Real-time Anomaly Detection on Liquid Argon Time Projection Chamber Wire Data

Seokju Chung, Jack Cleeve, Columbia University

June 18, 2025
AD4HEP Workshop

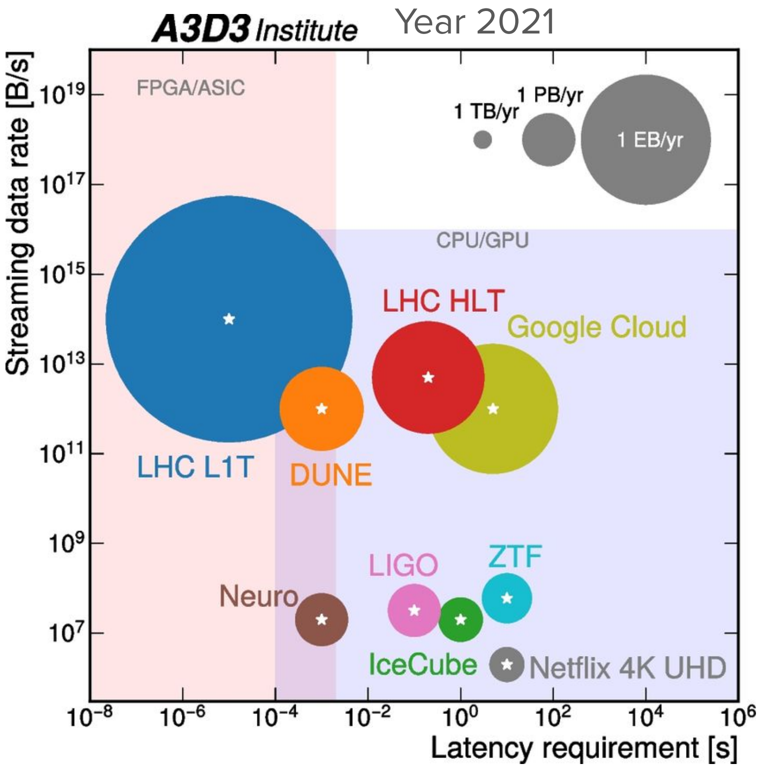
Introduction

- Real-Time Anomaly Detection: NSF-funded collaborative project between Columbia and Princeton Universities
  - Columbia (Neutrino) - G. Karagiorgi (PI), S. Chung, J. Cleeve, A. Malige (Now at BNL)
  - Princeton (Collider) - I. Ojalvo (PI), L. Gerlach, A. Ji, A. Pol (Now at Thomson Reuters Lab)
- Apply CMS CICADA network for anomaly detection in LArTPC raw data
  - Chung, et al., "Neural Network with Knowledge Distillation for Anomaly Detection in Liquid Argon Time Projection Chambers", *in preparation, to be submitted to JINST*
- Fast Inference for Rare Events based on Features in Liquid-argon ionization imagerY (FIREFLY)

FIREFLY

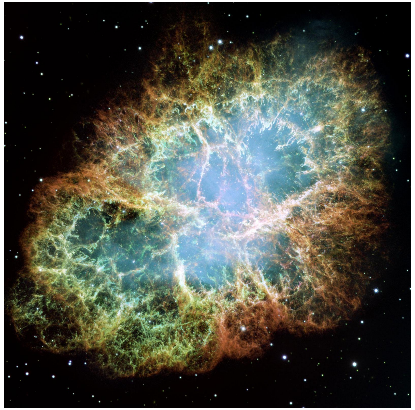
Why Real-time Triggering?

- Modern particle experiments generate large amount of data
- Impossible to save all; store in temporary buffer
- Need selection (trigger) which decides whether to keep buffer data or not



Why Trigger on Anomalies? - Data-driven Trigger

- Experiments utilize different triggers for different physics signals of interest
  - Short Baseline Neutrino (SBN) Program
    - Neutrinos from beam - External beam + coincidence light trigger
    - Supernova neutrinos - External SNEWS trigger from telescopes, delay ~ minutes
- Larger experiments (e.g. DUNE) will be generating much larger data rates
- Cannot afford buffering the data for long, requiring them to have a data-driven trigger



Why Trigger on Anomalies? - Anomaly Trigger

- Trigger needs to be designed based on expected particle signature
  ➡ could be model dependent, signature for new physics is unknown
- Model-independent; learning from data
- Anomaly trigger already being used in some CMS triggers

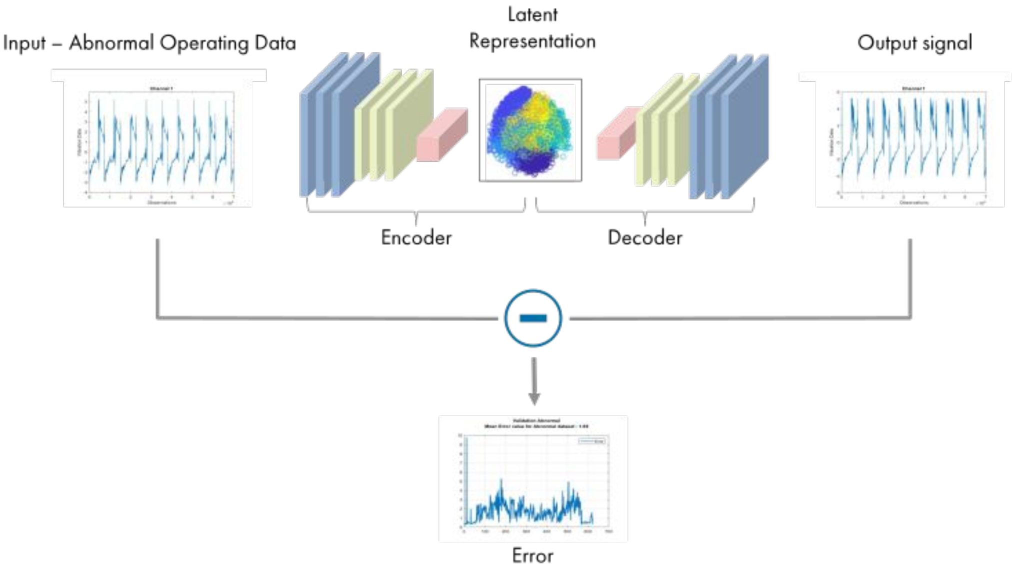
https://github.com/AdrianAlan/L1CaloTriggerAD
https://cds.cern.ch/record/2879816?ln=en

During this workshop:

- *CMS AXOL1TL Trigger*, Melissa Quinnan
- *CMS CICADA Trigger*, Kiley Kennedy

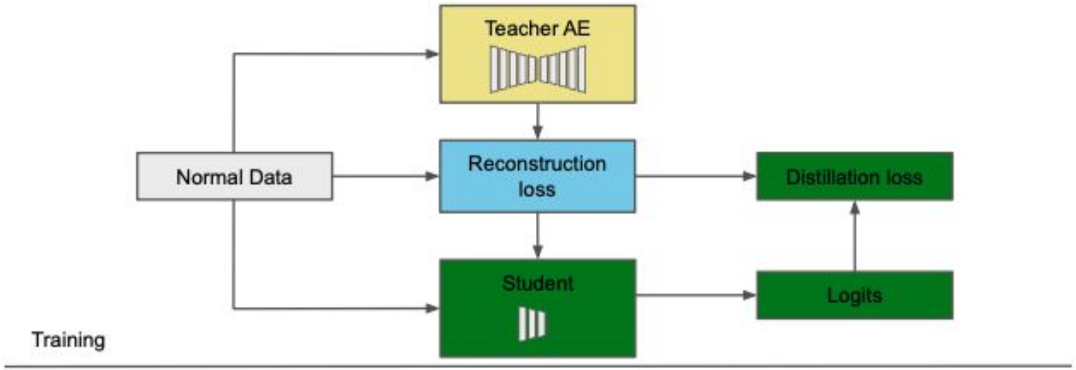
Detecting Anomalies

- Utilize **Autoencoder**, learns common features in data through unsupervised learning
- New Physics: rare, model independent
- “Anomalous” events will have a larger difference between input and output
- Difference quantified as **Anomaly Score**



Triggering on Anomalies

- Neural networks are effective; but, typically, their performance comes with a large computational resource consumption
- Using **Knowledge Distillation**, we project the performance of a (large, resource-intensive) Teacher Autoencoder to smaller **Student** quantized network



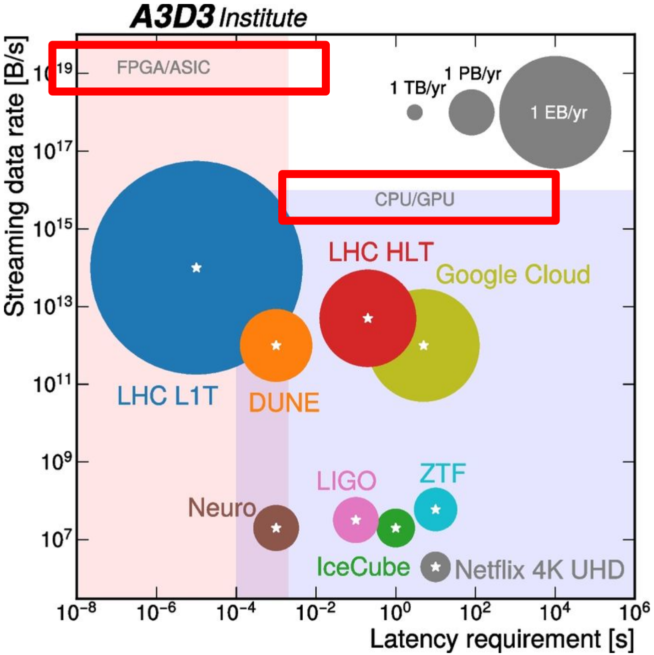
Adrian Alan Pol, Ekaterina Govorkova, Sonja Gronroos, Nadezda Chernyavskaya, Philip Harris et al. **Knowledge Distillation for Anomaly Detection**. Oct 9, 2023.

QKeras

Nevis Laboratories
Columbia University

Deploying on Hardware

- Input image processing rate needs to be faster than (generated) image streaming rate
- Require **hardware acceleration**
  ➔ use Field Programmable Gate Array (FPGA)
- Trained Student is converted using hls4ml
- Resource consumption benchmarking in progress

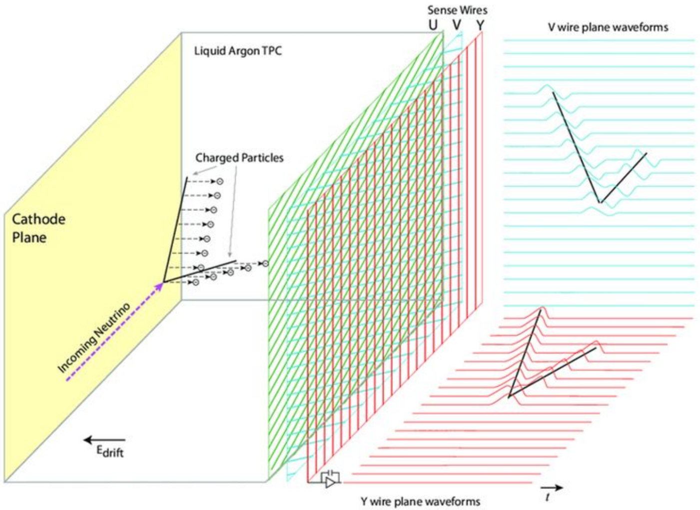


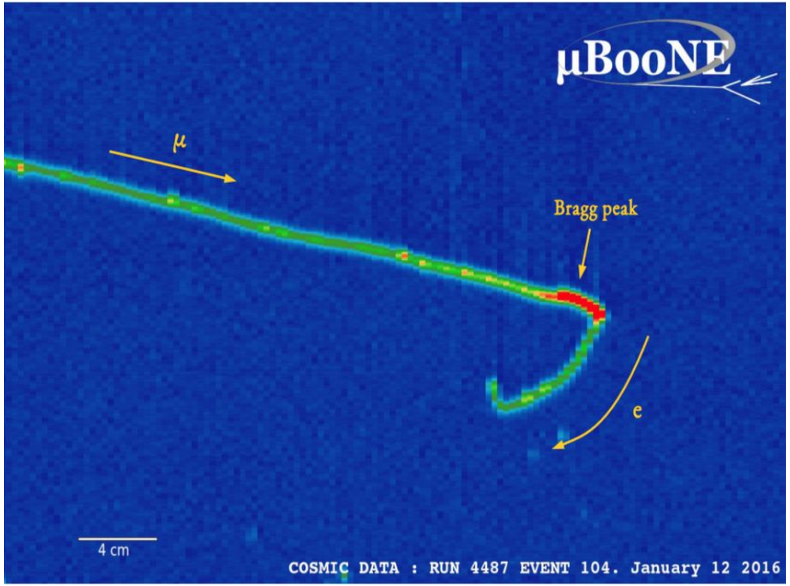
hls4ml

Nevis Laboratories
Columbia University

Liquid Argon Time Projection Chambers (LArTPCs)

