

Flavour tagging at Belle II

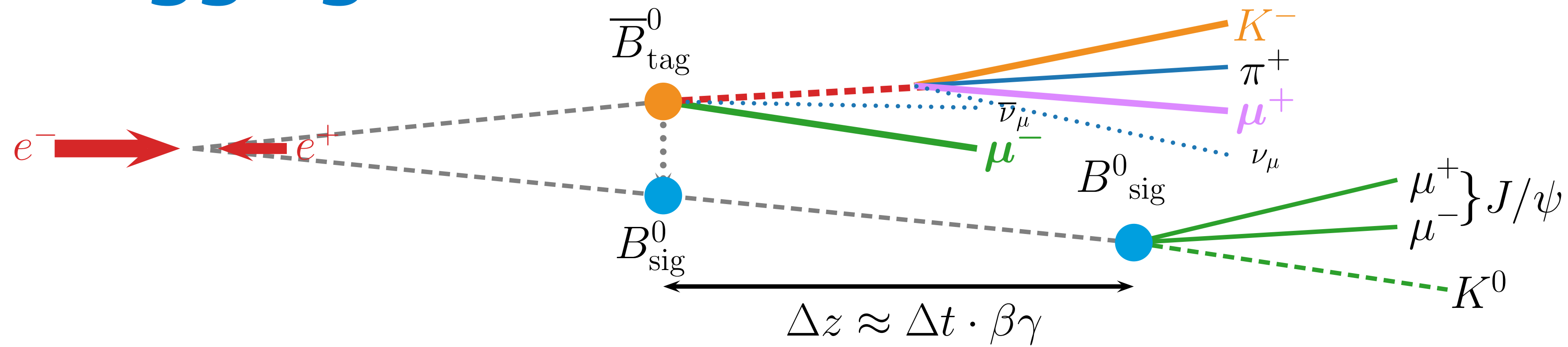
FCC-ee Flavour workshop (CPV group)



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN IN PUBLICA COMMODA
SEIT 1737

Lukas Herzberg, Thibaud Humair, Benjamin Schwenker, 31 March 2026

Flavour tagging at B factories



At Belle II, $B^0\bar{B}^0$ pairs are fully entangled: flavour tagging is determining if one B is \bar{B} or B

- No dilution from B_S^0 or B^0 oscillations
- No same-side tagging

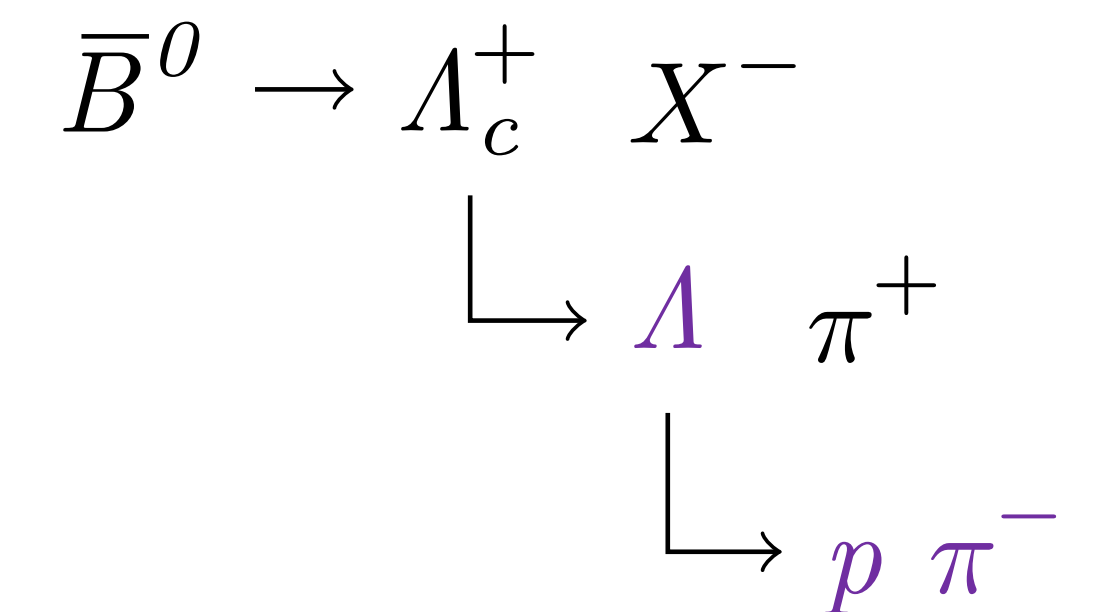
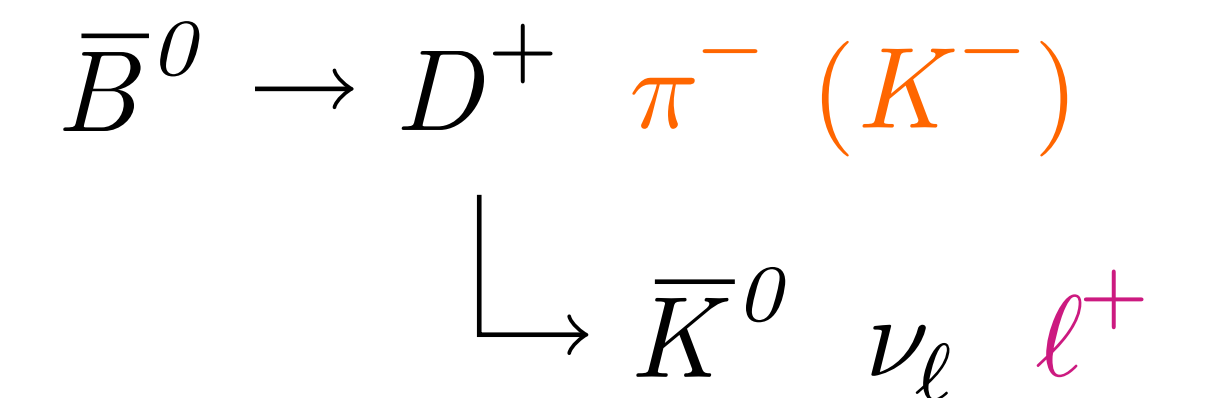
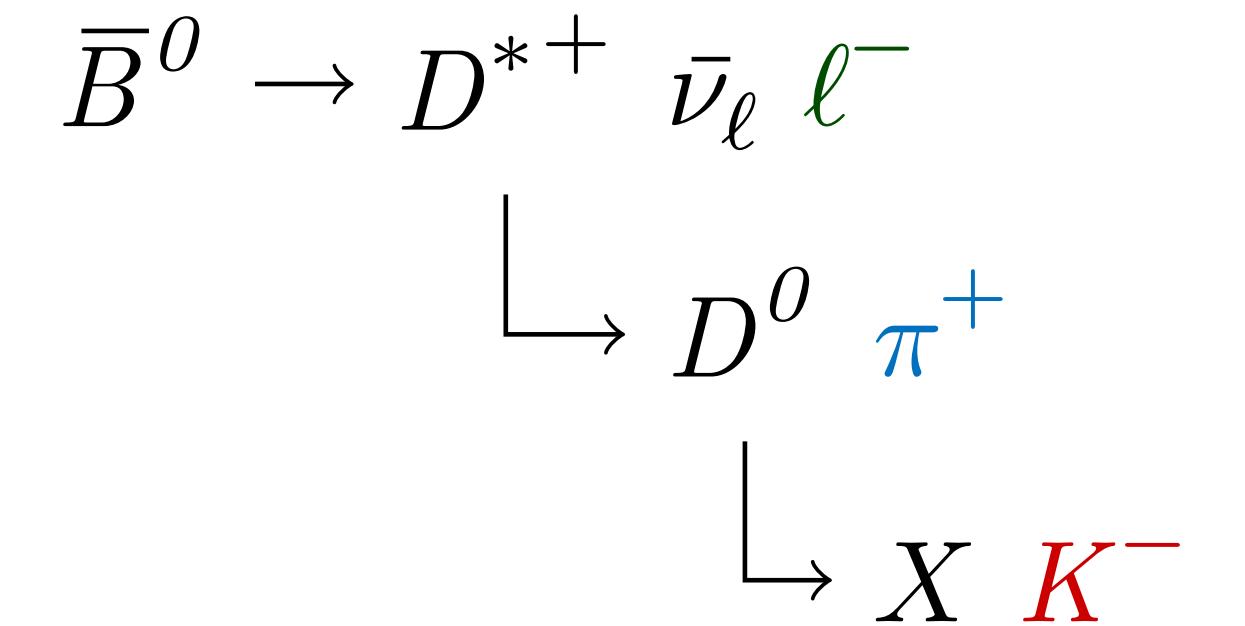
$$\text{Asym} = \frac{N(B_{\text{tag}}^0) - N(\bar{B}_{\text{tag}}^0)}{N(B_{\text{tag}}^0) + N(\bar{B}_{\text{tag}}^0)} = (1 - 2w)\sin(\Delta m_d \Delta t)\sin(2\beta)$$

