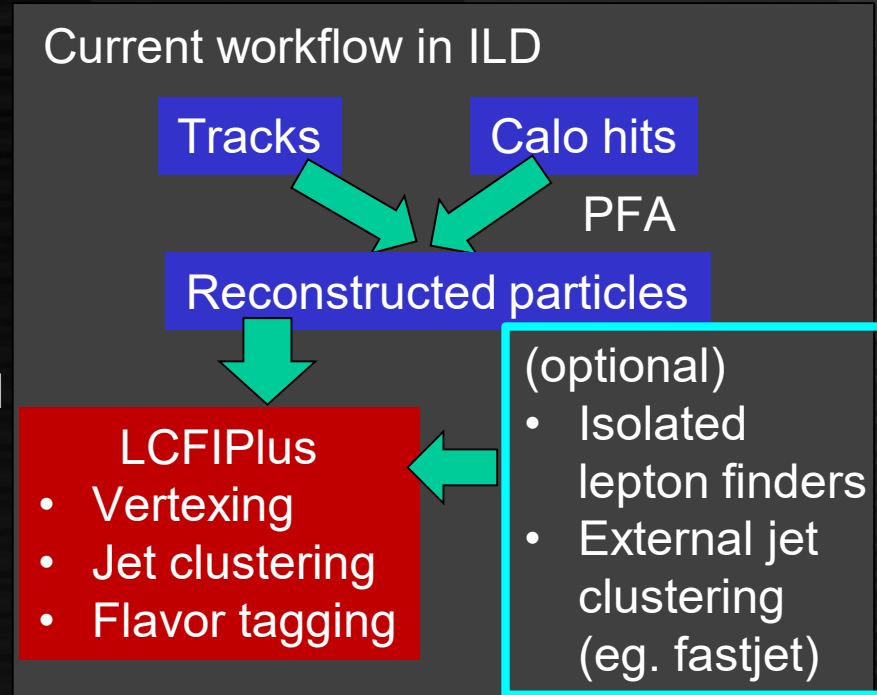


High-level reconstruction at future e^+e^- colliders (LCFIPlus and ongoing works with DNN)

Taikan Suehara
(Kyushu University)

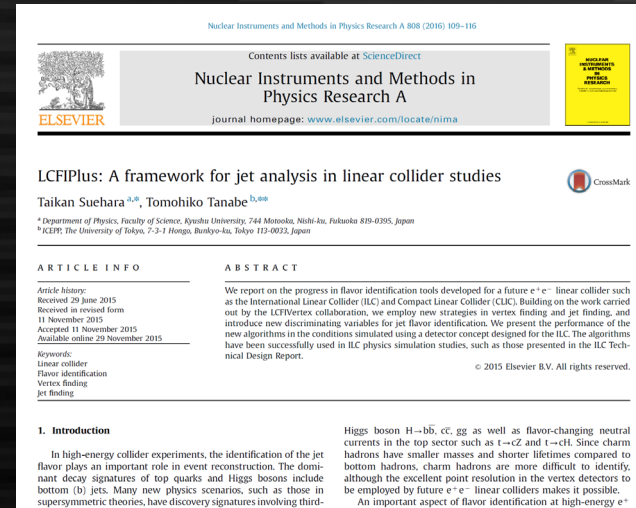
Contents

- LCFIPlus: integrated jet analysis tools since 2013
 - Vertex finding
 - Jet clustering
 - Flavor tagging
- Ongoing studies/plans
 - Attempting updates with DNN
 - Vertex finder with LSTM-based DNN
 - Flavor ID with GNN
 - Calorimeter clustering with GNN
 - Importing ideas from CMS HGCAL



LCFIPlus: introduction

- LCFIVertex developed by UK group
 - <https://doi.org/10.1016/j.nima.2009.08.059>
 - Based on topological clustering (ZVTOP for SLD)
 - Budget cutoff at UK in ~2010 stopped all activities
- LCFIPlus transfers the effort
 - Target: exceed the performance of LCFIVertex
 - More robust to multi-jet environment ($H \rightarrow bb/cc, ZHH$)
 - Published for **ILC DBD** in 2013, also used for **CLIC CDR**
 - Features
 - Build-up secondary vertex finder
 - Integrated jet clustering with vertex information
 - Beam-jet rejection with Durham/Valencia/KT algorithm
 - BDT (by TMVA) for flavor tagging



