

# 3-D Regression CNN for Vertex & Prong Reconstruction

Junze Liu, Ben Jargowsky, Pierre Baldi and Jianming Bian

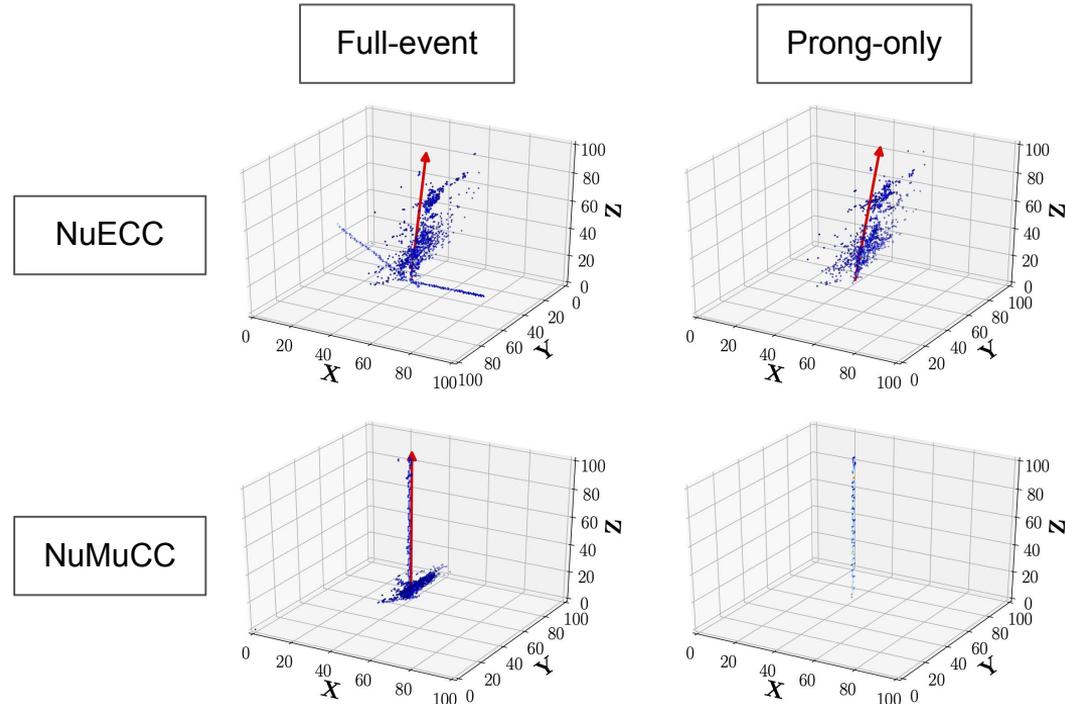
University of California, Irvine

# Summary

- Motivations
  - The deep-learning based particle energy, vertices and momentum (energy+direction) reconstruction are necessary for a full AI based event reconstruction chain.
  - Combining the particle mass with its kinetic energy and direction, a final state particle's 4-momentum can be obtained
- This talk
  - Vertex Reconstruction
    - 2-D CNN had good results with 2-stage training
    - Developing 3-D CNN for NuE CC
  - Efficiency improvements using alternative deep learning architectures

# 3-D Pixel Maps Visualization

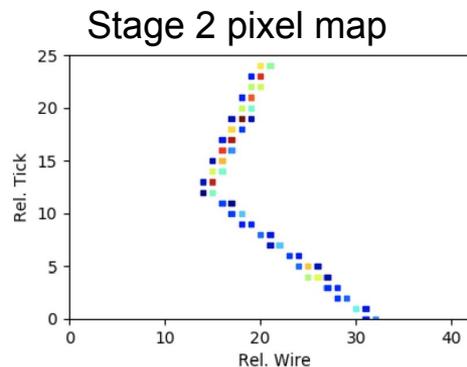
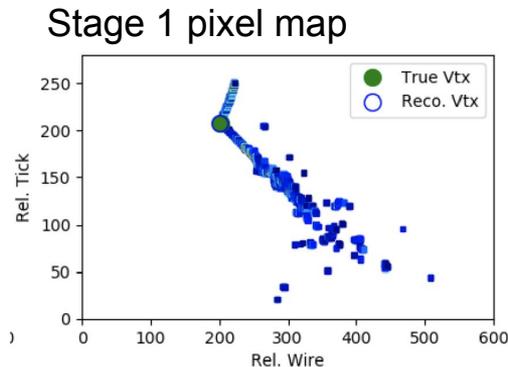
- Created by combining spatial and charge information from all 3 2-D planes.
- These 3-D pixelmaps are 100x100x100 pixels which are 125x125x250 cm for  $\nu_e$  and 500x500x1000 cm for  $\nu_{\mu^*}$ .