- Widely used technology for neutrino physics (ArgoNeuT, MicroBooNE, SBND, ICARUS, DUNE, etc.)
- Neutrino interacts with Ar nuclei, creating charged particles
- Charged particles create ionization electrons, which are drifted in a large electric field and sensed by wire sensor arrays





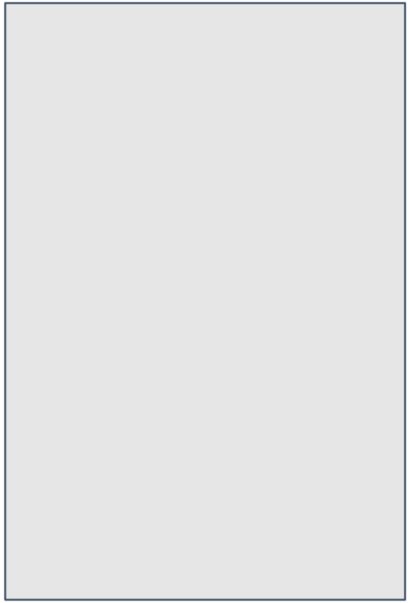
What Data Are We Using?

- MicroBooNE open samples: Simulated neutrino images overlaid on real cosmic images
- Data is labeled as neutrino or cosmic, but labels are not used in Teacher training
- To keep the Teacher computationally manageable, the input image was:

NEVIS LABORATORIES
COLUMBIA UNIVERSITY

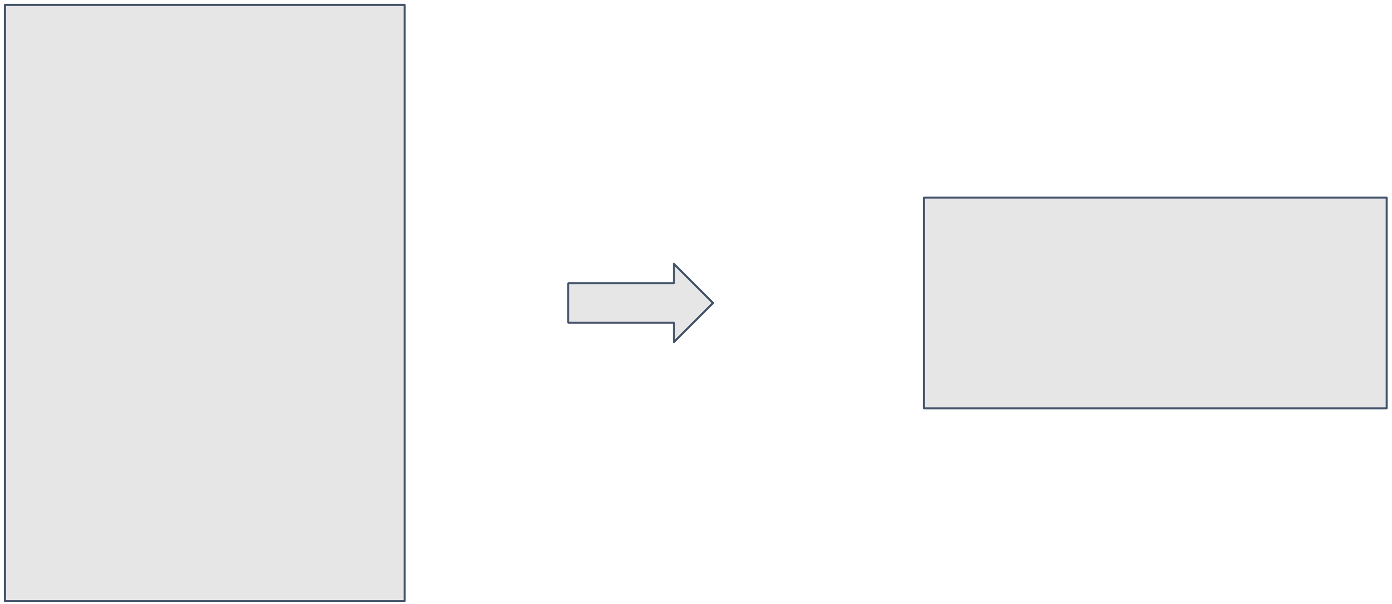
What Data Are We Using?

- To keep the Teacher computationally manageable, the input image was:
    - Originally 3456 × 6400 (Wires × Time)



What Data Are We Using?

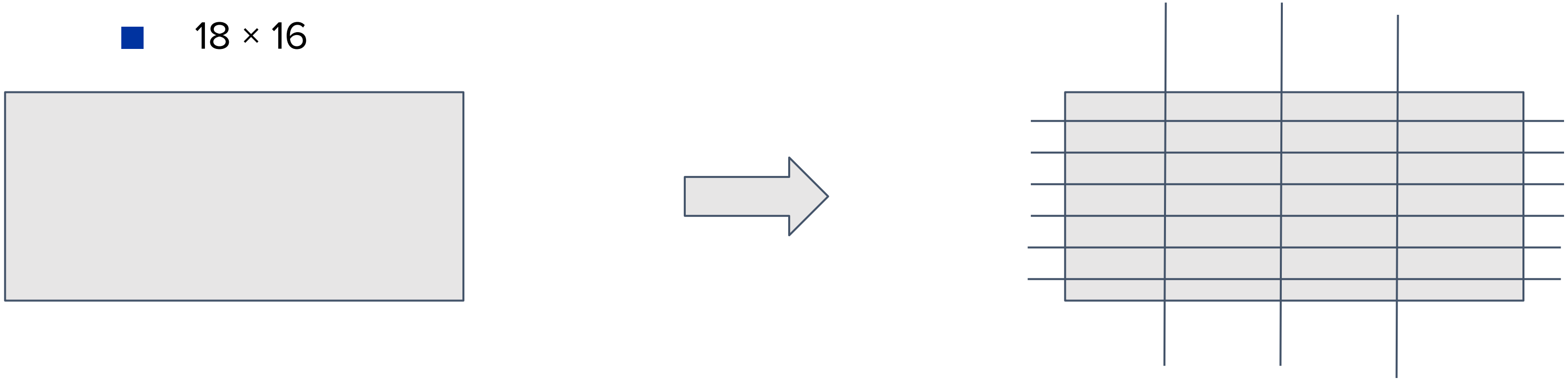
- To keep the Teacher computationally manageable, the input image was:
  - Originally 3456 × 6400 (Wires × Time)
  - Compressed by a factor of 10 in the time axis



Nevis Laboratories
Columbia University

What Data Are We Using?

