Jet flavor identification for FCCee

[clearly very relevant for FCC-hh as well]

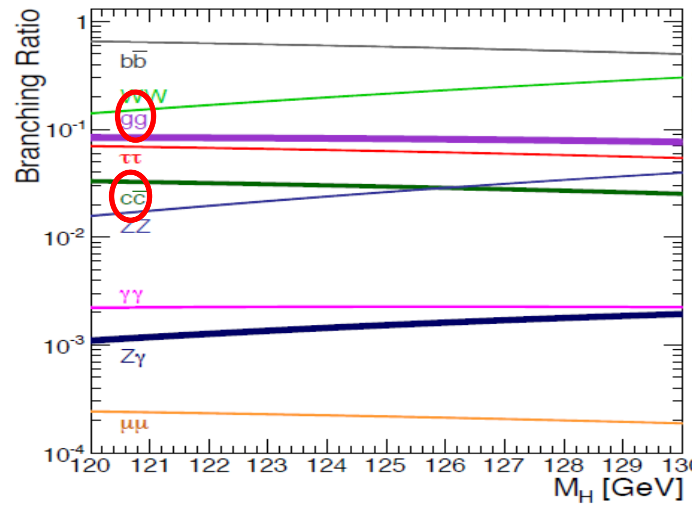
6th FCC-ee Physics Workshop
January 2023

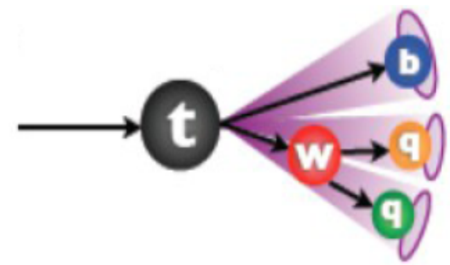
Franco Bedeschi, Michele Selvaggi, LG
[EPJ C 82 646 (2022) link]

New members: Andrea Del Vecchio, Laurent Forthomme, Dolores Garcia

Physics motivation

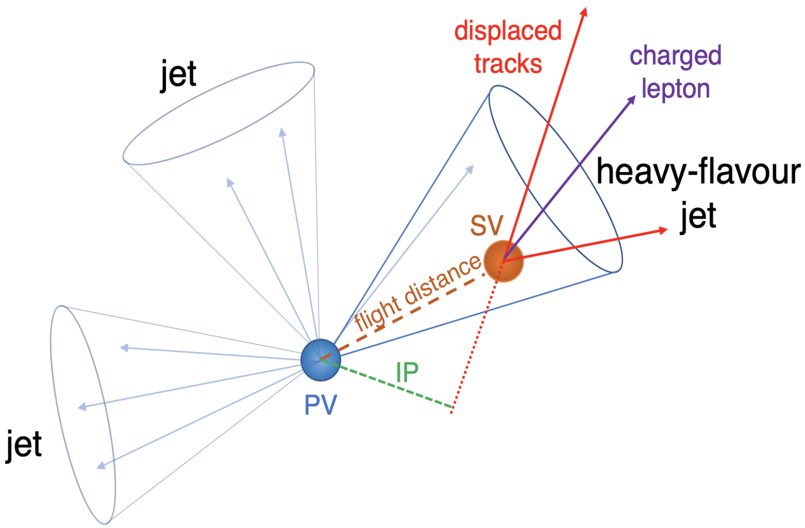
- Flavour tagging essential for the e^+e^- program, e.g.:
  - Higgs Sector:
    - (HL-)LHC can access 3rd gen. couplings and a few of 2nd generation
    - Future e^+e^-: Measure Higgs particle properties and interactions in challenging decay modes
      - E.g. cc, 1st gen quarks/fermions, gg [?]
  - Top quark physics [if E_{CM} sufficient]
    - Precise determination of top properties [mass, width, Yukawa]
  - QCD Physics
    - strong coupling (a_S), event shapes ..
    - modelling of hadronization, MC tuning, …
  - ….





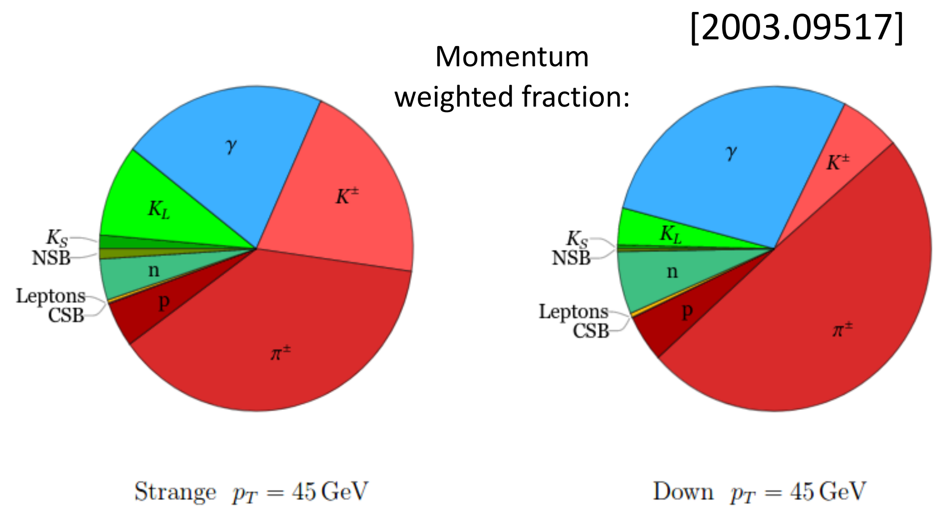
Basics for jet flavor identification

bottom/charm-tagging



- Large lifetime
- Displaced vertices/tracks
- Large track multiplicity
- non-isolated e/μ

strange-tagging



- Large Kaon content
  - Charged Kaon as track:
    - K/pi separation
  - Neutral Kaons:
    - $K_S \rightarrow \boldsymbol{\pi\pi}$, K_L

In the beginning: unclear what correlations existed among these

Ingredients for powerful jet taggers

- Detectors
  - Pixel/tracking systems: Little material, spatial resolution, precise track alignment
  - PID systems: timing capabilities, energy loss (gas/silicon)
- Algorithm design
  - optimal representation of jet/ optimal processing of detector signal & evt reconstruction
  - sophisticated algorithm design

- **Scope of this work:**
  Build a general framework for developing flavor tagging algorithms for future colliders [eg., e^+e^-]
  - Fast detector simulation
    - Understand detector requirements/ optimize design
      - eg., vertexing and PID capabilities of the FCCee detectors
  - Develop a versatile flavor tagger
    - Identify with high purity gluons, and ud, strange, charm, bottom quarks
    - Baseline: ParticleNet jet tagging algorithm
      - Results shown for FCCee & IDEA

Detectors characteristics in e^+e^-

- e^+e^- colliders provide a very clean environment
  - Lower occupancy , no pileup
- Powerful detectors:
  - Pixel/tracking detectors tailored for b/c tagging
    - Higher granularity wrt to LHC detectors
      - ATLAS/CMS pixel size: O(~100x100 μm^2)
    - Less tracking material
      - ~0.4% X_0/layer CMS/ATLAS Pixel, ~0.15-0.2% X_0/layer in e^+e^- detectors
      - better impact parameter resolution/ less multiple scattering
      - CMS/ATLAS Pixel resolution: O(10) μm; ~2-5 μm in e^+e^-
  - PID capabilities
    - dE/dx (Si tracker), dN/dx (Drift)
    - Time-of-flight [timing layer]

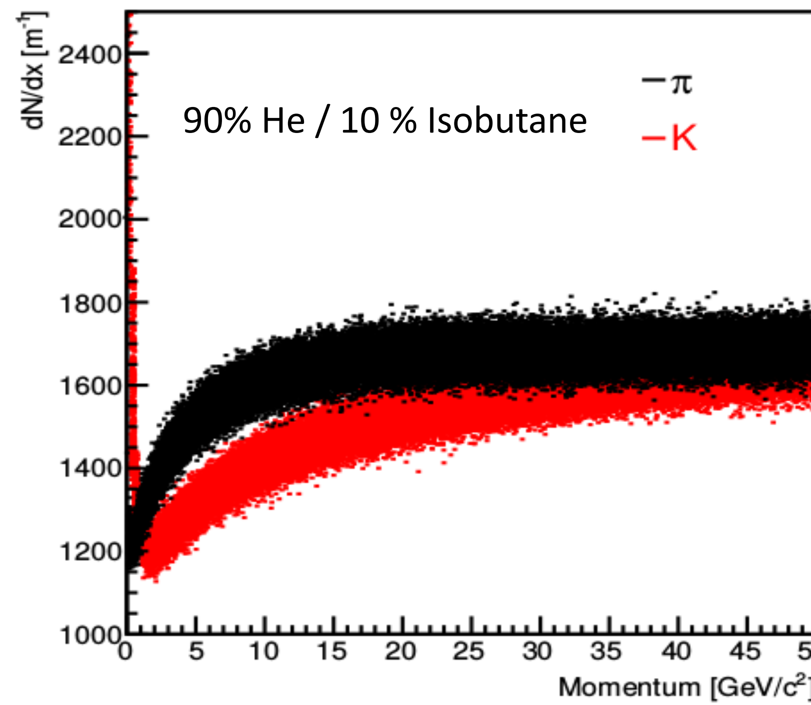
Numbers indicative concepts evolve rapidly

→ e^+e^-: Natural place to explore potential of jet tagging algorithms using advanced ML
→ A step further: Consider reconstructing the full event in e^+e^-

Particle ID: Cluster counting (dN/dx)

- Count number of **primary ionization** clusters along track path
- Avoids large Landau flukes
- Requires high granularity
- module added in Delphes

IDEA detector:



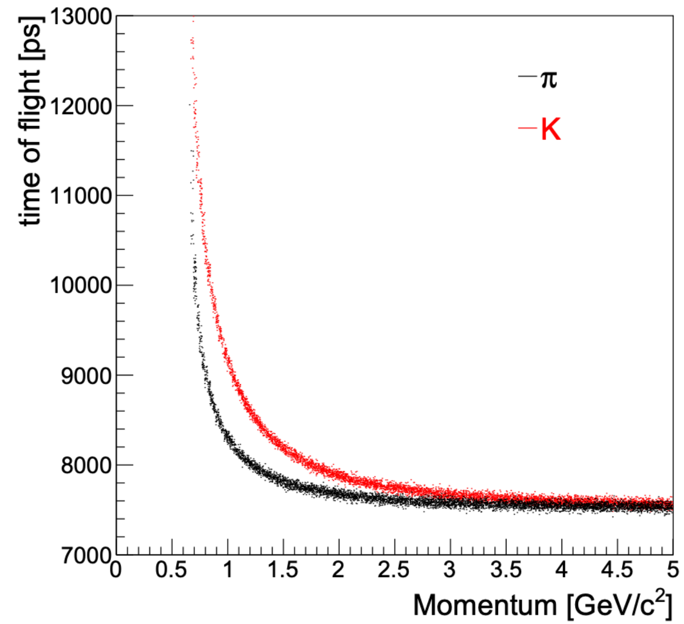

```
###################
# Cluster Counting
###################

module ClusterCounting ClusterCounting {

  add InputArray TrackSmearing/tracks
  set OutputArray tracks

  set Bz $B

  ## check that these are consistent with DCHCANI/DCHNANO parameters in TrackCovariance module
  set Rmin $DCHRMIN
  set Rmax $DCHRMAX
  set Zmin $DCHZMIN
  set Zmax $DCHZMAX

  # gas mix option:
  # 0:  Helium 90% - Isobutane 10%
  # 1:  Helium 100%
  # 2:  Argon 50% - Ethane 50%
  # 3:  Argon 100%

  set GasOption 0

}
```