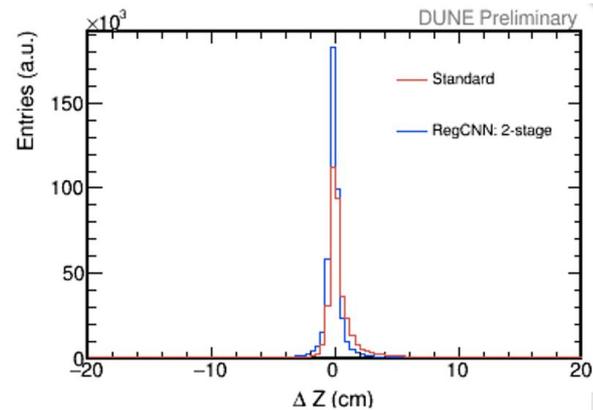
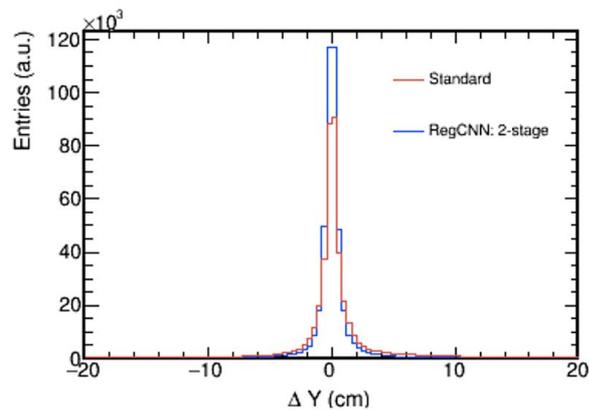
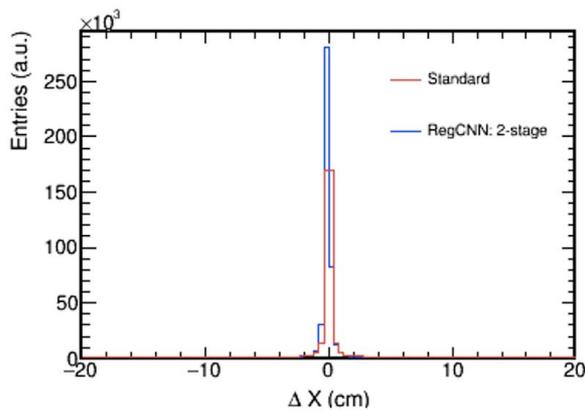
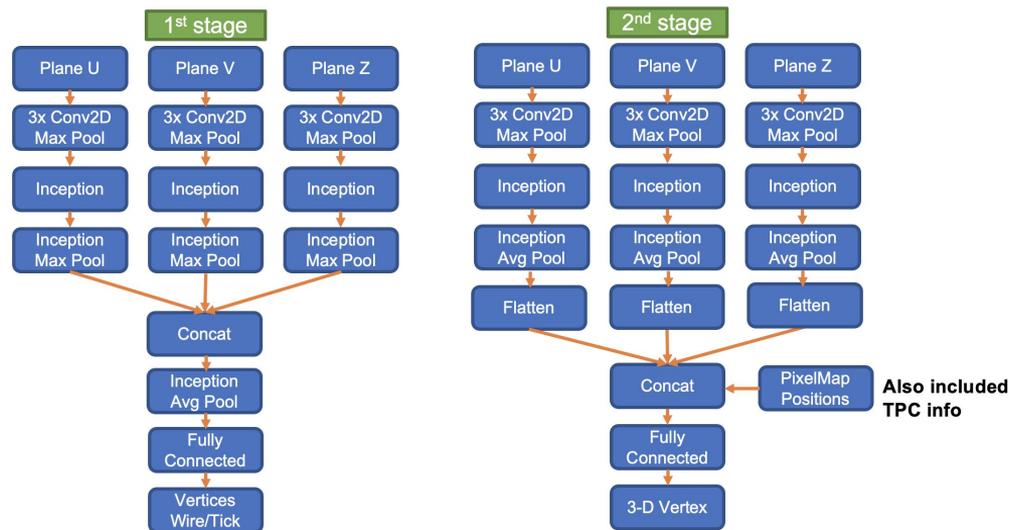
# Vertex Reconstruction

# Current 2-stage, 2-D CNN vertex reconstruction

- The issue may be caused by the large pixel map size (low statistics)
- In our current 2-D CNN vertex reco, which has better performance than standard method, we construct a 2-stage architecture.
- Stage 1: propose the vertex on each plane and crop each view and make smaller pixel map
- Stage 2: reconstruct the 3-D vertex with the smaller pixel map
- We will try similar method in 3-D CNN

