Pion Energy Reconstruction with Deep Neural Networks

Erik Buhmann[1], Gregor Kasieczka, Erika Garutti

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Bundesministerium
für Bildung
und Forschung

Friedrich Naumann
STIFTUNG FÜR DIE FREIHEIT

[1] eMail: erik.buhmann@desy.de

Overview

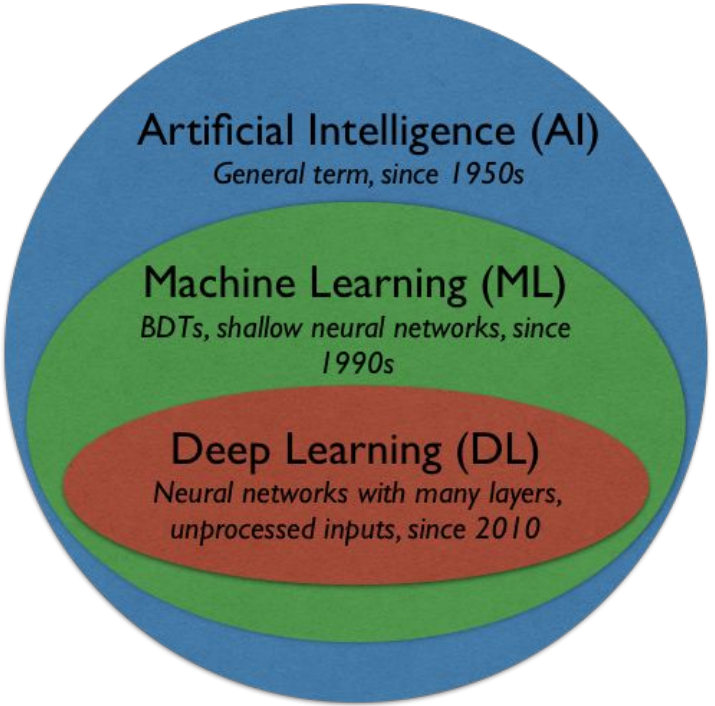
- Introduction
  - Neural Networks Basics
  - Network Architectures
- Preliminary Studies
  - Regression with 3D Convolutional Layer
  - Regression with Locally Connected Layer
- Summary & Outlook

Overview

- Introduction
  - Neural Networks Basics
  - Network Architectures
- Preliminary Studies
  - Regression with 3D Convolutional Layer
  - Regression with Locally Connected Layer
- Summary & Outlook

Terminology

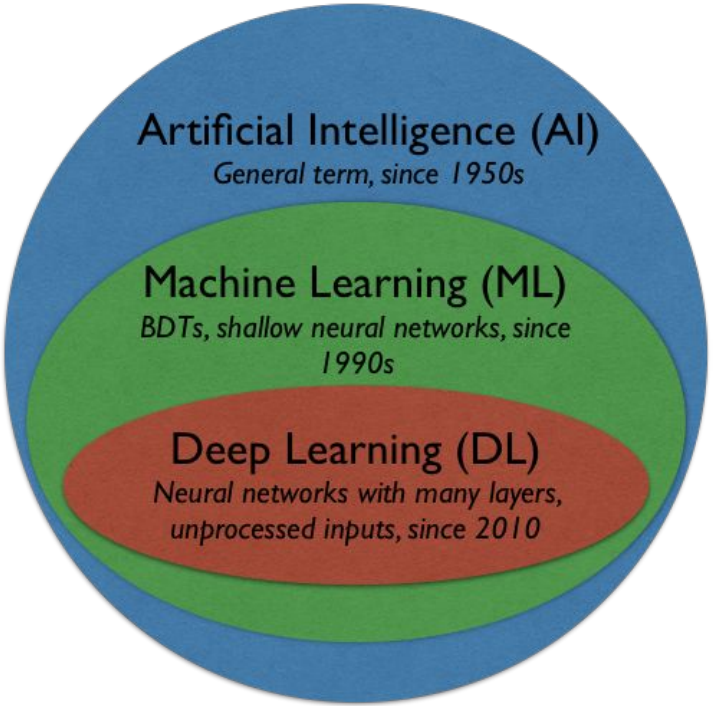
- What do we want to learn?
  - Classification (Cat or dog?)
  - Generation (New pictures of cats)
  - Regression (How old is the cat?)
  - Compression (Smaller cats)
  - ?
- What do we have available?
  - Labelled examples (supervised learning)
  - Limited labelled examples (weakly supervised training)
  - No examples (unsupervised training)



[1] Slide by Gregor Kasieczka

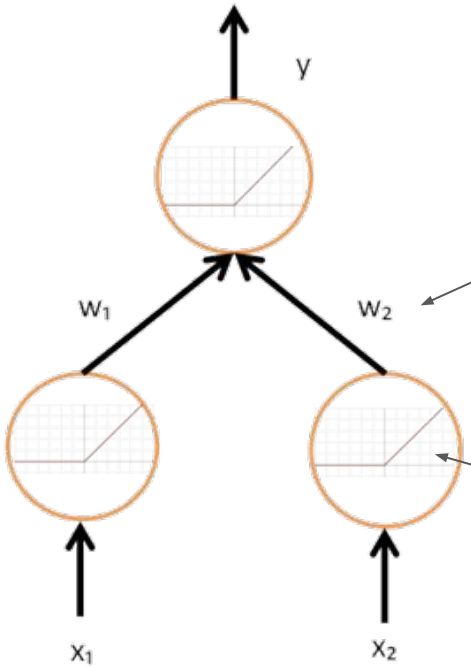
Terminology

- What do we want to learn?
  - Classification (Cat or dog?)
  - Generation (New pictures of cats)
  - **Regression (How old is the cat?)**
  - Compression (Smaller cats)
  - ?
- What do we have available?
  - **Labelled examples (supervised learning)**
  - Limited labelled examples (weakly supervised training)
  - No examples (unsupervised training)



[1] Slide by Gregor Kasieczka

A Basic Neural Network



Trainable weights
(expressing importance of connection)

$$y = f(f(x_1)w_1 + f(x_2)w_2)$$

$$f(x) = \Theta(x) \cdot x$$

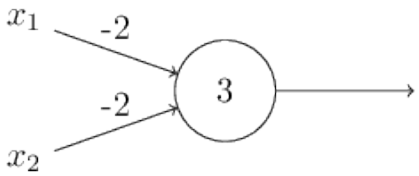
Activation function

[1] Slide by Gregor Kasieczka

Simple NN as logic circuit

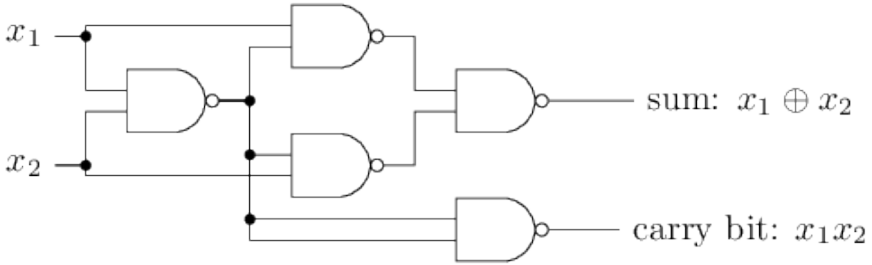
NAND logic function with "neuron":

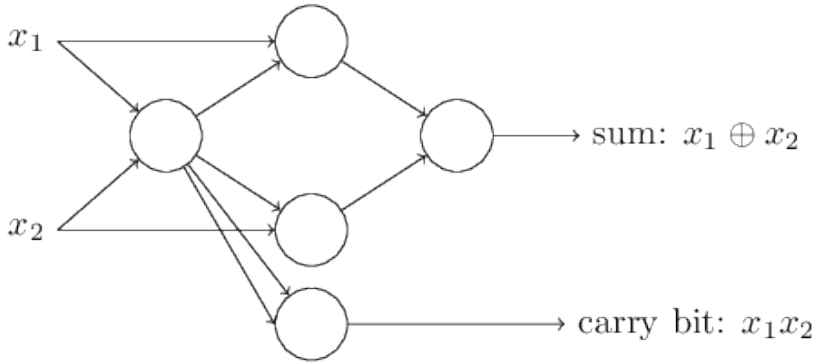
Weights w_1, w_2 = -2
Bias b = 3



Assuming binary input:

x1	x2	$(x_1w_1+x_2w_2)+b$	Out
0	0	3	1
0	1	1	1
1	0	1	1
1	1	-1	0





➔ A large enough neural network with correctly tuned (trained!) weights can model any decision making process

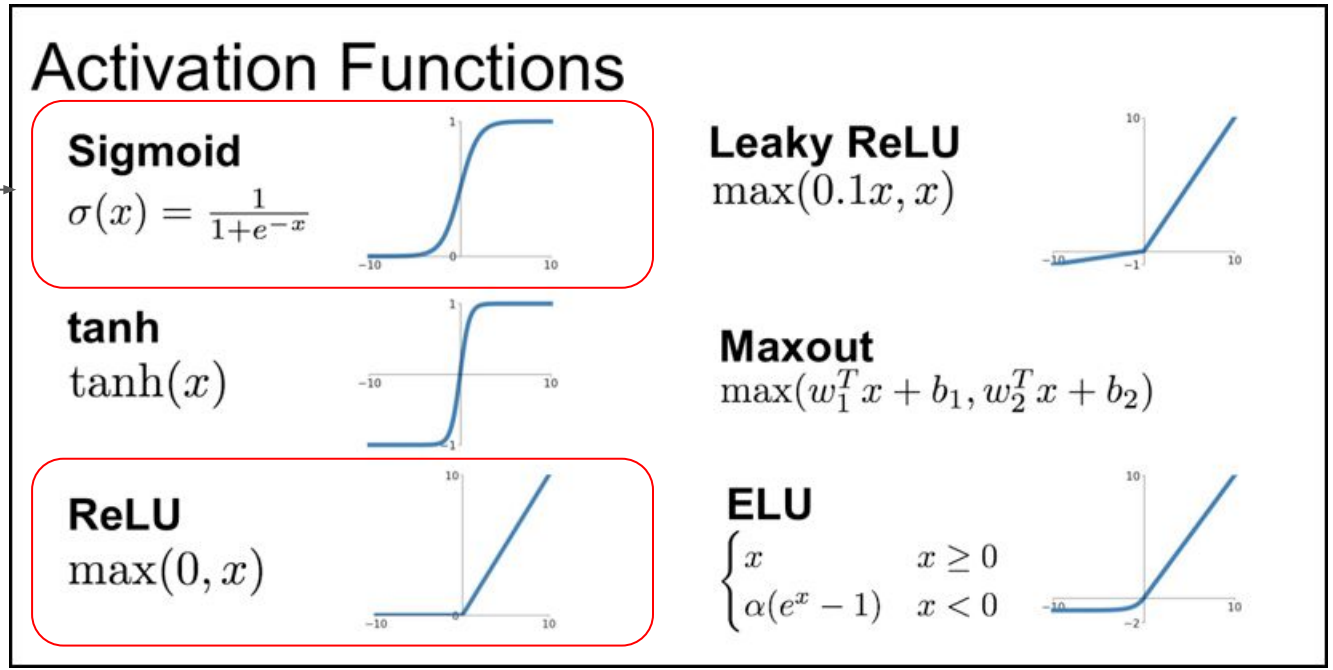
[1] http://neuralnetworksanddeeplearning.com

Activation Functions

Computation performed in each neuron - Examples:

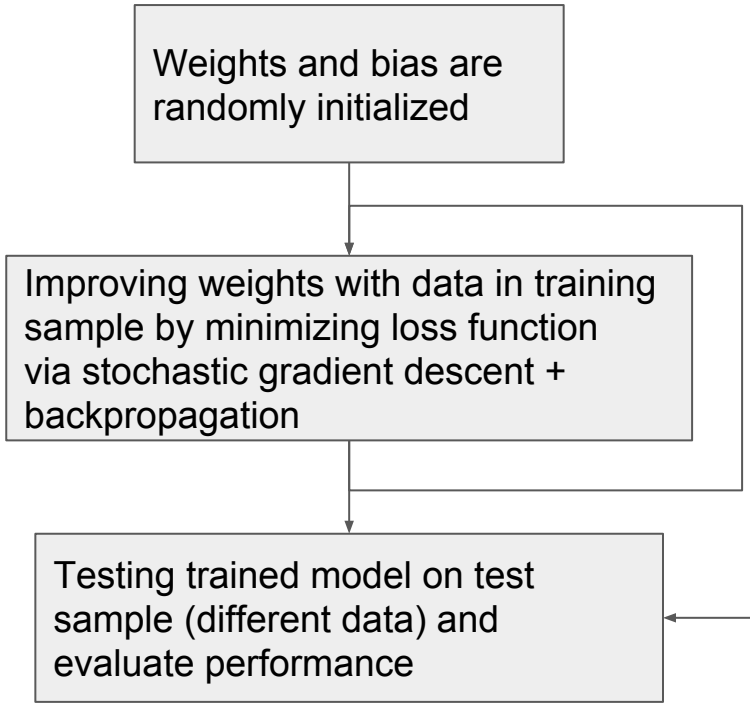
Historical:

Currently popular:

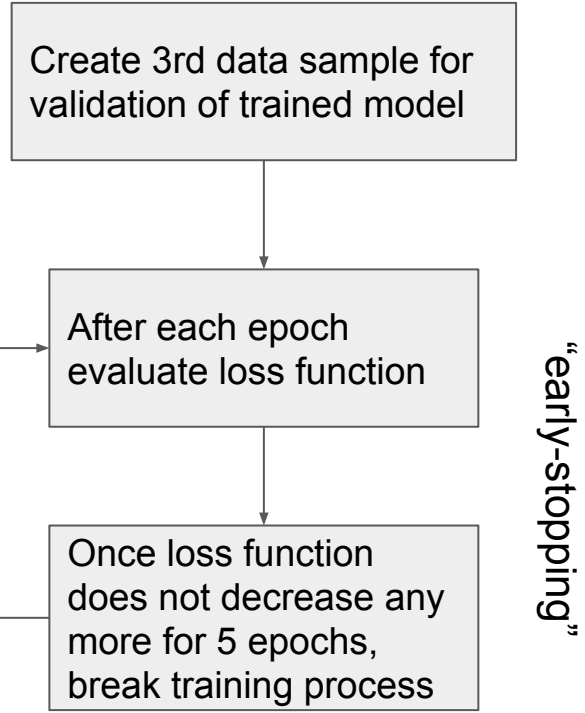


Learning Process

General process:



To avoid overfitting:



Loss function

Training by minimizing of loss function through adjusting weights and biases in every training step

Example 1) Mean Squared Error:

$$L(E_{i,true}, E_{i,pred}) = \frac{1}{N} \sum_i \left(E_{i,pred} - E_{i,true}\right)^2$$

Example 2) Mean Squared Relative Error:

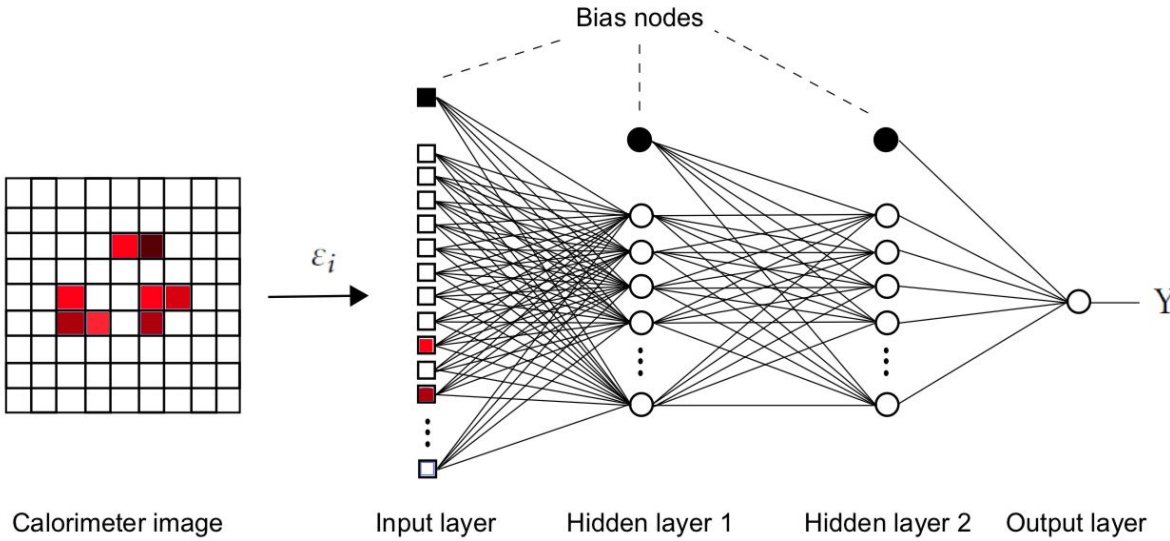
$$L(E_{i,true}, E_{i,pred}) = \frac{1}{N} \sum_i \left(\frac{E_{i,pred} - E_{i,true}}{E_{i,true}}\right)^2$$

Architecture

- Network based on connection of different layers and layer types
- Layer types we used so far:
  - Fully Connected layer
  - Locally Connected layer
  - Convolutional layer

Fully Connected Layer

- “Classical” Artificial Neural Network (ANN)
- Every node of each layer connected to every node of neighboring layers
- Many trainable weights
  - During training connections are strengthened or weakened



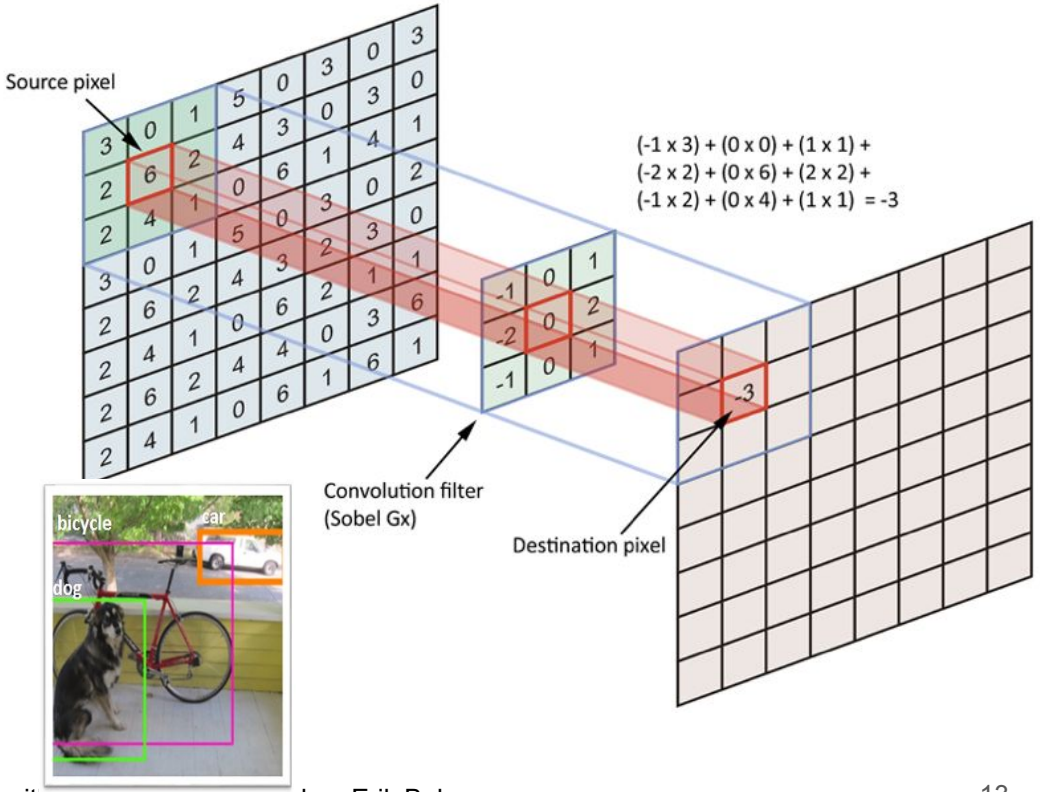
[1] *Almeida et al,* Playing Tag with ANN: Boosted Top Identification with Pattern Recognition, *arXiv:1501.05968*

Locally Connected Layer

- Output value of convolutional kernel:

$$O_{conv} = (w_1 E_1 + w_2 E_2 + w_3 E_3 + ...) + b$$

- Unshared weights
  - New kernel (here 3 x 3) is used at every position
  - Multiple kernels used to learn different features of the image
  - Could a (1 x 1) kernel be used to learn single calorimeter channel calibration?

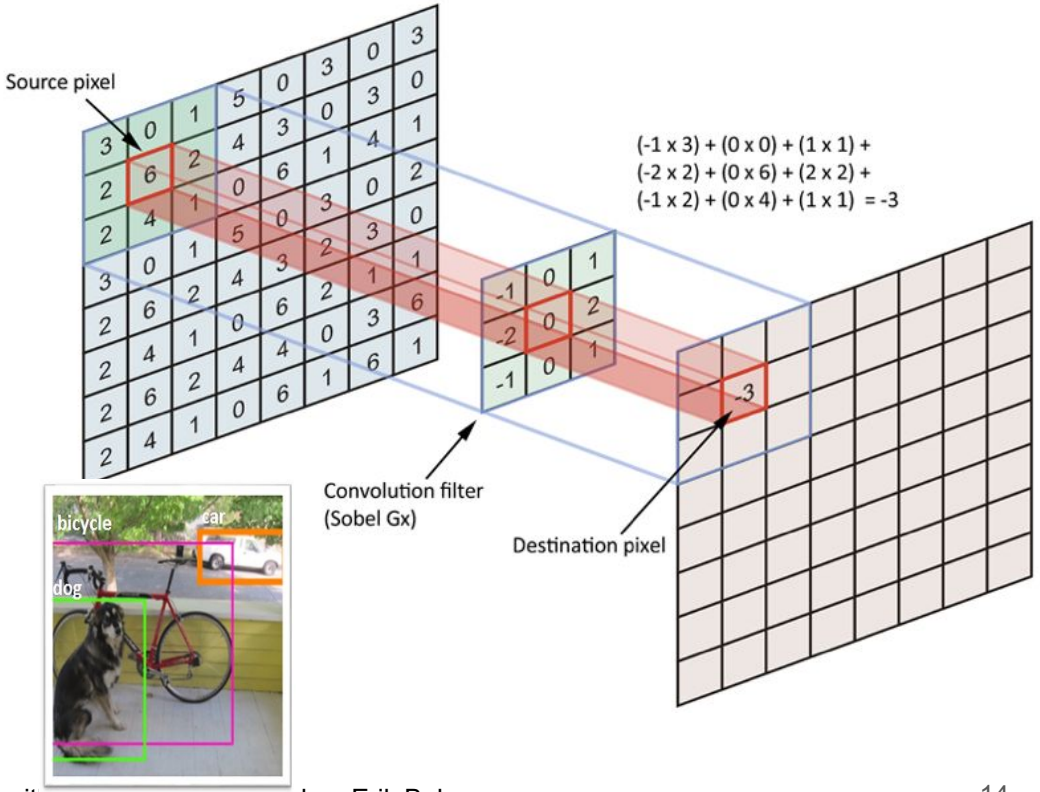


Convolutional Layer

- Output value of convolutional kernel:

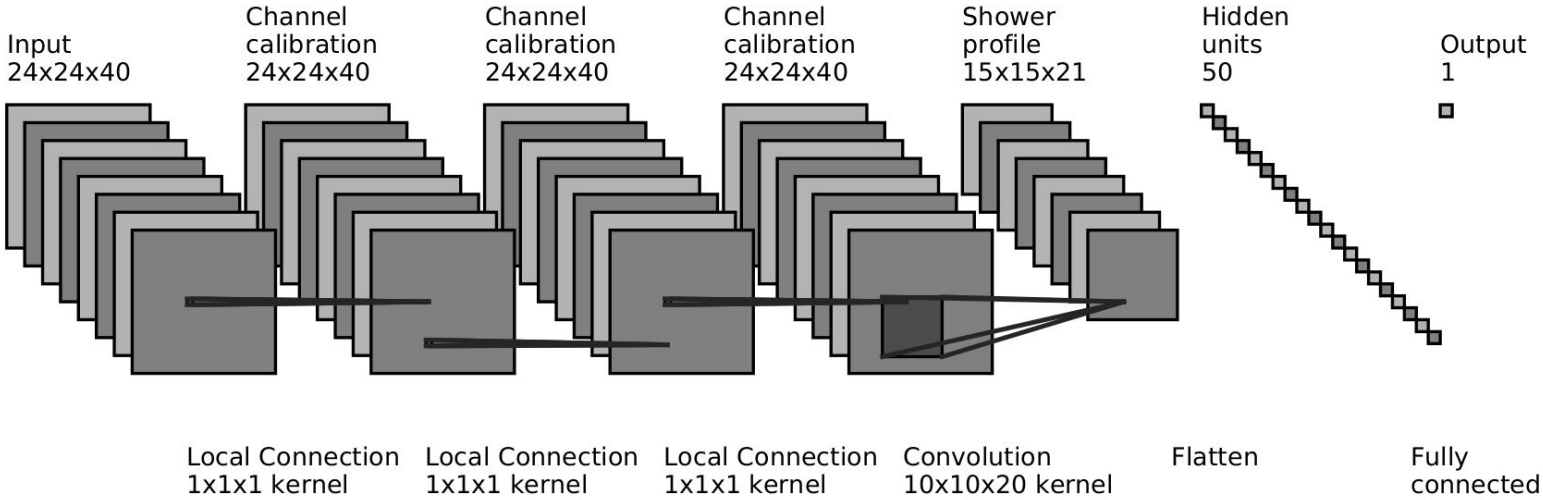
$$O_{conv} = (w_1 E_1 + w_2 E_2 + w_3 E_3 + ...) + b$$

- Shared weights
  - Same kernel (here 3 x 3) is used at every position
    - Less trainable weights
  - Multiple kernels used to learn different features of the image
  - Could a convolutional kernel be used to learn particle shower features in calorimeter?



Goal: Deep Neural Network

- Connecting multiple locally connected, convolutional and fully connected layer
- Physics based approach:
  - Locally Connected layer to find channel calibration
  - Convolutional filter to identify shower profile



Overview

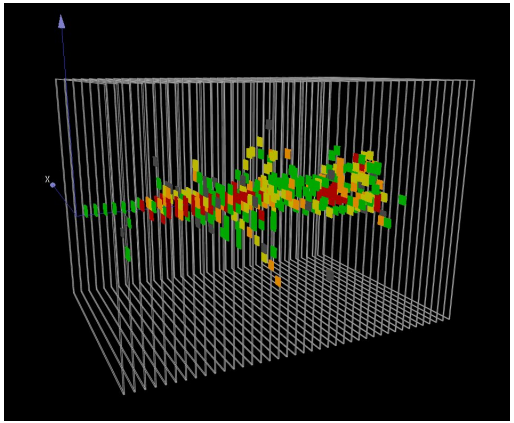
- Introduction
  - Neural Networks Basics
  - Network Architectures
- Preliminary Studies
  - Regression with 3D Convolutional Layer
  - Regression with Locally Connected Layer
- Summary & Outlook

Technicalities

- Network programmed in Python
  - Keras as high-level neural network API
  - Theano or tensorflow as back-end
- Training on Maxwell Cluster at DESY
  - Nodes with NVIDIA Tesla P100 GPU, 16GB RAM

<u>Input data for network:</u>

- Pion data from May testbeam @ SPS
- “Event display images” as (24,24,40) arrays
  - With hit energies at (I,J,K) coordinates
- E_{sum} > 200 MIP to cut Muon contamination

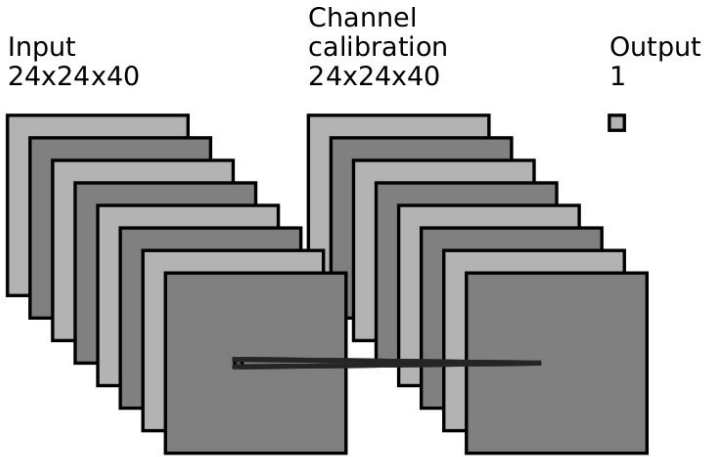


Regression task

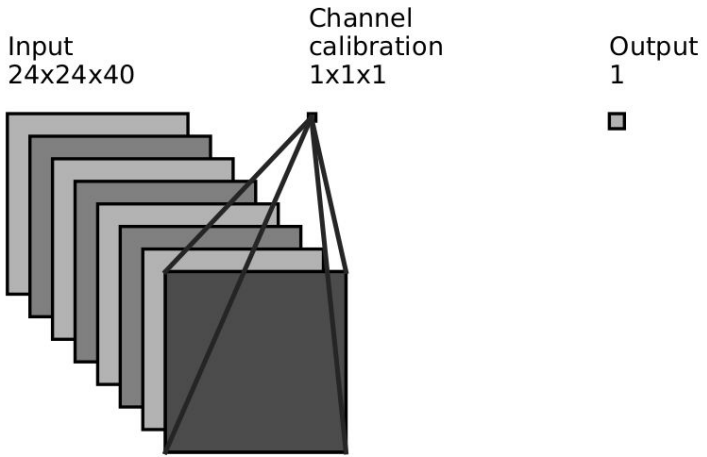
Plon energy [GeV]	Events in training sample	Events in validation sample	Events in test sample
10	12,000	4,000	4,000
15	0	0	4,000
20	12,000	4,000	4,000
30	0	0	4,000
40	12,000	4,000	4,000
50	0	0	4,000
60	12,000	4,000	4,000
80	0	0	4,000
100	12,000	4,000	4,000
120	0	0	4,000
160	12,000	4,000	4,000

Network Architectures

With locally connected layer:

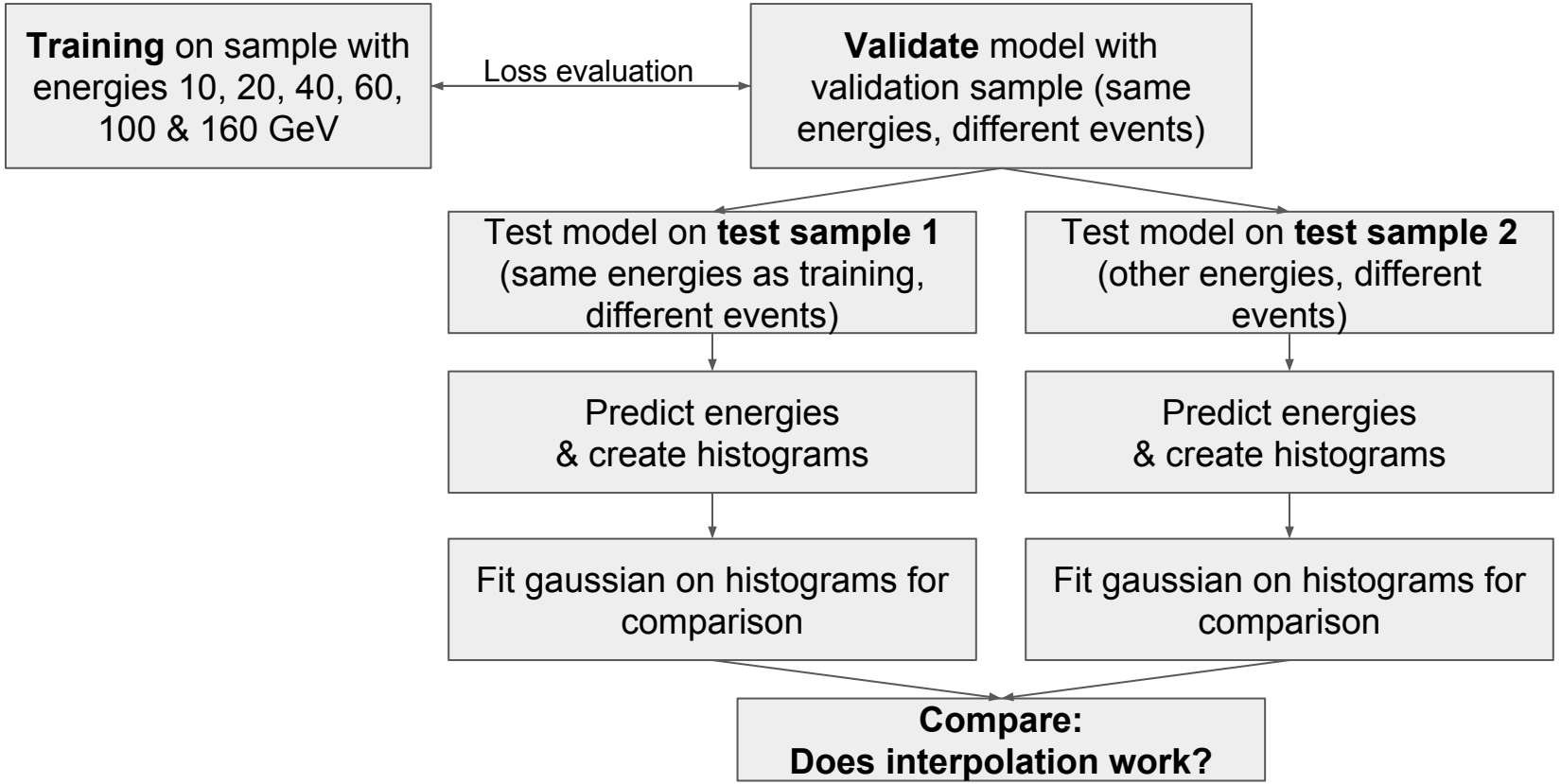


With 3D convolutional layer:

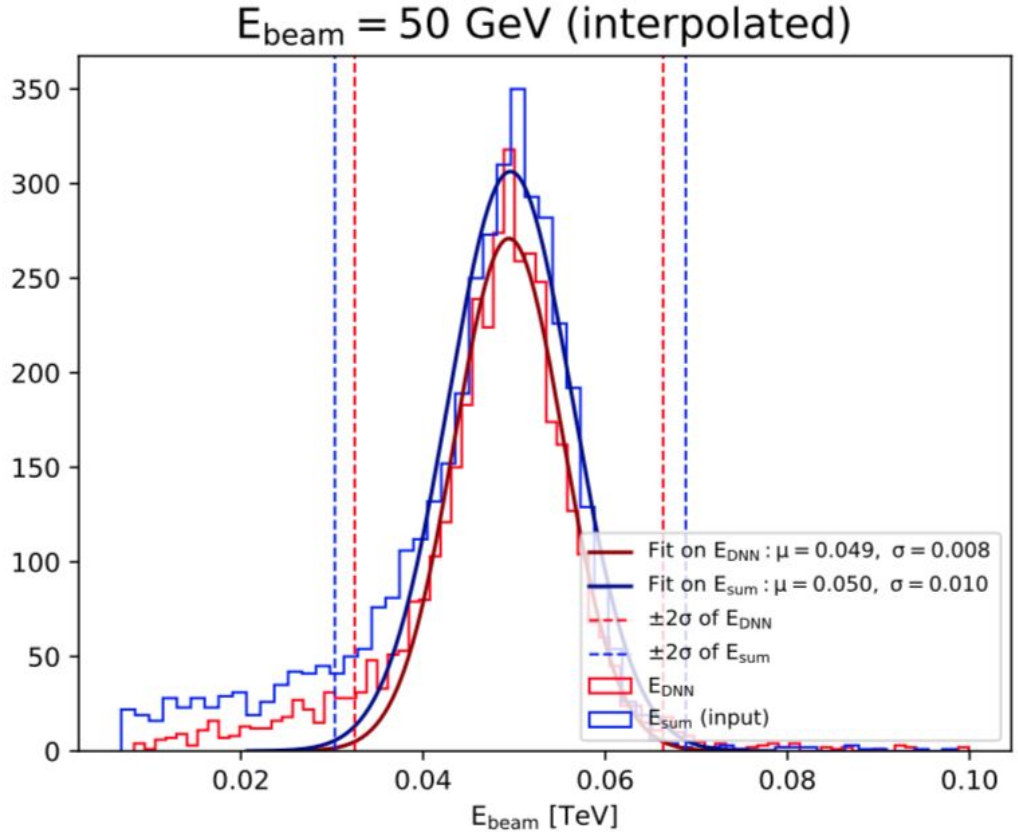


Both with loss function: Mean Squared Relative Error (MSRE)

Training & Testing Process

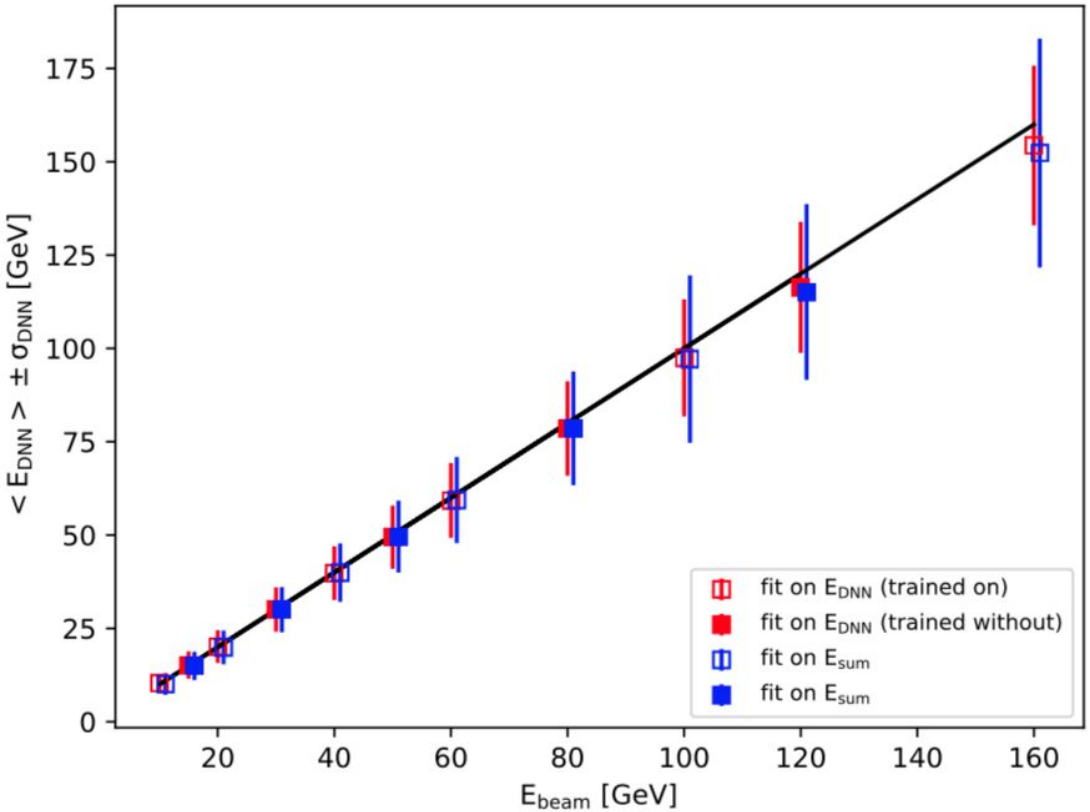


Reconstructed Energies



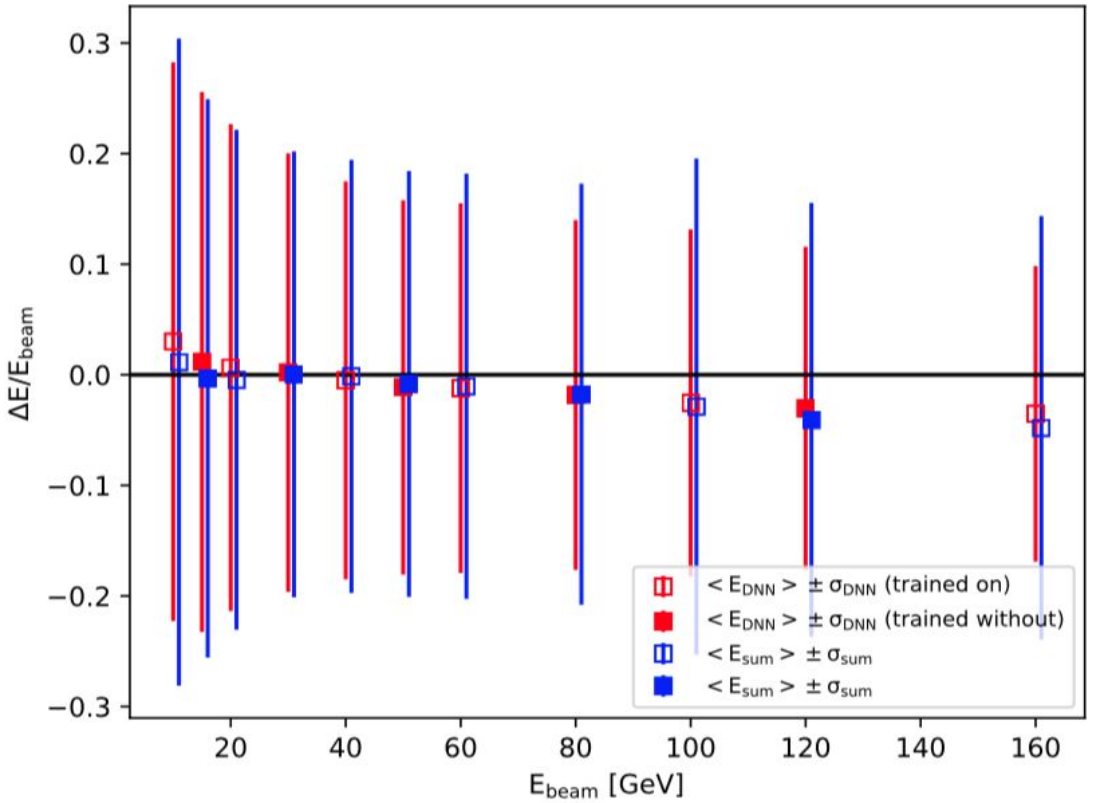
- ➔ Gaussian fit on histogram of reconstructed energy
- ➔ Comparison with original energy sum as “sanity check”
  - ◆ Based on a linear MIP-to-GeV factor of 29.7
- ➔ Histogram of reconstructed energy is always smaller than of energy sum
- ➔ Histograms and fits created for all energies of both test samples
- ➔ Tails through leakage (?)

Locally Connected Architecture



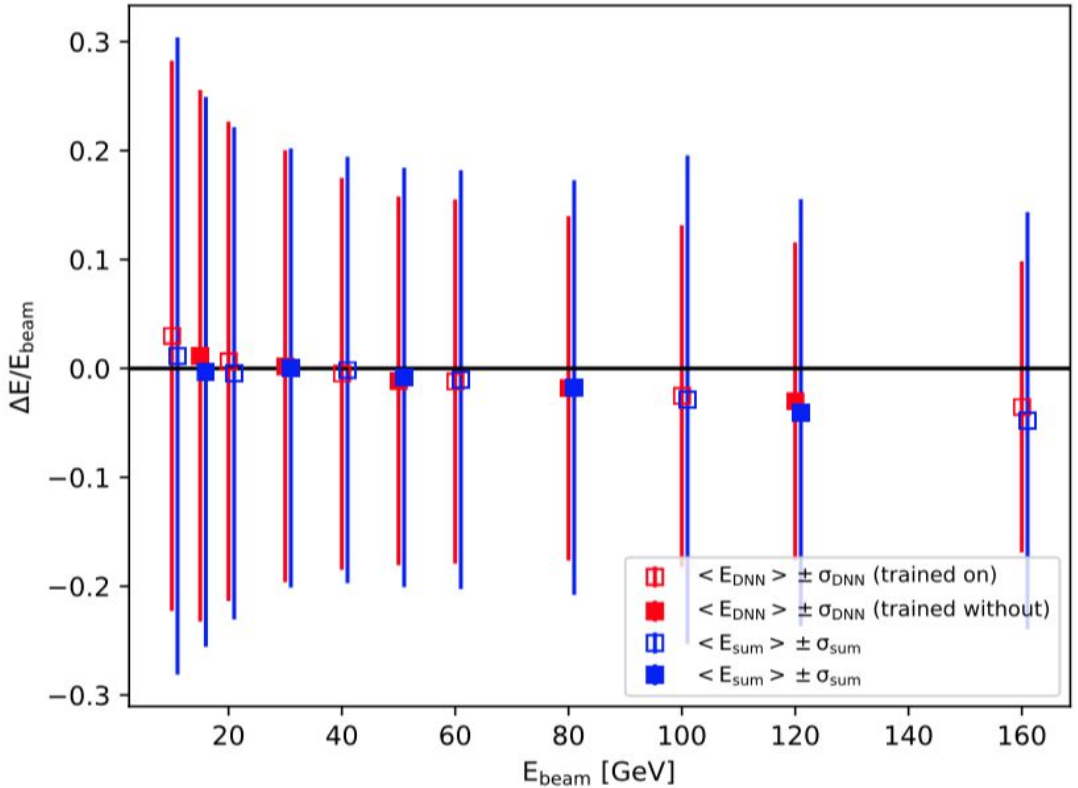
- ➔ Overview of fit parameters for all energies
- ➔ Error bars represent ±1σ intervall
- ➔ Both training samples
  - ◆ No systematic difference observable

Locally Connected Architecture



- ➔ No systematic difference between trained on and interpolated reconstructed energies
- ➔ σ for E_{reco} overall smaller than for E_{sum}
  - ◆ Network performance better than simple energy fit
- ➔ Simple network learns MIP-to-GeV calibration
- ➔ Per channel linear calibration is learned

Convolutional Architecture



➔ Similar performance to Locally Connected layer

Overview

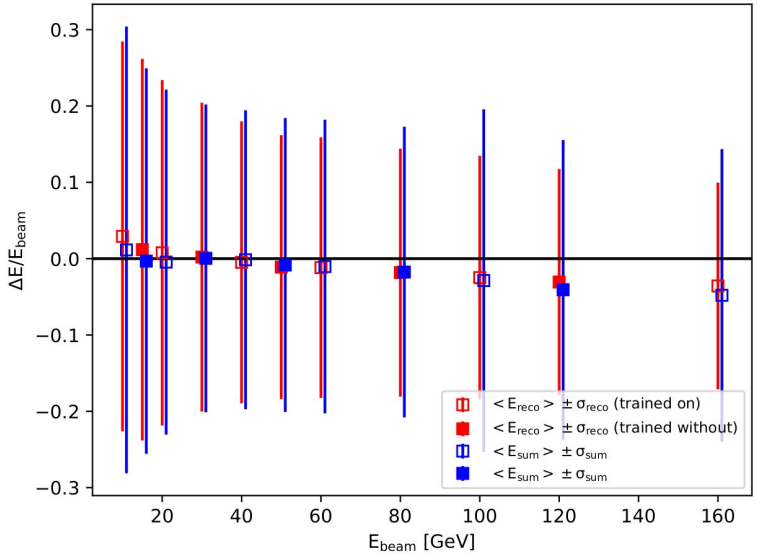
- Introduction
  - Neural Networks Basics
  - Network Architectures
- Preliminary Studies
  - Regression with 3D Convolutional Layer
  - Regression with Locally Connected Layer
- Summary & Outlook

Summary

- Conversion of TB data into file format for ML works
- Preliminary studies with energy reconstruction
- Energy interpolation learned by network
- Trained solely on real data, no MC simulations used
- MIP-to-GeV conversion learned by network
- Shallow architecture for easy understanding of network output
  - 3D convolutional layer to learn shower features
  - Locally Connected layer to learn calibration values for single channels

Outlook

- Implementation of deeper network architecture
- Energy reconstruction + Particle ID
- Training with MC simulation
- Usage of time information
- Studies with uncalibrated ADC data

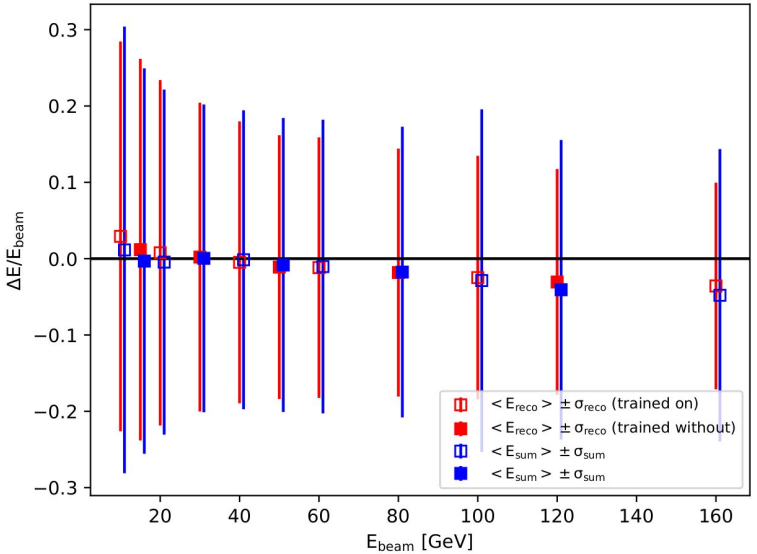


Outlook

- Implementation of deeper network architecture
- Energy reconstruction + Particle ID
- Training with MC simulation
- Usage of time information
- Studies with uncalibrated ADC data

Thank you!

