DUNE neutrino flavour identification using a CNN

Saul Alonso Monsalve and Leigh Whitehead

19/04/18

Introduction

- One of the most important parts of any neutrino oscillation analysis is the identification of the neutrino flavour

- We typically want to classify events into one of four types:
  - Charged-current ν_e, ν_μ, ν_τ or neutral-current

- Will focus a bit more on the ν_e analysis here since it is the analysis that will provide a measurement of *CP*-violation
  - DUNE's flagship analysis

- The technique will also provide very good selections of the other particle types, however

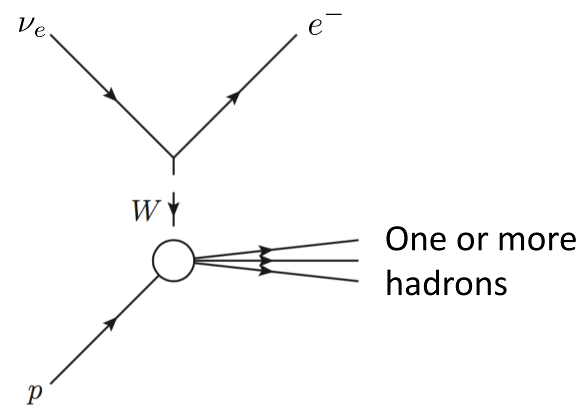
Electron Neutrino Analysis

- Our biggest challenge is the selection (rejection) of signal (background) interactions

- The analysis must therefore:
  - Efficiently select signal charged current electron neutrino interactions
  - Reject backgrounds arising from other neutrino interactions
    - Main one comes from neutral current interactions producing a π^0 meson
    - The two decay photons will pair produce

$$\gamma \rightarrow e^- e^+$$

    - Can mimic the electron signal in the TPC



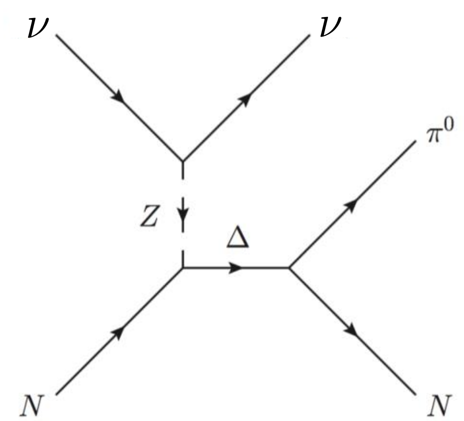


Image Recognition

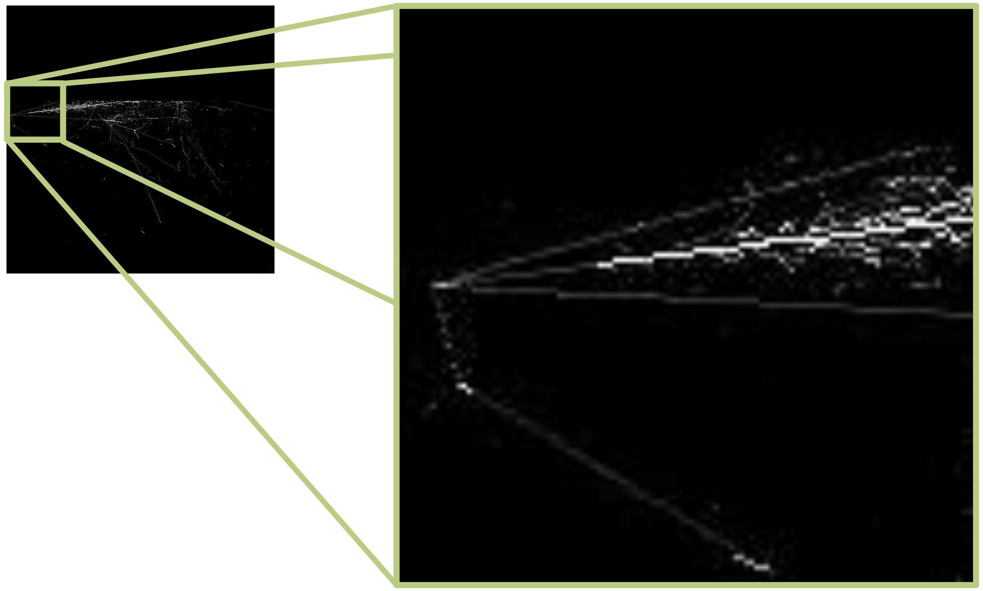
- The human eye is a remarkably good image recognition tool
  - Once you know what to look for, it is fairly easy to spot whether the shower is displaced from the vertex or not

- Realistically the experiment will produce too much data for scanning the interactions by eye

- We need to be able to train a computer to do this task
  - Recent years have shown rapid development of automated image recognition
  - One of the most promising approaches is the Convolutional Neural Network (CNN)

