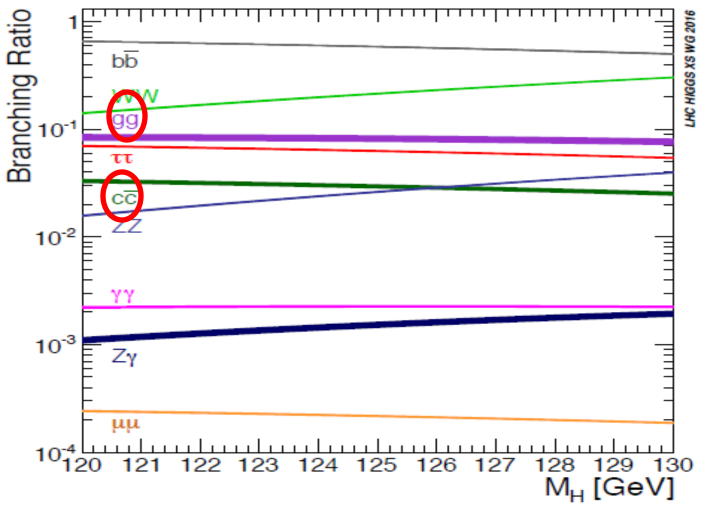
Jet flavor tagging for FCCee

Loukas Gouskos (CERN) and Michele Selvaggi (CERN)

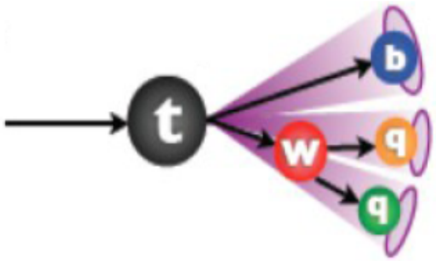
FCCee Physics performance meeting
Mon, Sep 21, 2020

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]



Scope of this talk:

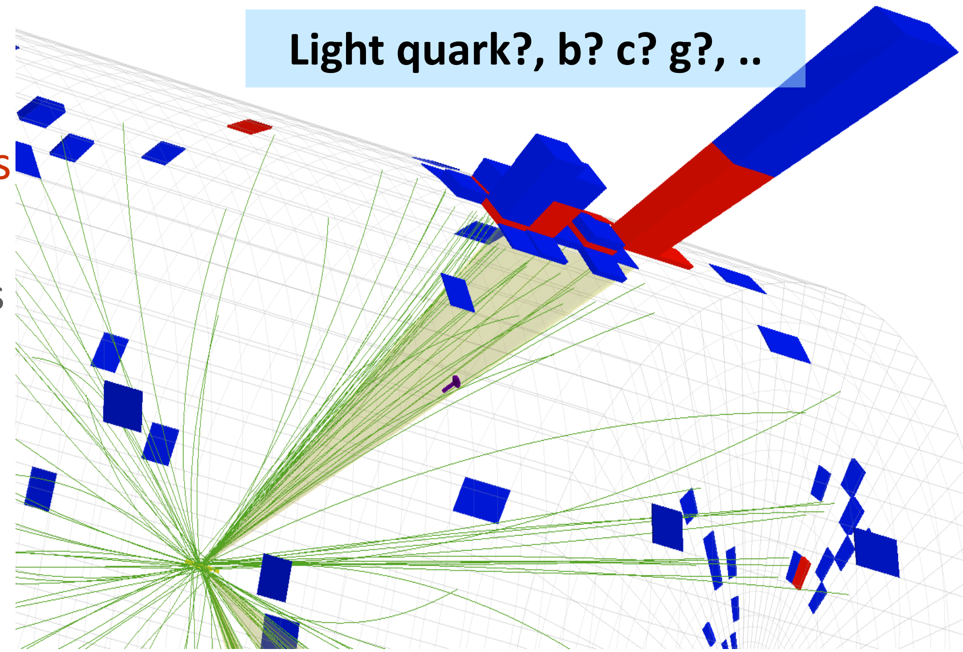
Present the first implementation of a jet flavor tagging algorithm for FCCee

By all means: not a final version; rather a demonstration of the full setup [e.g. Sample production, algorithm design, ..]

Introduction

- **A jet in theory:** Spray of particles produced by the hadronization of quarks and gluons
- **A jet experimentally:** A cone of reconstructed particles in the detector

- **Experience at LHC:**
  - particle-based jet tagging allows to explore much more of the detector's potentials
    - yielding the most powerful results [by far..]



- Precise PF event reconstruction critical for powerful jet tagging
  - Rich information for each particle
    - energy/momentum/position..
    - displacement from PV, particle type, track quality,..
  - [O(30) properties/particle] x [~30-40 particles/jet] ~O(1000) inputs/jet
    - **Perfect case** for DNN with "complex" architecture

Designing a jet tagging algorithm

- How to represent a jet is one of the key aspects of algorithms for jet physics
  - Improve performance $\rightarrow$ extend physics reach
  - Lead to fresh insight into jets $\rightarrow$ deepen our understanding of jet physics

- Particles [associated to each jet] are intrinsically unordered
  - i.e., ordering by p_T(particle) or displacement from PV: suboptimal
  - primary information: 2D coordinates in theta-phi space (or eta-phi at the LHC)
  - Additional features: energy, displacement, charge, track quality…

Jet tagging: From point clouds

Point cloud

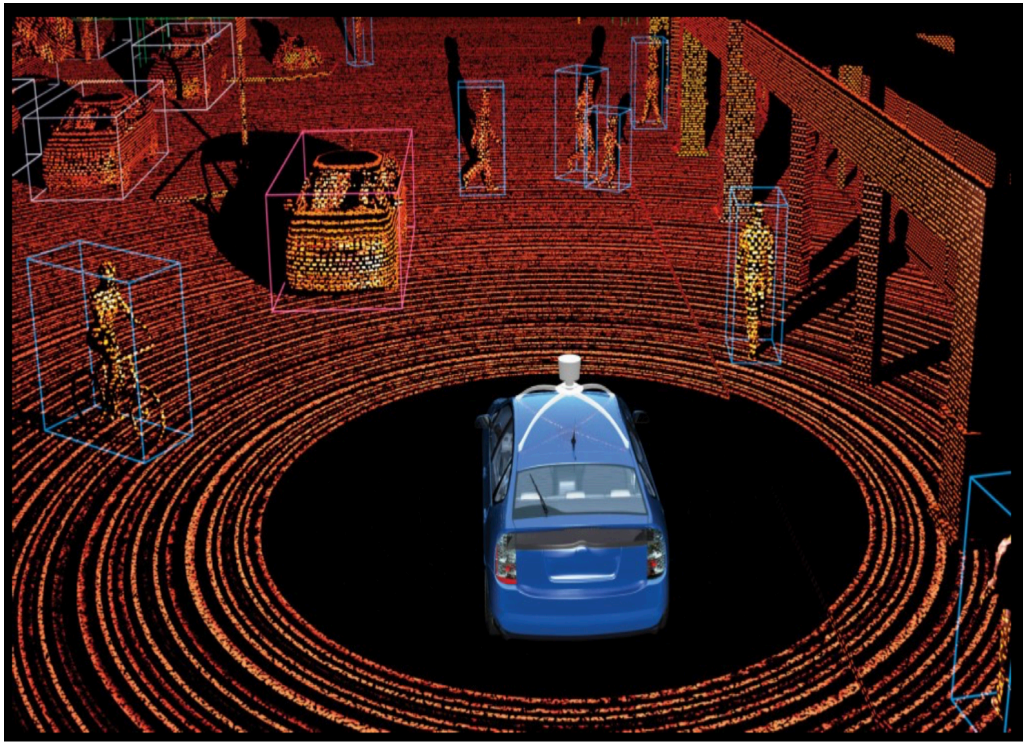


Image from:
https://news.voyage.auto/an-introduction-to-lidar-the-key-self-driving-car-sensor-a7e405590cff