- And more ideas?



Bonus Slides

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Bundesministerium
für Bildung
und Forschung

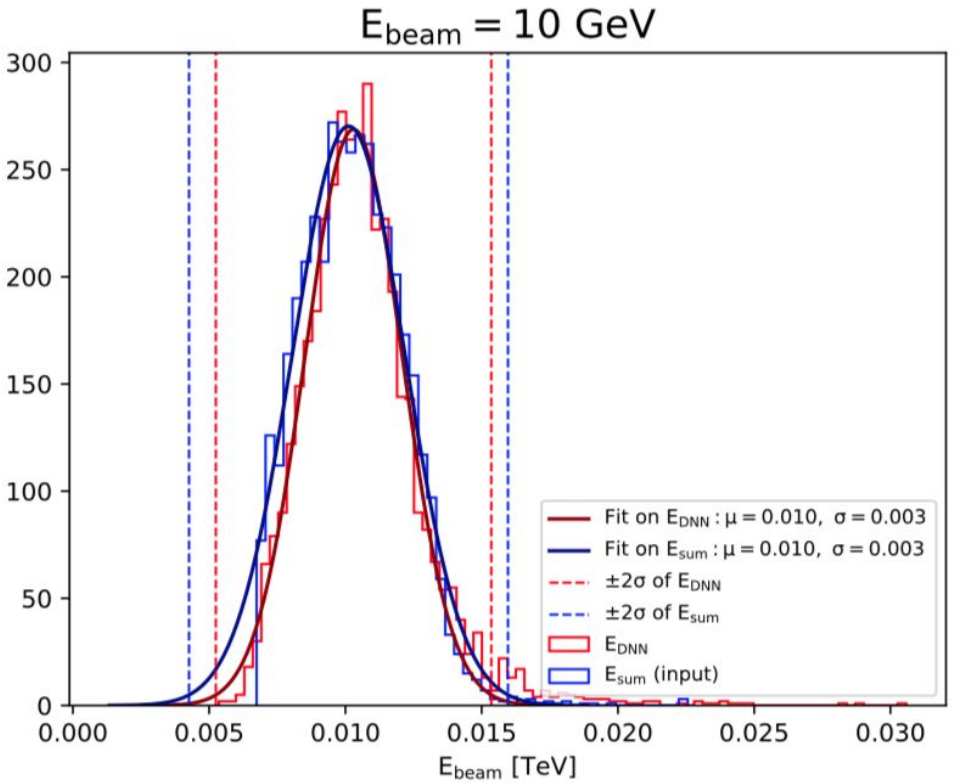
CALICE
Calorimeter for ILC

Friedrich Naumann
STIFTUNG FÜR DIE FREIHEIT

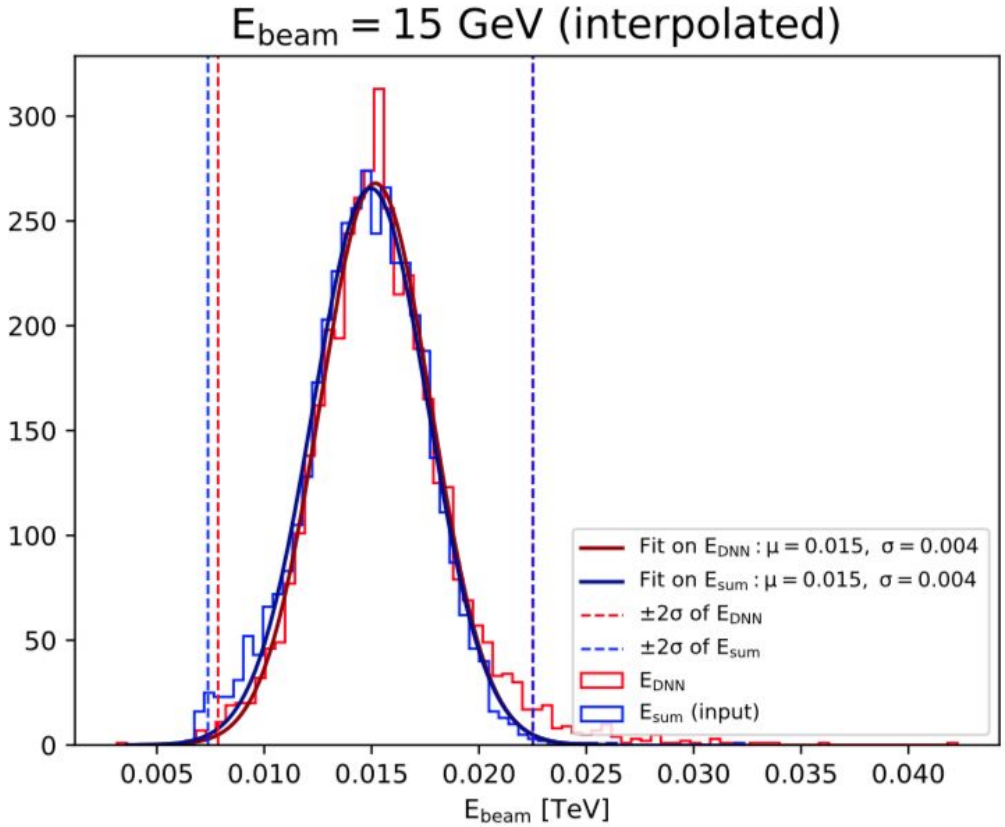
ALL THE OTHER HISTOGRAMS FOR EVERY ENERGY

FOR BOTH ARCHITECTURES

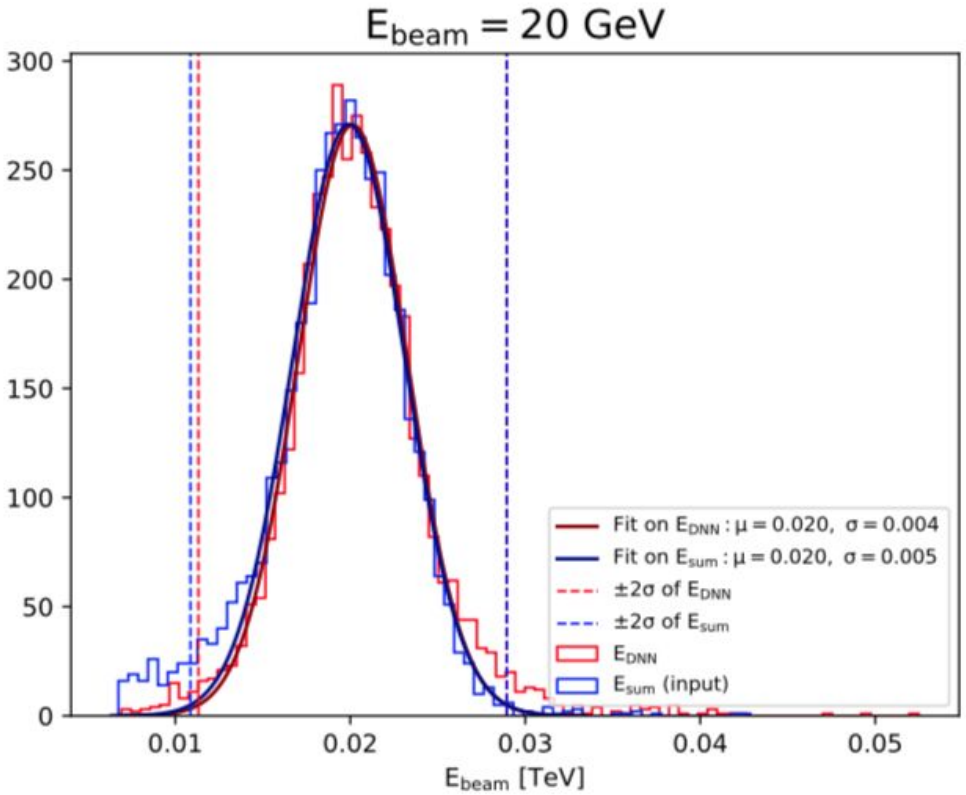
3D Conv


$E_{beam} = 10$ GeV
300
250
200
150
100
50
0
0.000
0.005
0.010
0.015
0.020
0.025
0.030
E_{beam} [TeV]
Fit on E_{DNN} : $\mu = 0.010$, $\sigma = 0.003$
Fit on E_{sum} : $\mu = 0.010$, $\sigma = 0.003$
$\pm 2\sigma$ of E_{DNN}
$\pm 2\sigma$ of E_{sum}
E_{DNN}
E_{sum} (input)

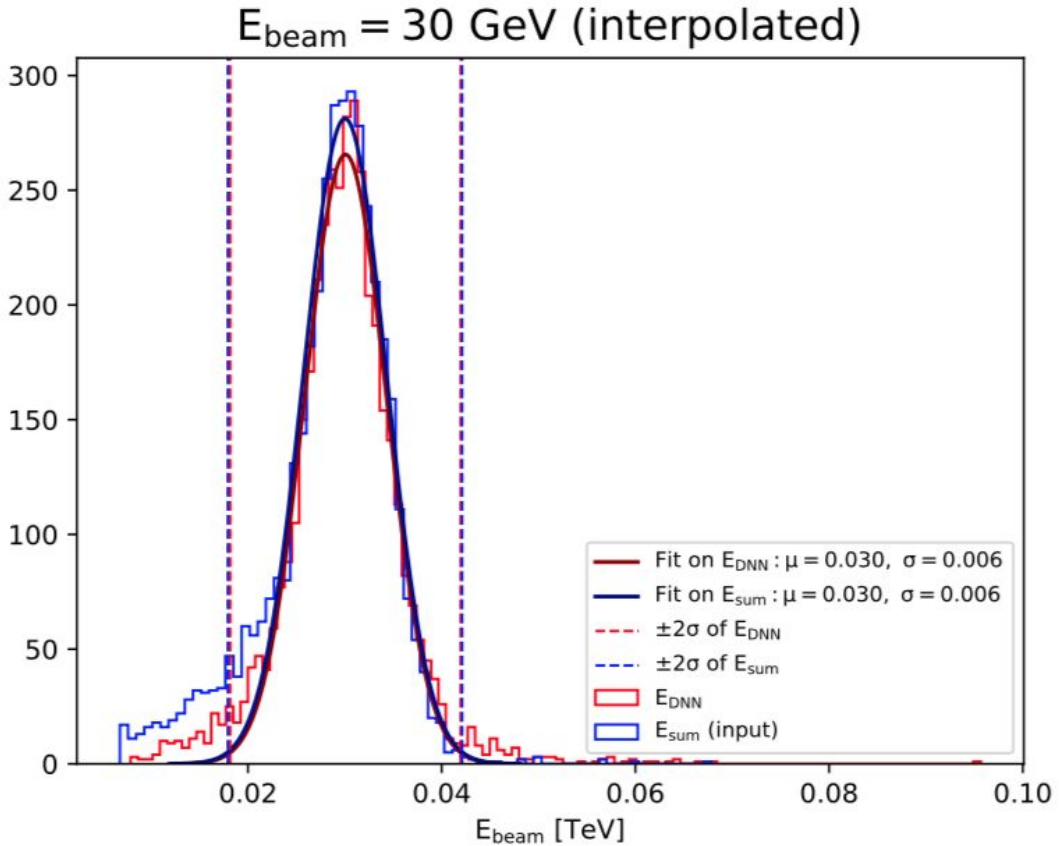
3D Conv


E_{beam} = 15 GeV (interpolated)
300
250
200
150
100
50
0
0.005
0.010
0.015
0.020
0.025
0.030
0.035
0.040
E_{beam} [TeV]
Fit on E_{DNN} : μ = 0.015, σ = 0.004
Fit on E_{sum} : μ = 0.015, σ = 0.004
±2σ of E_{DNN}
±2σ of E_{sum}
E_{DNN}
E_{sum} (input)

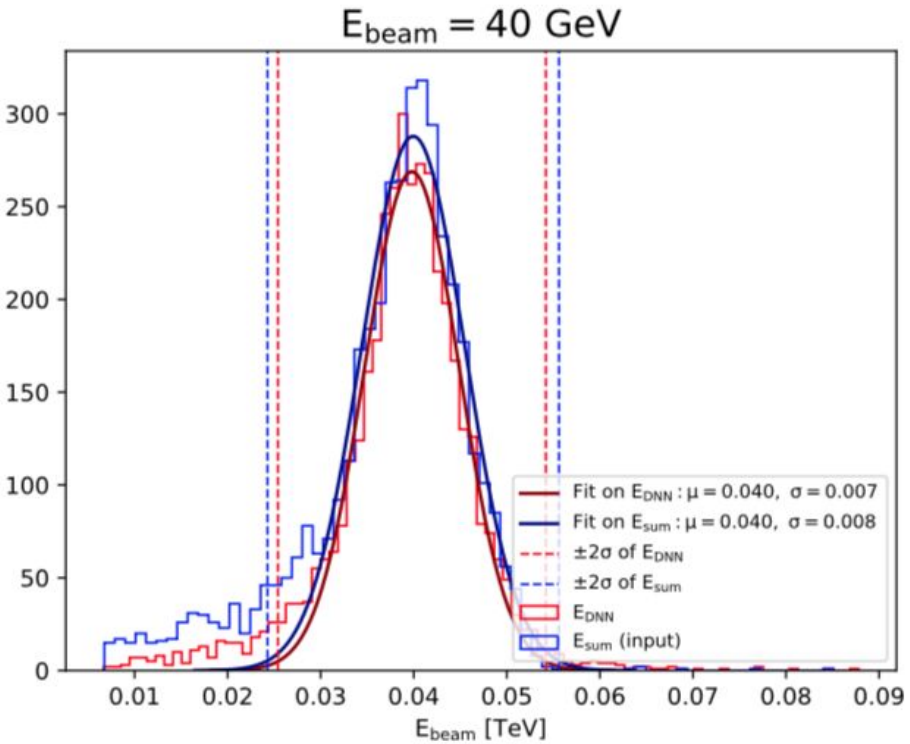
3D Conv


$E_{beam} = 20$ GeV
300
250
200
150
100
50
0
0.01
0.02
0.03
0.04
0.05
E_{beam} [TeV]
Fit on E_{DNN} : $\mu = 0.020$, $\sigma = 0.004$
Fit on E_{sum} : $\mu = 0.020$, $\sigma = 0.005$
$\pm 2\sigma$ of E_{DNN}
$\pm 2\sigma$ of E_{sum}
E_{DNN}
E_{sum} (input)

