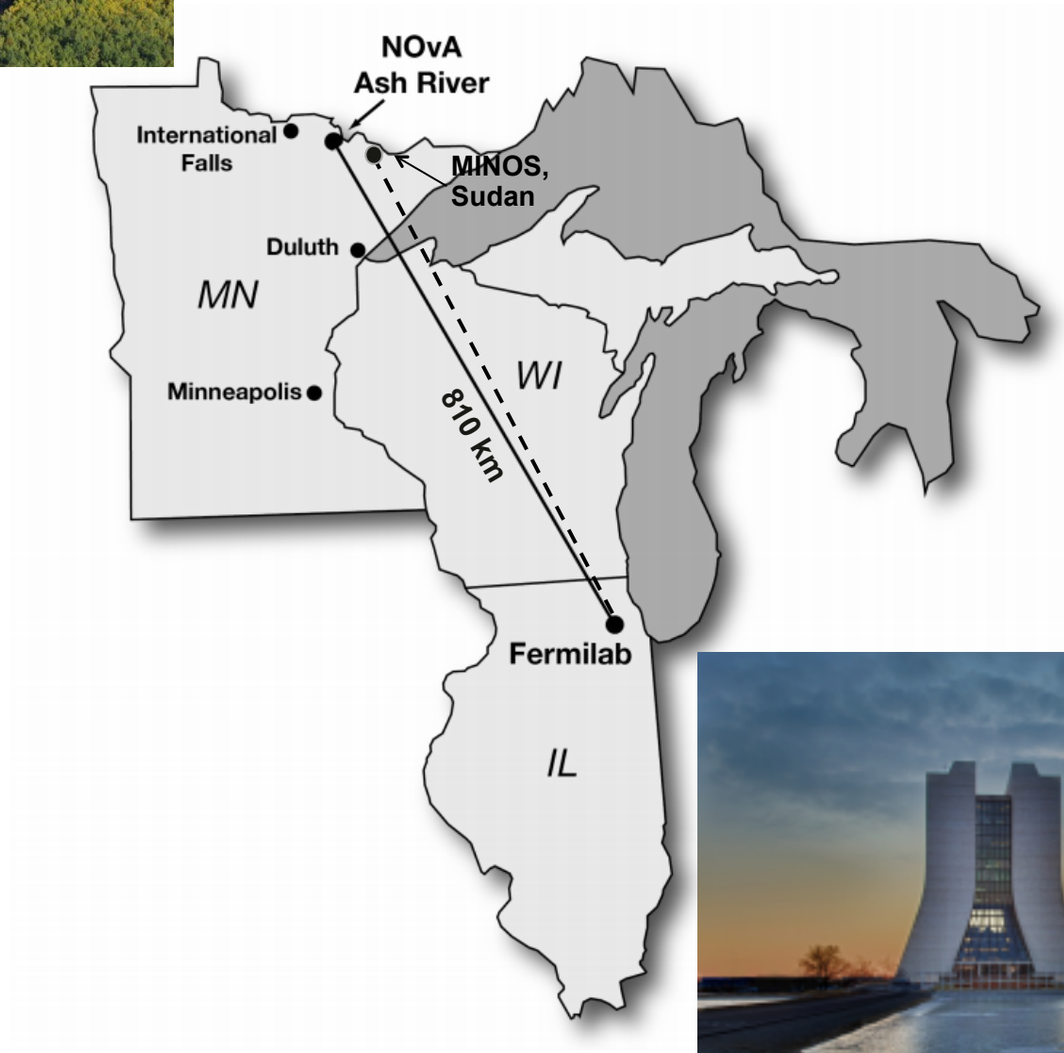
The College Of WILLIAM & MARY

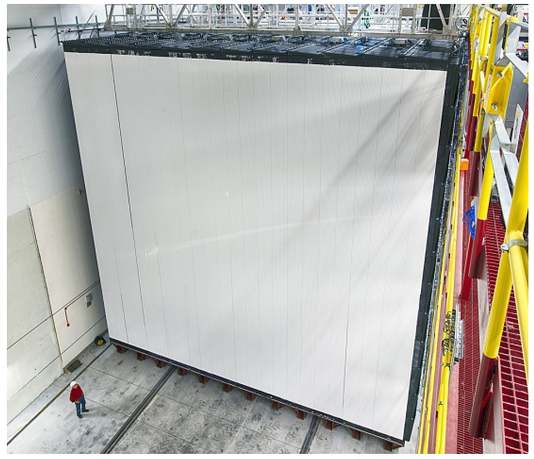
Deep Learning at NOvA with ART

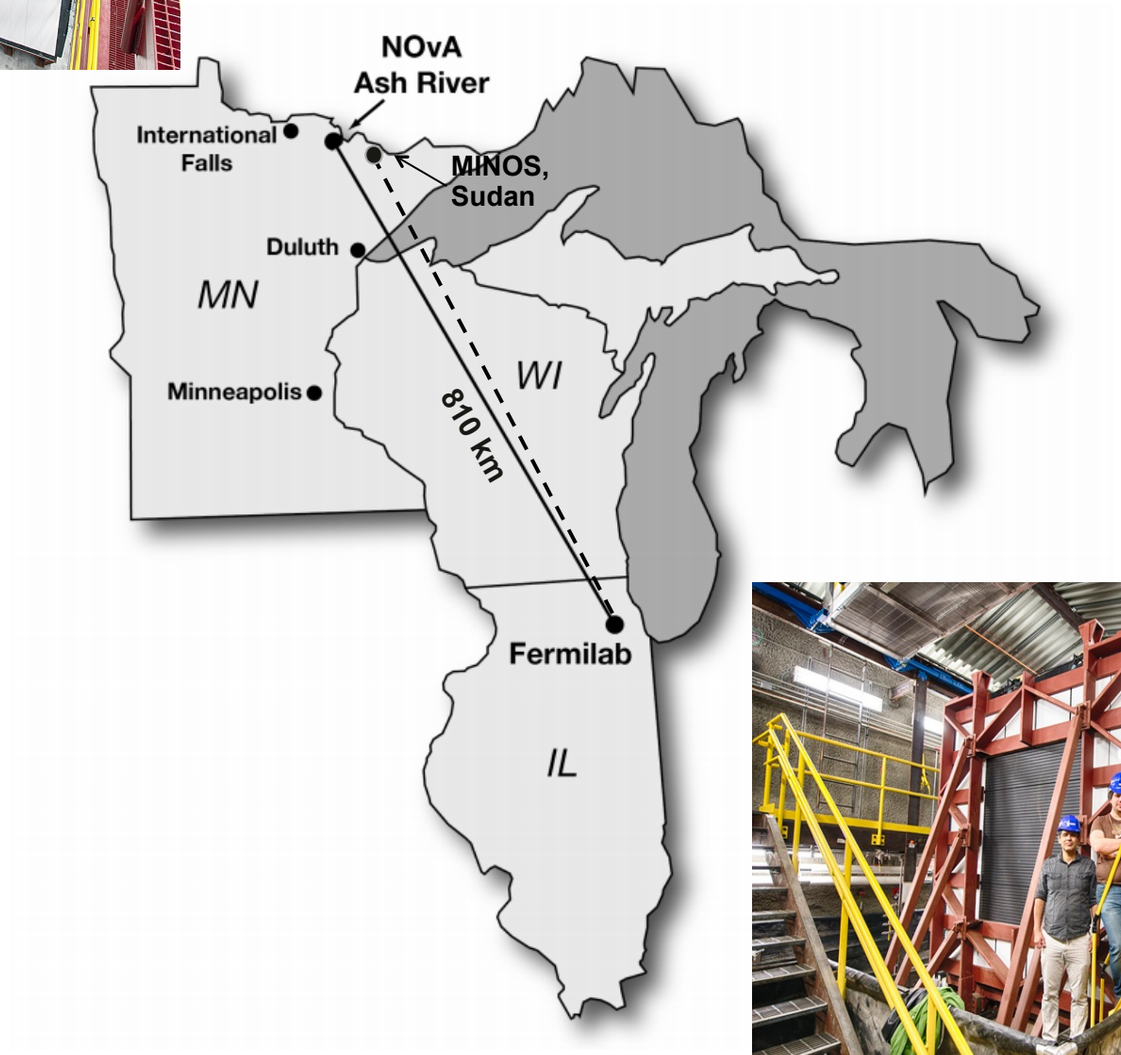
Alexander Radovic
College of William and Mary