- Point cloud (Wikipedia):
  - a set of data points in space
  - produced by 3D scanners, which measure a large number of points on the external surfaces of objects around them

Jet tagging: From point clouds to particle clouds

Point cloud

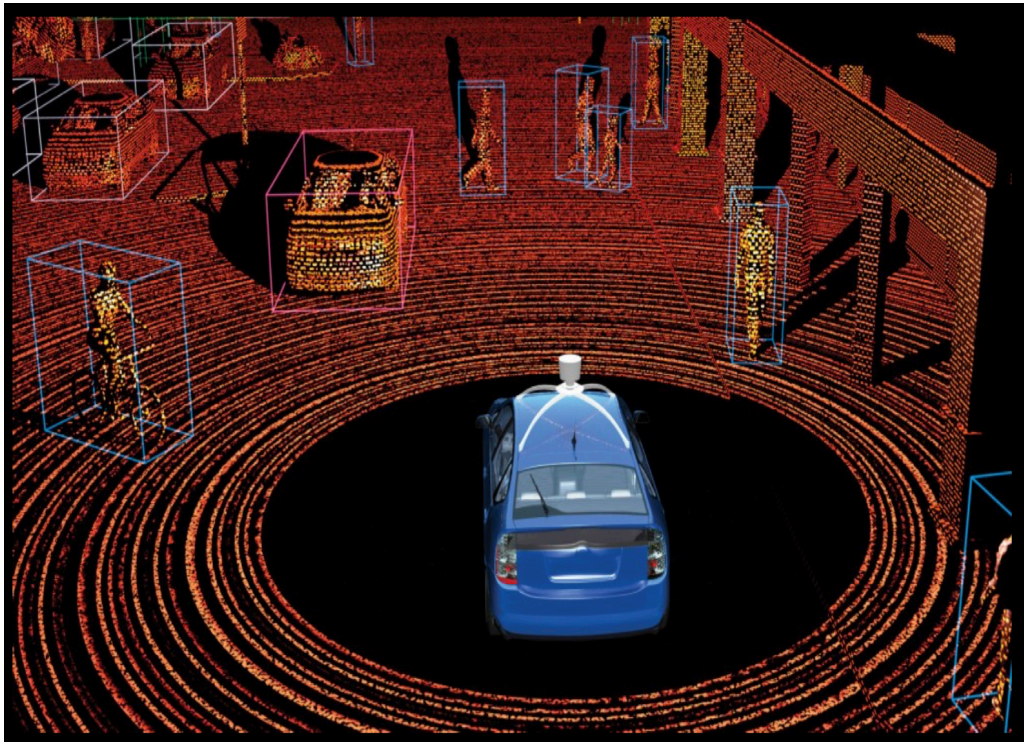
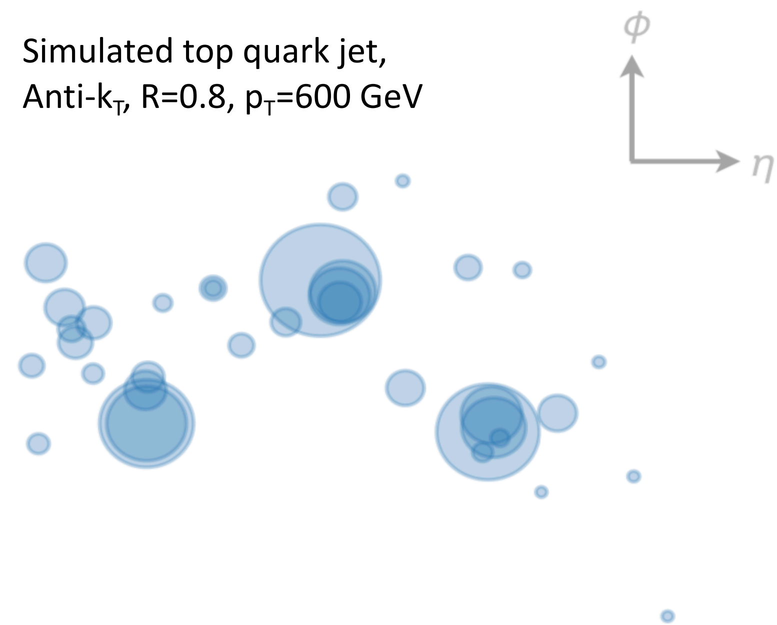


Image from:
https://news.voyage.auto/an-introduction-to-lidar-the-key-self-driving-car-sensor-a7e405590cff

Particle cloud



- Point cloud (Wikipedia):
  - a set of data points in space
  - produced by 3D scanners, which measure a large number of points on the external surfaces of objects around them
- Particle cloud (Wikipedia):
  - a set of particles points in space
  - produced by clustering a large number of particles measured by the detectors

ParticleNet for jet tagging

ParticleNet [Huilin Qu, LG]

