Construct
Deep Jet Clustering

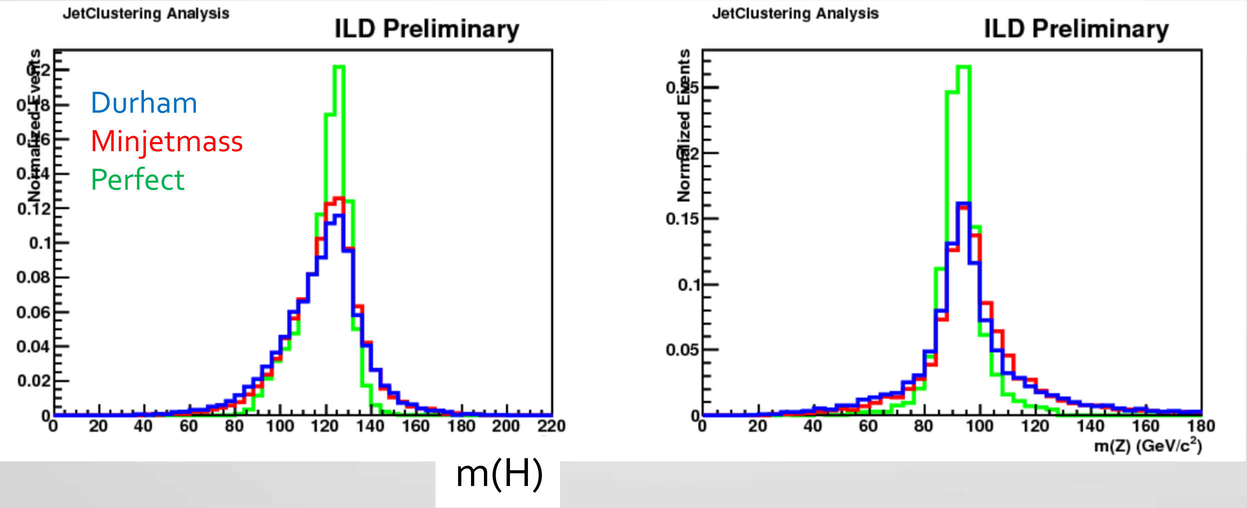
LCWS2021,

Masakazu Kurata

03/17/2021

Introduction

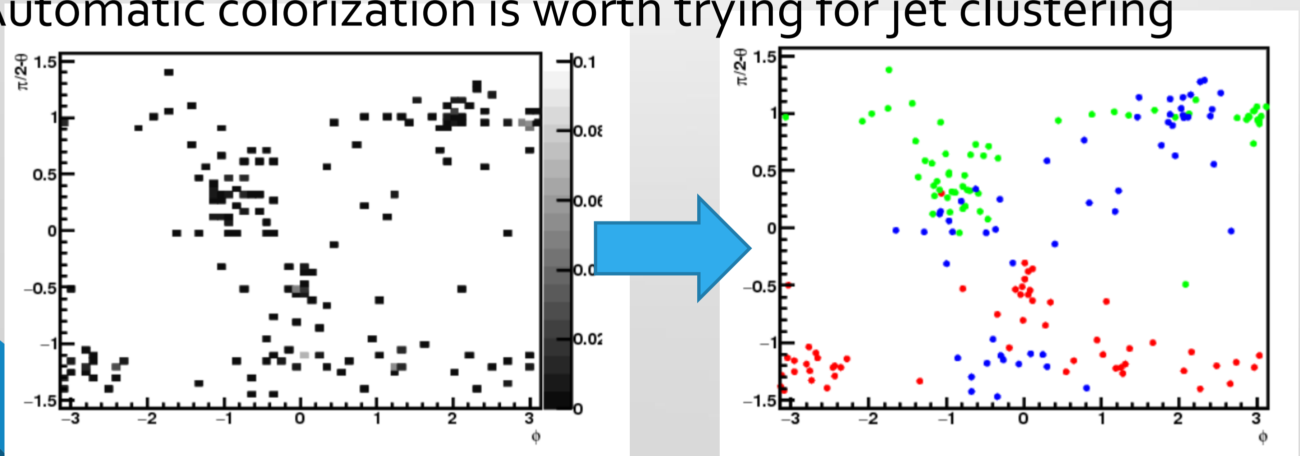
- Jet clustering is one of the main key to obtain better physics results
  - Physics results are strongly limited by mis-clustering
  - To obtain correct jets leads to improve the mass resolution of the resonances
- Present jet clustering is far from good tool for reconstructing jets
  - e.g. Higgs self-coupling@500GeV(ZHH): ~40% improvement if perfect!



- Even at 250GeV, clustering is very important
  - Separation of ZH/ZZ/WW in hadronic events

Use CNN for automatic colorization

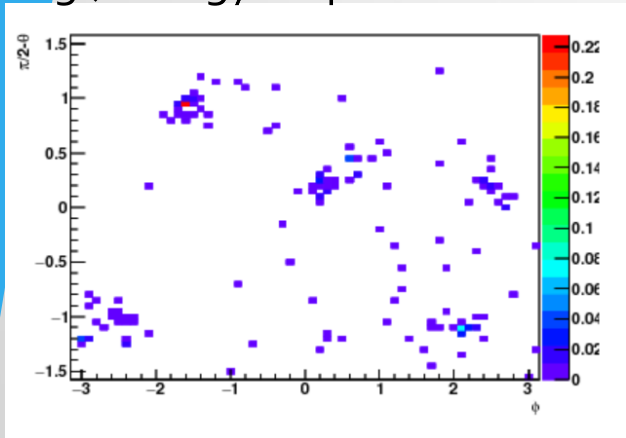
- For jet clustering, we need the global and local information for each event
  - Global: Where is the large energy located?
  - Local: Correlation between neighbors or large energy area?
- Using Convolutional Neural Network(CNN), we will extract both features
  - Encorder-Decorder type CNN is used (calls as u-network, mention later)
- Clustering is equivalent to “colorize” each particle in the same cluster
  - Grey scale ⇒ color
  - So, Automatic colorization is worth trying for jet clustering



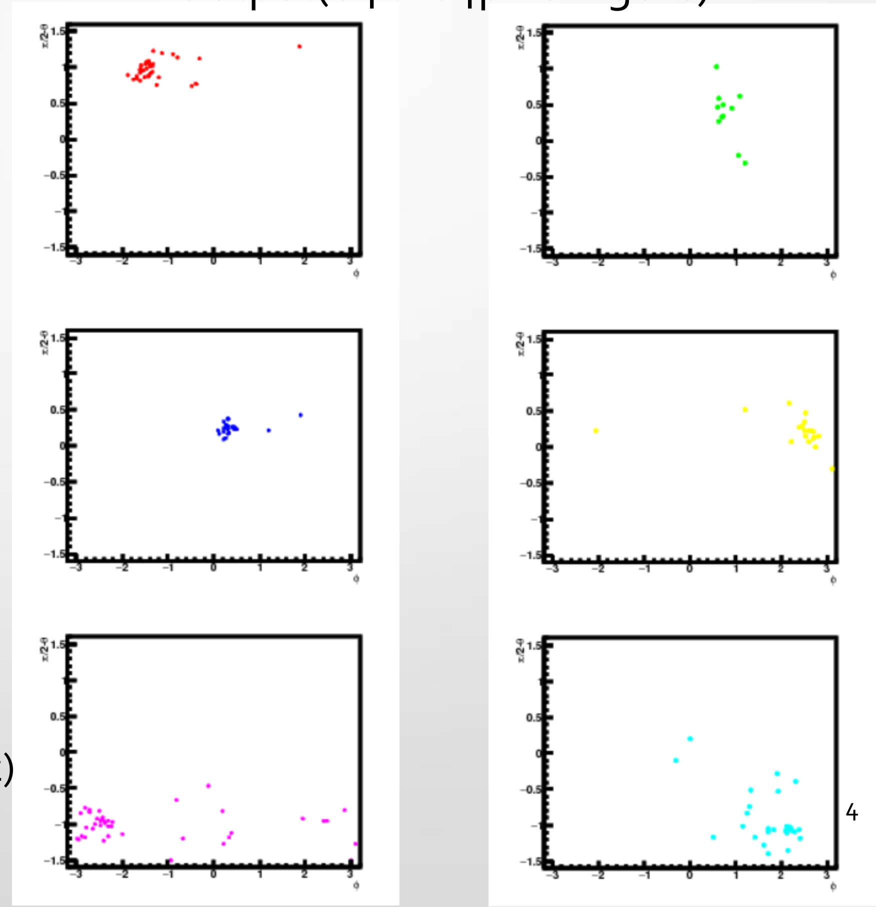
Trial

- Use Keras & tensorflow backend
- Using a certain map(s) of each event, estimate color of each track
  - Do not consider color-singlet state