Particle ID: TOF

- Good K/π separation at low-momenta:

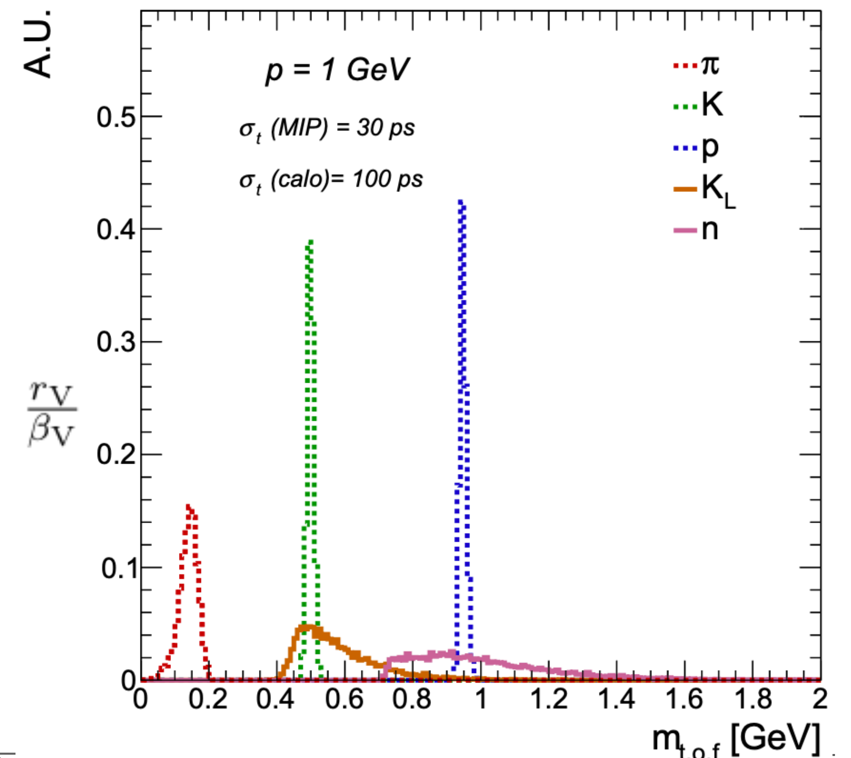
$$t_{\text{flight}} \equiv t_{\text{F}} - t_{\text{V}} = \frac{L}{\beta} = \frac{L\sqrt{p^2+m^2}}{p}$$

- Assumption on vertex time
[crucial for highly displaced K_s]

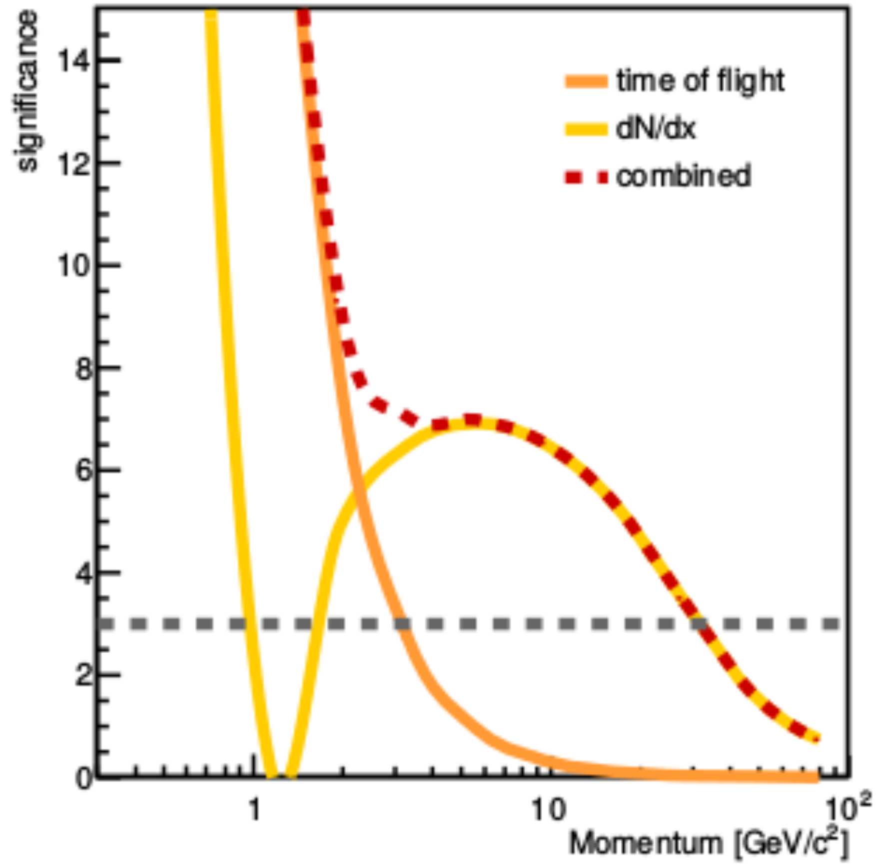



```
########################################
#   Time Of Flight Measurement
########################################

module TimeOfFlight TimeOfFlight {
  set TrackInputArray TimeSmearing/tracks
  set VertexInputArray TruthVertexFinder/vertices

  set OutputArray tracks

  # 0: assume vertex time tV from MC Truth (ideal case)
  # 1: assume vertex time tV = 0
  # 2: calculate vertex time as vertex TOF, assuming tPV=0

  set VertexTimeMode 2
}
```

$$t_{\text{V}} = \frac{r_{\text{V}}}{\beta_{\text{V}}}$$



ParticleID: Combined



3σ K/π separation for tracks w/ p<30 GeV

Designing a Graph-based tagger

- **Jet representation:** critical for powerful jet tagging algorithms
  - **In theory:** A spray of particles produced by the hadronization of q and g
  - **Experimentally:** A cone of reconstructed particles in the detector

- Reminder: Current and future experiments have / will have a **PF-based** event reconstruction
  - **Output:** mutually exclusive list of particles
    - Rich set of info/particle
      - Energy/momentum, position
      - Displacement, particle type
      - timing
      - …

[O(50) properties/particle]
x [~50-100 particles/jet]
~O(1000) inputs/jet

- **Until recently:** Jet taggers based on human-inspired higher-level observables
  - Inputs to cut-based or simple ML-based algorithms
- Move to **particle-based jet tagging**: i.e. exploit directly the PFcands
  - explore full potential of event reconstruction and detector granularity

Designing a Graph-based tagger (II)

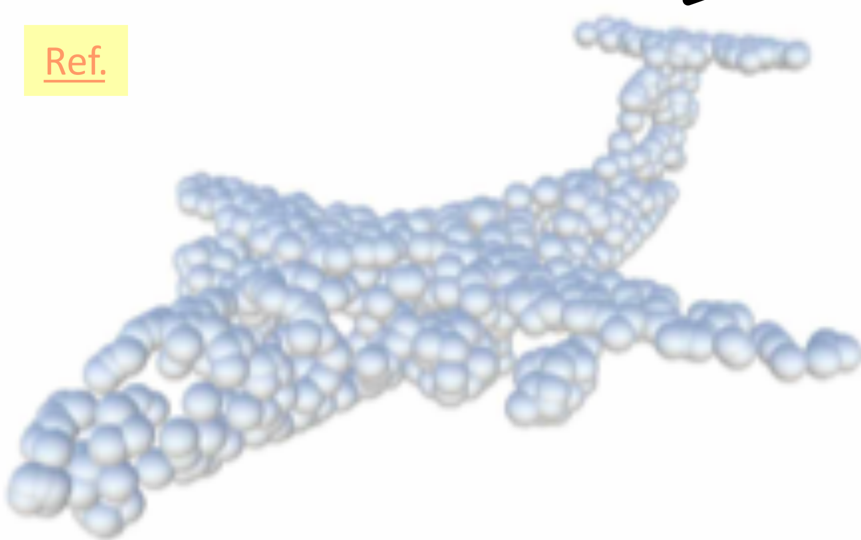
- **Jet:** intrinsically <u>unordered set</u> of particles with relationships b/w the particles
  - i.e. human-chosen ordering not optimal

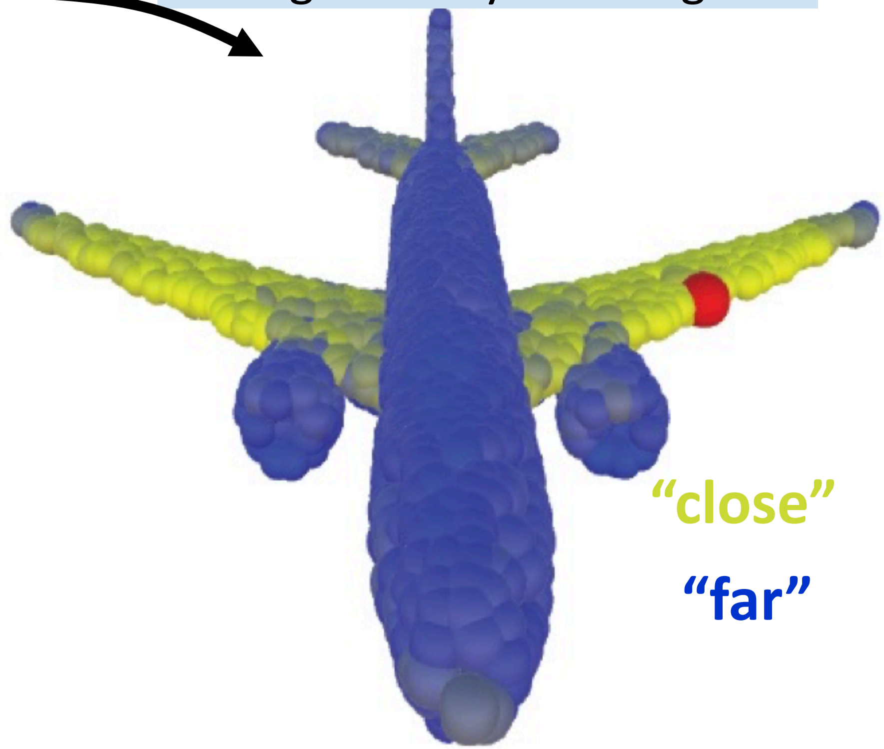
- A very active research area in ML community: **Point clouds**

Represent the object as a set of "points"

Group points based on similarity [usually using ML]
e.g. Identify the wings

Ref.

