Hello Nikhef!

Sébastien Rettie

Nikhef ATLAS Weekly Meeting, 9 May 2025

Student life

Student life

- Born and raised in Gatineau, Québec

Student life

- Born and raised in Gatineau, Québec
- [2010-2014] B.Sc. University of Ottawa

uOttawa

Student life

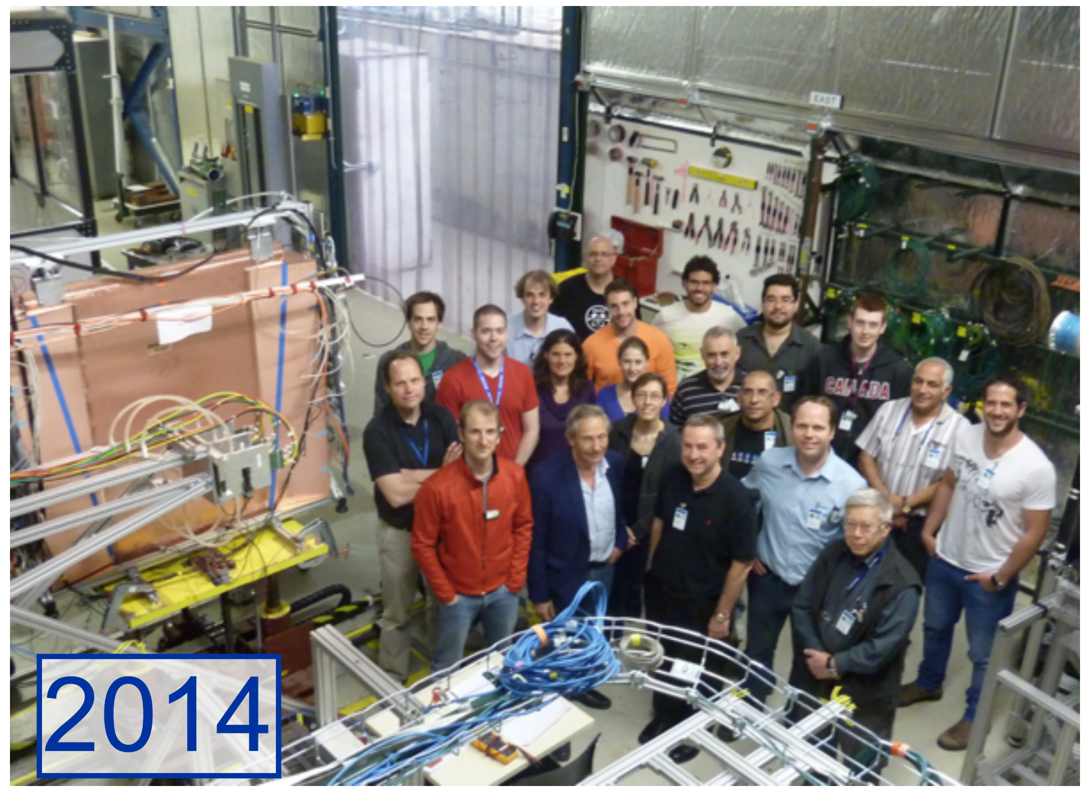
- Born and raised in Gatineau, Québec
- [2010-2014] B.Sc. University of Ottawa
  - CERN summer student during last year

Student life

- Born and raised in Gatineau, Québec
- [2010-2014] B.Sc. University of Ottawa
  - CERN summer student during last year
- [2014-2019] Ph.D. University of British Columbia