<https://doi.org/10.1016/j.nima.2015.11.054>



<https://github.com/lcfiplus/LCFIPlus>

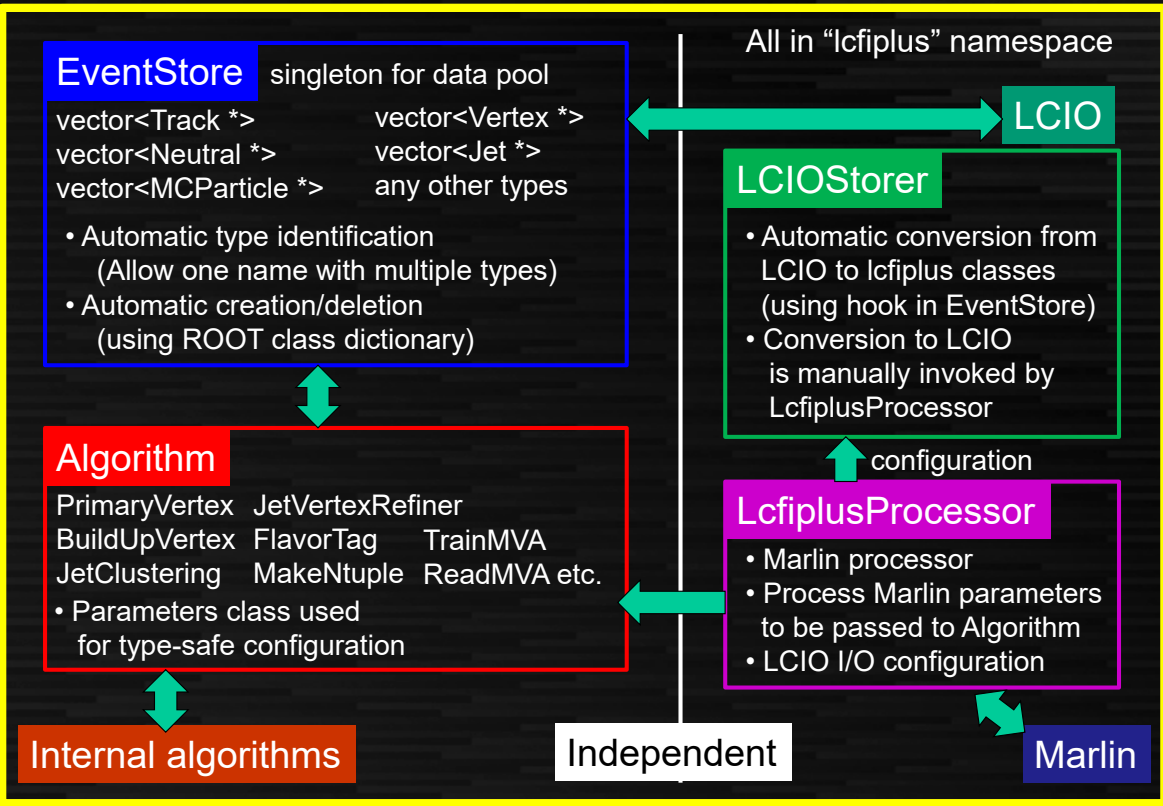
Currently mainly maintained
by R. Yonamine (KEK)

Record of a tutorial:

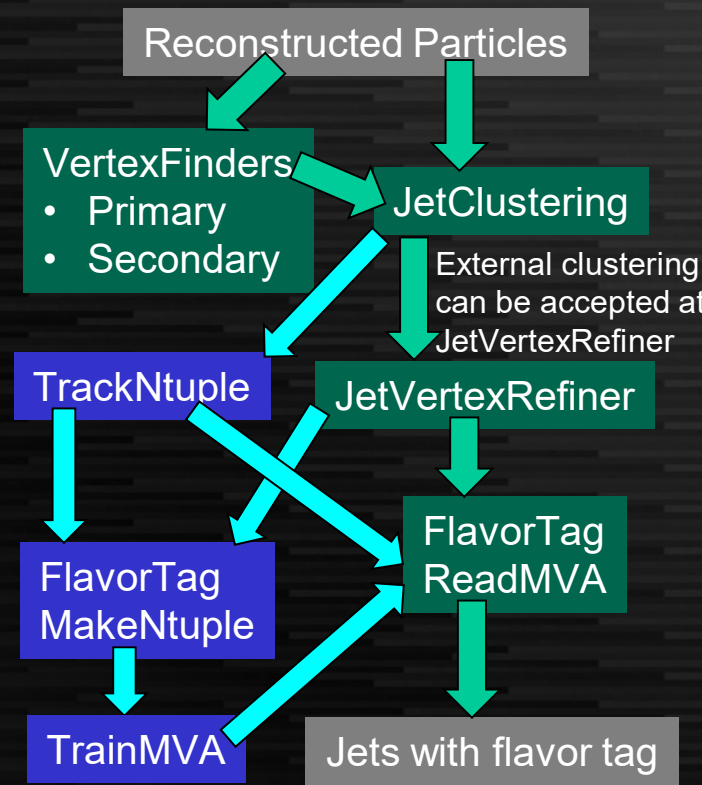
<https://agenda.linearcollider.org/event/9318/>

