Nevis Laboratories
Columbia University

Real-time Anomaly Detection for Charge-based Triggering in LArTPCs

Seokju Chung (Columbia University)

New Perspectives 2024
July 8th, 2024

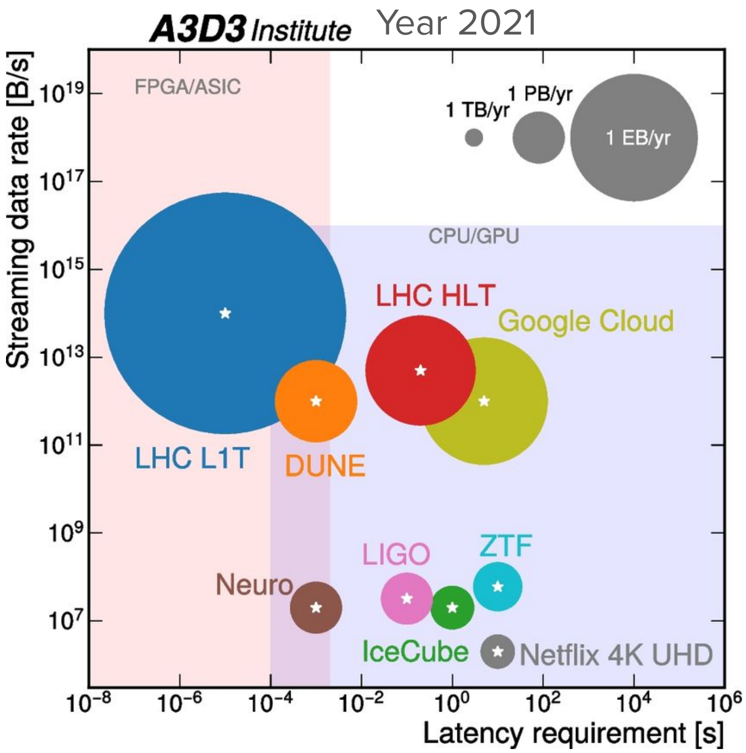
FERMILAB-SLIDES-24-0150-V

Real-Time Anomaly Detection

- NSF-funded collaborative project between Columbia and Princeton Universities
- Columbia (Neutrino) - G. Karagiorgi (PI), S. Chung, J. Cleeve, A. Malige
- Princeton (Collider) - I. Ojalvo (PI), L. Gerlach, A. Pol (Now at Thomson Reuters Lab)

Why Real-time Triggering?

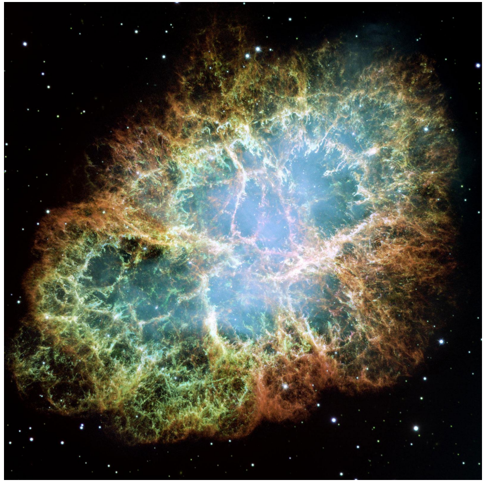
- Modern particle experiments generate large amount of data
- Impossible to save all; store in temporary buffer
- Need some kind of selection (trigger) which decides whether to keep buffer data or not



Nevis Laboratories
Columbia University

Why Trigger on Anomalies? - Data-Driven Trigger

- Experiments utilize different triggers for different physics signals of interest
  - SBN - External BNB trigger, 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