Student life

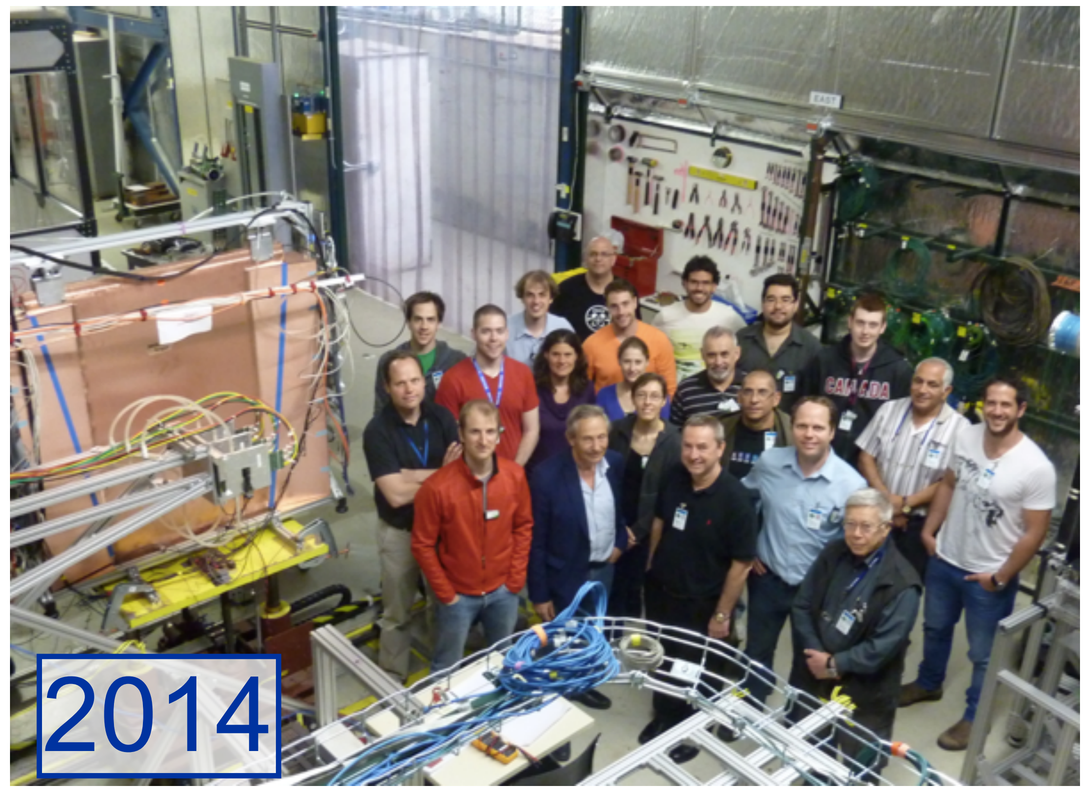
- Born and raised in Gatineau, Québec
- [2010-2014] B.Sc. University of Ottawa
  - CERN summer student during last year
- [2014-2019] Ph.D. University of British Columbia
  - Thesis on Z’ dilepton searches, muon CP and sTGC testbeams



Student life

- Born and raised in Gatineau, Québec
- [2010-2014] B.Sc. University of Ottawa
  - CERN summer student during last year
- [2014-2019] Ph.D. University of British Columbia
  - Thesis on Z’ dilepton searches, muon CP and sTGC testbeams


2013


2014

2018

Postdoc life

Postdoc life

- [2019-2022] Moved to London for postdoc at UCL for 4 months (based at CERN afterwards)

Postdoc life

- [2019-2022] Moved to London for postdoc at UCL for 4 months (based at CERN afterwards)
  - VH(bb) measurements, flavour tagging (GN1 and GN2), industry-related ML projects

Postdoc life

- [2019-2022] Moved to London for postdoc at UCL for 4 months (based at CERN afterwards)
  - VH(bb) measurements, flavour tagging (GN1 and GN2), industry-related ML projects
- [2022-2025] CERN fellow

Postdoc life

- [2019-2022] Moved to London for postdoc at UCL for 4 months (based at CERN afterwards)
  - VH(bb) measurements, flavour tagging (GN1 and GN2), industry-related ML projects
- [2022-2025] CERN fellow
  - Third year funded by Canadian Banting fellowship

Postdoc life

- [2019-2022] Moved to London for postdoc at UCL for 4 months (based at CERN afterwards)
  - VH(bb) measurements, flavour tagging (GN1 and GN2), industry-related ML projects
- [2022-2025] CERN fellow
  - Third year funded by Canadian Banting fellowship
  - Tracking (CTIDE convener), HH searches (HH4b analysis contact)

Postdoc life

- [2019-2022] Moved to London for postdoc at UCL for 4 months (based at CERN afterwards)
  - VH(bb) measurements, flavour tagging (GN1 and GN2), industry-related ML projects
- [2022-2025] CERN fellow
  - Third year funded by Canadian Banting fellowship
  - Tracking (CTIDE convener), HH searches (HH4b analysis contact)
- Drove from CERN to Amsterdam