Input(64×64pixel figure)
e.g.) energy map



Output(64×64pixel figure)

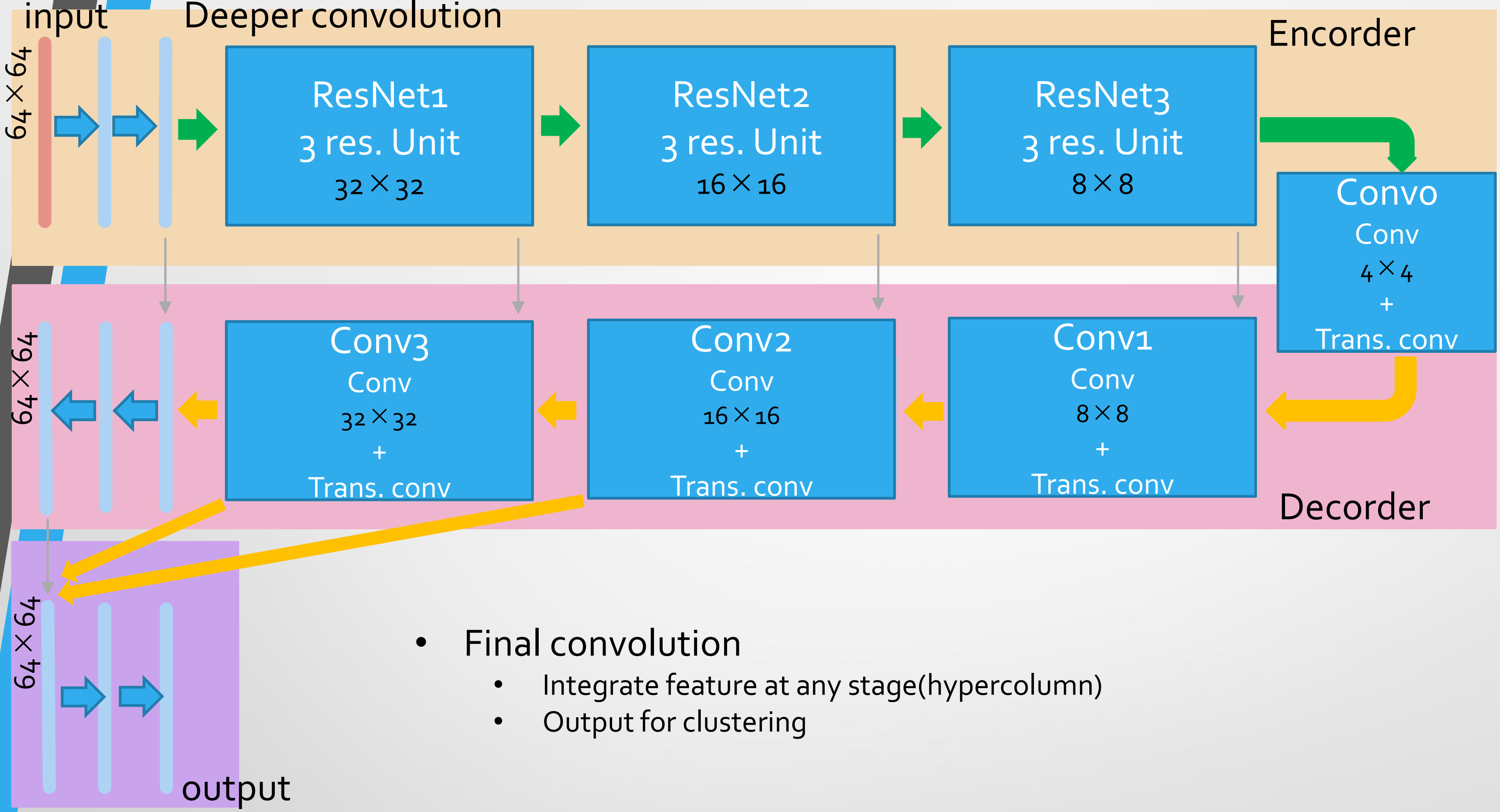


9 input images
Energy map
Charge map
Dosig map
Zosig map
Ecal map
Hcal map

+ direction vector(x, y, z)

+ no particle

Network Architechture



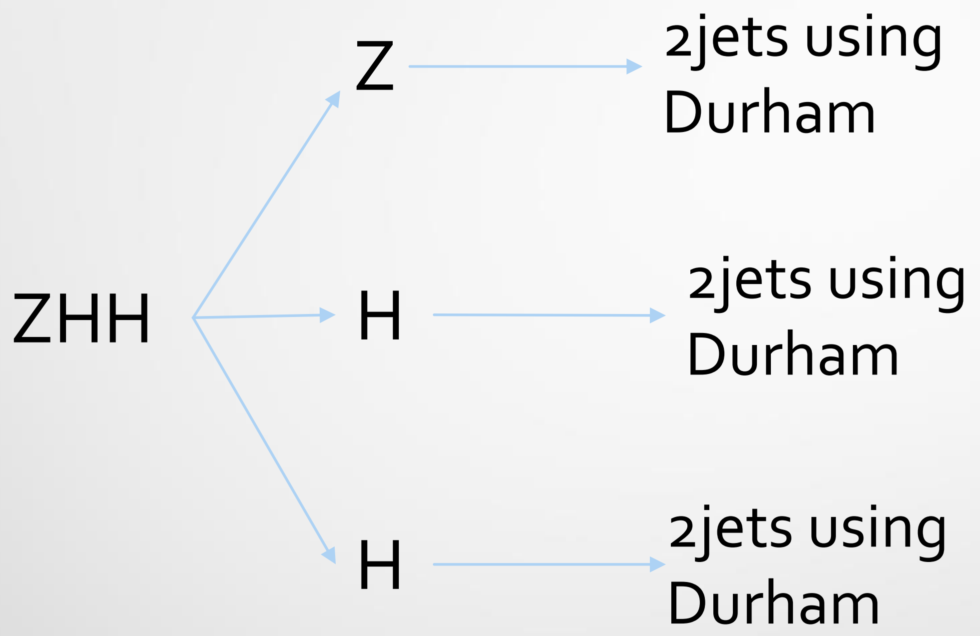
- Final convolution
  - Integrate feature at any stage(hypercolumn)
  - Output for clustering

- Encorder
  - Extract global & higher order feature
  - Downsample to make network robust for distortion & shift effect
  - Lost position information

- Decorder
  - Propagate obtained feature to local
  - Upsample to recover position information
  - Merge encoder nodes to get precise position information

Create answer

- Supervised learning - Create “answer” jets(numbering jets): perfect Durham jet clustering



  - Number is assigned based on energy ordering
    - Highest energy jet number is 0, lowest energy jet number is 5
- So far, do not consider color singlet state: number of jets is 6
  - ZHH→(qq)(bb)(bb)→6jets