Convolutional Neural Network

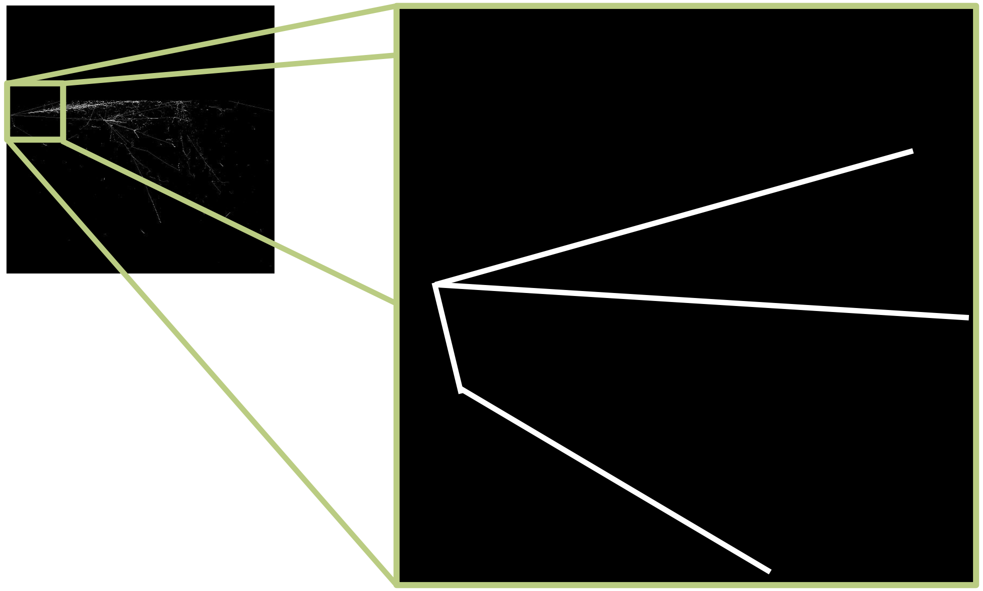
- CNNs are used to classify images by applying filters to small patches of the image (using a convolution)
- Scans over the image with a number of N x N pixel filters



- Each filter extracts some feature from the image

Convolutional Neural Network

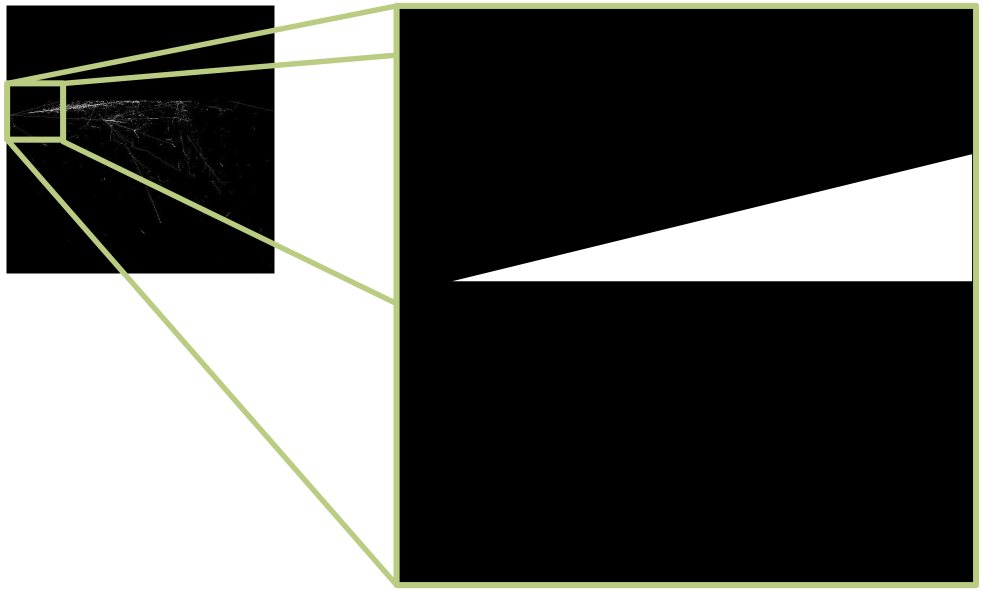
- CNNs are used to classify images by applying filters to small patches of the image (using a convolution)
- Scans over the image with a number of N x N pixel filters



- Each filter extracts some feature from the image
- For example, filter 1 may find tracks