Nevis Laboratories
Columbia University

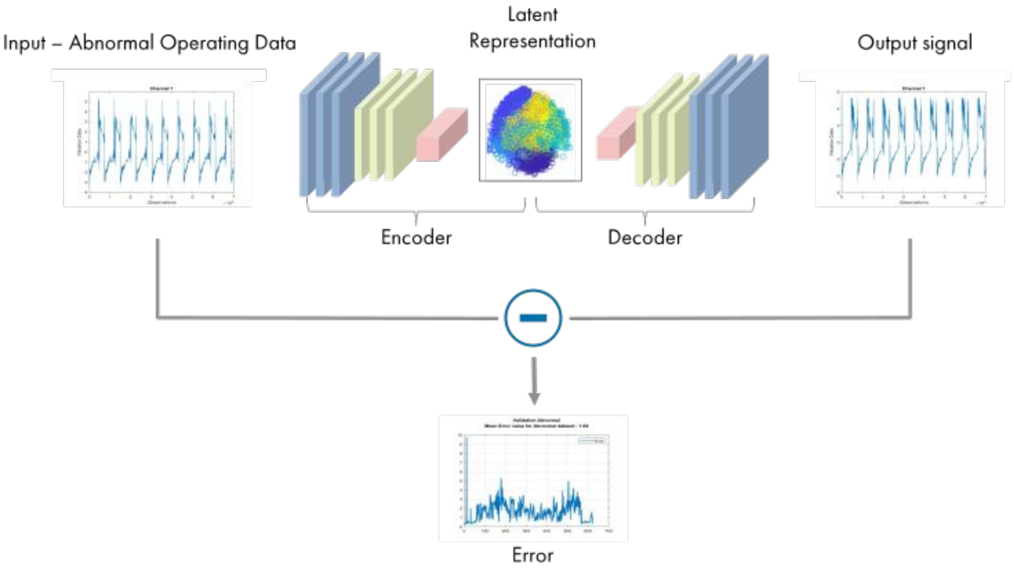
Why Trigger on Anomalies? - Anomaly Trigger

- 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

Detecting Anomalies

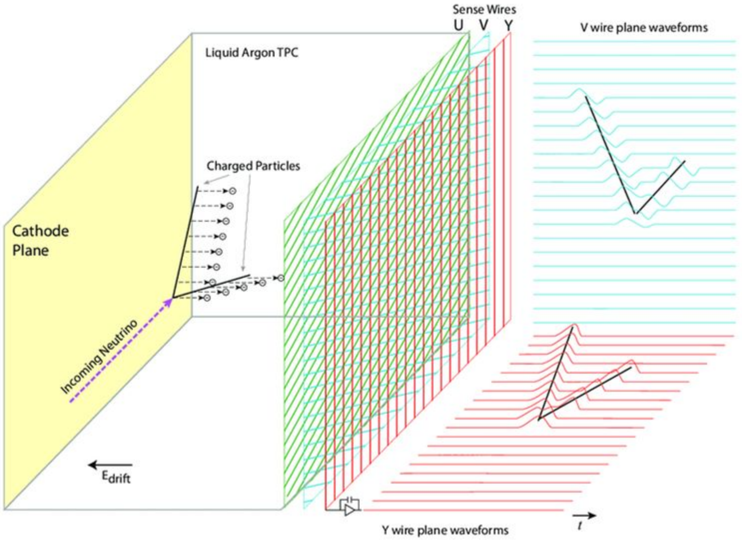
- Utilize **Autoencoder**, which encodes the data into a more compact format, and tries to "guess" what the original input was by decoding it
- Autoencoder learns common features in data through unsupervised learning
- "Anomalous" events will have a larger difference between input and output
- Difference quantified as **Anomaly Score**



Nevis Laboratories
Columbia University

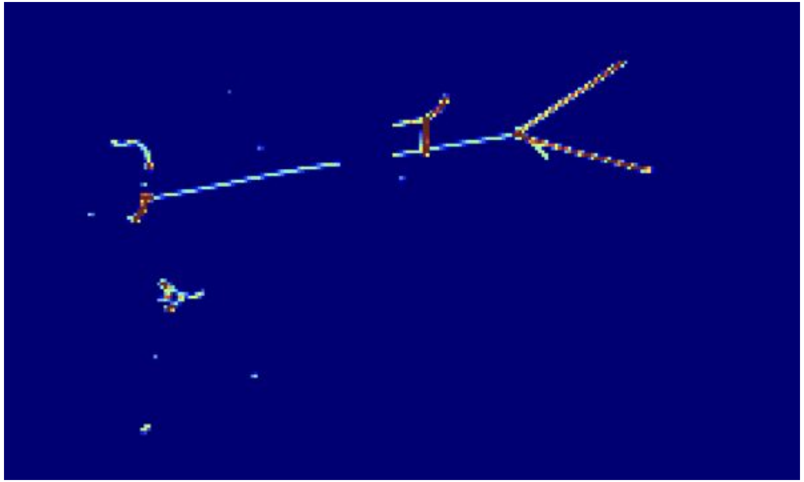
Liquid Argon Time Projection Chambers (LArTPCs)