LCFIPlus framework

 Training chain
 Reconstruction chain



LCFIPlus algorithms



LCIO/Marlin implementation via independent adapters

Vertex finder in LCFIPlus

- PrimaryVertex

- Tear-down method

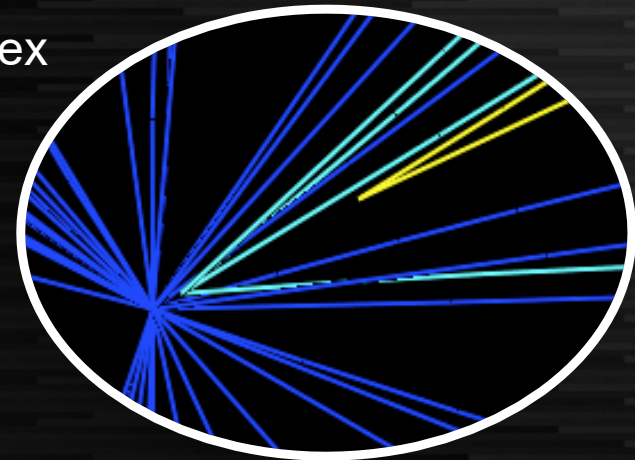
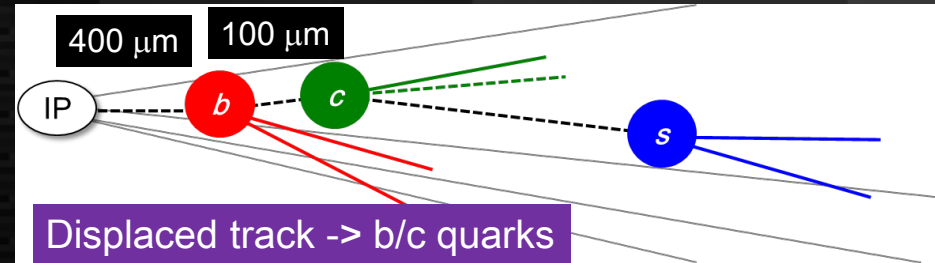
- Fit all tracks with beam constraint
 - Remove a track with worst χ^2
 - Repeat until all tracks within acceptable χ^2

- BuildUpVertex

- Secondary vertex finder to be run after PrimaryVertex

- Build-up method

- Examine every track pair if it is consistent with a vertex (χ^2 etc)
 - List “good” vertex candidates
 - Try to attach more tracks to the candidates
 - “Adaptive vertex fitter” recently added (optional)



Jet clustering / JetVertexRefiner in LCFIPlus

- Durham with beam rejection

$$y_{\text{beam}} = 2E^2\alpha^2(1-\cos\theta)/E_{\text{vis}}^2$$

- Plain Durham (still available, of course)

- kT algorithm

– No need to run it outside any more

- Valencia algorithm

– Intermediate algorithm of Durham and kT

$$d_{ij} = \min(E_i^{2\beta}, E_j^{2\beta})(1 - \cos\theta_{ij})/R^2 \quad d_{iB} = p_T^{2\beta}$$

- JetVertexRefiner

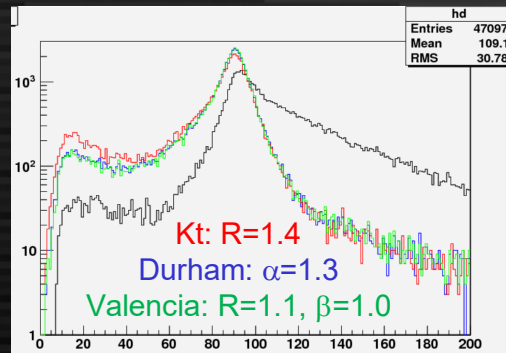
– Assign vertices to jets if jets are created externally

– Looking for “single track vertices”

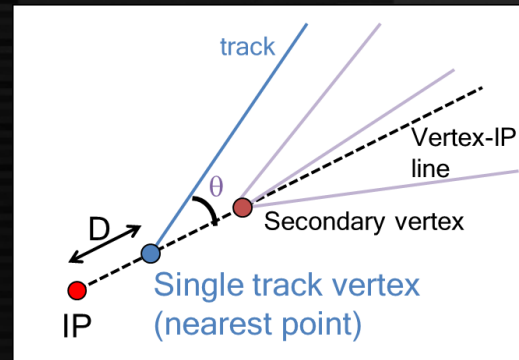
– Limit vertices/jet to two,

by forced combining of existing vertices if # vtx > 3 (based on χ^2)

– “Bness” recently added



Invariant mass of Z
($Z_{\nu\nu} \rightarrow qq_{\nu\nu}$, 500 GeV, 2-jet clustering)



Flavor tagging in LCFIPlus

- Flavor tagging – BDTG in TMVA

- Training algorithms

- TrackNtuple and TrackProb.C
 - To prepare data files to be used in flavor tagging
- MakeNtuple ← TrackProb files
 - To prepare training data
- Train ← MakeNtuple files
 - Run TMVA to train the flavor tagging BDT

- FlavorTag/ReadMVA

- Evaluate RefinedJets with trained BDT data
- Output “n”-likenesses, categories can be defined (default: b, c, other. Can add eg. bb.)

Flavor tagging variables
(selectable in steering file)

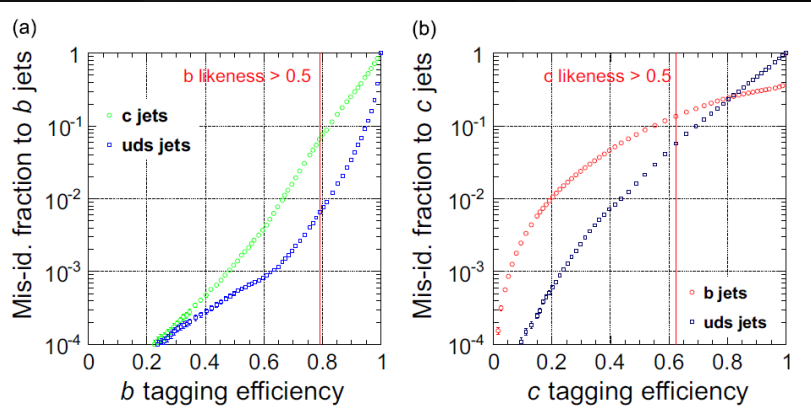
- Categories by # reco vtx / jet (0, 1, 1+1 single track, 2)
- Track variables
d0/z0 significance, pt, joint probability, b/c/q prob., # leptons, track mass
- Vertex variables (only 1+ vtx)
decay length, angle, mass, pt-corrected mass, momentum, multiplicity, probability, significance
- 2-vertex variables (> 1 vtx)
single vertex prob., length and significance of 1st-2nd vertex
- New variables possible