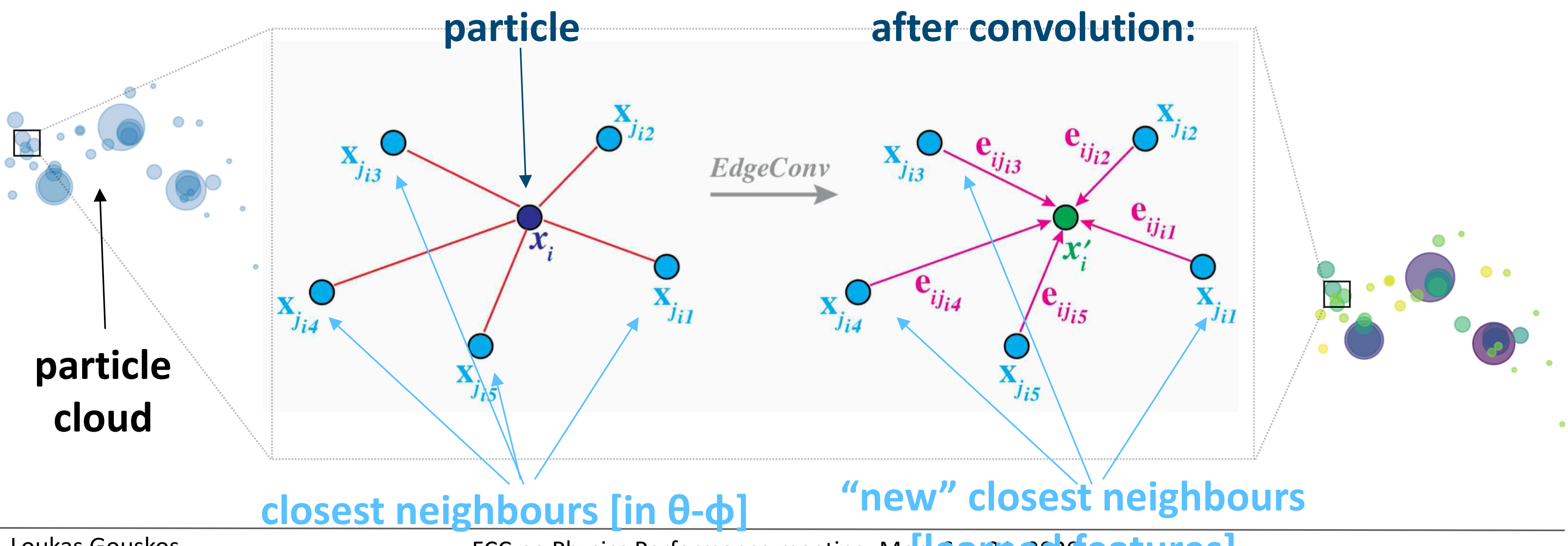
PRD 101 (2020) 5, 056019
CMS-DP-2020-002

- Jet representation:
  - Inspired by “point clouds” we developed “ParticleNet”: unordered set of particles
    - Particles organized using the theta-phi coordinates
    - Then for each particle: include information (features) related to energy,. displacement, track info, track type .. [full list in the back-ups]

- Use state-of-the-art ML techniques:
  - Treat the particle cloud as a graph
    - each particle is a vertex, connections between particles are the edges
  - Follow a hierarchical learning approach: from ”local” info to more “global”
    - i.e. convolution operation
    - add references

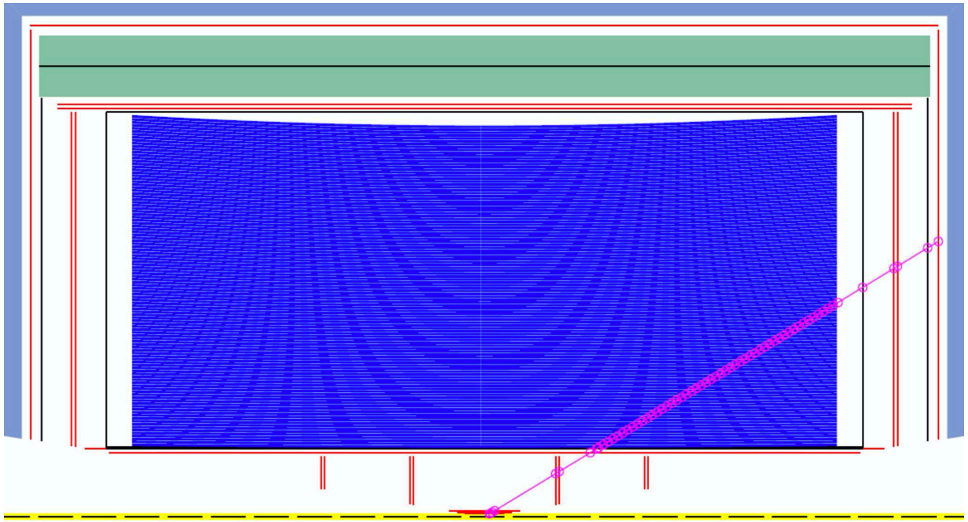
ParticleNet for jet tagging (II)

- In a nutshell:
  - For each point of the graph find the k-nearest neighbors [distance defined based on the particle “coordinates”]
    - **Initially:** closest neighbors are defined based on theta-phi
    - **Then:** after each convolution step explore additional particle features [energy, displacement, charge ..] and output a set of “learned” features
      - these features can be interpreted as coordinates in a high-dimensional space
  - Dynamically update the Graph: Recompute the “distances” between the particles after each convolution step [layer]: update the k-nearest neighbors



Sample/generation details

- Samples generated using Delphes for the detector response
  - including FastTrackCovariance [from Franco Bedeschi]
    - Standalone C++ implementation
    - TRK implementation including material
    - Computes the full covariance matrix
    - Included multiple scattering
    - computes smeared track using the off-diagonal terms

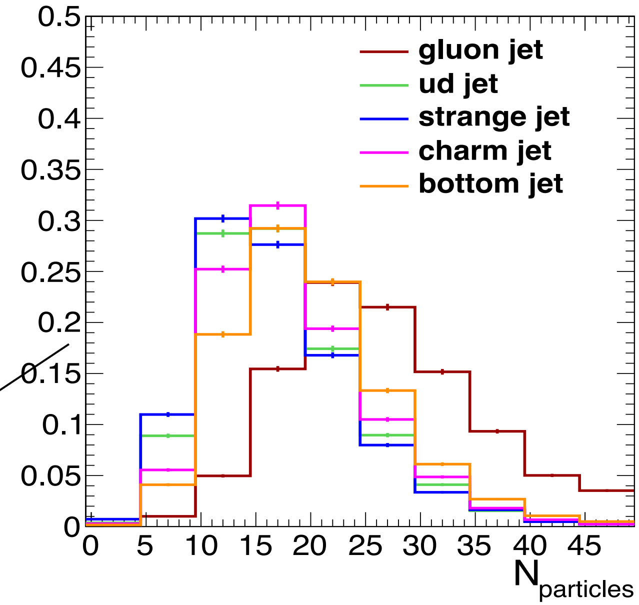


- MG5+Pythia8 used to generate ZH->vvXX events where X:g, ud, s, c, b

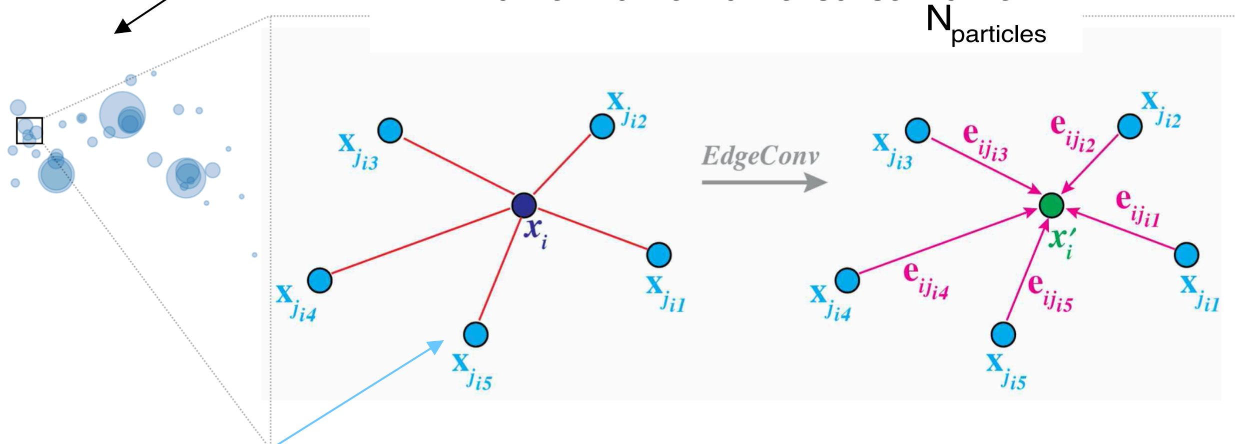
- Jets clustered with the generalized-k_T algorithm [similar to anti-kT] with R=1.5
  - IRC safe, etc..