- 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



LArTPCs - Input Data

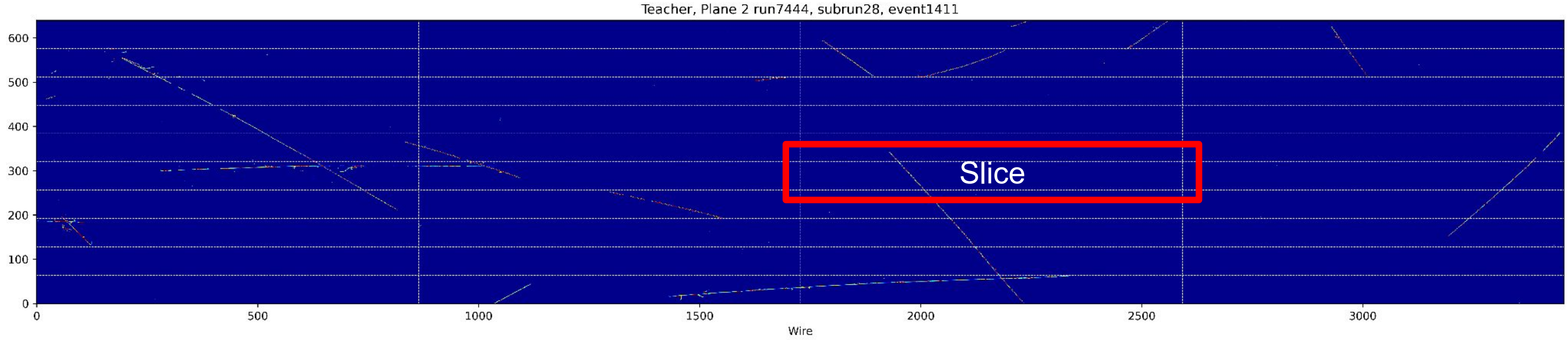
- Combine wire information (sensed ionization charge) as a function of time to get "image" of particle trajectories
- Neural Network: effective in image processing
- Use MicroBooNE Public Dataset for proof of concept on Autoencoders



Made with MicroBooNE Public Dataset
10.5281/zenodo.7262009

Input Data

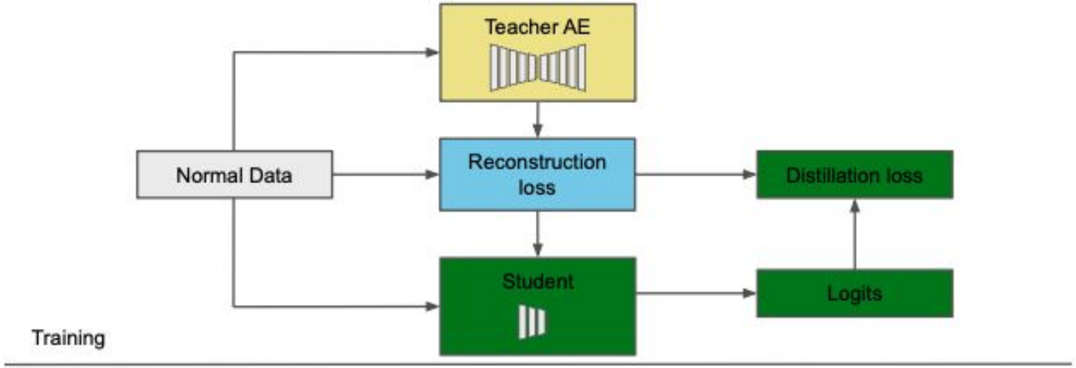
- 3456 Wires X 6400 Time Ticks → 3456 X 640 → 864 X 64



Nevis Laboratories
Columbia University

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.

