Binary Classification Model for Decay Events

Prince Bhaura, Yijie Wang, Murdock Aubry

University of Toronto

November 22, 2023

Table of Contents

Table of Contents

Introduction to NuGraph

1. NuGraph is a graph neural network (GNN) designed for reconstructing particle interactions in neutrino physics detector environments.
2. Trained and tested on DUNE data, this model is primarily used for the classification of detector hit particle type.
3. Additional functions include background hit rejection, event classification, clustering and vertex reconstruction.

Github: NuGraph

Data Collection: Time Projection Chamber

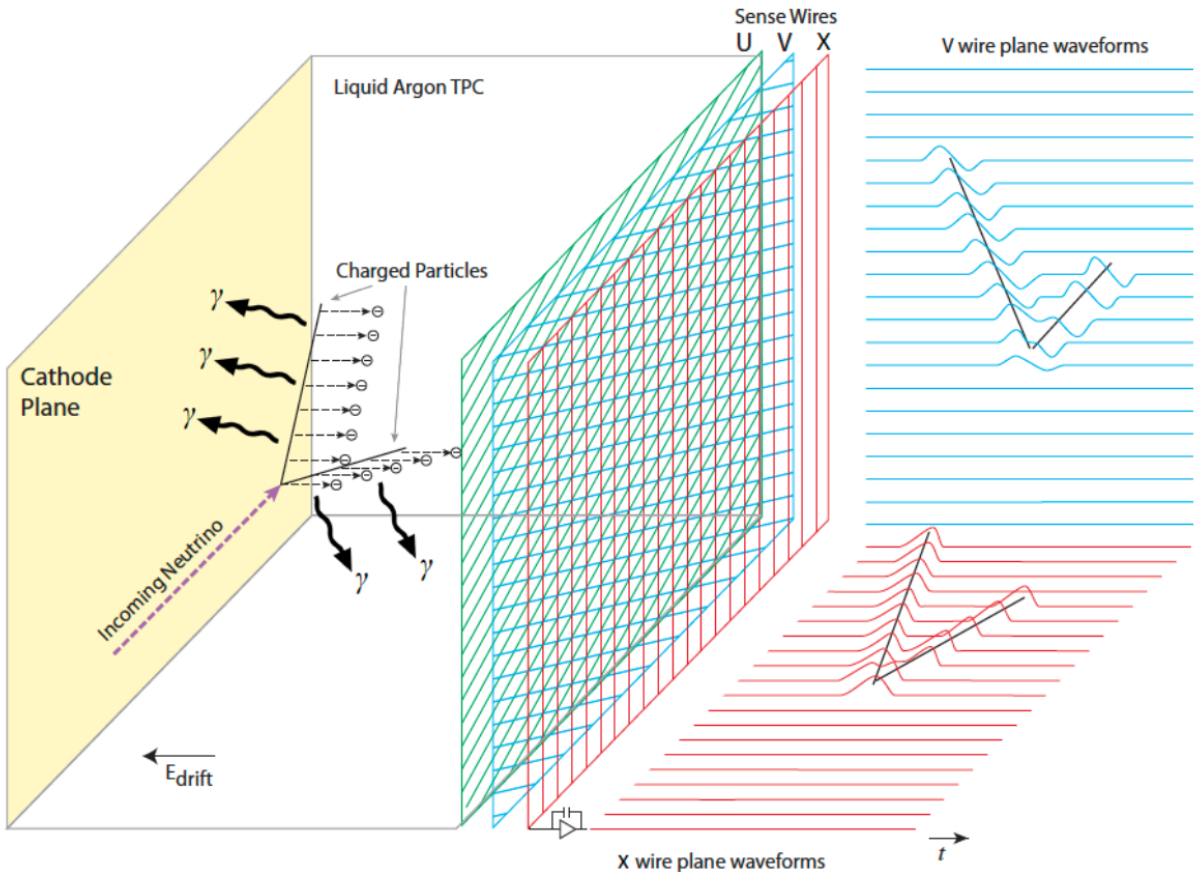


Figure: Source: Introduction to DUNE. Vol 1

Data Collection: Time Projection Chamber

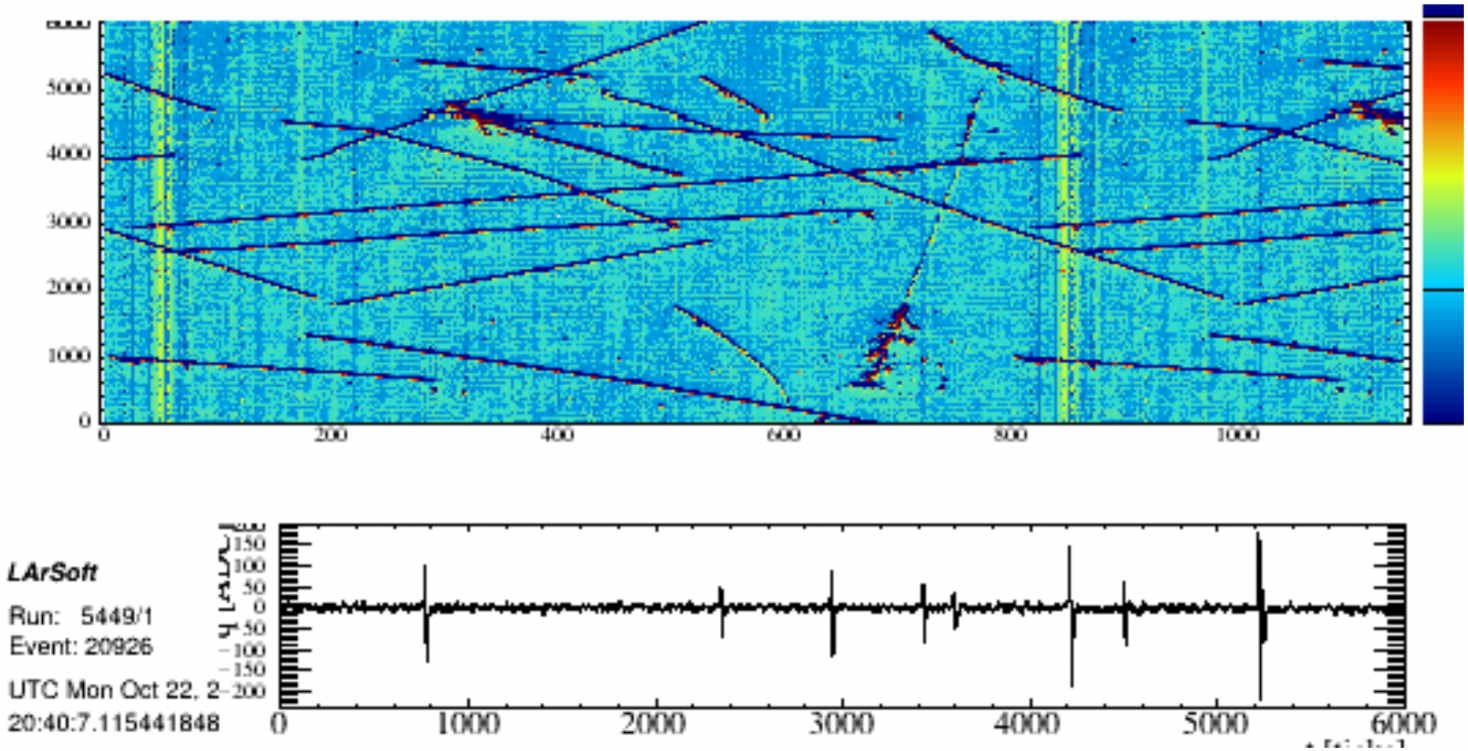


Figure: Example of pedestal-subtracted data for one ProtoDUNE-SP wire plane.
Source: Introduction to DUNE. Vol 1