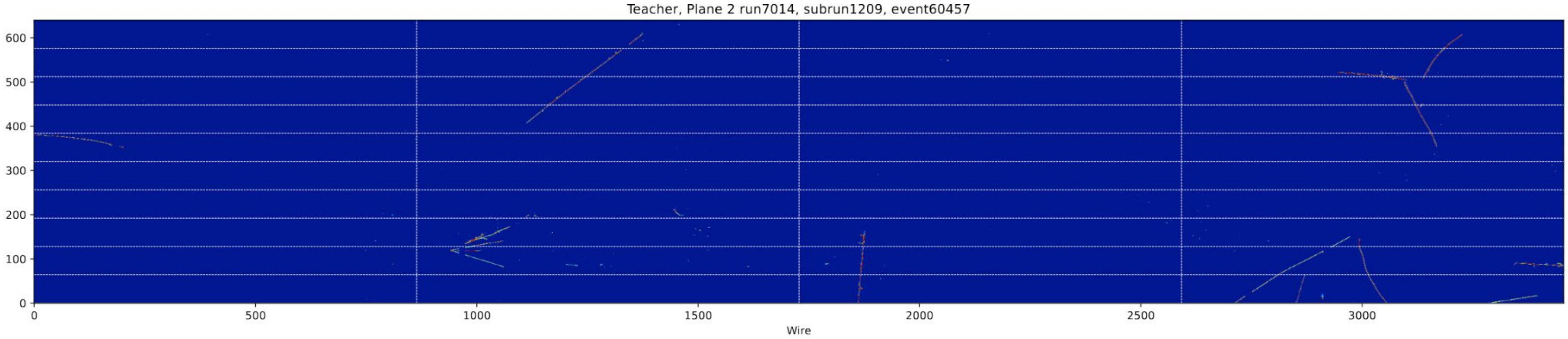
- To keep the Teacher computationally manageable, the input image was:
  - Originally 3456 × 6400 (Wires × Time)
  - Compressed by a factor of 10 in the time axis
  - Split into smaller images with four different sizes:
    - 864 × 64
    - 64 × 32
    - 18 × 16



Nevis Laboratories
Columbia University

What Data Are We Using?

- “Pixel Intensity” values were processed with:
  - Saturation at 100
  - Cutoff at 10
- Example processed image (864 × 64)



Triggering on Anomalies

- Size reduction by factor of ~75 (250 MB ➜ 3.4 MB)
- Teacher and Student Anomaly Scores are correlated

Model: "teacher"

Layer (type)	Output Shape	Param #
teacher_inputs_ (InputLayer)	[(None, 864, 64, 1)]	0
teacher_reshape (Reshape)	(None, 864, 64, 1)	0
teacher_conv2d_1 (Conv2D)	(None, 864, 64, 20)	200
teacher_relu_1 (Activation)	(None, 864, 64, 20)	0
teacher_pool_1 (AveragePooling2D)	(None, 432, 32, 20)	0
teacher_conv2d_2 (Conv2D)	(None, 432, 32, 30)	5430
teacher_relu_2 (Activation)	(None, 432, 32, 30)	0
teacher_flatten (Flatten)	(None, 414720)	0
teacher_latent (Dense)	(None, 80)	33177680

...
Total params: 66789361 (254.78 MB)
Trainable params: 66789361 (254.78 MB)
Non-trainable params: 0 (0.00 Byte)

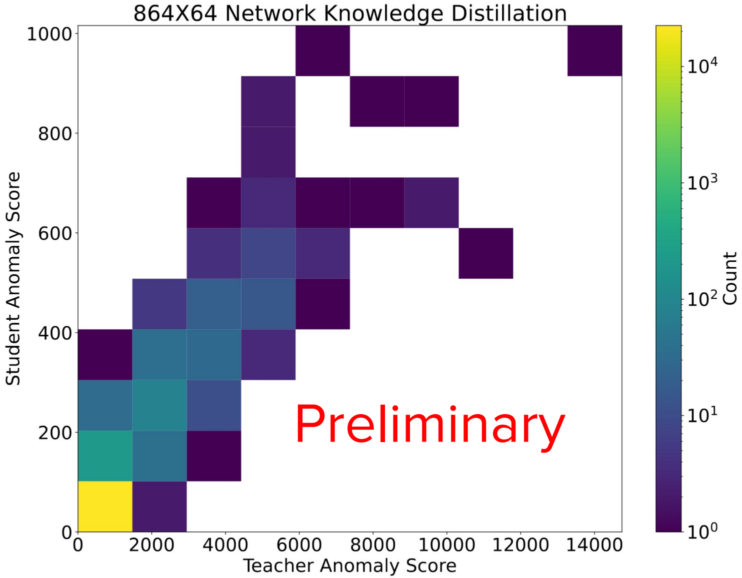
Teacher

Model: "v1_16X12"

Layer (type)	Output Shape	Param #
inputs_ (InputLayer)	[(None, 55296)]	0
dense1 (QDenseBatchnorm)	(None, 16)	884817
relu1 (QActivation)	(None, 16)	0
dropout_1 (Dropout)	(None, 16)	0
dense2 (QDense)	(None, 1)	16
outputs (QActivation)	(None, 1)	0

Total params: 884833 (3.38 MB)
Trainable params: 884800 (3.38 MB)
Non-trainable params: 33 (136.00 Byte)