Convolutional Neural Network

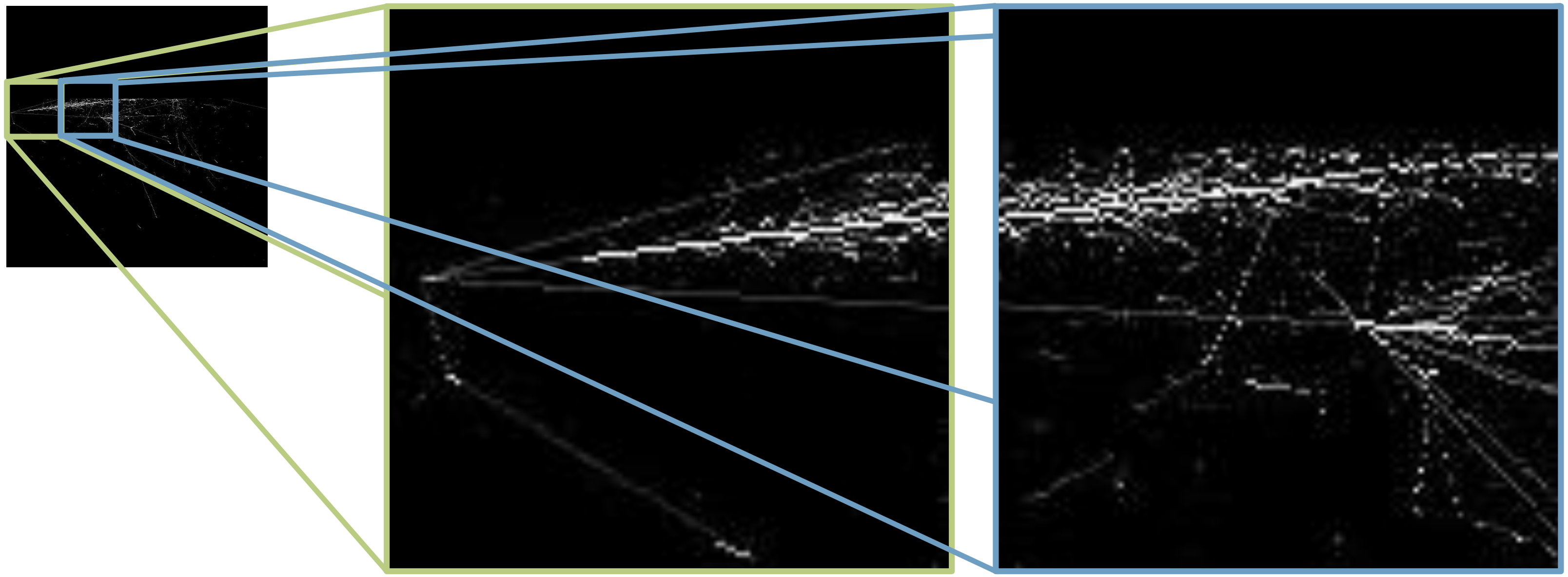
- CNNs are used to classify images by applying filters to small patches of the image (using a convolution)
- Scans over the image with a number of N x N pixel filters



- Each filter extracts some feature from the image
- For example, filter 1 may look for tracks
- Filter 2 might look for showers

Convolutional Neural Network

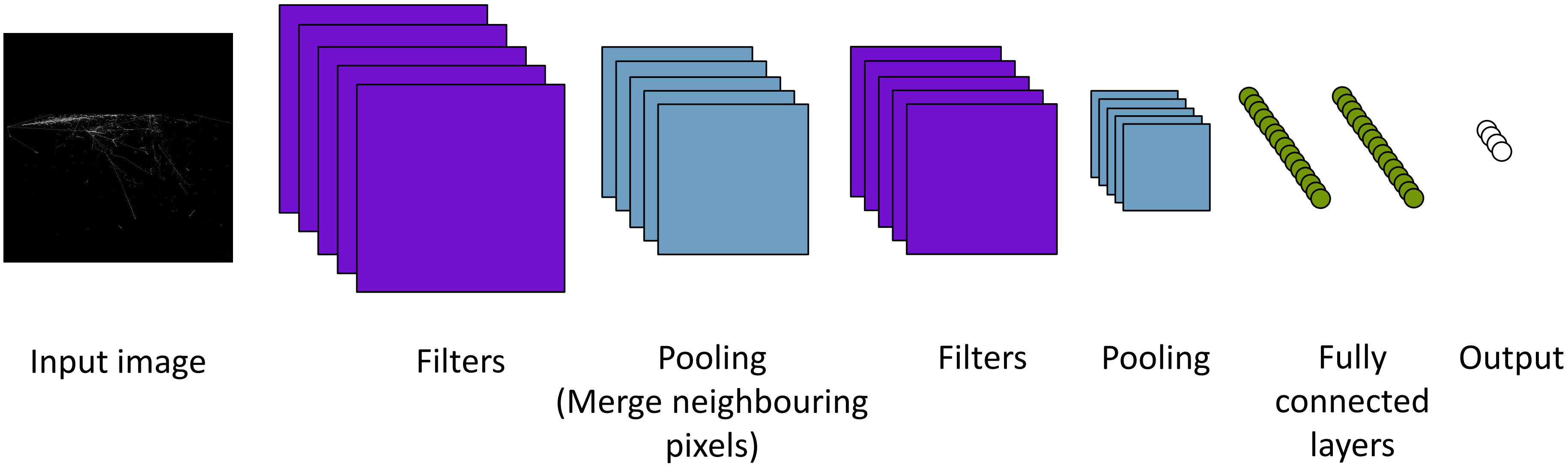
- CNNs are used to classify images by applying filters to small patches of the image (using a convolution)
- Scans over the image with N x N pixel filters