ParticleNet for jet tagging at FCCee

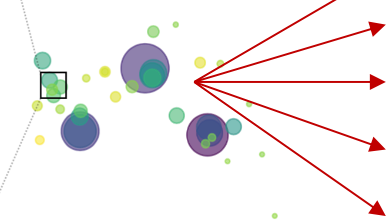
Inputs:
30 particles/jet



Training details:
– ~10K jets / class
– equal distributed across classes
[obviously a very small sample; used for setting up the machinery]



Output:



g
[ud]
[s]
c
b

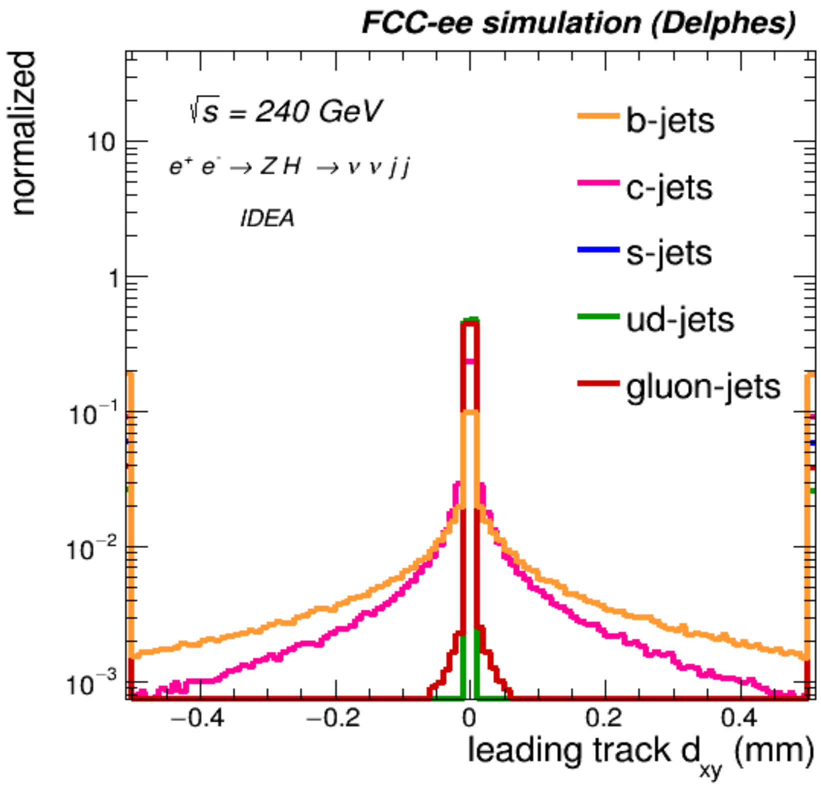
Particle features: 21 /particle

particle kinematics, charge,
impact parameter [xy, z] and significance,
particle type [el, mu, gamma..],..

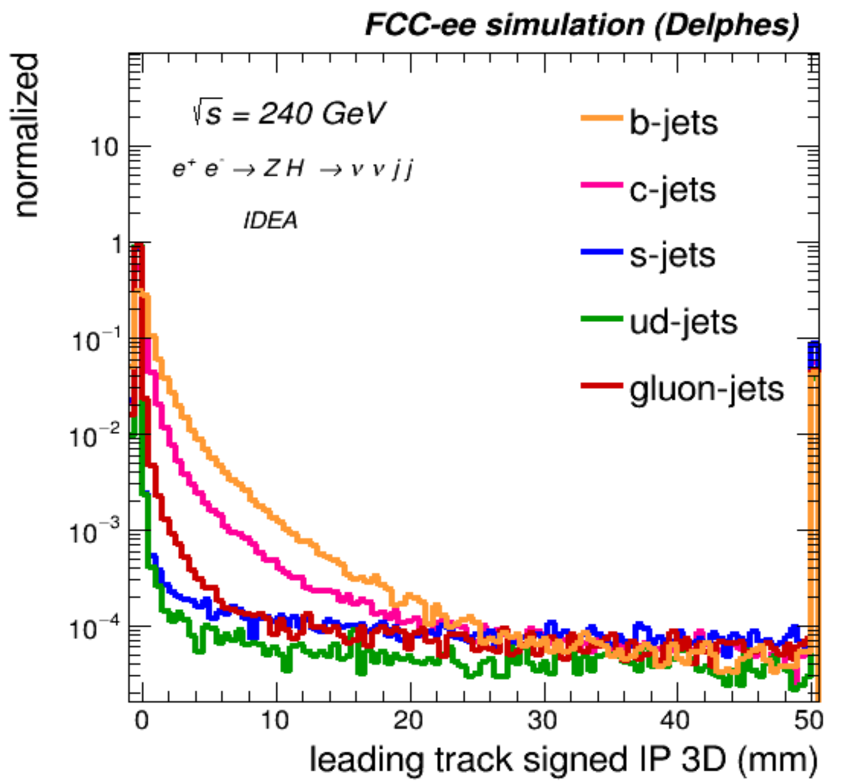
Input variables

- For the leading [in energy] particle:

dxy impact parameter



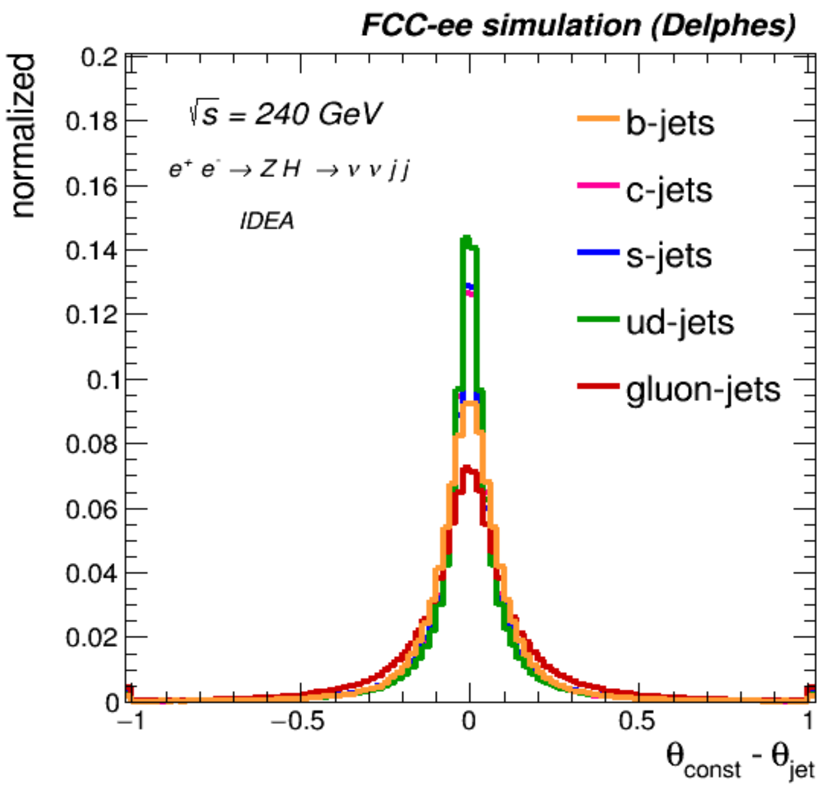
3D signed IP; in the jet direction



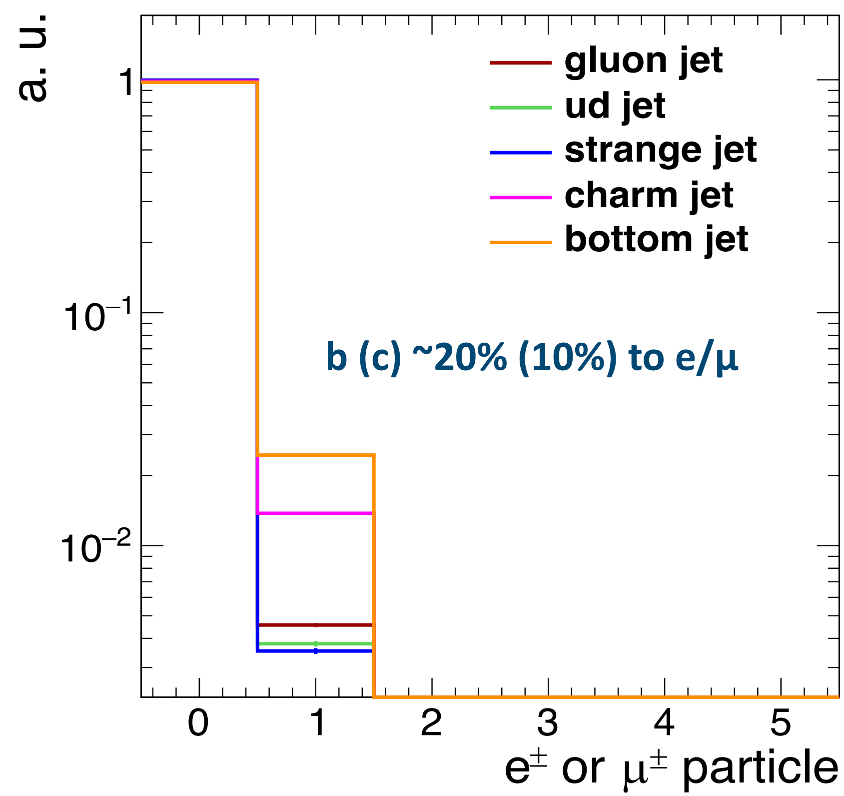
- More information: jet types comparison & between detectors [IDEA vs. CLD]:
  - https://selvaggi.web.cern.ch/selvaggi/FCC/FCCee/FlavourTagging/

Input variables (II)