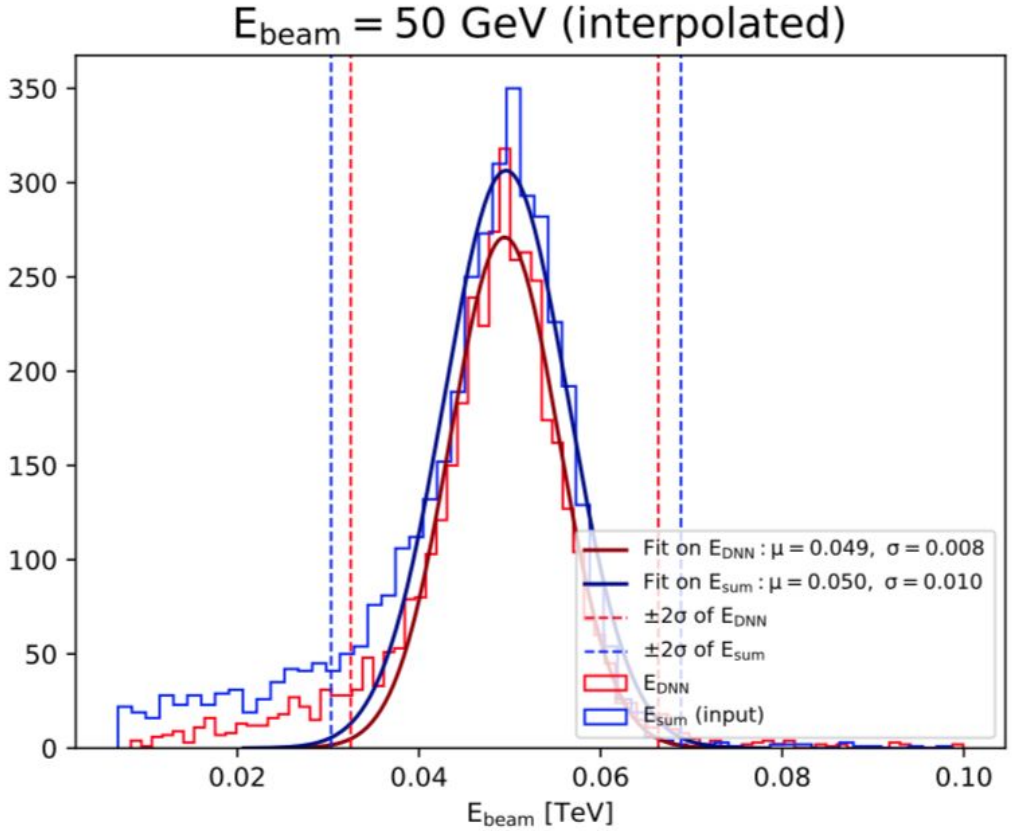
3D Conv


E_{beam} = 30 GeV (interpolated)
300
250
200
150
100
50
0
0.02
0.04
0.06
0.08
0.10
E_{beam} [TeV]
Fit on E_{DNN} : μ = 0.030, σ = 0.006
Fit on E_{sum} : μ = 0.030, σ = 0.006
±2σ of E_{DNN}
±2σ of E_{sum}
E_{DNN}
E_{sum} (input)

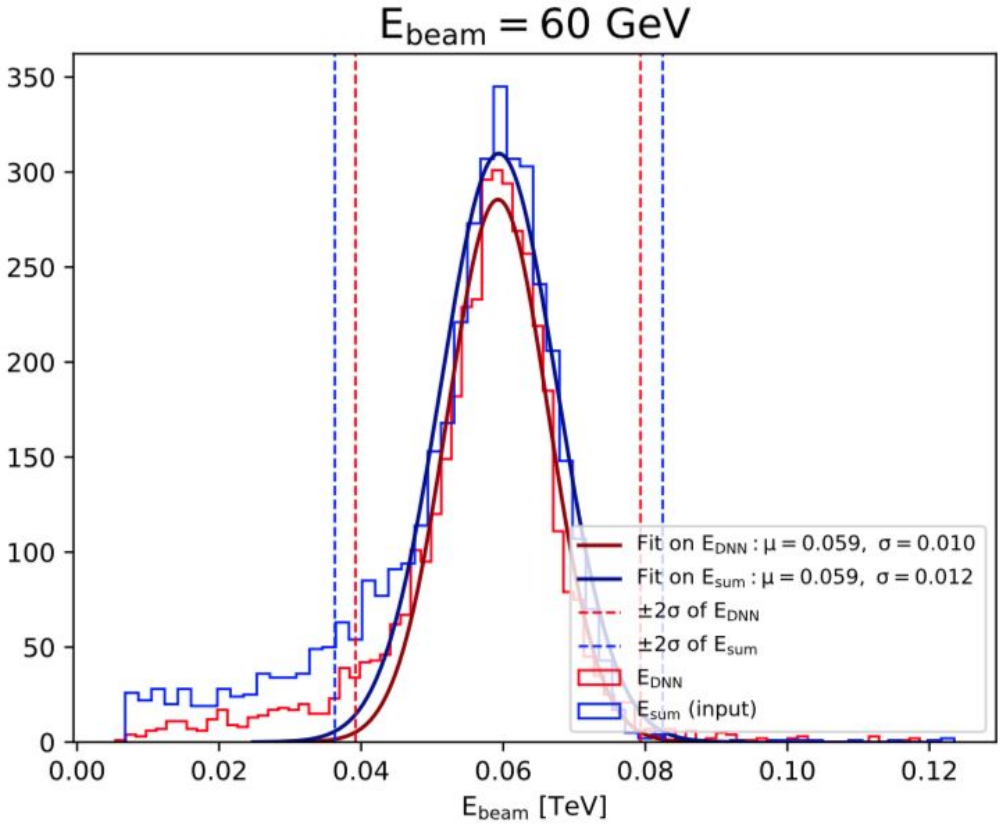
3D Conv


$E_{beam} = 40$ GeV
300
250
200
150
100
50
0
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
0.09
E_{beam} [TeV]
Fit on E_{DNN} : $\mu = 0.040$, $\sigma = 0.007$
Fit on E_{sum} : $\mu = 0.040$, $\sigma = 0.008$
$\pm 2\sigma$ of E_{DNN}
$\pm 2\sigma$ of E_{sum}
E_{DNN}
E_{sum} (input)

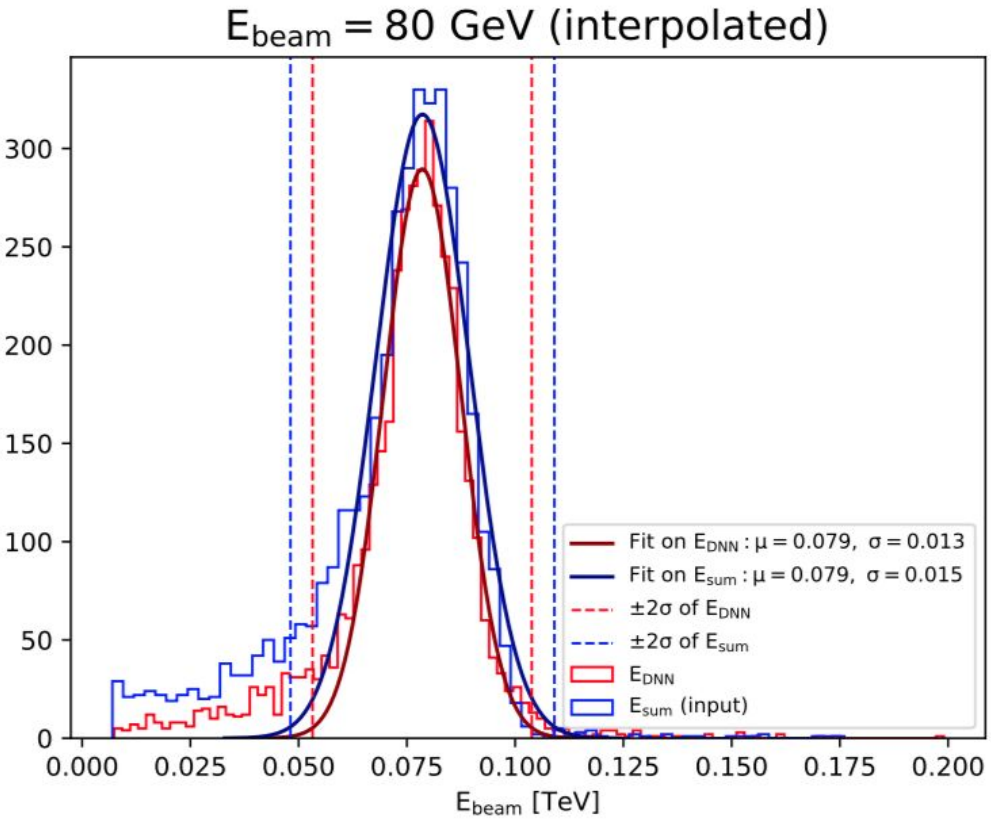
3D Conv


E_{beam} = 50 GeV (interpolated)
350
300
250
200
150
100
50
0
0.02
0.04
0.06
0.08
0.10
E_{beam} [TeV]
Fit on E_{DNN} : μ = 0.049, σ = 0.008
Fit on E_{sum} : μ = 0.050, σ = 0.010
±2σ of E_{DNN}
±2σ of E_{sum}
E_{DNN}
E_{sum} (input)

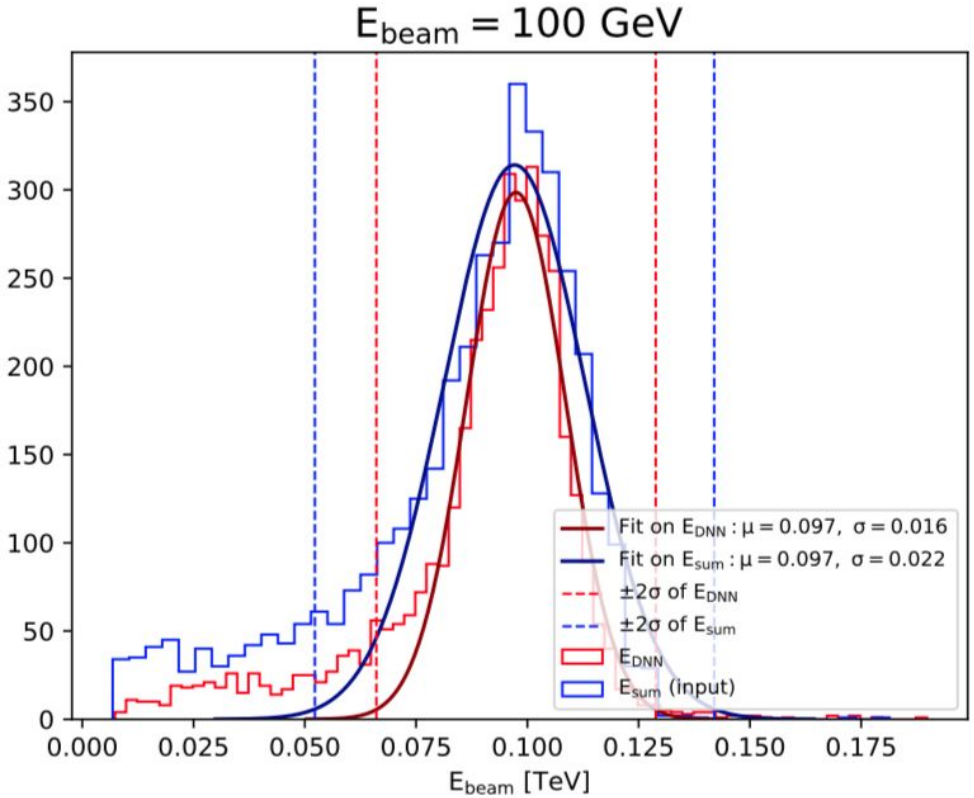
3D Conv


$E_{beam} = 60$ GeV
350
300
250
200
150
100
50
0
0.00
0.02
0.04
0.06
0.08
0.10
0.12
E_{beam} [TeV]
Fit on E_{DNN} : $\mu = 0.059$, $\sigma = 0.010$
Fit on E_{sum} : $\mu = 0.059$, $\sigma = 0.012$
$\pm 2\sigma$ of E_{DNN}
$\pm 2\sigma$ of E_{sum}
E_{DNN}
E_{sum} (input)

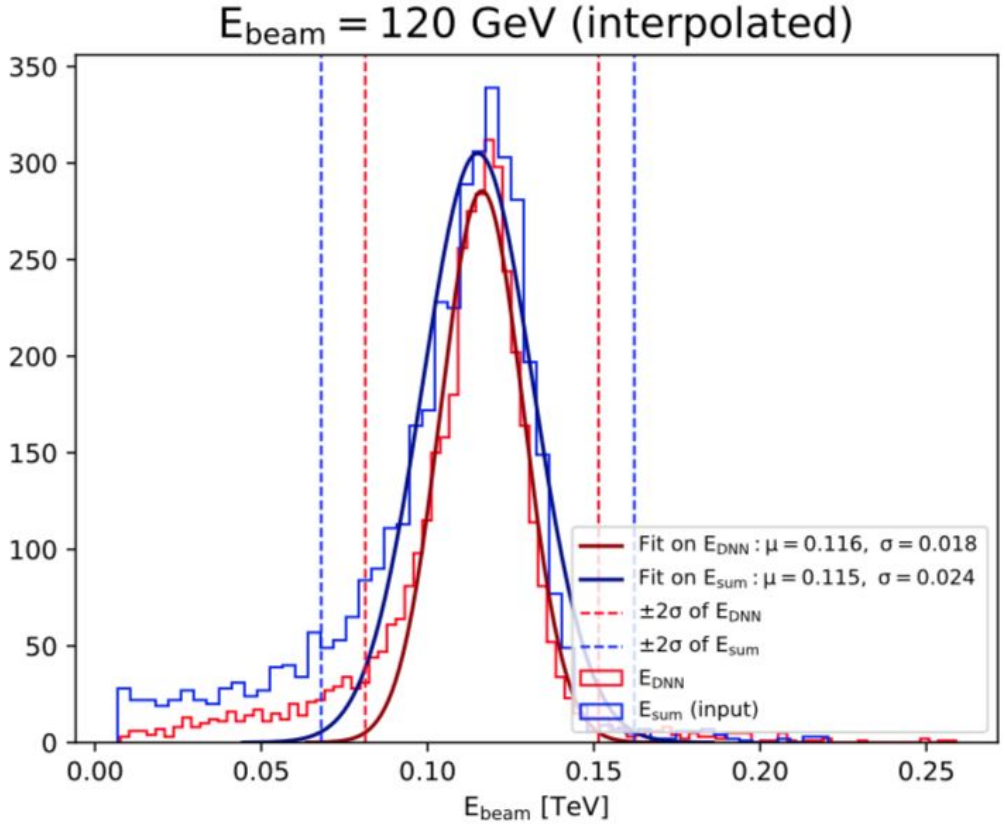
3D Conv


$E_{beam} = 80$ GeV (interpolated)
300
250
200
150
100
50
0
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0.200
E_{beam} [TeV]
Fit on E_{DNN} : $\mu = 0.079$, $\sigma = 0.013$
Fit on E_{sum} : $\mu = 0.079$, $\sigma = 0.015$
$\pm 2\sigma$ of E_{DNN}
$\pm 2\sigma$ of E_{sum}
E_{DNN}
E_{sum} (input)