- Then move onto the next patch of the image and repeat the process

Convolutional Neural Network

- The output from each filter then forms the basis of the next layer which can include further filters



- Different architectures can be considerably more complex than the above toy example

Using CNNs

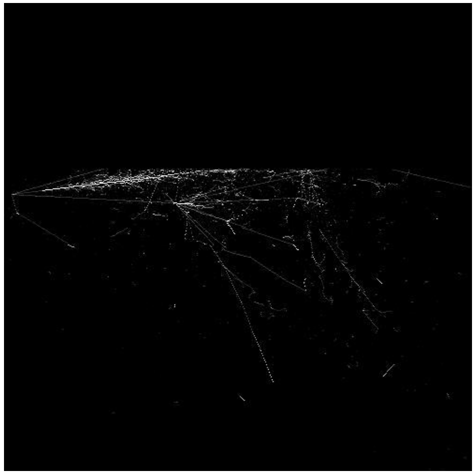
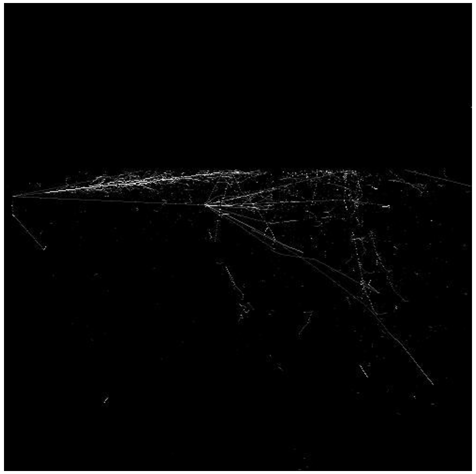
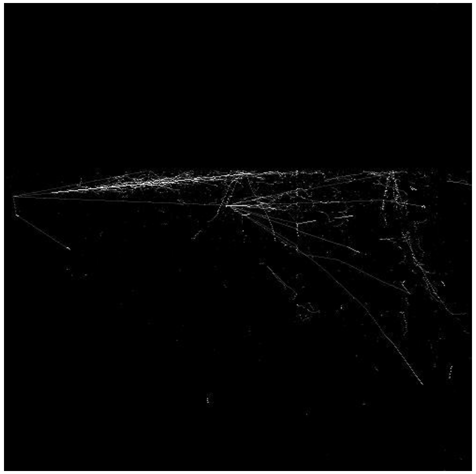
- Use millions of images of neutrino interactions with the true neutrino flavour known
  - Allows the CNN to learn the features of each type of neutrino interaction
  - The CNN filters are not predefined – it needs to learn which filters to use to extract the information required to classify events

- Once the CNN is trained it is applied to images with no truth information attached – eventually the experimental data

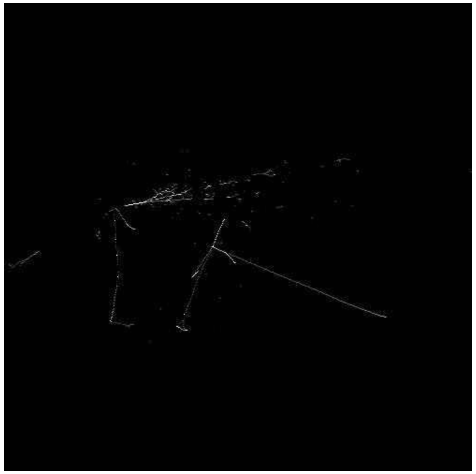
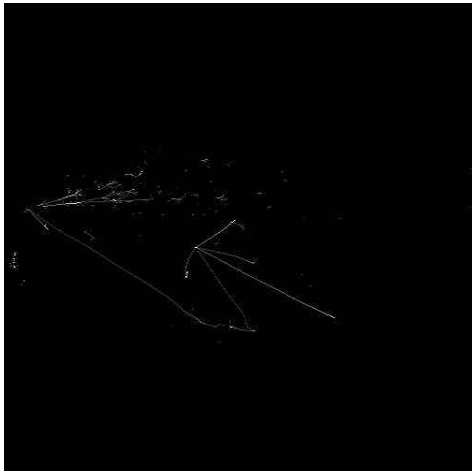
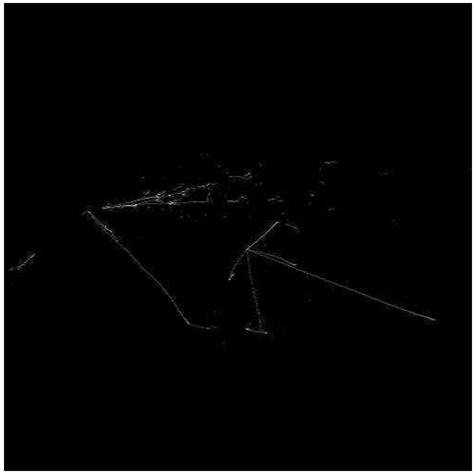
- The CNN gives probabilities for each event to be the following:
  - Charged-current ν_e, ν_μ, ν_τ.
  - Neutral-current (all flavours).