Triggering on Anomalies

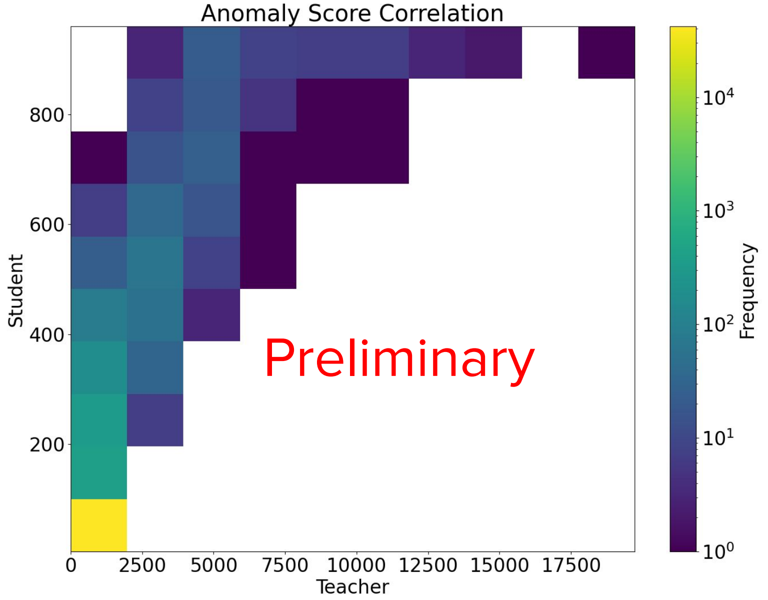
- Size reduction by factor of ~75 (250 MB ➔ 3.4 MB)
- Teacher and Student Anomaly Scores are correlated

```
Model: "teacher"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 teacher_inputs_ (InputLaye  [(None, 864, 64, 1)]      0
 r)

 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 (AveragePoo  (None, 432, 32, 20)       0
 ling2D)

 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

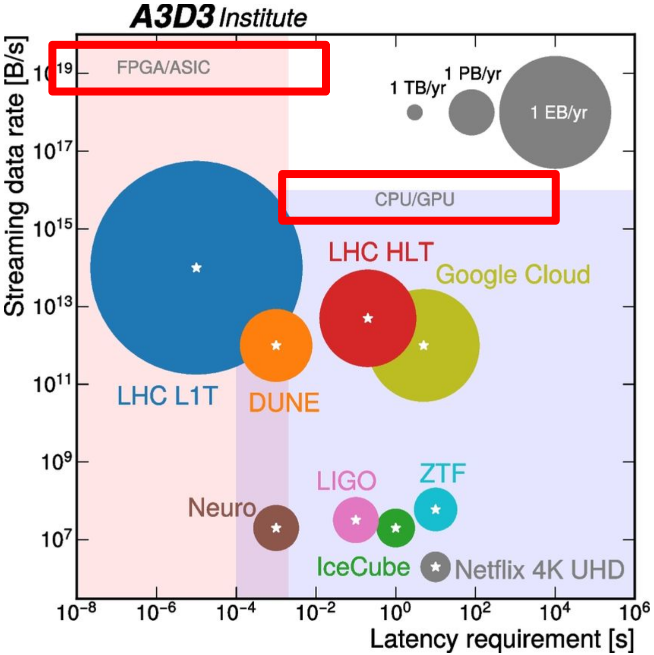


NEVIS LABORATORIES
COLUMBIA UNIVERSITY

