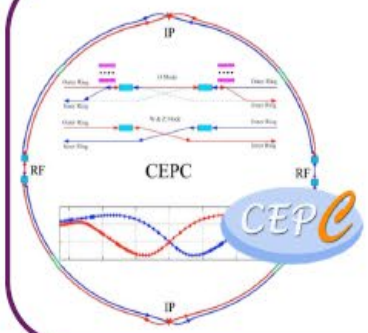
Particle Transformer for Quark Flavor Tagging at Higgs factories

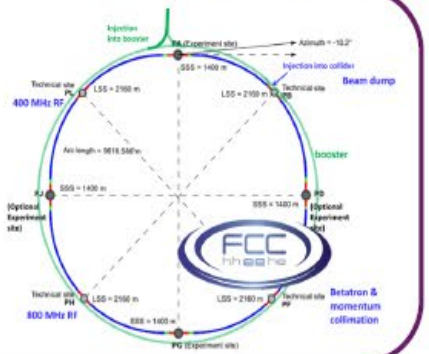
Risako Tagami, Taikan Suehara (ICEPP, U. Tokyo)
Gui Lai (Imperial College London, 2023 summer student)

Higgs factories and detectors

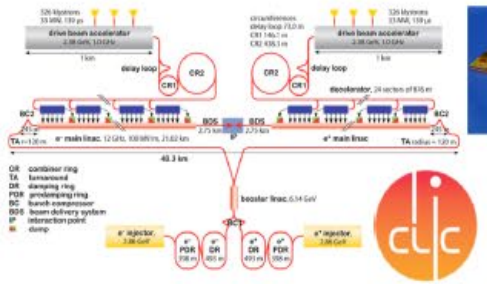
e+e- Colliders

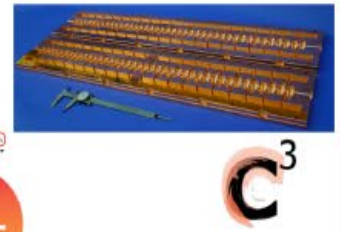
circular



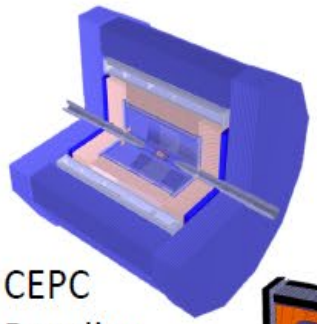


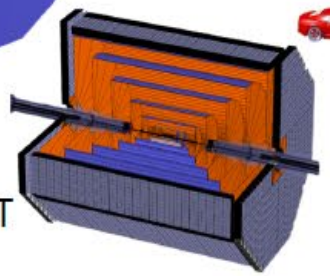
Linear

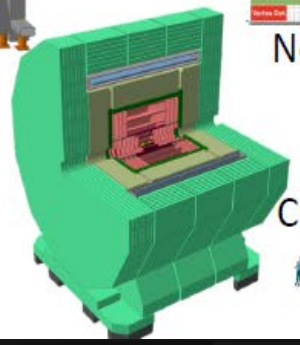


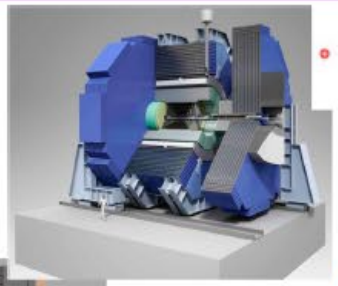


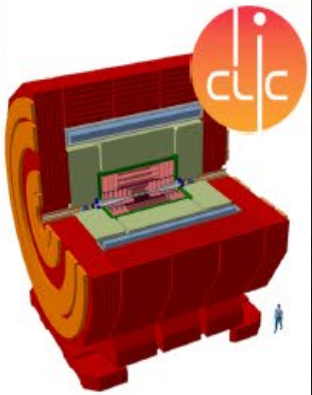
Detector Concepts











M.C Fouz

e^+e^- collider projects

- Linear colliders: upgradable to TeV collider
  - ILC (Japan) 250 GeV (initial) → multi-TeV **ILC Technology Network is active**
    Superconducting LC to be started in end of 2030s. The most mature project.
  - CLIC (CERN) 380 GeV → 3 TeV
    Normal conducting (X-band) LC. The alternative option to FCC in EPPSU. Affordable for CERN.
  - CCC (US) 250 GeV → multi-TeV
    Cooled normal conducting (C-band) LC. Currently at Pre-CDR. Realization in > 2040.
  - HELEN (US)
    Superconducting LC. High gradient realized by traveling wave cavities. Still rough design stage.
- Circular colliders: replacement to hadron collider foreseen
  - FCCee (CERN) 91 GeV → 250 GeV → 350 GeV **Feasibility study towards 2025**
    Coupled with 100 TeV hadron collider. 13 BCHF (2 x ILC) Operation start at 2048 (at Z-pole)
  - CEPC (China)
    Slightly conservative than FCCee. TDR under preparation. To be upgraded to SppC (hadron collider)

ECFA Higgs factory studies

Overview

Based on the recommendations of the European Strategy for Particle Physics Update, the European Committee for Future Accelerators (ECFA) has launched a series of workshops on physics studies, experiment design, and detector technologies towards a future electron-positron Higgs/EW/Top factory. The aim is to bring together the efforts of various e^+e^- projects, to share challenges and expertise, to explore synergies, and to respond coherently to this high-priority strategy item.

To set up the relevant structures and to define a path towards such workshops, an International Advisory Committee (IAC) was formed, which established three Working Groups led by conveners from both experiment and theory:

WG 1: Physics Potential
Conveners: Patrick Koppenburg (NIKHEF), Jenny List (DESY), Fabio Maltoni (UC Louvain / Bologna) and Jorge de Blas (Univ. Granada)
More information on WG 1 activities

WG 2: Physics Analysis Methods
Conveners: Patrizia Azzi (INFN-Padova / CERN), Fulvio Piccinini (INFN Pavia) and Dirk Zerwas (IJCLab/DMLab)
More information on WG2 activities

WG 3: Detector R&D
Conveners: Mary Cruz Fouz (CIEMAT Madrid), Giovanni Marchiori (APC Paris), Felix Sefkow (DESY)
More information on WG3 activities

While the first two working groups began their work in spring 2021, the third one was formed later, after finalisation of the ECFA Detector R&D Roadmap.

Physics cases and detector technologies are mostly common to all Higgs factory projects

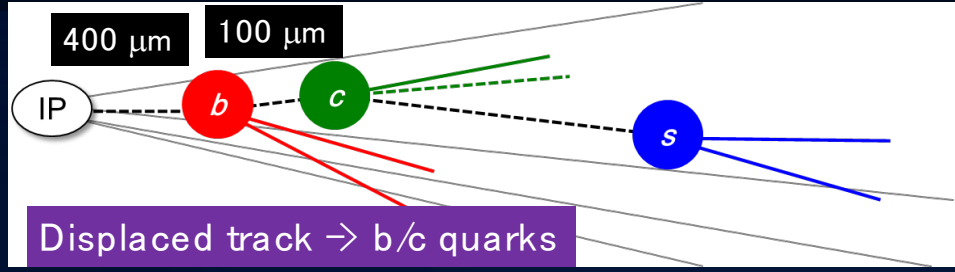
Common framework of e+e- Higgs factory study of physics and detectors (FCCee, ILC, …) is efficient

Activities and report towards 2025

Common software development is one of the key topics

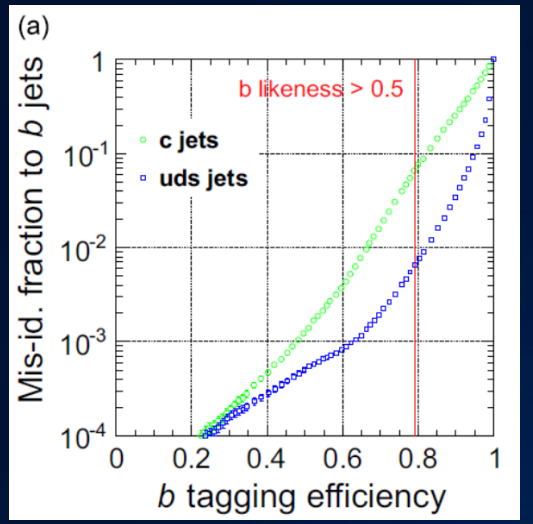
- Flavor tagging, particle ID
- Particle flow
- …

