Workflow training (PyTorch) & inference (iLCSoft/Marlin)

- run PV & SV finder, jet clustering and vertex refinement of LCFIPlus
- run Marlin processor that calculates and stores features needed for the flavor taggers

} iLCSoft/Marlin

- store variables in **root files** with four trees (charged, neutral, jets, sv)

} iLCSoft/Marlin

Training (python scripts & PyTorch):

- convert trees in root files to **pandas dataframes**, do some **checks** and **cleaning,** store dataframes in **hdf5-files**
- do **further pre-processing** and **training** in **PyTorch**
- use **torch library to convert trained model** into model that can be used in C++

Inference (iLCSoftMarlin)

- store variables via **PIDHandler** (**not optimal** in terms of memory, might be changed)
- run **Marlin processor** for **tagging with ParticleNet Model**
  - **read feature values** from PIDHandler
  - **store them** in the **vectors needed by the ParticleNet Model** (coordinates of const., features of the const., coordinates of SV, features of SV)
  - **convert vectors to torch tensors** and do the **pre-processing**
  - do the **inference** with the **converted model**
  - store output again using PIDHandler
- run Marlin processor to store outputs in trees and histogram that can be used to calculate ROCs etc.

ParticleNet: input features

jet constituents: coordinates

$\Delta\eta$, $\Delta\Phi$

jet constituents: features

$\Delta\eta$, $\Delta\Phi$

$\log(p_T)$, $\log(E)$, $\log(p_T/p_T^{jet})$, $\log(E/E^{jet})$, $\vec{p}_{track} \cdot \vec{p}^{jet}/p^{jet}$

ΔR

q

isElectron, isMuon, isChargedHadron, isNeutralHadron, isPhoton

impact parameter & significances

track used in PV?

lepton related variables

pid variables

$E_{HCAL}/E_{HCAL+ECAL}$

$\chi 2$/ndf

28 input features

secondary vertices: coordinates

$\Delta\eta$, $\Delta\Phi$

secondary vertices: features

$\Delta\eta$, $\Delta\Phi$

$\log(p_T)$, E_{SV}/E_{jet}, E_{SV}

η

m_{SV}

$N_{tracks\ in\ SV}$

$\chi 2$/ndf

impact parameters &significances

cos(flight direction$_{SV}$, $\vec{p}_{SV}$)

14 input features

2 SVs & all jet constituents considered, no ordering of inputs

DeepJet: input features

global variables

p^{jet}, p_T^{jet},

$N_{charged\ jet\ const.}$, $N_{neutral\ jet\ const.}$, N_{SV}

additional global variables from LCFIPlus

21 input features

charged jet constituents

p^{track}/p^{jet}, p_T^{track} (rel. jet), $\vec{p}^{track} \cdot \vec{p}^{jet}/p^{jet}$

ΔR(track, jet)

impact parameter & significances

track reconstructed in PV?

lepton related variables

pid variables

$\chi 2$/ndf

19 input features

neutral jet constituents

$p^{neutral\ const.}$, $p^{neutral\ const.}/p^{jet}$

ΔR(jet, neutral const.)

is photon?

$E_{HCAL}/E_{HCAL+ECAL}$

5 input features

secondary vertices

m_{SV}

$N_{tracks\ in\ SV}$

ΔR(SV, jet)

E_{SV}/E_{jet}, E_{SV}

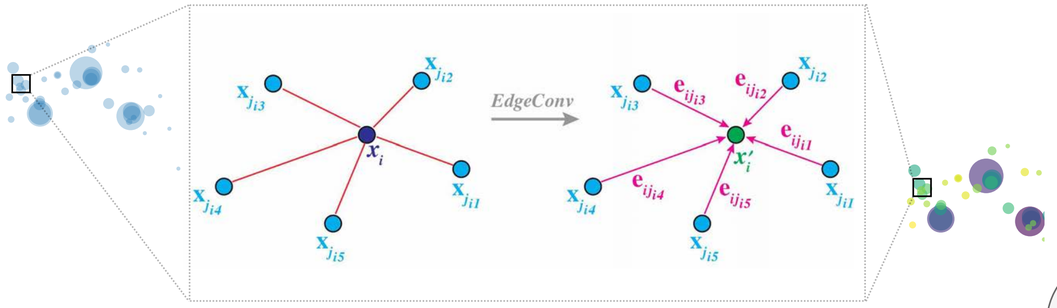
cos(flight direction$_{SV,}$ $\vec{p}_{SV}$)