Flavor tagging: performance at ILD

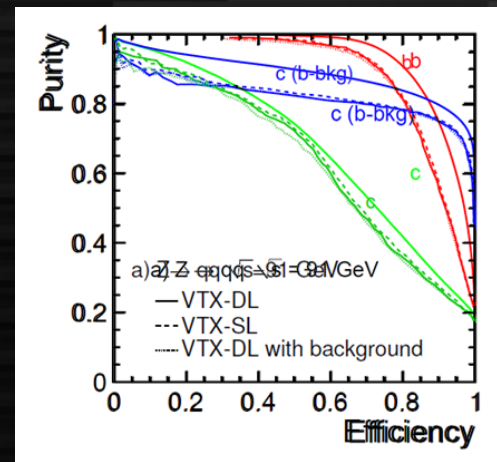
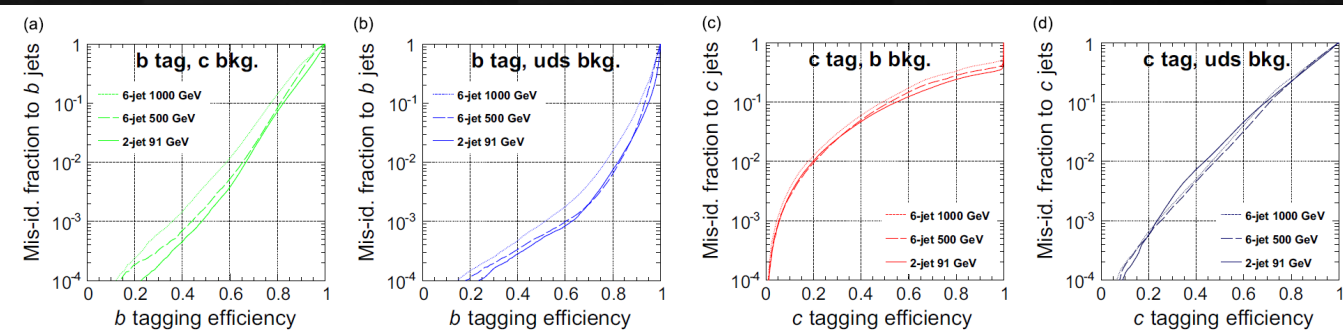
ILD DBD

$Z \rightarrow qq$, 91 GeV

Performance of vertex finder



(#vtx, #pseudo-vtx)	b jet (%)	c jet (%)	uds jet (%)
(0, 0)	21.3	59.3	98.1
(0, 1)	1.61	0.17	0.01
(1, 0)	39.7	39.8	1.80
(1, 1)	13.5	0.54	0.02
(2, 0)	23.8	0.19	0.04



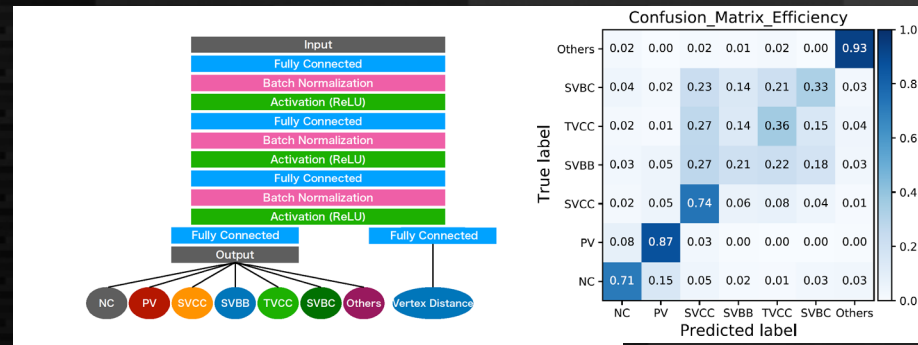
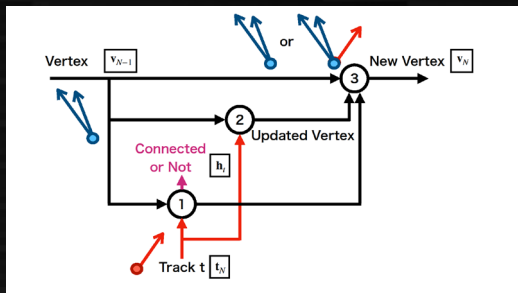
Comparison with LCFIVertex

Prospects of high-level reconstruction with modern DNN technologies

- Flavor tagging
- Particle flow

Vertex finder with DNN (modified LSTM) arXiv:2101.11906 Under review at NIMA

- Replacing build-up vertex with DNN
 - Track-pair classification with NN
 - Track association with LSTM-based RNN with attention
 - Similar perf. to LCFIPlus obtained w/o vertex fitting