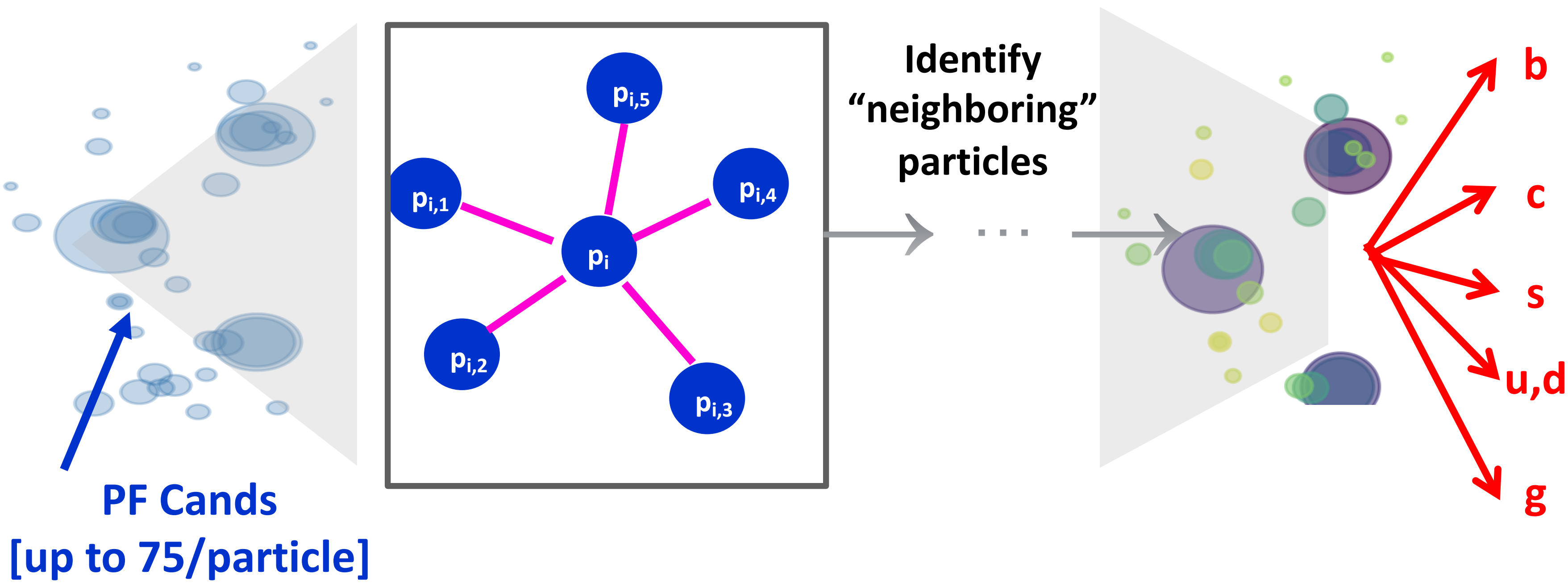




ParticleNet(-ee)

H. Qu and LG
PRD 101 056019 (2020)
F. Bedeschi, M. Selvaggi, LG
EPJ C 82 646 (2022)

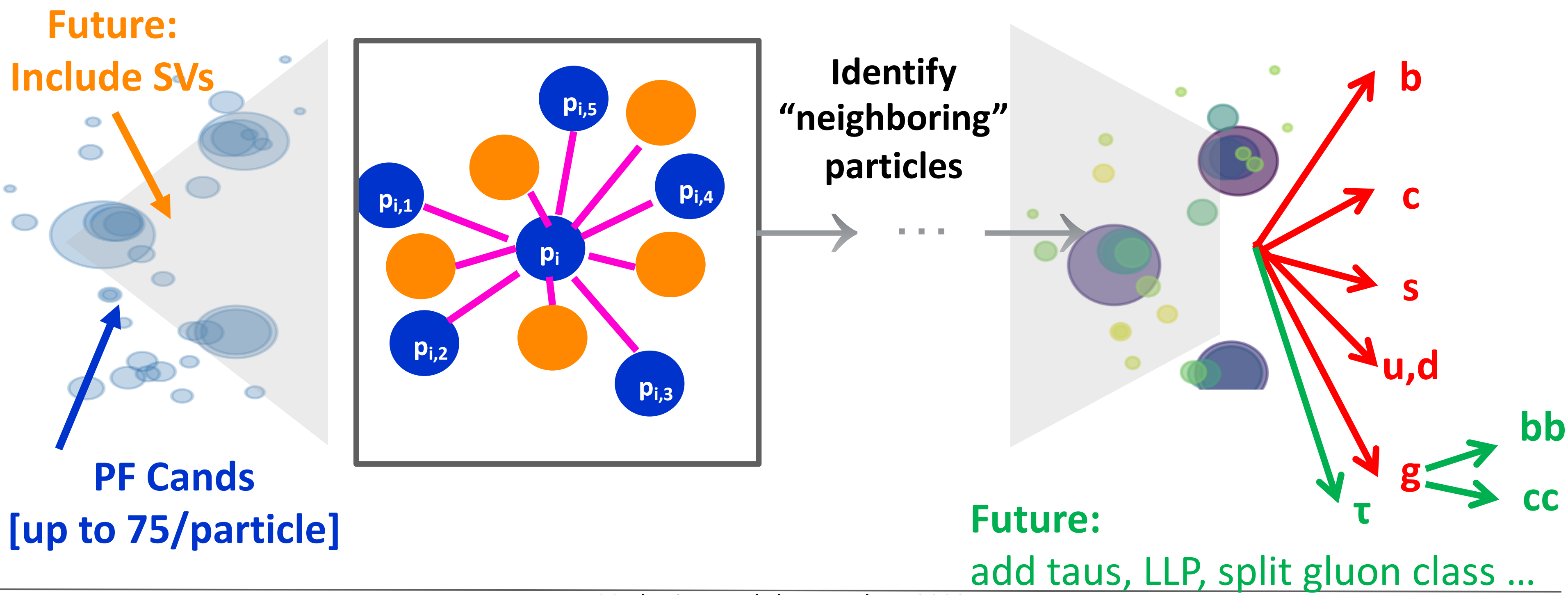
- Jet representation: *"Point Cloud"* → *"Particle Clouds"*
  - Treat the jet as an unordered set of particles
- Algorithm design: Graph Neural Networks
  - Particle cloud represented as a graph
    - Each particle: **vertex** of the graph; Connections between particles: the **edges**
- Follow a hierarchical learning approach
  - First learn local structures → then move to more global ones



ParticleNet(-ee)

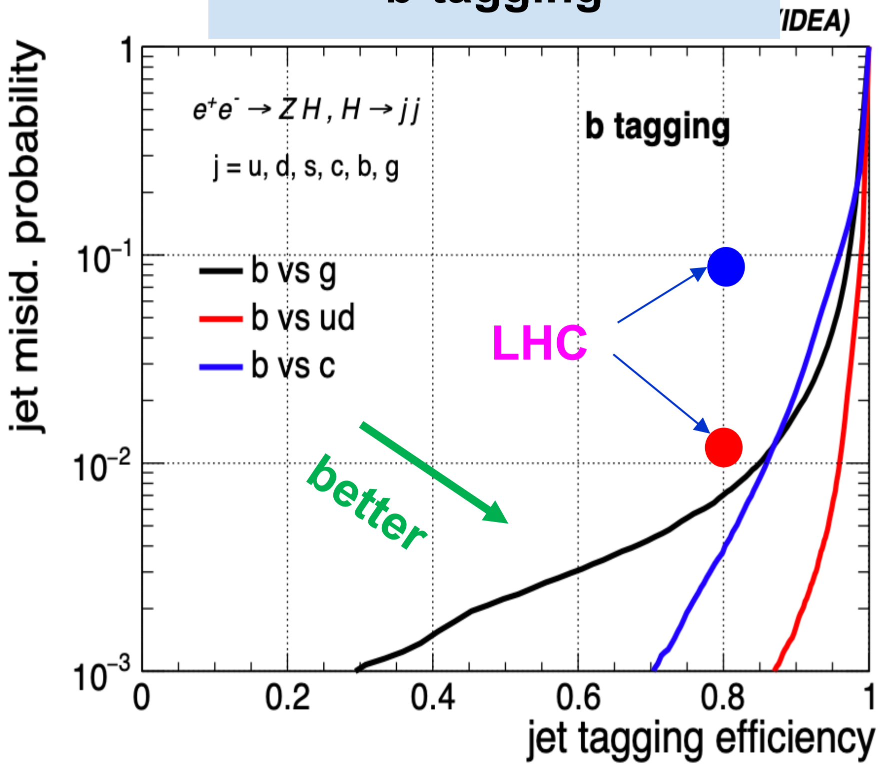
H. Qu and LG
PRD 101 056019 (2020)
F. Bedeschi, M. Selvaggi, LG
EPJ C 82 646 (2022)

- Jet representation: *"Point Cloud"* → *"Particle Clouds"*
  - Treat the jet as an unordered set of particles
- Network architecture: Graph Neural Networks
  - Particle cloud represented as a graph
    - Each particle: **vertex** of the graph; Connections between particles: the **edges**
- Follow a hierarchical learning approach
  - First learn local structures → then move to more global ones



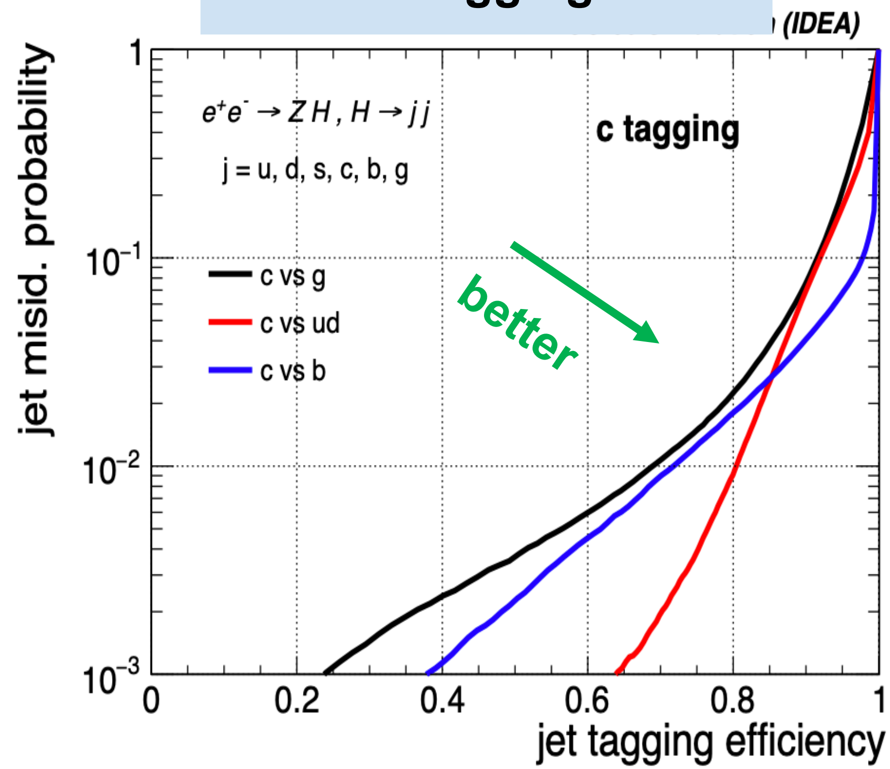
ParticleNet@FCCee: b/c tagging

b-tagging



WP	Eff (b)	Mistag (g)	Mistag (ud)	Mistag (c)
Loose	90%	2%	0.1%	2%
Medium	80%	0.7%	<0.1%	0.3%