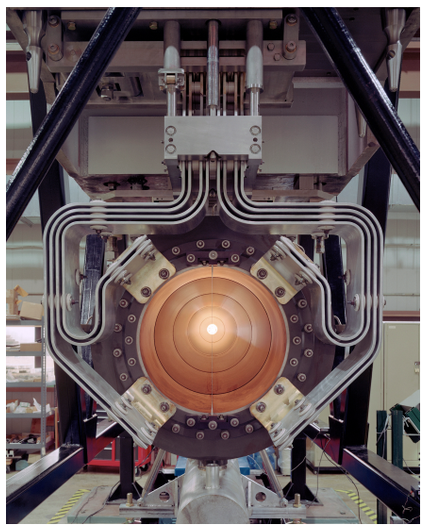
The NOvA Experiment



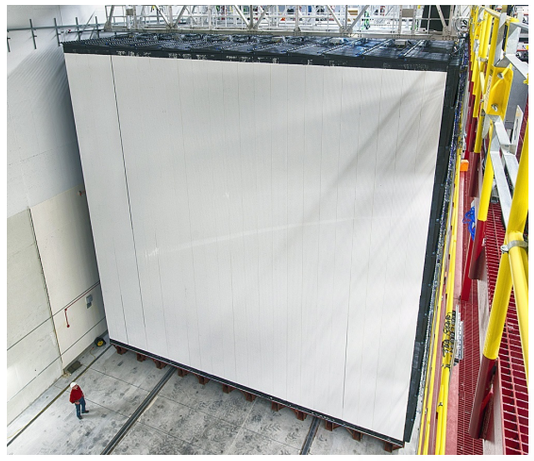
The NOvA Experiment

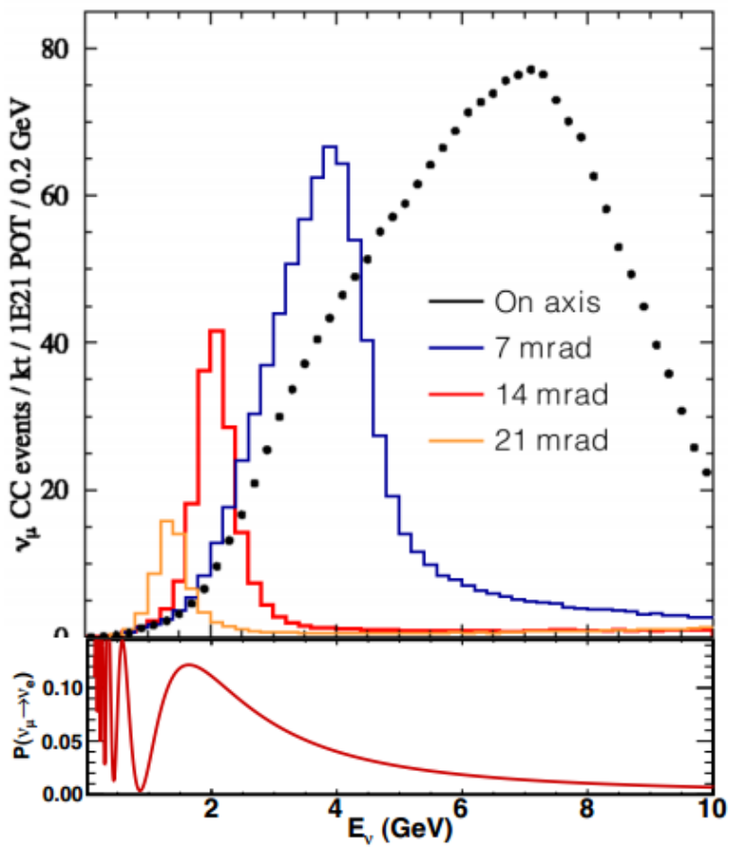


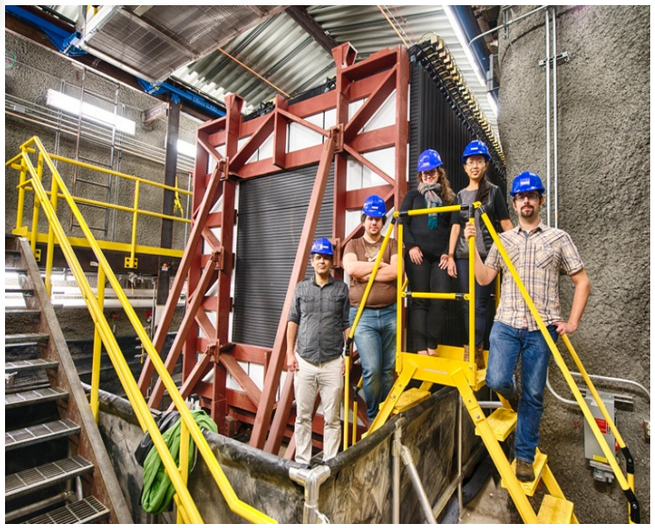


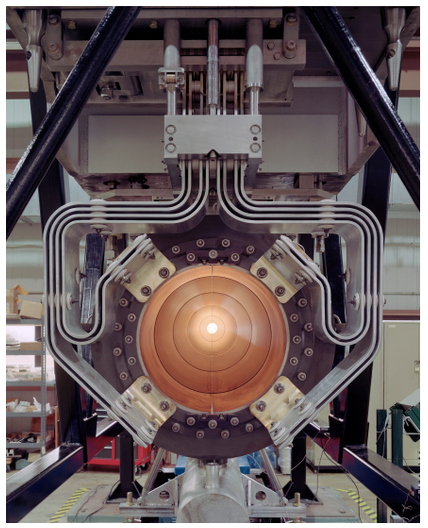


The NOvA Experiment

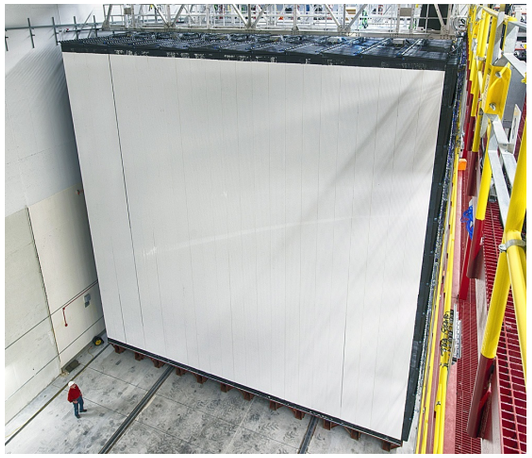


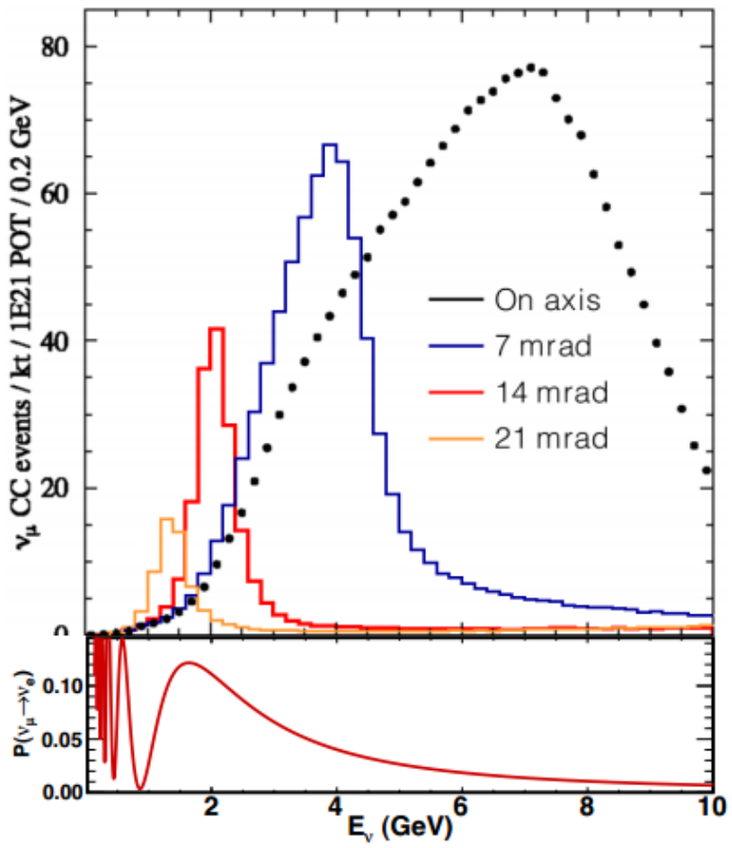


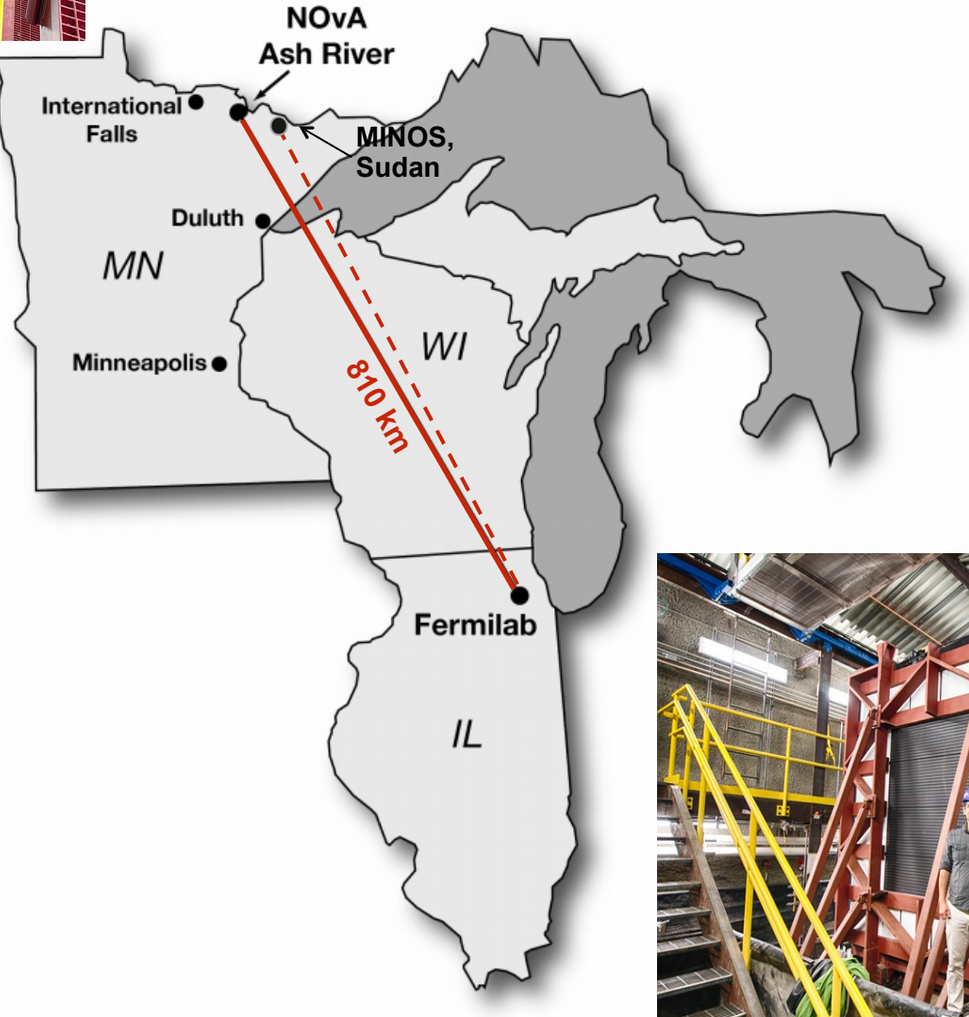


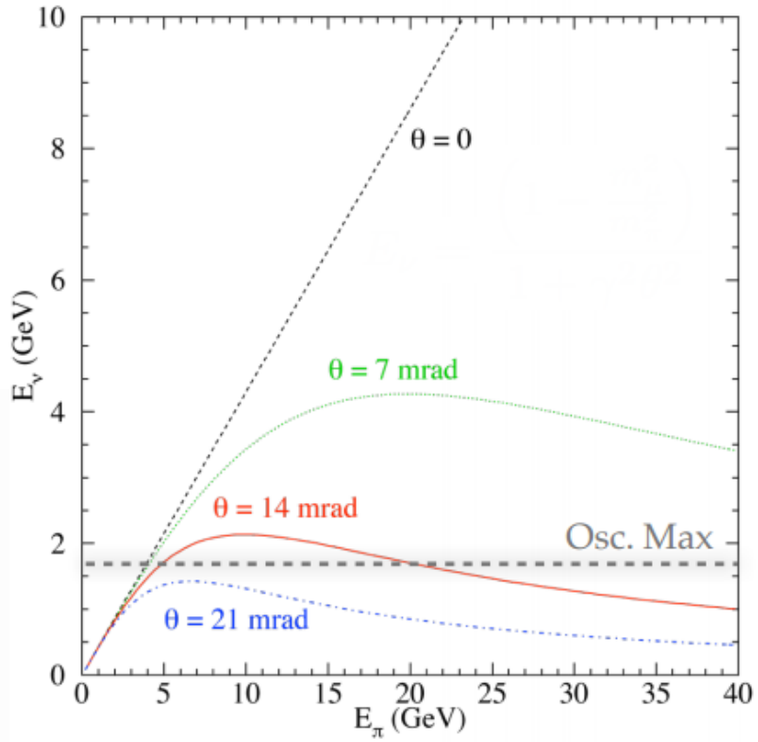


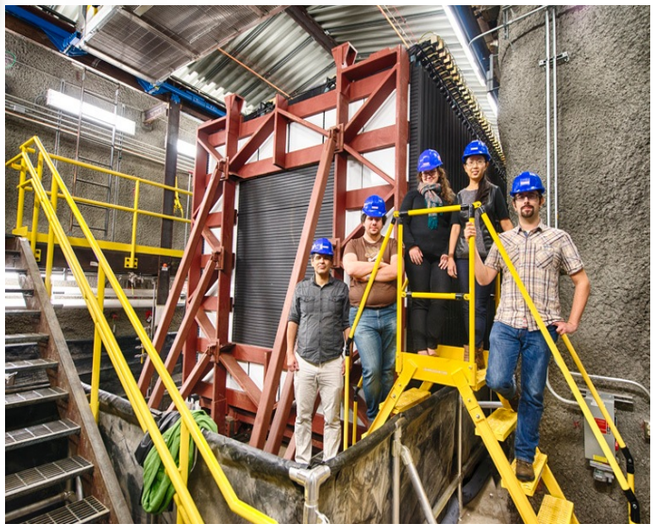
The NOvA Experiment

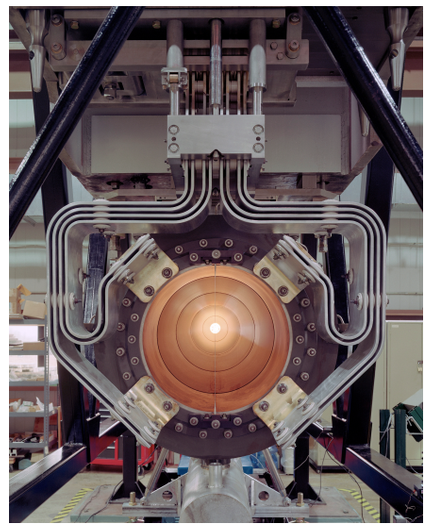




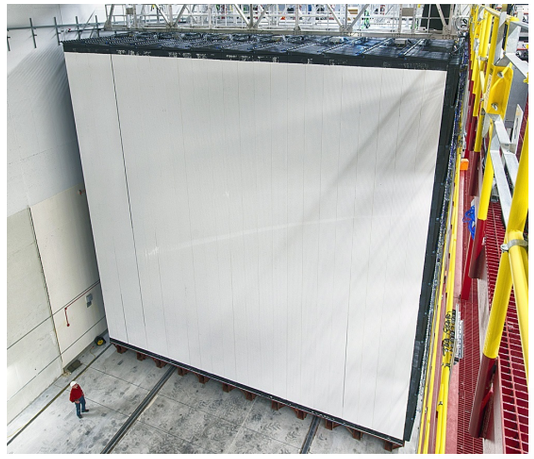


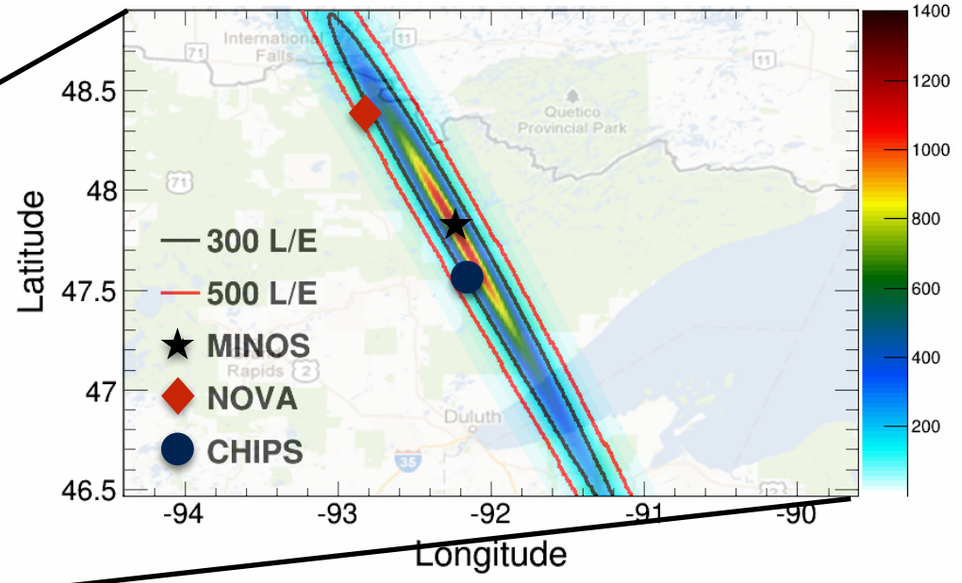


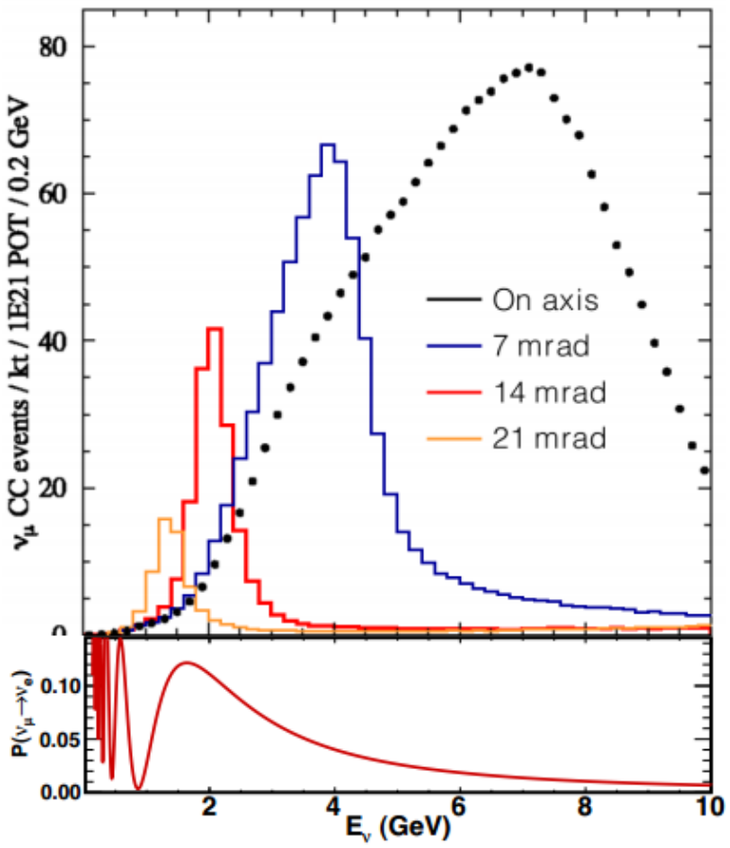


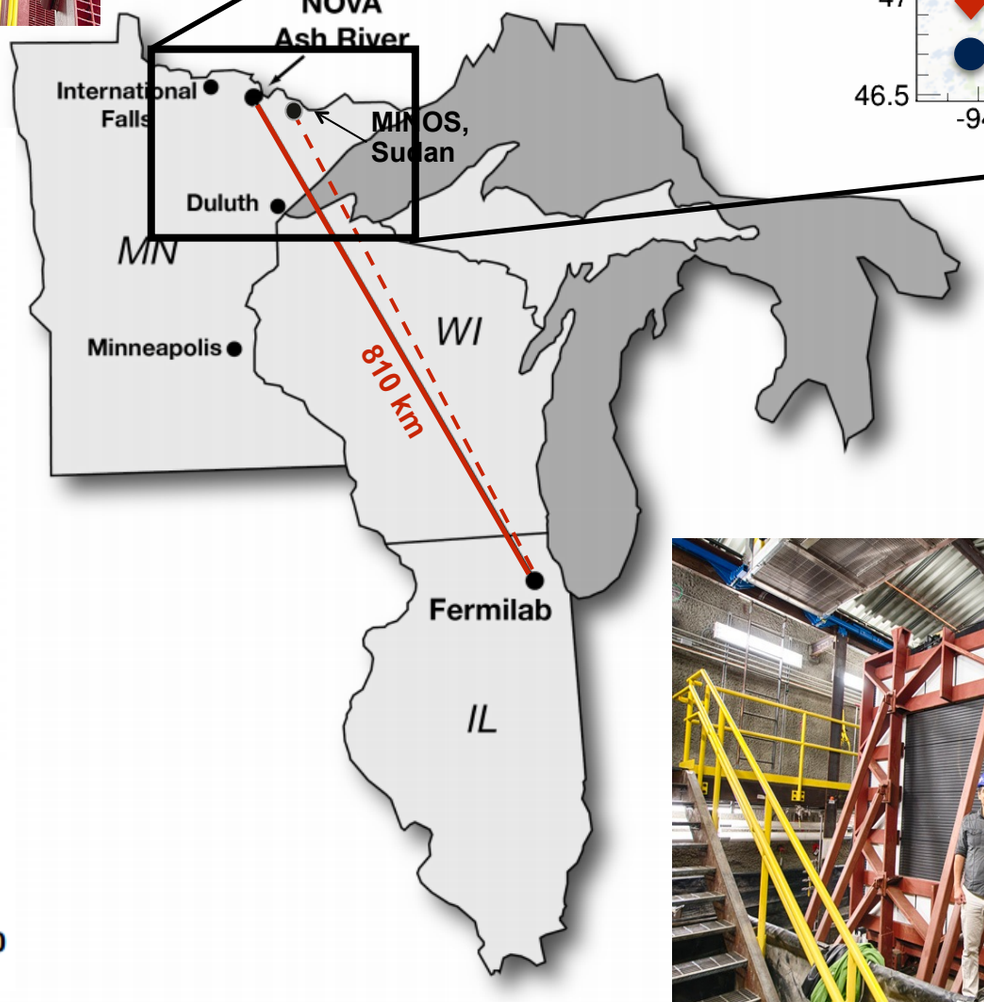