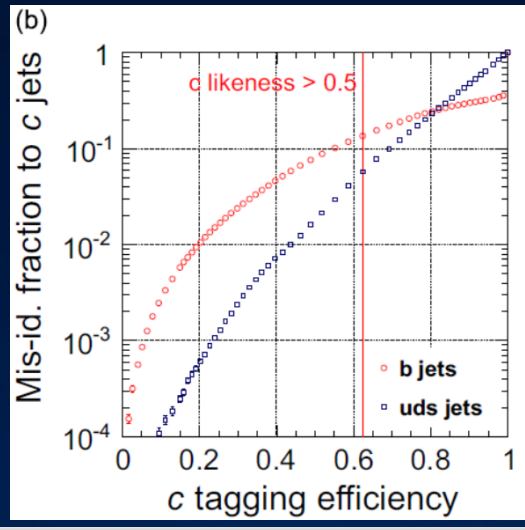
Flavor tagging for Higgs factories

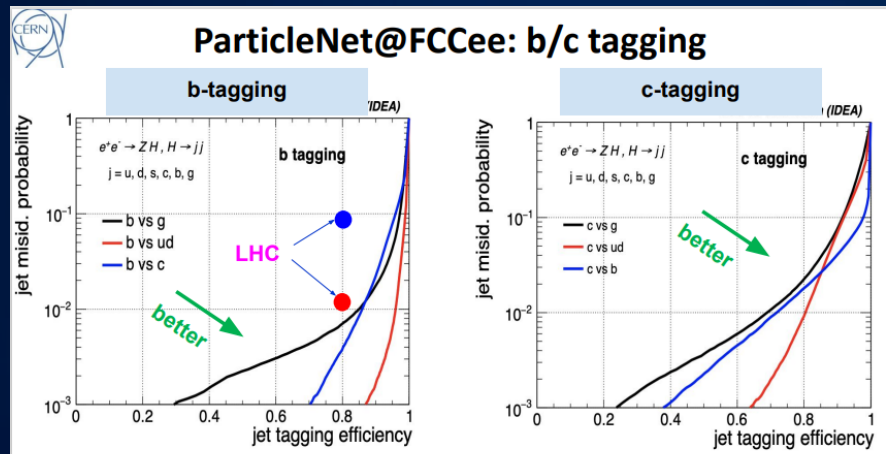


- Jet flavor tagging is essentially important for Higgs studies (including self coupling)
- LCFIPlus (published 2013)[1] was long used for flavor tagging
  - b-tag: ~80% eff., 10% c / 1% uds acceptance;
  - c-tag: ~50% eff., 10% b / 2% uds acceptance.
- Recently FCCee reported ~10x better rejection using ParticleNet (GNN)
  - To be confirmed with full simulation (with latest algorithm: Particle Transformer (ParT)

→ If good, consider to apply to physics analyses hopefully with common framework





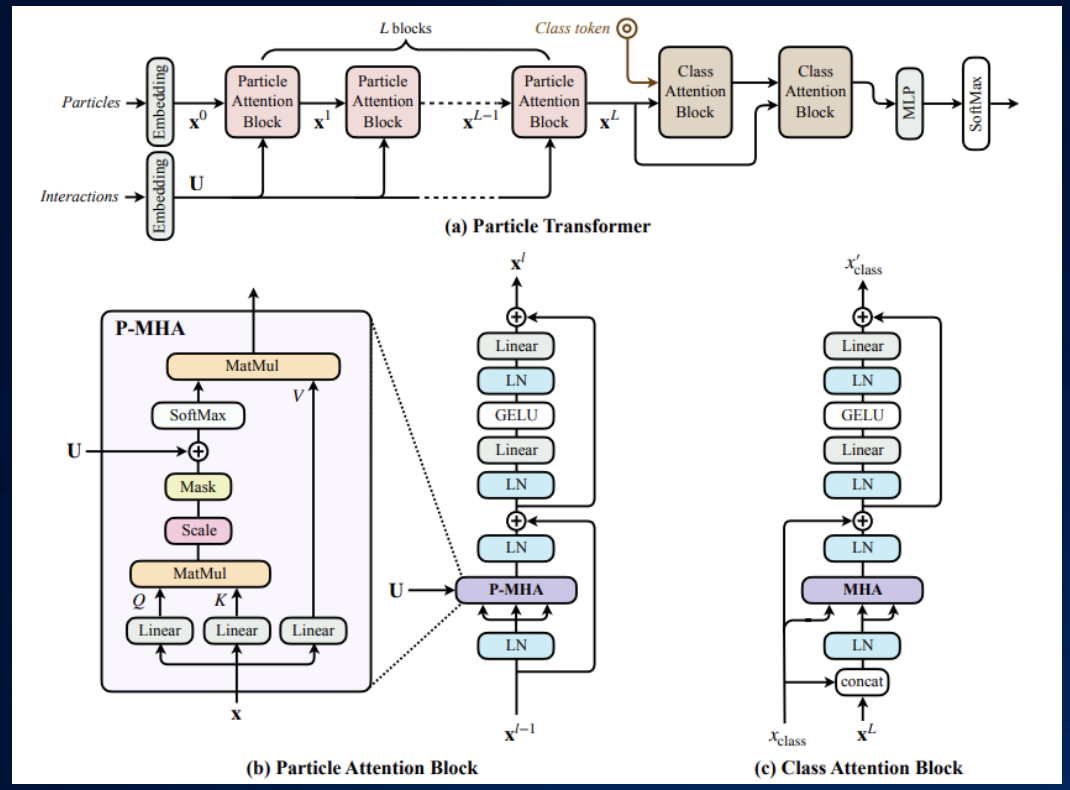


WP	Eff (b)	Mistag (g)	Mistag (ud)	Mistag (c)
Loose	90%	2%	0.1%	2%
Medium	80%	0.7%	<0.1%	0.3%

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

Particle Transformer (ParT)

- Transformer: self-attention based algorithm intensively used for NLP (e.g. chatGPT)
  - Weak biasing: possible to train big samples efficiently (with more learnable weights)
    but demanding big training sample for high performance
- ParT is a new Transformer-based architecture for Jet tagging, published in 2022[2].
- Surpasses the performance of previous architectures



Performance on event categorization (ie. not direct flavor tagging but flavor information is essential for the categorization)

	All classes		$H \to b\bar{b}$	$H \to c\bar{c}$	$H \to gg$	$H \to 4q$	$H \to \ell\nu qq'$	$t \to bqq'$	$t \to b\ell\nu$	$W \to qq'$	$Z \to q\bar{q}$
	Accuracy	AUC	$\text{Rej}_{50\%}$	$\text{Rej}_{50\%}$	$\text{Rej}_{50\%}$	$\text{Rej}_{50\%}$	$\text{Rej}_{99\%}$	$\text{Rej}_{50\%}$	$\text{Rej}_{99.5\%}$	$\text{Rej}_{50\%}$	$\text{Rej}_{50\%}$
PFN	0.772	0.9714	2924	841	75	198	265	797	721	189	159
P-CNN	0.809	0.9789	4890	1276	88	474	947	2907	2304	241	204
ParticleNet	0.844	0.9849	7634	2475	104	954	3339	10526	11173	347	283
ParT	**0.861**	**0.9877**	**10638**	**4149**	**123**	**1864**	**5479**	**32787**	**15873**	**543**	**402**
ParT (plain)	0.849	0.9859	9569	2911	112	1185	3868	17699	12987	384	311

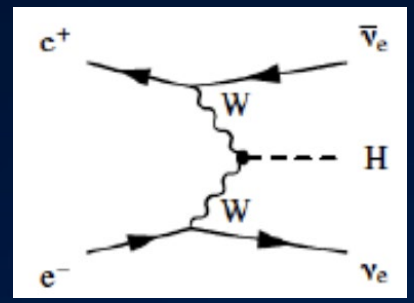
Data Used For Investigation

- ILD full simulation:

  1. e+ e- ⟶ qq (at 91 GeV)
     (DBD sample used for initial LCFIPlus study)
  2. e+ e- ⟶ νν**H** ⟶ ννqq (at 250 GeV)
     (2020 production, process ID: 410001-410006)

  With 1M jets (500k events) each

q = b,c,uds
ν = neutrino

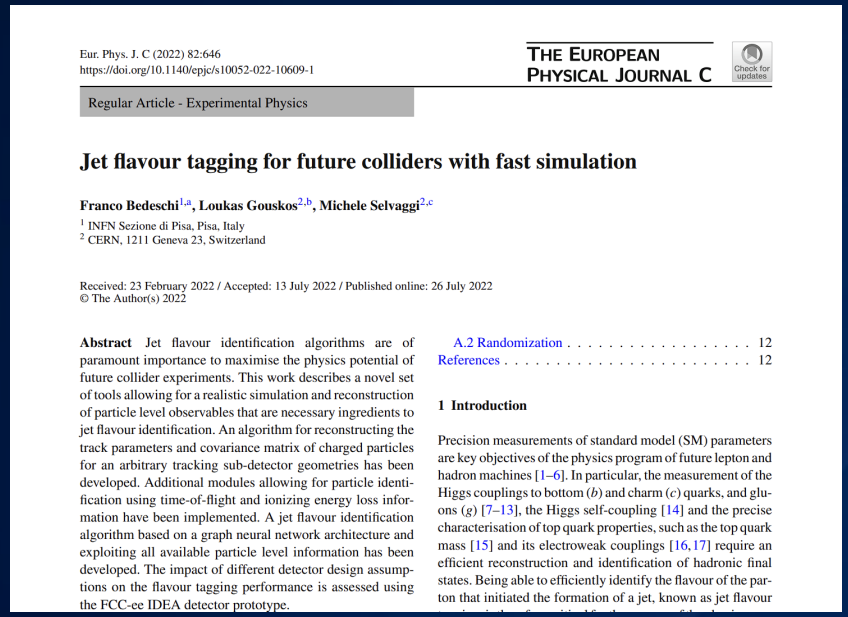


- FCCee fast simulation (Delphes with IDEA detector):

  e+ e- ⟶ νν**H** ⟶ ννqq (at 240 GeV)

  With 10M jets (5M events) each
  (provided as ROOT files, no modification possible)

- 80% are used for training, 5% for validation, 15% for test



Eur. Phys. J. C (2022) 82:646
https://doi.org/10.1140/epjc/s10052-022-10609-1

THE EUROPEAN PHYSICAL JOURNAL C

Regular Article - Experimental Physics

Jet flavour tagging for future colliders with fast simulation

Franco Bedeschi[1,a], Loukas Gouskos[2,b], Michele Selvaggi[2,c]

[1] INFN Sezione di Pisa, Pisa, Italy
[2] CERN, 1211 Geneva 23, Switzerland

Received: 23 February 2022 / Accepted: 13 July 2022 / Published online: 26 July 2022
© The Author(s) 2022

Abstract Jet flavour identification algorithms are of paramount importance to maximise the physics potential of future collider experiments. This work describes a novel set of tools allowing for a realistic simulation and reconstruction of particle level observables that are necessary ingredients to jet flavour identification. An algorithm for reconstructing the track parameters and covariance matrix of charged particles for an arbitrary tracking sub-detector geometries has been developed. Additional modules allowing for particle identification using time-of-flight and ionizing energy loss information have been implemented. A jet flavour identification algorithm based on a graph neural network architecture and exploiting all available particle level information has been developed. The impact of different detector design assumptions on the flavour tagging performance is assessed using the FCC-ee IDEA detector prototype.

A.2 Randomization 12
References 12

1 Introduction

Precision measurements of standard model (SM) parameters are key objectives of the physics program of future lepton and hadron machines [1–6]. In particular, the measurement of the Higgs couplings to bottom (b) and charm (c) quarks, and gluons (g) [7–13], the Higgs self-coupling [14] and the precise characterisation of top quark properties, such as the top quark mass [15] and its electroweak couplings [16,17] require an efficient reconstruction and identification of hadronic final states. Being able to efficiently identify the flavour of the parton that initiated the formation of a jet, known as jet flavour

https://link.springer.com/article/10.1140/epjc/s10052-022-10609-1

Software for Particle Transformer

- Public in github, with instruction provided
  - https://github.com/jet-universe/particle_transformer
- Input: ROOT files for training (80%), validation (5%), test (15%)
  - Input variables can be provided via steering file (XML)
    - Input for each particle (tracks, neutral clusters)
    - Input for “interaction” → currently momentum only
    - Input for “coordinate” → theta/phi plan wrt. jet axis
- Output: ROOT files including evaluation results (likeness) for test events
  - To be analyzed with ROOT or so
- We implemented a processor (inside LCFIPlus) to produce ROOT files for input as much as compatible to FCCee variables
  - Except for PID values, which are not fully implemented
- Easy for testing, but not direct to be used for physics analyses

Input Variables - Features

*Naming follows FCCee scheme – may not express exact meaning

- Impact Parameter (6):
  - pfcand_dxy
  - pfcand_dz
  - pfcand_btagSip2dVal
  - pfcand_btagSip2dSig
  - pfcand_btagSip3dVal
  - pfcand_btagSip3dSig

*d0/z0 and 2D/3D impact parameters, -9 for neutrals

- Jet Distance (2):
  - pfcand_btagJetDistVal
  - pfcand_btagJetDistSig

*Displacement of tracks from line passing IP with direction of jet
-9 for neutrals

- Particle ID (6):
  - pfcand_isMu
  - pfcand_isEl
  - pfcand_isChargedHad
  - pfcand_isGamma
  - pfcand_isNeutralHad
  - pfcand_type

* Not including strange-tagging related variables (TOF, dE/dx etc.)
* Simple PID for ILD, not optimal

- Kinematic (4):
  - pfcand_erel_log
  - pfcand_thetarel
  - pfcand_phirel
  - pfcand_charge

*Fraction of the particle energy wrt. jet energy (log is taken)

- Track Errors (15):
  - pfcand_dptdpt
  - pfcand_detadeta
  - pfcand_dphidphi
  - pfcand_dxydxy
  - pfcand_dzdz
  - pfcand_dxydz
  - pfcand_dphidxy
  - pfcand_dlambdadz
  - pfcand_dxyc
  - pfcand_dxyctgtheta
  - pfcand_phic
  - pfcand_phidz
  - pfcand_phictgtheta
  - pfcand_cdz
  - pfcand_cctgtheta

*each element of covariant matrix
-9 for neutrals

Input Variables - Interactions

- FCC data uses p (scalar momentum) as interaction:

  - pfcand_p

- ILD data contains p_x, p_y, p_z (vector momentum) as interaction:

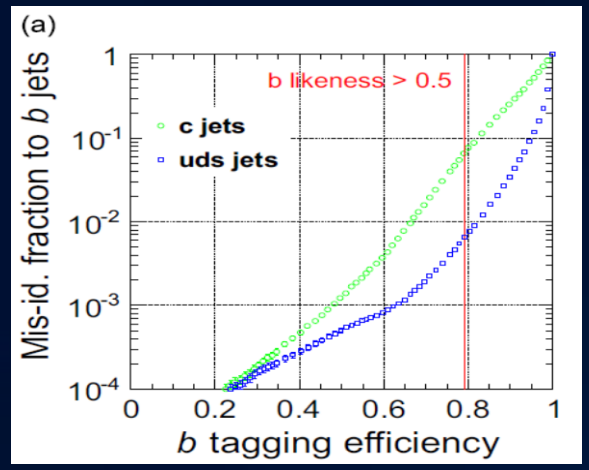
  - pfcand_px
  - pfcand_py
  - pfcand_pz