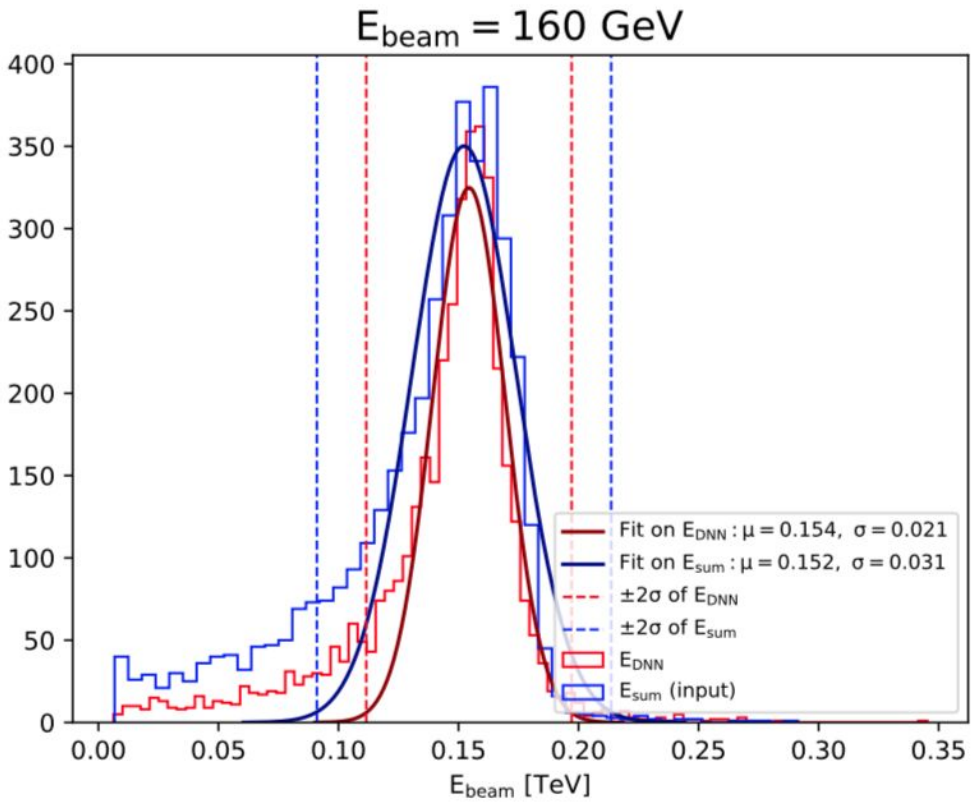
3D Conv


$E_{beam} = 100$ GeV
350
300
250
200
150
100
50
0
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
E_{beam} [TeV]
Fit on E_{DNN} : $\mu = 0.097$, $\sigma = 0.016$
Fit on E_{sum} : $\mu = 0.097$, $\sigma = 0.022$
$\pm 2\sigma$ of E_{DNN}
$\pm 2\sigma$ of E_{sum}
E_{DNN}
E_{sum} (input)

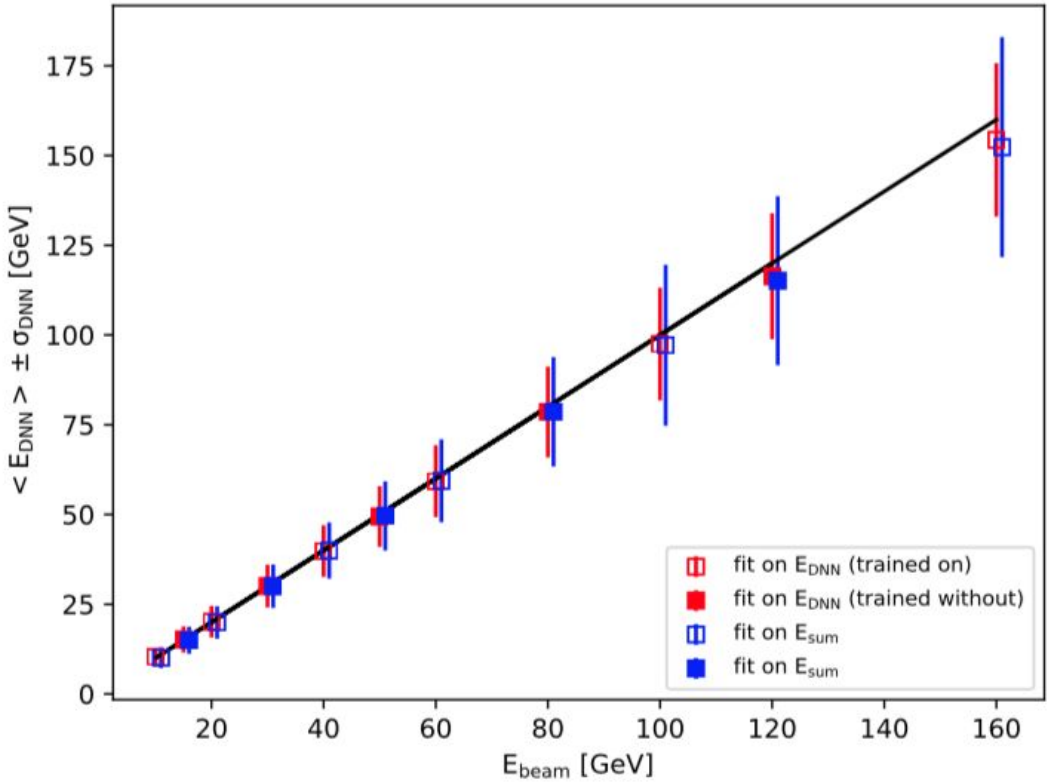
3D Conv


E_{beam} = 120 GeV (interpolated)
350
300
250
200
150
100
50
0
0.00
0.05
0.10
0.15
0.20
0.25
E_{beam} [TeV]
Fit on E_{DNN} : μ = 0.116, σ = 0.018
Fit on E_{sum} : μ = 0.115, σ = 0.024
±2σ of E_{DNN}
±2σ of E_{sum}
E_{DNN}
E_{sum} (input)

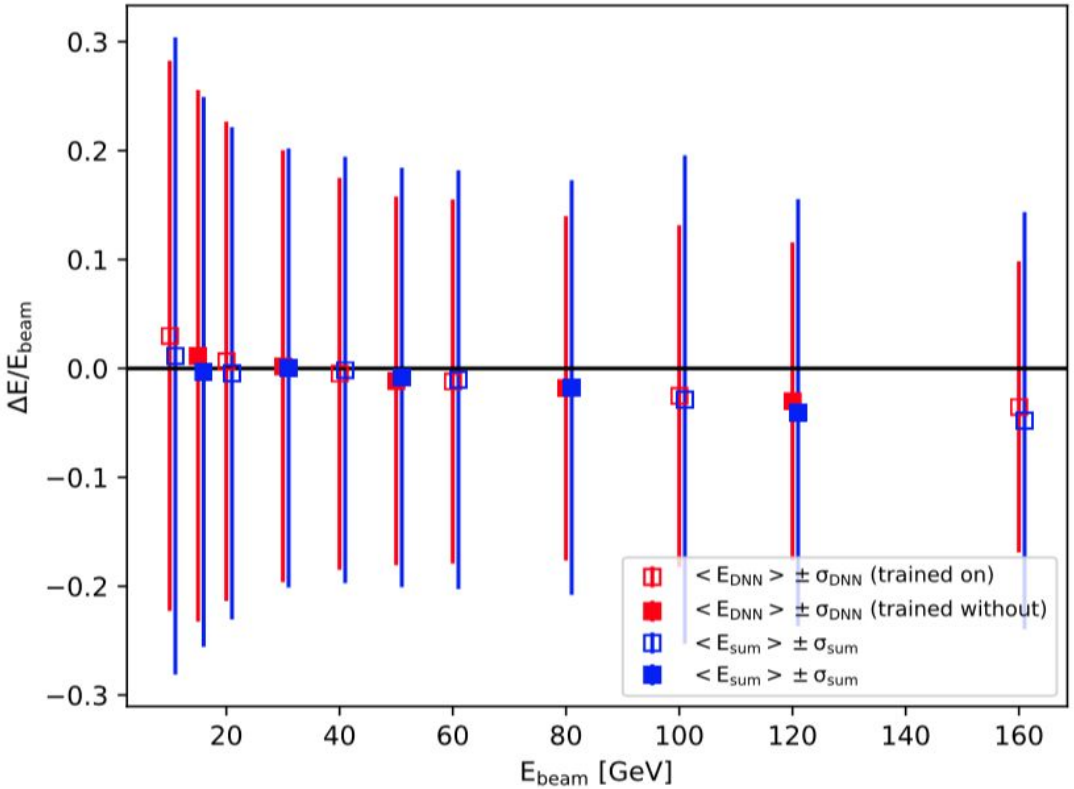
3D Conv


$E_{beam} = 160$ GeV
400
350
300
250
200
150
100
50
0
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
E_{beam} [TeV]
Fit on E_{DNN} : $\mu = 0.154$, $\sigma = 0.021$
Fit on E_{sum} : $\mu = 0.152$, $\sigma = 0.031$
$\pm 2\sigma$ of E_{DNN}
$\pm 2\sigma$ of E_{sum}
E_{DNN}
E_{sum} (input)

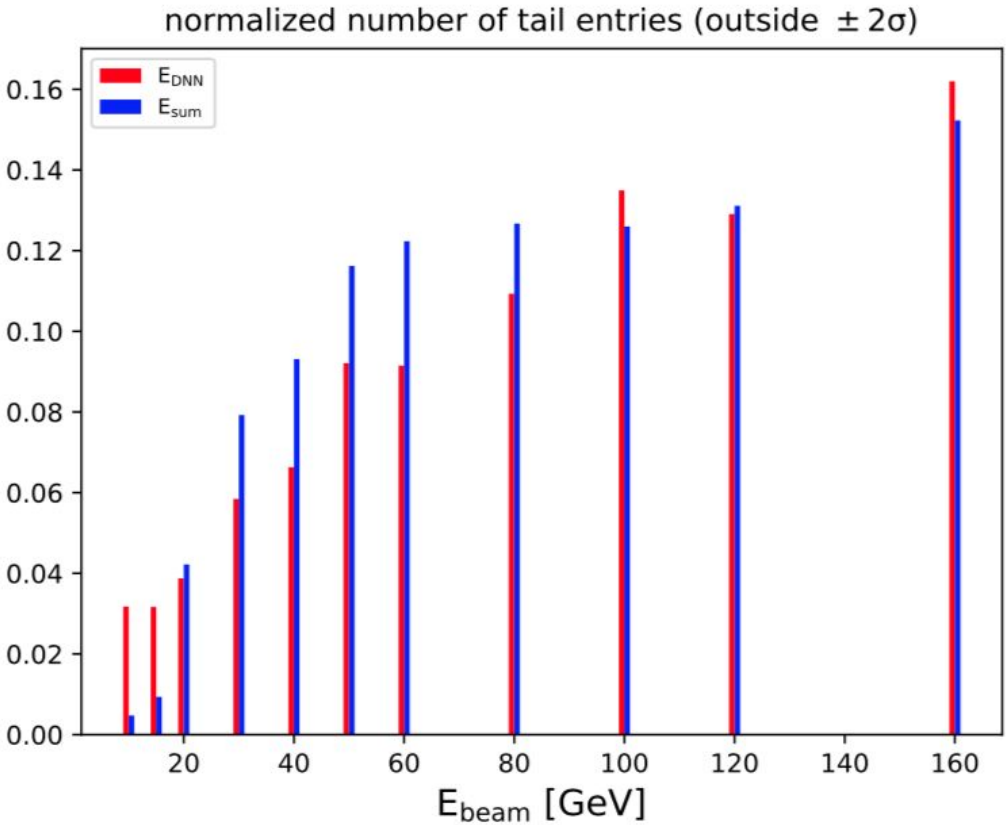
3D Conv


175
150
125
100
75
50
25
0
20
40
60
80
100
120
140
160
< E_DNN > ± σ_DNN [GeV]
E_beam [GeV]
fit on E_DNN (trained on)
fit on E_DNN (trained without)
fit on E_sum
fit on E_sum

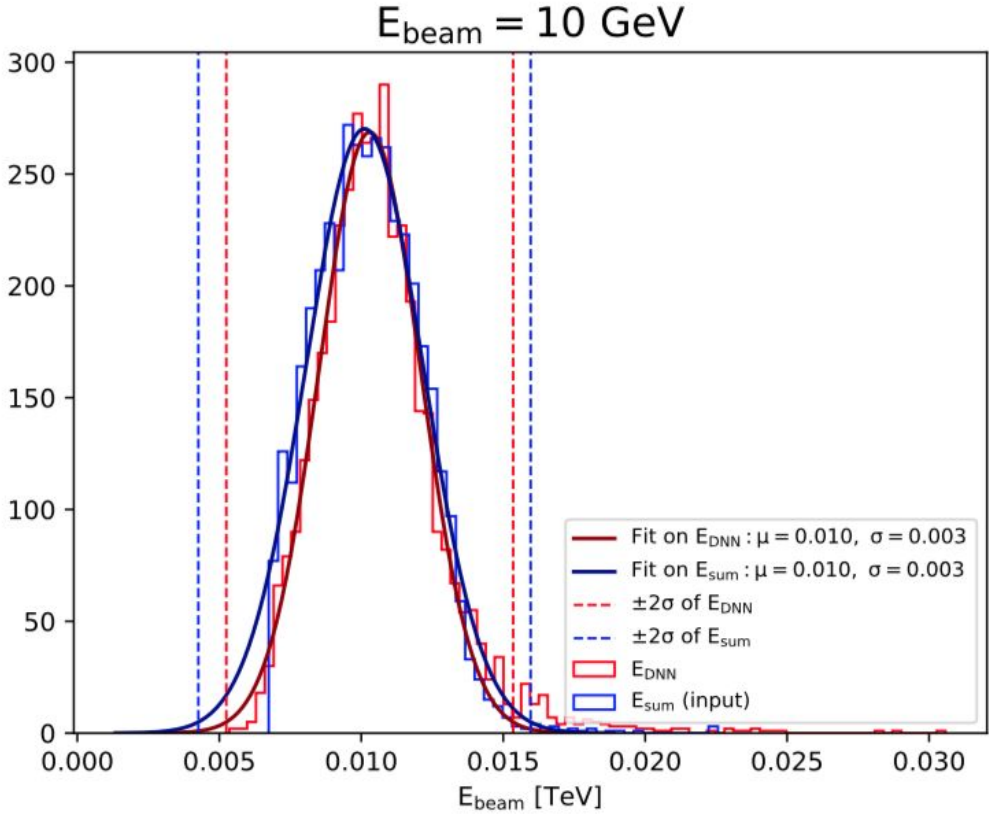
3D Conv


0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
ΔE/E_beam
20
40
60
80
100
120
140
160
E_beam [GeV]
< E_DNN > ± σ_DNN (trained on)
< E_DNN > ± σ_DNN (trained without)
< E_sum > ± σ_sum
< E_sum > ± σ_sum