Basics: How to train a network

- A network is trained to minimize loss function
- If output of the network is probability, cross-entropy is a popular loss function

$$L = -\frac{1}{n}\sum_n \sum_i t_i \log y_i$$

  - Answer is zero(data is not belong to this node) or one(data is belong to this node)
  - Only one node is 1, the other nodes are 0 (one-hot vector)
  - Smaller value of L means better performance
- Considering better loss function is one of the important points to obtain better performance

Output

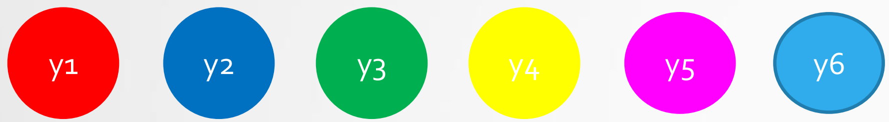
$$\sum_i y_i = 1$$

Answer
0 or 1

Pseudo-labelling

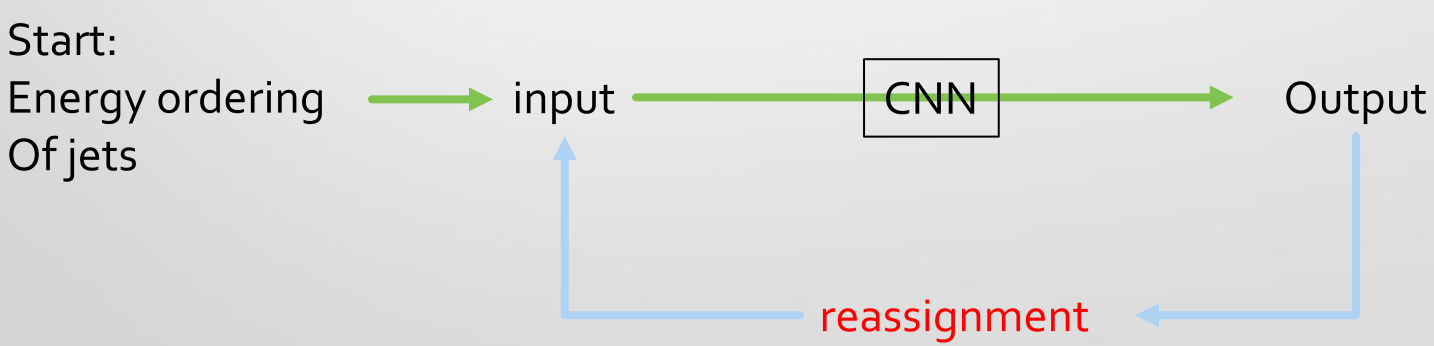
- Output: inference of the probability of the color to be assigned
  - $\sum y_i = 1.0$

output



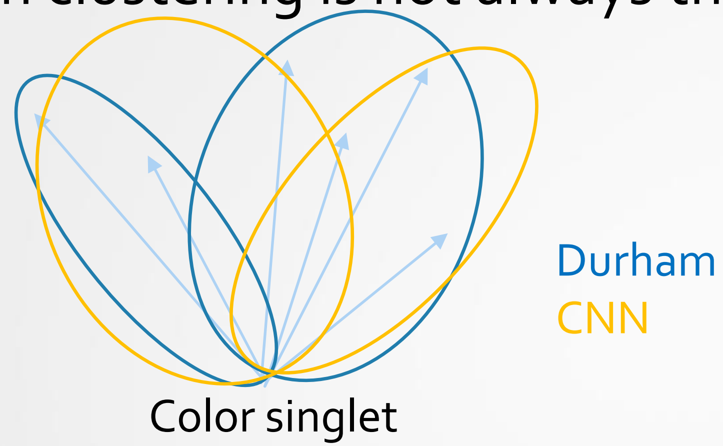
+ no particle

- The combination of color assignments is arbitrary, so assign them so that the loss function is minimized.
  - Using preliminary results after a training, re-assign the color combination
  - Minimize cross entropy $L = -\frac{1}{n}\sum t_i \log y_j$ event by event

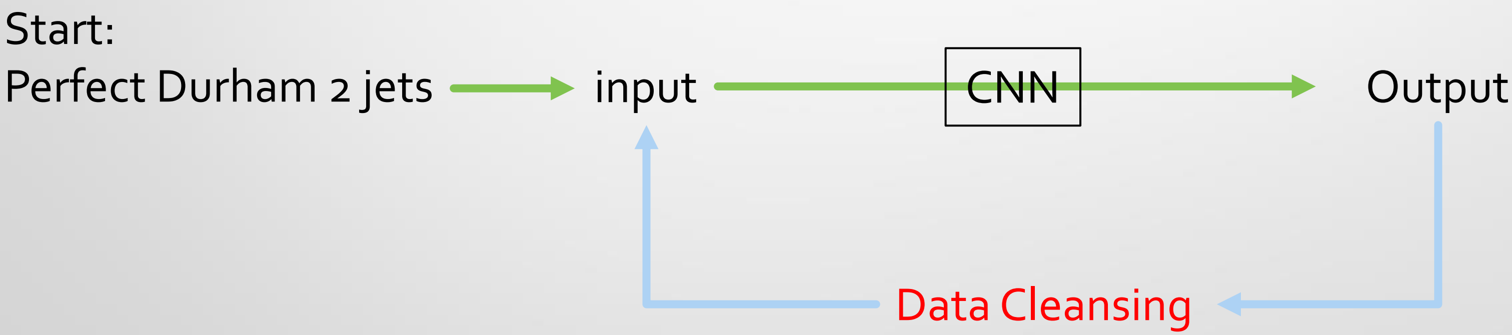


Data Cleansing

- Perfect Durham clustering is not always the best clustering into 2 jets for CNN



- By using the preliminary training weights, clustering into 2 jets is performed



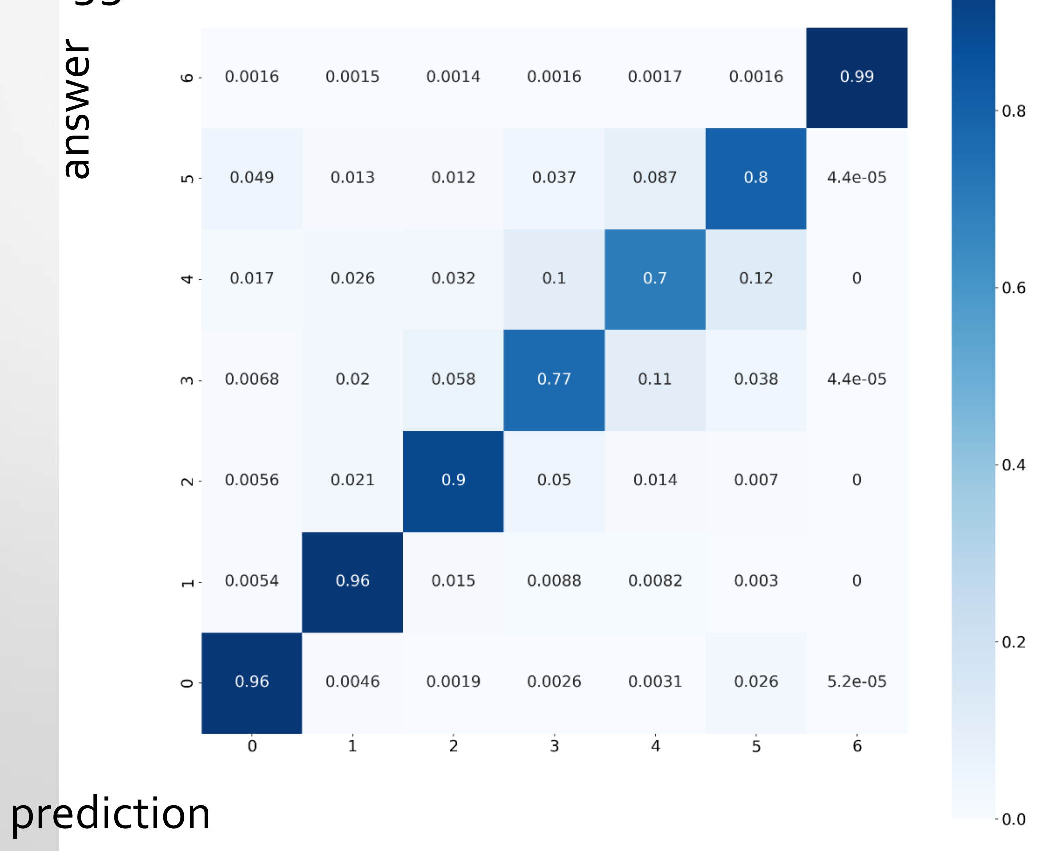
- Clustering particles to make loss function minimum