where w is the fraction of wrongly tagged B_{tag} , want it small!

Traditional Tagger

Until few years ago, used to flavour-tag in two steps:

1. Look for **tagging signatures**: for each B_{tag} track, use BDT to tell whether it belongs to one tagging categories:
 - Fast lepton from B
 - Intermediate kaon from D
 - 11 others: fast hadron, slow pions, intermediate lepton from D ...
2. **Combiner** BDT:
 - Inputs: the highest output value of each category
 - Outputs: $q = \pm 1$ for B^0 or \bar{B}^0 , and $r \in [0,1]$ for the confidence



Performance of the flavour tagger

- The performance is characterised by the effective tagging efficiency:

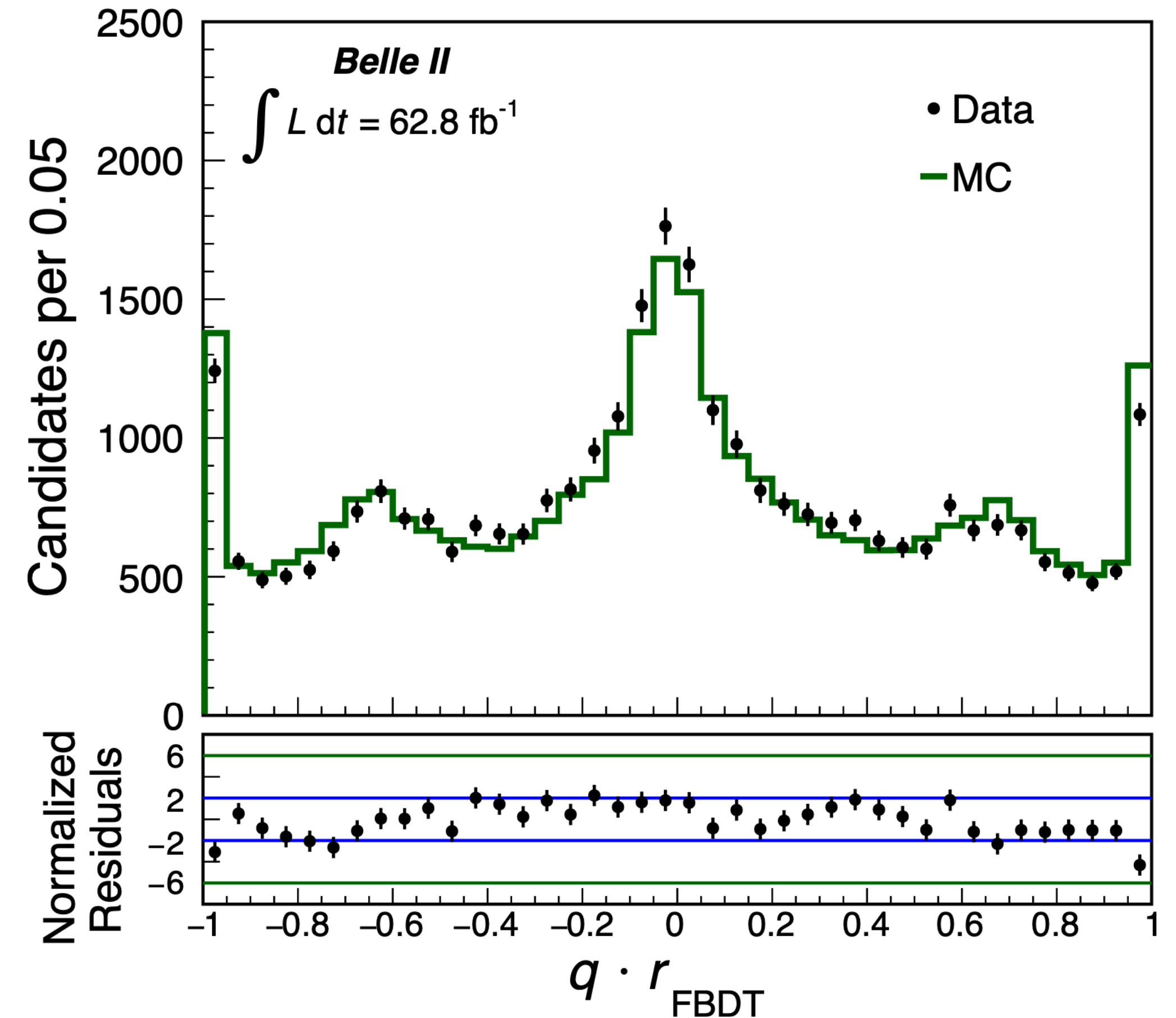
- $$\epsilon_{\text{tag}} = \sum_i \epsilon_i (1 - 2w_i)^2$$