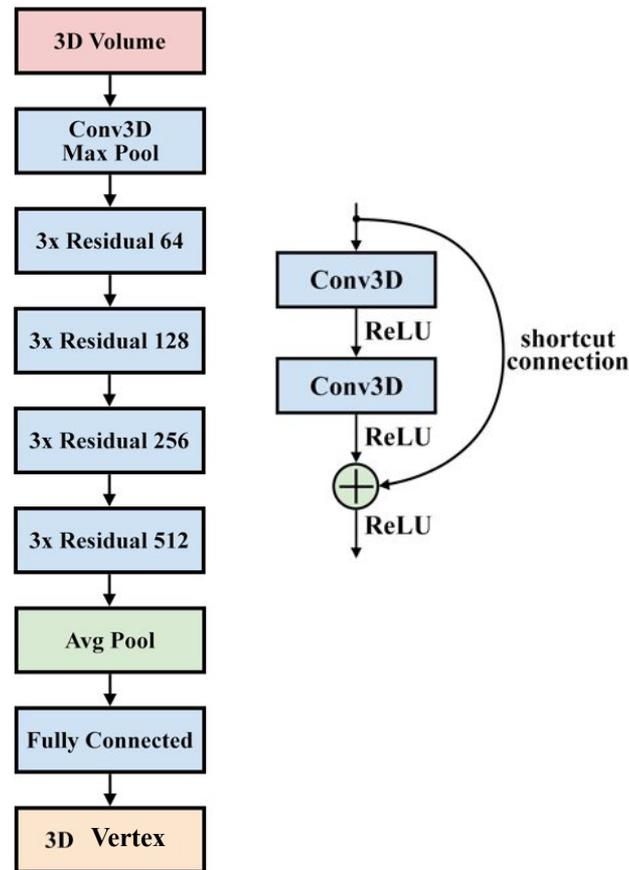


# 2-CNN architecture and results (Ilsoo Seong)



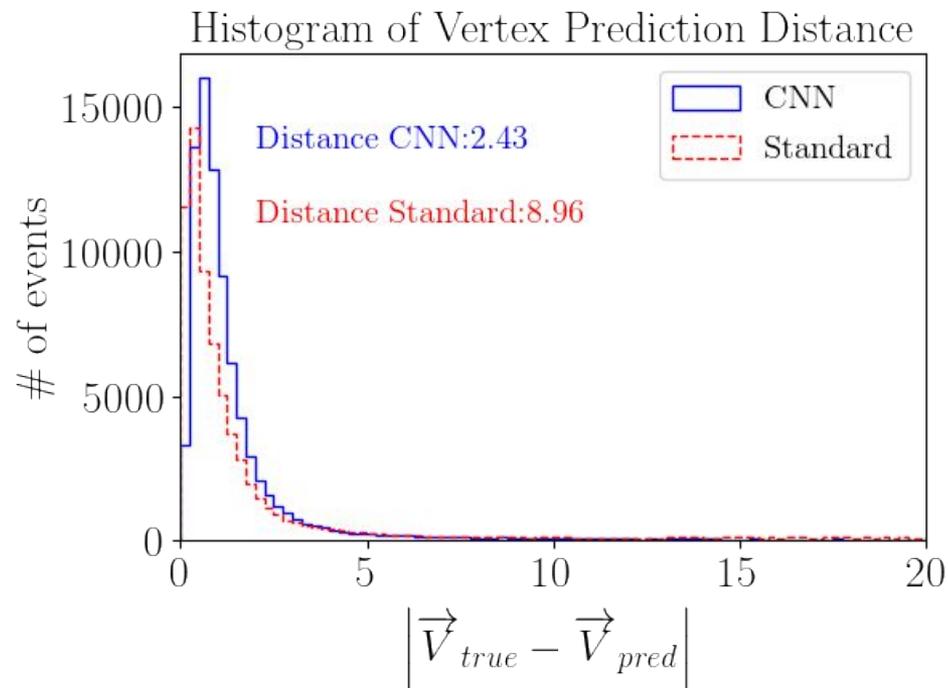
# 3-D Architecture

- Loss Function
  - Log-MSE for training
  - Euclidean distance in real coordination for evaluation
- Training
  - 50 epochs
- Models Trained
  - Event RegCNN
    - Input: 3-D full-event pixel map
    - Output: 3-D vertex coordinate
  - Prong RegCNN
    - Input: 3-D prong-only pixel map
    - Output: 3-D vertex coordinate



## Log-MSE as Loss function

- $\text{Log}(\text{Mean}((x_{true} - x_{pred})^2, (y_{true} - y_{pred})^2, (z_{true} - z_{pred})^2))$
- Calculate the logarithm of MSE of the 3D directions in the relative coordination
  - Consistent with the input volume
  - The input volumes are converted from the real coordination
    - To have a fixed shape (100x100x100)
- The best performance so far