status

- Use ZHH→(qq)(bb)(bb): 6jets clustering
  - q: uds + c + b   ratio: 9:9:5   need to consider the effect of flavor ratio
    - If a network can really feel flavor
- Use 230000 events for training(207000 train, 23000 validation)
  - Very weak or no over fitting can be seen between train vs. validation
- Don't consider color singlet state for network training so far
- Input: 6 + 3 images   output: 6 + 1 images

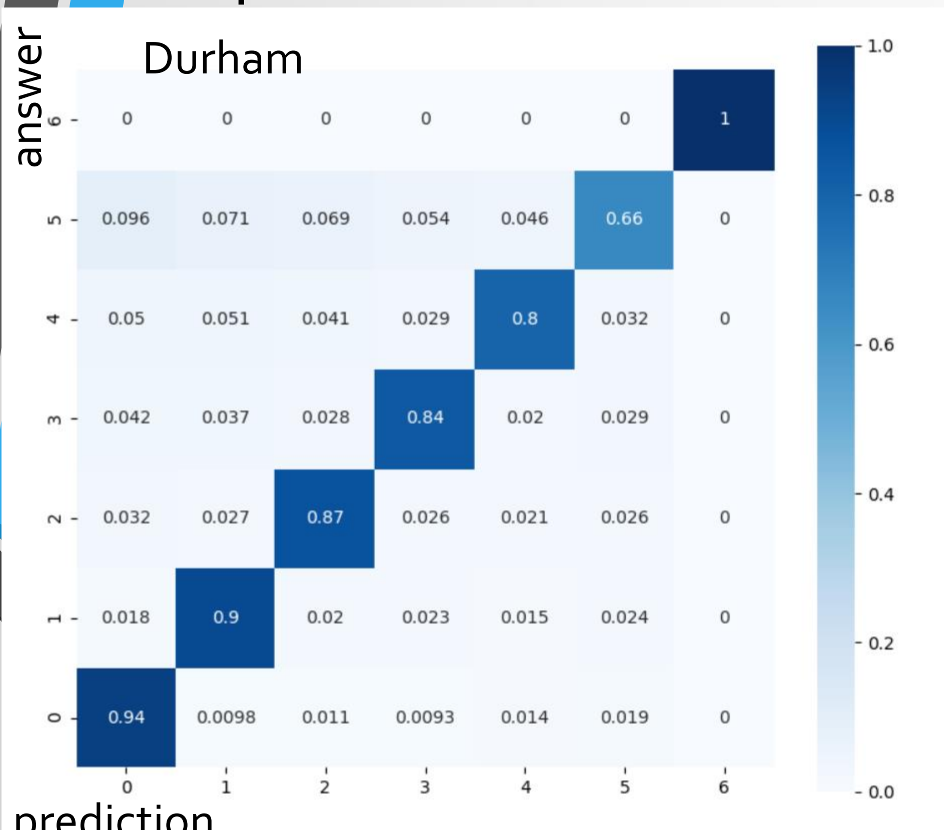
Preliminary results

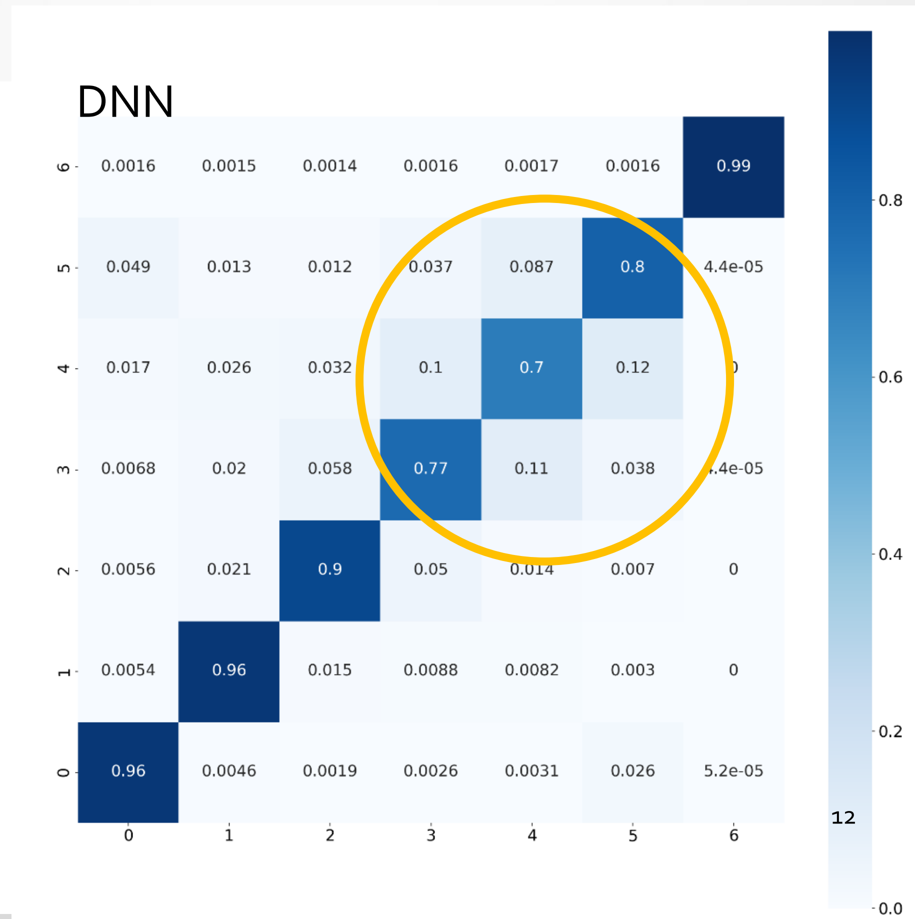
- Confusion matrix
  - 1000 events
  - Num. 0-5: jet number 6: no particle
  - Off-diagonal means mis-assignment
  - Some particles locate at adjacent jets of their answer jet: should be reduce this correlation
  - Now under investigation
- Need to check resolution of Higgs mass
  - Need to improve



Comparison with Durham

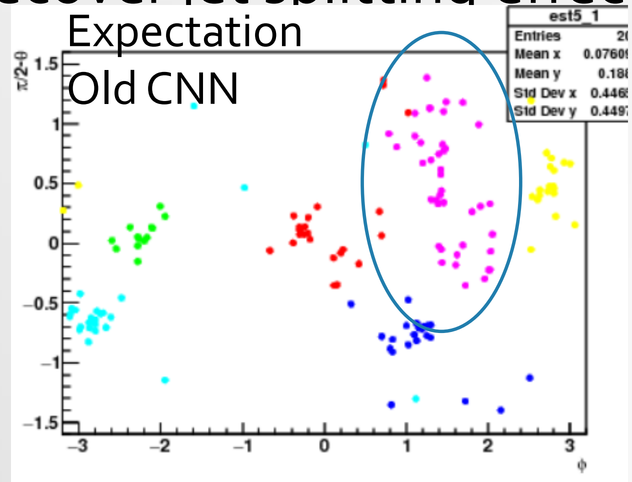
- First 3 jets: start to improve
- Last 3 jets: a bit worse
  - Worst efficiency is improved, but need to improve more
- Correlation between adjacent jets, and asymmetric
- Need to improve circle efficiency to get better performance