Data Collection: Time Projection Chamber

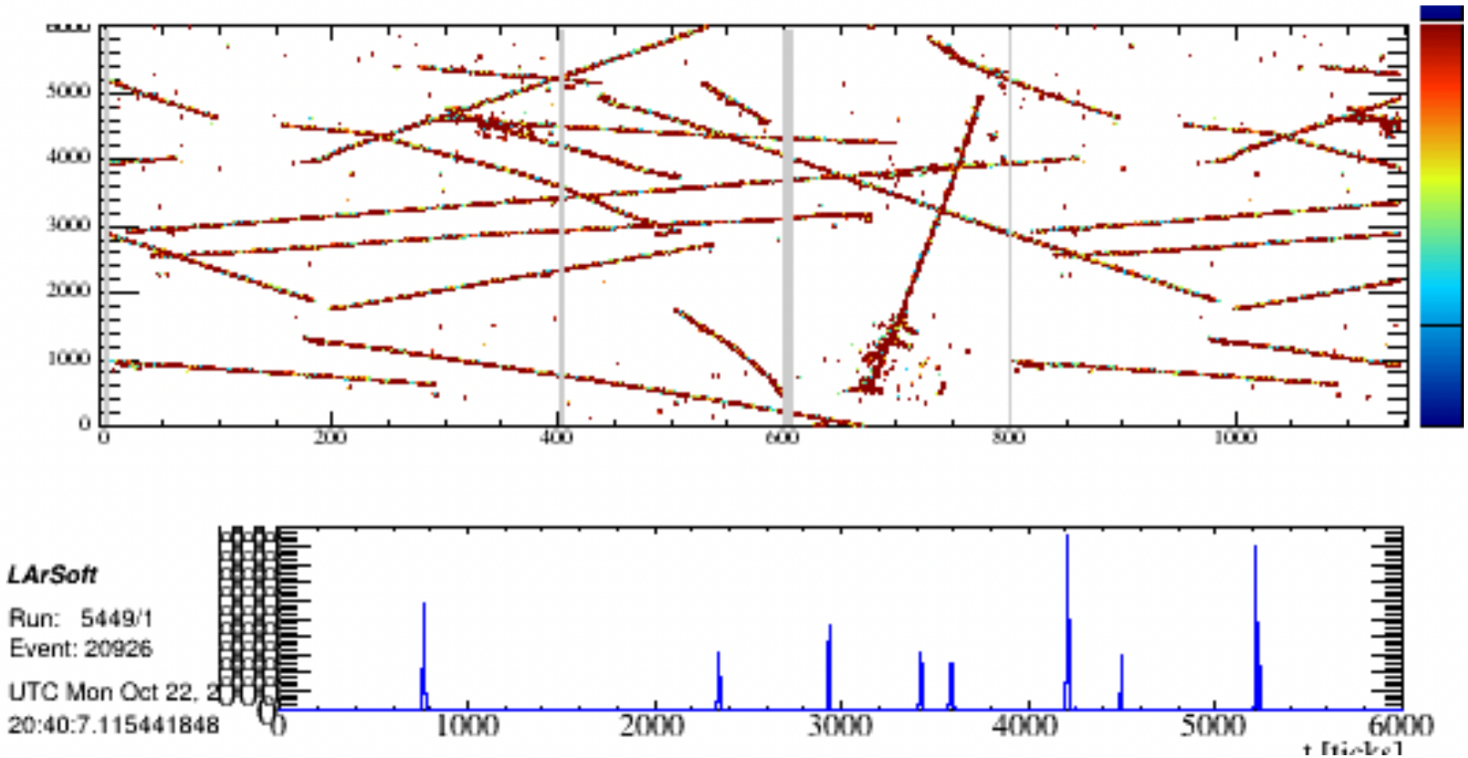


Figure: Calibrated, deconvoluted pedestal-subtracted data for one ProtoDUNE-SP wire plane. Source: Introduction to DUNE. Vol 1