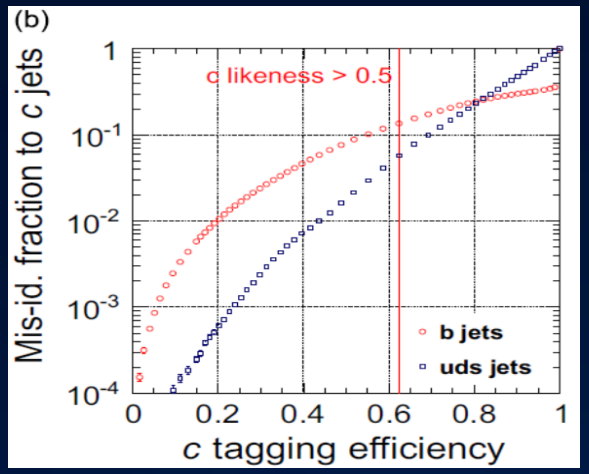
- But it’s possible to transfer ILD’s interaction to FCC’s form for fair comparison:

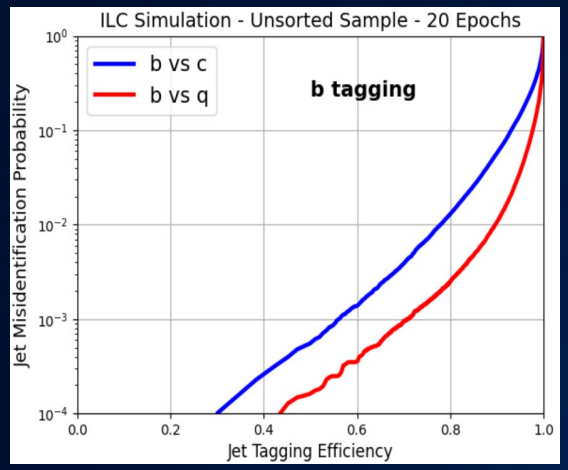
$$p = \sqrt{{p_x}^2 + {p_y}^2 + {p_z}^2}$$

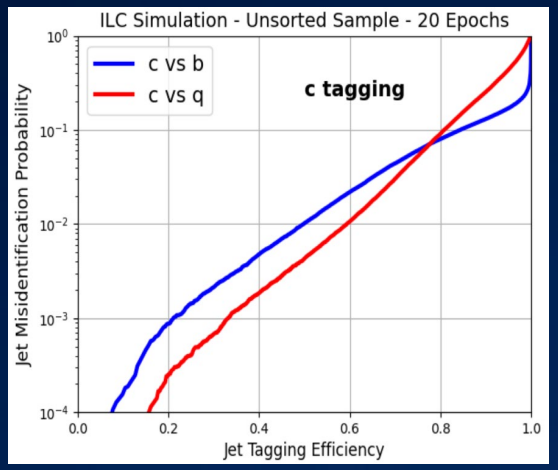
Application of ParT to ILD data
(ILD qq 91 GeV, 0.8M jets for training)

- Jet tagging performance is greatly improved by ParT immediately.
- The performance is improved by 4.05 – 9.80 times compared to LCFIPlus with the same set of data.
- 20 epochs are taken,
  200 epochs do not help improving performance but give overtraining





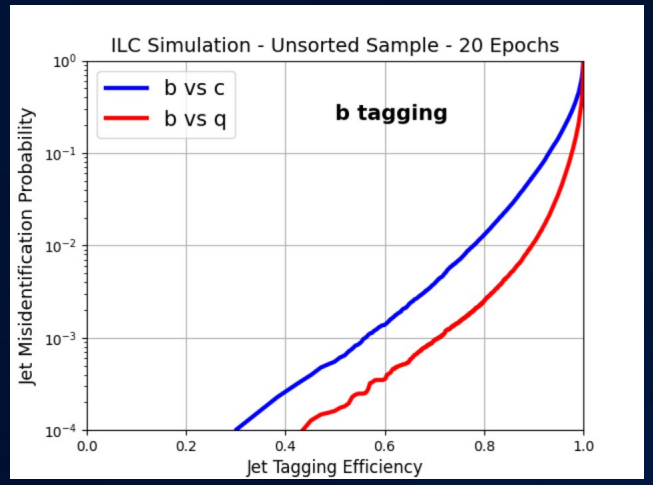


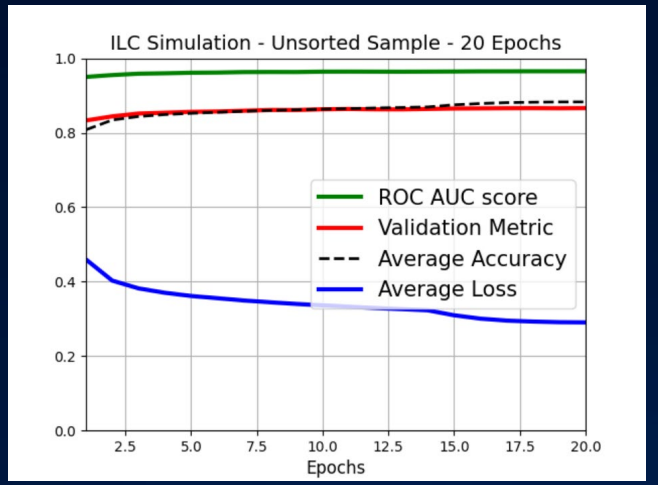


	b-tag 80% eff.		c-tag 50% eff.	
Method	**c-bkg acceptance**	**uds-bkg acceptance**	**c-bkg acceptance**	**uds-bkg acceptance**
LCFIPlus	**10%**	**1%**	**10%**	**2%**
ParT	**1.29%**	**0.25%**	**1.02%**	**0.43%**

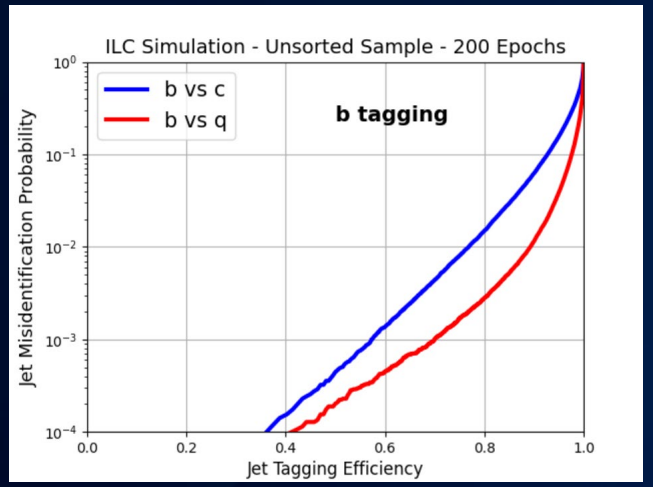
Training parameters - epochs

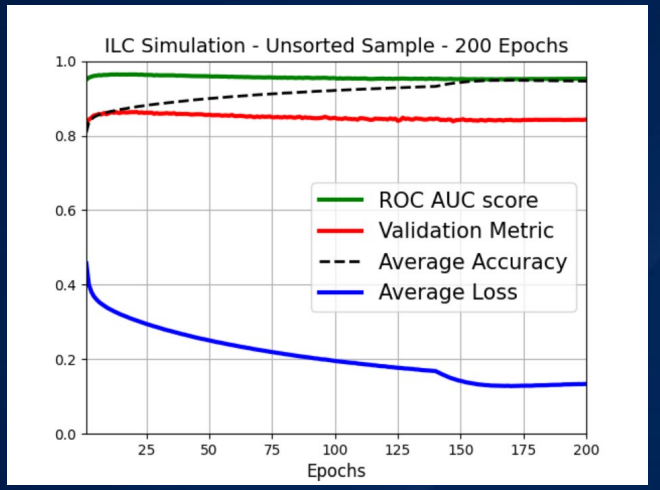
- Run on NVIDIA TITAN RTX (memory: 24 GB)
  - 20 Epochs: 3 hours
  - 200 Epochs: 30 hours
- No significant improvement in tagging efficiency
- Both ROC AUC score and Validation Metric reaches a maximum around 20 epochs.
- Overtraining after 20 epochs.
- Hence 20 epochs of training is selected to avoid overtraining.