spatial distribution



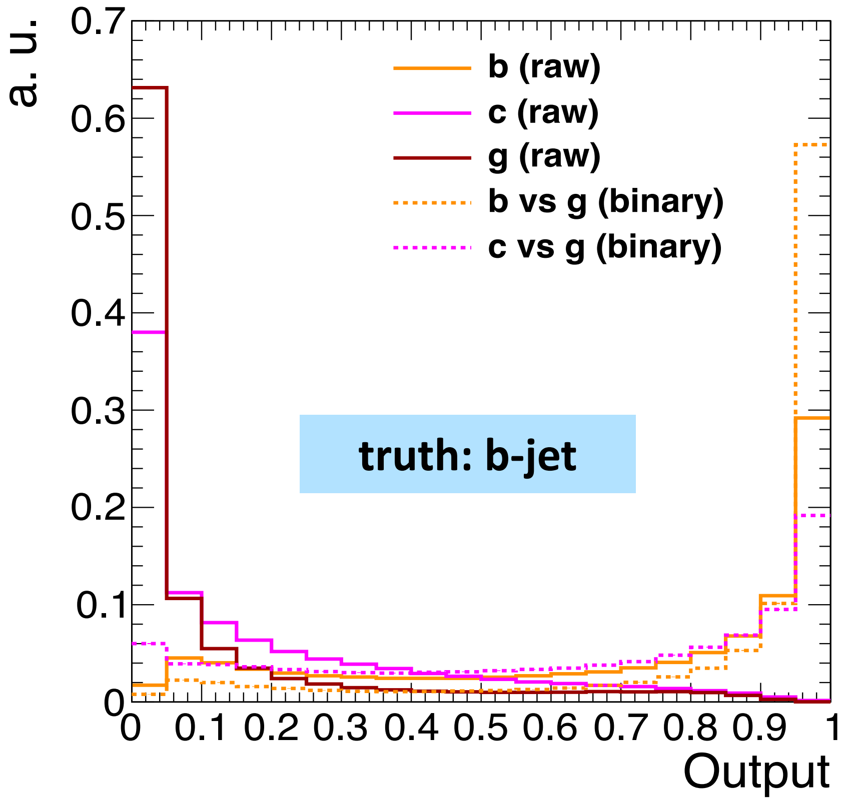
Particle type

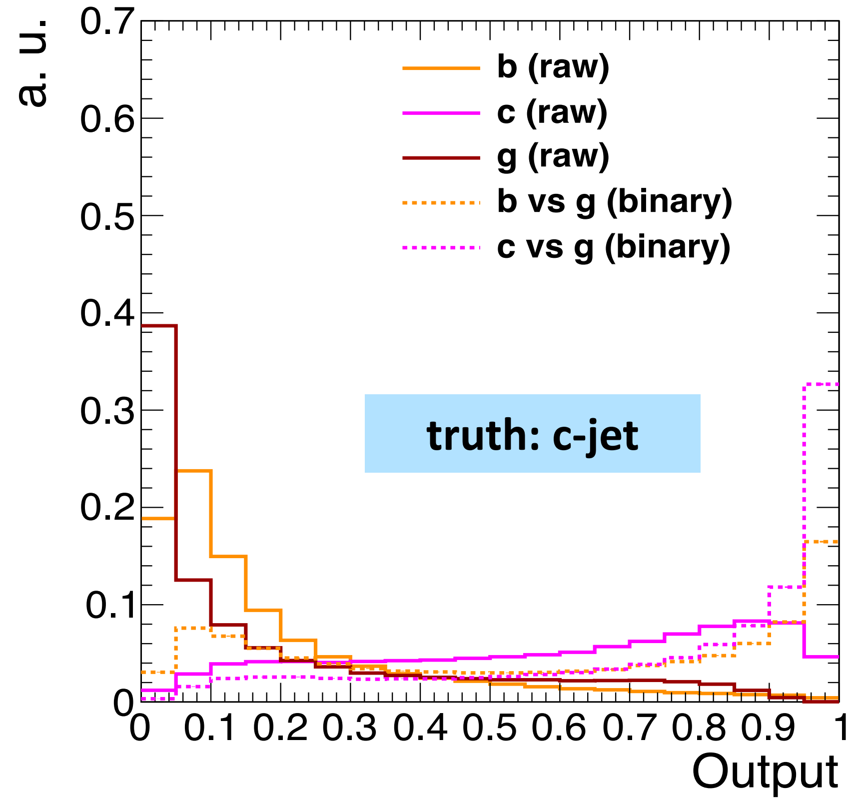


- More information: jet types comparison & between detectors [IDEA vs. CLD]:
  - https://selvaggi.web.cern.ch/selvaggi/FCC/FCCee/FlavourTagging/

Performance [IDEA detector]

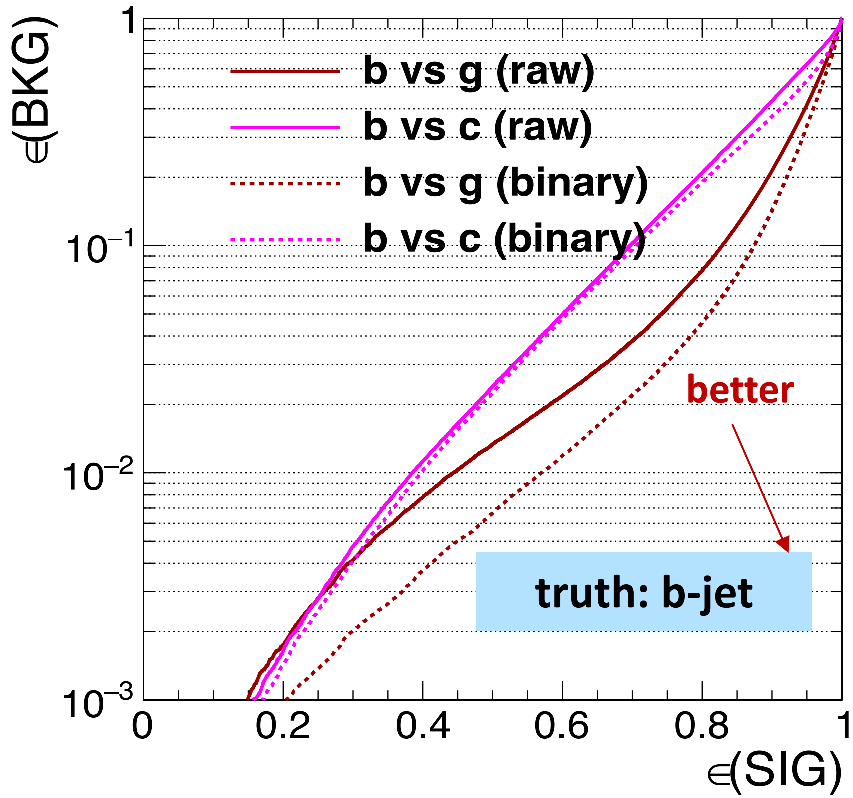
- Output scores:

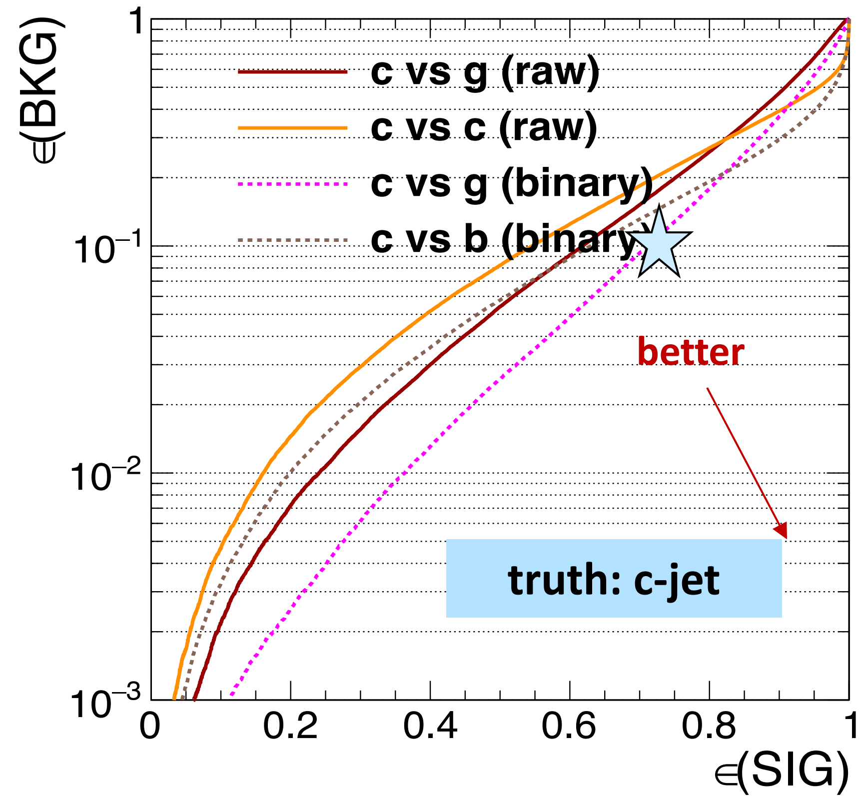




- **binary**: separation between two particle species
- **raw**: separation among of particle species used for the training

Performance: ROC curves

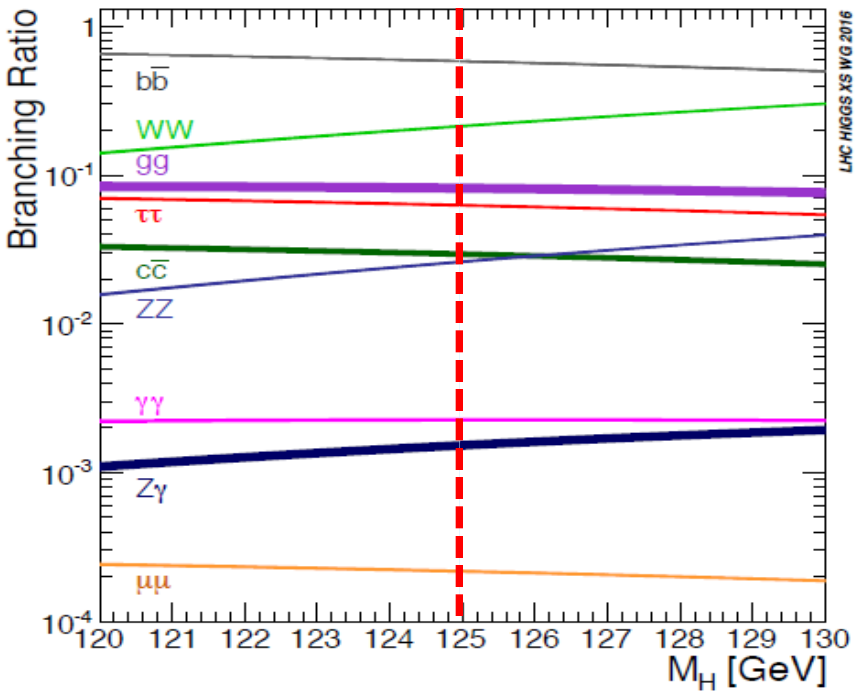




***e.g.: ~70% tagging eff for ~10% b-mistag**

- Performance already pretty competitive: particularly for charm–tagging
  - NB. highly suboptimal tuning of the algorithm