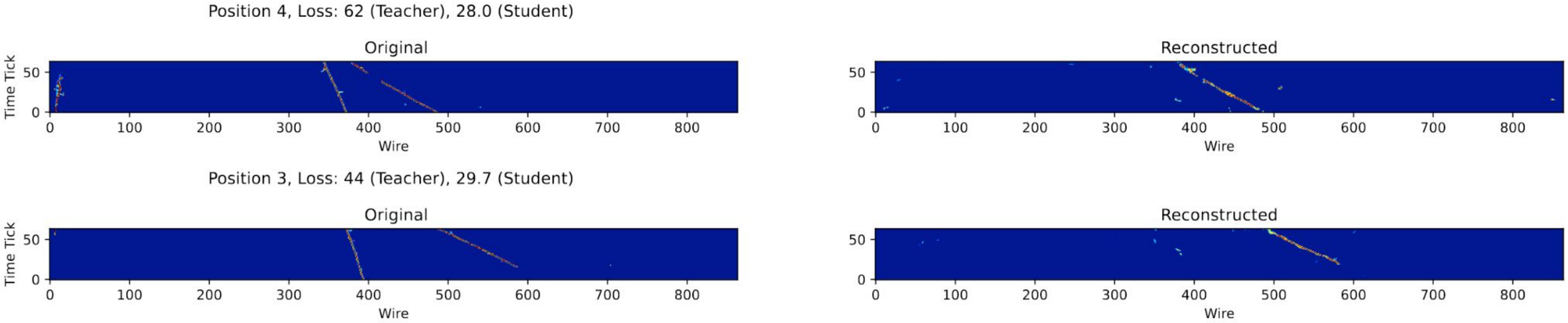
Student



NEVIS LABORATORIES
COLUMBIA UNIVERSITY

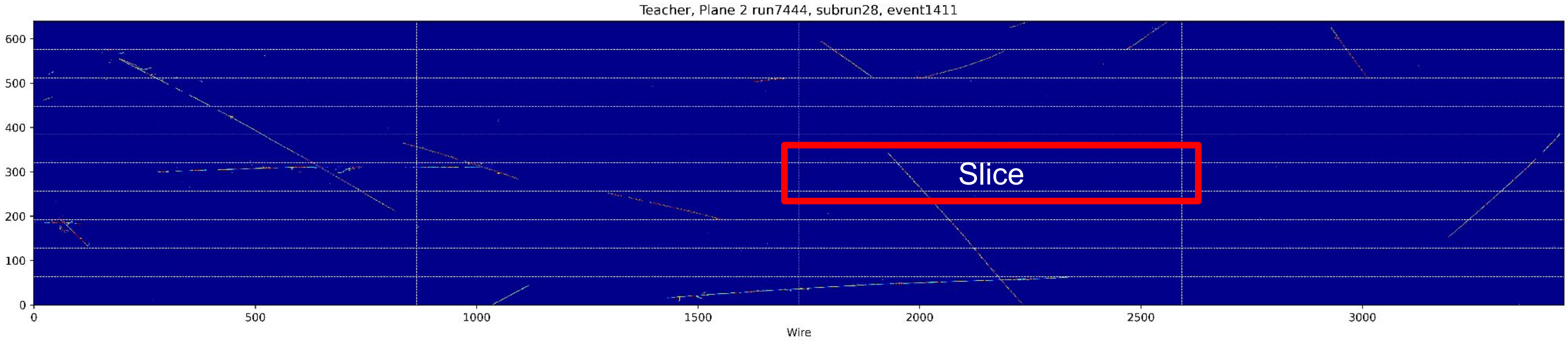
Understanding Model Performance - Anomalous Events

- The Teacher Autoencoder provides a visual representation of the differences between input and output images
- We analyzed individual Teacher outputs with high anomaly scores to identify features of anomalous images
- As shown in Figure, inputs with multiple tracks are not reproduced correctly, indicating anomalous behavior



Understanding Model Performance - Anomalous Events

- This behavior is qualitatively explainable
- The Teacher identifies common features in the input
- Most input images are either empty or have a single long track
- Therefore, images with multiple tracks will produce a higher anomaly score



Correlation - 864X64

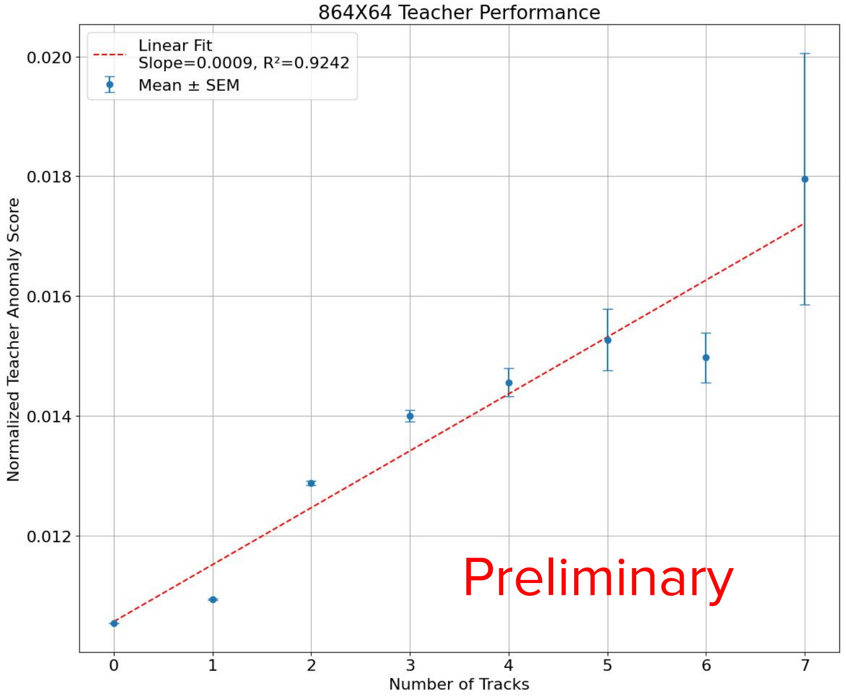


Table 2. ROC-AUC values for different anomaly scores with input size 864×64. The signal in each entry was defined as having exactly n tracks.

n Tracks	Teacher	Normalized Teacher	Student	Normalized Student
1	0.9676	0.9629	0.8595	0.8584
2	0.9660	0.9720	0.8777	0.8778
3	0.9714	0.9779	0.9341	0.9335
4	0.9752	0.9807	0.9521	0.9515
5	0.9810	0.9854	0.9585	0.9572
6	0.9943	0.9863	0.9951	0.9901
7	0.9855	0.9938	0.9835	0.9795