Triggering on Anomalies

- 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

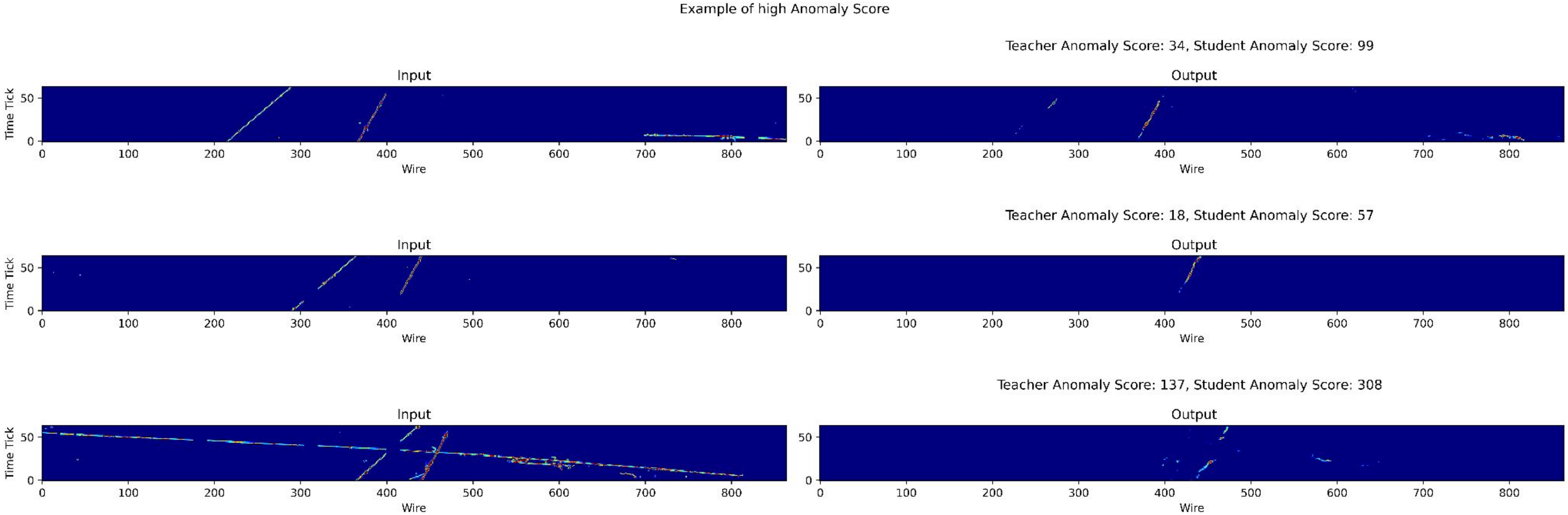
hls4ml



Nevis Laboratories
Columbia University

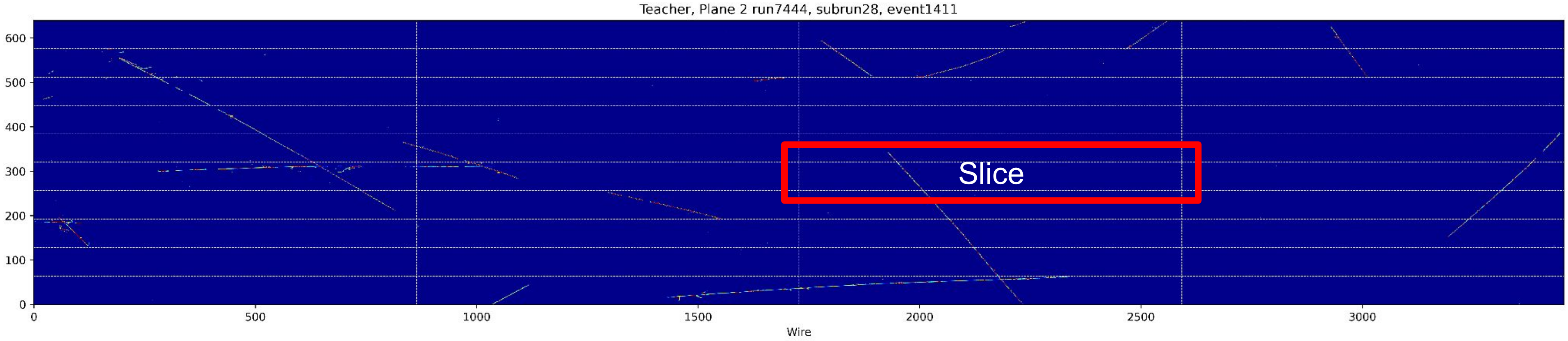
Network Performance

- See correlation between Anomaly Score and Number of Tracks in a given input image to the Teacher
- Study of performance metric in progress



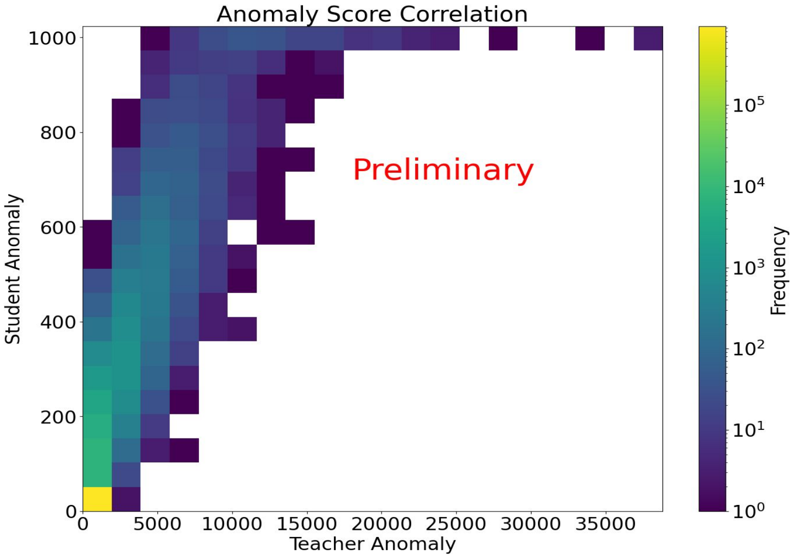
Network Performance

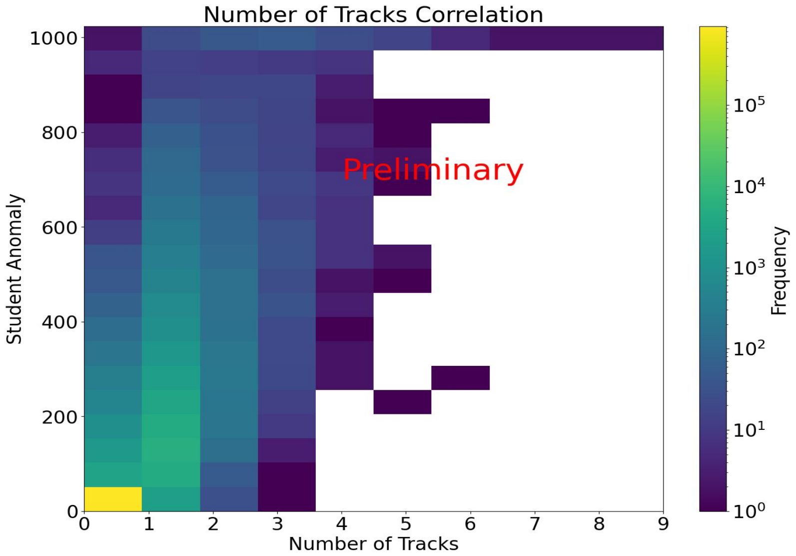
- Common feature: Empty or single track
- Anomaly expected to be having multiple tracks