jet tagging in practice



$\sqrt{s}$ (GeV)	240		365	
Luminosity (ab^{-1})	5		1.5	
$\delta(\sigma BR)/\sigma BR$ (%)	HZ	$\nu\bar{\nu}$ H	HZ	$\nu\bar{\nu}$ H
$H \to$ any	±0.5		±0.9	
$H \to b\bar{b}$	±0.3	±3.1	±0.5	±0.9
$H \to c\bar{c}$	±2.2		±6.5	±10
$H \to gg$	±1.9		±3.5	±4.5
$H \to W^+W^-$	±1.2		±2.6	±3.0
$H \to ZZ$	±4.4		±12	±10
$H \to \tau\tau$	±0.9		±1.8	±8
$H \to \gamma\gamma$	±9.0		±18	±22
$H \to \mu^+\mu^-$	±19		±40	
$H \to$ invis.	< 0.3		< 0.6	

Back of the envelop:

– FCCee: σ_{ZH}~200 fb, L~5 ab^{-1} (2IP)~**1M ZH**
[600k H→bb, 100k H→gg, 30k H→cc]

– **Scenario:**
c-tag: 70%, b-mistag: 10%, g-mistag:~10%
δ(σBR)/σBR (%) ~ 1.5 [no systematics]

– **Improving b/g rejection by a factor of 2** would result to ~1% uncertainty

Results look promising

Ref: Patrick's talk at the CDR Symposium; March 2019

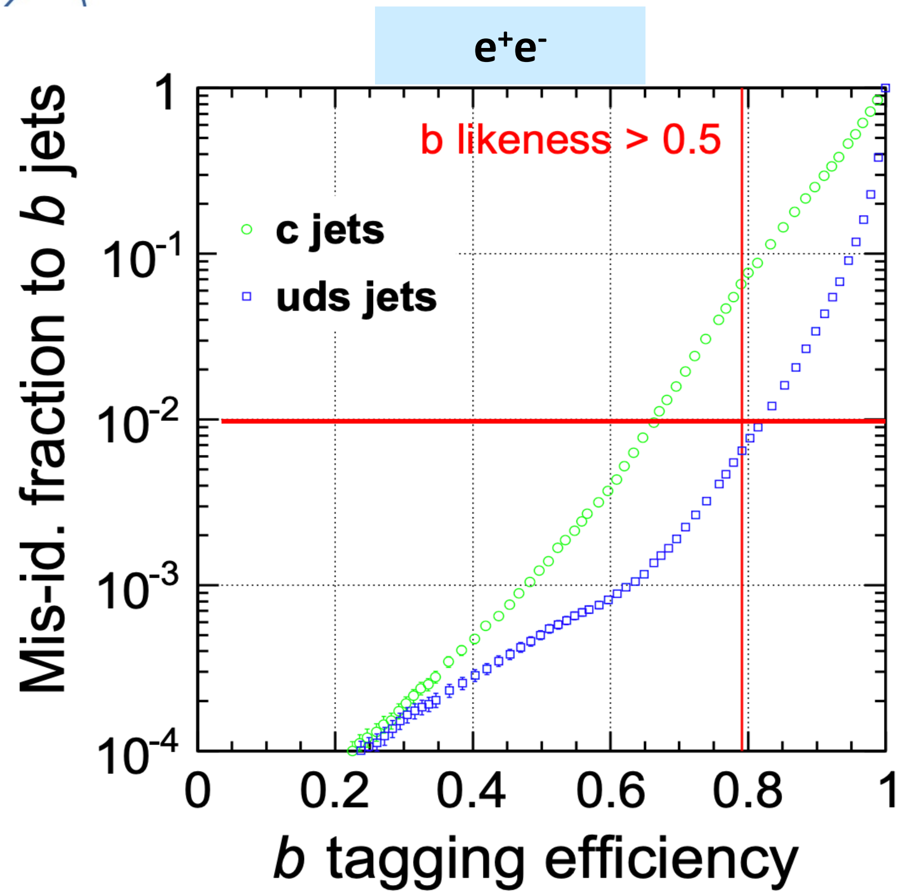
Summary & outlook

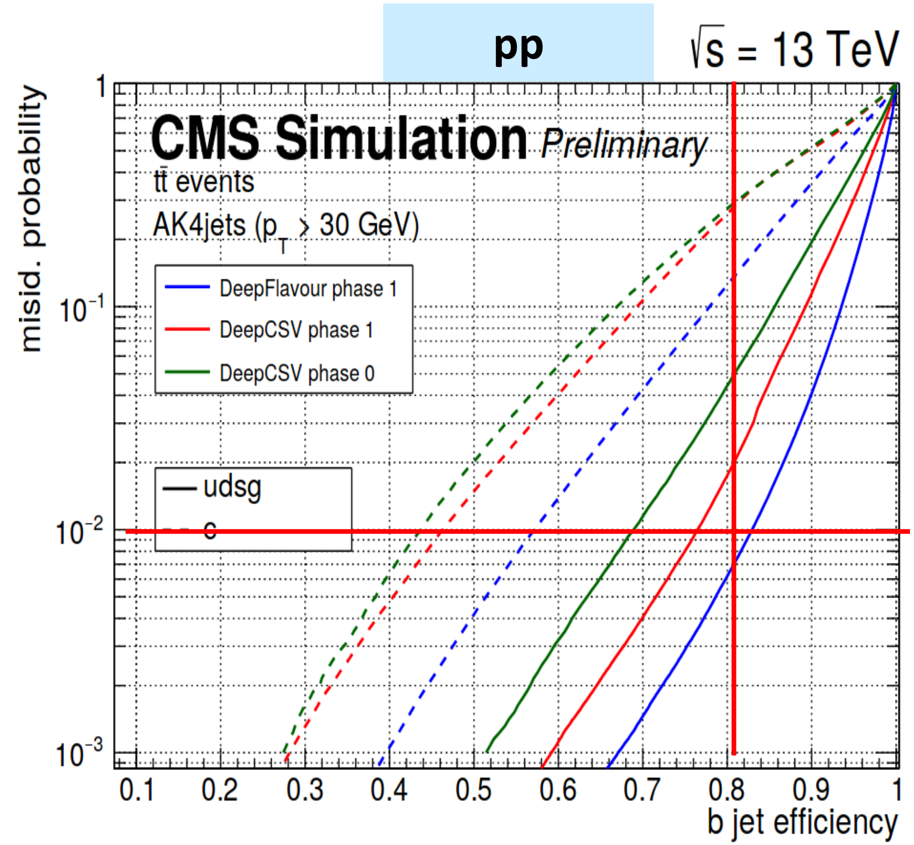
- Current status:
  - The entire chain from designing the jet tagging algorithm, production of input samples and definition of metrics in place
    - final touches to the network and we will provide a recipe
      - based on “weaver”: https://github.com/hqucms/weaver
  - First very preliminary results look promising