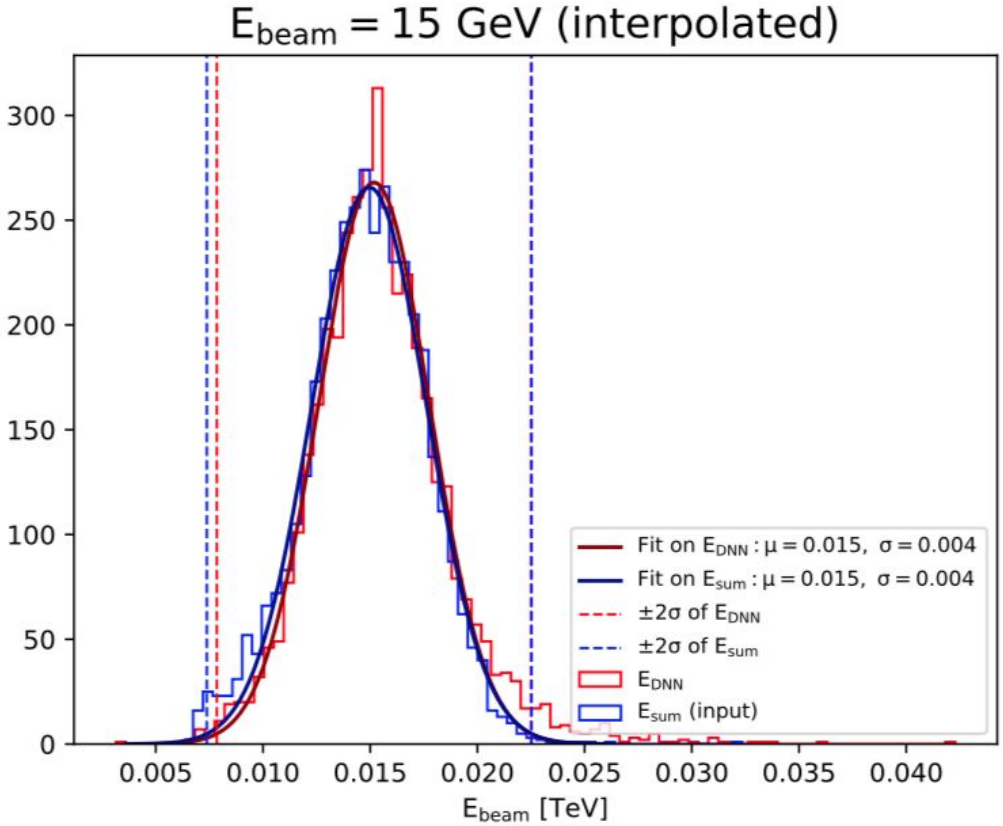
3D Conv


normalized number of tail entries (outside ± 2σ)
E_DNN
E_sum
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
20
40
60
80
100
120
140
160
E_beam [GeV]