- $$\sigma(\text{CPV}) \sim 1/\sqrt{\epsilon_{\text{tag}}}$$

- Performance with this algorithm:

- Belle II: $\epsilon_{\text{tag}} = 31.7\%$

- Belle: $\epsilon_{\text{tag}} \sim 30\%$



Evolution 1: Graph Neural Network

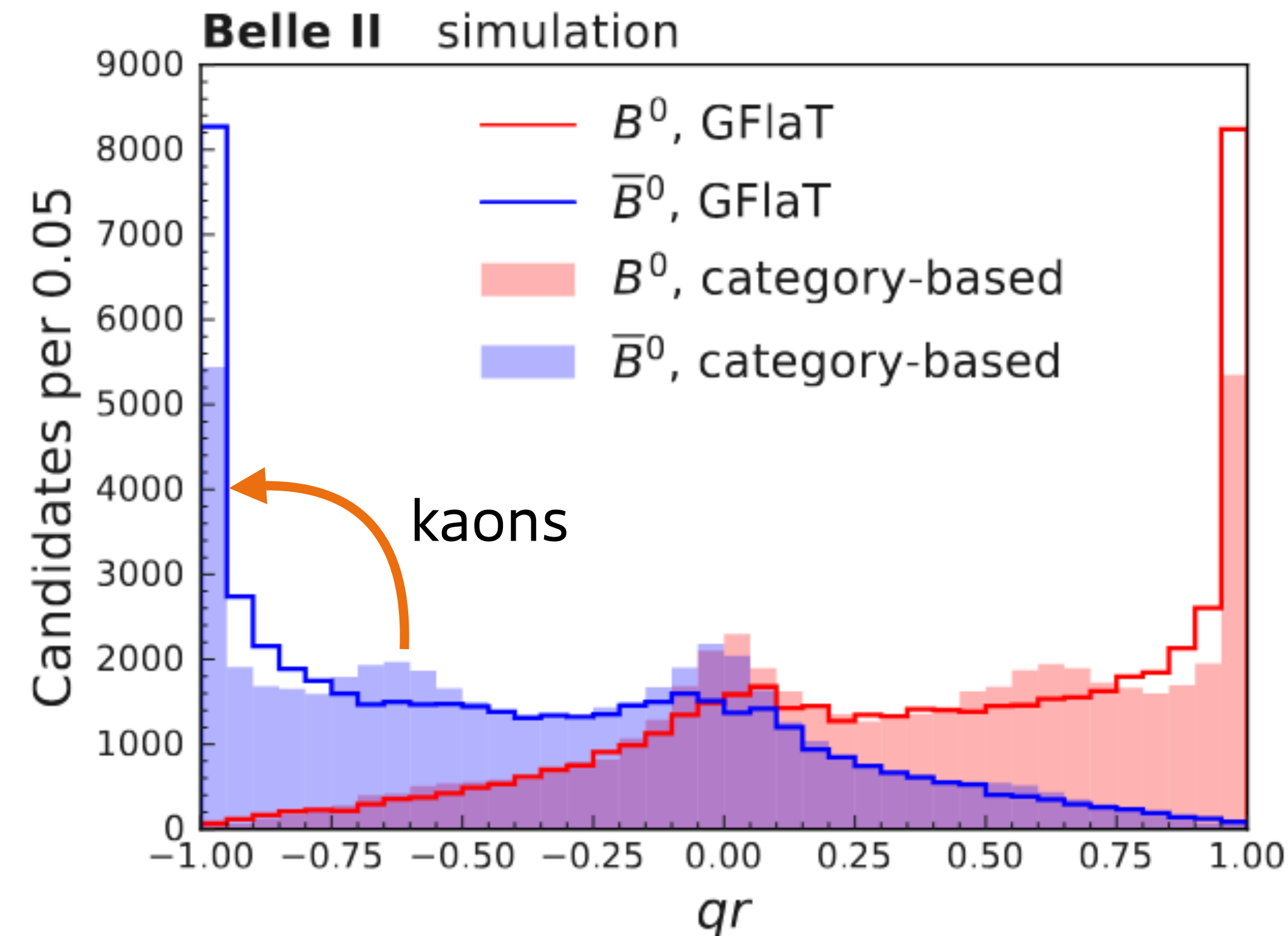
Instead of only retaining the best output of each category, builds a "particle cloud", or graph, with all charged tracks from the tag B decay

→ **Graph flavour tagger GFlat**

- Nodes:
 - 13 category outputs
 - 3-momenta
- Edges: distance between the tracks

Significant gain wrt category BDT: $\epsilon_{\text{tag}} = 38.4\%$ (data)

- gain from retaining the information of how the category output is shared among all tracks
- especially useful with events with multiple kaons
- Several analyses published with this, eg. [sin2beta](#)