- Planned improvements/goals:
  - Include additional jet types: ud, s, [together with g, c and b]
  - Additional input variables [e.g., full covariance matrix, secondary vertex information] + jet clustering algorithms
  - Optimize network parameters
  - Bonus: Estimate calibration uncertainties using Z bosons

- **Ultimate goal:** Compare performance for different detector configurations **and find the best Tracker design**

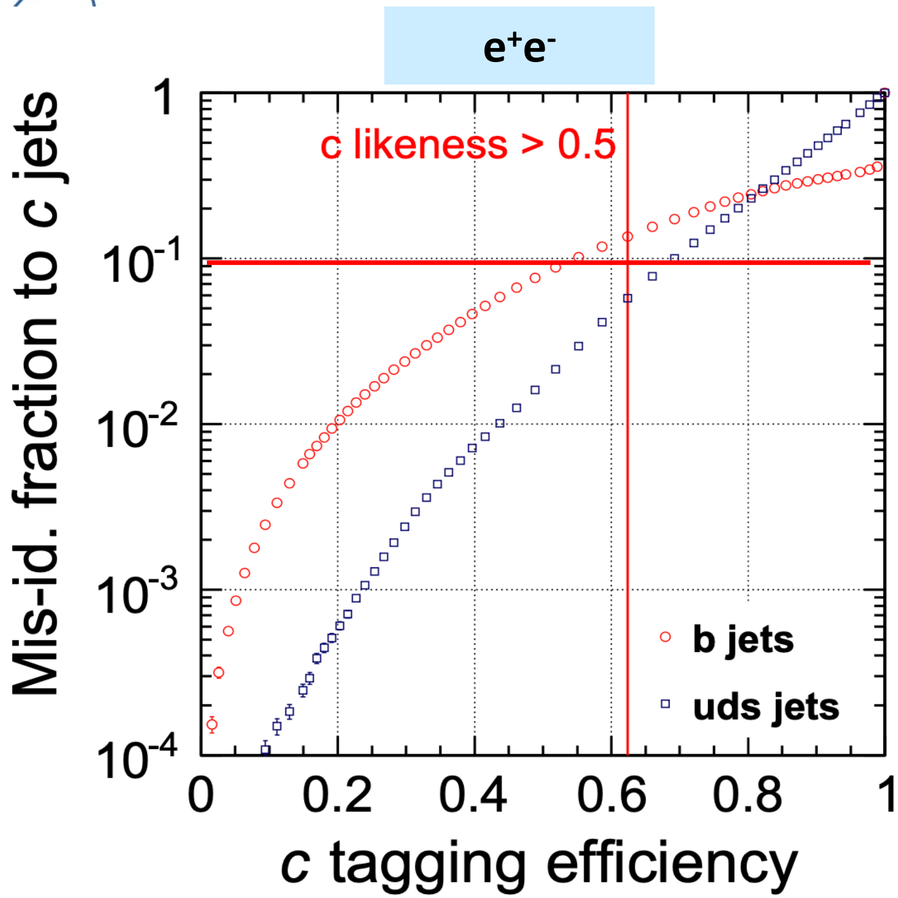
b-tagging performance

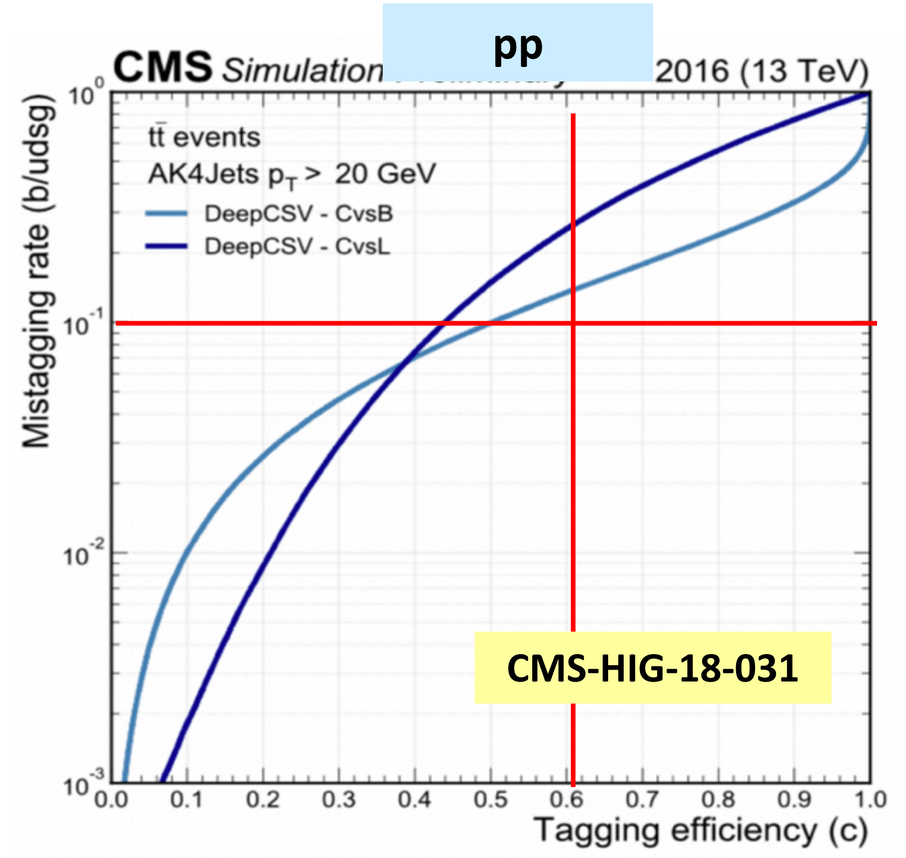




- Similar performance [take it with a lot of grain of salt]
  - yet conditions and detector potential very different [favoring the e^+e^- case]
- Definitely worth exloring the recent developments in pp colliders
  - (a) Improve performance and/or achieve necessary performance with less complex (cheaper) detector solutions

c-tagging performance





- **Charm-bottom separation:** similar performance
- **Charm-light separation:** e^+e^- shows better performance, but for the pp case:
  - results derived <u>before</u> the upgraded PIXEL detector
  - Algorithm does not explore the latest tagging developments
    - i.e. low-level features and advanced ML architecture