20 epochs (ILD qq 91 GeV)

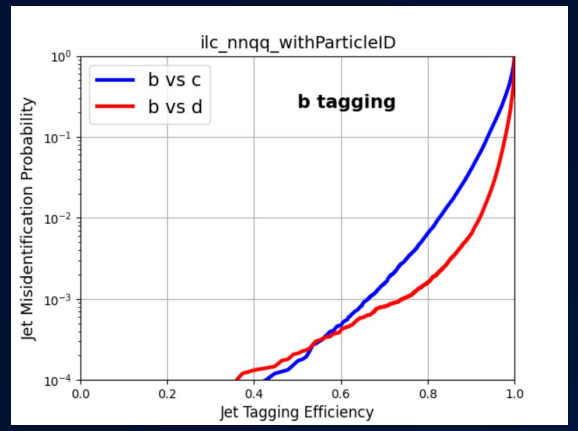


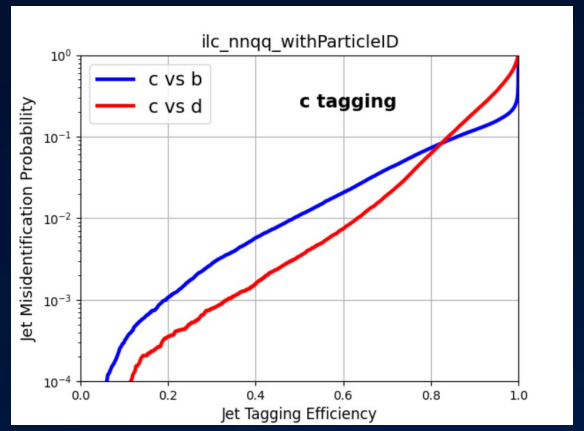


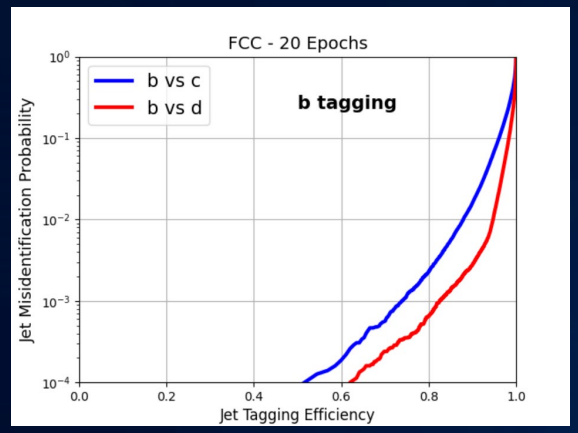
200 epochs (ILD qq 91 GeV)

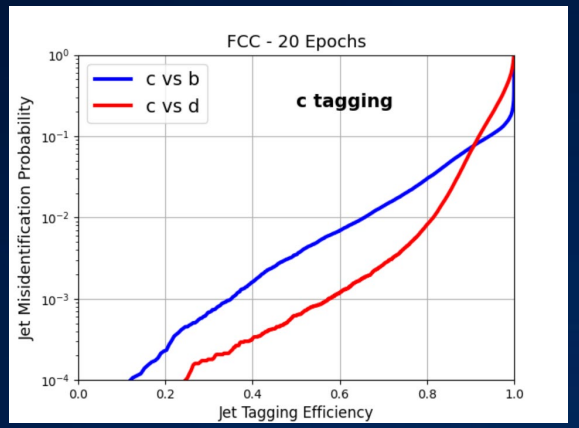
Comparison with FCC data[3]

- Trained with same condition as ILD data for fair comparison. (800k data size, 20 epochs, etc.)
- FCC data has ~ 3 times the performance compared to ILD data.
- Possible cause of the difference:
  - Particle ID: too pessimistic for ILD
  - Definition of some variables
    - Theta, phi etc.
  - Difference on full and fast sim
    - Especially different on tails of distributions
  - Assumed detector resolution (?)





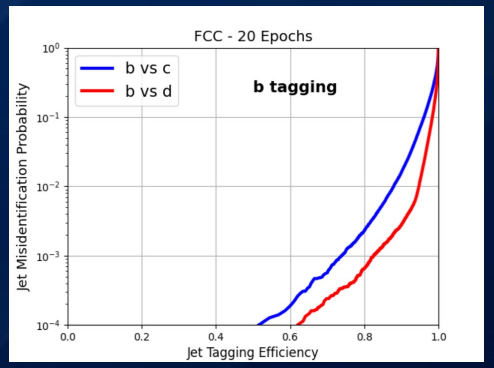




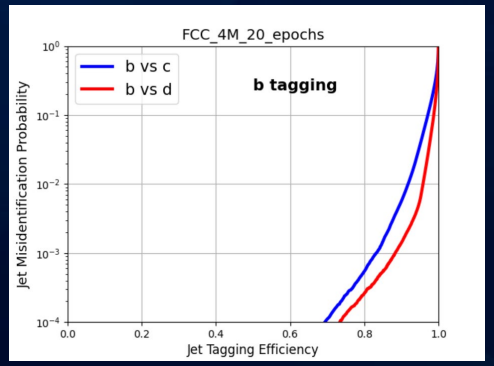
Data	Particle ID	Impact Parameters	Jet Distance	Track Errors	c-bkg acceptance @ b-tag 80% eff.	b-bkg acceptance @ c-tag 50% eff.
ILD (vvqq 250 GeV)	*	*	*	*	**0.64%**	**1.09%**
FCC	*	*	*	*	**0.23%**	**0.35%**

Sample size affects performance (FCCee sample)

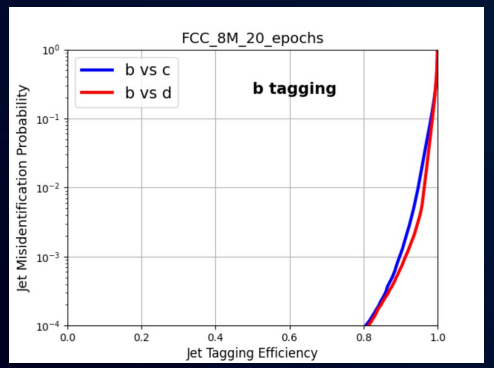
(1)



(2)



(3)



Plot Index	Particle ID	Impact Parameters	Jet Distance	Track Errors	Training Sample size	c-bkg acceptance @ b-tag 80% eff.	b-bkg acceptance @ c-tag 50% eff.
(1)	●	●	●	●	800k	0.23%	0.35%
(2)	●	●	●	●	4M	0.054%	0.20%
(3)	●	●	●	●	8M	0.0076%	0.10%

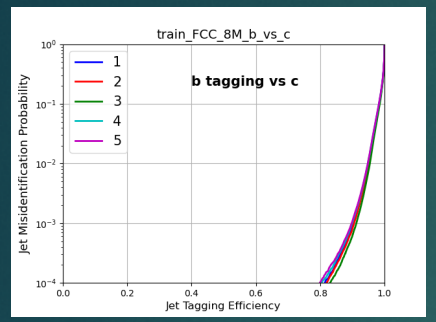
Unreasonably good!

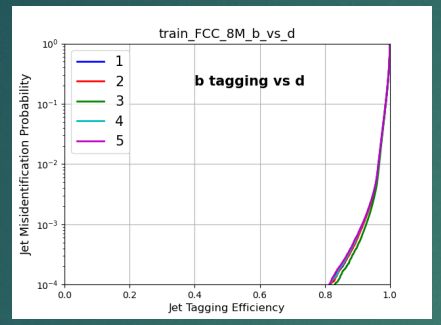
- Training performance significantly improved with bigger data sample size
- Training sample size change of FCC data:

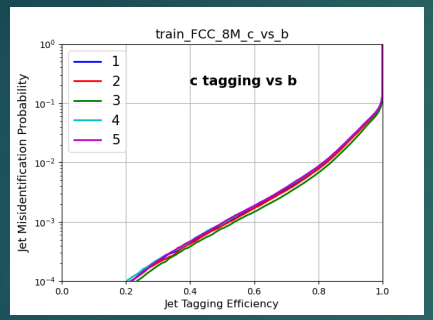
  800k -> 4M : 4 times better performance (b-tagging)

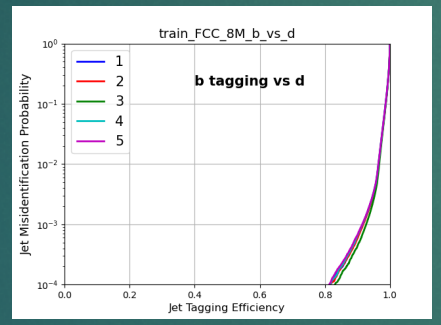
  4M -> 8M: 5 times better performance (b-tagging)
- This non-linearity of increase in performance should be further investigated.
- Bigger data size of ILD should be obtained for better performance, as well as comparison with FCC data for further investigation on its behaviour.