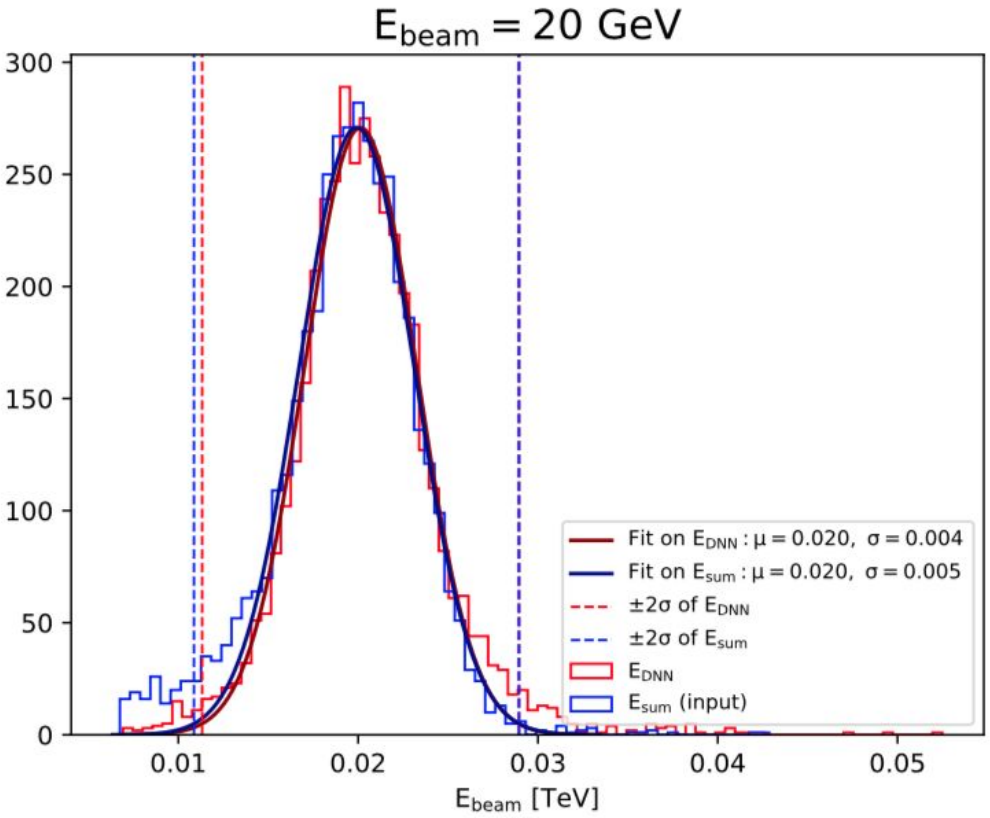
Locally Connected



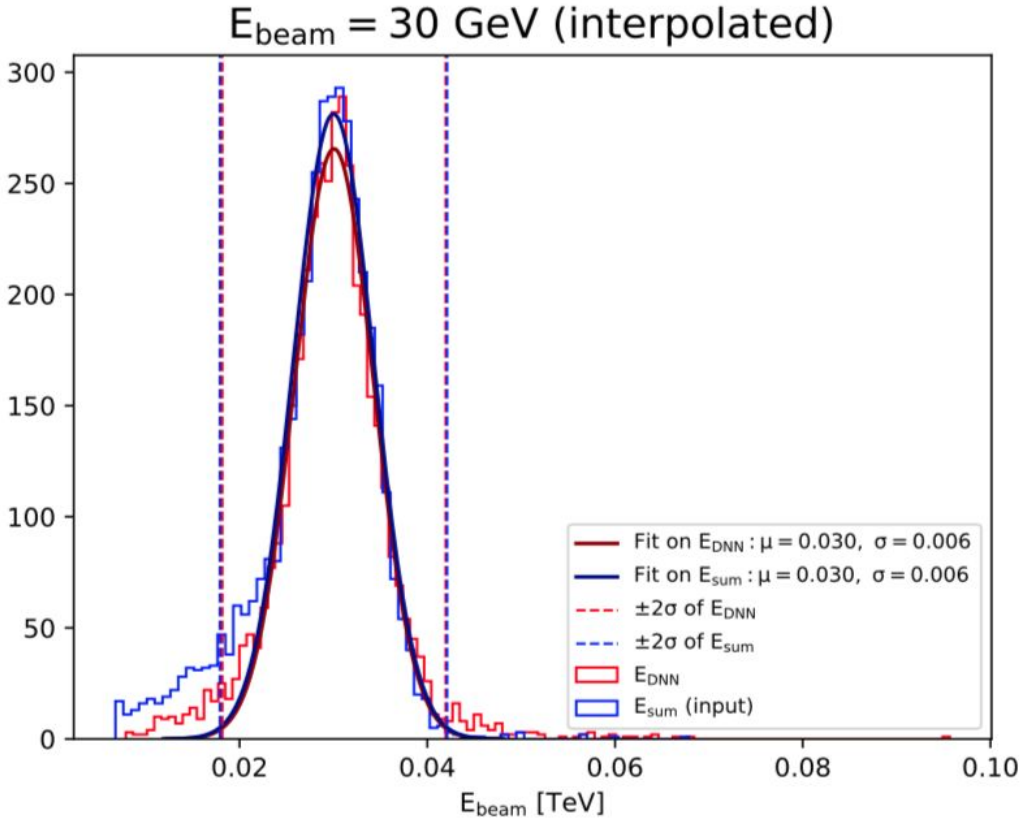
Locally Connected



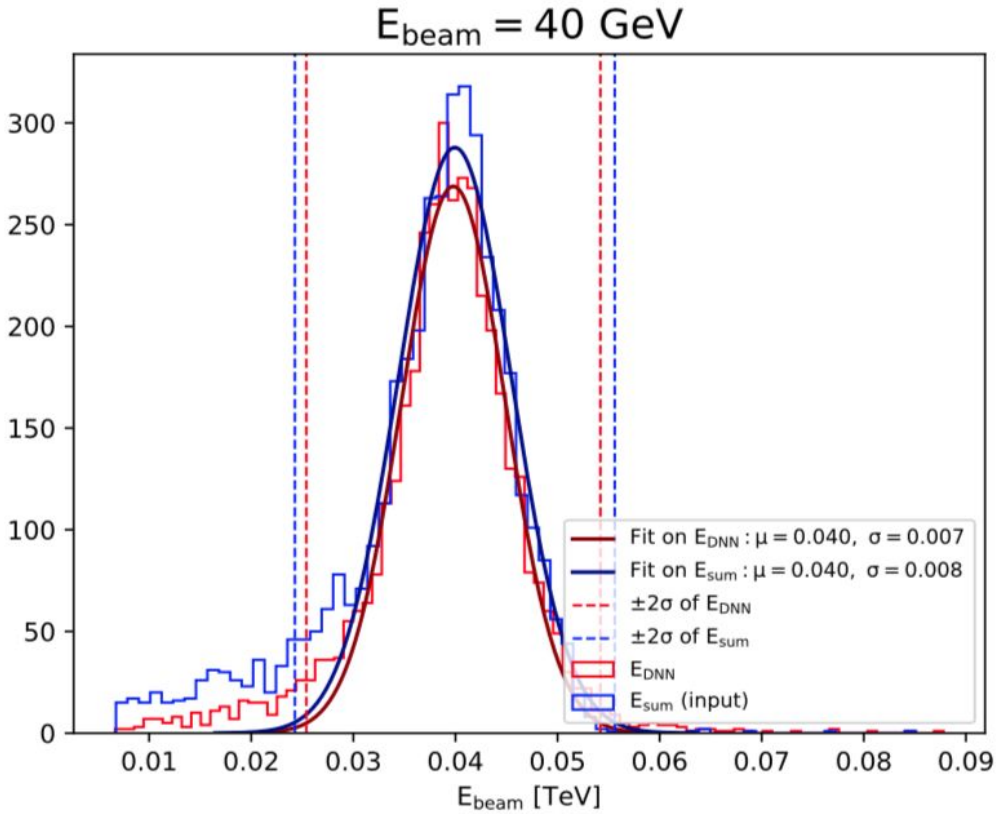
Locally Connected



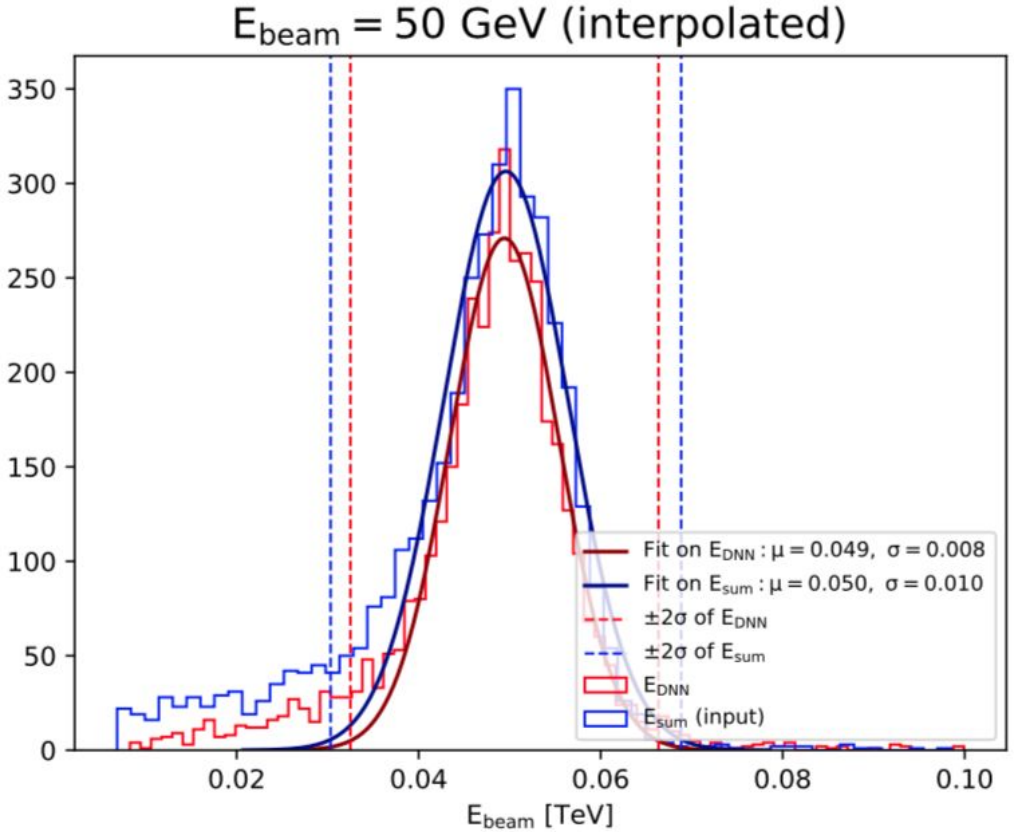
Locally Connected



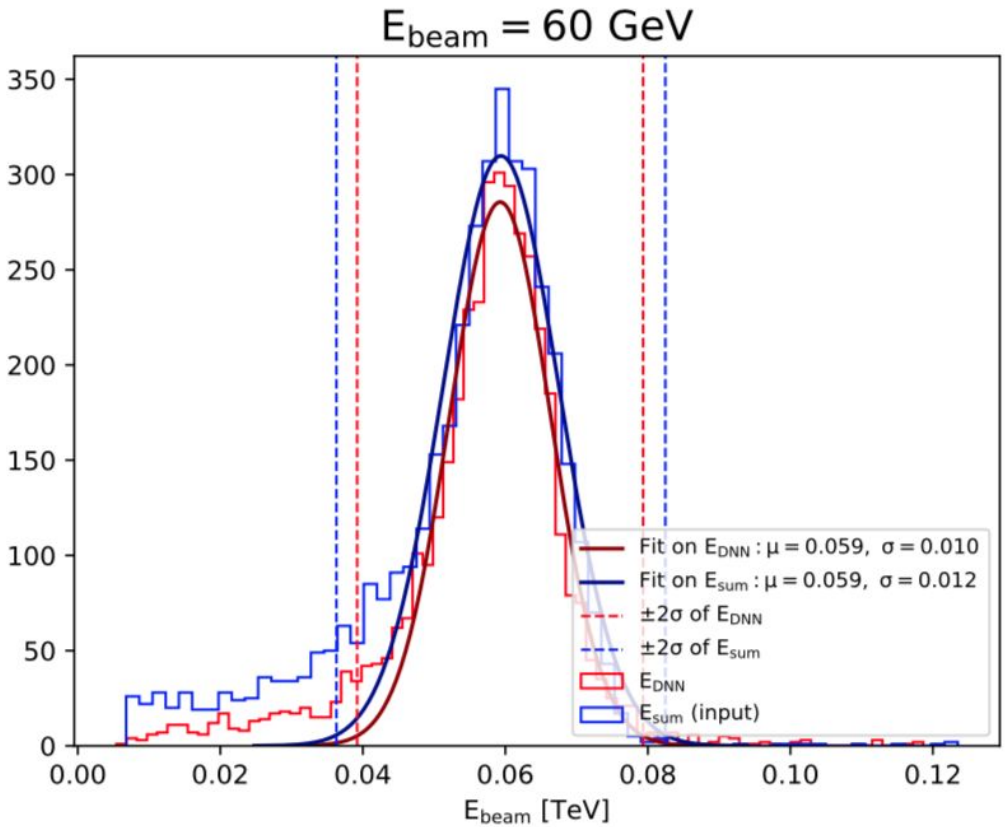
Locally Connected



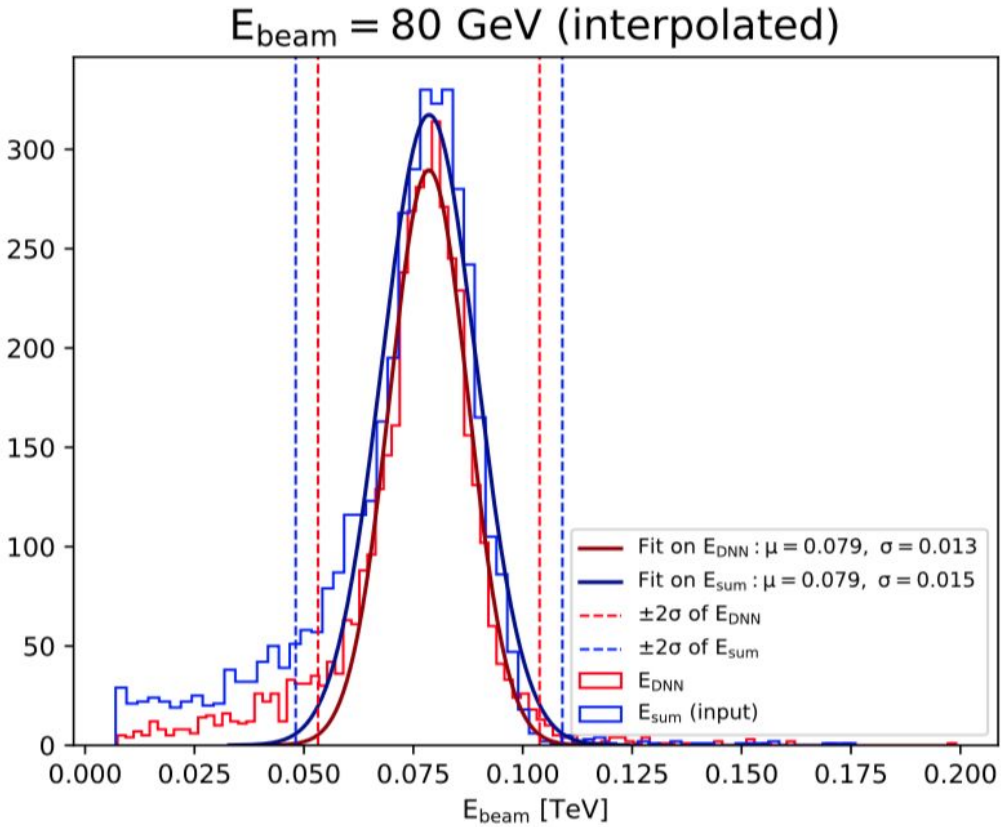
Locally Connected



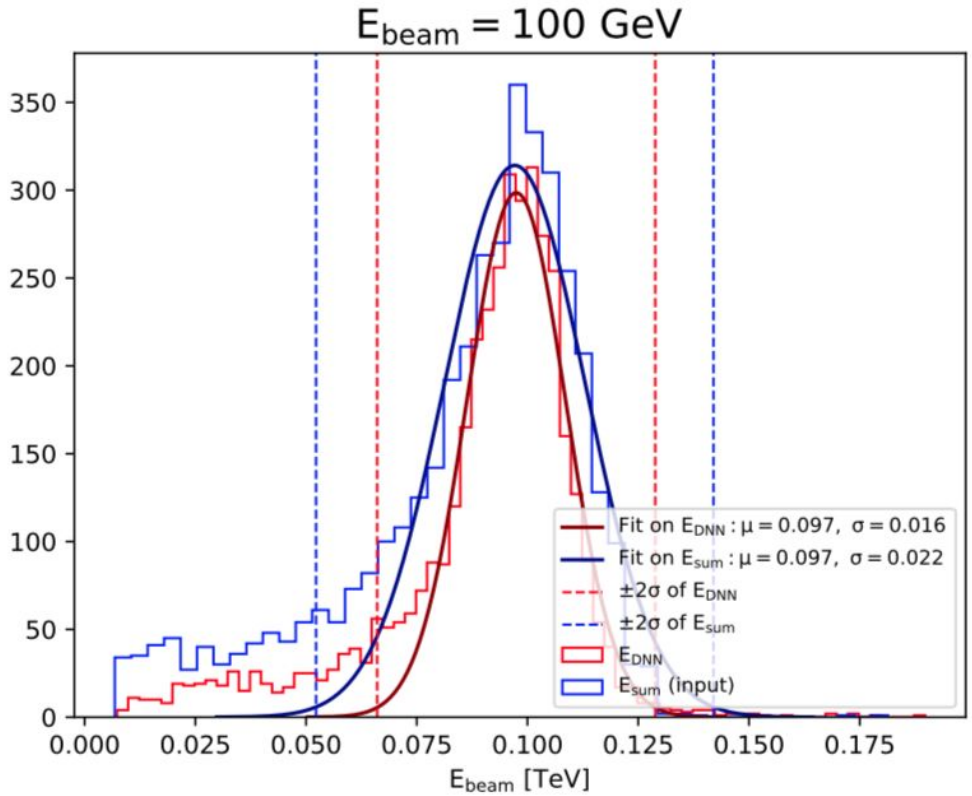
Locally Connected



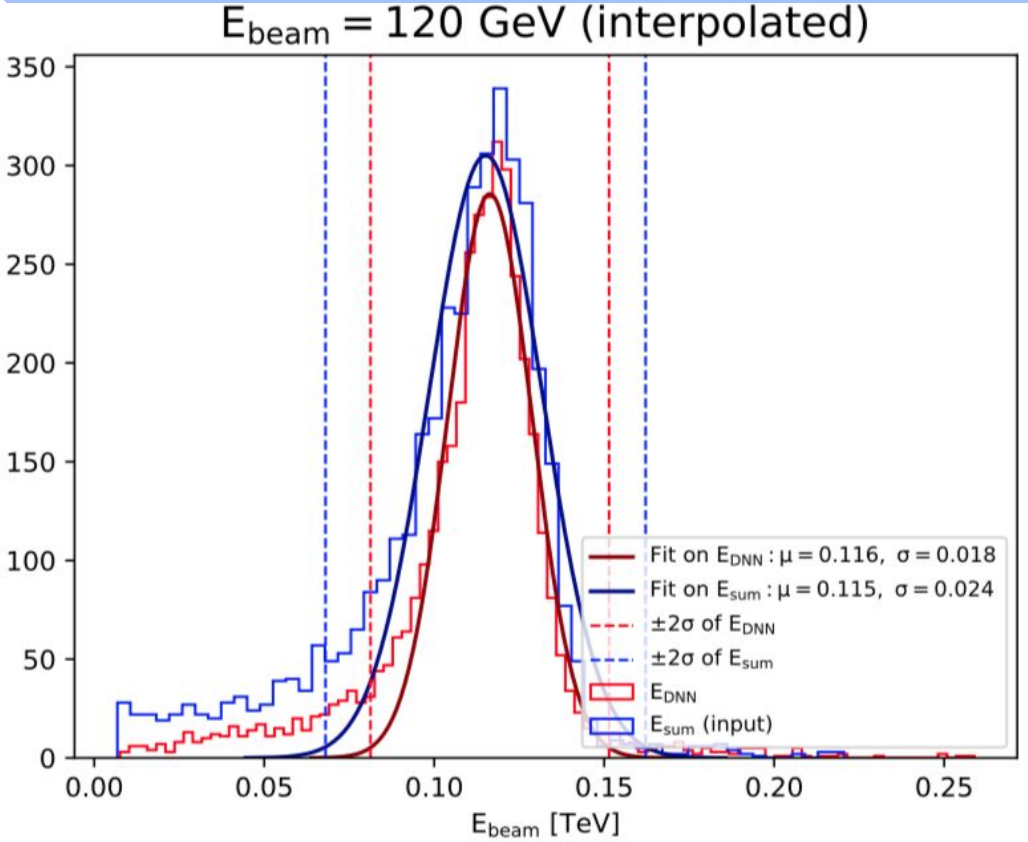
Locally Connected



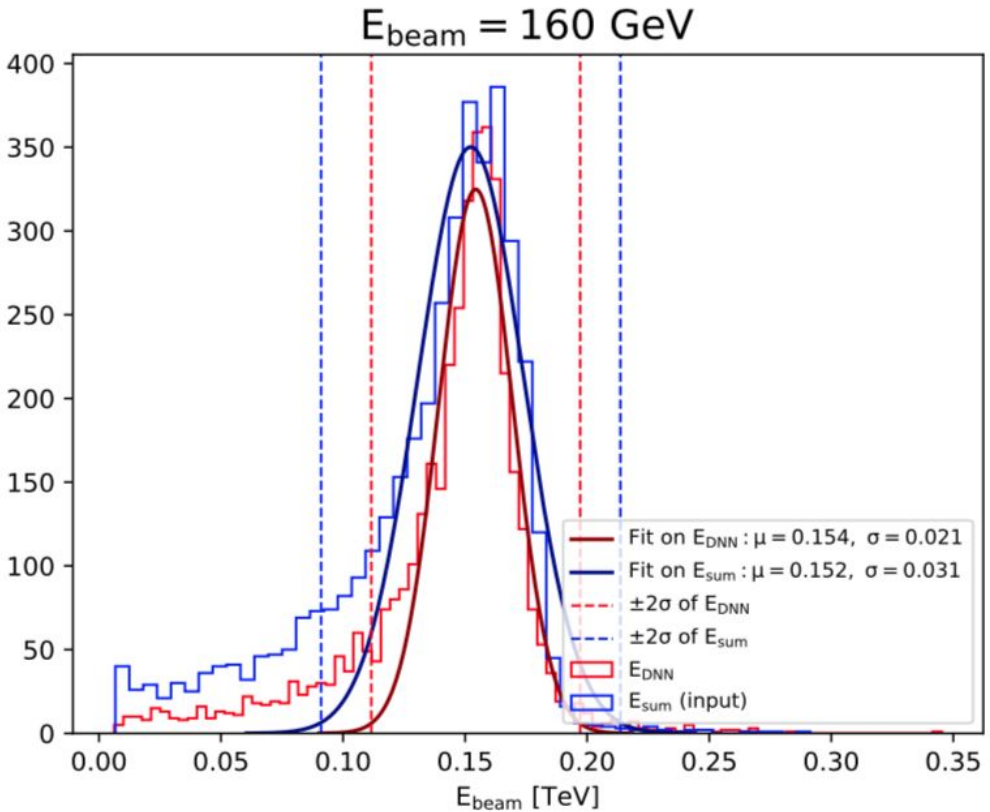
Locally Connected



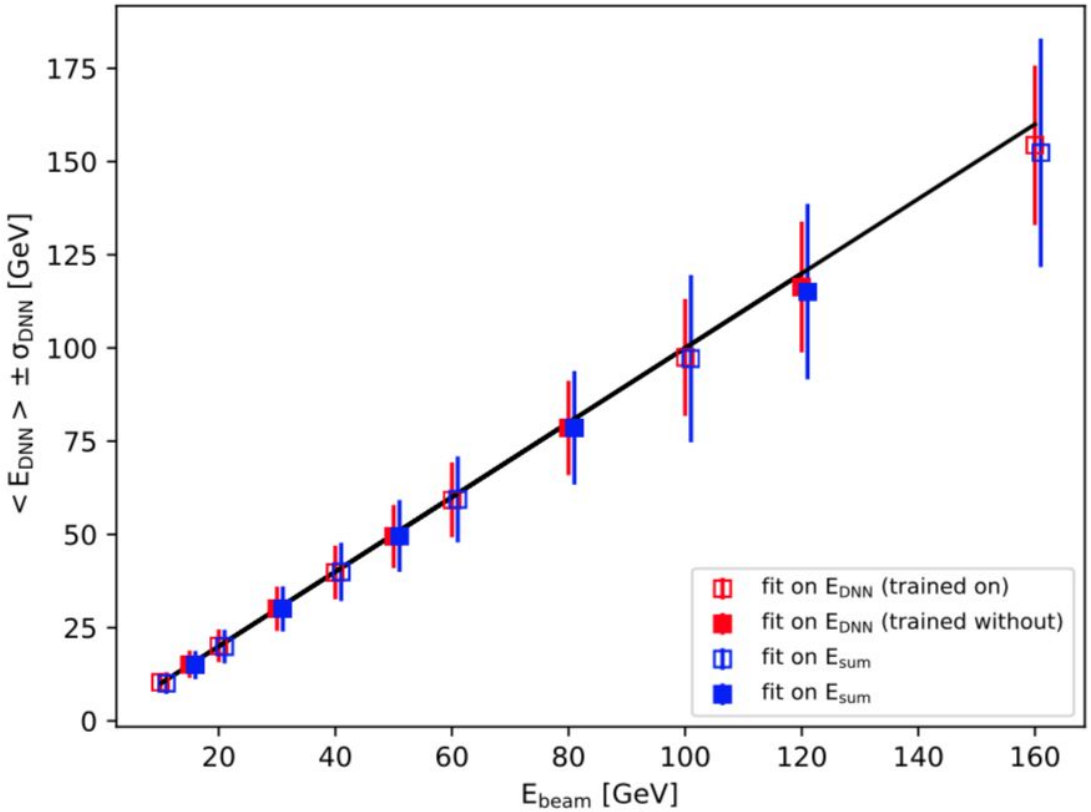
Locally Connected



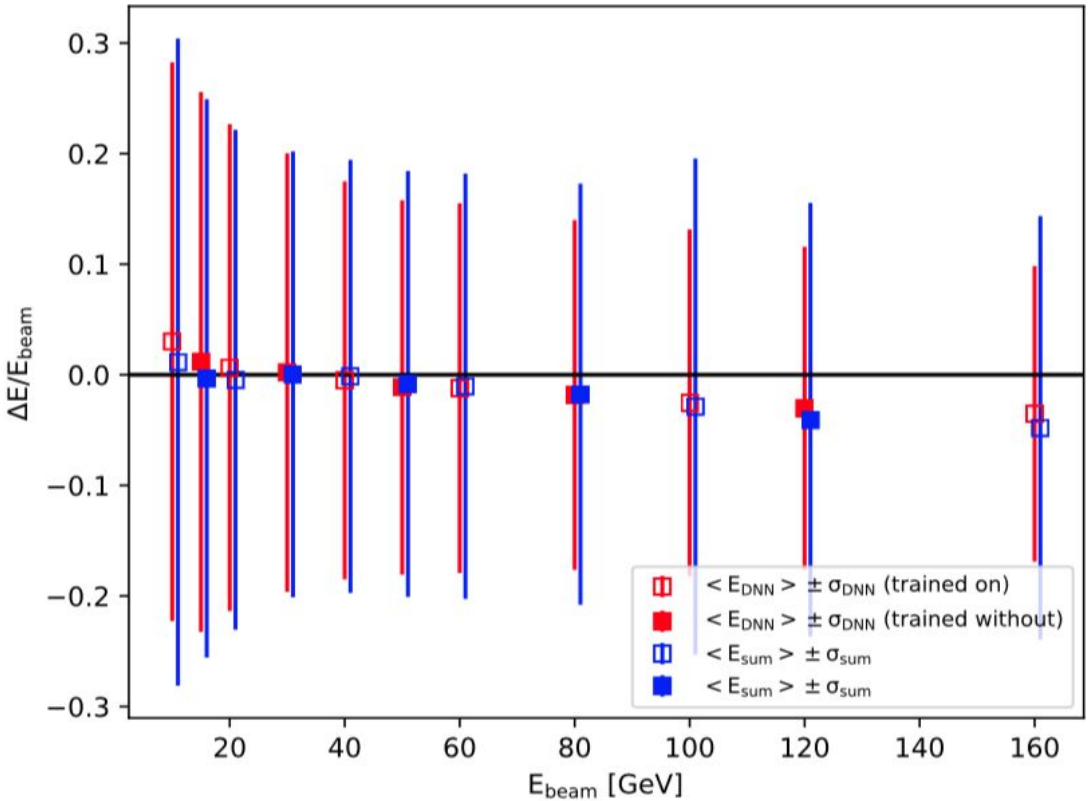
Locally Connected



Locally Connected



Locally Connected



Locally Connected