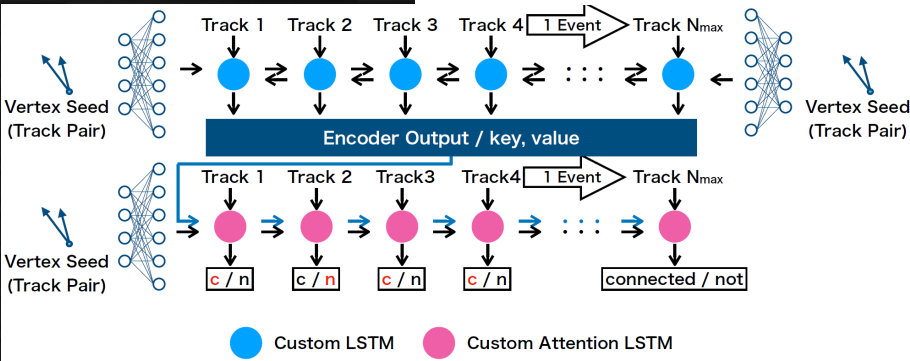


Vertex reconstruction eff. with DNN

Criteria / True label	Primary	Bottom	Charm	Others
All tracks	307 657	187 283	180 143	42 888
In secondary vertex	2.2%	63.3%	68.4%	9.5%
- of same decay chain		62.3%	67.2%	
- of same parent		38.1%	36.2%	6.4%

Vertex reconstruction eff. with LCFIPlus for comparison

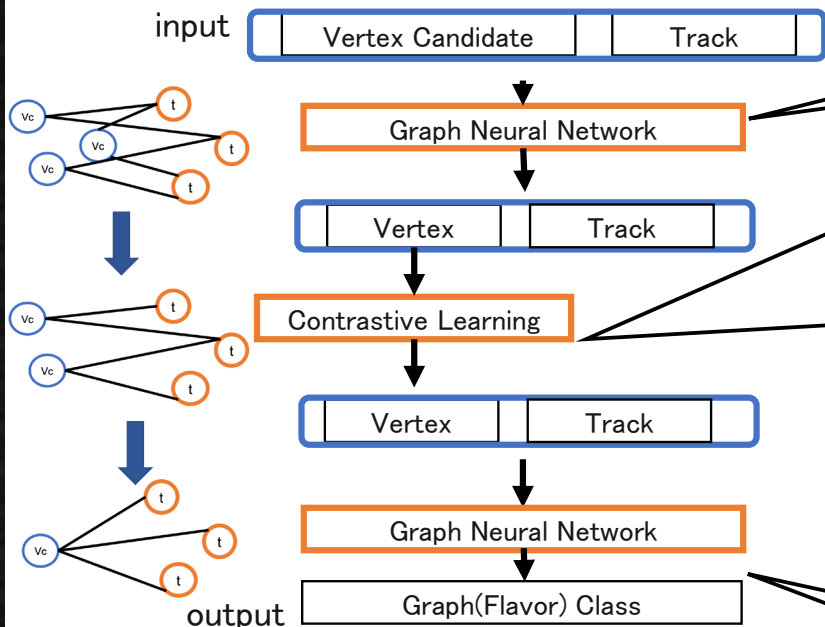
Criteria / True label	Primary	Bottom	Charm	Others
All tracks	307 657	187 283	180 143	42 888
In secondary vertex	0.2%	57.9%	60.3%	0.5%
- of same decay chain		57.5%	59.9%	
- of same parent		34.0%	37.2%	0.3%



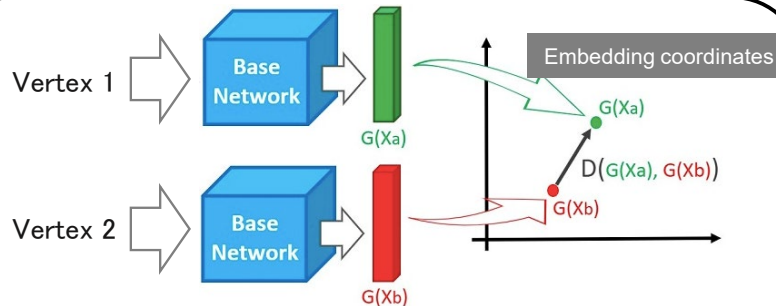
Vertex finder with GNN (ongoing work)

Input graph nodes:

- Any track pairs forming vertex candidates
- Tracks connected to vertex candidates



Judge whether a vertex candidate is an existing vertex or not as node classification.



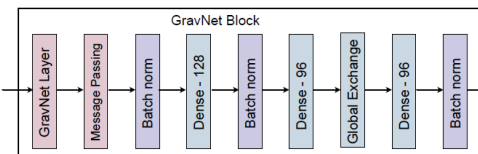
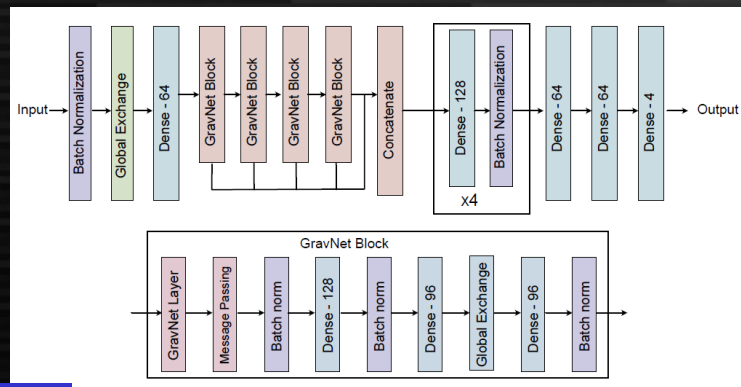
Deep Metric learning brings vertices of the same parent particle closer together.