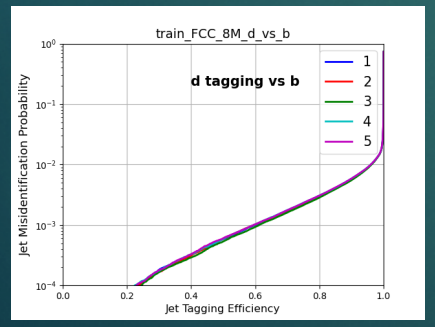
Multiple Training Runs

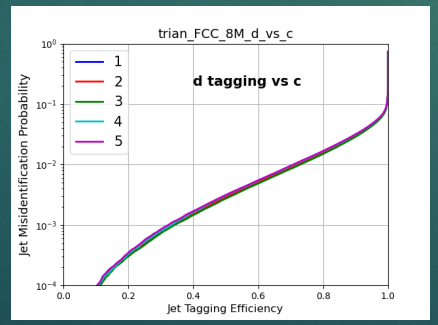










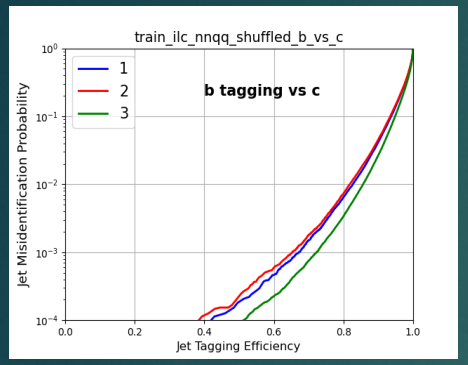


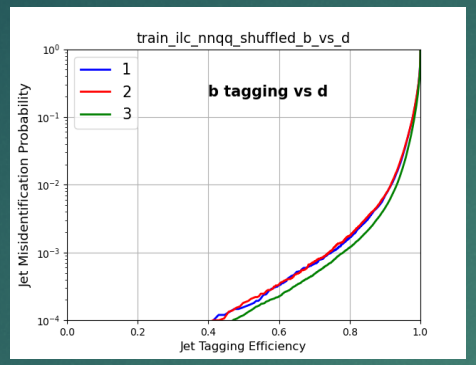
5 times training of FCC_8M data

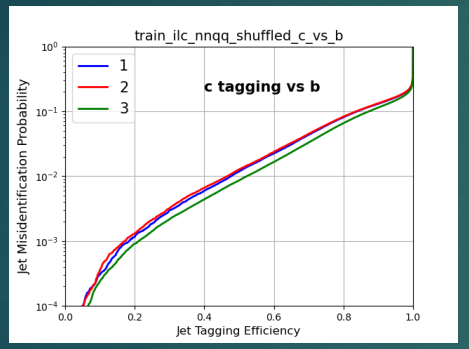
- Multiple training runs don't give significant impacts on results.
- The smaller data size is, the bigger impacts on results multiple runs give.
- The results of no Particle ID trainings varies more than those of with Particle ID.

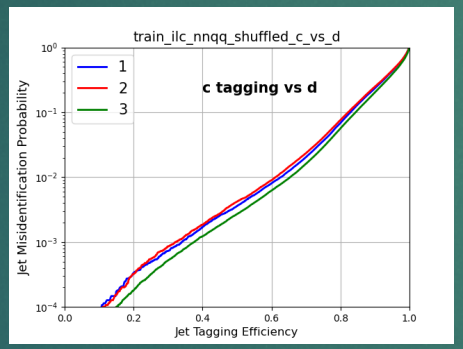
data	Particle ID	b vs c 0.8 Score	variation
FCC 4M	O	4.82e-4	0.43e-4
FCC 8M	O	8.14e-5	1.58e-5
FCC 4M	×	1.69e-3	0.14e-3
FCC 8M	×	7.04e-4	3.49e-4

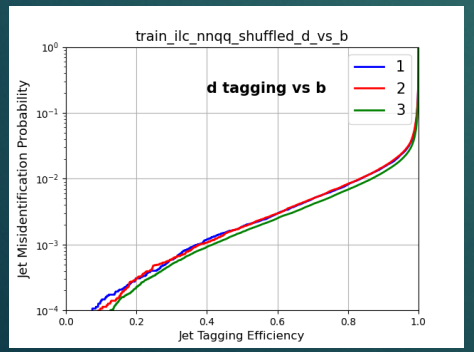
Data Shuffled

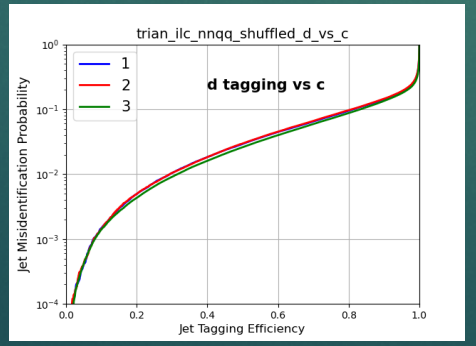












- Shuffled the order of train/test/val making root files
- The resulting values are highly variable.

data	b vs c 0.8 score
Shuffle pattern 1	0.00647
Shuffle pattern 2	0.00734
Shuffle pattern 3	0.00338

Fine tuning

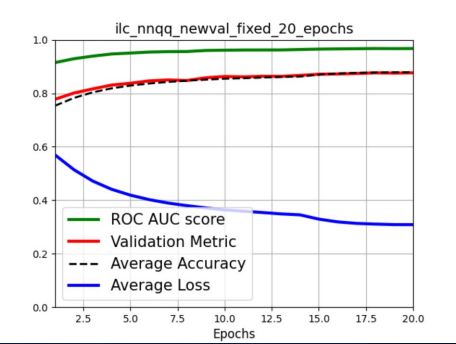
Two objectives
- Pretrained with fast sim and fine-tune with full sim
- Pretrained with large central production and fine-tune with dedicated physics samples in each analysis

							c-bkg acceptance @ b-tag 80% eff.		b-bkg acceptance @ c-tag 50% eff.	
Particle ID	Impact Parameters	Jet Distance	Track Errors	Fine-Tuning Sample	Training Sample	Similar theta/phi ?	No Fine-Tuning	With Fine-Tuning	No Fine-Tuning	With Fine-Tuning
✗	●	●	●	FCC 240 GeV (8M)	ILD 250 GeV (800k)	✗	0.62%	1.37%	1.14%	1.95%
✗	●	●	●	FCC 240 GeV (8M)	ILD 250 GeV (800k)	●	1.77%	1.32%	2.22%	2.01%
●	●	●	●	ILD 250 GeV (800k)	ILD 91 GeV (80k)	●	4.49%	0.97%	3.79%	1.53%

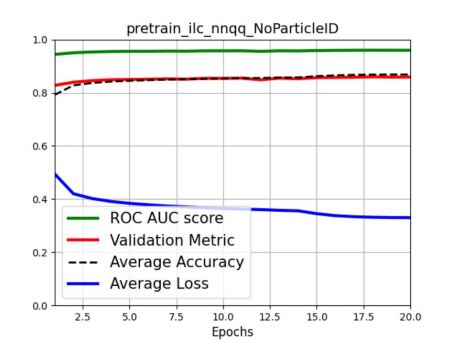
- Use result of 8M FCC data to train ILD 800k data
- Improves performance only when setups are similar
- Training of same setup (pretrain ILD 91 GeV data with ILD 250 GeV data) gives best performance
- Further investigation should be conducted on how to maximise the outcome for fine-tuning between different data sets