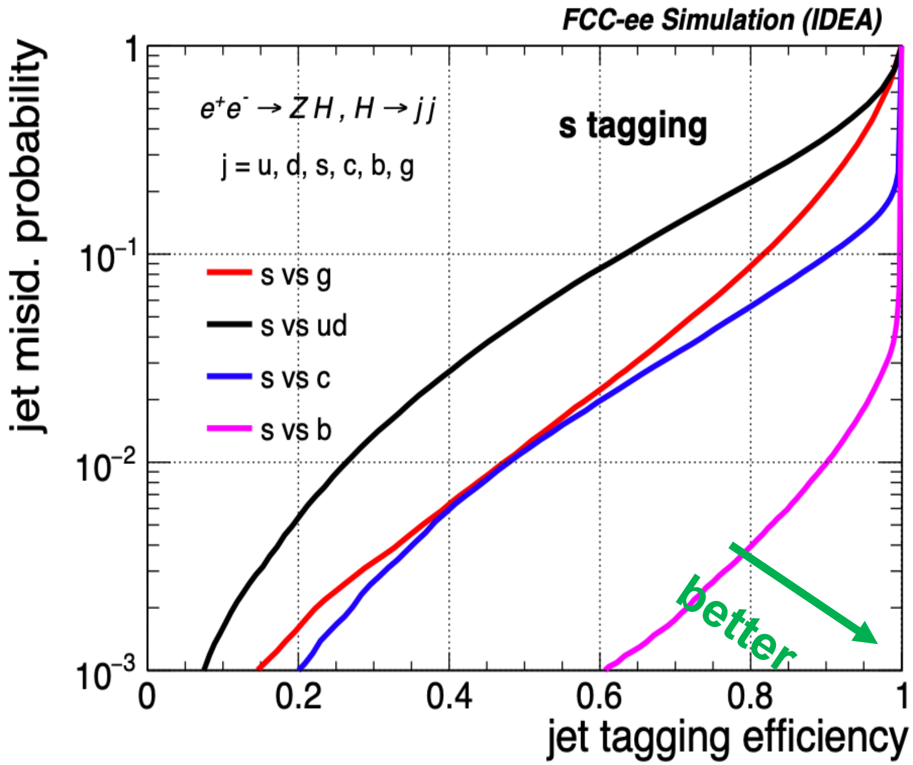
c-tagging



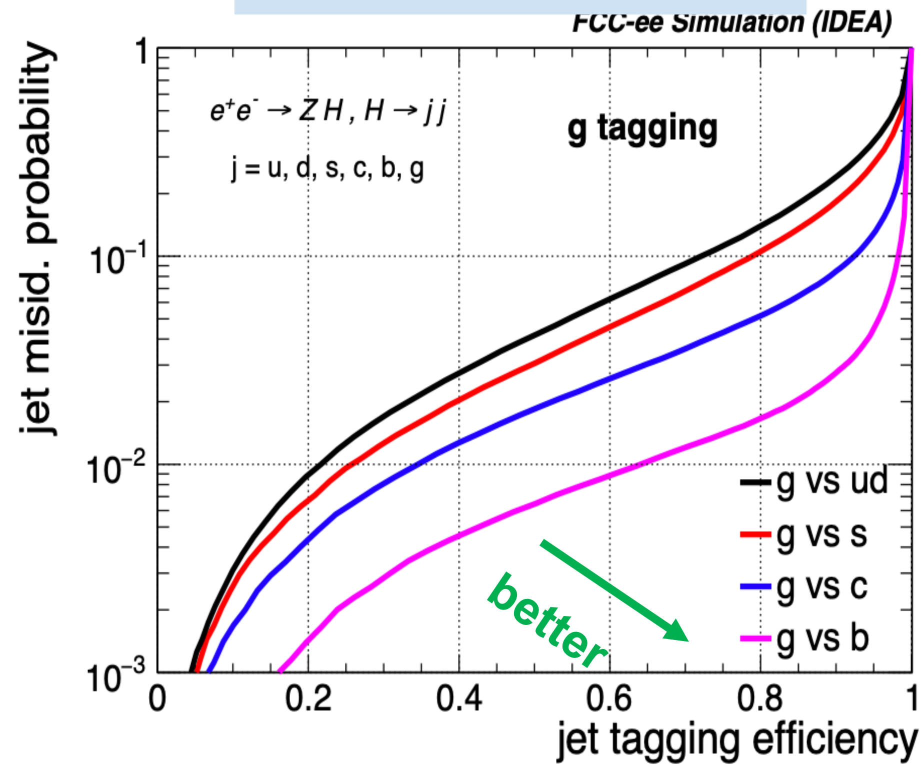
WP	Eff (c)	Mistag (g)	Mistag (ud)	Mistag (b)
Loose	90%	7%	7%	4%
Medium	80%	2%	0.8%	2%

ParticleNet@FCCee: s/g tagging

strange-tagging



gluon -tagging

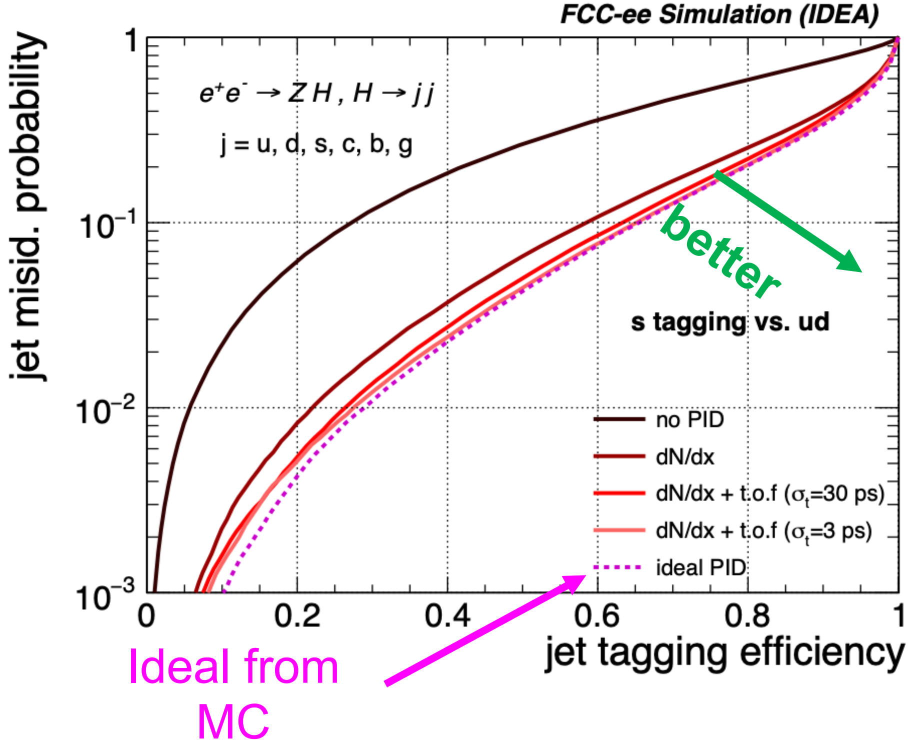


WP	Eff (s)	Mistag (g)	Mistag (ud)	Mistag (c)	Mistag (b)
Loose	90%	20%	40%	10%	1%
Medium	80%	9%	20%	6%	0.4%

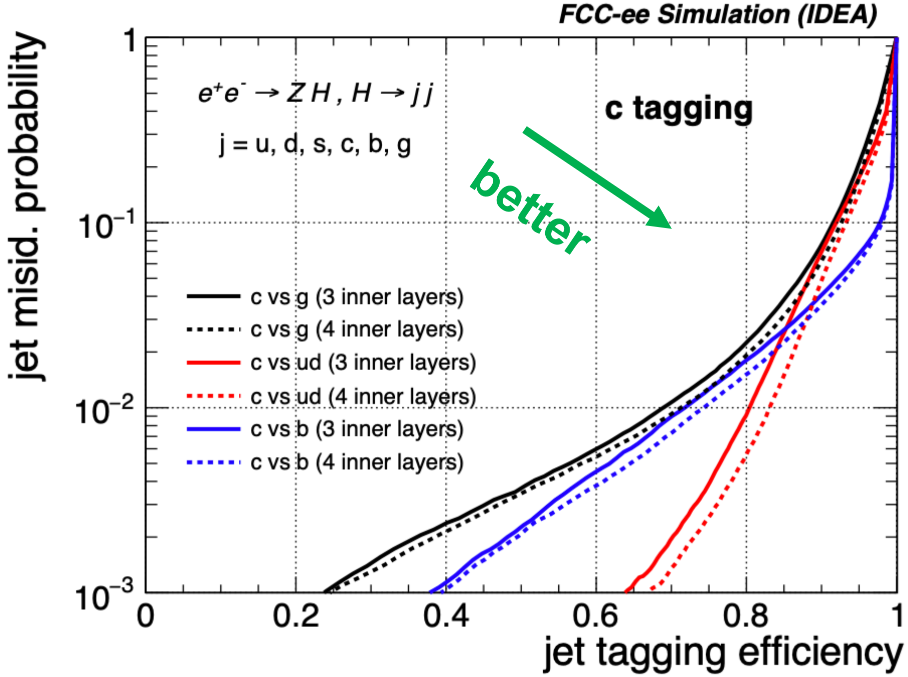
WP	Eff (g)	Mistag (ud)	Mistag (c)	Mistag (b)
Loose	90%	25%	7%	2.5%
Medium	80%	15%	5%	2%

Impact of detector configurations

Strange tagging [PID]



c-tagging [PIX layers]



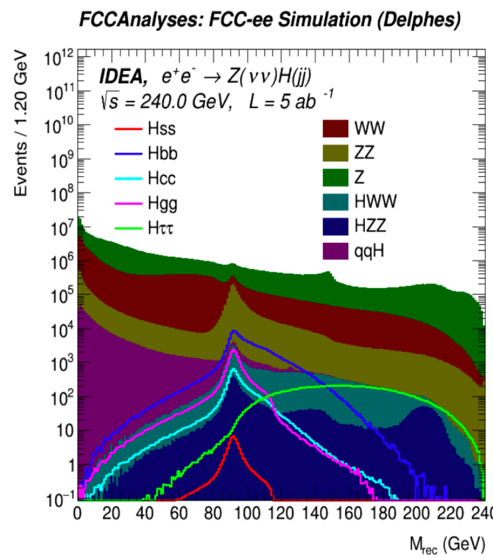
- dN/dx brings most of the gain additional gain w/ TOF (30ps)
  - TOF (3ps): marginal improvement
  - dN/dX + TOF(30ps) ~ perfect PID
- Additional pixel layer:
  - 2x improved BKG rejection in c-tagging
  - marginal/no improvement in b-tagging

Teaser from the analysis front

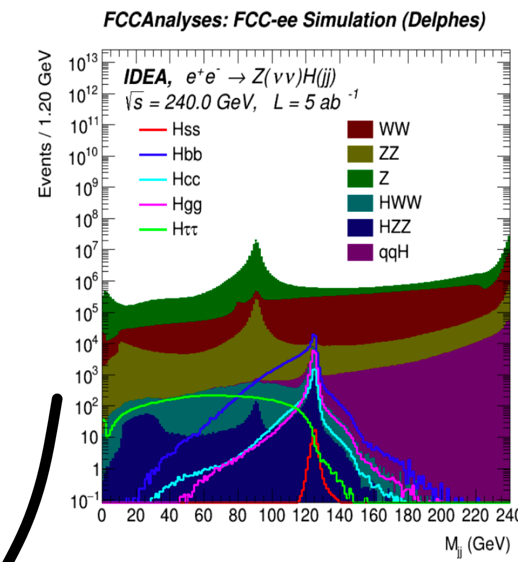
- Tools fully incorporated in FCCSW [details]
  - Example: Z(→vv)H(→qq)

Signal extraction: 2D fit

m(rec)

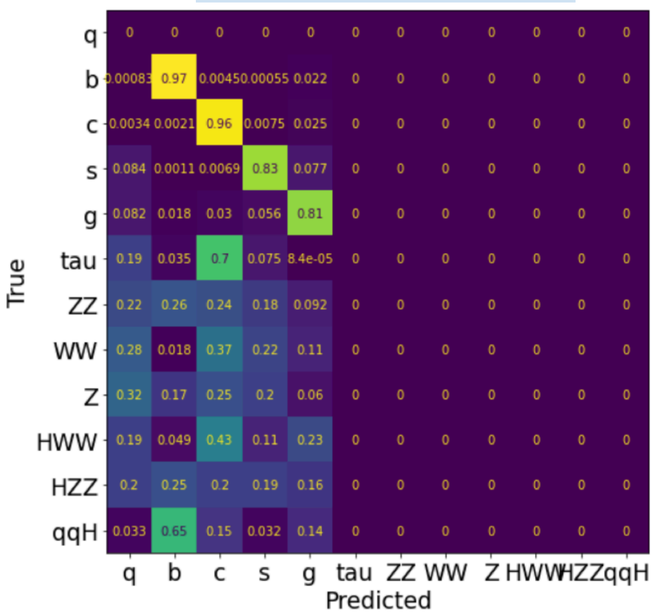


m(jj)