Classify the entire graph into flavor types

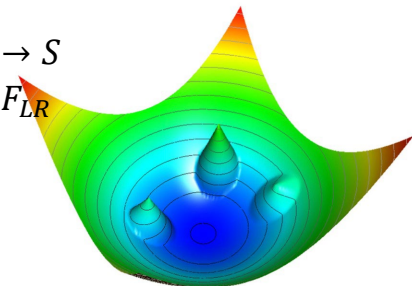
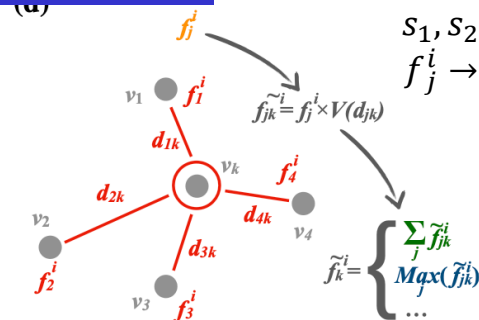
Calorimeter clustering (\rightarrow PFA) with DNN (GravNet for CMS HGCal)

Collaboration with
L. Gray (Fermilab) et al.

- Aiming to replace PandoraPFA
 - With Modern DNN algorithm
 - Easier to use for detector optimization
 - Using timing information
 - Integrated PID
- CMS HGCal reconstruction
 - GravNet layer x 4 (concatenated)
 - Contrastive learning on virtual coordinate
 - Object condensation loss function
 - Make one “condensation point” per cluster
 - Attract hits to condensation points with the same true cluster in (another) virtual coordinate



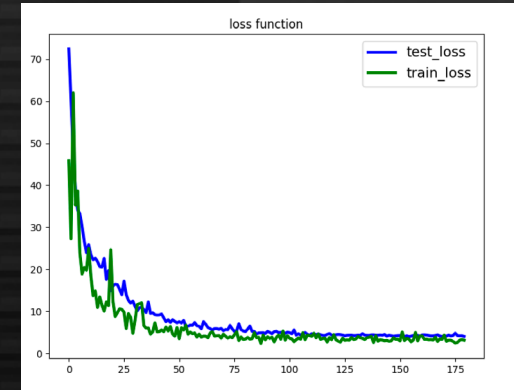
GravNet layer



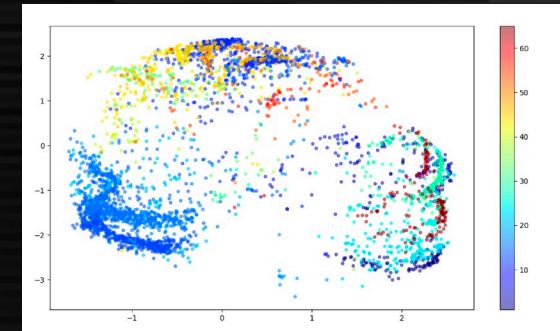
Object condensation

Attempt of introducing GravNet to ILC PFA

- First try with $Z \rightarrow qq$ at 500 GeV with ILD fullsim
 - Using ECAL + HCAL barrel hits
 - True clusters as MCParticles associated to hits converted to ROOT Tree \rightarrow numpy array
 - Output of “condensation variable” $\beta + n$ coordinate ($n = 2$ for drawing) per each hit
 - Any additional outputs (energy, time, PID, ...) possible
 - Clustering based on distance from condensation points (simple method for final clustering from HGCal script)
- What to implement / study
 - Track-cluster matching to complete PFA (how to input tracks, maybe special input)
 - Performance comparison with PandoraPFA
 - Detector optimization including timing layers



Training curve by epoch



Same color = same true cluster

Summary

- LCFIPlus is a successful software for jet analysis used since 2013- present in LC studies
- Reasonable performance
 - b-tag performance: 80% eff with 10% c-bkg, 60% with 1% c-bkg
 - c-tag performance: 50% eff with 10% b-bkg
 - Relatively robust to 4- or 6- jet environment and high energy jets
 - Can be usable at any efficiency point by selecting MVA threshold
- Update of flavor tagging with DNN being investigated
- Update of PFA with HGCal algorithm is also ongoing

LCFIPlus Summary

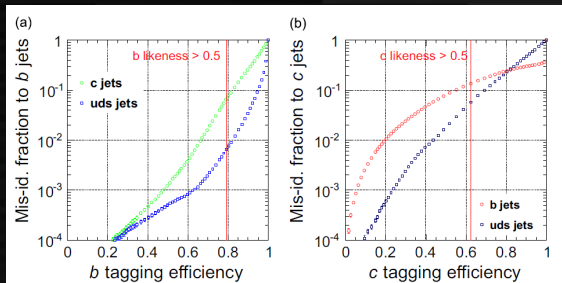
<https://doi.org/10.1016/j.nima.2015.11.054>