Personal life

Personal life

- Free time: kayaking, hiking, camping, skiing, anything with mountains

Personal life

- Free time: kayaking, hiking, camping, skiing, anything with mountains
  - This will undoubtedly change now that I’m in the Netherlands 😅

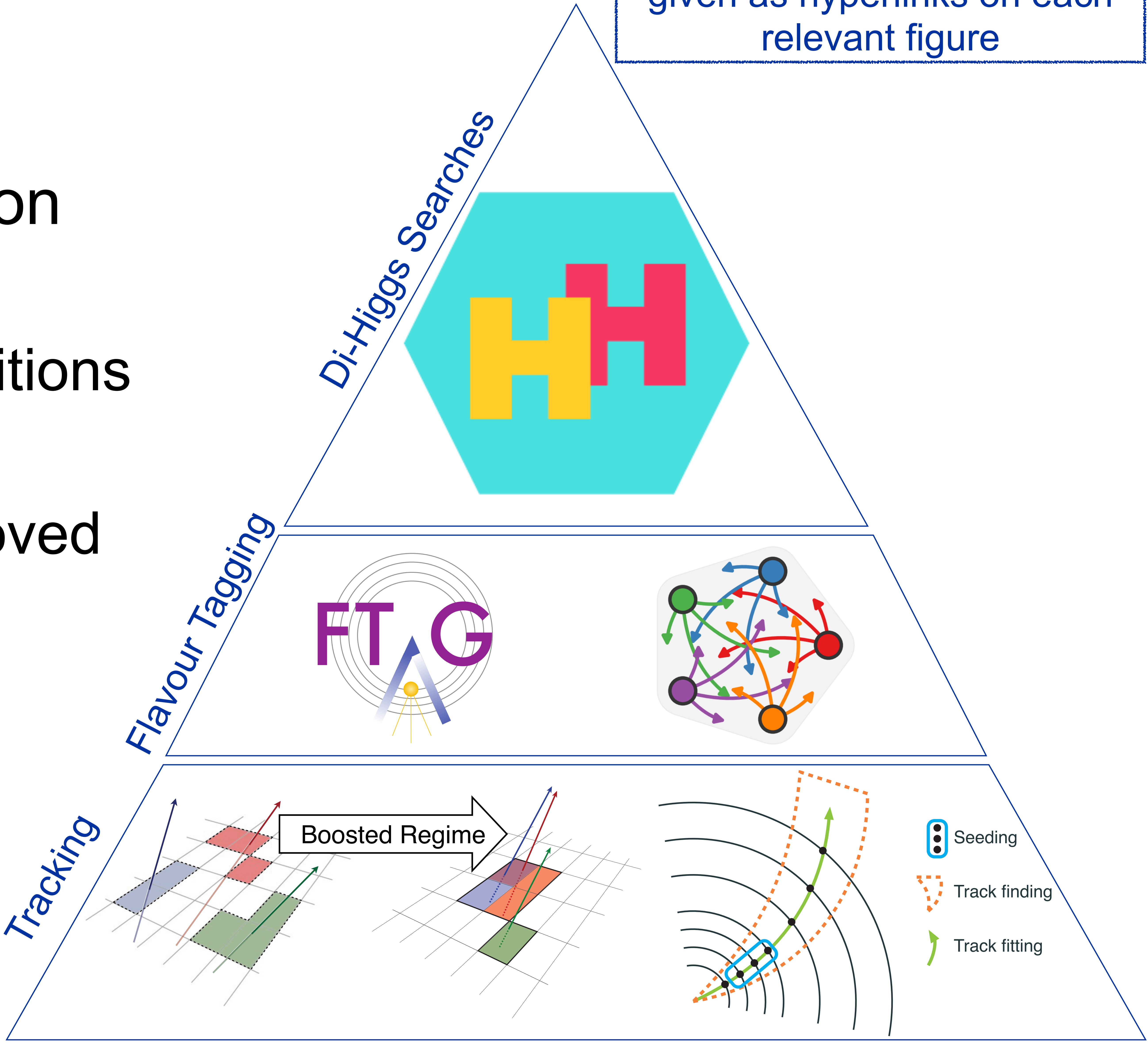
Personal life

- Free time: kayaking, hiking, camping, skiing, anything with mountains
  - This will undoubtedly change now that I'm in the Netherlands 😅
- Married to Megan, Lola joined us in 2022