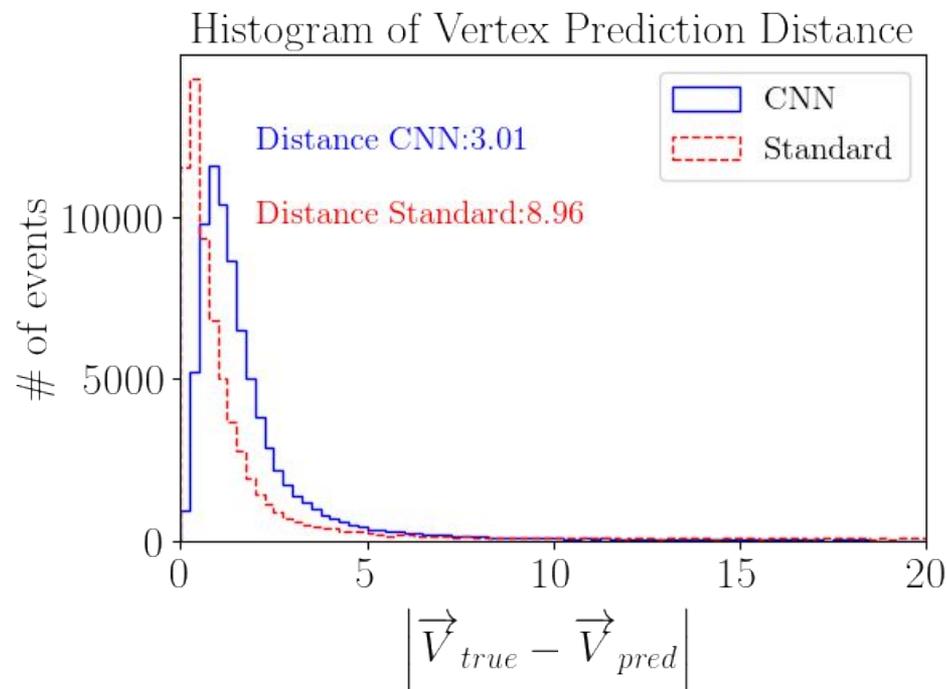


# Z-axis Adjustment for The Loss Function

- The final evaluation is made in the real coordination
  - Loss function in relative coordination could be an inaccurate metric for the training
- Rel-real coordinates conversion
  - $x_{rel} = (x_{real} - center_x + length_x/2) * (bins_x/length_x)$
  - $y_{rel} = (y_{real} - center_y + length_y/2) * (bins_y/length_y)$
  - $z_{rel} = (z_{real} - center_z + length_z/4) * (bins_z/length_z)$
- Lengths in the true coordination on z-axis are always 2 times as on x/y-axis
  - “center” and “bins” are consistent in x/y/z-axis
- Z-axis adjusted loss functions
  - Distance between true and predicted vertices on z-axis gets multiplied by 2

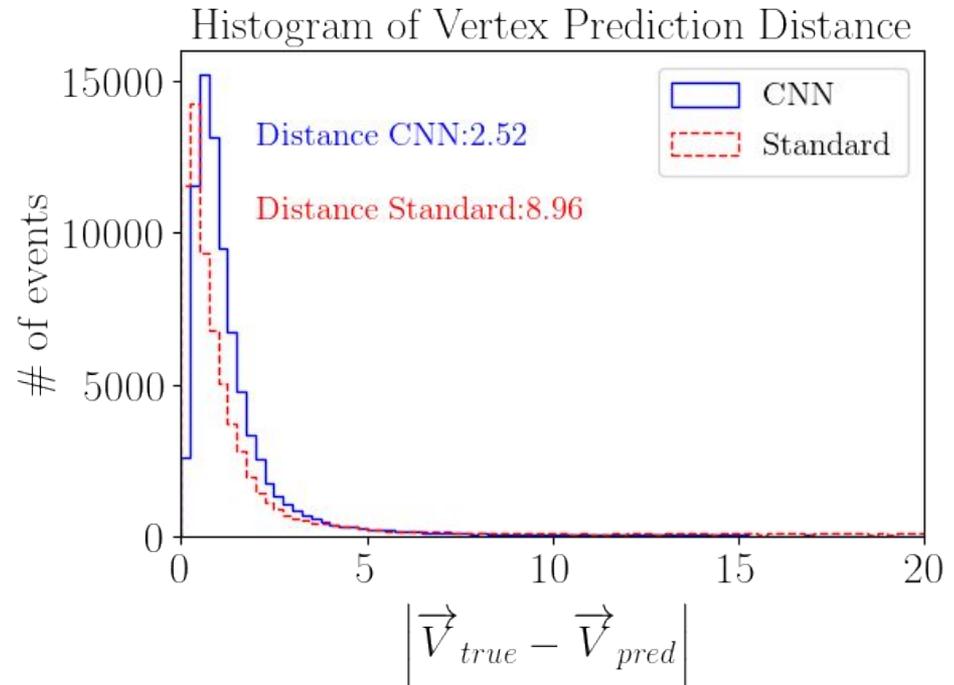
## Z-MSE as Loss function

- $Mean((x_{true} - x_{pred})^2, (y_{true} - y_{pred})^2, (2z_{true} - 2z_{pred})^2)$
- Calculate the Z-MSE of the 3D directions in the relative coordination
  - Consistent with the real coordination
- Not as good as the Log-MSE