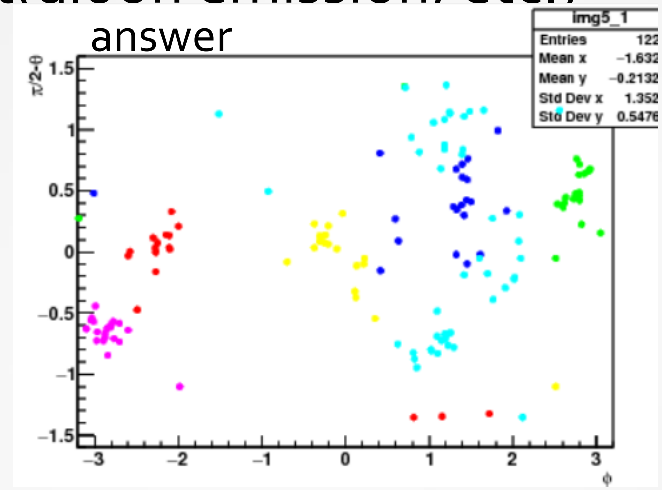




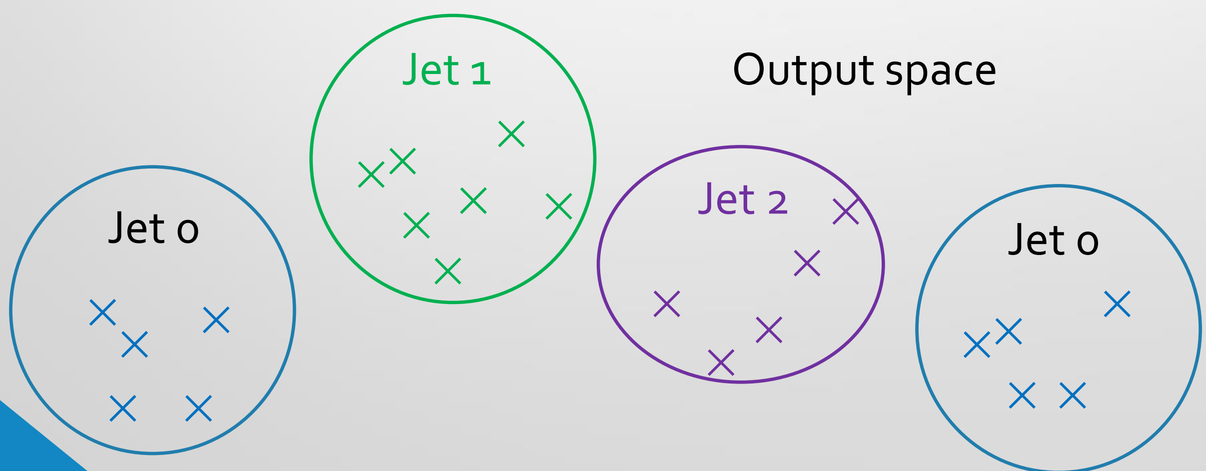
Multi Center

- Try to recover jet splitting effect(gluon emission, etc.)



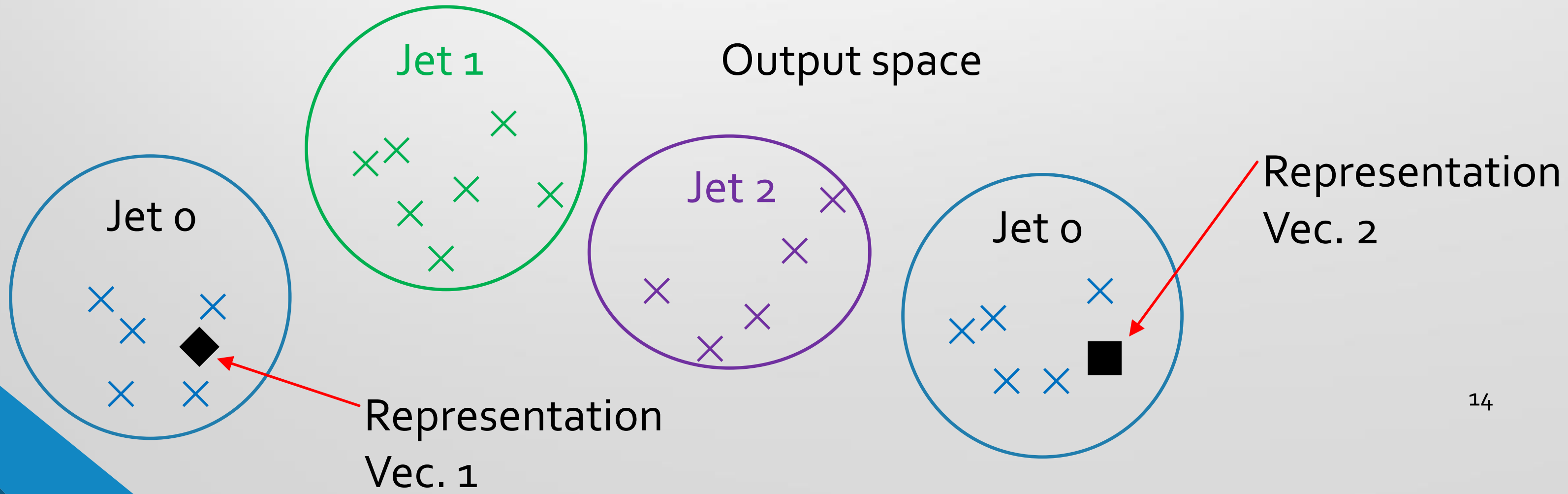


- Split clusters are difficult to merge
  - Each cluster will be located at different space
  - So, distribution will be multimodal



Multi Center

- Weights of before output layer in the network are equivalent to representation vectors of each cluster
- So, increasing the number of final weights is increasing the number of representation vectors
  - Automatically assign each representation vector to each (sub) cluster
  - Automatically determine num. of representation vectors through training
  - Done via applying constraint on weights



Status for this network

- Use ZHH→(qq)(bb)(bb): 6jets clustering
  - q: uds + c + b
- Use 200000 events for training(180000 train, 20000 validation)
- Don't consider color singlet state for network training so far
- Input: 6 + 3 images output: 6 + 1 images