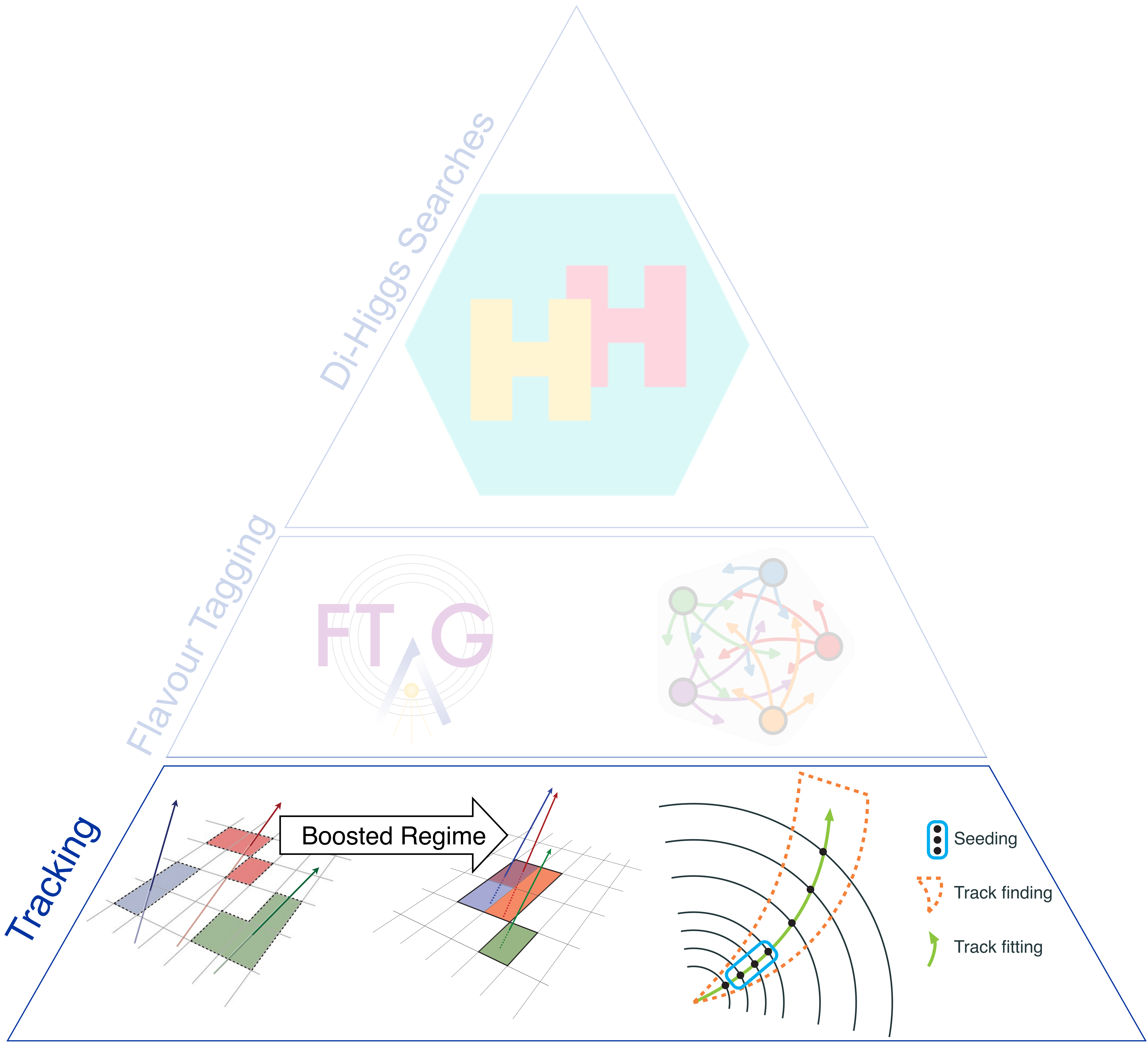
Outline

- Tracking underpins all reconstruction
  - Crucial to ensure continued tracking performance in harsh HL-LHC conditions
  - Clustering and tracking in dense environments (CTIDE) can be improved
- MaskFormer primer
- Two applications of MaskFormer
  - Full-event tracking with the trackML dataset [2411.07149]
  - CTIDE in ATLAS