New approach: low level transformer

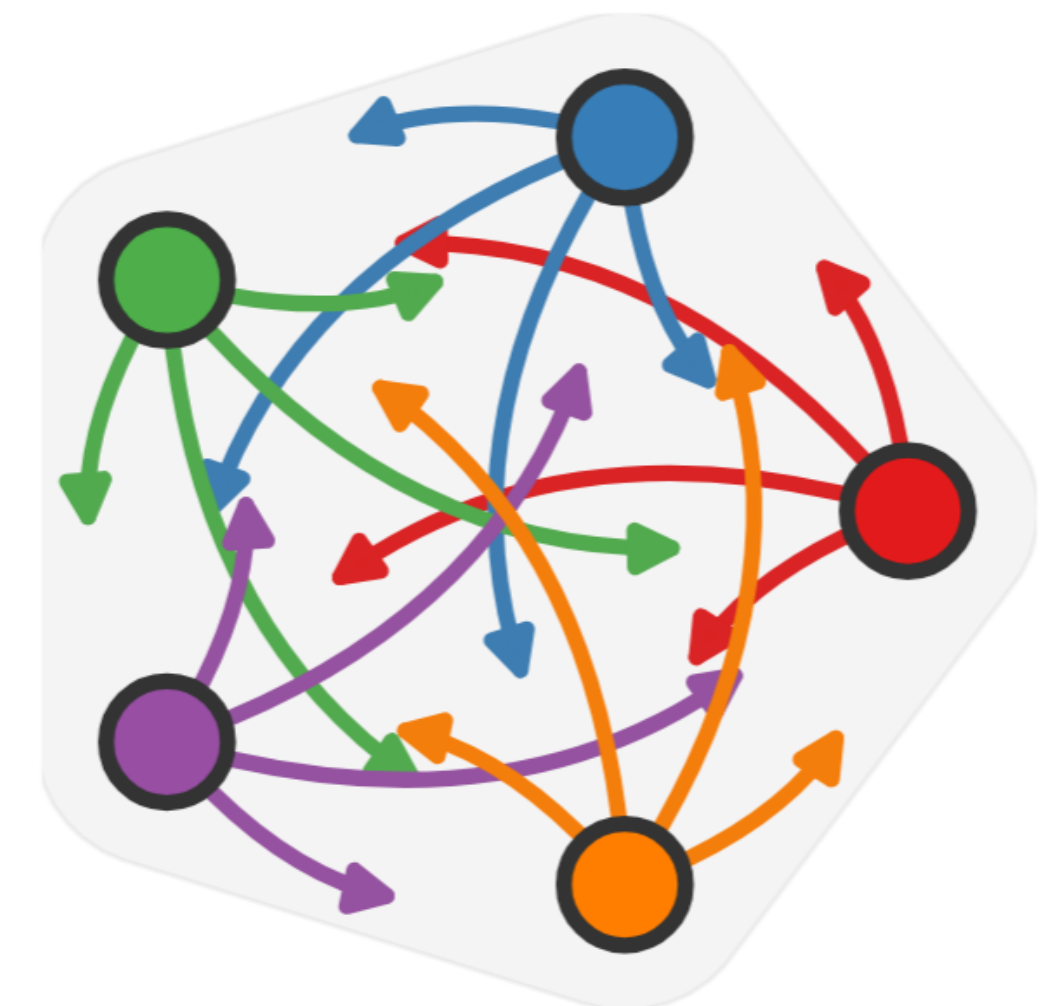
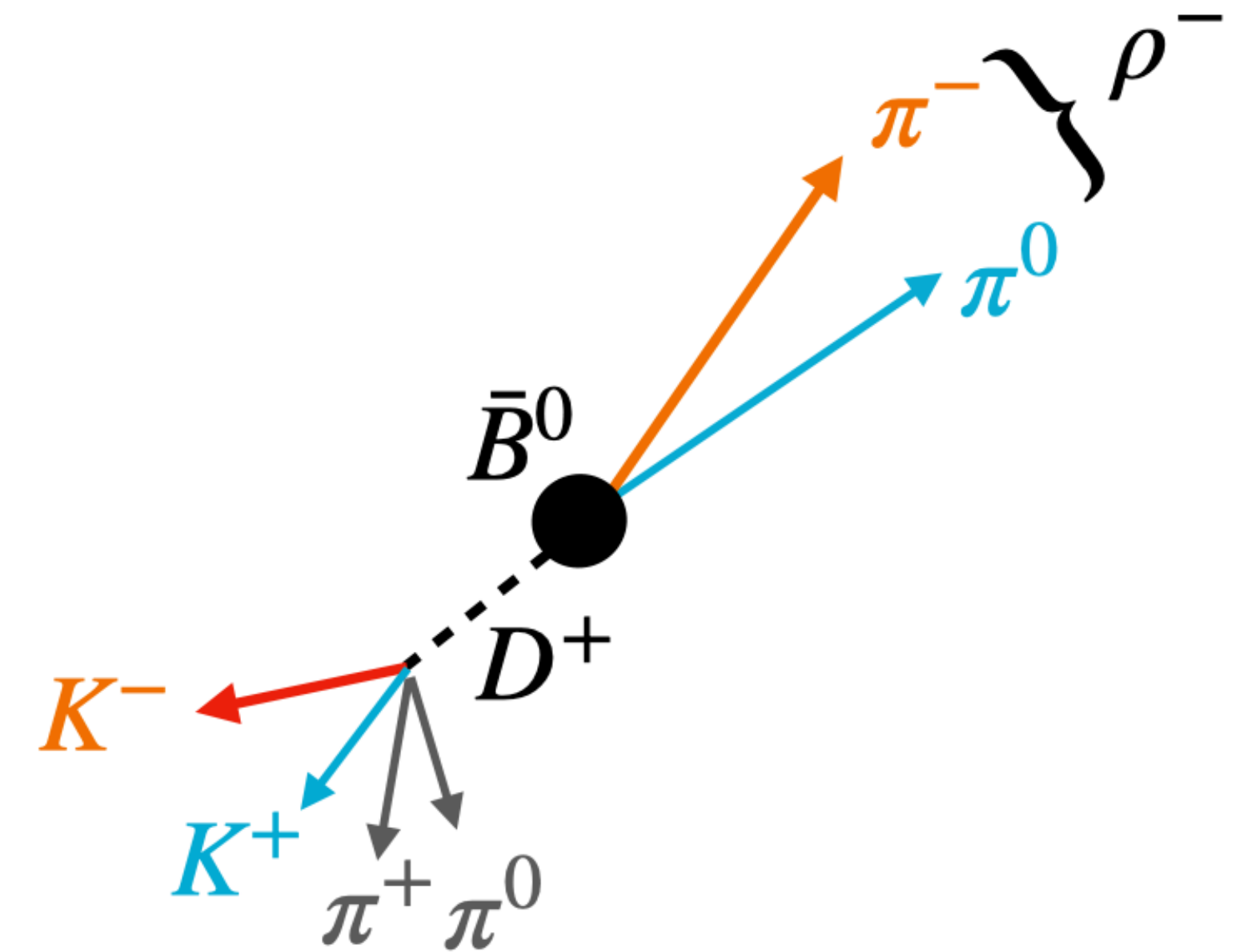
Extra π^+ and π^0 s may not directly carry flavour information, but might help identifying the topology of the decay:

- e.g. identifying a fast ρ^- recoiling against a D^+

This motivated a new approach:

- End to end flavour tagger, using only low level track- or cluster-level information
→ no more categories
- Use transformer algorithm, to learn inter-particle relationships directly from data

→ **Transformer flavour tagger TFlat**



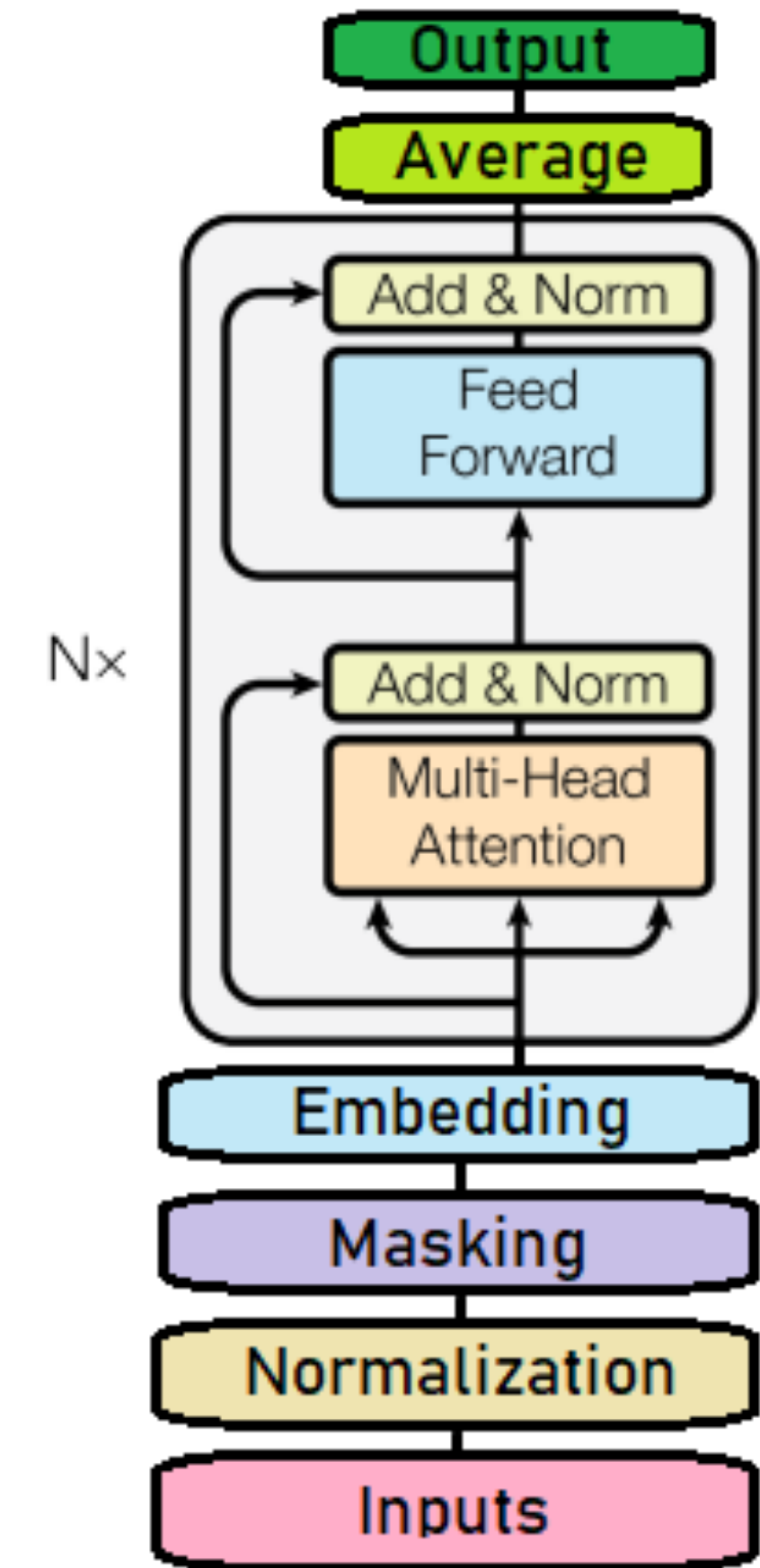
Transformers

Transformers are a type of ML algorithm well adapted for this task, as they can model complex relations/dependencies in the data

- Widely used in language processing and b-jet tagging at the LHC

In detail:

- Inputs are embedded into tokens. 1 token per track/cluster.
- Tokens pass through alternating Multi-head attention and MLP layers
- Tokens are averaged into single vector
- Final MLP projects output vector to $q \cdot r$



Tracks				Clusters				ROE
			event 1					
			event 2					
			event 3					

 Token

Inputs (for reference)

Modality	Discriminating input variable
Charged tracks	$Q, p^*, \cos \theta^*, \phi^*, \mathcal{L}_e, \mathcal{L}_\mu, \mathcal{L}_K, \mathcal{L}_\pi, \mathcal{L}_p, \mathcal{L}_{K \pi}^{\text{CDC}}, \mathcal{L}_{K \pi}^{\text{TOP}},$ $\mathcal{L}_{K \pi}^{\text{ARICH}}, \mathcal{L}_{e \pi}^{\text{TOP}}, \mathcal{L}_{e \pi}^{\text{ARICH}}, \mathcal{L}_{e \pi}^{\text{ECL}}, \mathcal{L}_{\mu \pi}^{\text{TOP}}, \mathcal{L}_{\mu \pi}^{\text{ARICH}}, \mathcal{L}_{\mu \pi}^{\text{KLM}},$ $\mathcal{L}_{\pi K}^{\text{TOP}}, \mathcal{L}_{\pi K}^{\text{ARICH}}, N_{\text{PXD}}/2, N_{\text{SVD}}/8, dx - dx', dy - dy',$ $dz - dz', E/p, S_{\text{Lat}}$
Photon clusters	$p^*, \cos \theta^*, \phi^*, \text{clusterE1E9}, \text{clusterE9E21}, S_{\text{Lat}}$
Global attributes	$N_{\text{photons}}^{\text{roe}}/8, N_{\text{trk}}^{\text{roe}}/6, N_{K_S}^{\text{roe}}, \sum_{\text{roe}} p_t$