Algorithm Structure

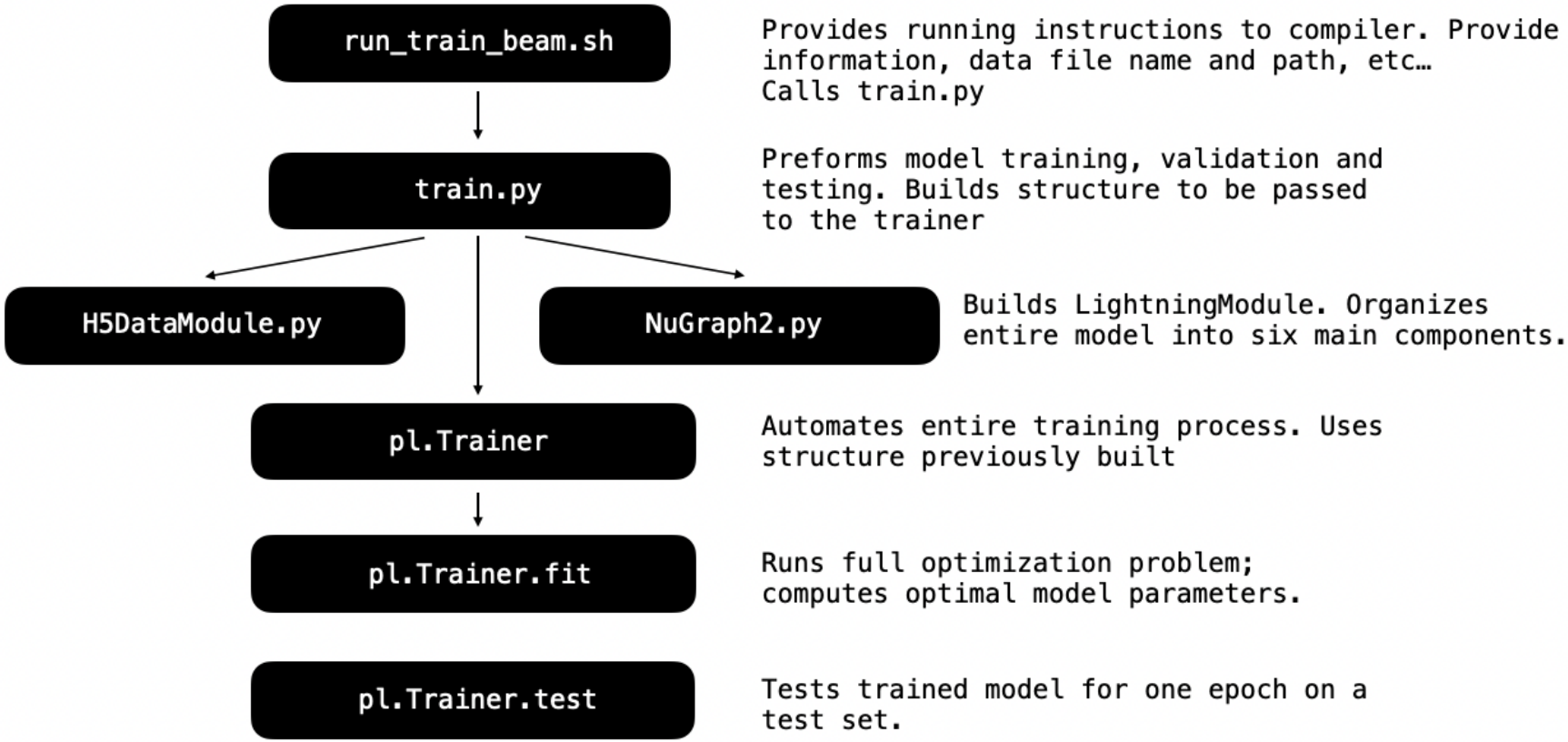


Figure: Description of NuGraph Neural Network. See LighningModule for further documentation.