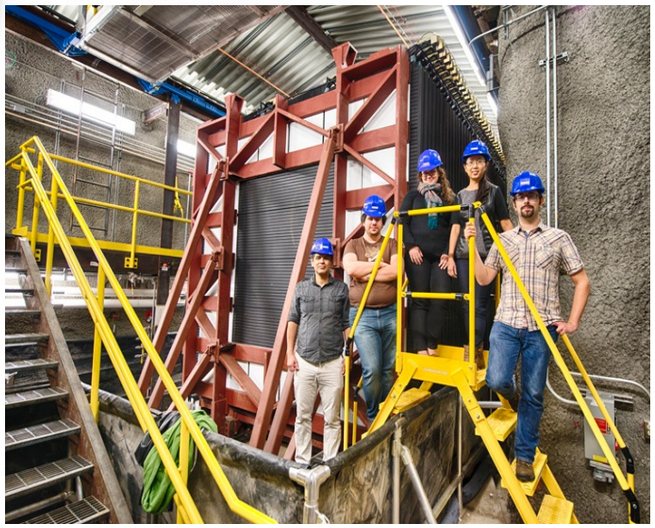
The NOvA Experiment

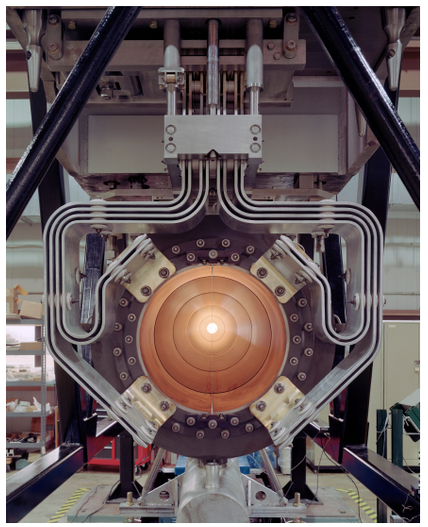




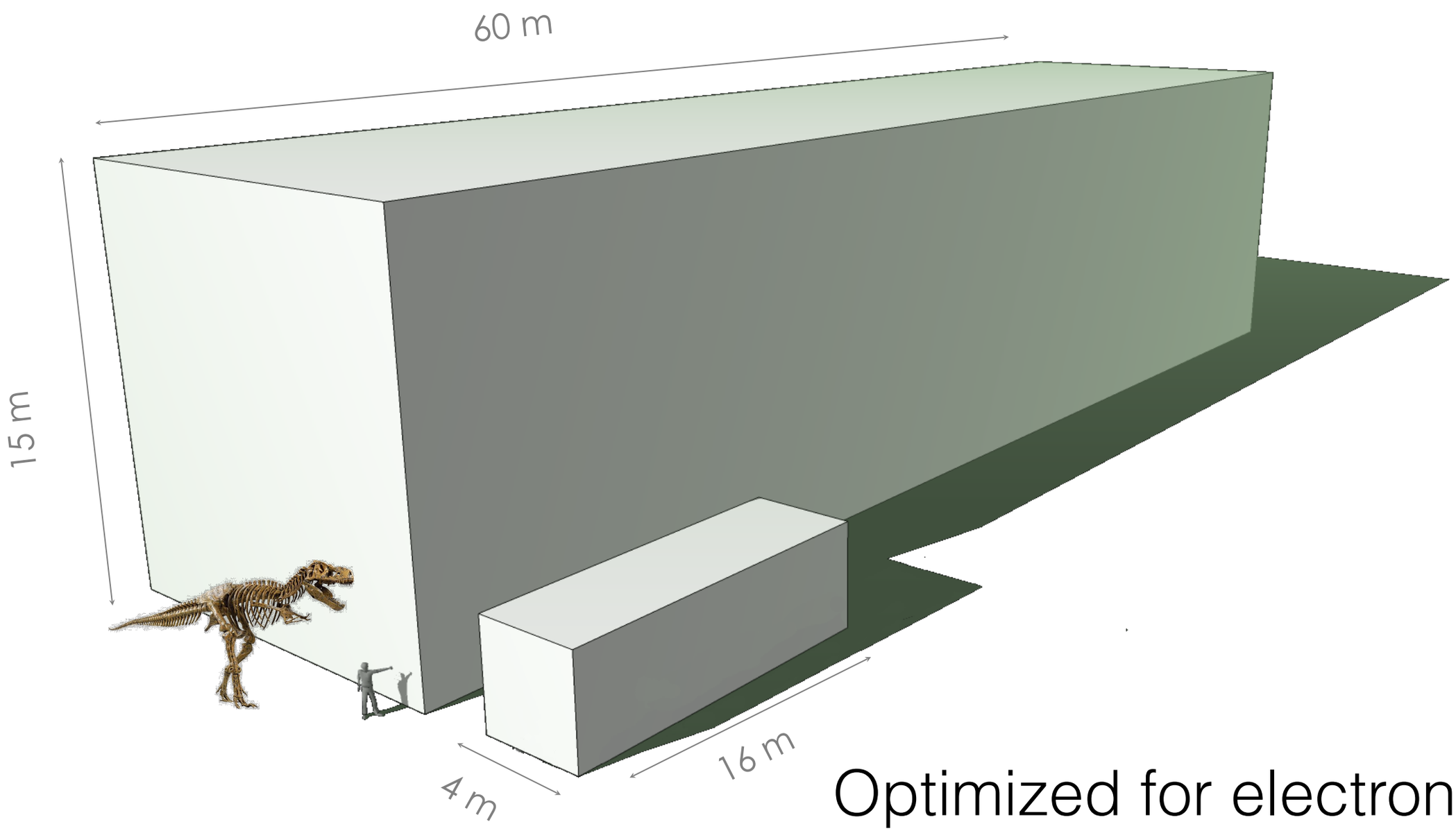






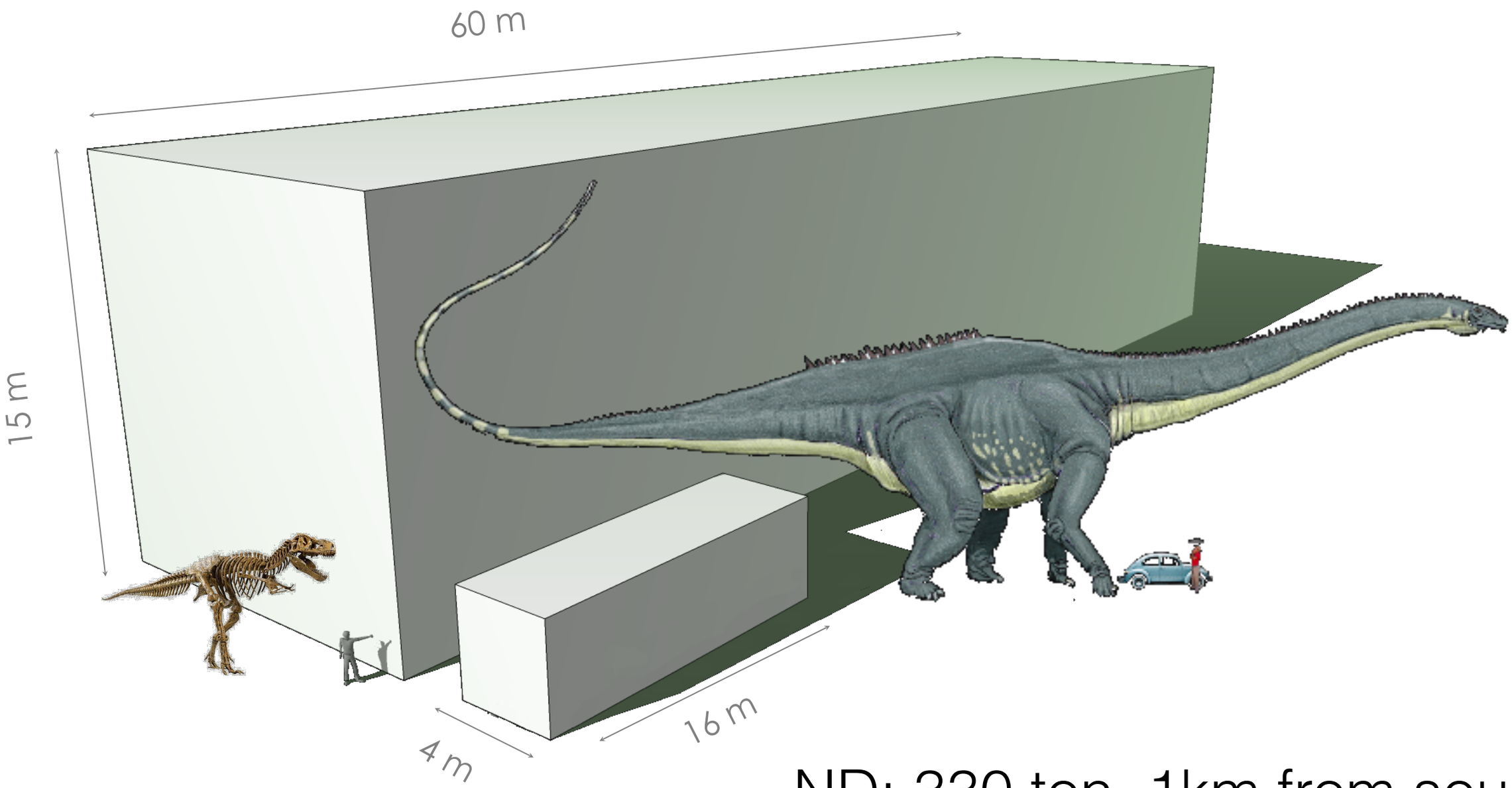


The NOvA Detectors



Optimized for electron ID, fine segmentation, Low-Z, and 65% Active

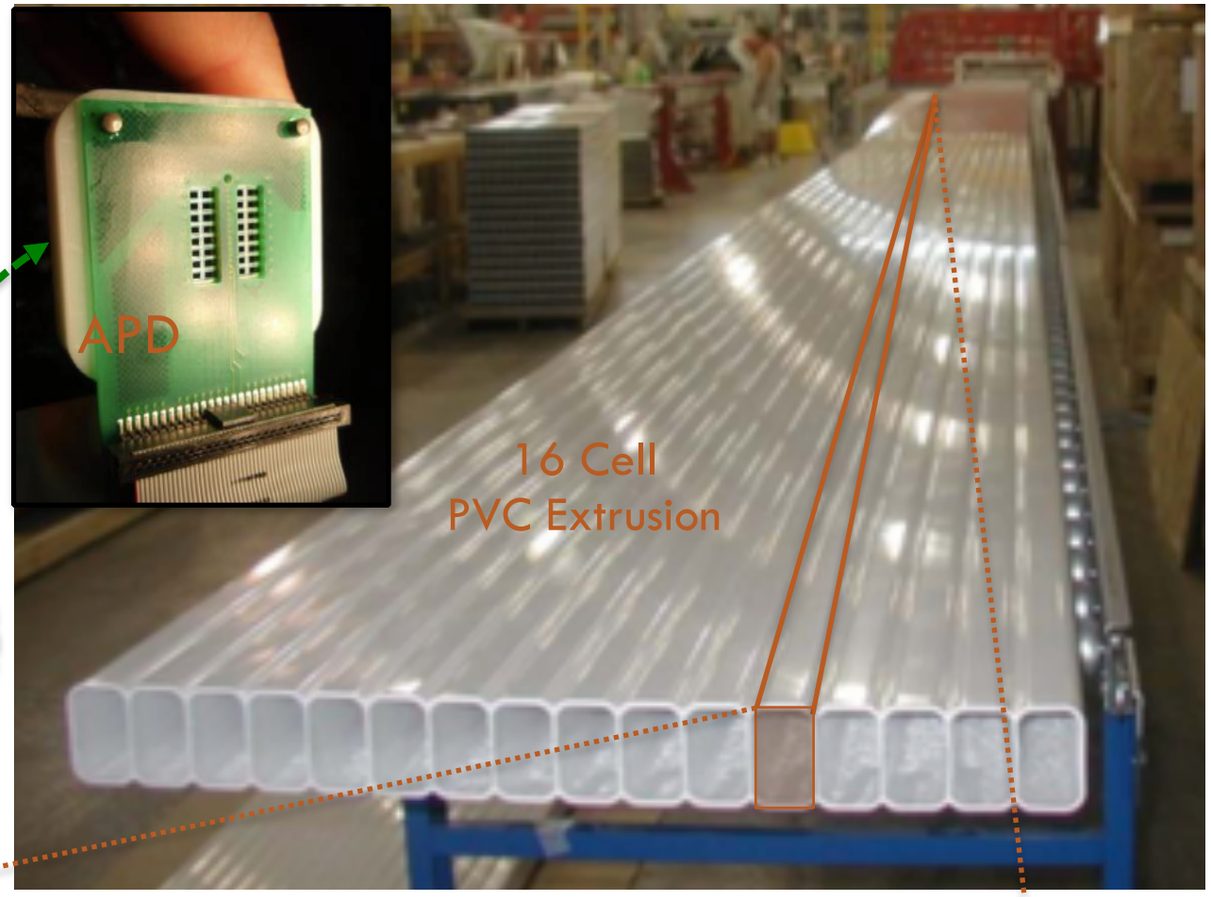
The NOvA Detectors

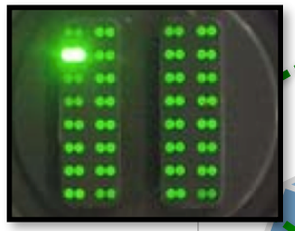


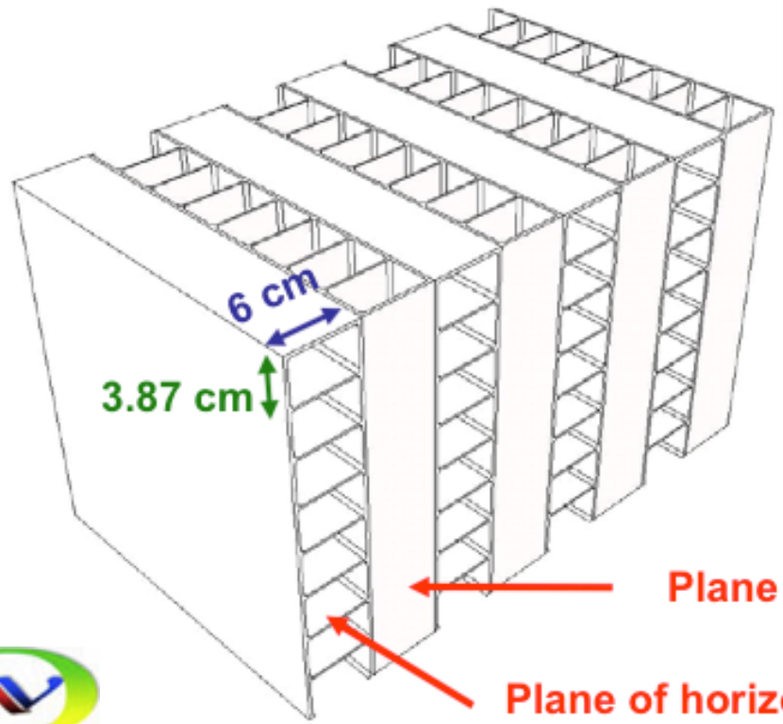
ND: 330 ton, 1km from source
FD: 14 kton, 810km from source

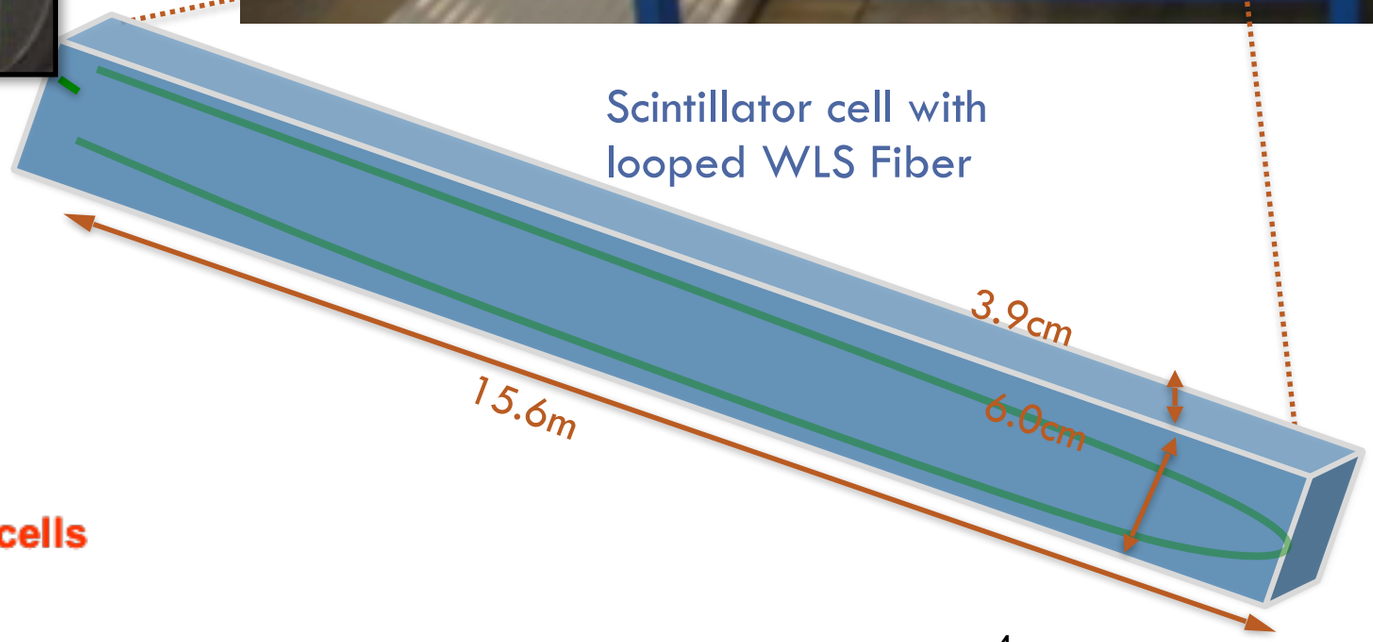
Detector Technology

- PVC extrusion + Liquid Scintillator
  - mineral oil + 5% pseudocumene
- Read out via WLS fiber to APD
  - FD has 344,064 channels
  - muon crossing far end~25 PE
- Layered planes of orthogonal views