Multi Center

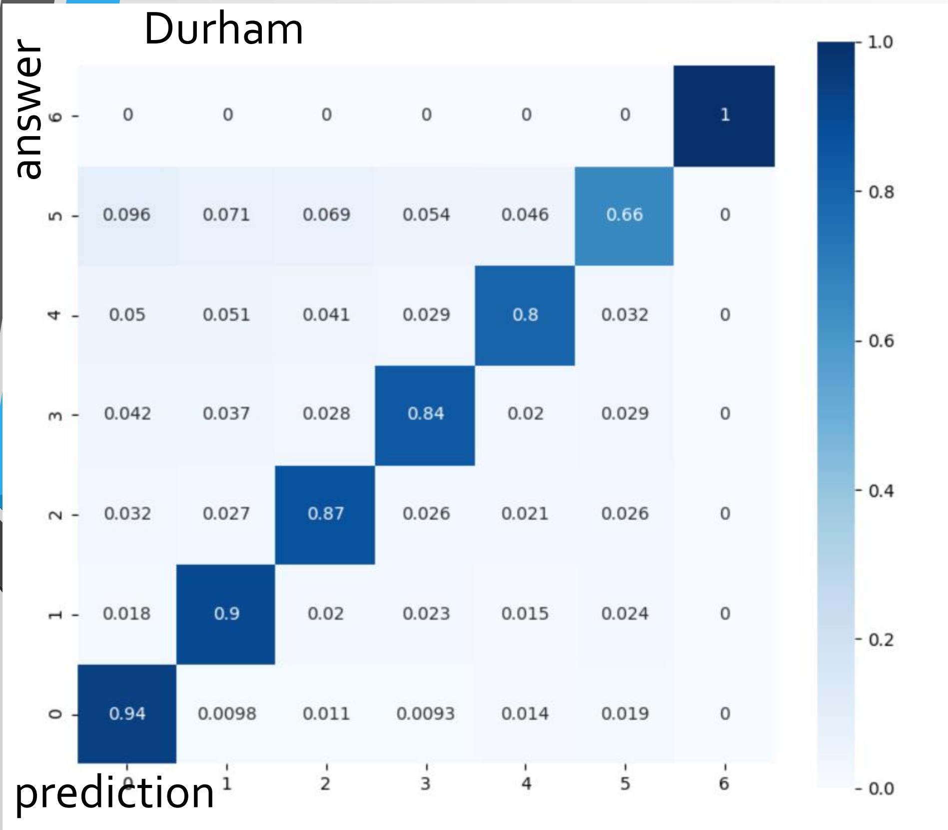
- Training on going: Very preliminary
- Check the effect on multicenter

	jet0	jet1	jet2	jet3	jet4	jet5
Num. of weights prepared	5	5	5	5	5	5
Num. of weights survived	2	2	2	2	2	1

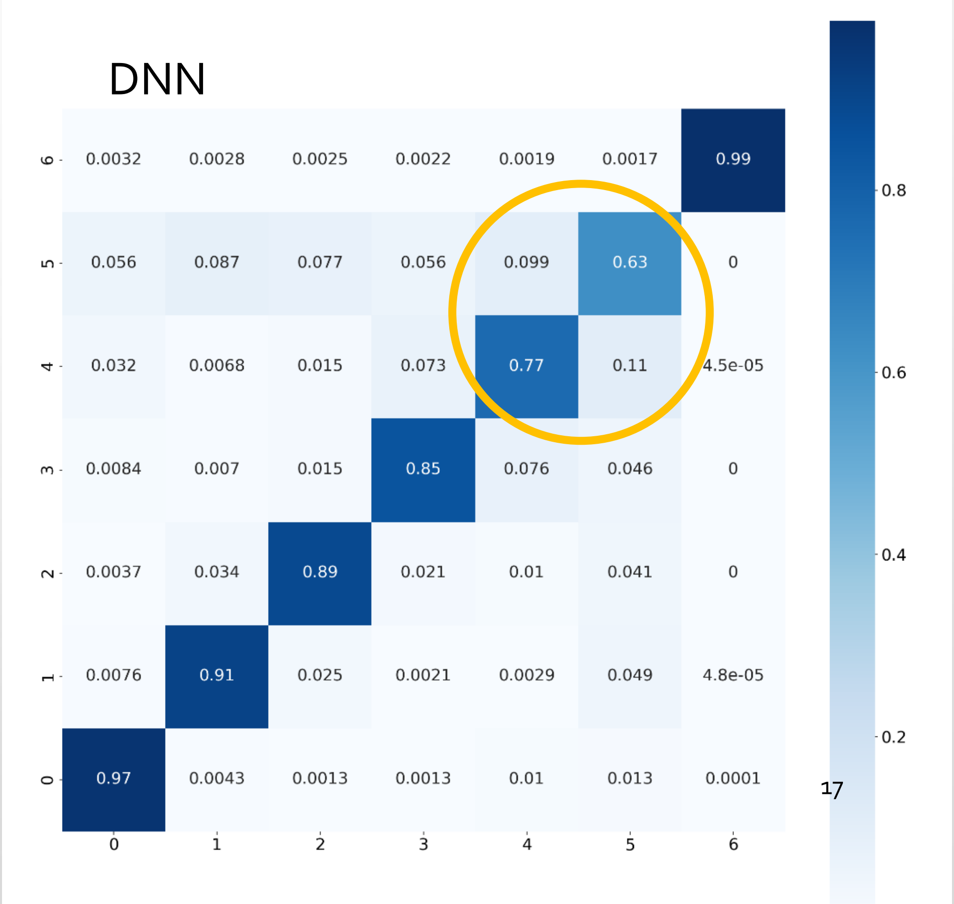
  - Looks network automatically catches feature in different(?) output space
- Just start training
  - Need more precise study

(Very) Preliminary Result

- 4 jets: start to improve
- Last 2 jets: a bit worse
  - Still not enough to catch features for low energy/ soft clusters
- Different behavior from single center case
  - Should be checked
- Need to study more



6	0	0	0	0	0	0	1
5	0.096	0.071	0.069	0.054	0.046	0.66	0
4	0.05	0.051	0.041	0.029	0.8	0.032	0
3	0.042	0.037	0.028	0.84	0.02	0.029	0
2	0.032	0.027	0.87	0.026	0.021	0.026	0
1	0.018	0.9	0.02	0.023	0.015	0.024	0
0	0.94	0.0098	0.011	0.0093	0.014	0.019	0
	0	1	2	3	4	5	6



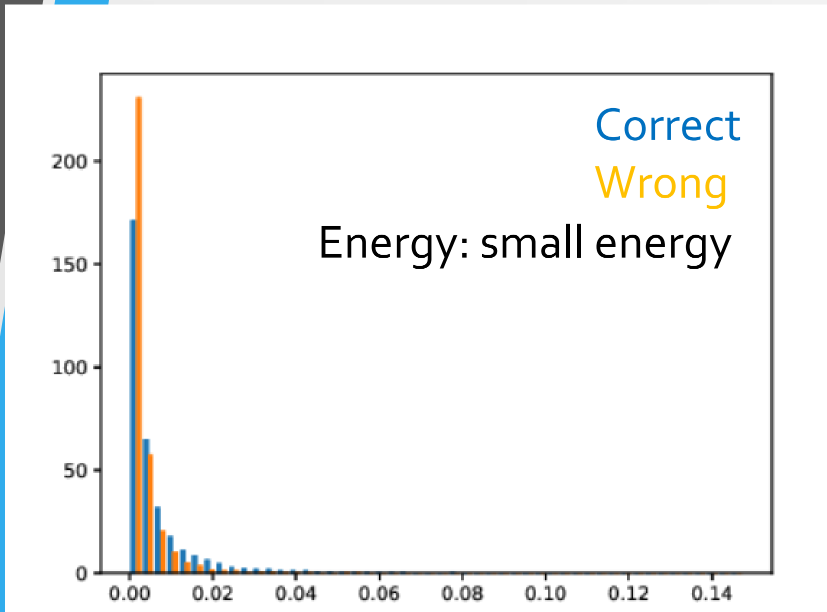
6	0.0032	0.0028	0.0025	0.0022	0.0019	0.0017	0.99
5	0.056	0.087	0.077	0.056	0.099	0.63	0
4	0.032	0.0068	0.015	0.073	0.77	0.11	4.5e-05
3	0.0084	0.007	0.015	0.85	0.076	0.046	0
2	0.0037	0.034	0.89	0.021	0.01	0.041	0
1	0.0076	0.91	0.025	0.0021	0.0029	0.049	4.8e-05
0	0.97	0.0043	0.0013	0.0013	0.01	0.013	0.0001
	0	1	2	3	4	5	6