Images example

- Simulated electron neutrino interaction (signal)

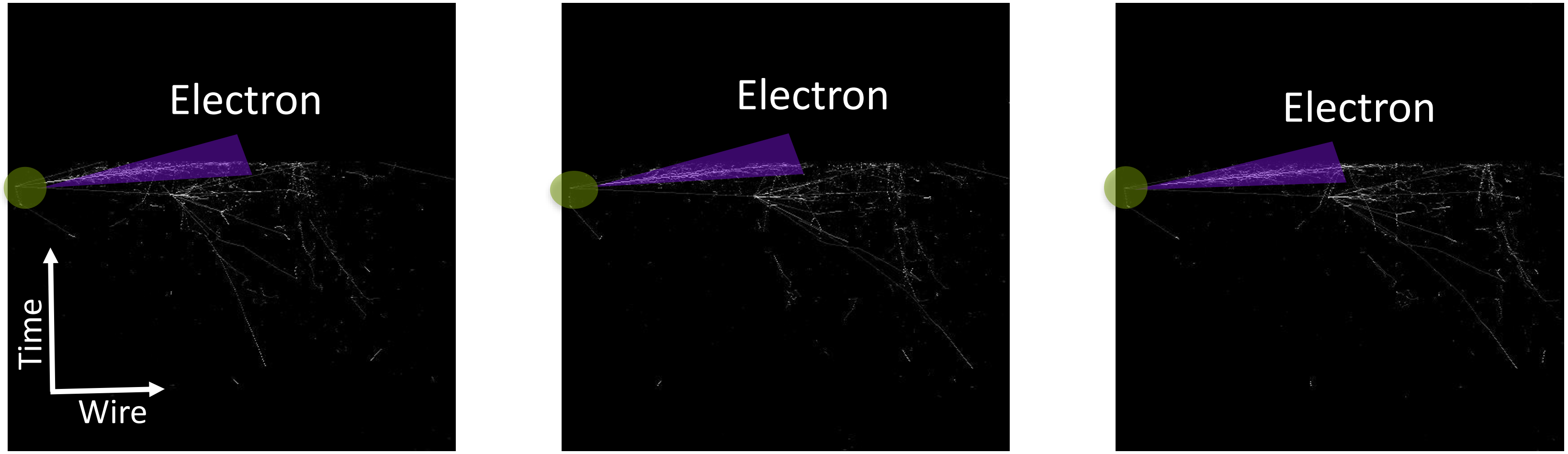
  

- Simulated neutral current π^0 interaction (background)

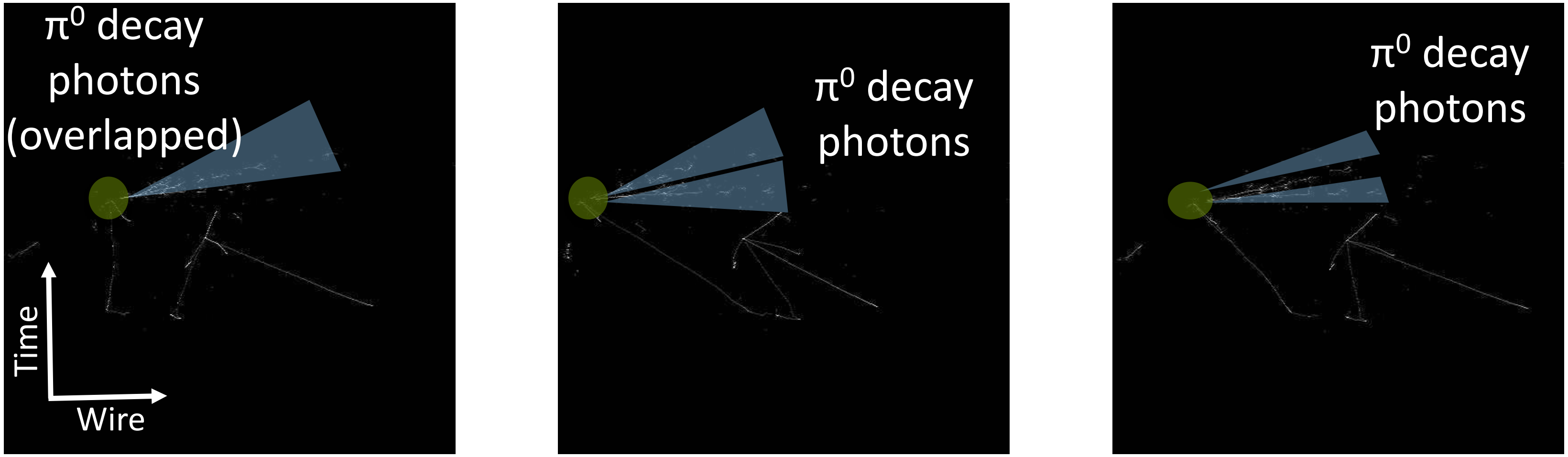
  

Images example

- Simulated electron neutrino interaction (signal)