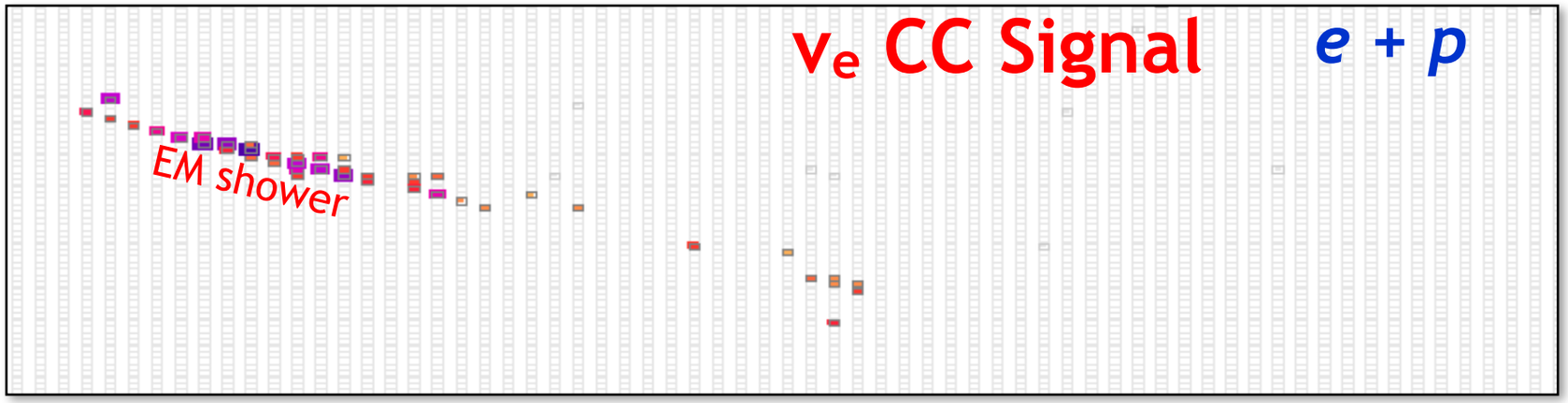


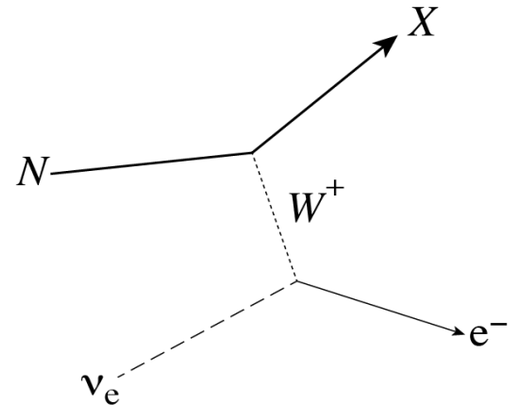


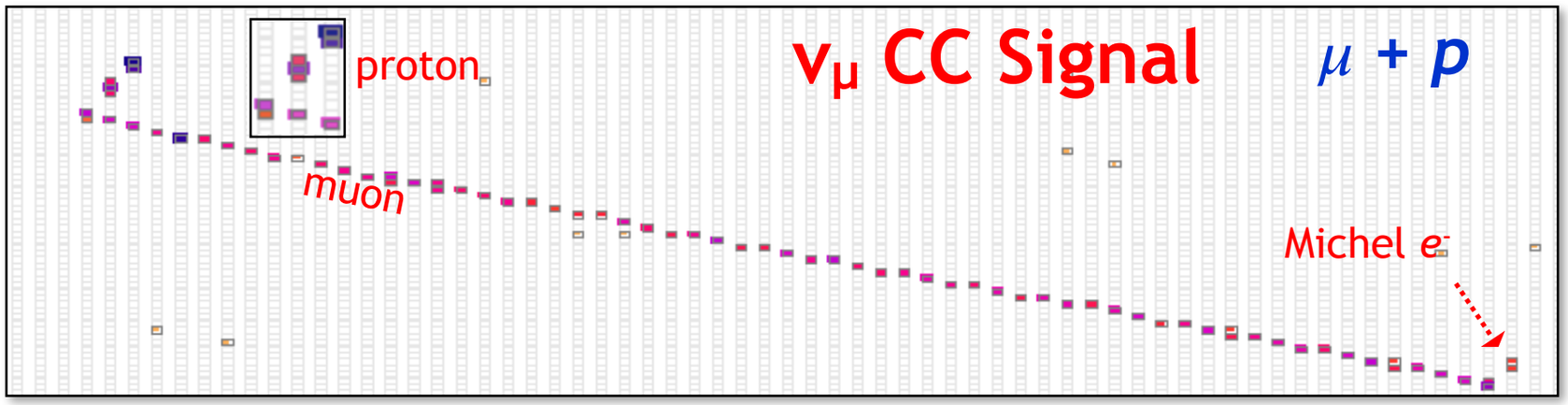


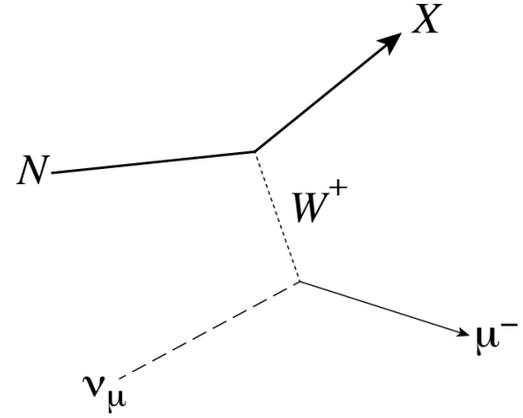


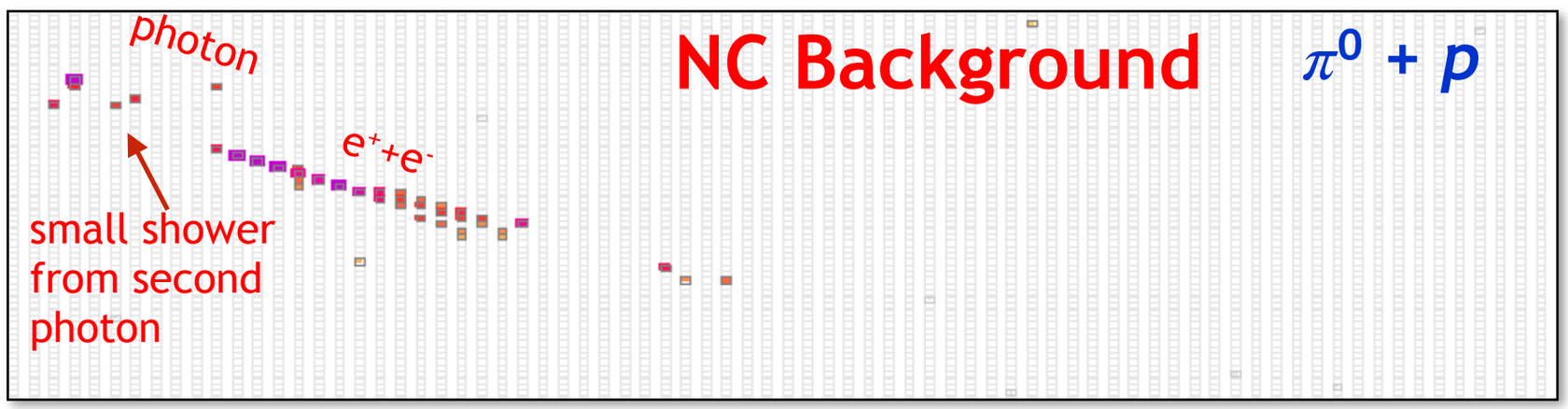
NOvA Event Topologies

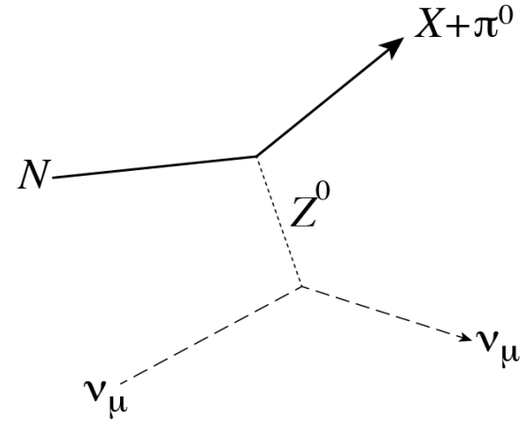






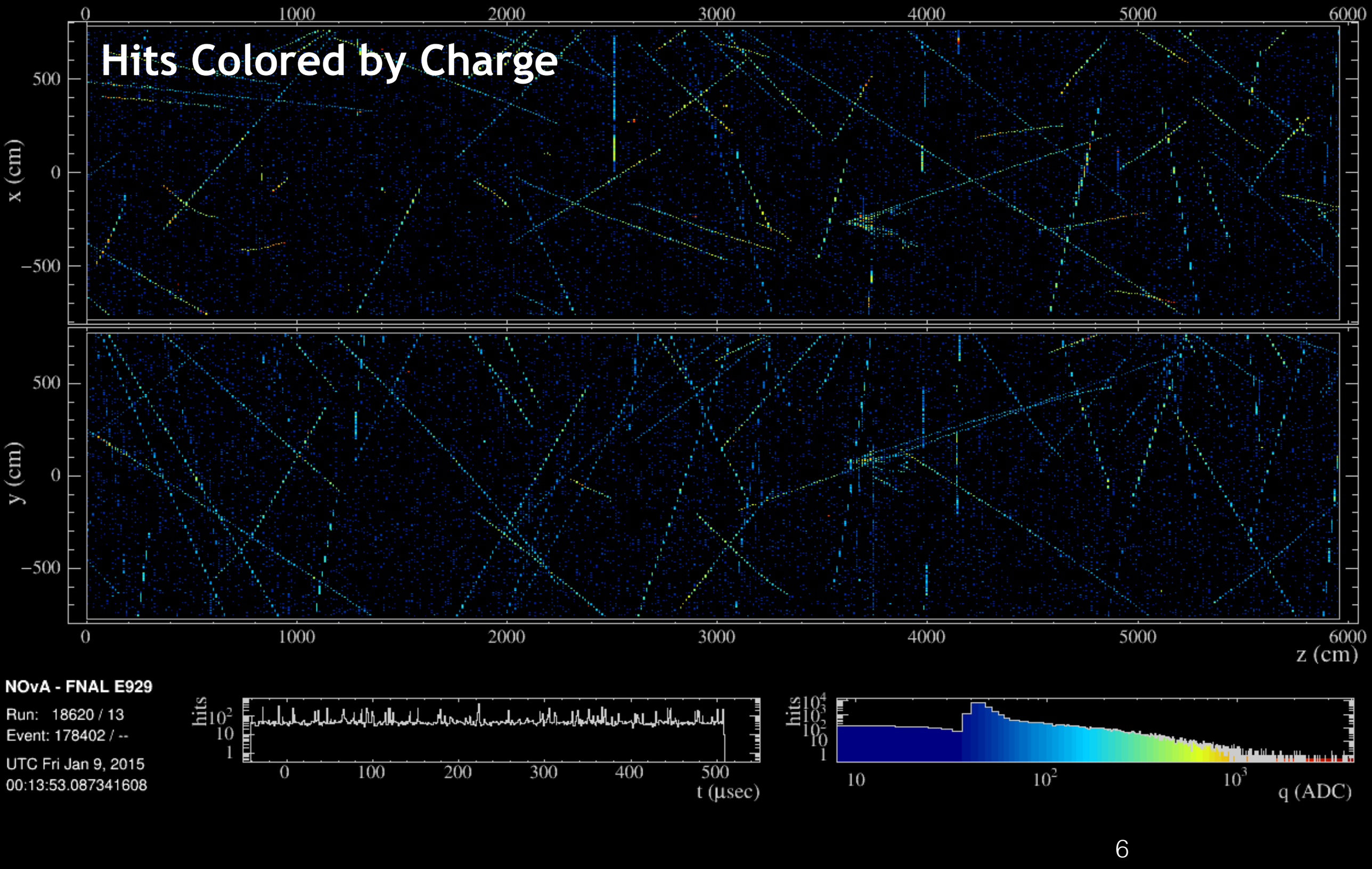




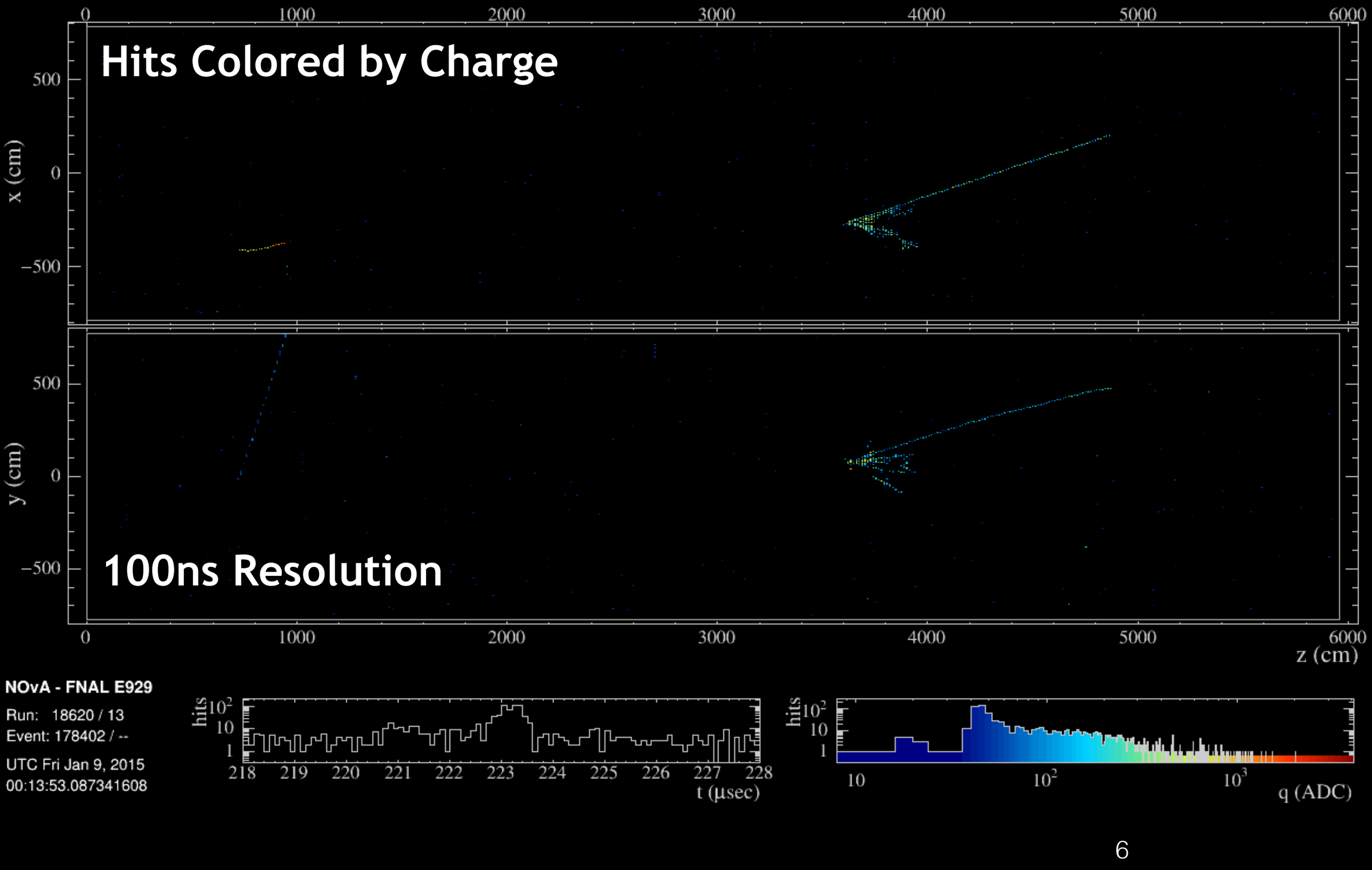


1 radiation length = 38cm (6 cell depths, 10 cell widths)

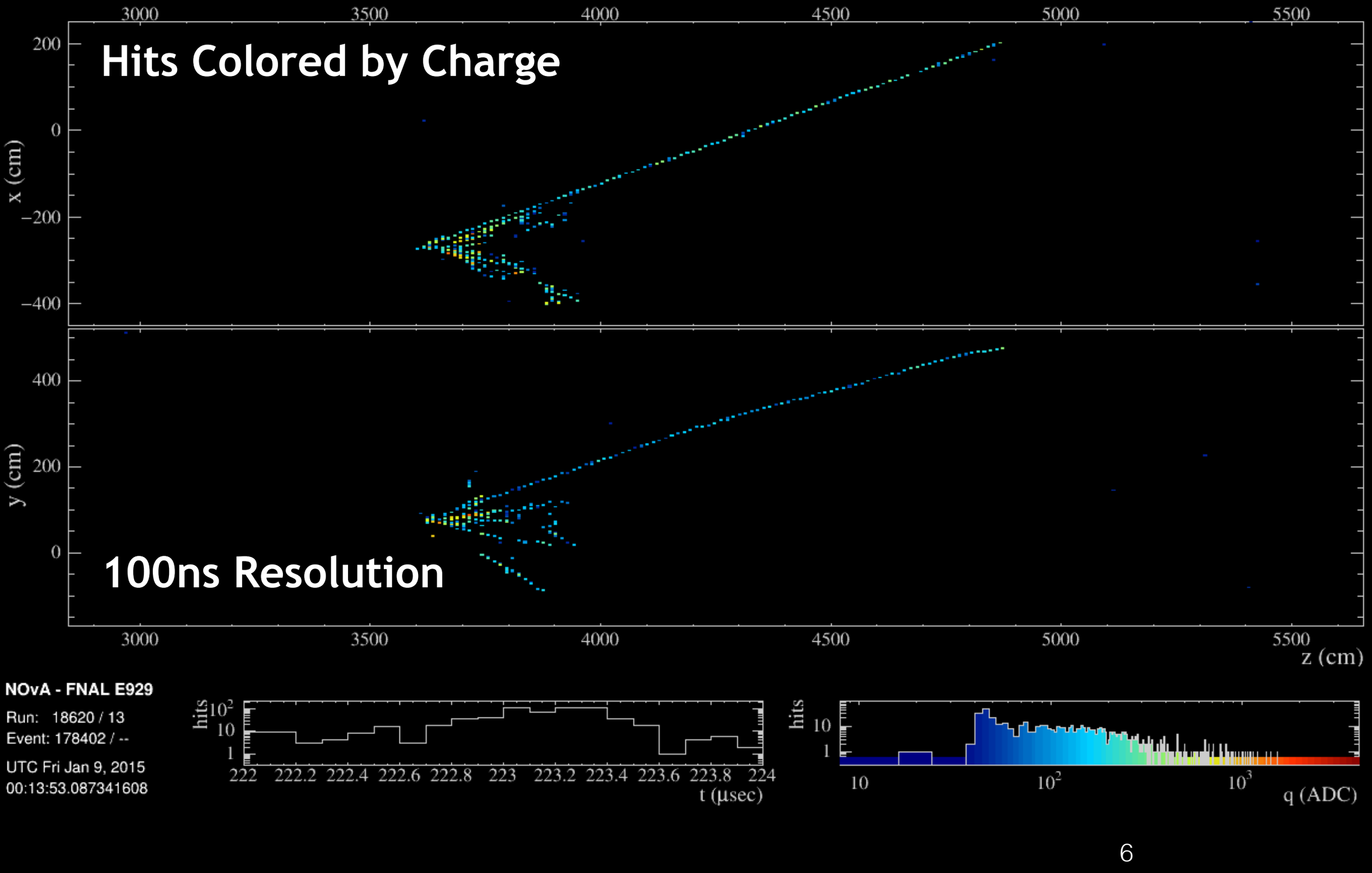
550 μs exposure of the Far Detector



Time-zoom on 10 μs interval during NuMI beam pulse

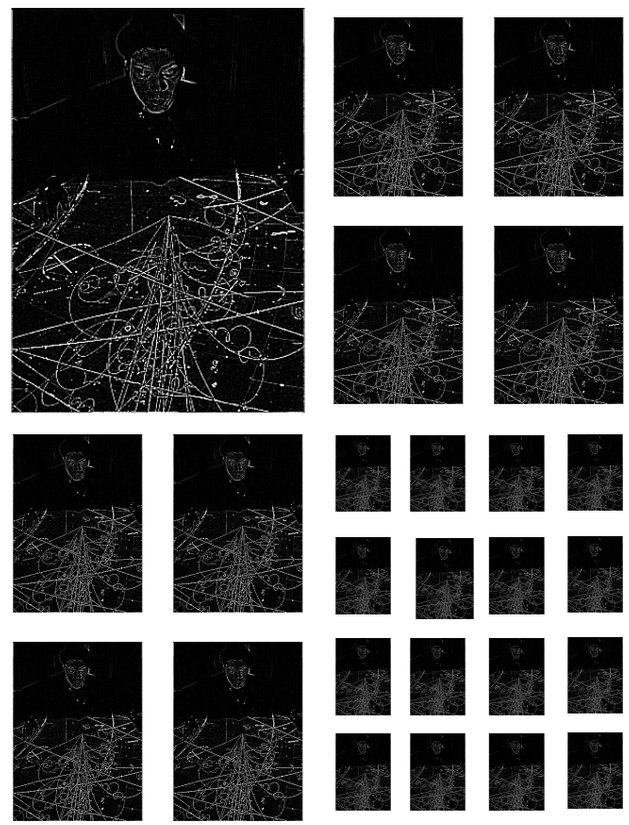


Close-up of neutrino interaction in the Far Detector



Deep Learning

What if we use the tools of the deep learning and computer vision communities to try and classify events?

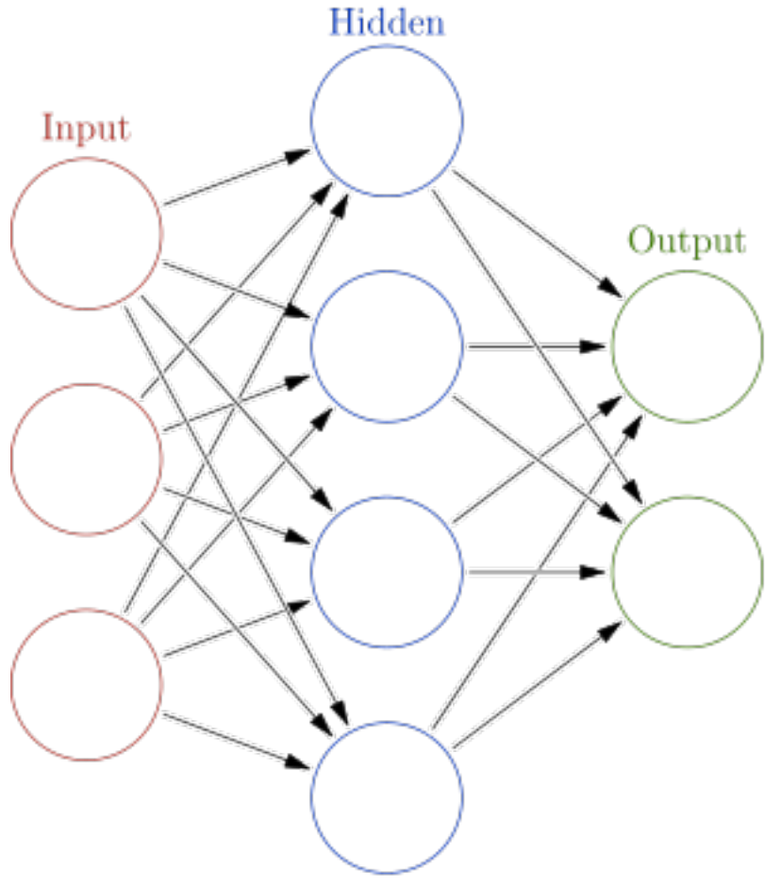


Ye Olde MLP

Neural networks are one of the most powerful machine learning tools available to us.

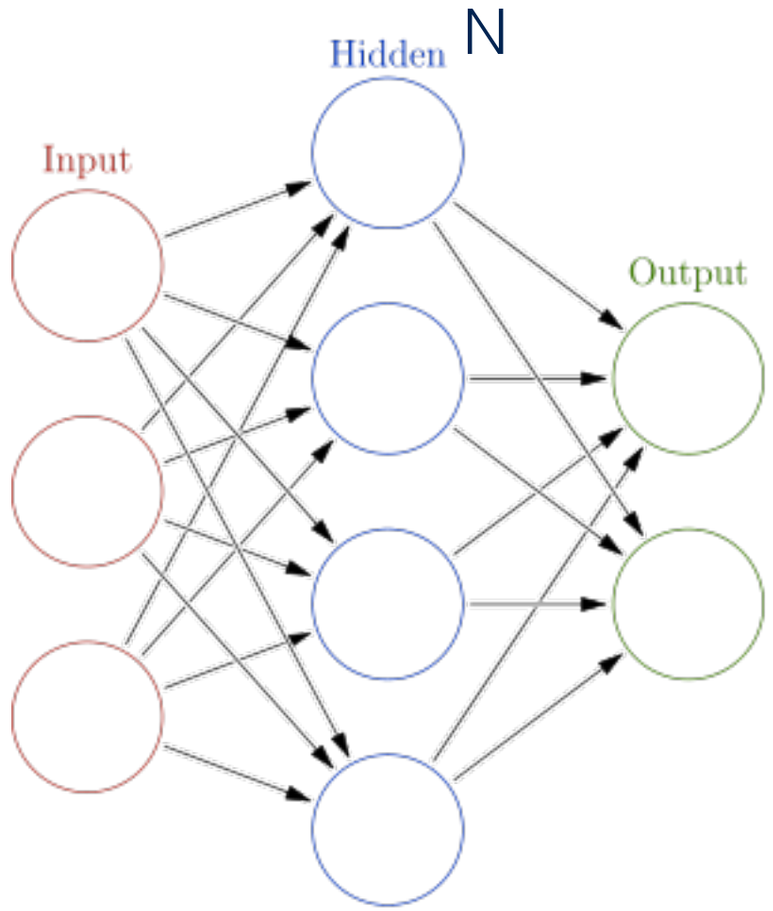
Hand extracted input variables are fed through a network of weighted nodes, optimized by iterative training, to finally give the output we've trained for- usually in HEP a binary classifier.