<https://github.com/lcfiplus/LCFIPlus>

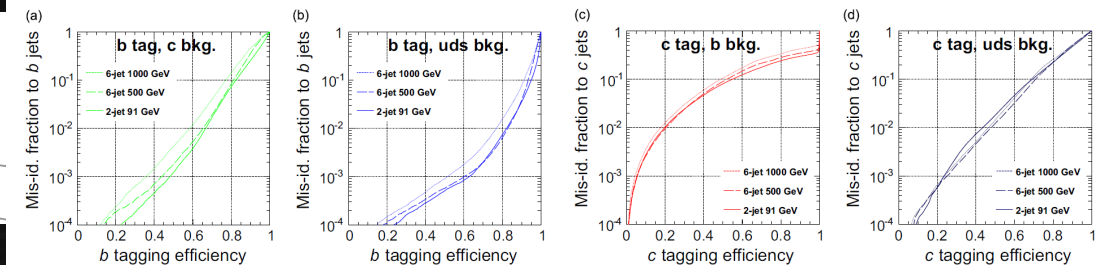
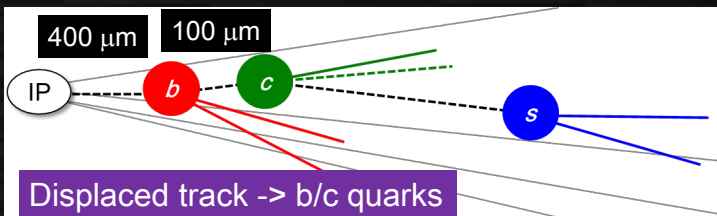
- Standard flavor tagging software for ILD/CLIC since 2013
- Integrated software for
 - Vertex finding (tear-down, build-up)
 - Jet clustering (incl. beam rejection)
 - Flavor tagging (b/c/q separation) based on BDT/TMVA
- Data/process model independent of LCIO/Marlin structure

Performance of vertex finding

(#vtx, #pseudo-vtx)	b jet (%)	c jet (%)	uds jet (%)
(0, 0)	21.3	59.3	98.1
(0, 1)	1.61	0.17	0.01
(1, 0)	39.7	39.8	1.80
(1, 1)	13.5	0.54	0.02
(2, 0)	23.8	0.19	0.04



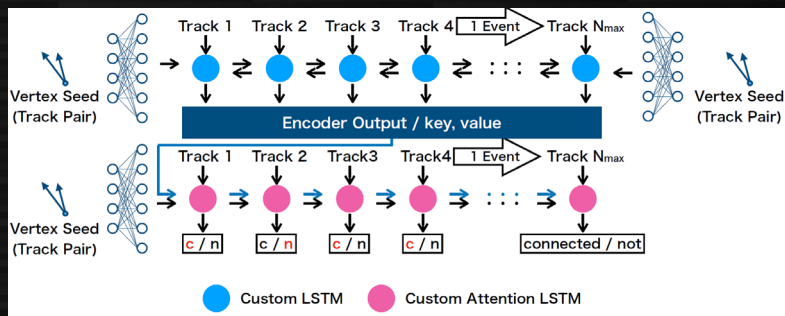
Performance of b/c tagging with $Z \rightarrow qq$ 91 GeV and 6q 1000/500 GeV



DNN Summary

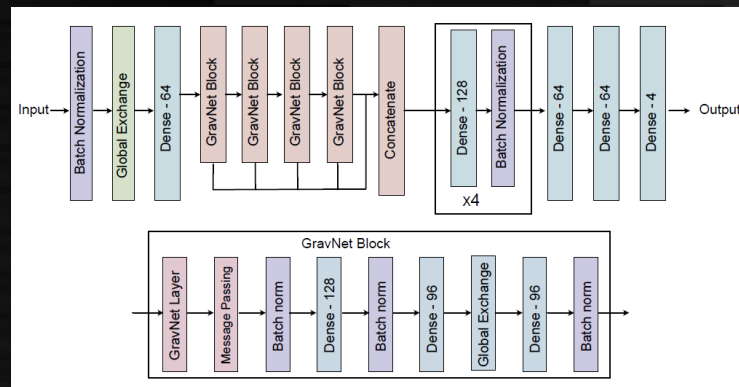
Vertex finding / Flavor tagging

- Vertex finder with LSTM
 - Following LCFIPlus strategy
 - Replacing pair selection and track association with NN and LSTM
 - Similar performance to LCFIPlus
- Flavor tagging with GNN ongoing



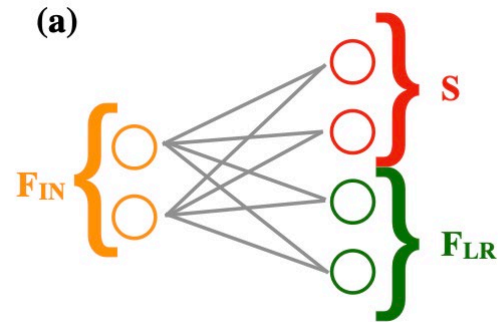
Calo clustering / PFA

- Introducing GravNet / object condensation (from HGCal) to ILC
- GravNet: contrastive GNN layers
- Object condensation: to gather true hits
- First implementation done: performance evaluation ongoing



GravNet - Network -

- ▶ Input Data : $B \times V \times F_{IN}$
 - B : Number of examples including in a batch
 - V : Number of hits for each detector
 - F_{IN} : Number of the features for each hit
- ▶ S : Set of coordinates in some learned representation space
- ▶ F_{LR} : learned representation of the vertex features