Network Performance

- See correlation between Anomaly Score and Number of Tracks in a given input image to the Teacher
- Study of performance metric in progress





Conclusion

- Real-time Anomaly detection trigger can be developed using an Autoencoder with Knowledge Distillation
- We are exploring its applicability to LArTPC data using the MicroBooNE Public Dataset; hardware implementation and physics performance metrics in progress
- Not limited to LArTPC only, easily adaptable to different input datasets and applications across HEP

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].

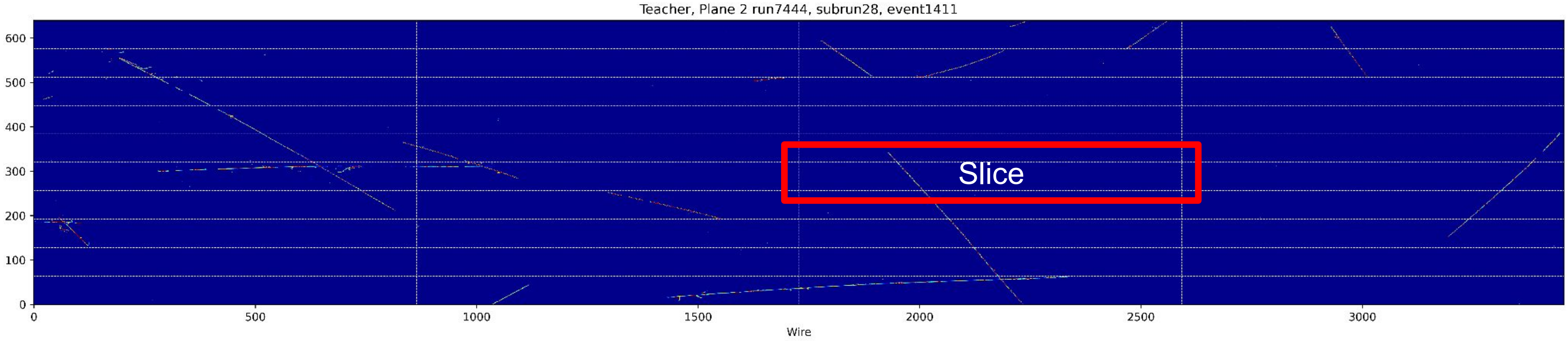
I thank URA for supporting my travel to Fermilab this summer (on SBND Activities)

Thank you

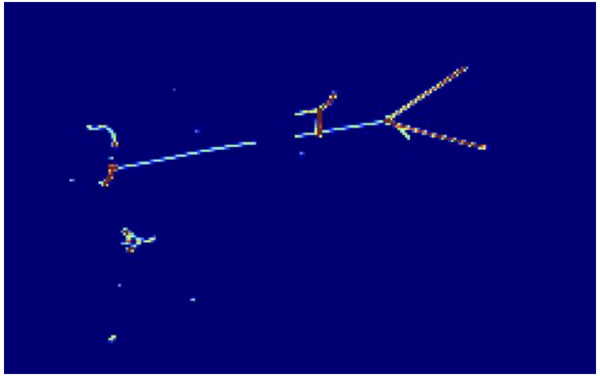
Backup

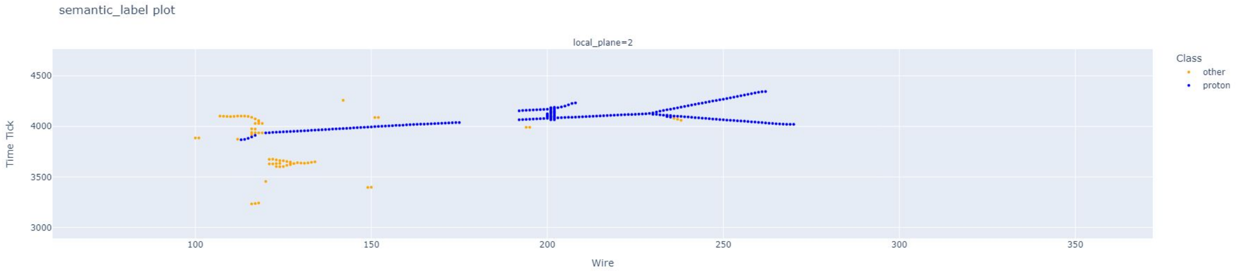
Input Data

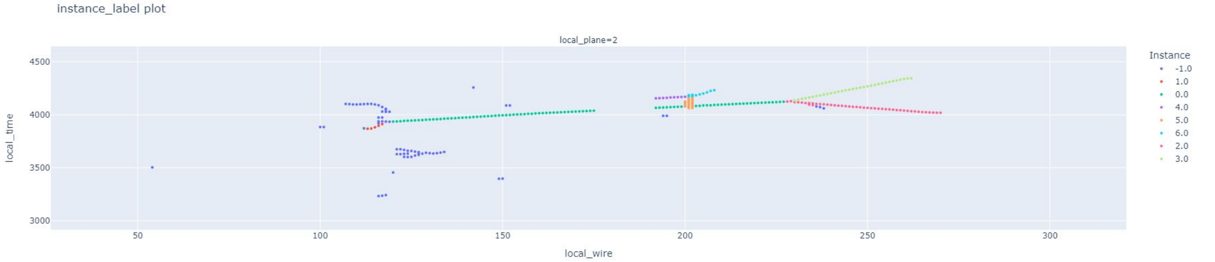
- 3456 Wires X 6400 Time Ticks → 3456 X 640 → 864 X 64



Input Data




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


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

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")
```