But why rely on human ingenuity and competence to reliably identify and extract the most powerful inputs?



DNN

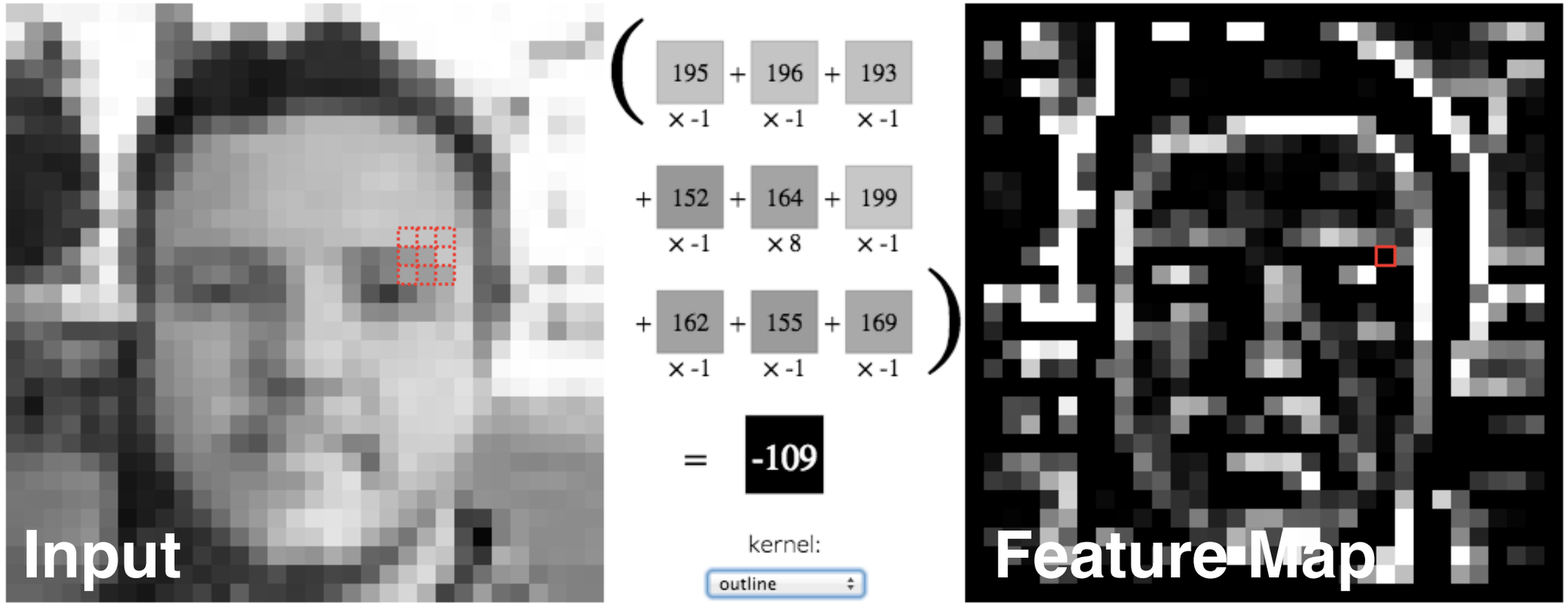
What if we instead try to keep all the input data? Why not rely on a wide, extremely Deep Neural Network (DNN) to learn the features it needs?



Three potential pitfalls:
-Prone to overtraining
-Expensive to run
-(Until recently) almost impossible to train

CNN

A form of DNN called a Convolutional Neural Network (CNN) provides a powerful and robust way to train feature extractions using a deep network of learned kernel operations followed by a conventional multilayer perceptron.

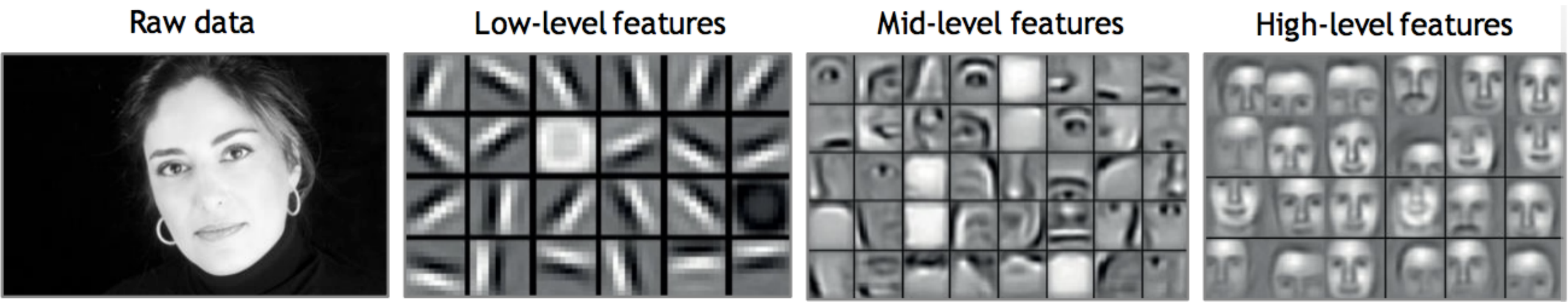


Kernel

http://setosa.io/ev/image-kernels/

CNN

These kernel operations, initial applied to our input “image” and then later to the output of previous kernel operations allow the network to learn how to extract increasingly abstract features.



https://developer.nvidia.com/deep-learning-courses

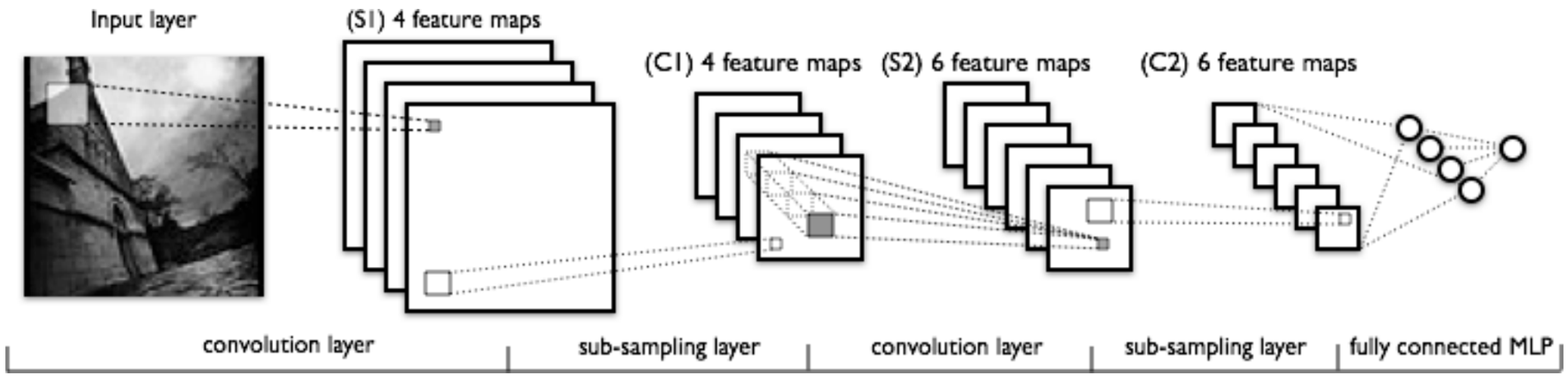
The LeNet

In its simplest form a convolutional neural network is a series of convolutional, sub-sampling, and MLP layers.

Each convolutional layer trains an array of NxWxH kernels to produce
a series of feature maps, where every kernel is applied across the entire image to form a single feature map.

Each subsampling layer uses “max-pooling” to shrink feature maps, keeping only the maximum value in a series of non overlapping rectangles.

A final conventional MLP uses the extracted features as its input.

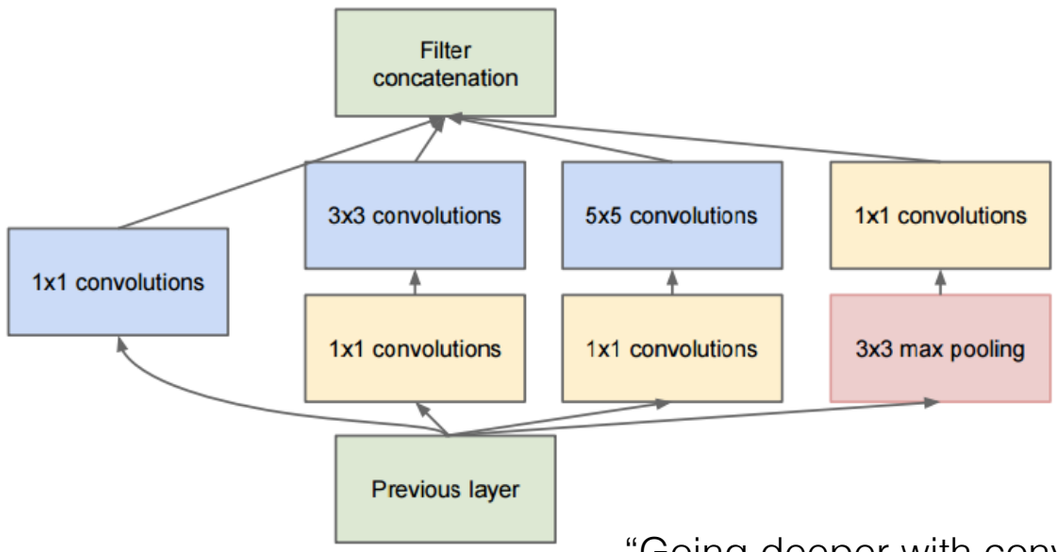


The “LeNet” circa 1989

http://deeplearning.net/tutorial/lenet.html http://yann.lecun.com/exdb/lenet/

Modern CNNs

Parallel processing in comparatively cheap GPUs, non saturating activation functions, and “big data” has lead to a renaissance in CNN use over the last few years, with increasingly complex network-in-network models that allow for deeper learning of more complex features.



“Going deeper with convolutions” arXiv:1409.4842

The brilliance of this inception module is that it uses kernels of several sizes but keeps the number of feature maps under control by use of a 1x1 convolution.

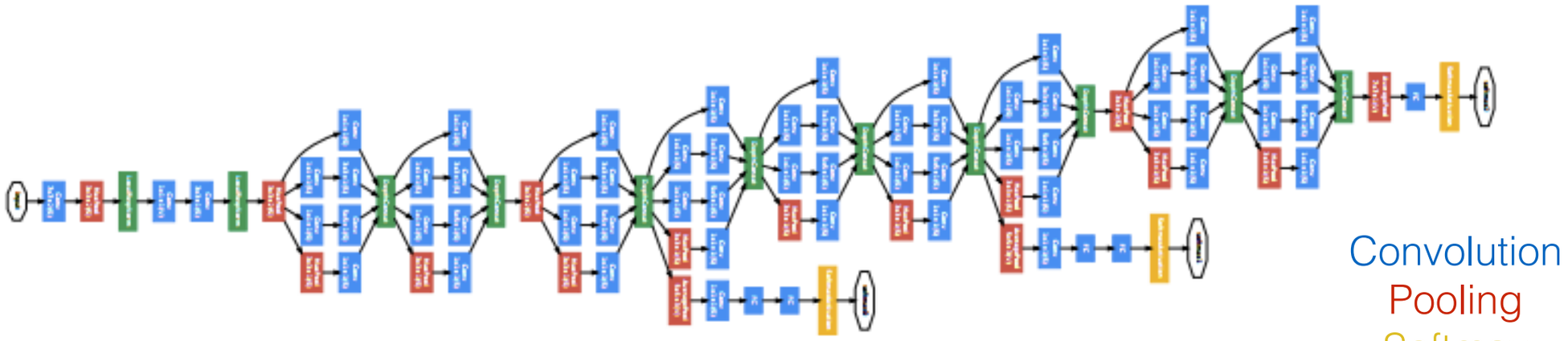
Modern CNNs

Parallel processing in comparatively cheap GPUs, non saturating activation functions, and “big data” has lead to a renaissance in CNN use over the last few years, with increasingly complex network-in-network models that allow for deeper learning of more complex features.

The brilliance of this inception module is that it uses kernels of several sizes but keeps the number of feature maps under control by use of a 1x1 convolution.

Modern CNNs

Parallel processing in comparatively cheap GPUs, non saturating activation functions, and “big data” has lead to a renaissance in CNN use over the last few years, with increasingly complex network-in-network models that allow for deeper learning of more complex features.

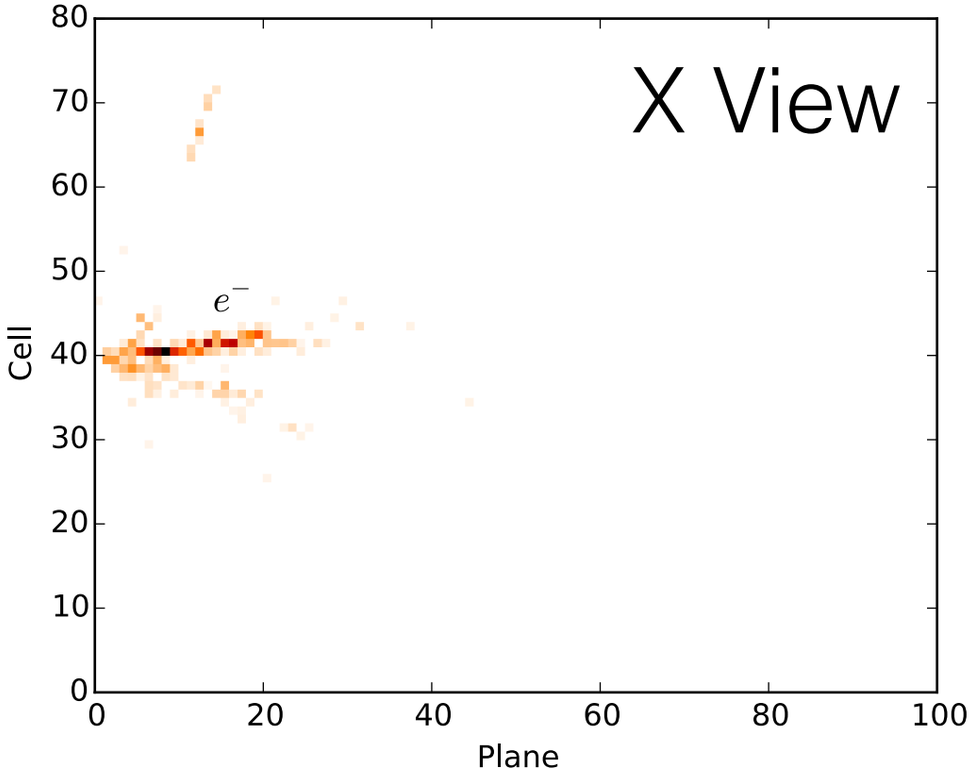


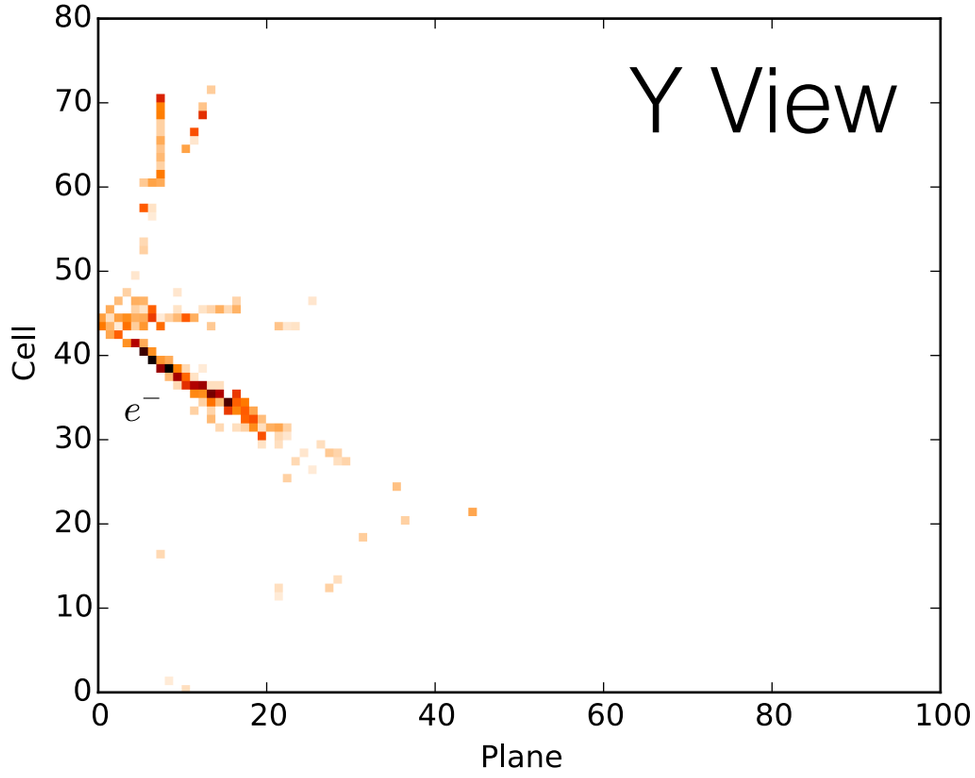
The “GoogleNet” circa 2014

The brilliance of this inception module is that it uses kernels of several sizes but keeps the number of feature maps under control by use of a 1x1 convolution.

Our Input

Our input “image”, is a pair of maps of the hits in a tight space/time window. One for the X view and another for the Y view. Each “pixel” is the calibrated energy response in that cell, rescaled to fit in a uint8. All “images” are the same dimension- 100 planes by 80 cells.