Loss function - Network Learning -

- ▶ The object condensation approach :
Aiming to accumulate all object properties in condensation points

Assignment of vertices
for each shower

Identification of noise

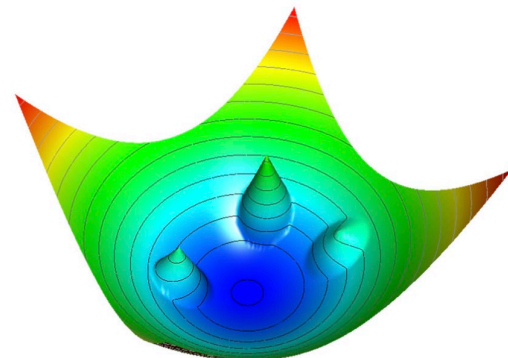
Update of
loss term

- ▶ The value of β_i ($0 < \beta_i < 1$) is used to define a charge q_i per vertex i
 $q_i = \operatorname{arctanh}^2 \beta_i + q_{\min}$ ($\beta_i \rightarrow 1 : q_i \rightarrow +\infty$)

- ▶ The charge q_i of each vertex belonging to an object k
defines a potential $V_{ik}(x) \propto q_i$

- ▶ The force affecting vertex j can be described by $q_j \cdot \nabla V_k(x_j) = q_j \nabla \sum_{i=1}^N M_{ik} V_{ik}(x_j, q_i)$

$$M_{ik} = \begin{cases} 1 & (\text{vertex } i \text{ belonging to object } k) \\ 0 & (\text{otherwise}) \end{cases}$$



Loss function

- ▶ The L_V has the minimum value for $q_i = q_{\min} + \epsilon \forall i$
- ▶ To enforce one condensation point per object, and none for background or noise vertices, the following additional loss term L_β is introduced :

$$L_\beta = \frac{1}{K} \sum_k (1 - \beta_{\alpha k}) + s_B \frac{1}{N_B} \sum_i^N n_i \beta_i,$$

s_B : hyperparameter describing the background suppression strength
 K : Maximum value of objects
 N_B : Number of background
 n_i : Noise tag (if noise, it equals 1.)

- ▶ The loss terms are also weighted by $\text{arctanh}^2 \beta_i$:

$$L_p = \frac{1}{\sum_{i=1}^N \xi_i} \cdot \sum_{i=1}^N L_i(t_i, p_i) \xi_i, \text{ with}$$

$$\xi_i = (1 - n_i) \text{arctanh}^2 \beta_i.$$

p_i : Features
 $L_i(t_i, p_i)$: Loss term (Difference between true labels and outputs of network)

Loss function

- ▶ If high efficiency instead of high purity is required :

$$L'_p = \frac{1}{K} \sum_{k=1}^K \frac{1}{\sum_{i=1}^N M_{ik} \xi_i} \cdot \sum_{i=1}^N M_{ik} L_i(t_i, p_i) \xi_i.$$

- ▶ In practice, individual loss terms might need to be weighted differently :

$$L = L_p + s_C(L_\beta + L_V)$$

Update of
loss term

Identification of noise

Assignment of vertices
for each sower

Input graph nodes:

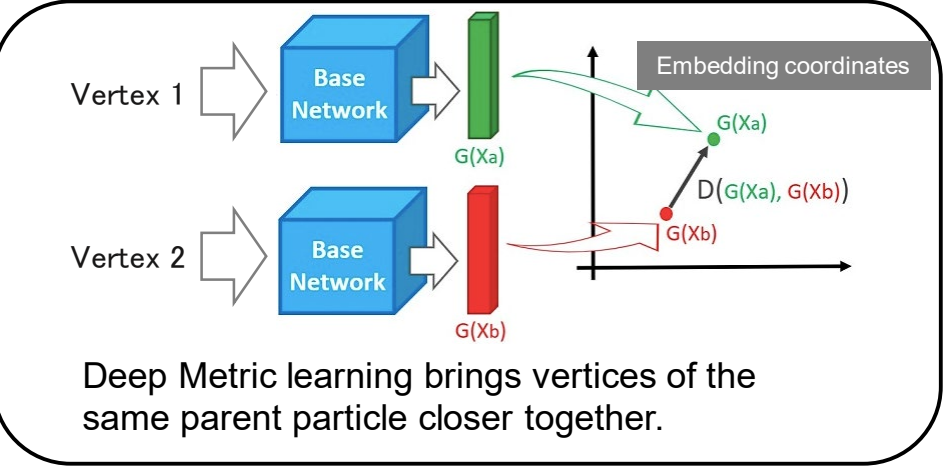
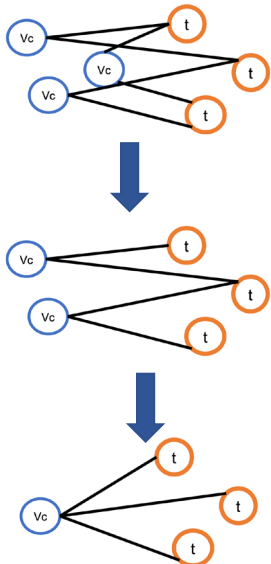
- Any track pairs forming vertex candidates
- Tracks connected to vertex candidates



Judge whether a vertex candidate is an existing vertex or not as node classification.



 : Graph data



Deep Metric learning brings vertices of the same parent particle closer together.

Classify the entire graph into flavor types