Model Performance - 864X64

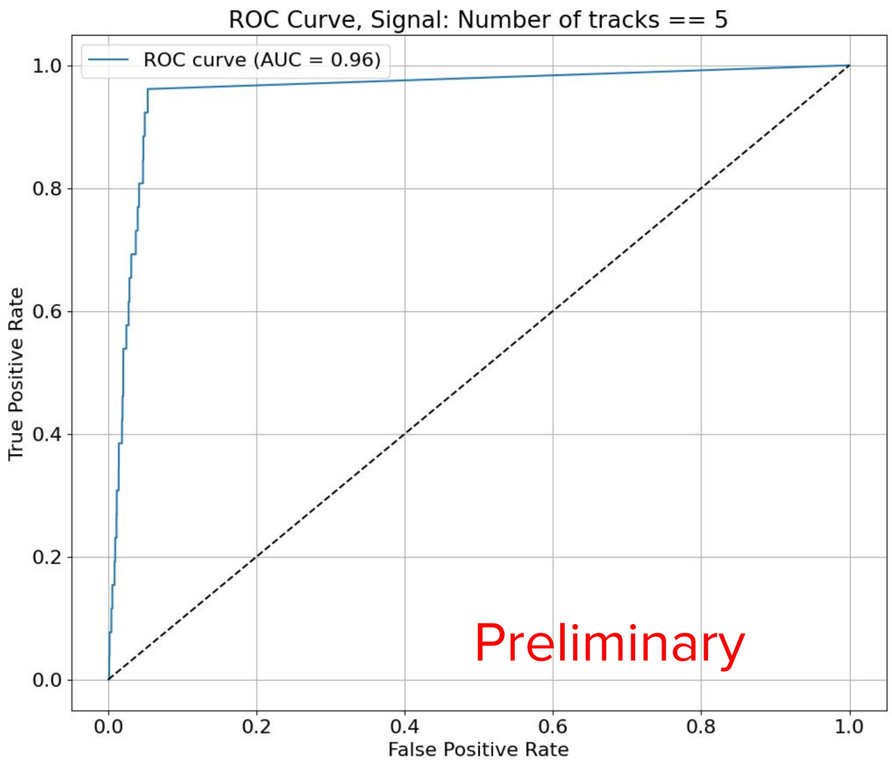


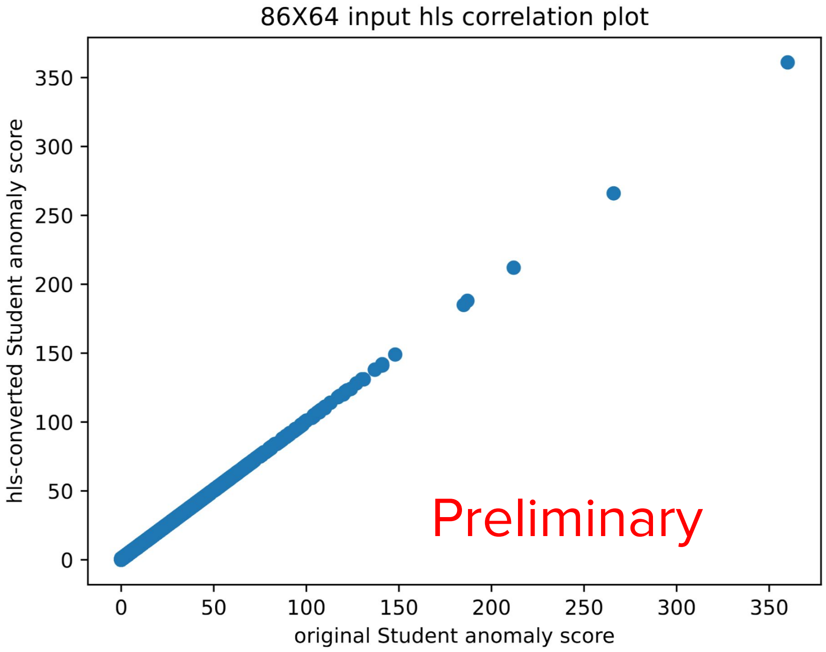
Table 2. ROC-AUC values for different anomaly scores with input size 864×64. The signal in each entry was defined as having exactly n tracks.

n Tracks	Teacher	Normalized Teacher	Student	Normalized Student
1	0.9676	0.9629	0.8595	0.8584
2	0.9660	0.9720	0.8777	0.8778
3	0.9714	0.9779	0.9341	0.9335
4	0.9752	0.9807	0.9521	0.9515
5	0.9810	0.9854	0.9585	0.9572
6	0.9943	0.9863	0.9951	0.9901
7	0.9855	0.9938	0.9835	0.9795

Nevis Laboratories
Columbia University

HLS-converted Student Performance

- Anomaly score comparison between original (python) and hls-converted (c) model



HLS-converted Student Performance

- Resource consumption estimates of middle and small Student network
- Both are over-utilizing available resources, work in process in reducing network size while maintaining performance

Name	BRAM_18K	DSP	FF	LUT	URAM
DSP	-	-	-	-	-
Expression	-	-	0	6	-
FIFO	-	-	-	-	-
Instance	0	26872	786957	2202910	0
Memory	-	-	-	-	-
Multiplexer	-	-	-	54	-
Register	-	-	131767	-	-
Total	0	26872	918724	2202970	0
Available	5376	12288	3456000	1728000	1280
Available SLR	1344	3072	864000	432000	320
Utilization (%)	0	218	26	127	0
Utilization SLR (%)	0	874	106	509	0

middle-sized student

Name	BRAM_18K	DSP	FF	LUT	URAM
DSP	-	-	-	-	-
Expression	-	-	0	6	-
FIFO	-	-	-	-	-
Instance	0	5329	127014	305312	0
Memory	-	-	-	-	-
Multiplexer	-	-	-	54	-
Register	-	-	19129	-	-
Total	0	5329	146143	305372	0
Available	5376	12288	3456000	1728000	1280
Available SLR	1344	3072	864000	432000	320
Utilization (%)	0	43	4	17	0
Utilization SLR (%)	0	173	16	70	0

small-sized student

NEVIS LABORATORIES
COLUMBIA UNIVERSITY

Conclusion and Outlook

- Autoencoders enable anomaly detection without prior knowledge or specific models
- Knowledge Distillation allows compression of computationally expensive Teacher model into more efficient Student model for hardware deployment
- Model successfully detects anomalies, particularly in multi-track events in LArTPC data
- Approach is scalable and applicable to LArTPC-based neutrino experiments
- Plan to apply same network for detector monitoring

This work was supported by the National Science Foundation under Grant No. OAC-2209917.

We acknowledge the MicroBooNE Collaboration for making publicly available the data sets [10.5281/zenodo.7262009] employed in this work. These data sets consist of simulated neutrino interactions from the Booster Neutrino Beamline overlaid on top of cosmic data collected with the MicroBooNE detector [2017 JINST 12 P02017].