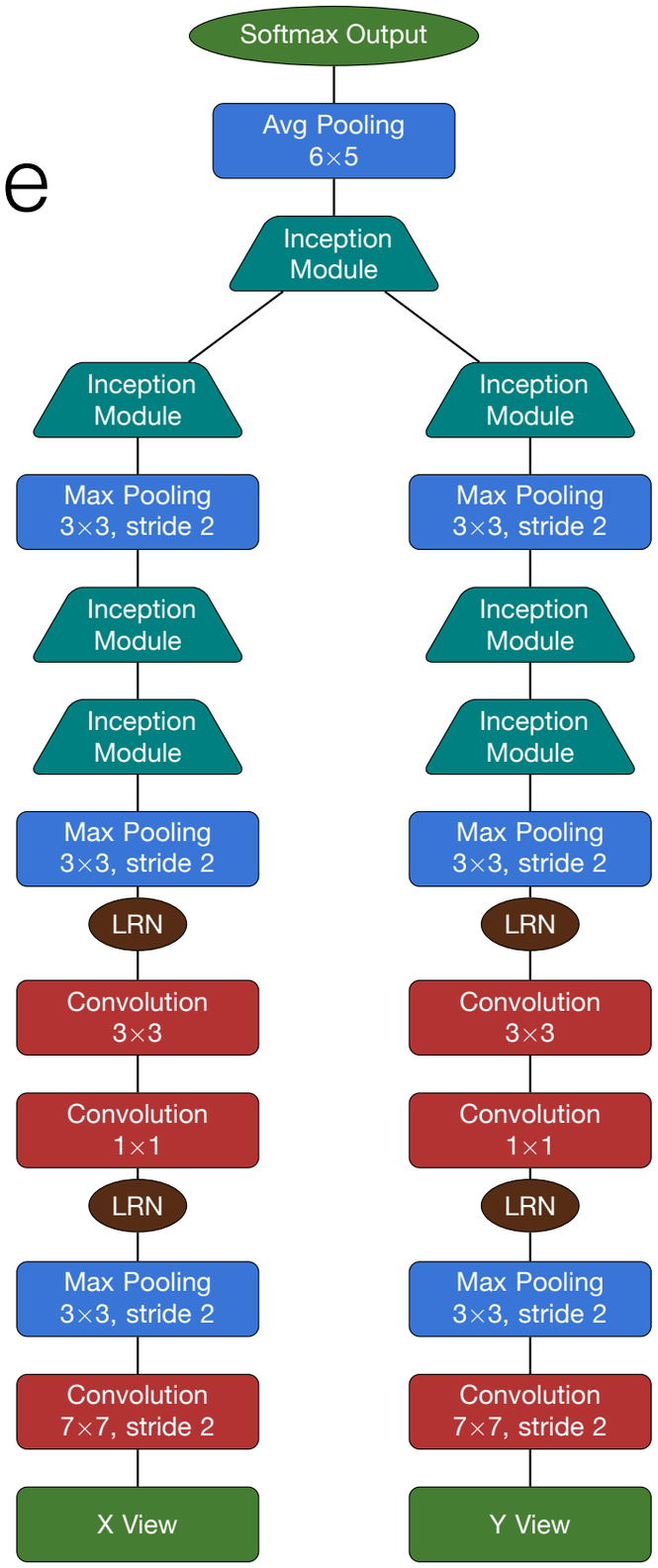
Our Architecture

Based on the first googlenet. Largest innovation is splitting each view into a separate sequence of layers and concatenating the outputs near the end of the network.

The architecture attempts to categorize events as $\{\nu_\mu, \nu_e, \nu_\tau\} \times \{QE,RES,DIS\}$ or cosmic rays.

Designed to be a universal classifier (rather than ν_e only).

Built in the excellent CAFFE framework: http://caffe.berkeleyvision.org/



But What About ART?

NOvA is an ART centered experiment with the main sequence of generation/reconstruction framework build with ART and around ART products.

By using the c++ based CAFFE it was very easy to pull in the libraries we need to classify our events in the normal “production chain” at NOvA.

https://cdcvs.fnal.gov/redmine/projects/novaart/repository/show/trunk/CVN

Evaluation:
https://cdcvs.fnal.gov/redmine/projects/novaart/repository/changes/trunk/CVN/art/CVNEvaluator_module.cc
https://cdcvs.fnal.gov/redmine/projects/novaart/repository/entry/trunk/CVN/art/CVNEvaluator.fcl

Exporting to leveldb:
https://cdcvs.fnal.gov/redmine/projects/novaart/repository/entry/trunk/CVN/standalone/cvnCreateLevelDB.cc

But What About ART?

CAFFE makes it very easy to train on an accelerated machine before running inference on a regular CPU.

ART made it easy to export from ART data products to read speed optimized databases like LevelDB and LMDB

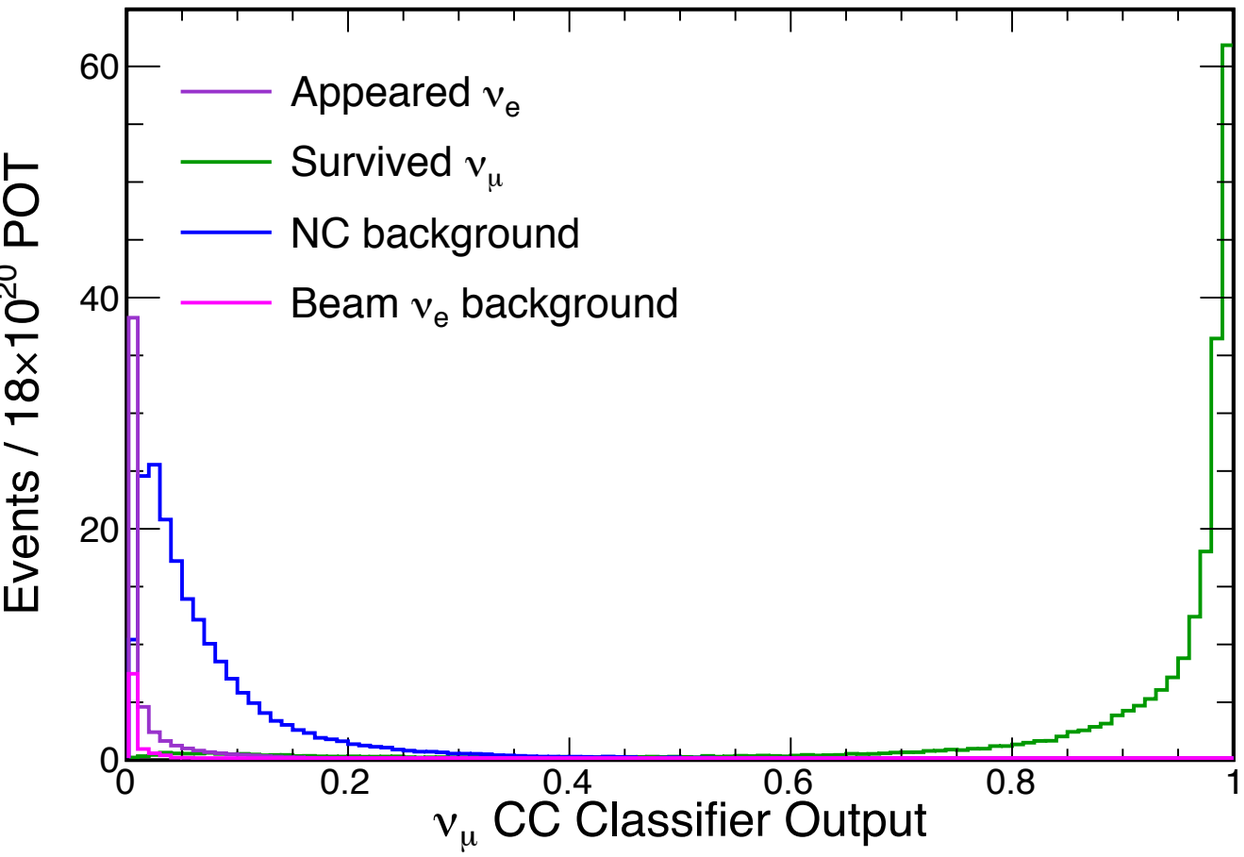
Match made in heaven. “Agile” network development on a gpu cluster followed by painless routine deployment on the grid.

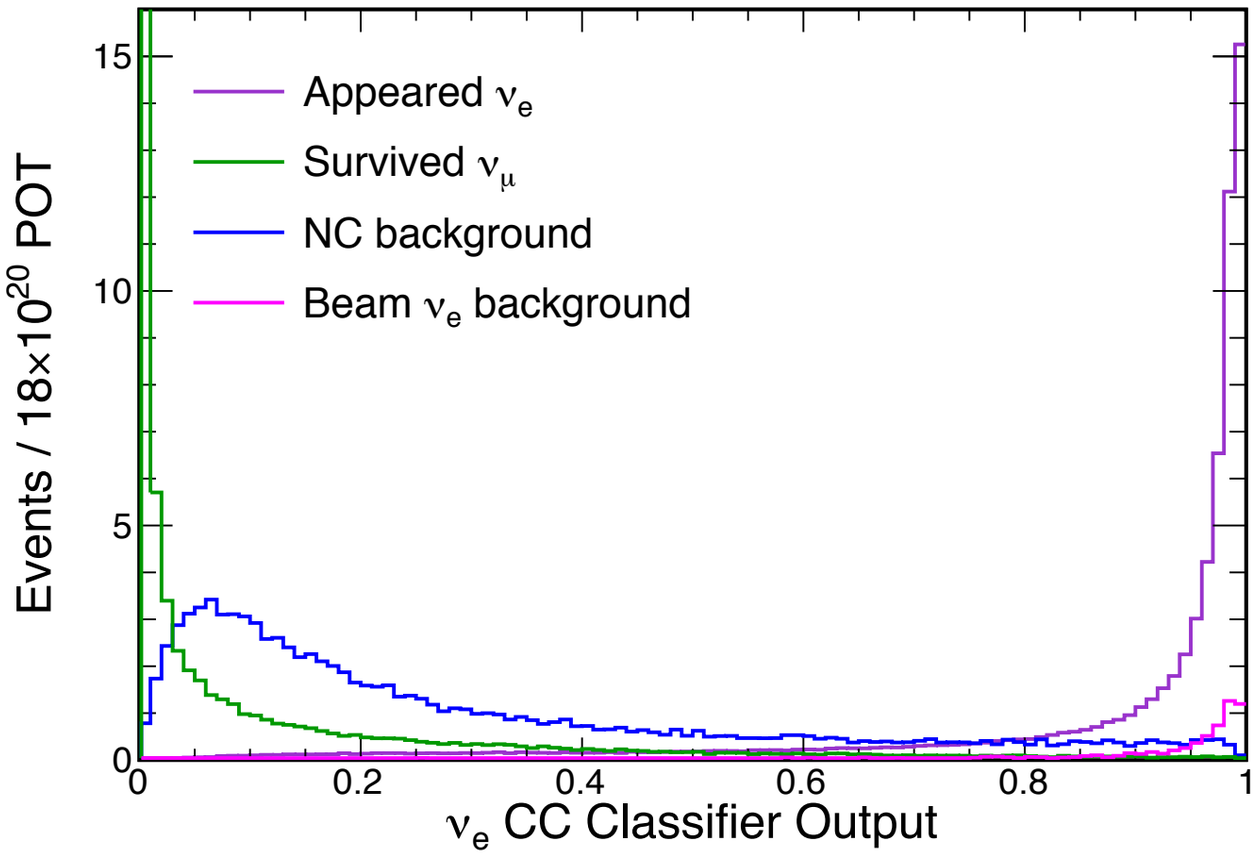
https://cdcvs.fnal.gov/redmine/projects/novaart/repository/show/trunk/CVN

Evaluation:
https://cdcvs.fnal.gov/redmine/projects/novaart/repository/changes/trunk/CVN/art/CVNEvaluator_module.cc
https://cdcvs.fnal.gov/redmine/projects/novaart/repository/entry/trunk/CVN/art/CVNEvaluator.fcl

Exporting to leveldb:
https://cdcvs.fnal.gov/redmine/projects/novaart/repository/entry/trunk/CVN/standalone/cvnCreateLevelDB.cc

The Bottom Line





Our CNN achieves 49% efficiency on ν_e selection compared to the previous the NOvA ν_e PIDs 35% efficiency at no loss of purity.

Conclusions

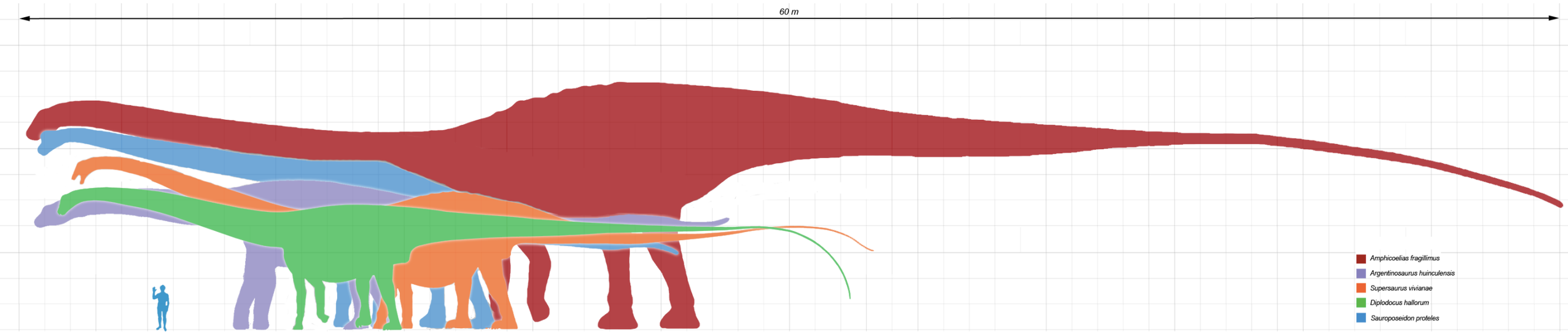
Just the tip of the iceberg! Huge amounts of room to optimize our classification network.

Why not try to identify small reconstructed objects like prongs?

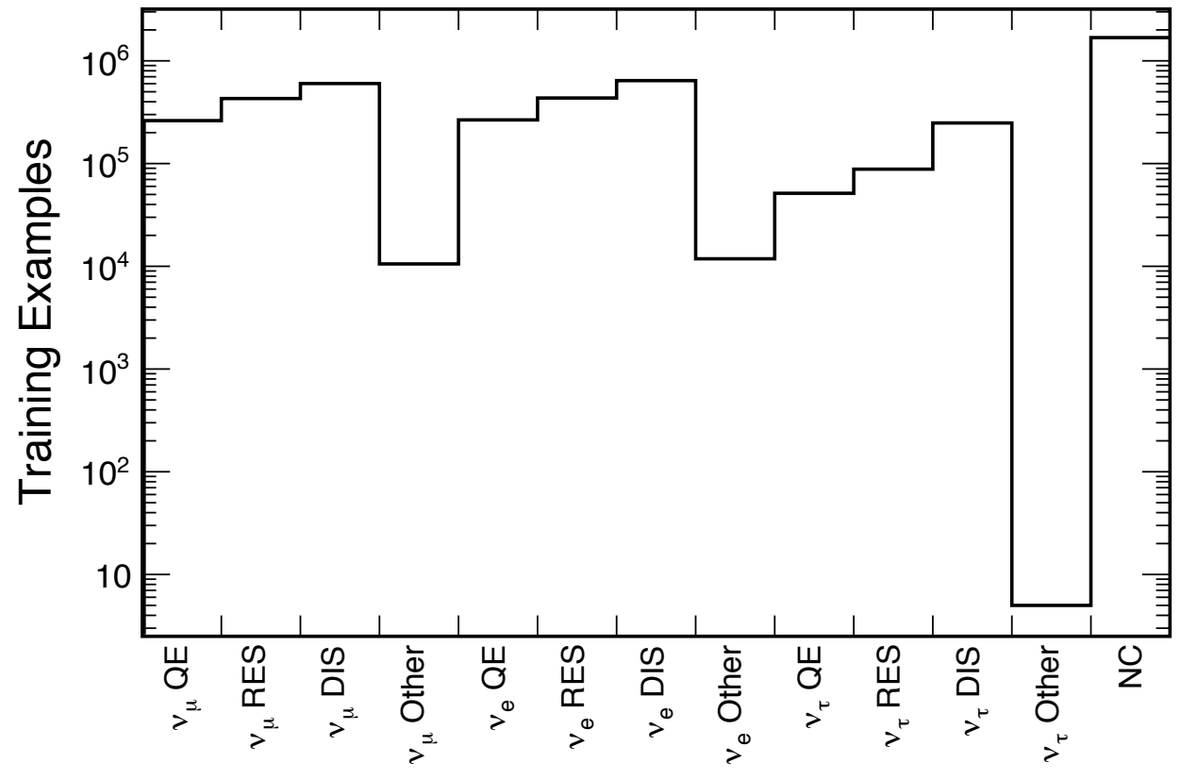
Or take advantage of semantic segmentation to classify *every cell* in a NOvA event.

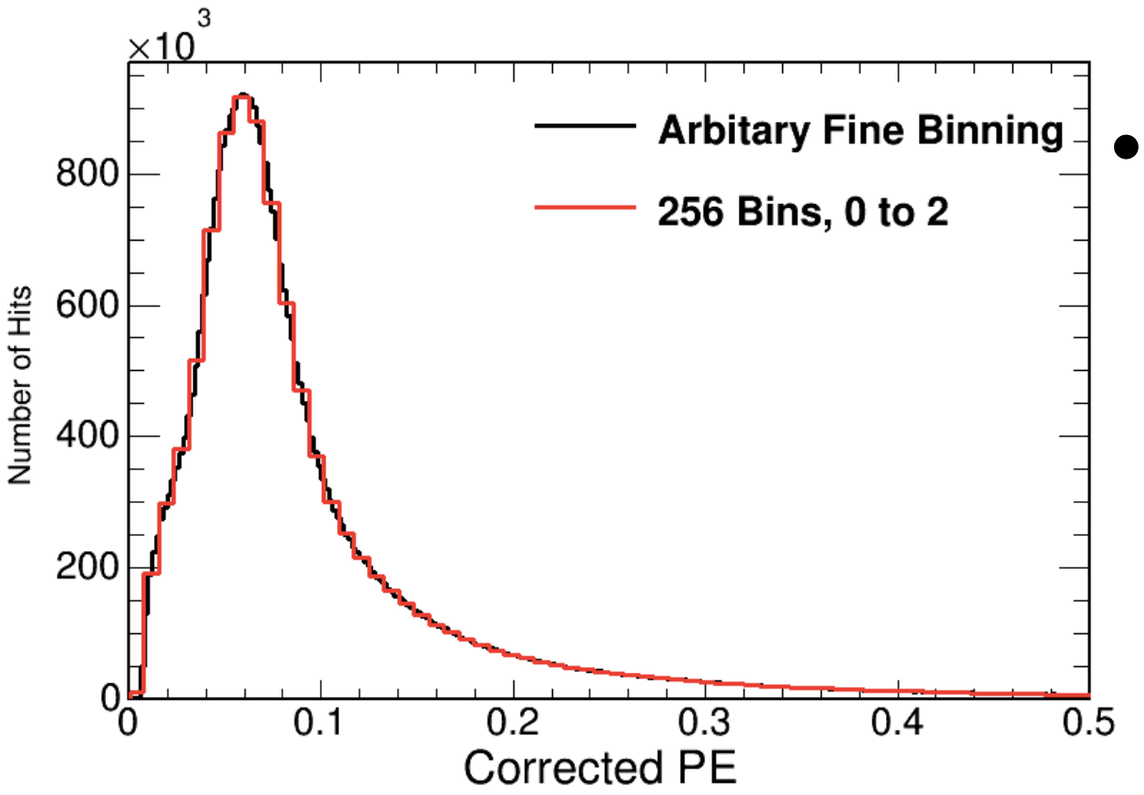
While you wait you should check out our recent paper:
“A Convolutional Neural Network Neutrino Event Classifier”
A. Aurisano, A. Radovic, D. Rocco et al
https://arxiv.org/abs/1604.01444

Did Dinosaurs Really Come That Large?