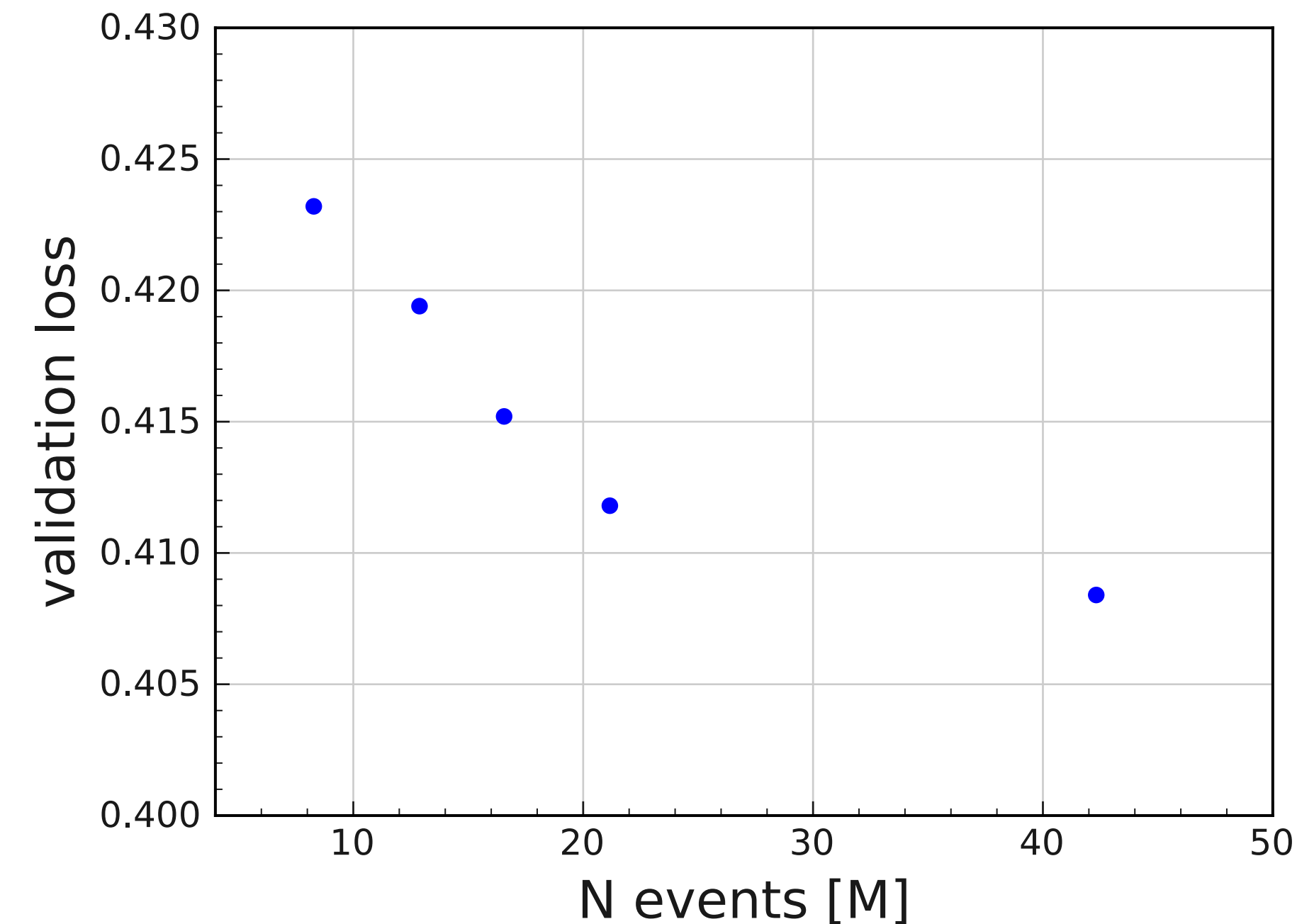
Training procedure

At Belle II, the flavour tagger only uses information from the tag B :

- performance independent of the B_{sig} decay
- For simplicity, use $B_{\text{sig}} \rightarrow \text{invisible}$ to train