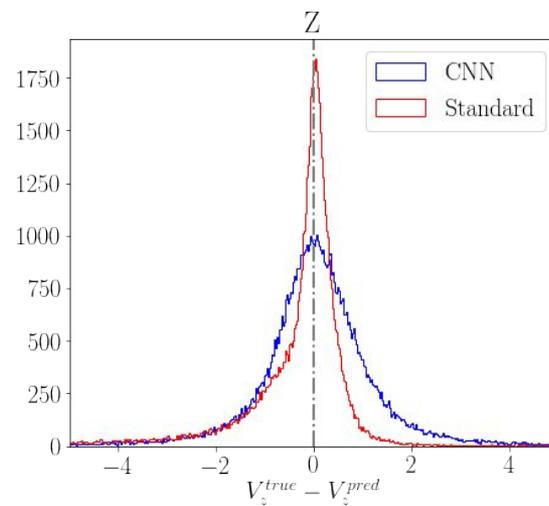
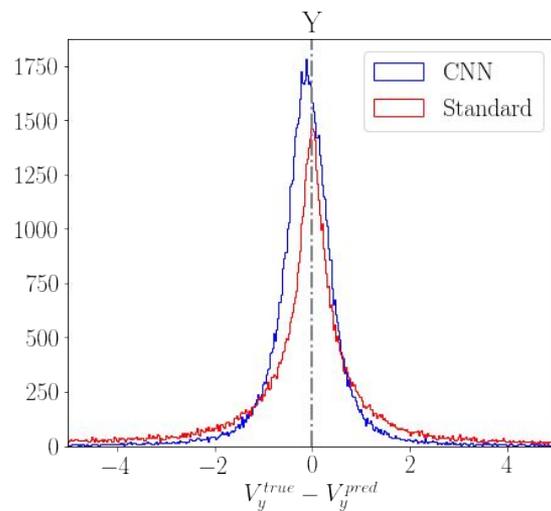
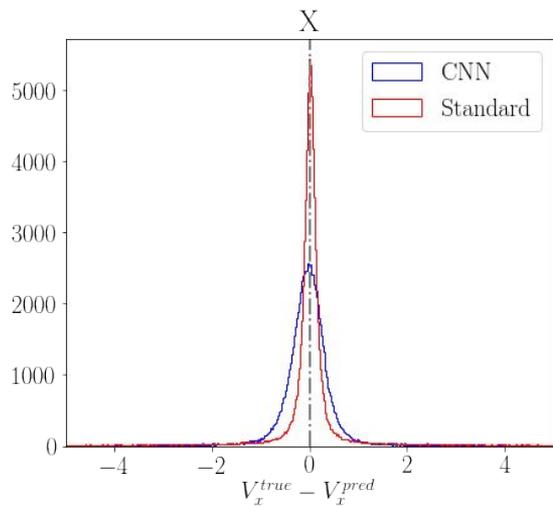
# Log-Z-MSE as Loss function

- $\text{Log}(\text{Mean}((x_{true} - x_{pred})^2, (y_{true} - y_{pred})^2, (2z_{true} - 2z_{pred})^2))$
- Calculate the logarithm of Z-MSE of the 3D directions in the relative coordination
  - Consistent with the real coordination
- Slightly worse than Log-MSE
  - Z-axis adjustment doesn't bring improvement



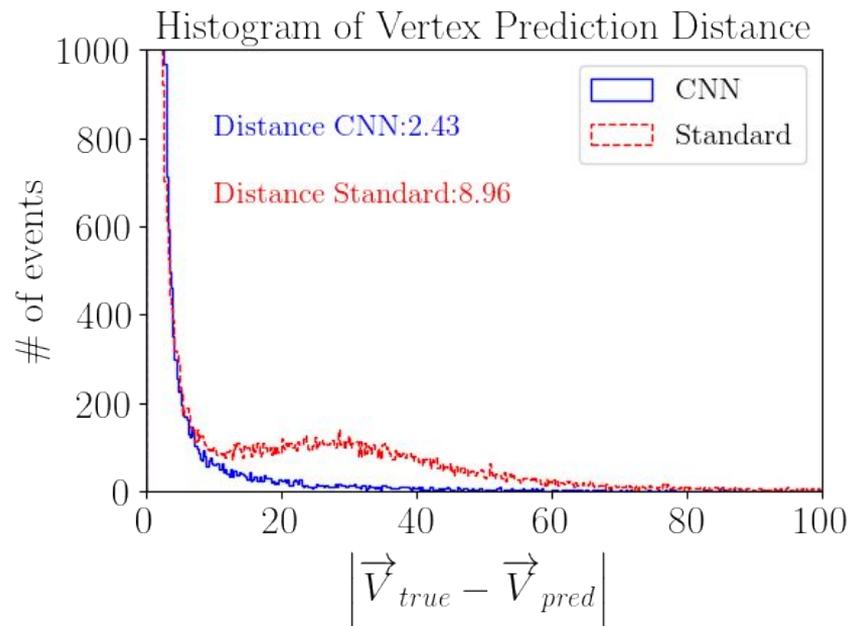
# NuECC - Vertex Regression Distance Histogram Per Axis