Training Sample





- Push into LevelDB databases: 80% for training and 20% for testing.
- Rescale calibrated energy depositions to go from 0 to 255 and truncate to chars for dramatically reduced file size at no loss of information
- Initial training attempted to classify for a number of interaction modes:
  - $\{\nu_\mu, \nu_e, \nu_\tau\} \times \{QE,RES,DIS\}$ +NC

Regularization & Data Augmentation

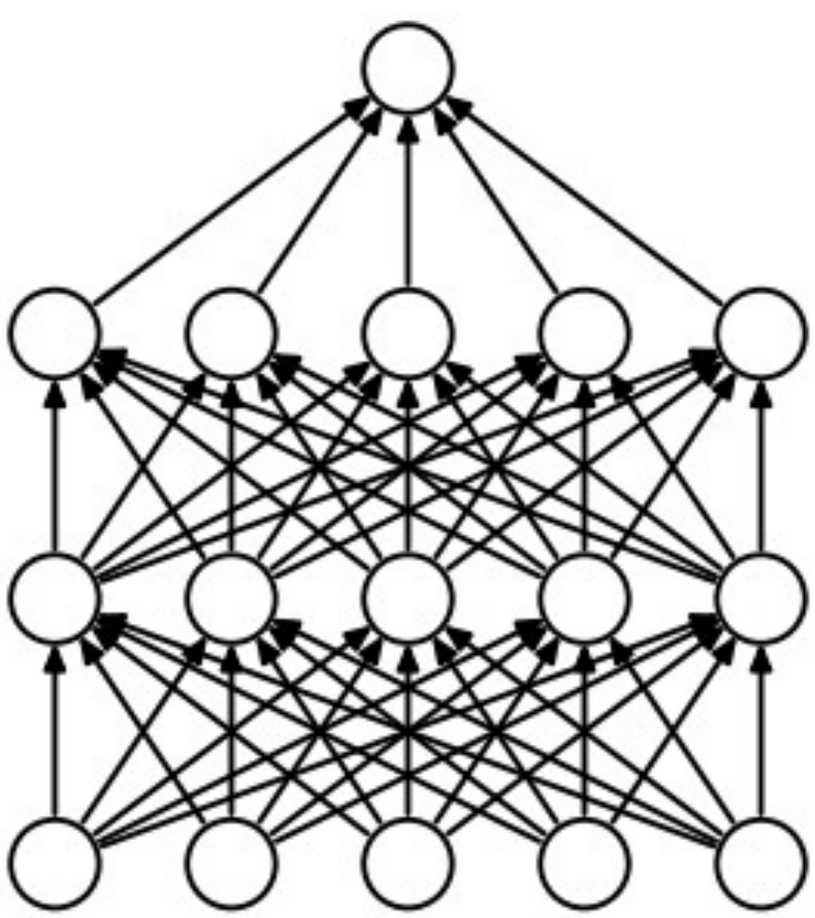
- Most common anti-overtraining tool in machine learning, add a penalty term of the form:

$$\frac{1}{2}\lambda \sum w_i^2$$

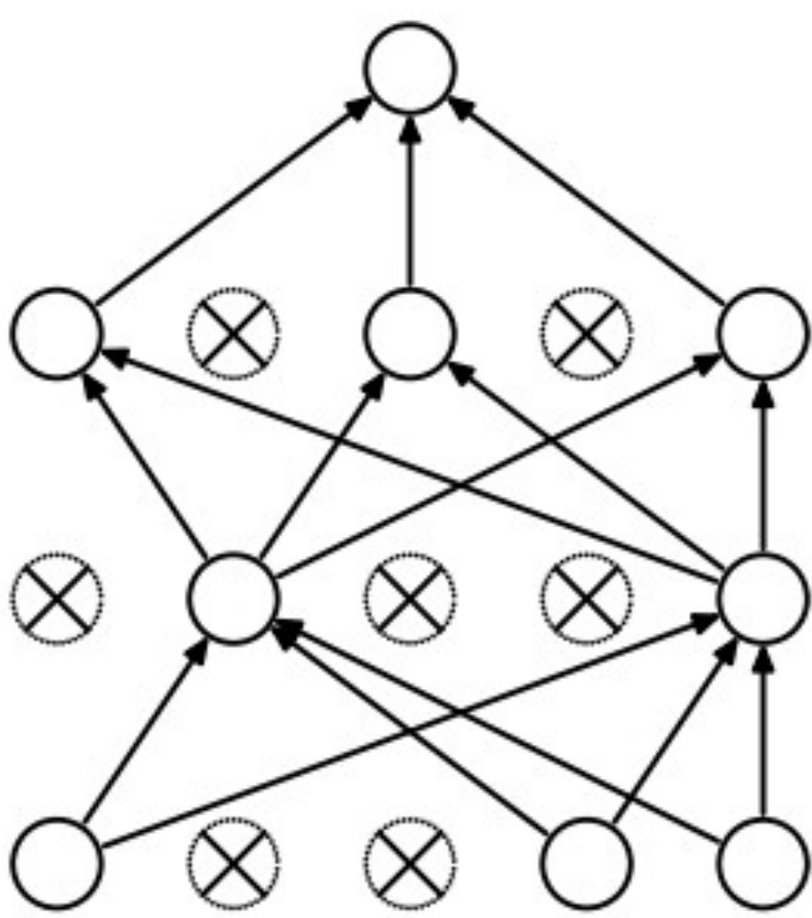
  to our training such that large weights are penalized.
- Simple but highly effective at penalizing fine tuning of weights.
- Perhaps the most effective way to fight overtraining is to simple have more data, in the absence of an infinitely large dataset however you can augment you input images to artificially increase the variety of the dataset
- We make use of image mirroring and random pixel shuffling during our training

Dropout

- Same goal as conventional regularization- prevent overtraining.
- Works by randomly removing whole nodes during training iterations. At each iteration, randomly set XX% of weights to zero and scale the rest up by 1/(1 – 0.XX).
- Forces the network not to build complex interdependencies in the extracted features.



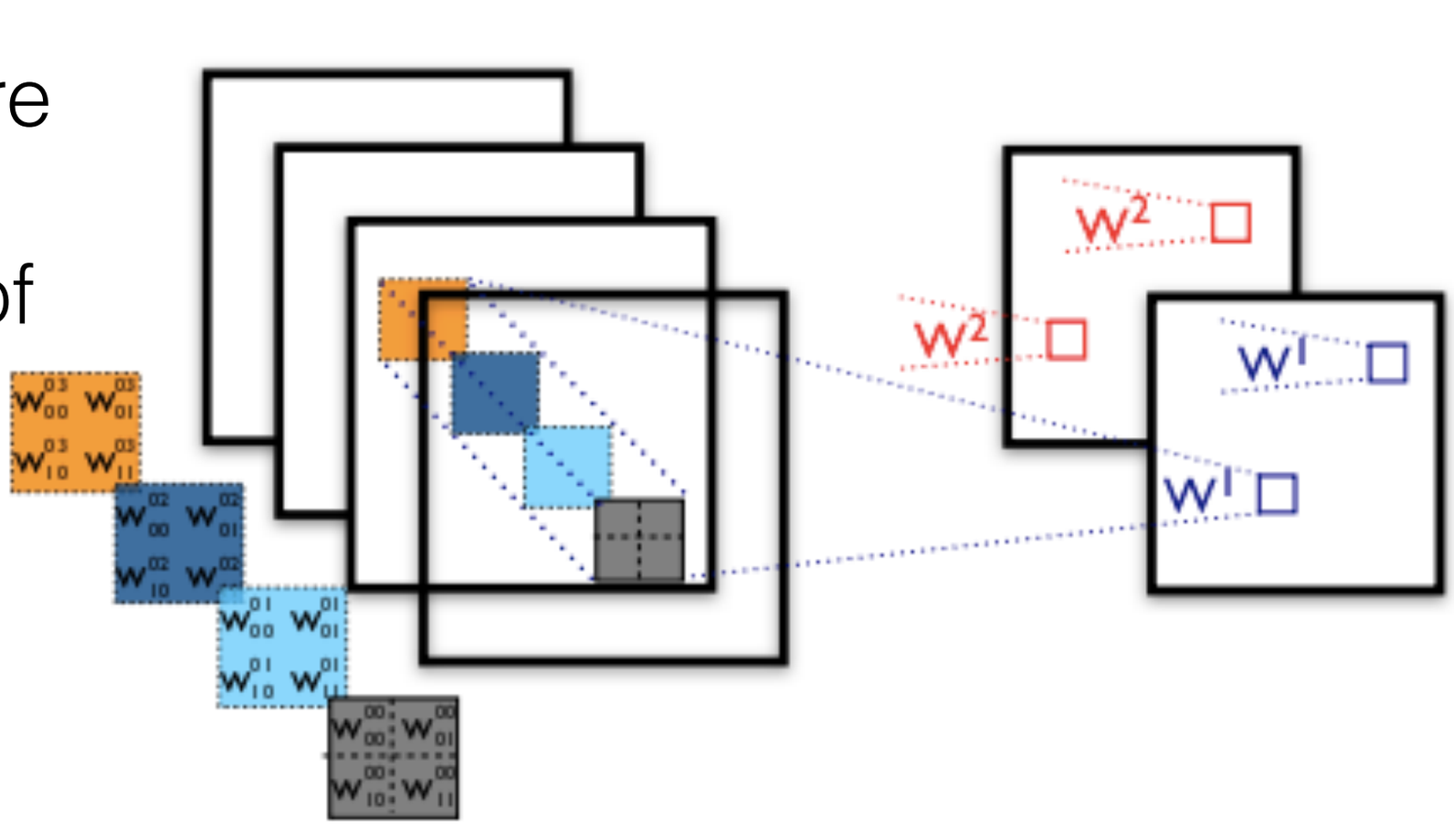
(a) Standard Neural Net



(b) After applying dropout.

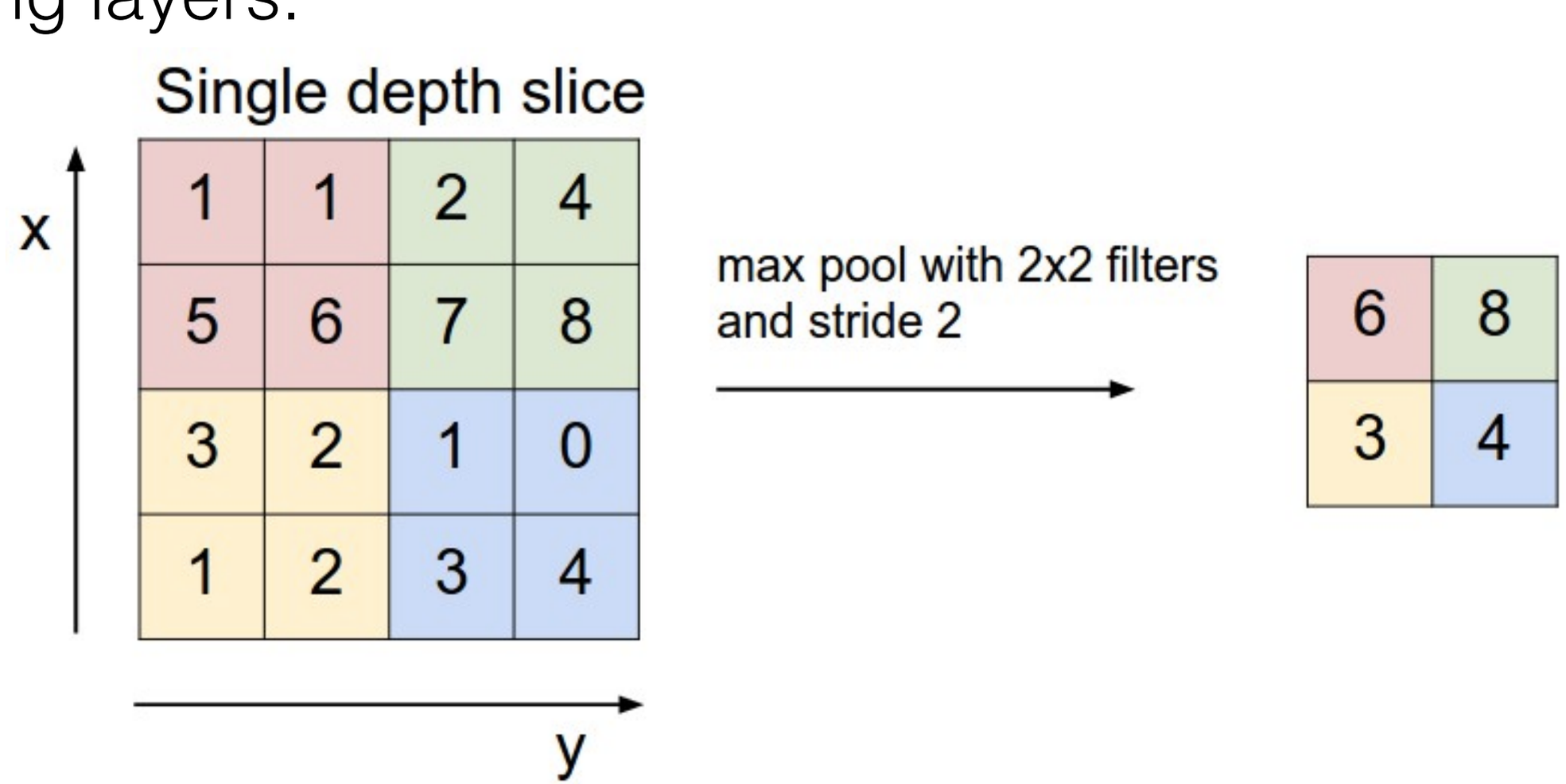
Convolutional Layers

- Every trained kernel operation is the same across an entire input image or feature map.
- Each convolutional layer trains an array of kernels to produce output feature maps.
- Weights for a given convolutional layer are a 4D tensor of NxMxHxW (number of incoming features, number of outgoing features, height, and width)

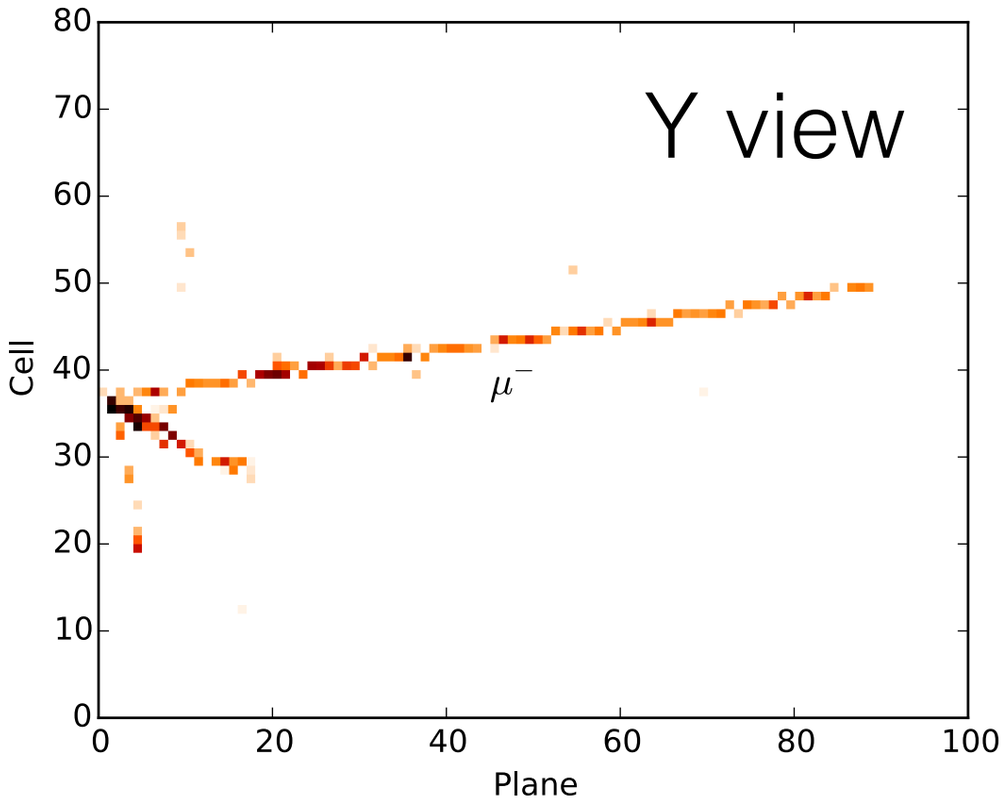


Pooling Layers

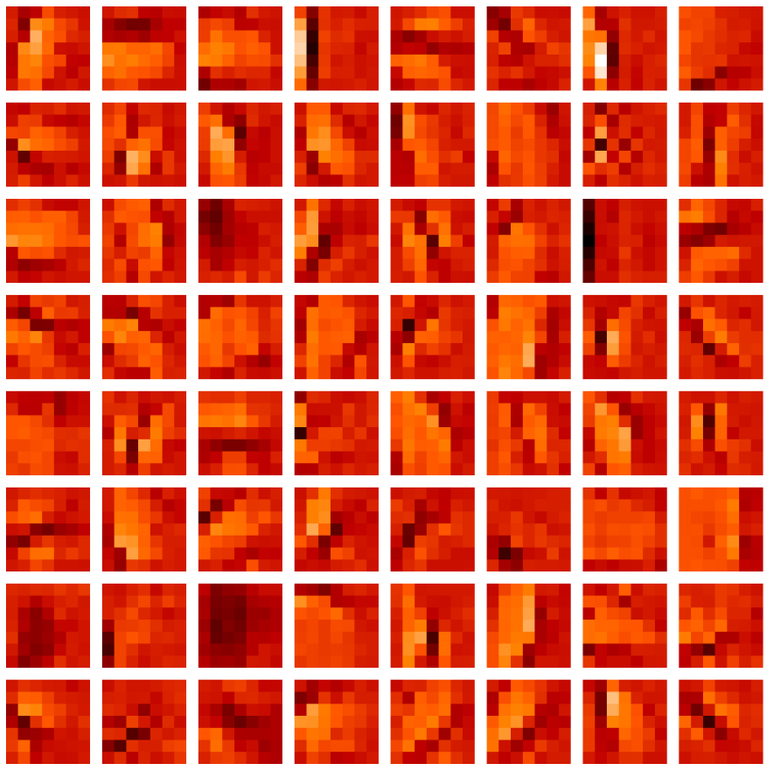
- Intelligent downscaling of input feature maps.
- Stride across images taking either the maximum or average value in a patch.
- Same number of feature maps, with each individual feature map shrunk by an amount dependent on the stride of the pooling layers.