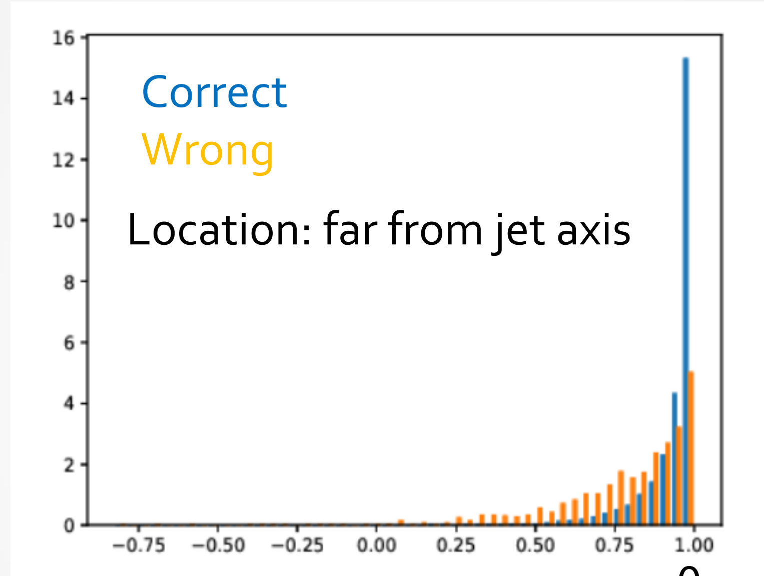
Summary

- Now exploring the possibility of better jet clustering algorithm
  - Can we obtain using Deep Learning?
- So far, the performance of Deep JetClustring seems almost comparable to Durham
  - We are almost there
  - Need to evaluate reconstructed mass resolution next step
- Multi-Center:
  - Network seems to catch multimodal feature in one cluster, but it is still not enough to catch clustering feature especially low/soft energy clusters
  - Need tuning
- Todo:
  - Evaluation with mass resolution
  - Check the correlation between adjacent clusters
  - Check more about splitting cluster(grouping and train classification such event types(?))

backups

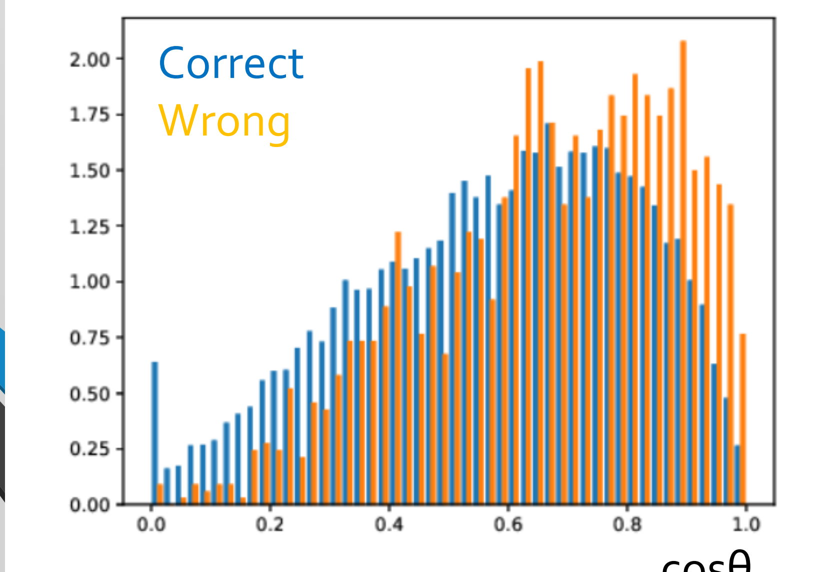
Check the mis-assignment particles





cosθ$_{jet}$

Near adjacent jet axis



cosθ$_{adjacent}$

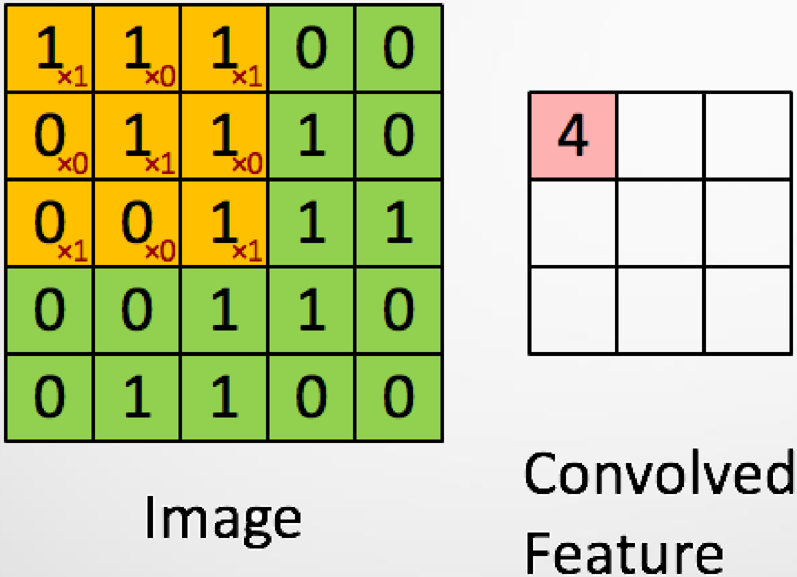
- Mis-assigned particles:
  - Small energy
  - Located at jet boundary
  - 2 jets are very close
- Now trying to solve to assign such particles correctly

Basics: convolution

- Convolution: Apply the filters to extract the feature
  - Sum of the product of each pixel and filter weights:

    $$y_{kl} = \sum_{i,j} w_{ij} \cdot x_{(k+i)(l+j)} \; (+b)$$

  - Slide filters over all the pixels



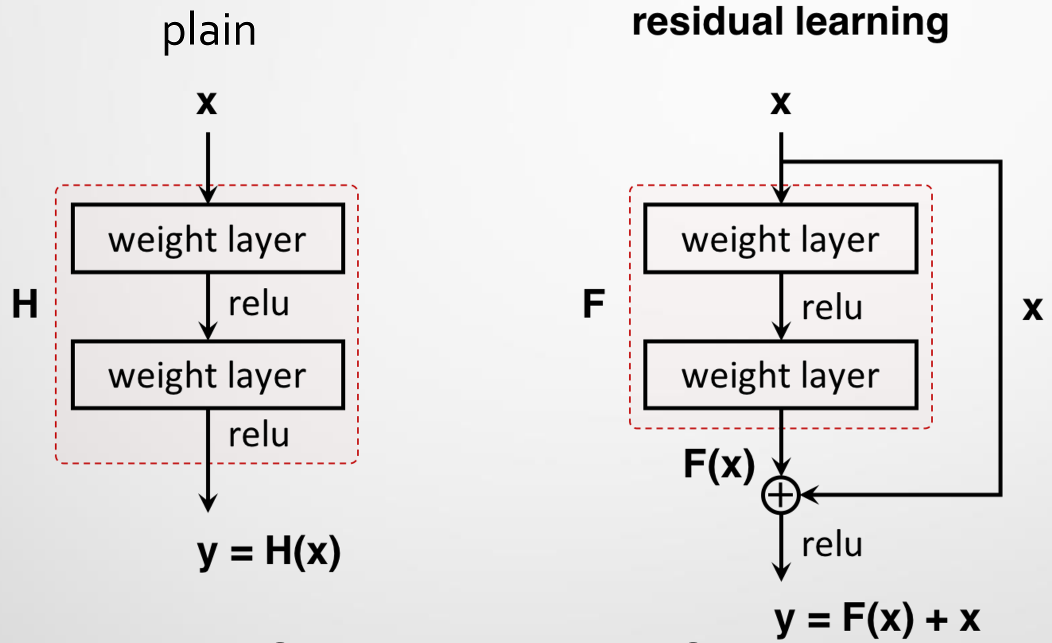
  - Filters are parameters: CNN can obtain them automatically
- After the convolutional operation, apply non-linear transform