Fine tuning – Training curves

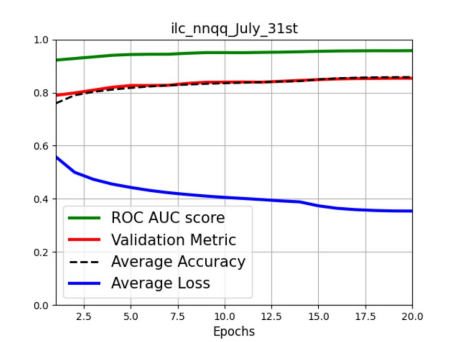
(1)



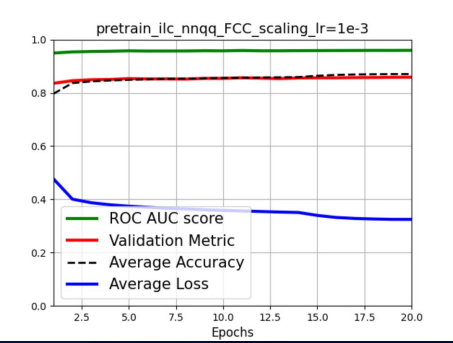
(2)



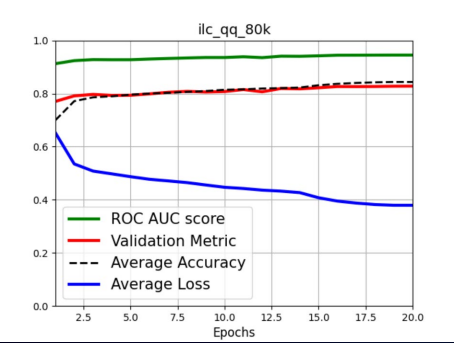
(3)



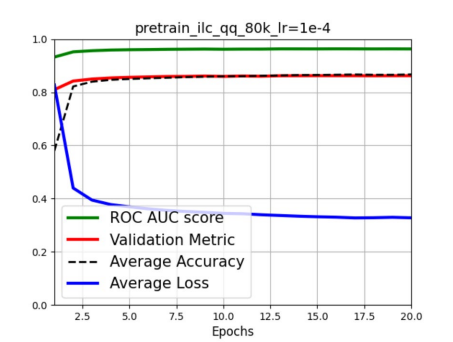
(4)



(5)



(6)



							Plot Indices	
Particle ID	Impact Parameters	Jet Distance	Track Errors	Fine-Tuning Sample	Training Sample	Similar theta/ phi?	No Fine-Tuning	With Fine-Tuning
✗	●	●	●	FCC 240 GeV (8M)	ILD 250 GeV (800k)	✗	(1)	(2)
✗	●	●	●	FCC 240 GeV (8M)	ILD 250 GeV (800k)	●	(3)	(4)
●	●	●	●	ILD 250 GeV (800k)	ILD 91 GeV (80k)	●	(5)	(6)

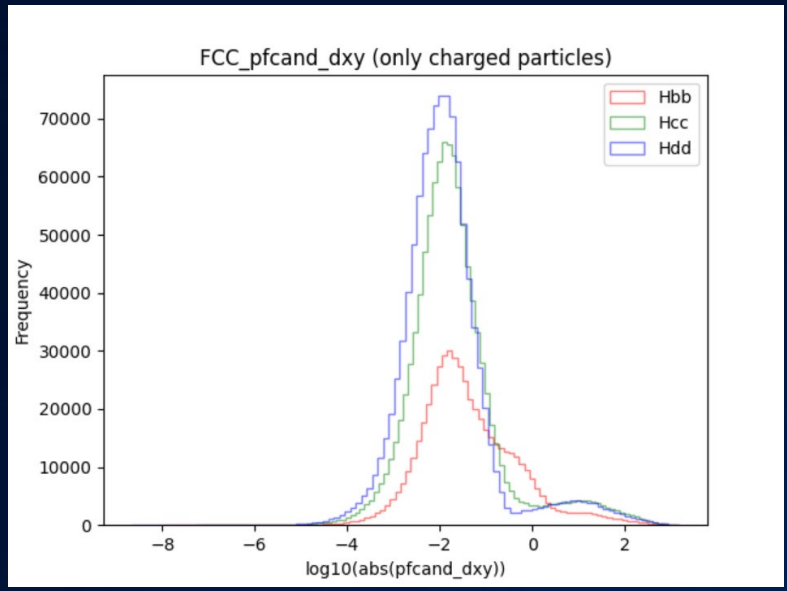
- With fine-tuning, the training is obviously accelerated for the initial epochs (even for those with worse eventual performance)
- This is particularly obvious between plots (5) & (6) – similar simulation setup data

What we are working (and plan to work) now

- Conversion of variables
  - Linear normalization implemented in weaver, but errors/impact parameters are important only in < 5-10 sigma ranges → truncate with sigmoid or log?
- Treatment of neutral particles
  - Removing neutrals significantly degrade performance, but having -9 is very unnatural
  - Trying different encoding to tracks and neutrals: how is the best implementation?
- Strange tagging
  - Including hadron ID (proton/kaon/pion) obtained by dE/dx information at gaseous tracker
  - Also consider to have timing information (TOF)
  - Under preparation for ILD
- Inference at our reconstruction framework
  - Software stack is fully C++, “processors” treat reconstruction
  - Weaver is not so friendly to the integration
  - Maybe exporting network and use in C++ (with libTorch implementation)
  - Any experience in LHC?

Potential Improvement: log(abs)

Particle ID	Impact Parameters	Jet Distance	Track Errors	c-bkg acceptance @ b-tag 80% eff.	b-bkg acceptance @ c-tag 50% eff.
-	*	*	*	**0.62%**	**1.14%**
-	* +log(abs)	* +log(abs)	* +log(abs)	**0.54%**	**1.06%**
-	*	* +log(abs)	* +log(abs)	**0.79%**	**1.33%**
-	*	* +log(abs)	*	**0.78%**	**1.36%**
-	* +log(abs)	*	*	**0.47%**	**1.03%**
-	log(abs)	log(abs)	log(abs)	**0.82%**	**1.32%**
-	*	log(abs)	log(abs)	**0.80%**	**1.37%**
-	*	*	log(abs)	**0.82%**	**1.38%**

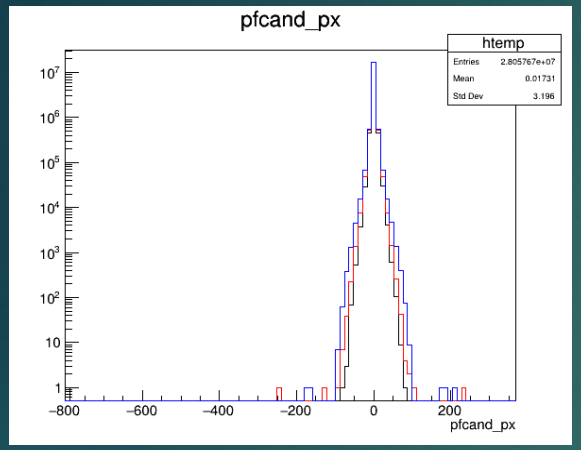


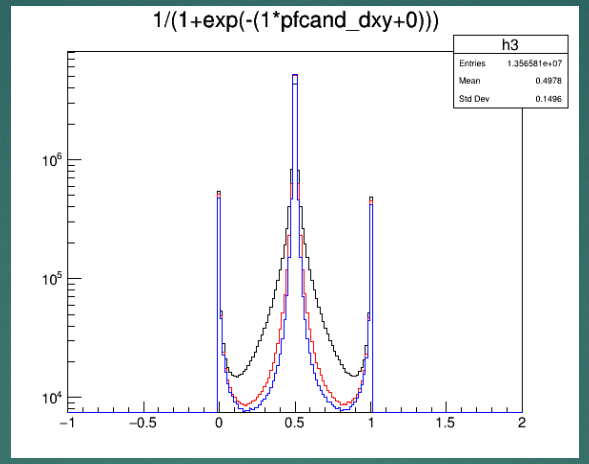
Impact Parameter

ML prefers “gaussian-like” distribution
Not sensitive to small values
(because of linear weighting)

Track errors or impact parameters should convert with e.g. log function
→ slightly improving performance
(but not much as expected…)

Sigmoid





- Applying Sigmoid function to the variables with wide distribution
- The score is better than that of not applying sigmoid.