3D IP and significance

$\chi 2$, ndf

10 input features

ParticleNet



- treat jet as „particle cloud“
- input: **jet constituents**

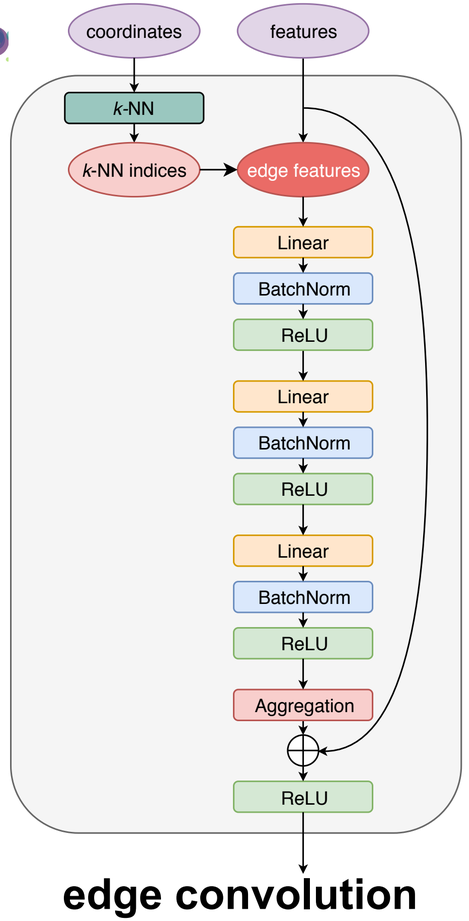
key building block: **edge convolution**

- particle cloud: graph, each point: vertex, connections between each point & k nearest neighboring points: edges
- learn an „**edge feature**“ for each pair:

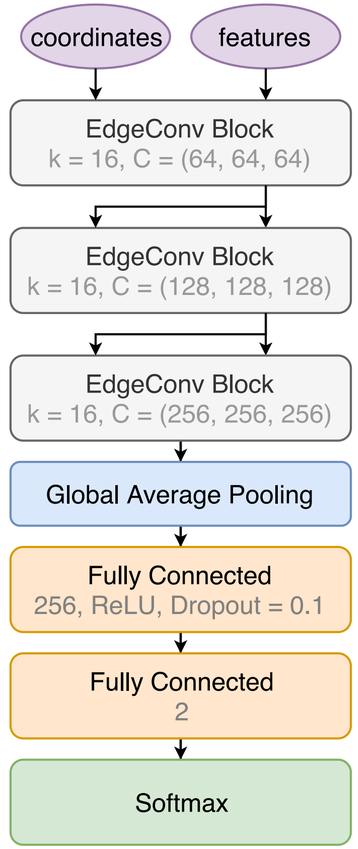
$$e_{ij} = \mathrm{MLP}(x_i, x_j)$$

- **MLP**: parameters **shared among all edges**
- **aggregation** of edge features: $x_i' = \mathrm{mean}_j\, e_{ij}$

arXiv:1902.08570, *Pushing the Limit of Jet Tagging With Graph Neural Networks,* Huilin Qu, talk at ML4Jets2021, July 7, 2021

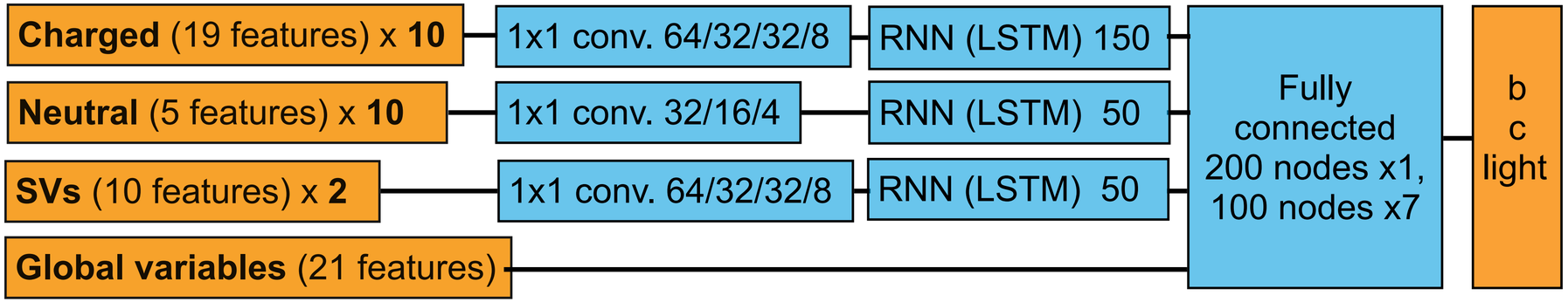


edge convolution



ParticleNet

DeepJet: architecture



- classify jets into **three classes**: b jets, c jets & light jets
- **ordering of input particles** by (as applied in CMS)
  - impact parameter significance for charged jet constituents
  - shortest angular distance to a secondary vertex (by momentum if there is no secondary vertex) for neutral jet constituents
  - flight distance significance for secondary vertices