Thank you

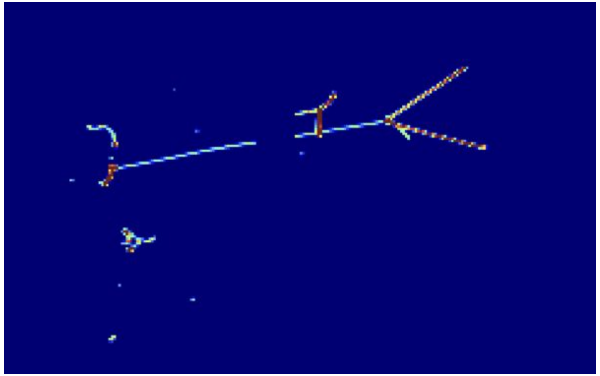
Backup

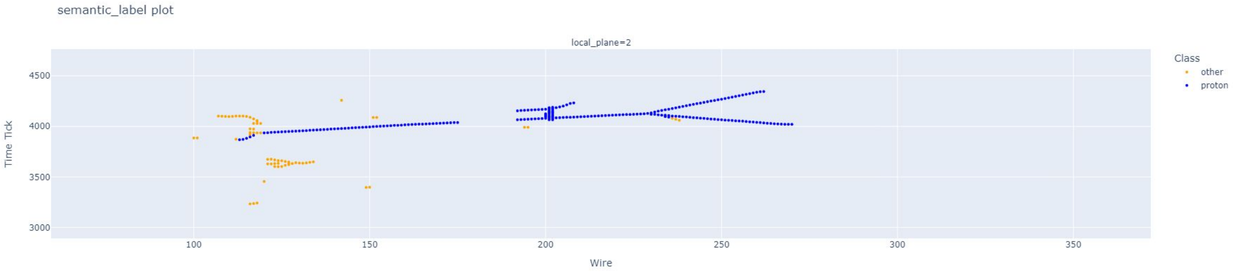
Network Structure

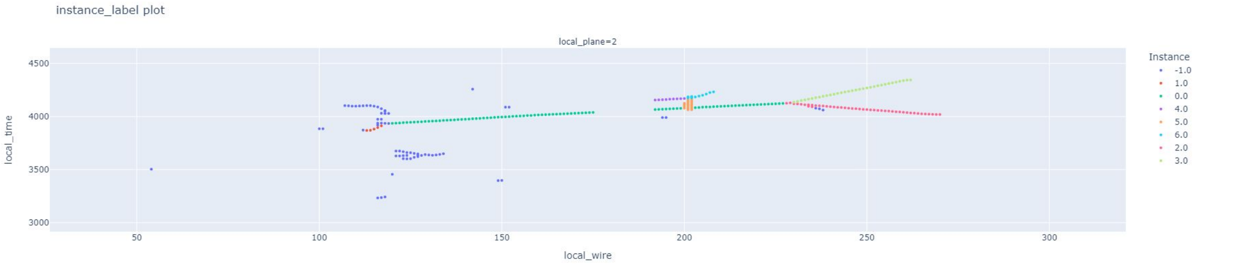
```python
class TeacherAutoencoder:
    def __init__(self, input_shape: tuple):
        self.input_shape = input_shape

    def get_model(self):
        inputs = Input(shape=self.input_shape, name="teacher_inputs_")
        x = Reshape((864, 64, 1), name="teacher_reshape")(inputs)
        x = Conv2D(20, (3, 3), strides=1, padding="same", name="teacher_conv2d_1")(x)
        x = Activation("relu", name="teacher_relu_1")(x)
        x = AveragePooling2D((2, 2), name="teacher_pool_1")(x)
        x = Conv2D(30, (3, 3), strides=1, padding="same", name="teacher_conv2d_2")(x)
        x = Activation("relu", name="teacher_relu_2")(x)
        x = Flatten(name="teacher_flatten")(x)
        x = Dense(80, activation="relu", name="teacher_latent")(x)
        x = Dense(432 * 32 * 30, name="teacher_dense")(x)
        x = Reshape((432, 32, 30), name="teacher_reshape2")(x)
        x = Activation("relu", name="teacher_relu_3")(x)
        x = Conv2D(30, (3, 3), strides=1, padding="same", name="teacher_conv2d_3")(x)
        x = Activation("relu", name="teacher_relu_4")(x)
        x = UpSampling2D((2, 2), name="teacher_upsampling")(x)
        x = Conv2D(20, (3, 3), strides=1, padding="same", name="teacher_conv2d_4")(x)
        x = Activation("relu", name="teacher_relu_5")(x)
        outputs = Conv2D(
            1,
            (3, 3),
            activation="relu",
            strides=1,
            padding="same",
            name="teacher_outputs",
        )(x)
        return Model(inputs, outputs, name="teacher")
```

```python
class V1_16X16:
    def __init__(self, input_shape: tuple):
        self.input_shape = input_shape

    def get_model(self):
        inputs = Input(shape=self.input_shape, name="inputs_")
        x = QDenseBatchnorm(
            16,
            kernel_quantizer=quantized_bits(16, 4, 1, alpha=1.0),
            bias_quantizer=quantized_bits(8, 3, 1, alpha=1.0),
            name="dense1",
        )(inputs)
        x = QActivation("quantized_relu(10, 6)", name="relu1")(x)
        x = Dropout(1 / 8)(x)
        x = QDense(
            1,
            kernel_quantizer=quantized_bits(12, 3, 1, alpha=1.0),
            use_bias=False,
            name="dense2",
        )(x)
        outputs = QActivation("quantized_relu(16, 8)", name="outputs")(x)
        return Model(inputs, outputs, name="v1_16X16")
```