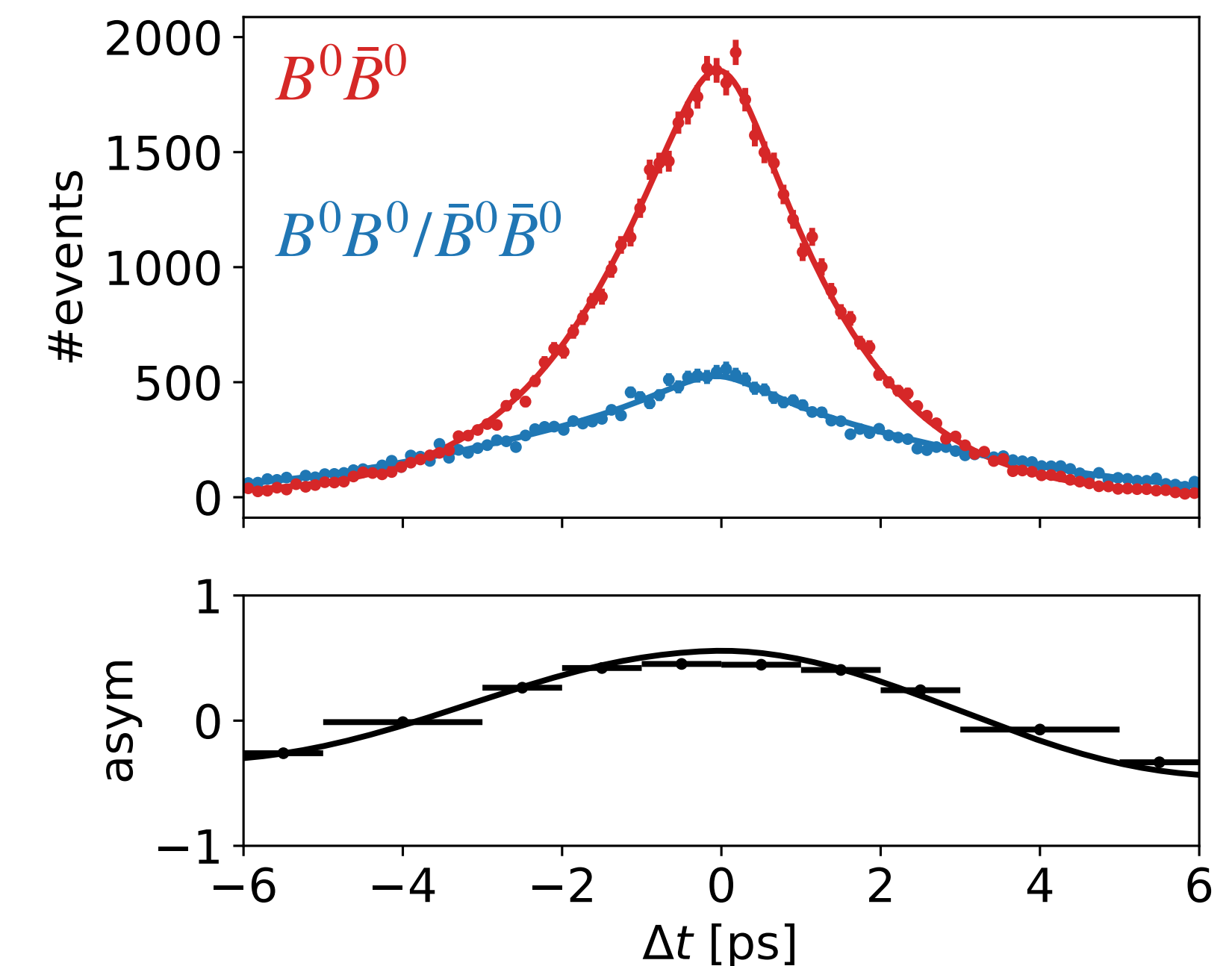
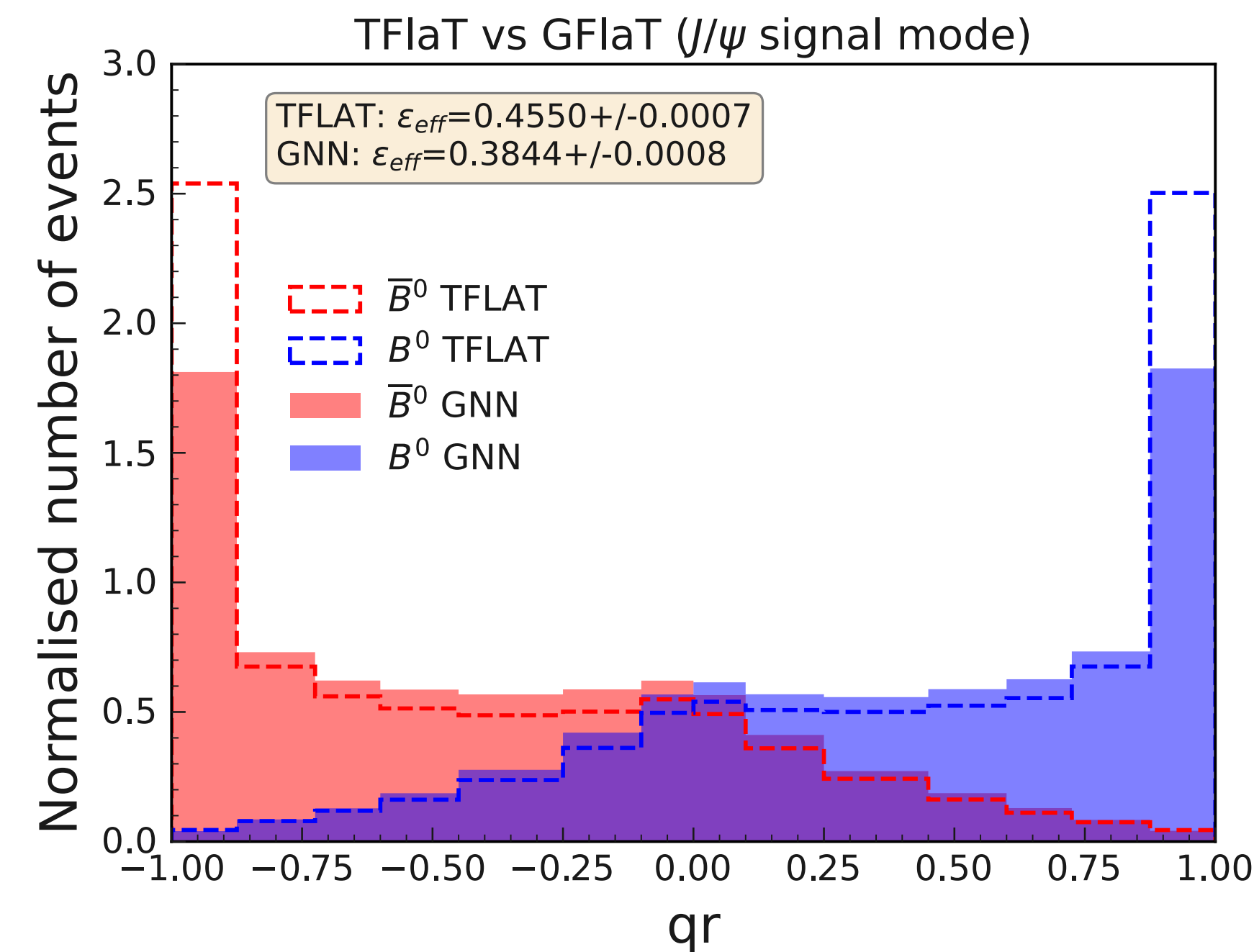
The transformers model is comparatively large, ~ 2 Mio parameters vs ~ 250 k for GFlat (but still smaller than LHC jet tagger)

- Training takes 4 days on a NVIDIA A100 GPU
- Performance significantly improves with sample size
→ use training sample with 50 Mio events
- We did not try to scale up the complexity of the model.
Maybe in the future...



TFlat: Performance

- Significant gain in performance wrt GFlat
- Gain confirmed in data, where we used TFlat to measure B^0/\bar{B}^0 mixing with $B^0 \rightarrow D^{(*)-}\pi^+$
 - $\epsilon_{\text{tag}} = 44.4\%$ (data)
 - Almost 50% relative improvement wrt "traditional" tagger
- Very good data/MC agreement (so training on MC is fine)
- Not yet used in published analysis (brand new...) but we plan to
- Now trying it on "old" Belle data: preliminary results also show a clear improvement



Where does the improvement come from?