Categorize events: bb, cc, ss, gg
Sub-categories w/ different S/B

ParticleNet-ee



Results @ 5ab^{-1}
(syst: 5% BKG, 0.1% SIG)

Z(→vv) H(→qq)	bb	cc	ss	gg
δμ/μ (%)	0.4	2.9	160	1.2

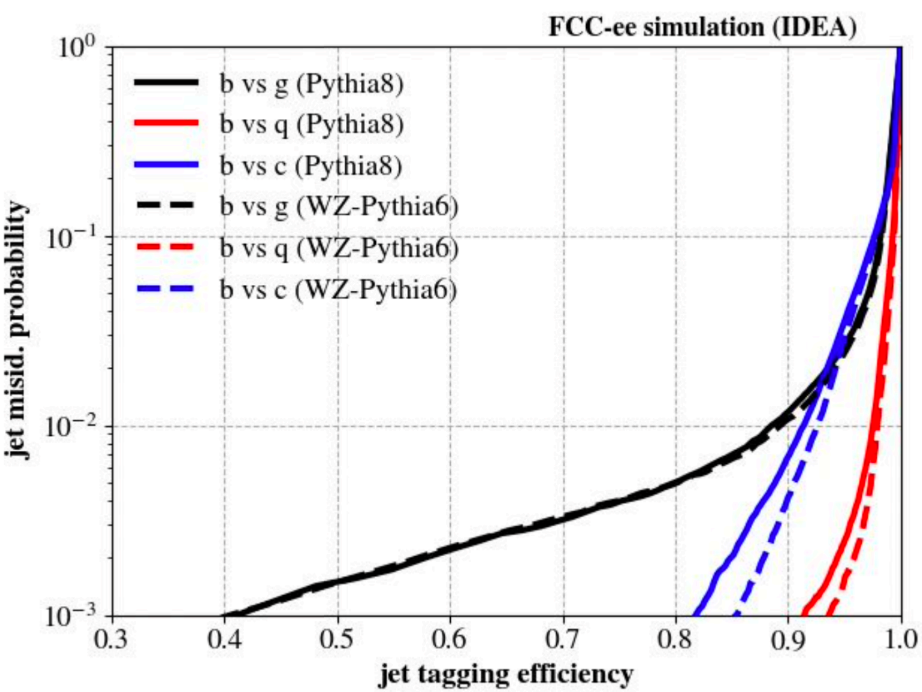
*** |κ_s|<1.9**

More on Friday:
G. Marchiori

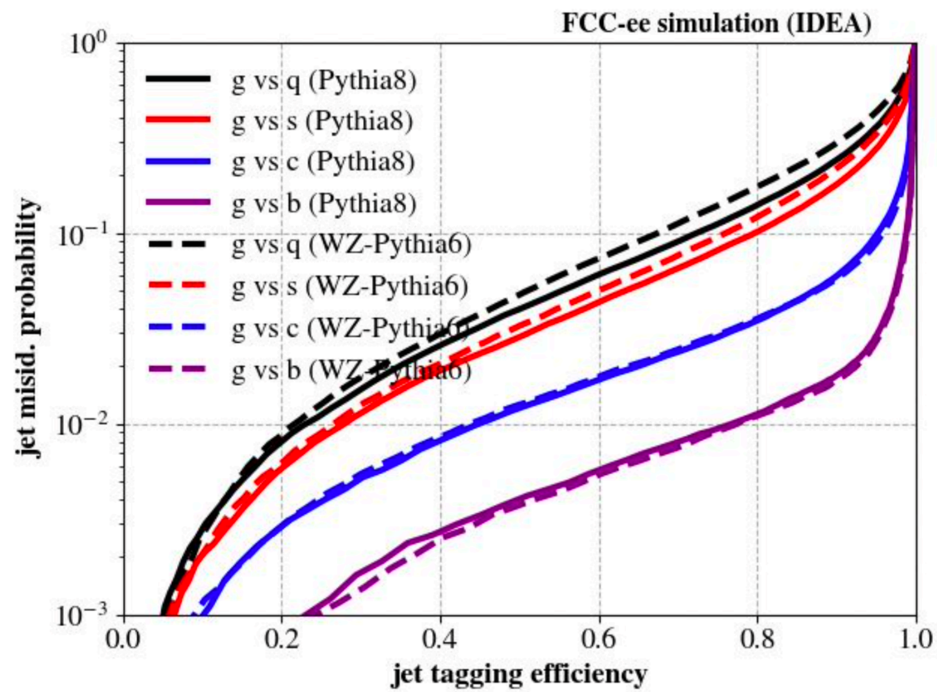
Robustness

- ParticleNet-ee trained using ***Pythia 8*** samples
  - tested on ***Pythia 8*** [solid lines]
  - tested on ***WZ-Pythia6*** [dashed lines]

b-tagging



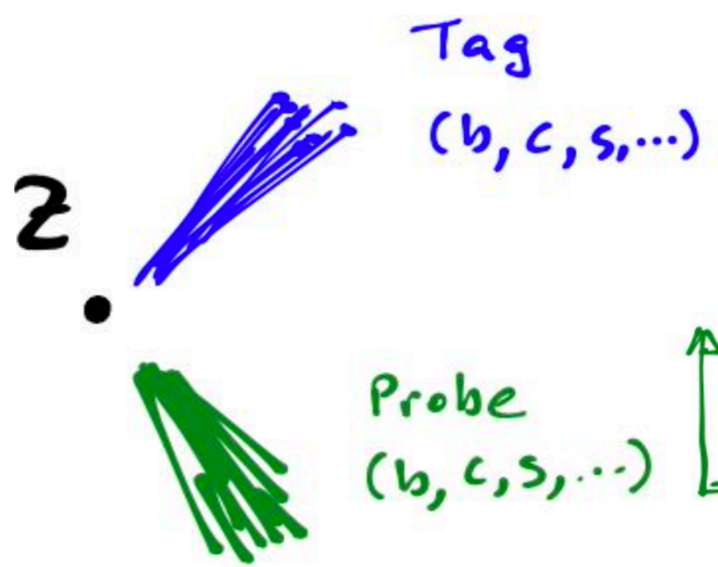
gluon -tagging



- Modest dependence on choice of generator
  - still many tricks in our bag to further reduce the dependence

Improving robustness

- Current development relies solely on MC
  - Full control of class definition, lot's of [MC] data [~2M jets/ jet flavor]
    - but: MC != Data; potentially lead to large uncertainties
    - NB: it's also not Full SIM ..
- Another route: Use data
  - [Obvious] advantage: much smaller syst unc.
- How: Tag-and-probe @ Z pole
  - First: **Tag** one of the two jets with high purity
    - e.g. by using a pretrained MC-based algo
  - Then: create a **training** sample using the **2^{nd} jet (probe)**.



FCC-ee @ Zpole

Z→hadrons	~70%	0.7×10^6 M
→ uu/cc	~12%/flavor	8.4×10^4 M/ flavor
→ dd/ss/bb	~15%/flavor	1.1×10^5 M/ flavor

Improving robustness (II)

- Take into account tagging performance [& mistag rates]
  - NB: Each class does not have to be 100% pure on specific jet flavor or have the same population

Best case: b-tagging

WP	Eff (b)	Mistag (g)	Mistag (ud)	Mistag (c)
Loose	90%	2%	0.1%	2%
Medium	80%	0.7%	<0.1%	0.3%

"Worst" case: s-tagging

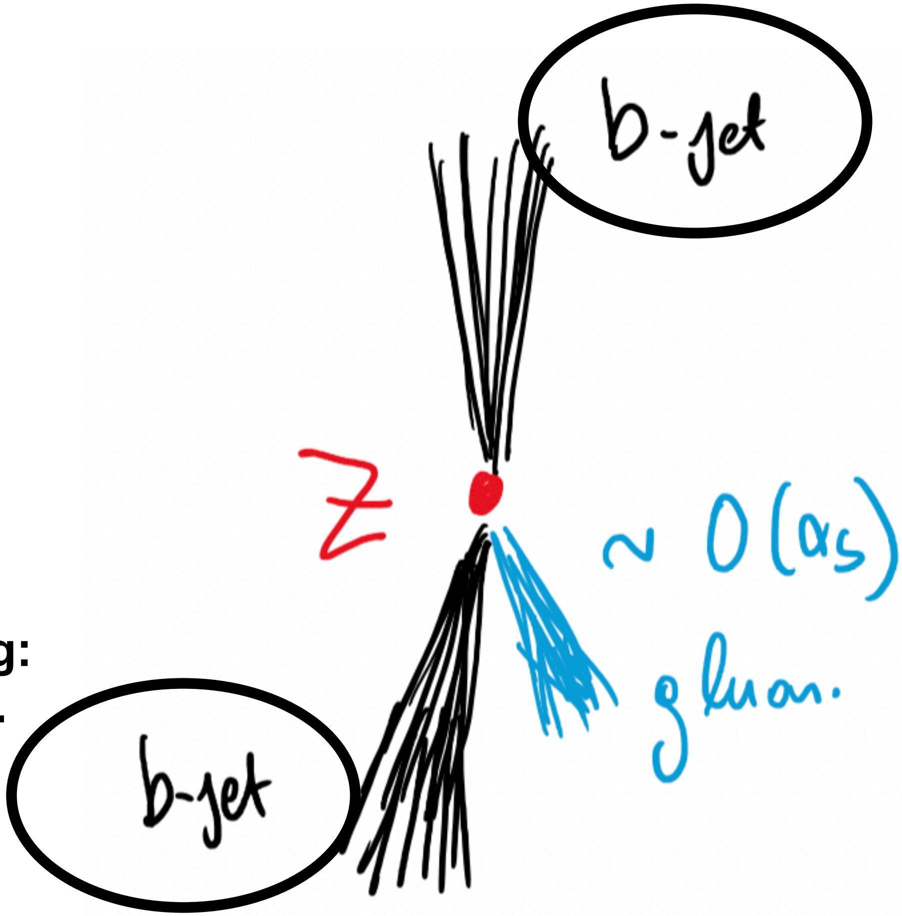
WP	Eff (s)	Mistag (g)	Mistag (ud)	Mistag (c)	Mistag (b)
Loose	90%	20%	40%	10%	1%
Medium	80%	9%	20%	6%	0.4%

- Back-of-the-envelope: Training sample @ Zpole
  - **bottom jets:** ~1x10^5 M, **strange jets:** ~8.8x10^4 M
    - all other jet flavors in between

Much larger training sample than what used for the MC-based training sample

Gluon tagging using data?

- Challenging… topic of discussion and brainstorming
  - For instance:



b-tagging: ~90% eff.

b-tagging: ~90% eff.