Input Data

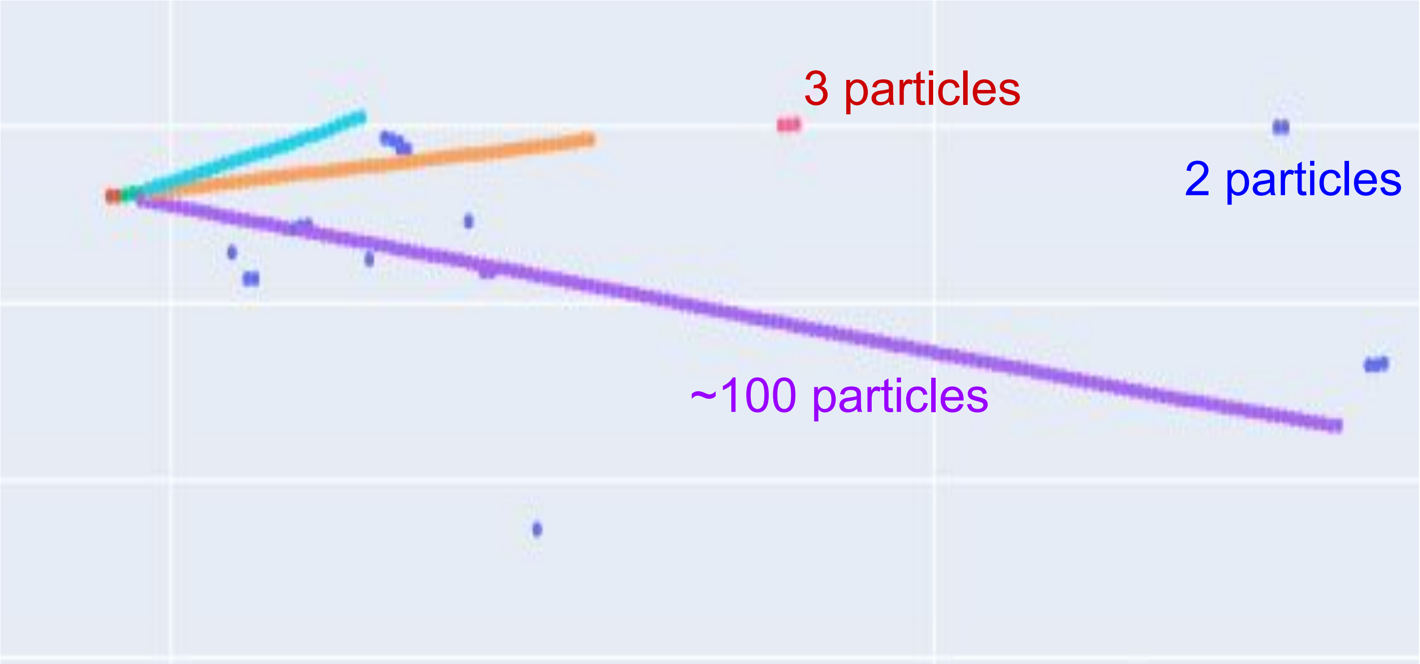



semantic_label plot
local_plane=2
Time Tick
4500
4000
3500
3000
100
150
200
250
300
350
Wire
Class
other
proton


instance_label plot
local_plane=2
local_time
4500
4000
3500
3000
50
100
150
200
250
300
local_wire
Instance
-1.0
1.0
0.0
4.0
5.0
6.0
2.0
3.0

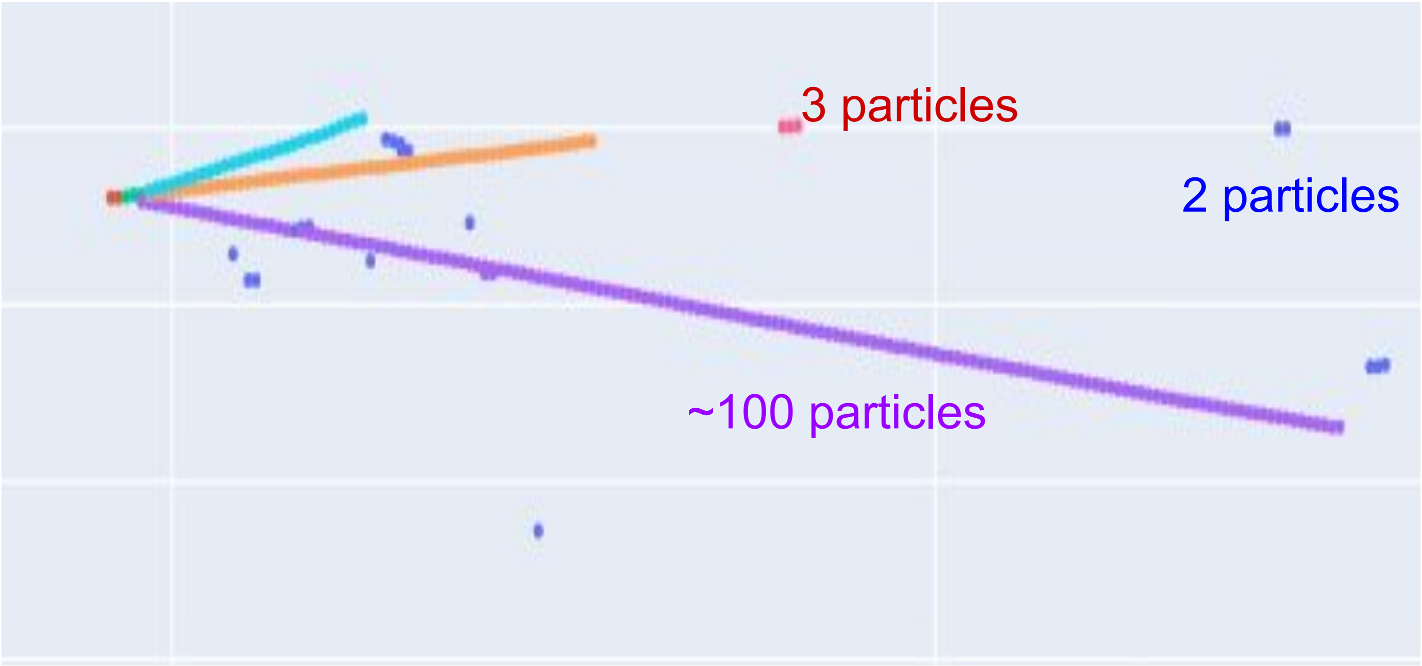
Model Performance

- Track_n_i: [n_i <= numbers of particles < n_(i+1)] of the same type need to be inside the input image to be recognized as a single track



Track Definition

- Track_n_i: [n_i <= numbers of particles < n_(i+1)] of the same type need to be inside the input image to be recognized as a single track



NEVIS LABORATORIES
COLUMBIA UNIVERSITY

Number of Tracks

- To verify the assumption, we used truth-level information from simulated neutrino data
- A set of criteria was applied to match visual tracks to the truth-level information
- Each interaction point is labeled with a Geant4 label, linking it to the particle causing the interaction
- To determine the number of tracks in the input image, we counted the different Geant4 label sets
- To distinguish track qualities, we divided track length by the number of particle interactions forming a single track
- Tracks criteria - Track_{n_x}: $n_{x-1} <$ number of interaction points $< n_{x}$ points

Correlation

- The anomaly score for an autoencoder is defined as the difference between input and output
- This anomaly score is proportional to the absolute amount of “ADC” in the input image
- To normalize the anomaly score, we divide it by the sum of “ADC” values in the input image

Correlation

- The different correlation plots between the normalized anomaly score and the number of tracks are shown
- Center dots represent the mean of the anomaly score distribution, and error bars indicate the standard deviation
- For the three different input image sizes, we observe a clear correlation between the normalized anomaly score and the number of tracks
  - 2592 mm X 512 mm, 192 mm X 256 mm, 96 mm X 72 mm
    (Full image is 10368 mm X 5120 mm)
- This indicates that our model is sensitive to multi-track inputs