- The 3-D RegCNN is still not as good as the standard method in X/Y-axis
  - While the RegCNN-predicted average distance is smaller than the standard method's
    - CNN: 2.43
    - Standard: 8.96



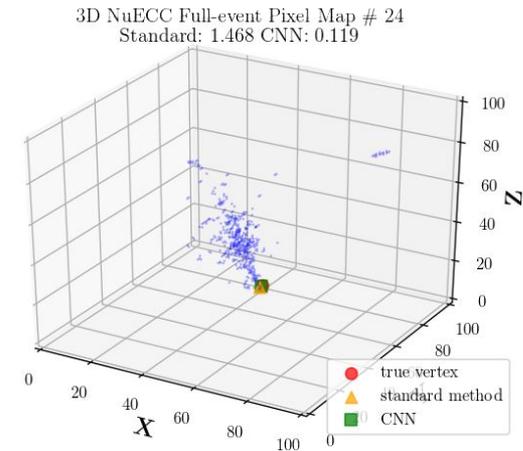
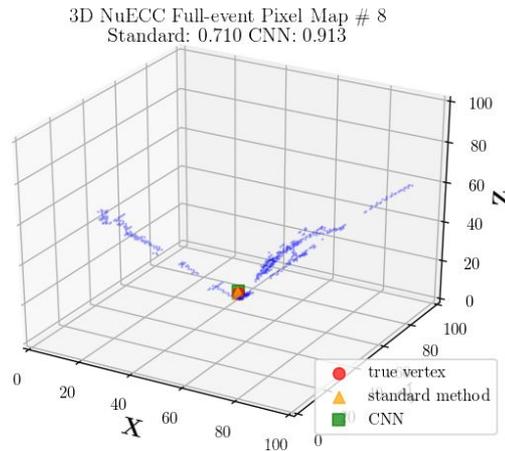
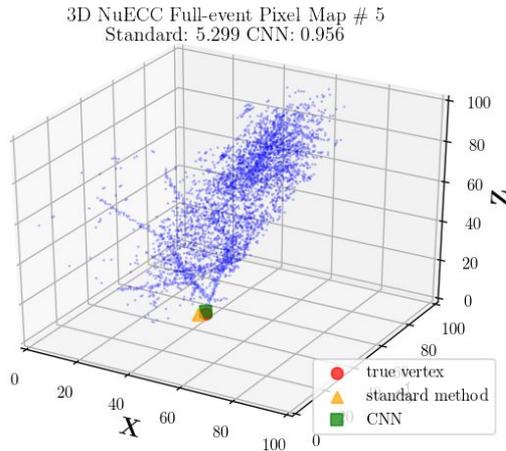
# NuECC - Vertex Regression Distance Histogram Per Axis

- The 3-D RegCNN is still not as good as the standard method in X/Y-axis
  - While the RegCNN-predicted average distance is smaller than the standard method's
    - CNN: 2.43
    - Standard: 8.96
- The standard method makes larger mistakes ( $10\sim 60\text{ cm}$ ) on more events than the 3-D RegCNN



# 3-D Visualization of Vertex Reconstructions

- Rendering the vertices in the relative coordination volumes
  - Each pixel map is of fixed shape 100x100x100 in relative coordination

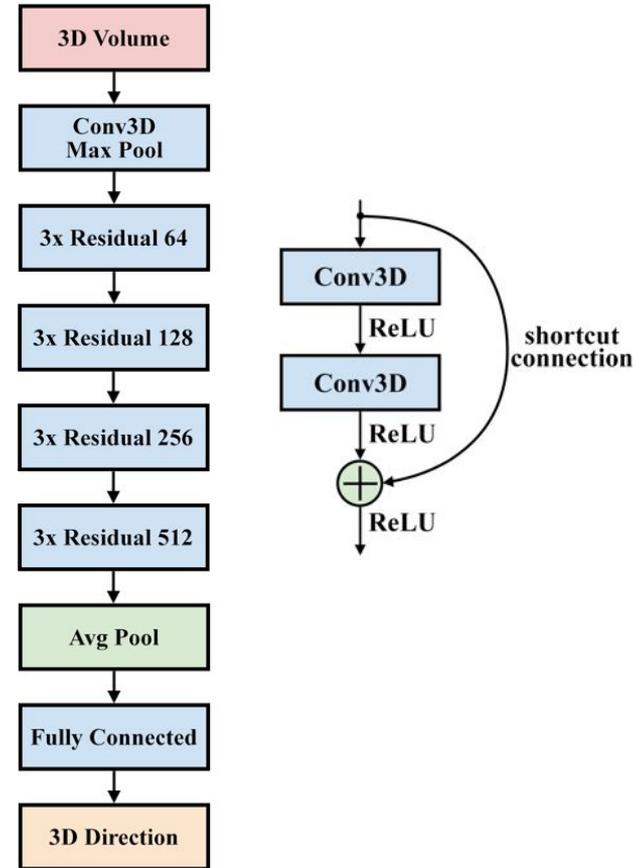


# Efficiency Improvements using Alternative Architectures

On Direction Reconstruction

# 3-D CNN Architecture

- Loss Function
  - Cosine Distance:  $\min(1 + \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \cdot \|\mathbf{b}\|}, 1 - \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \cdot \|\mathbf{b}\|})$
  - Angular resolution (angle difference) in radians/degrees:  $\arccos(\frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \cdot \|\mathbf{b}\|})$
- Training
  - 100 epochs
- Models Trained
  - Event RegCNN
    - Input: 3D full-event pixel map
    - Output: primary electron/muon prong direction
  - Prong RegCNN
    - Input: 3D prong-only pixel map
    - Output: electron/muon prong direction