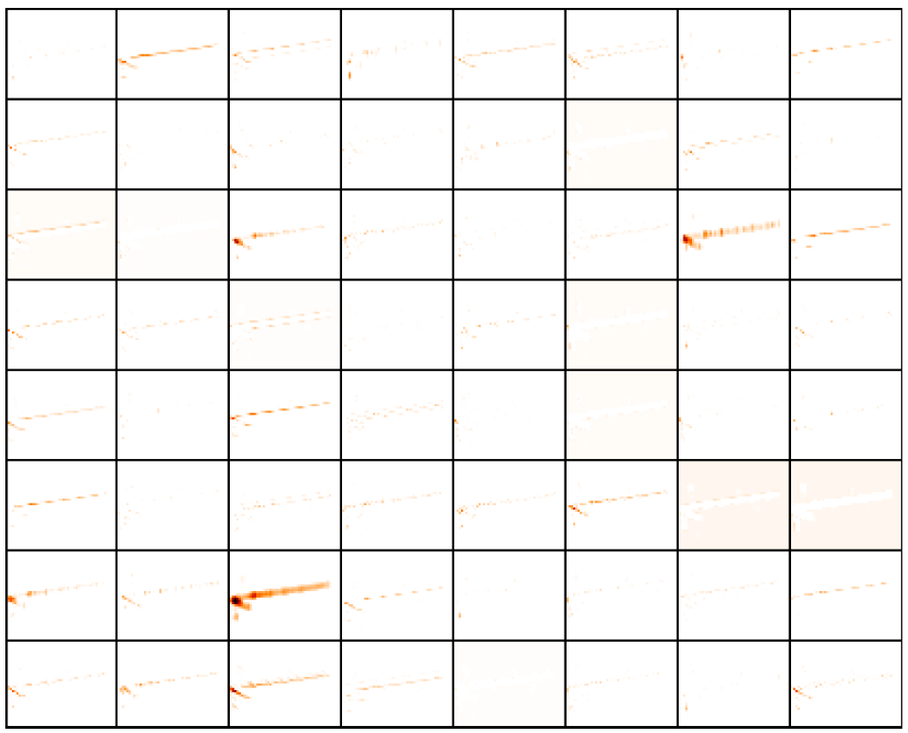
Example CVN Kernels In Action: First Convolution



X



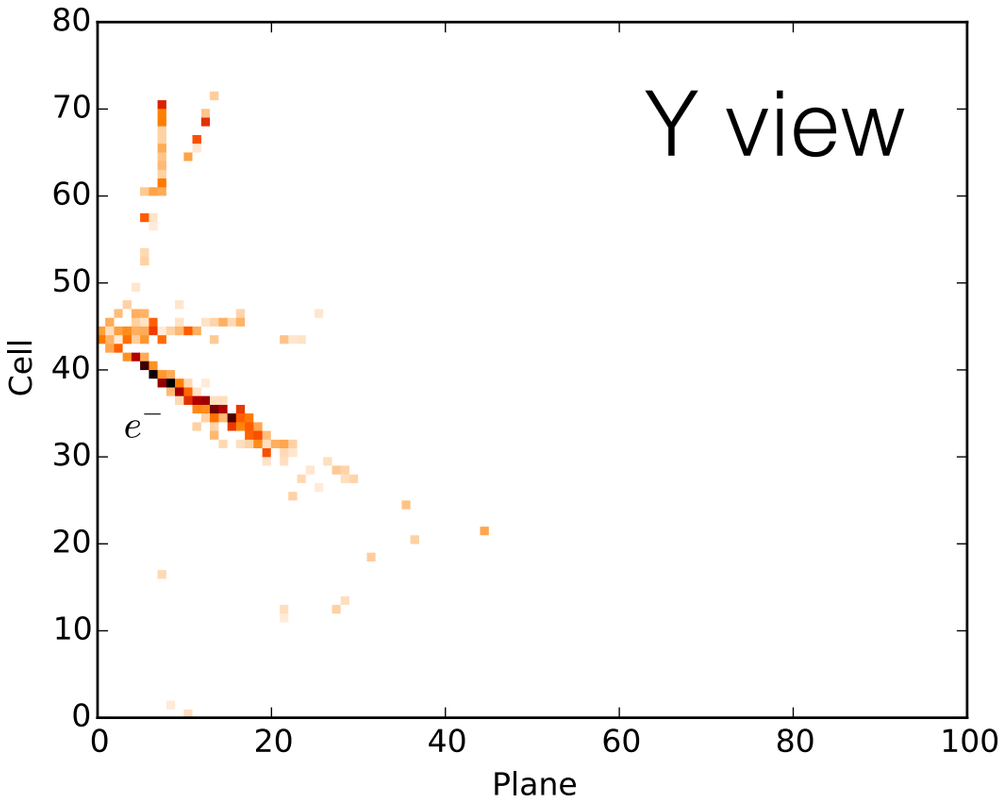
=



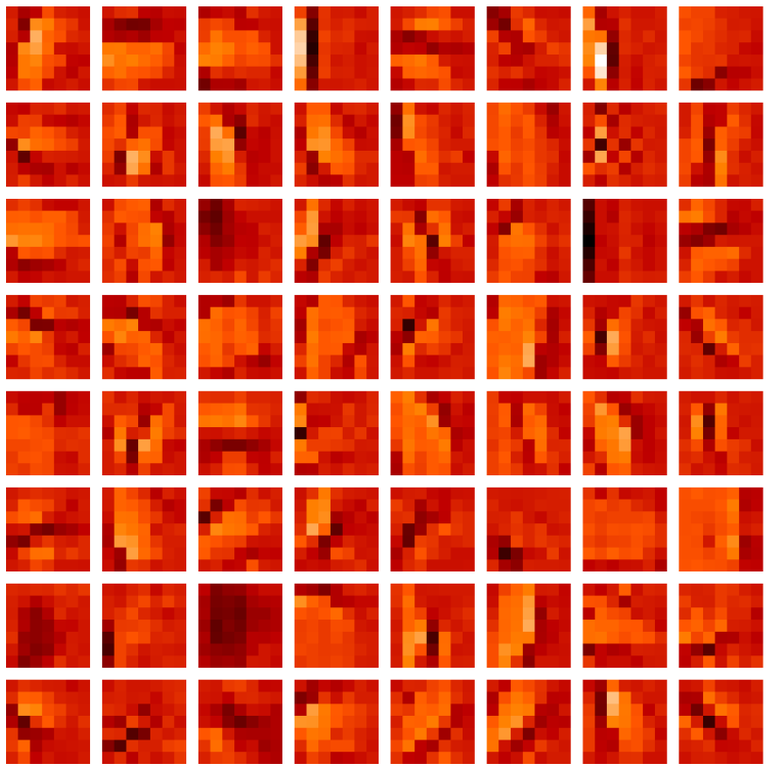
Here the earliest convolutional layer in the network starts by pulling out primitive shapes and lines.

Already “showers” and “tracks” are starting to form.

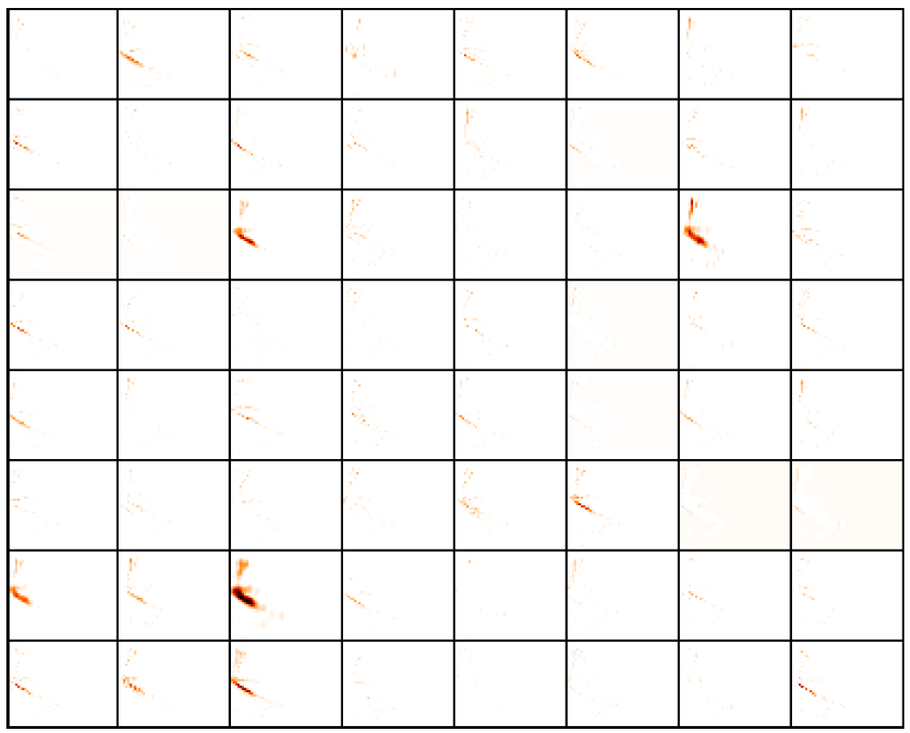
Example CVN Kernels In Action: First Convolution



X



=



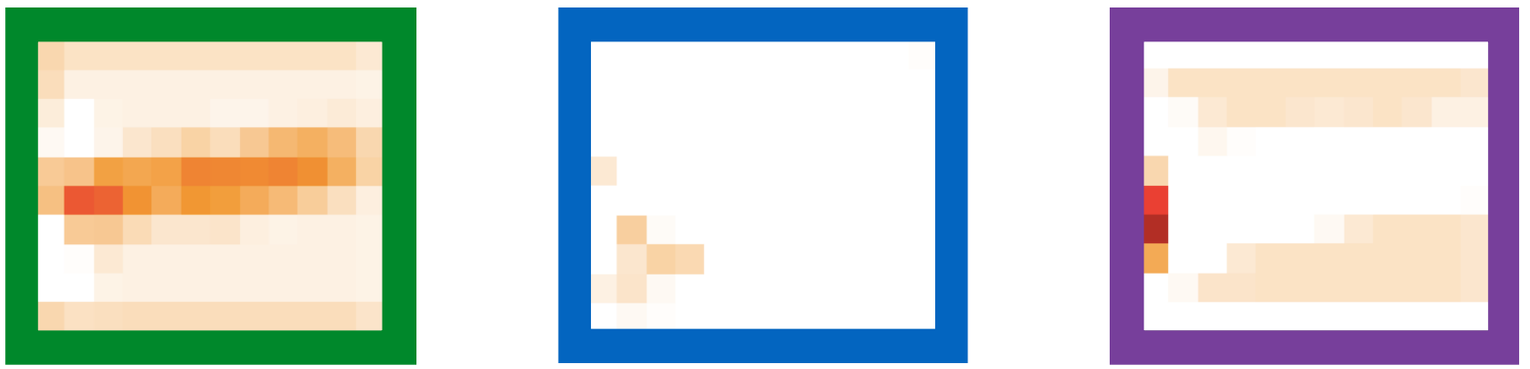
Here the earliest convolutional layer in the network starts by pulling out primitive shapes and lines.

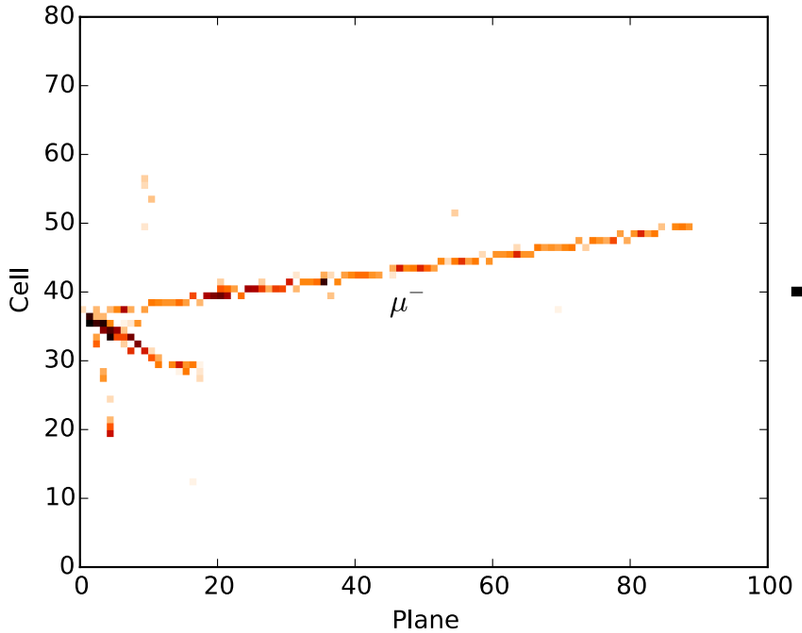
Already “showers” and “tracks” are starting to form.

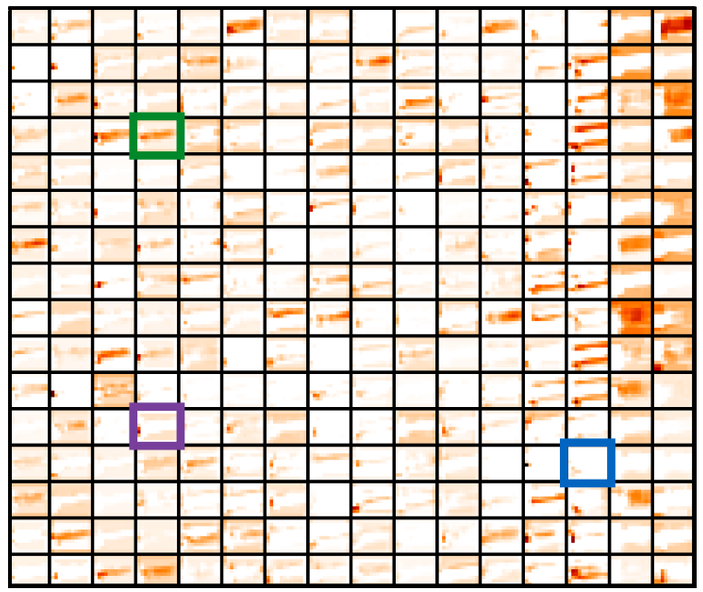
Example CVN Kernels In Action: First Inception Module Output

Deeper in the network, now after the first inception module we can see more complex features have started to be extracted.

Some seem particularly sensitive to muon tracks, EM showers, or hadronic activity.



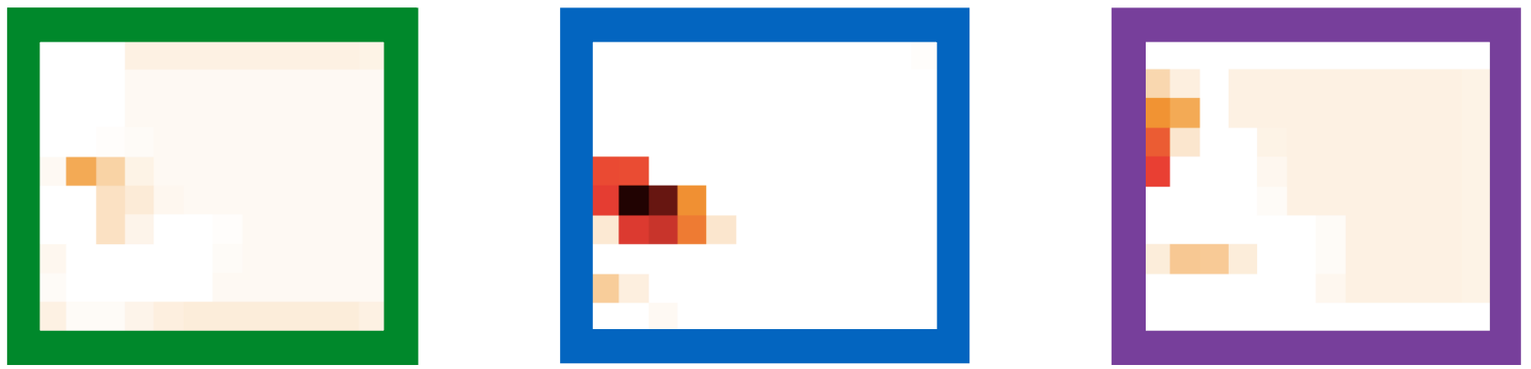


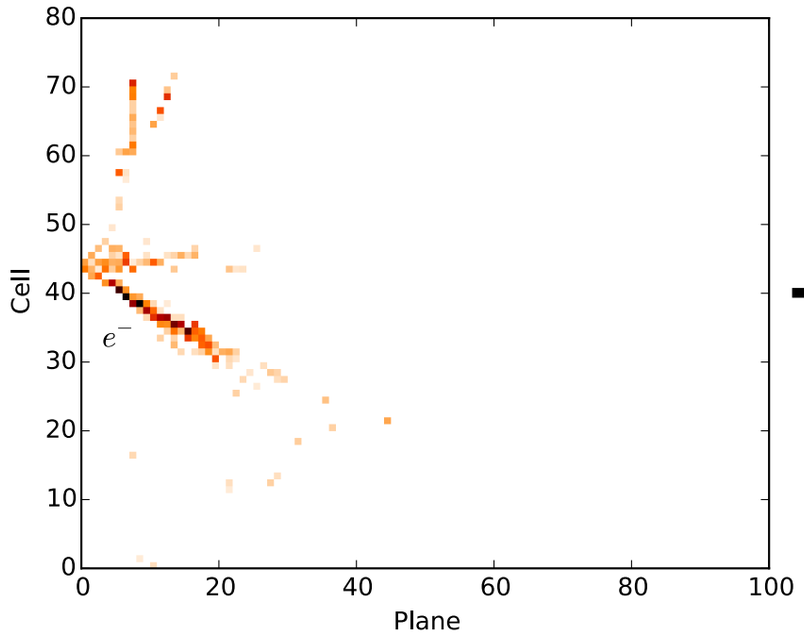


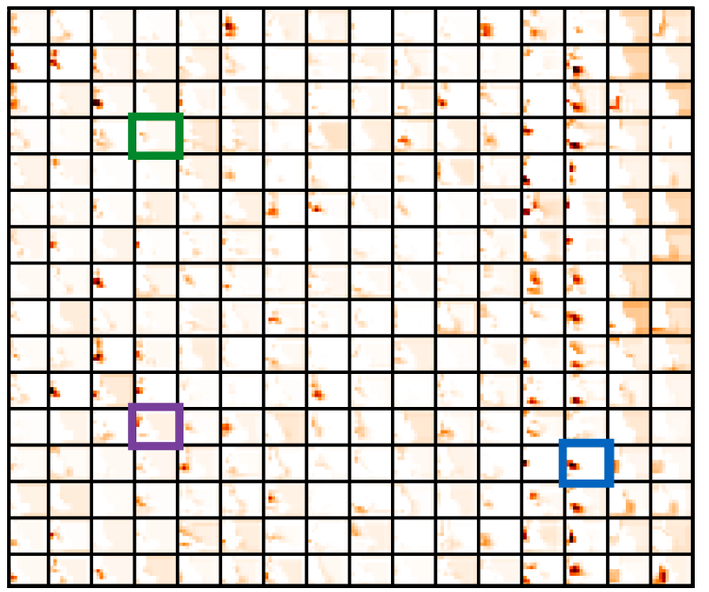
Example CVN Kernels In Action: First Inception Module Output

Deeper in the network, now after the first inception module we can see more complex features have started to be extracted.

Some seem particularly sensitive to muon tracks, EM showers, or hadronic activity.







Example CVN Kernels In Action: First Inception Module Output

Deeper in the network, now after the first inception module we can see more complex features have started to be extracted.

Some seem particularly sensitive to muon tracks, EM showers, or hadronic activity.