Note: citations for this talk are given as hyperlinks on each relevant figure



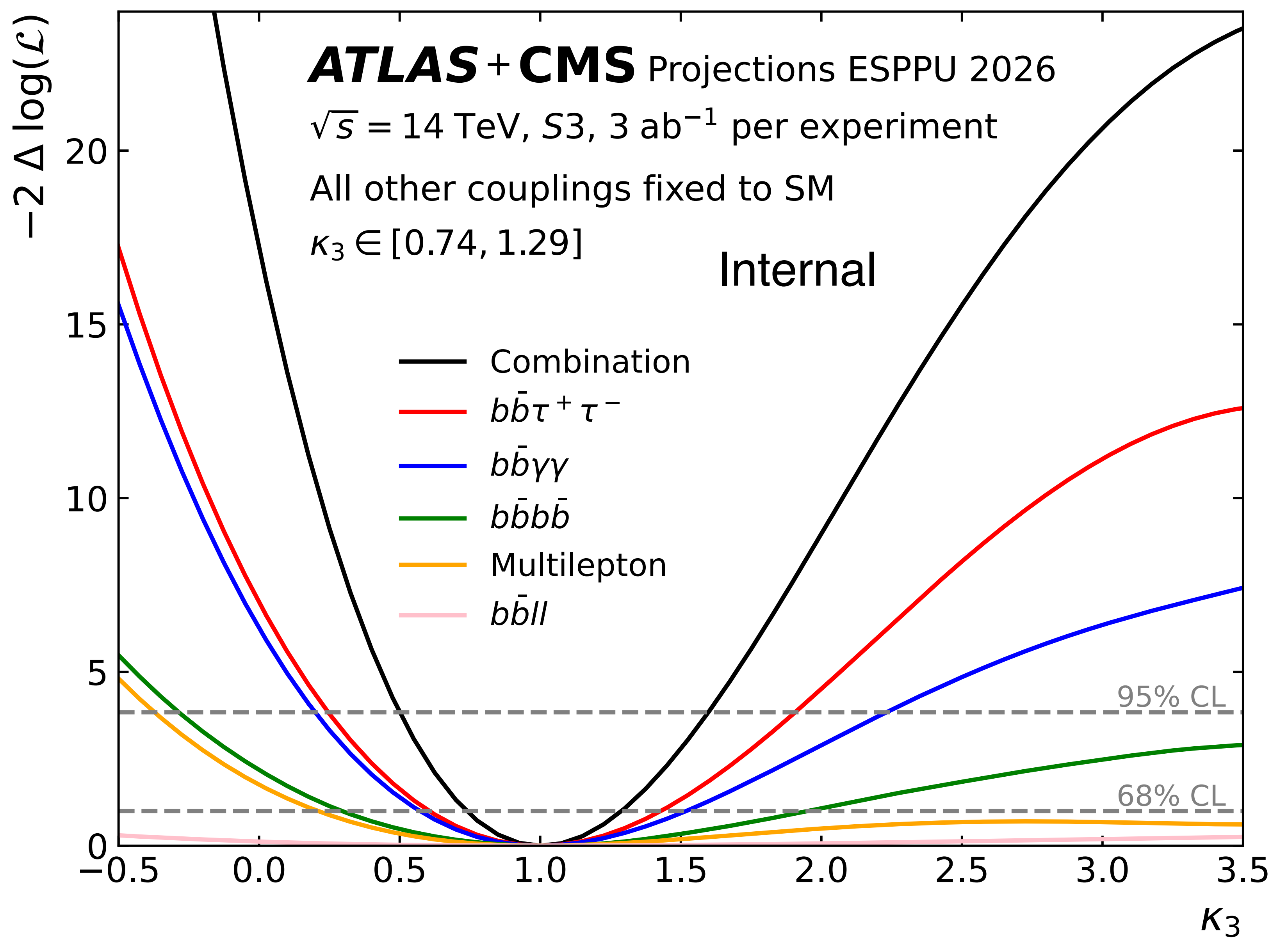
How can we ensure tracking keeps up with the harsh HL-LHC environment?