- **3rd jet is a gluon: O(1-10%) depending on momentum, angle**
- **Still more than enough for developing the gluon-category**

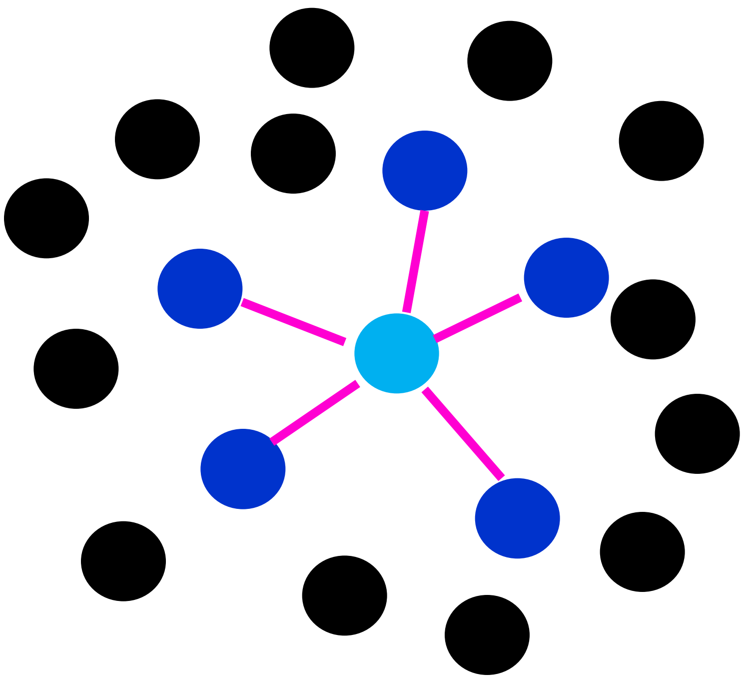
To be tested

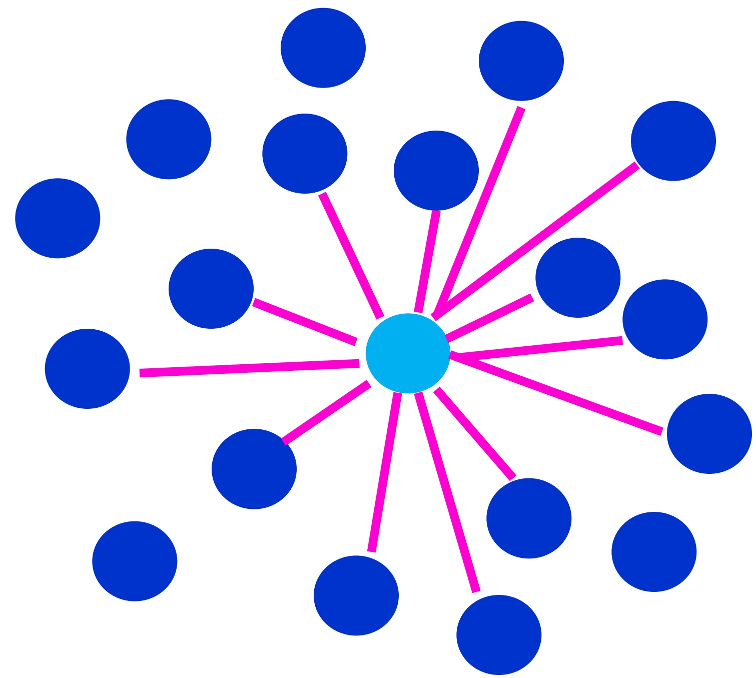
Pushing the limits further

based on:
H. Qu, C. Li, S. Qian
ICML 2022

ParticleNet-EE

ParticleTransformer

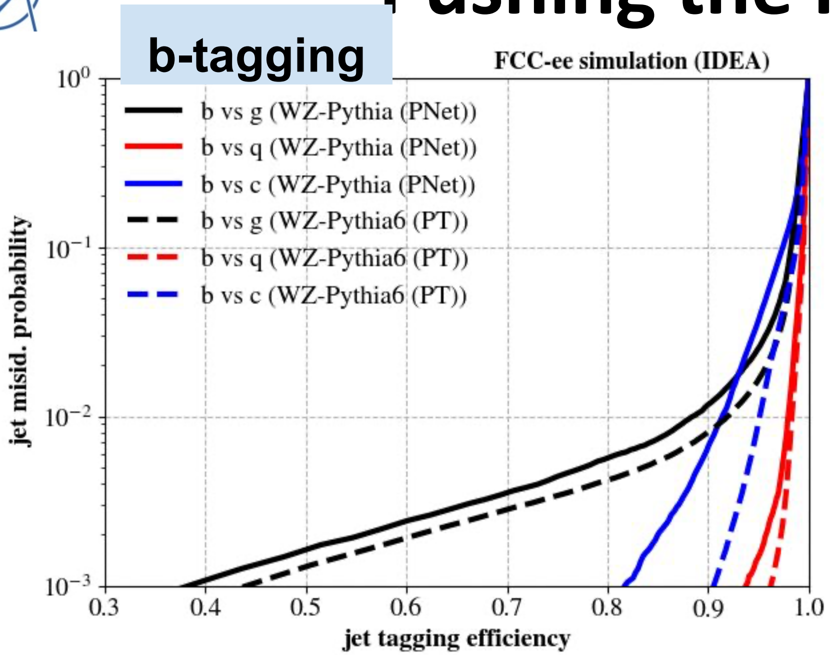


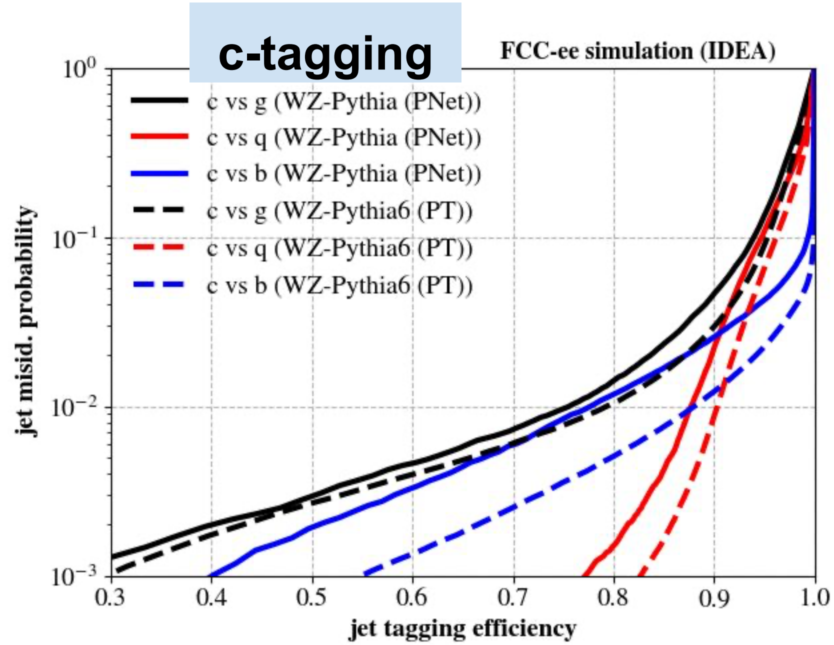


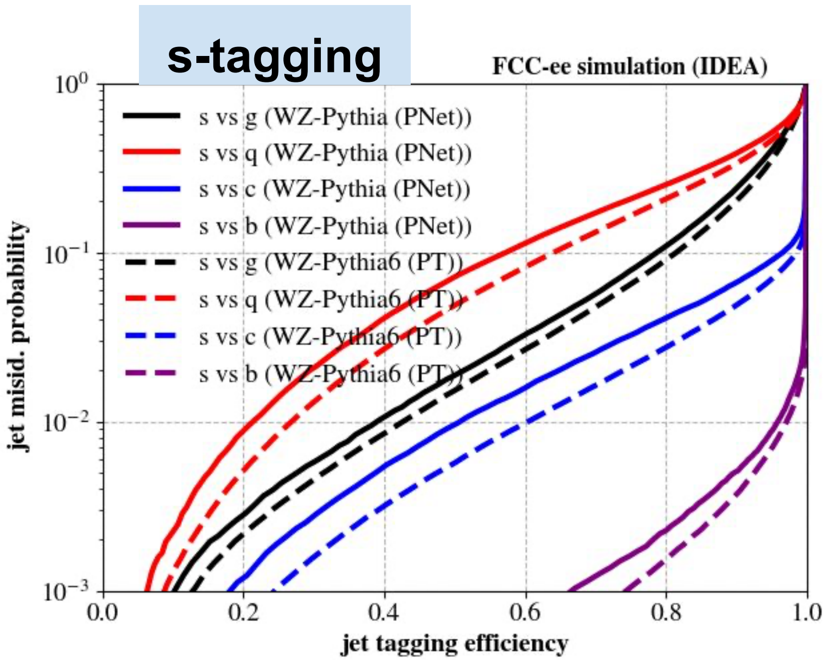
Use the *k*-nearest particles
[*k*=8 for ParticleNet-EE]

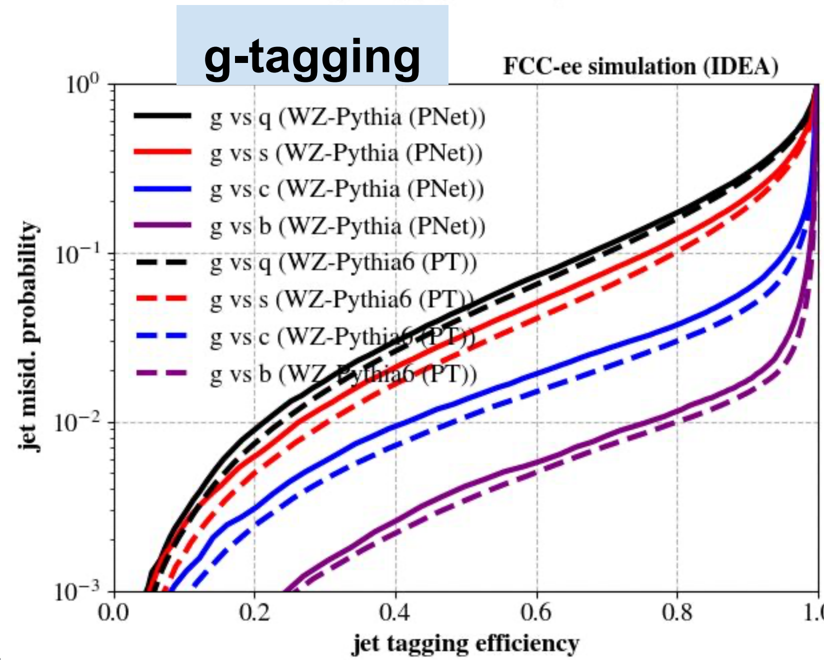
- **Fully connected graph**
- **Include per-particle-pair properties more directly**

Pushing the limits further (II)









Summary

- Powerful jet flavour identification important for the e^+e^- physics program

- Sophisticated jet tagging algorithms developed for e^+e^- experiments
  - Striking improvement in tagging performance compared to previous tools
    - allows us to explore more of the detector and event reconstruction potential
  - Fully integrated in FCCSW [data preparation, training, validation, inference, analysis] and explored in FCCee physics analyses [More tomorrow]
  - Still room for improvement / other ideas to try
    - Strong interest by the theory and experiment communities

- Why not testing them in actual e^+e^- data? → LEP
  - 5 M Z bosons / experiment → O(100K) training events / jet flavour
  - Great opportunity to commission these novel tools with real data
    - Bonus: Extract more physics out from LEP data [?]