NuGraph2.py Structure

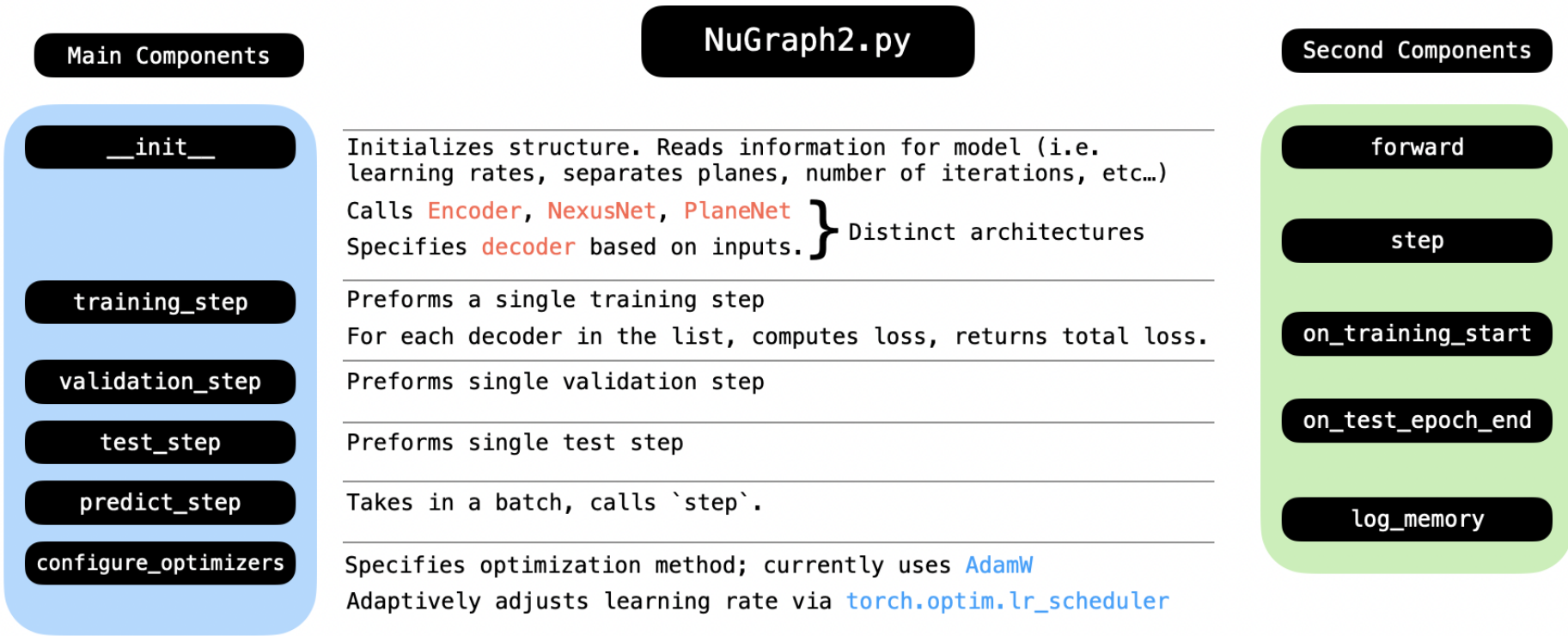


Figure: See AdamW

, OneCycleLR.

Layer Description

Encoder

NexusNet

PlaneNet

Feedforward Layers

Two-layer sequential network with one linear and one non-linear layer:

$$z = \tanh(Wx + b)$$

W, b are learned parameters

EdgeNet: Two-layer sequential network with Softmax.

$$z = \sigma(Wx + b) \qquad (\sigma(x))_i = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

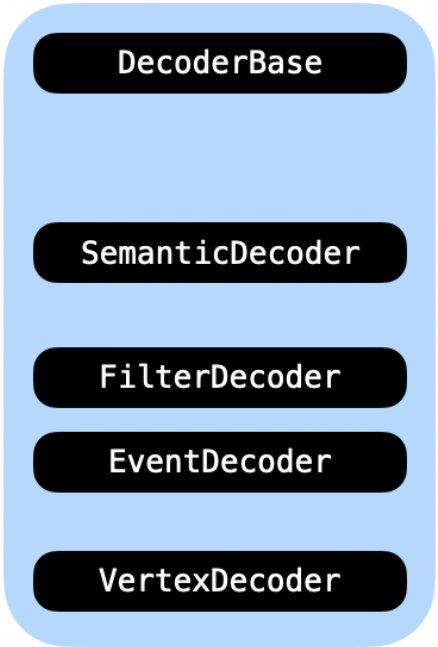
NodeNet: Four-layer sequential network.

$$z = \tanh\left(W_2 \tanh(W_1 x + b_1) + b_2\right)$$

W_i, b_i are learned parameters

Very similar to NexusNet

Decoder Descriptions



DecoderBase	Base class for all NuGraph decoders Specifies loss update based on cooling parameters Updates metrics (loss information over time)
SemanticDecoder	Convolve down to a single score per semantic class for each plane Uses RecallLoss
FilterDecoder	Convolve down to a single score, filters out noise. Uses BCELoss
EventDecoder	Convolve graph node features down to a single score for entire event Uses RecallLoss
VertexDecoder	Uses LogCoshLoss

score: model's guess

Loss Functions

RecallLoss

Really just cross-entropy loss:

$$\mathscr{L}(\theta) = \text{Avg}_i \; CE(y_i, \sigma(f(x_i, \theta))$$

$$CE(p, \hat{p}) = -\sum_i p_i \ln(\hat{p}_i)$$

LogCoshLoss

Uses the soft plus function;

$$X = \sqrt{\sum_i \left(f(x_i, \theta) - y_i\right)^2}$$

$$L(x) = \log(1 + \exp(x))$$

Smooth analog of ReLU

$$\mathscr{L}(\theta) = \text{Avg}_i \left(X + F(-2X) - \log(2)\right)$$

H5Dataset.py & H5DataModule.py Description

- H5Dataset.py defines a class which represents neutrino decay data sets
- H5DataModule.py also defines a similar data class based on the PyTorch LightningDataModule (inheritance from the LightingDataModule)

Table of Contents

Building Training Data

- New training data will need to be configured as follows:
  - Event follows the decay mode we are interested in
  - Event does not follow the decay mode we are interested in
- This configuration can be done by looking at the physical characteristics (e.g energy, momentum, spin) which can be found in the existing data
- Create a script to classify decay events of interest from existing data
  - Once classified, attach a label (e.g $1 \rightarrow$ decay of interest has happened, $0 \rightarrow$ decay of interest has not happened) and send to new training model

Pion Decay Modes

- Focus will be on looking for decays modes involving τ, ν_τ ($+$ other particles), and Pions
- Current plan is to look for Pion decay modes with the correct spins
- We are going to look into this more starting this week

Table of Contents

Model Adjustments

- Alter model to be able to handle this new class of data
- Reconfigure `H5DataModule.py`
- Choose ideal architecture (i.e. mimic `NexusNet`, `PlaneNet`, `etc...`)
- Choose ideal loss function (i.e CE? LogCosh?)