# Submanifold Sparse Convolutional Network

- Submanifold Sparse Convolutional Network (SSCN) was designed to improve the training efficiency for high-dimension sparse data (Graham, 2017).
  - Perfectly suits our scenario: only a small portion of voxels in our 100x100x100 3-D pixel maps contain hits.
  - SSCN can provide similar performance while reducing the computation and memory requirements by ~50%
- Researchers in the DUNE community have already applied SSCN in their models and obtained validated improvements (Domine, 2019).

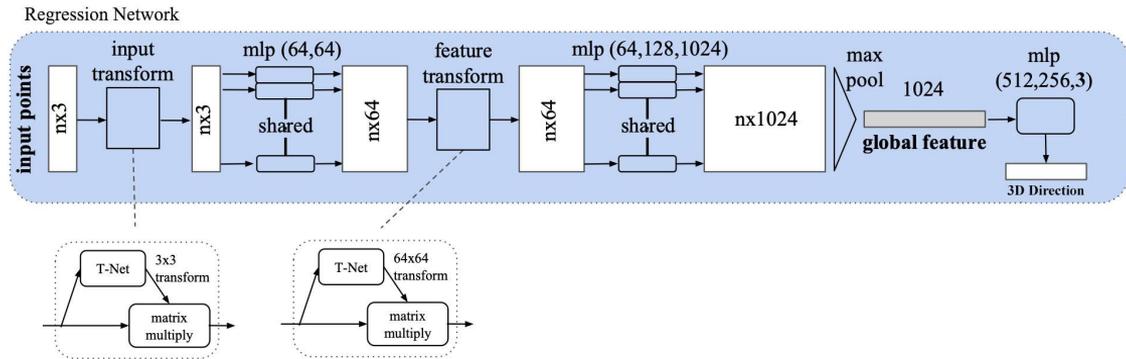
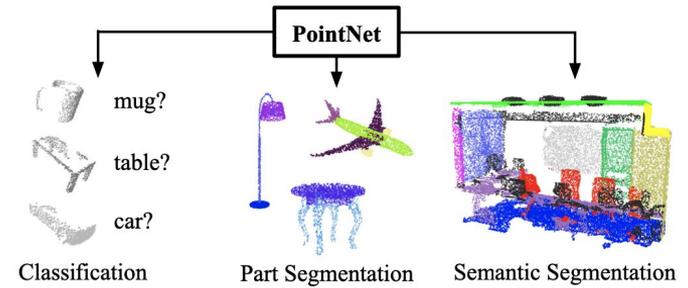


Figure 1: Example of “submanifold” dilation. **Left:** Original curve. **Middle:** Result of applying a regular  $3 \times 3$  convolution with weights  $1/9$ . **Right:** Result of applying the same convolution again. The example shows that regular convolutions substantially reduce the sparsity of the feature maps.

(Graham, 2017)

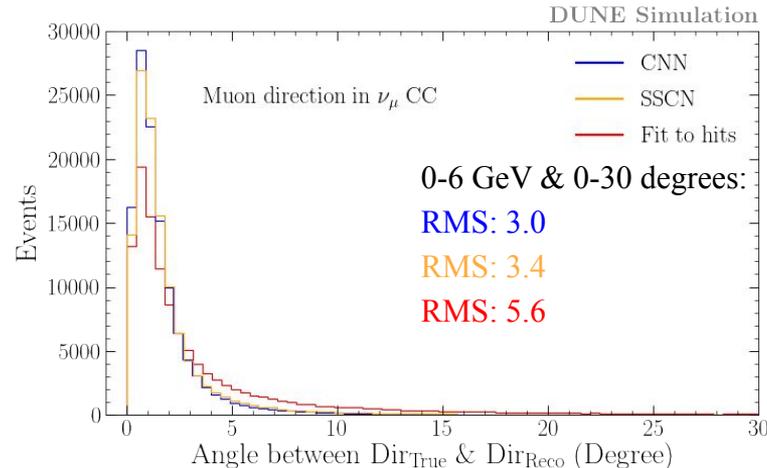
# PointNet

- PointNet is a novel type of neural network that directly consumes point clouds. (Qi, 2017)
- It is designed for 3D recognition tasks including object classification, part segmentation and semantic segmentation
- It obtains on par or better results than state of the arts on standard benchmarks.