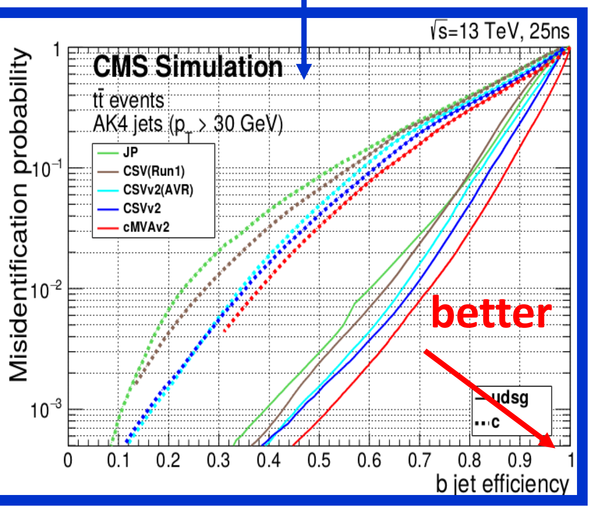
Review of a decade [CMS]

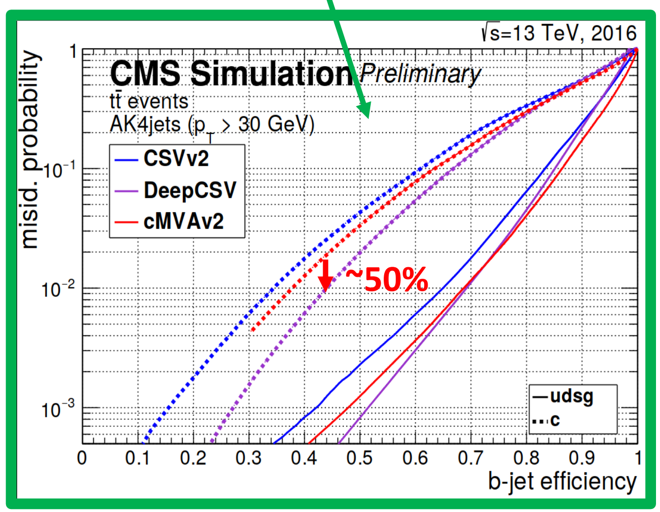
- Enormous progress over the last few years:

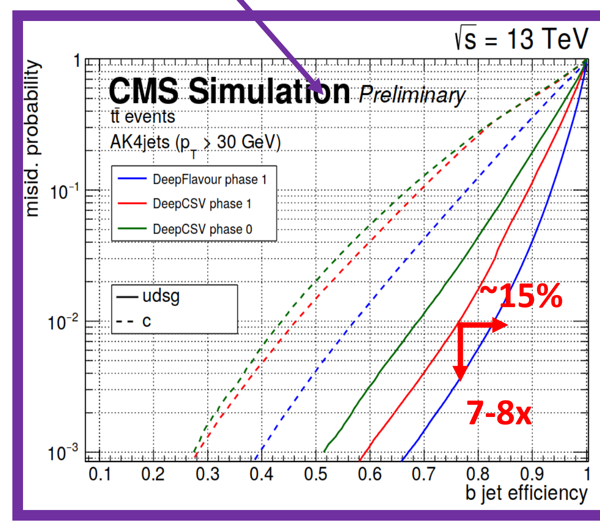
e.g., b-tagging

The early days: Human – inspired high-level variables + Cut-based selection	**Early Run 2:** Human – inspired high-level variables + Simple ML	**The "Game changer":** Inputs: low-level info [as particle sequences] + Advanced ML (CNN, RNN)	**Pushing limits further:** Inputs: low-level info [as unordered sets] + Graph Neural Nets

Run 1 **Run 2** **Run 3**





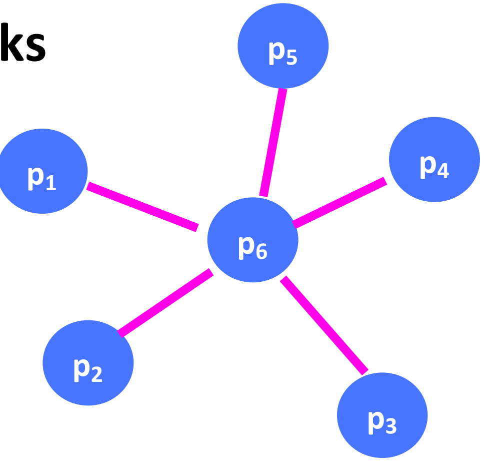


[Disclaimer: Focus on CMS results; similar methods developed by the other LHC experiments]

Designing a Graph-based tagger (II)

- Improve jet representation: "***Particle Sequences"*** → "***Particle Clouds"***
  - Treat the jet as an <u>unordered set of particles</u>
  - Rich set of information per particle
    - can be "viewed" as the coordinates of each particle in an abstract space

- Improved Network architecture: **Graph Neural Networks**
  - Particle cloud represented as a graph
    - Each particle: **vertex** of the graph
    - Connections between particles: the **edges**



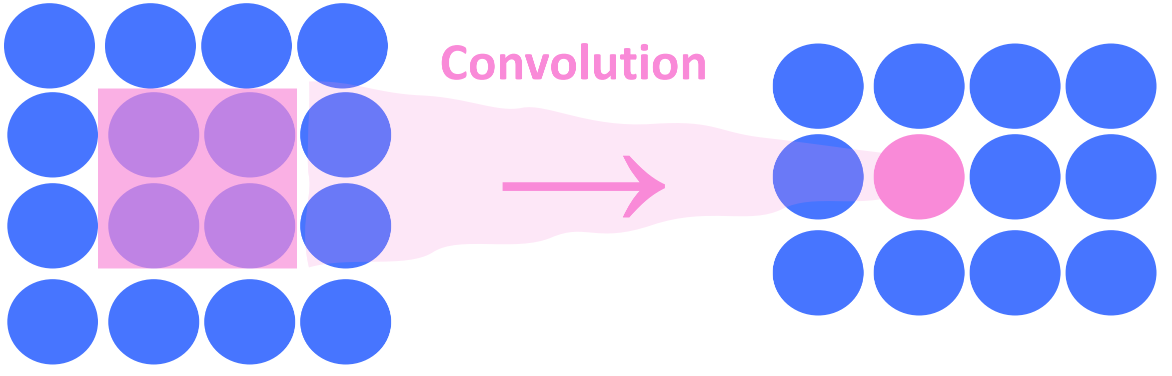
- **Build** the graph:
  - One approach: Fully connected Graph [but computationally very expensive]
  - Another possibility: apply some criteria
    - e.g., *k*-Nearest Neighbors (*k*NN)

Designing a Graph-based tagger (III)

- Last step: **Learn** from the graphs
  - Follow a **hierarchical learning** approach:
    First learn <u>local</u> structures and **then more <u>global</u>** ones

- Convolution operations proven to be very powerful

Fixed grid:

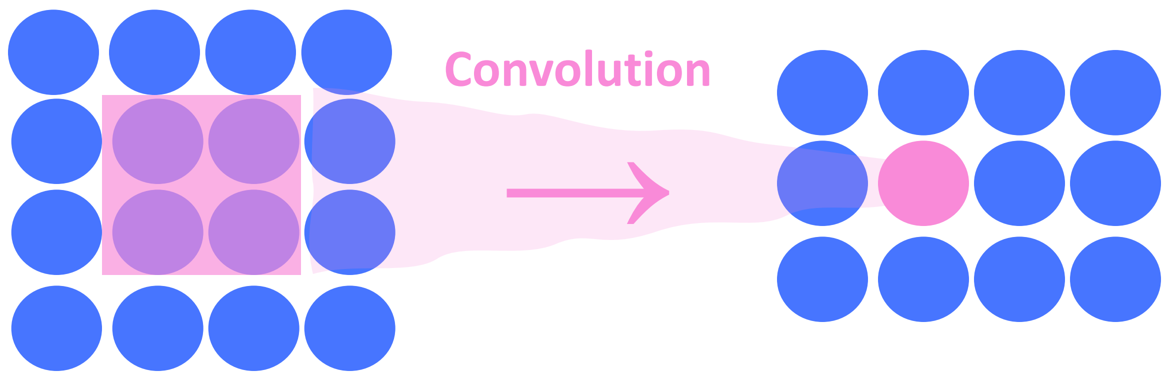


Designing a Graph-based tagger (IV)

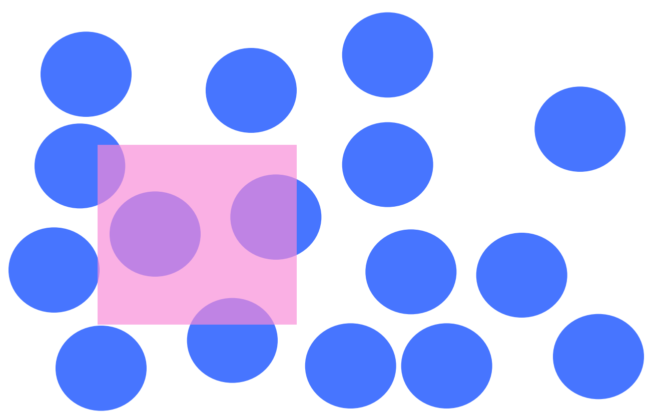
- Last step: **Learn** from the graphs
  - Follow a **hierarchical learning** approach:
    First learn <u>local</u> structures and **then more <u>global</u>** ones

- Convolution operations proven to be very powerful

Fixed grid:



point/particle cloud:

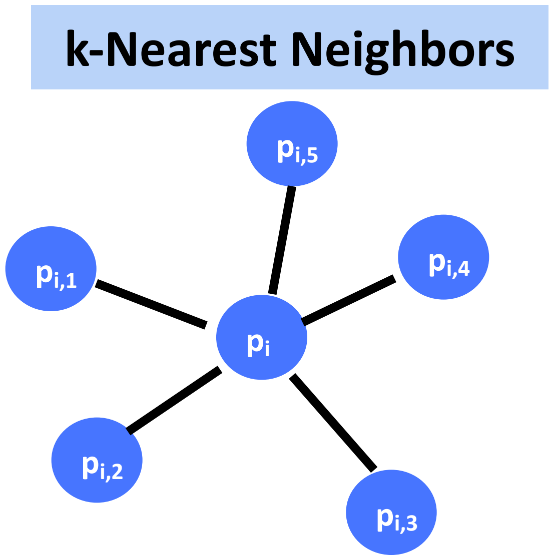


… but not straightforward on point/particle clouds
- Irregular and unordered sets
- Requires a permutation invariant convolutional operation

EdgeConv: Convolution on point clouds

Y. Wang et al.

- Find the ***k*-nearest neighbors** of each point

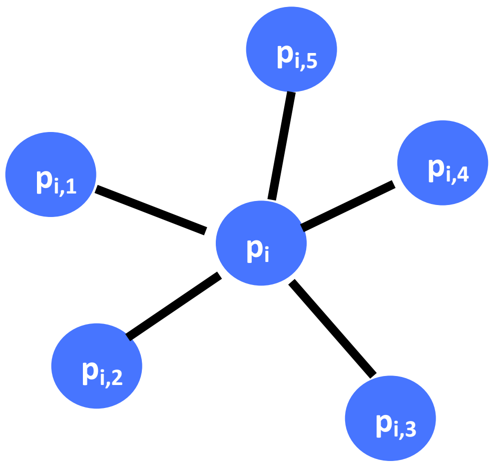


EdgeConv: Convolution on point clouds

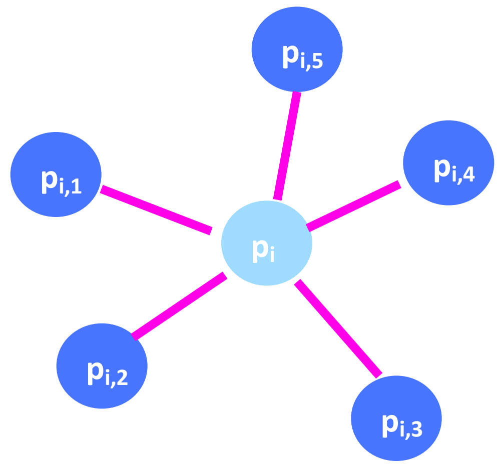
Y. Wang et al.