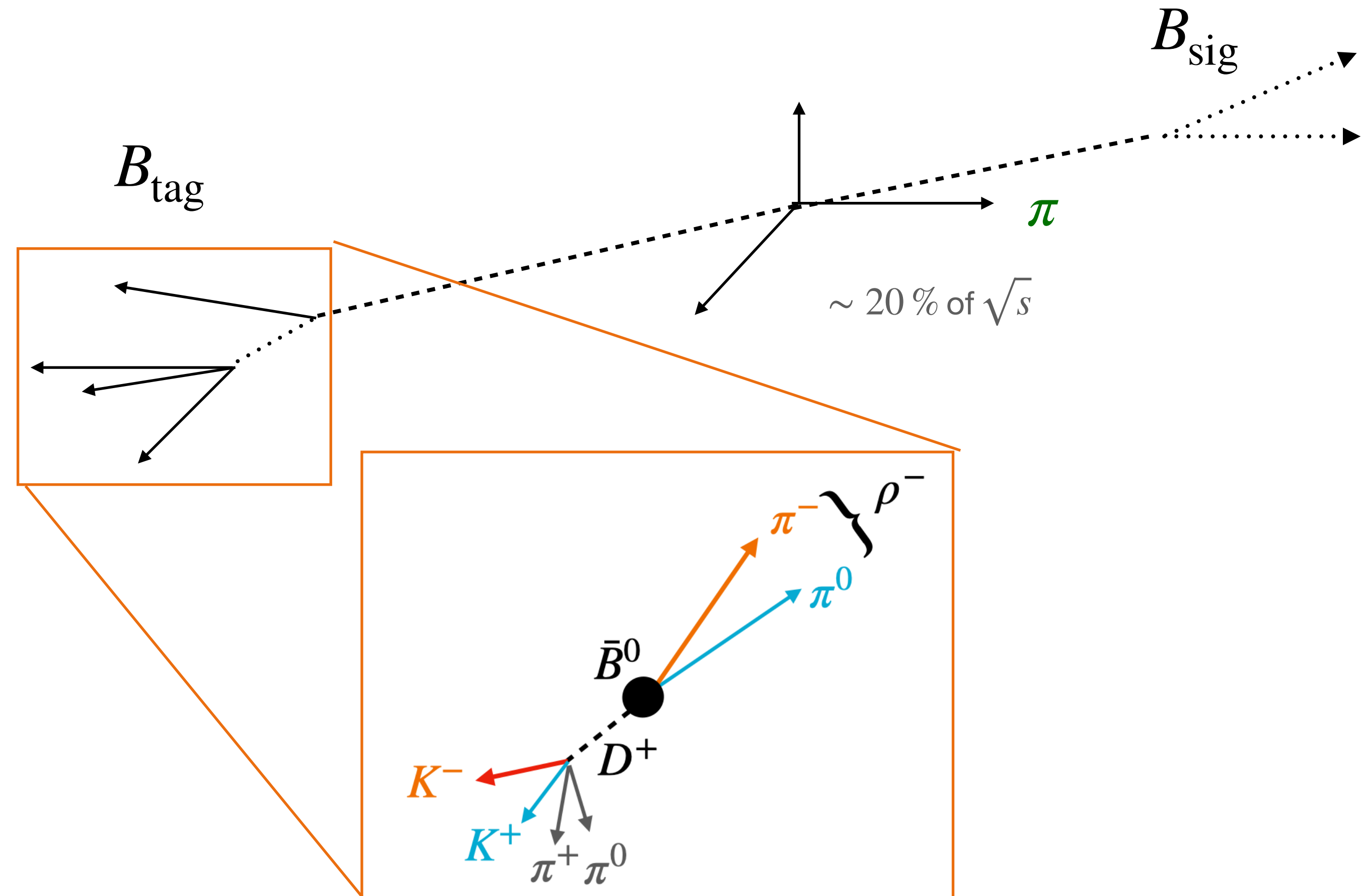
B_{tag} decay	ϵ_{tag} GFlat	ϵ_{tag} TFlat	improvement
$B \rightarrow D\ell X$	75%	82%	+9%
$B \rightarrow D\pi X$	30%	36%	+20%
$B \rightarrow DDX$	22%	31%	+41%

Seems clear that TFlat handles better B_{tag} categories with complex topologies, with less clear tagging signatures

TFlat at FCC-ee?

What if we applied TFlat on FCC-ee MC?

- What we lose:
 - B^0/\bar{B}^0 not entangled: dilution from mixing
 $\sim 5\%$ of initial b become B^0 that mixes
 - B_s^0 unusable for tagging
 $\sim 10\%$ of initial b
- What we gain:
 - Expect higher tagging performance for B_{tag}^+ , Λ_b , and B_{tag}^0 due to much better vertex resolution
 - Same-side tagging: prompt π^\pm/K^\pm flying close to thrust axis might bring extra information



→ My wild guess is that a TFlat-like flavour tagger at FCC-ee would work with a very similar performance as what we have at Belle II, much better than LEP's $\epsilon_{\text{tag}} \sim 20\%$

TFlat at FCC-ee? practicalities

- Train with $B_{\text{sig}} \rightarrow$ invisible:
 - probably tagging response pretty independent on B_{sig} decay mode
 - good to train a flavour tagger that works with decays with missing energy
- Maybe need to add input variables for higher multiplicity due to fragmentation system
 - Similar input variables for charged tracks: momentum, PID, impact parameters
 - Similar input variables for clusters, here the performance might depend a lot on the calo
 - Include HCAL clusters?
- TFlat code uses keras with pytorch backend, and can easily be factorised out of the Belle II software
- We don't really have personpower to help directly but:
 - plan to share code and training dataset on Zenodo, can help with questions/provide code
 - also note you need a big GPU

Summary

- At Belle II, we gained significantly by using state-of-the-art ML techniques for flavour tagging
- ... and there is certainly still margin for improvement
- It is clear that similar techniques can make tagging at FCC-ee very performant
- Could also be interesting to benchmark detectors (PID, calo, vertex) in terms of how good ϵ_{tag} can be
- Happy to help if you want to train TFlat on FCC-ee MC!

Tagger	ϵ_{eff}
FBDT run1	31.7%
GFlat run1	38.5%
TFlat run1	44.4%