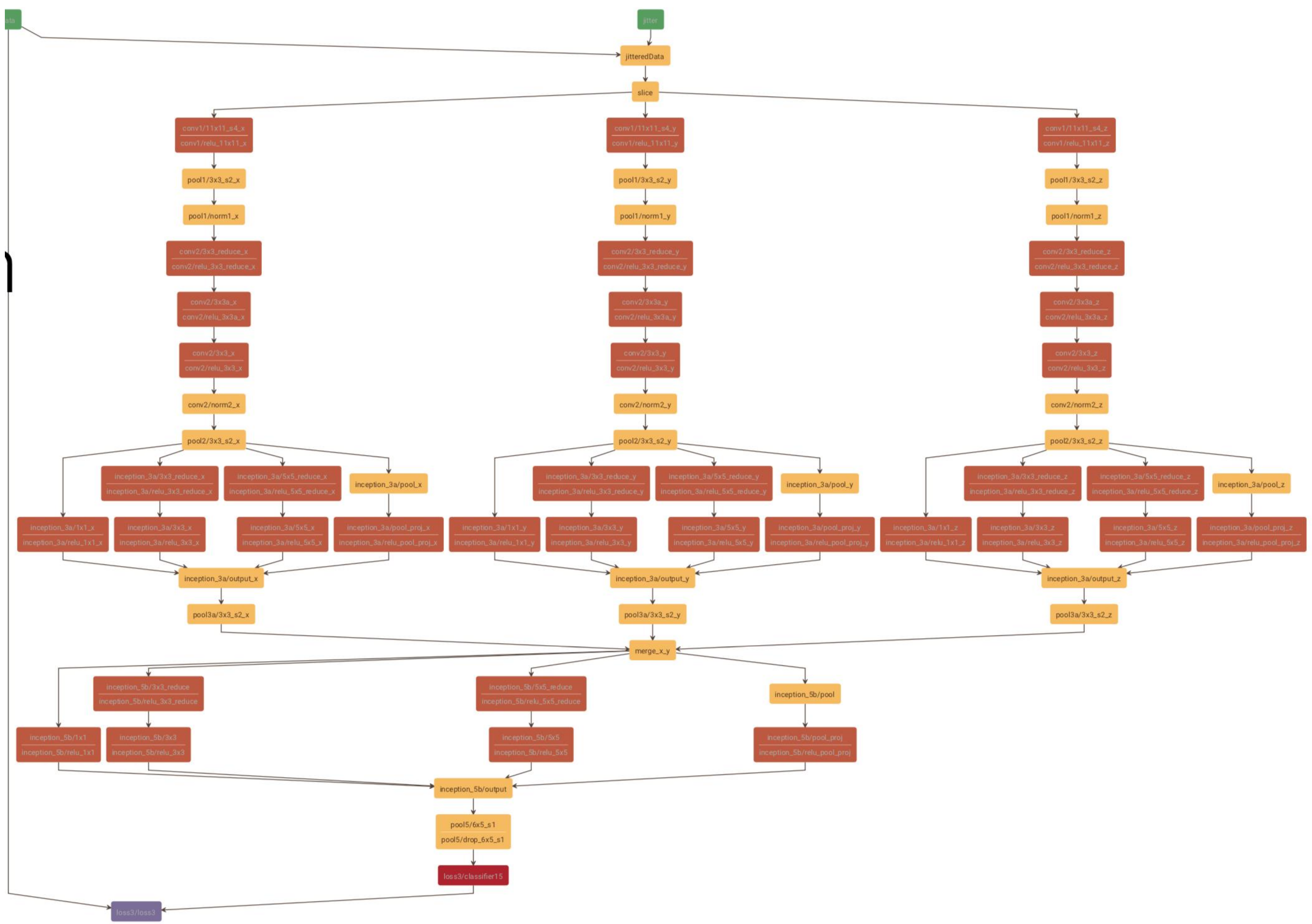
- Simulated neutral current π^0 interaction (background)



Last CVN state (January 2018)

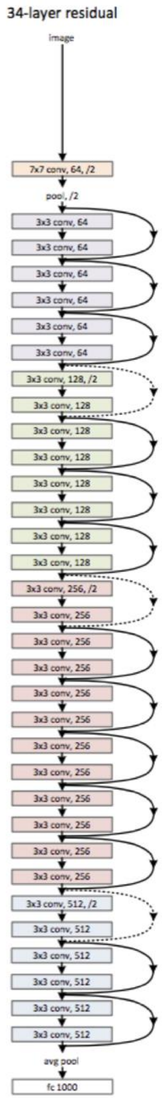
- CVN: Convolutional Visual Network.
- Input: 500x500 pixel images.
- 13 interaction types.
- CVN inspired by the GoogleNet (arXiv:1409.4842) CNN architecture.
- Deep Learning Framework: Caffe (developed by Berkeley AI Research).
- Trained on the GPUs in the Wilson cluster (Fermilab).
- Accuracy: ~68%...