  $$z_{kl} = \sigma(y_{kl})$$

  - “Non-linear” is important to get good expression
- Stack these operations

Basics: Residual convolution

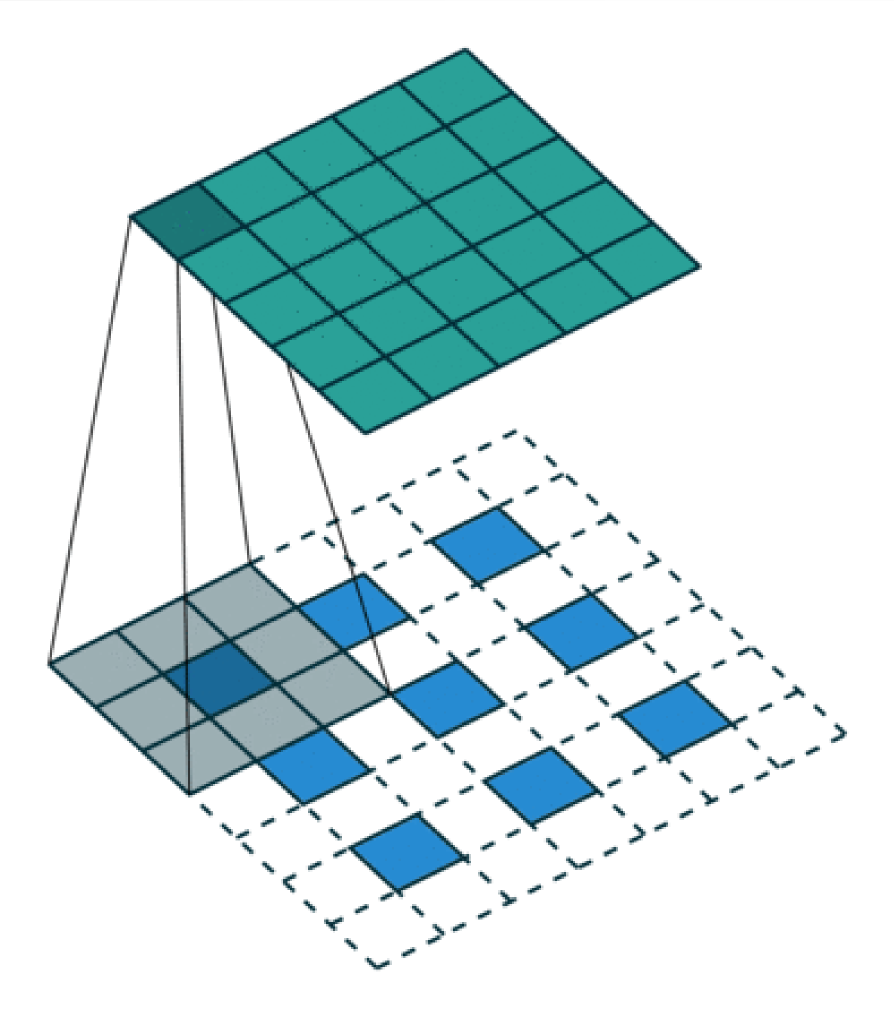
- Stream is divided into 2 paths:
  - Path with convolution
  - Path without any operation
- Sum up these 2 path in downstream



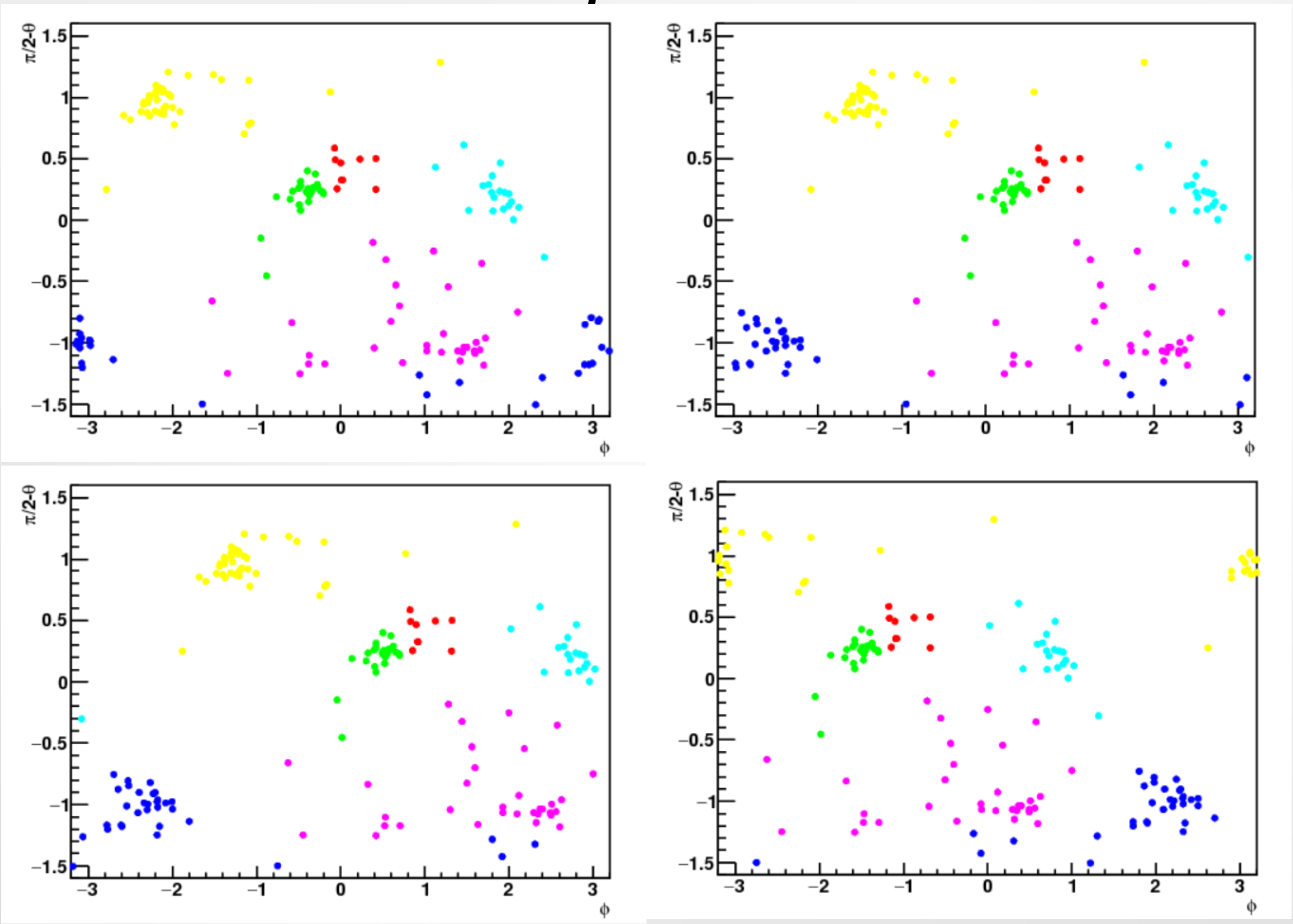
- Can learn "Residuals" of previous layer features
- Can construct very deep network
  - >100 layers can be constructed
  - Deeper will be better performance

Basics: Transposed convolution

- Reverse operation of convolution
  - After adding padding, do convolution
  - Use for upsampling



Data Augmentation



- Random shift for x axis
  - Considering periodic condition of ϕ angle (f(Φ+2π) = f(Φ))
  - To suppress over fitting
- Add random y-flip (I think not good from physics point of view, but suppress over-fitting is important)