- Find the ***k*-nearest neighbors** of each point
- Design a permutation invariant **convolution operation**
  - Define an **edge feature** function → **aggregate** edge features w/ a symmetric func.

k-Nearest Neighbors



Convolution operation



- In a nutshell:

$$p_i' = \mathop{\square}_{j=1}^{k} h_\Theta(p_i, p_{ij} - p_i)$$

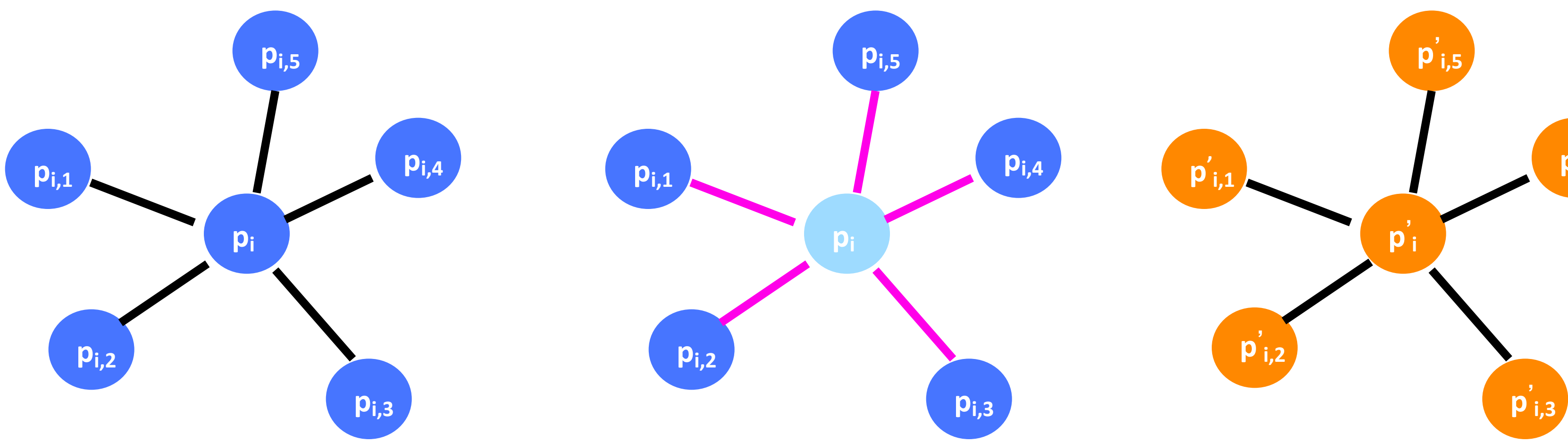
ParticleNet:
h_Θ: MLP [shared across edges]
$\square$ **: average over all *k*-NN**

EdgeConv: Convolution on point clouds

Y. Wang et al.

- Find the ***k*-nearest neighbors** of each point
- Design a permutation invariant **convolution operation**
  - Define an **edge feature** function → **aggregate** edge features w/ a symmetric func.
- **Update Graph (ie Dynamic Graph CNN, DGCNN):** Using *k*NN in the feature space produced after EdgeConv
  - Can be viewed as a mapping from one particle cloud to another

k-Nearest Neighbors | **Convolution operation** | **Update Graph**



- In a nutshell:

$$p_i' = \square_{j=1}^{k} h_\Theta(p_i, p_{ij} - p_i)$$

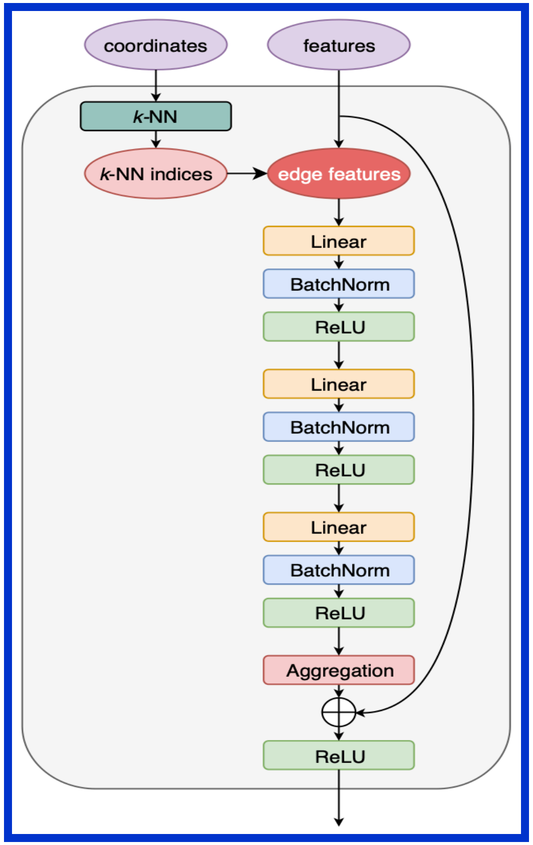
ParticleNet:
h_Θ: MLP [shared across edges]
$\square$ **: average over all *k*-NN**

ParticleNet for jet tagging

H. Qu and LG
PRD 101 056019 (2020)

- Based on EdgeConv and DGCNN
  - but customized for the jet tagging task

EdgeConv block



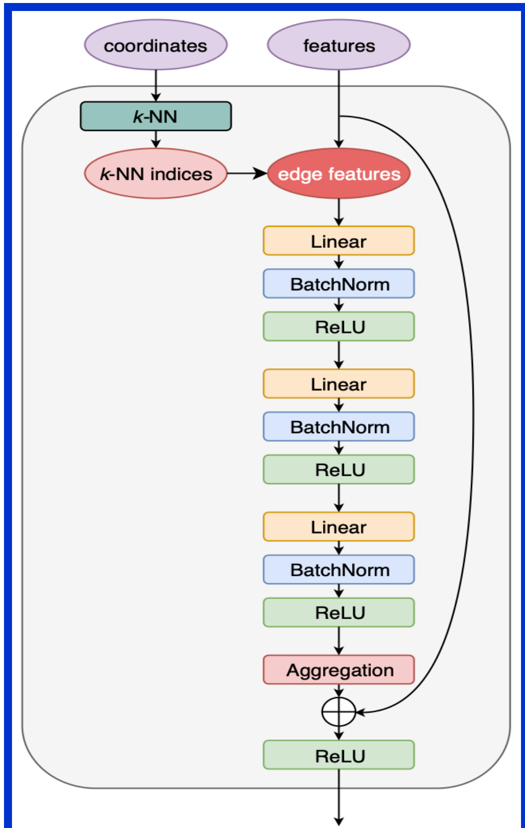
Introduced:
- features beyond spatial coordinates
- residual connections
- MLP conf.

ParticleNet for jet tagging (II)

H. Qu and LG
PRD 101 056019 (2020)

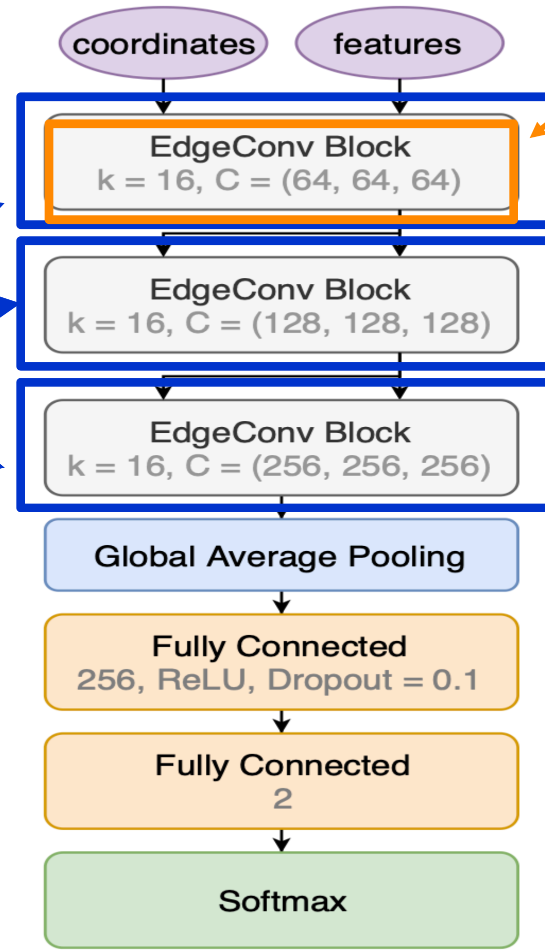
- Based on EdgeConv and DGCNN
  - but customized for the jet tagging task

EdgeConv block



Introduced:
- features beyond spatial coordinates
- residual connections
- MLP conf.

ParticleNet Architecture

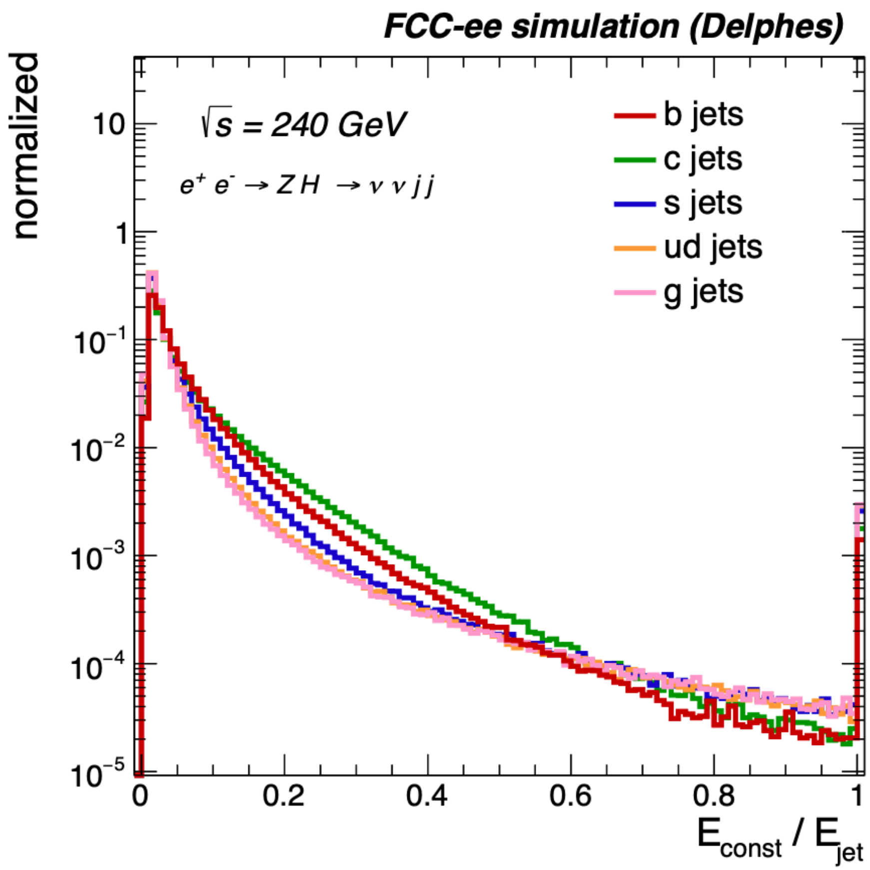


particles distributed in η-ϕ

From **local** to more **global** structures

Example of input features

Constituent relative energy



Impact parameter (d_0)