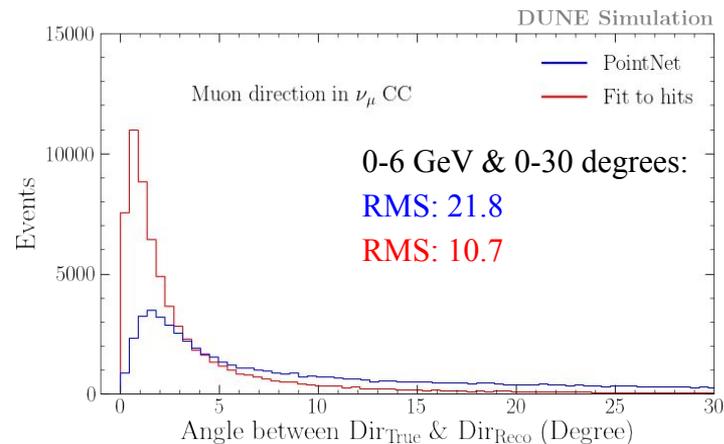
# SSCN on Direction Reconstruction

- SSCN is trained on cropped prong-only 3-D NuMuCC pixel maps
  - The 100x100x100 pixel maps are cropped to 32x32x32 centered at their vertices
- Provided similar performance compared with the regular 3-D CNN



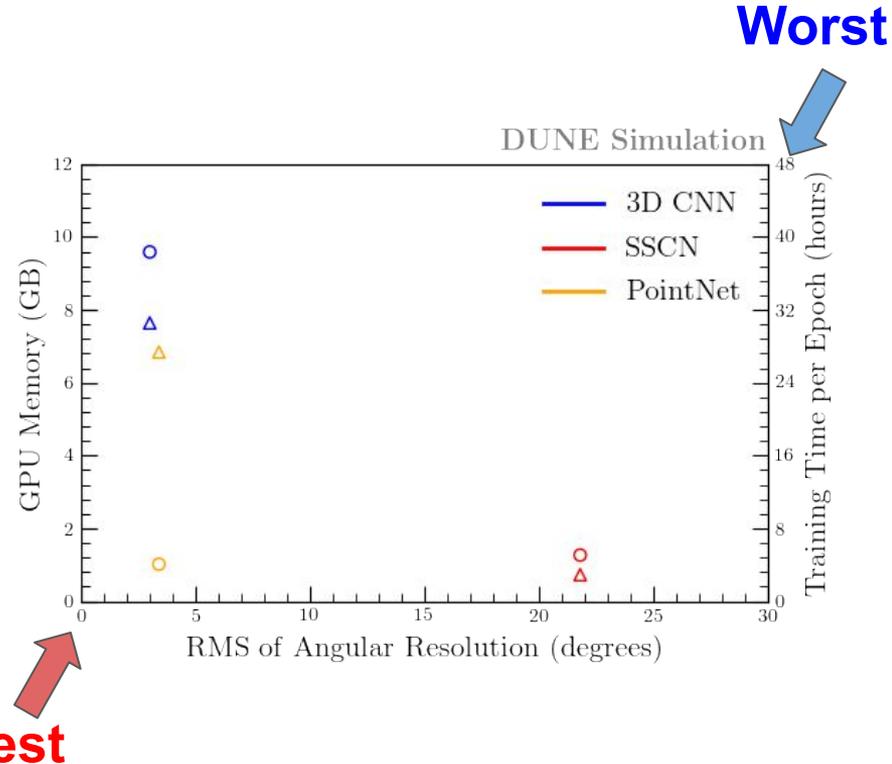
# PointNet on Direction Reconstruction

- PointNet is trained on un-cropped full-event 3-D NuMuCC pixel maps
  - 186 hits on average
- Picked 100 hits (points) with the highest energy deposit
  - PointNet requires the fixed number of points per input
  - Events having fewer than 100 hits are discarded
  - Only ~50% events (with more hits) are kept
- The performance of PointNet could be limited by the number of points per input
  - Regularly, the number of points per input should be ~2000



# Summary of Efficiency-oriented Models

- Two efficiency-oriented models have been explored
  - SSCN and PointNet
  - Both sacrifice the accuracy, in different degrees, for the time/memory efficiency
- Visualization of performance vs efficiency
  - All use un-cropped 3D pixel maps
    - 100x100x100
  - Training batch size is fixed to 16
  - The training time is sensitive to the hardware setup (I/O bandwidth, CPU, RAM, etc.)



# Current Status

- 3-D RegCNN provides comparable performance on vertex reconstruction
  - NuECC
    - Full-event RegCNN performs similarly to the standard method
    - Prong-only RegCNN performs similarly to the standard method
- Efficiency-oriented models improved the computation and memory efficiency but sacrificed the accuracy
  - Except, SSCN provides better memory efficiency to the 3-D RegCNN but doesn't sacrifice much accuracy

## Future Work

- Improve the computational efficiency of SSCN on 3-D pixel maps
- Improve the performance of vertex reconstruction using 3-D RegCNN
  - 2-stage training on fine-grained pixel maps

Thank you!