Last CVN state (January 2018)



Moving forward

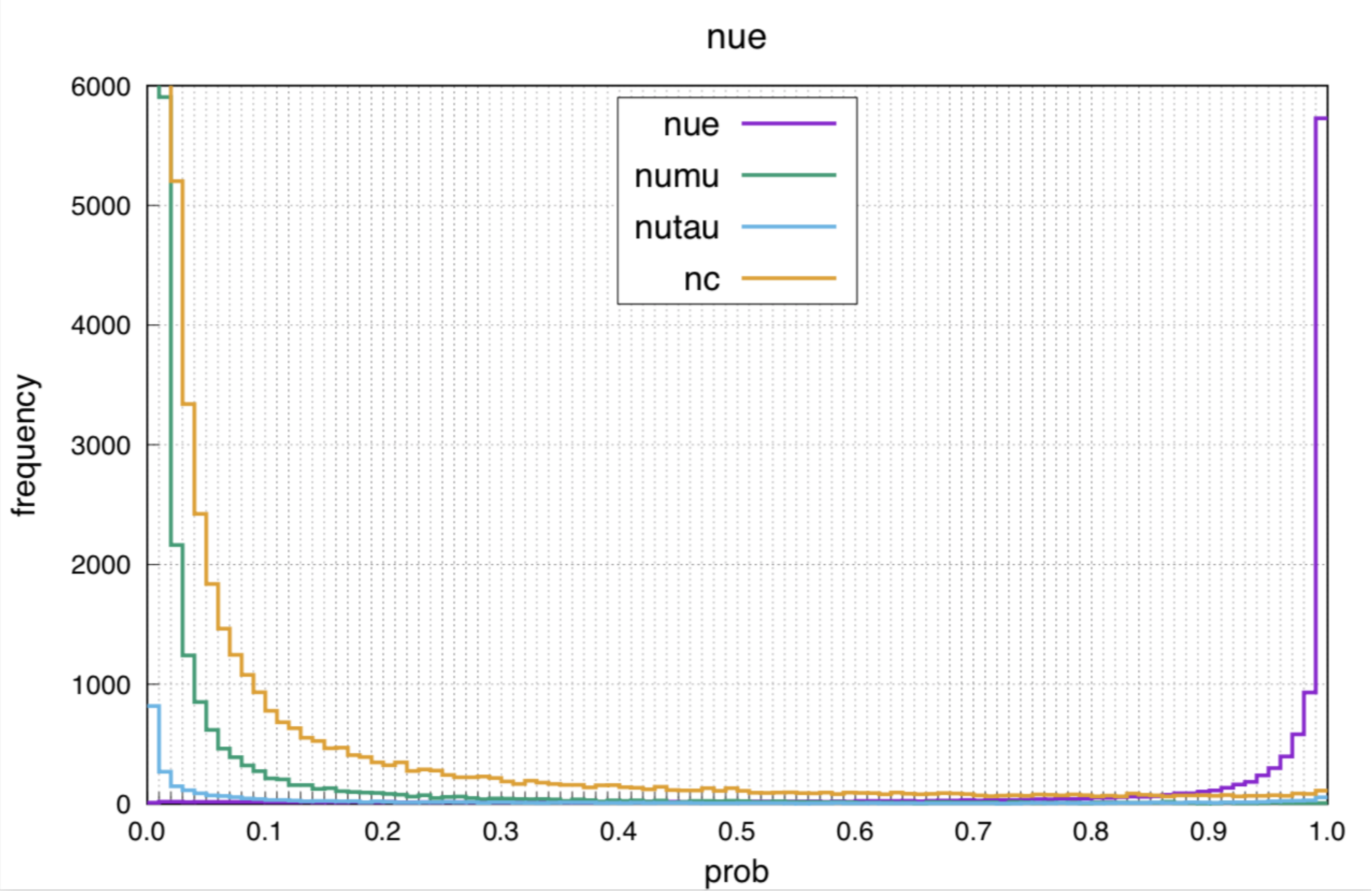
- Move from Caffe to Tensorflow (developed by Google Brain Team) for ease of design for more complex network topologies.
  - Tensorflow is also included in LArSoft.
  - Efficient image format.
- Use a more novel and sophisticated network architecture: Residual Neural Network (ResNet)
  - It is still a convolutional neural network.
  - The input of a lower layer is made available to a node in a higher layer.
  - The current worldwide best CNNs are based on ResNets.
  - Versions with 18, 34, 50, 101, and 152 layers.