data	sigmoid	b vs c 0.8 score
llc_nnqq	×	0.00647±0.00054
llc_nnqq	O	0.00535±0.00032

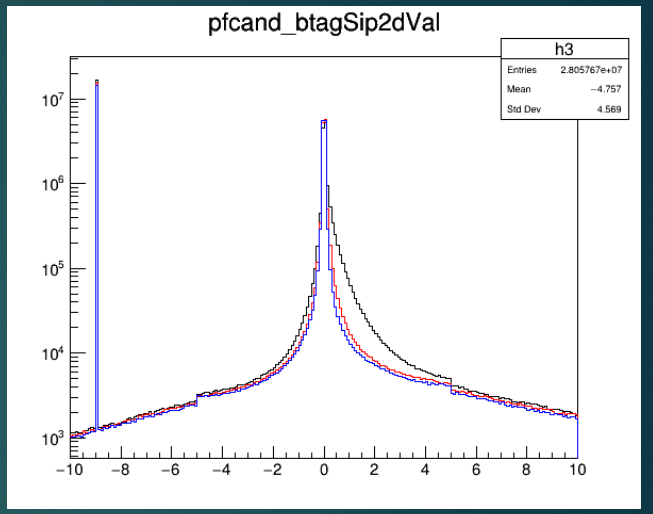
Each of the score is the average of 3 times training with standard variation

- Processed variables (8)

pfcand_dxy
pfcand_dz
pfcand_btagSip2dVal
pfcand_btagSip2dSig
pfcand_btagSip3dVal
pfcand_btagSip3dSig
pfcand_dxydz
pfcand_dphidxy

Different Networks for Neutrals

- Currently tracks and neutrals are passing same embedding layer
  - For parameters not available in neutrals, "-9" is set (right figure)
  - Without neutrals, performance is significantly degraded
    - b/c separation (b selection eff. = 80%) in ILC nnqq sample (1M jets):
      - With neutrals: rejection ratio = 123 (acceptance: 0.647%)
      - Without neutrals: rejection ratio = 62.5 (acceptance: 1.28%)
- Tracks and neutrals have flags (like "charge")
- At the initial stage of transformer, they should be separated and going through different embedding network
  - But variables like energy/momentum are common: need some treatment?
- Combine tracks and neutrals
  - Should keep some flags to discriminate tracks and neutrals?



Attention variables

- "Interactions" are used to calculate pairwise variables (but hard-coded)
  - Rapidity is not usually used in e+e- colliders (since particles are not very concentrated in forward region)
  - Maybe better to have invariant mass or y-value for (kt-like) jet clustering

```
def pairwise_lv_fts(xi, xj, num_outputs=4, eps=1e-8, for_onnx=False):
    pti, rapi, phii = to_ptrapphim(xi, False, eps=None, for_onnx=for_onnx).split((1, 1, 1), dim=1)
    ptj, rapj, phij = to_ptrapphim(xj, False, eps=None, for_onnx=for_onnx).split((1, 1, 1), dim=1)

    delta = delta_r2(rapi, phii, rapj, phij).sqrt()
    lndelta = torch.log(delta.clamp(min=eps))
    if num_outputs == 1:
        return lndelta
```

```
def to_ptrapphim(x, return_mass=True, eps=1e-8, for_onnx=False):
    # x: (N, 4, ...), dim1 : (px, py, pz, E)
    px, py, pz, energy = x.split((1, 1, 1, 1), dim=1)
    pt = torch.sqrt(to_pt2(x, eps=eps))
    # rapidity = 0.5 * torch.log((energy + pz) / (energy - pz))
    rapidity = 0.5 * torch.log(1 + (2 * pz) / (energy - pz).clamp(min=1e-20))
    phi = (atan2 if for_onnx else torch.atan2)(py, px)
    if not return_mass:
        return torch.cat((pt, rapidity, phi), dim=1)
    else:
        m = torch.sqrt(to_m2(x, eps=eps))
        return torch.cat((pt, rapidity, phi, m), dim=1)
```

Hyperparameters

```yaml
inputs:
    pf_points:
        pad_mode: wrap
        length: 75
        vars:
            - pfcand_thetarel
            - pfcand_phirel
    pf_features:
        pad_mode: wrap
        length: 75
        vars:
```

- 75 particle maximum

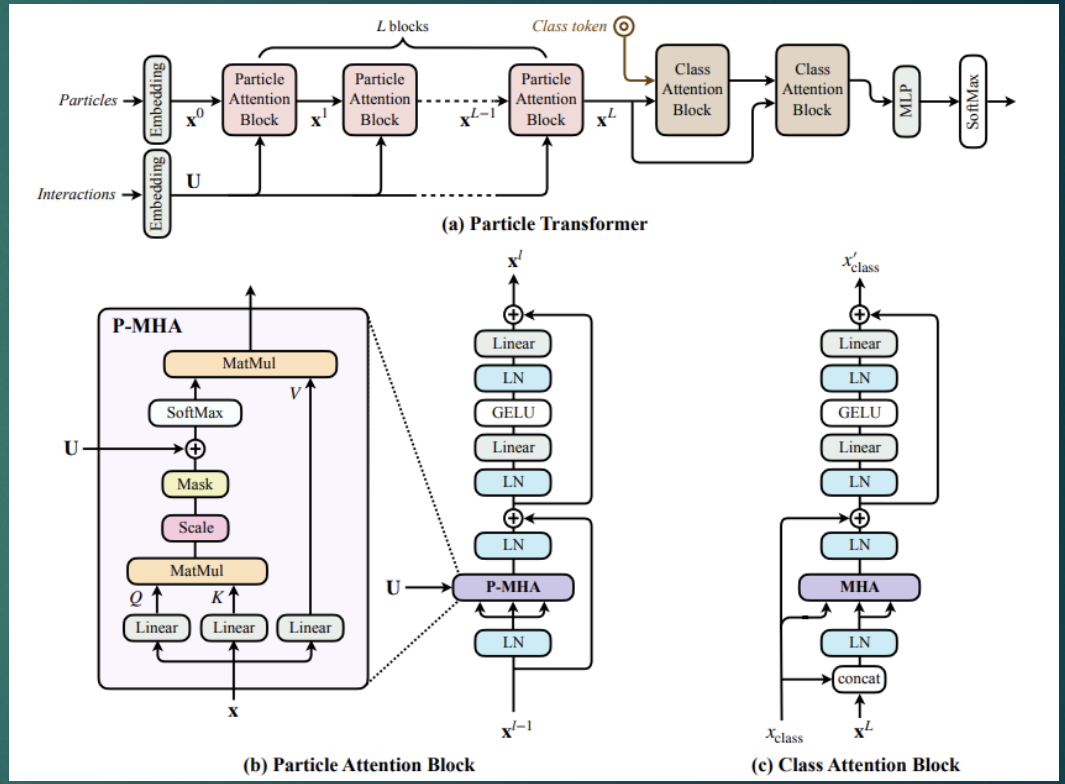
```python
class ParticleTransformer(nn.Module):

    def __init__(self,
                 input_dim,
                 num_classes=None,
                 # network configurations
                 pair_input_dim=4,
                 pair_extra_dim=0,
                 remove_self_pair=False,
                 use_pre_activation_pair=True,
                 embed_dims=[128, 512, 128],
                 pair_embed_dims=[64, 64, 64],
                 num_heads=8,
                 num_layers=8,
                 num_cls_layers=2,
```

- Optimization?
- Should change by sample size?

Network structure

- Interaction is currently introduced at later stage of multi-head attention
  - Positional encoding in Transformer is at the beginning
  - Q/K without positional information
  - Any thought of this implementation?



Other thoughts

- Vertex finding – any idea to integrate vertex finder?
  (e.g. provide vertex fitter results to interaction?)
- Quark charge – sometimes important
  can be implemented at classification
- Effect on detector performance – can be used for detector optimization?
- Foundation model – how to absorb difference on detector/simulation/event topologies
  - Fine-Tuning embedding layer?

- Is ParT or similar algorithm applicable to particle flow with highly-granular calorimeter?
  - PFA is not a classification job – how to implement?
    - But partially a classification job to separate clusters associated with tracks
  - What modification is necessary?