M. Acero, G. Adamov, D. Adams, M. Adinolfi, Z. Ahmad, J. Ahmed, T. Alion, S. Alonso Monsalve, and et al., Physical Review D **102**, 10.1103/physrevd.102.092003 (2020).
[23] C. Adams *et al.* (NEXT), arXiv:2005.06467 [physics.ins-det] (2020).
[24] M. Kronmueller and T. Glauch (IceCube), PoS **ICRC2019**, 937 (2020), arXiv:1908.08763 [astro-ph.IM].
[25] Z. Li *et al.* (nEXO), JINST **14** (09), P09020, arXiv:1907.07512 [physics.ins-det].
[26] B. Abi *et al.* (DUNE), arXiv:1706.07081 [physics.ins-det] (2017).
[27] C. Adams *et al.* (MicroBooNE Collaboration), Phys. Rev. D **99**, 092001 (2019).
[28] P. Baldi, J. Bian, L. Hertel, and L. Li, Phys. Rev. D **99**, 012011 (2019).
[29] I. Seong, L. Hertel, J. Collado, L. Li, N. Nayak, J. Bian, and P. Baldi (2019).
[30] M. Huennefeld (IceCube), EPJ Web Conf. **207**, 05005 (2019).
[31] B. Graham and L. van der Maaten, Submanifold sparse convolutional networks (2017), arXiv:1706.01307 [cs.NE].
[32] B. Graham, M. Engelcke, and L. van der Maaten, 3d semantic segmentation with submanifold sparse convolutional networks (2017), arXiv:1711.10275 [cs.CV].
[33] MicroBooNE Collaboration (2020).
[34] M. Kekic *et al.* (NEXT), Demonstration of background rejection using deep convolutional neural networks in the next experiment (2020), arXiv:2009.10783 [physics.ins-det].
[35] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, Graph neural networks: A review of methods and applications (2018), arXiv:1812.08434 [cs.LG].
[36] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, in *Advances in Neural Information Processing Systems 28*, edited by C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Curran Associates, Inc., 2015) pp. 2224–2232.
[37] Y. Shen, H. Li, S. Yi, D. Chen, and X. Wang, in *The European Conference on Computer Vision (ECCV)* (2018).
[38] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, Dynamic Graph CNN for Learning on Point Clouds (2018), arXiv:1801.07829 [cs.CV].
[39] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec, in *Advances in Neural Information Processing Systems 31*, edited by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Curran Associates, Inc., 2018) pp. 4800–4810.
[40] B. Perozzi, R. Al-Rfou, and S. Skiena, in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14 (Association for Computing Machinery, New York, NY, USA, 2014) p. 701–710.
[41] T. N. Kipf and M. Welling, Semi-Supervised Classification with Graph Convolutional Networks (2016), arXiv:1609.02907 [cs.LG].
[42] W. L. Hamilton, R. Ying, and J. Leskovec, Inductive Representation Learning on Large Graphs (2017), arXiv:1706.02216 [cs.SI].
[43] N. Choma, F. Monti, L. Gerhardt, T. Palczewski, Z. Ronaghi, M. Prabhat, W. Bhimji, M. Bronstein, S. Klein, and J. Bruna (2018) pp. 386–391.
[44] F. Drielsma, Q. Lin, P. C. de Soux, L. Dominé, R. Itay, D. H. Koh, B. J. Nelson, K. Terao, K. V. Tsang, and T. L. Usher, Clustering of electromagnetic showers and particle interactions with graph neural networks in liquid argon time projection chambers data (2020), arXiv:2007.01335 [physics.ins-det].
[45] S. Farrell, P. Calafiura, M. Mudigonda, Prabhat, D. Anderson, J.-R. Vlimant, S. Zheng, J. Bendavid, M. Spiropulu, G. Cerati, L. Gray, J. Kowalkowski, P. Spentzouris, and A. Tsaris, Novel deep learning methods for track reconstruction (2018), arXiv:1810.06111 [hep-ex].
[46] S. R. Qasim, J. Kieseler, Y. Iiyama, and M. Pierini, The

European Physical Journal C **79**, 10.1140/epjc/s10052-019-7113-9 (2019).

[47] X. Ju, S. Farrell, P. Calafiura, D. Murnane, Prabhat, L. Gray, T. Klijnsma, K. Pedro, G. Cerati, J. Kowalkowski, G. Perdue, P. Spentzouris, N. Tran, J.-R. Vlimant, A. Zlokapa, J. Pata, M. Spiropulu, S. An, A. Aurisano, J. Hewes, A. Tsaris, K. Terao, and T. Usher, Graph Neural Networks for Particle Reconstruction in High Energy Physics detectors (2020), arXiv:2003.11603 [physics.ins-det].

[48] S. Hochreiter and J. Schmidhuber, Neural Computation **9**, 1735 (1997), https://doi.org/10.1162/neco.1997.9.8.1735.

[49] F. Rosenblatt, Cornell Aeronautical Laboratory (1957), report 85-460-1.

[50] C. Andreopoulos *et al.*, Nucl. Instrum. Meth. A **614**, 87 (2010), arXiv:0905.2517 [hep-ph].

[51] K. Abe *et al.* (T2K), Phys. Rev. **D87**, 012001 (2013), [Addendum: Phys. Rev.D87,no.1,019902(2013)], arXiv:1211.0469 [hep-ex].

[52] S. Agostinelli *et al.* (GEANT4), Nucl. Instrum. Meth. **A506**, 250 (2003).

[53] J. B. Birks, Proceedings of the Physical Society. Section A **64**, 874 (1951).

[54] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, in *Advances in Neural Information Processing Systems 32* (Curran Associates, Inc., 2019) pp. 8024–8035.

[55] D. P. Kingma and J. Ba, CoRR **abs/1412.6980** (2014), arXiv:1412.6980.

[56] K. He, X. Zhang, S. Ren, and J. Sun, CoRR **abs/1512.03385** (2015), arXiv:1512.03385.