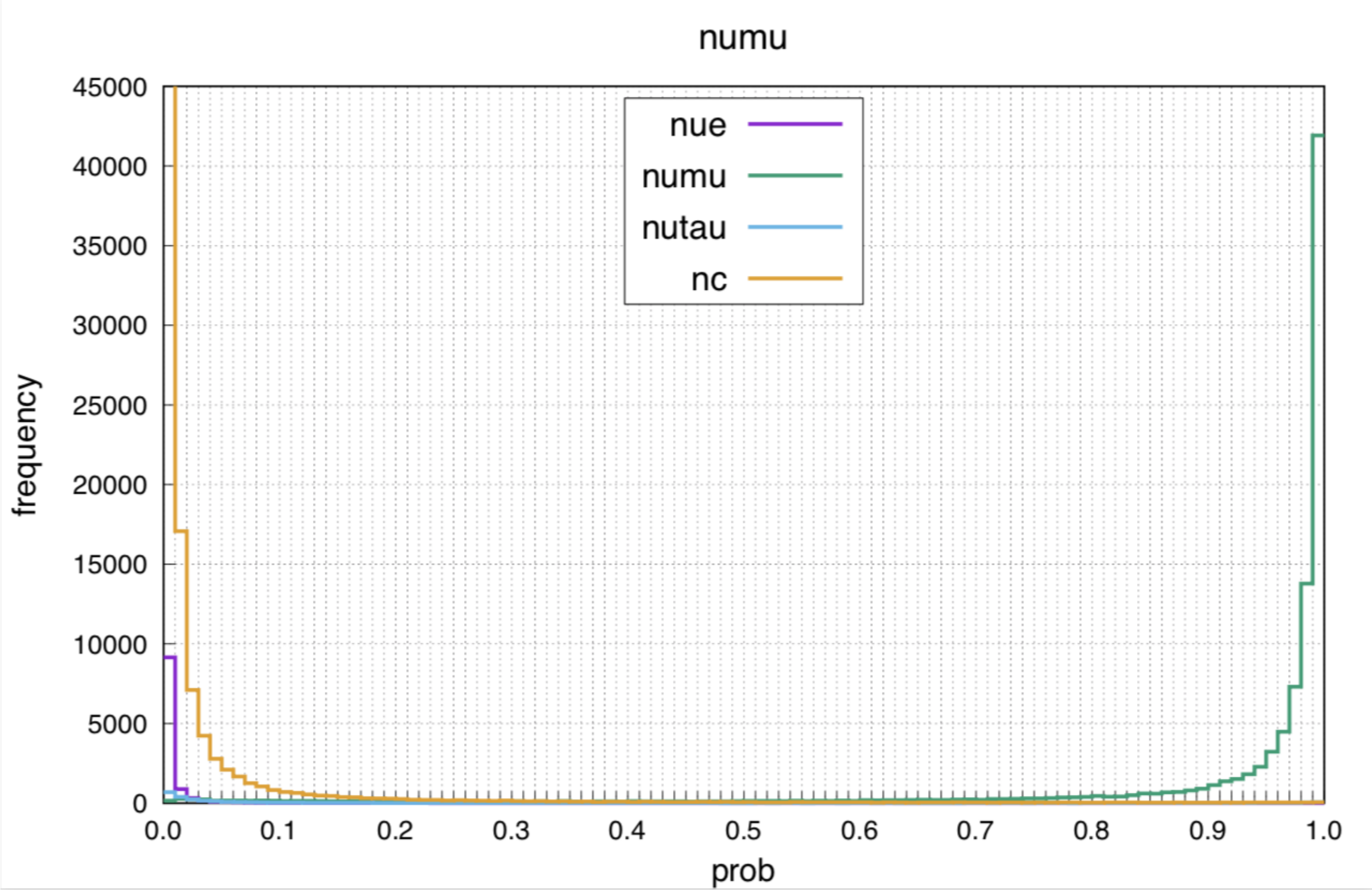
Some results

- The network is still training.
- Images generation:
  - Before: ~1 week. Now: ~1 day.
- Number of images used while training:
  - Before: ~1 million. Now: ~4 million.
- CVN accuracy:
  - Before: ~68%. Now: ~80%.

Nue probability histogram



Numu probability histogram



Conclusions

- We have implemented a new Tensorflow framework.
- CVN architecture: from GoogleNet to ResNet.
- We are now working on developing the performance with this new architecture:
  - The current CVN outperforms the previous one.

- Next steps:
  - Get it interfaced into LarSoft (as part of the standard DUNE processing).
  - Once we have the network ready in LArSoft we can process things through the DUNE software to produce the new CP-violation sensitivity plot.