HL-LHC challenge

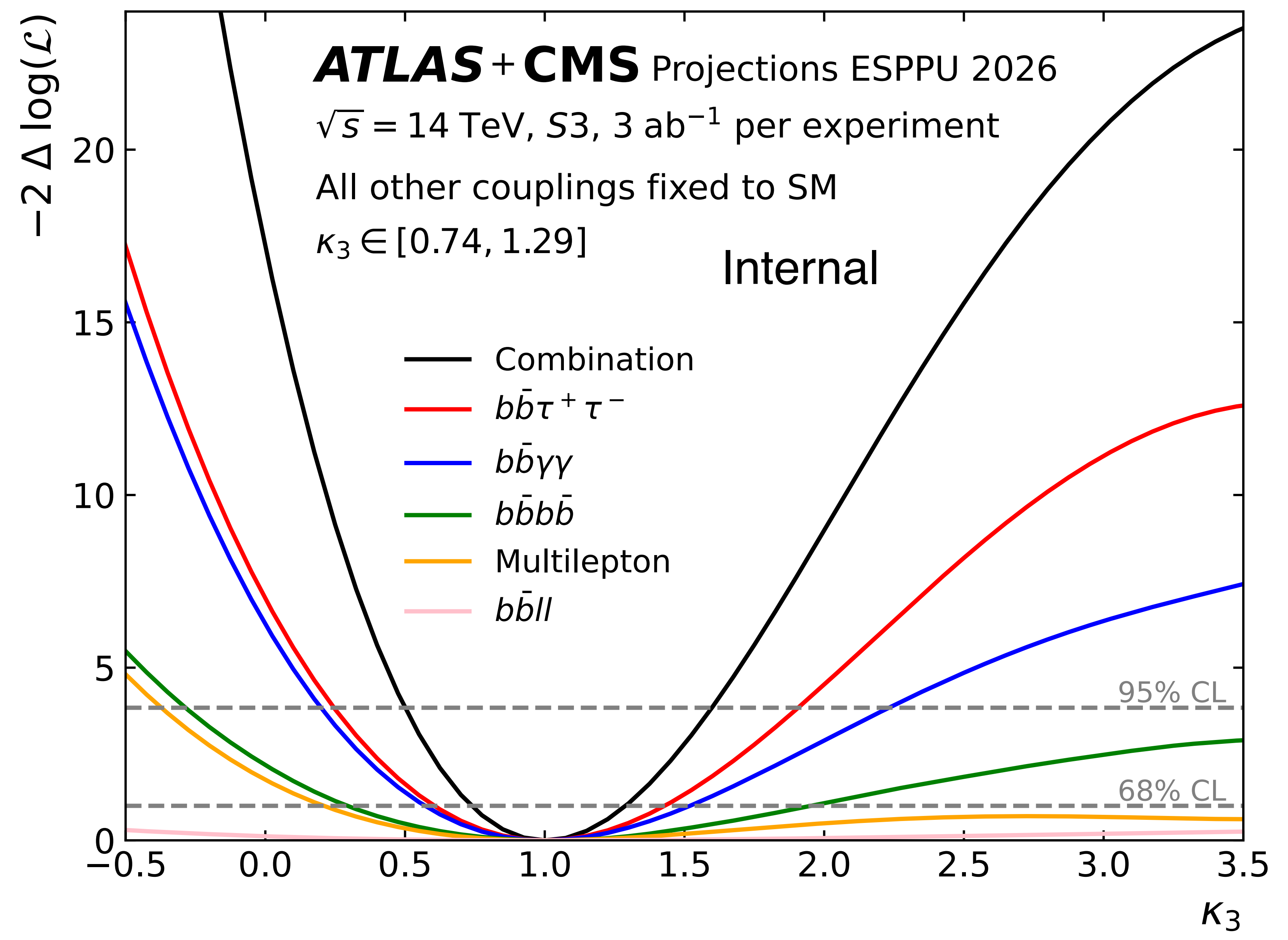
HL-LHC challenge

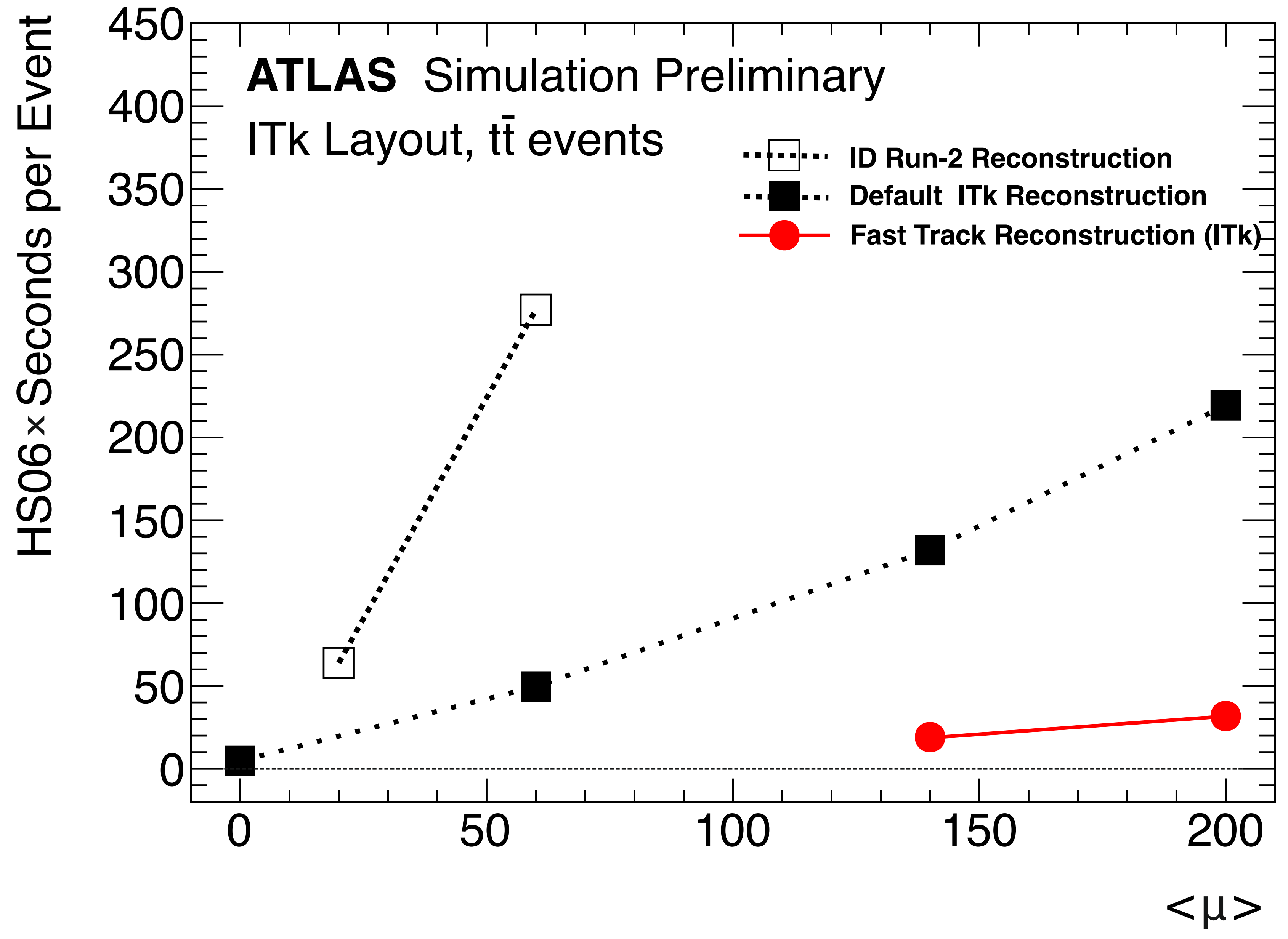
- Recent HL-LHC projections provide promising sensitivity to Higgs self-coupling and beyond, and assume sustained or improved tracking performance; critical to ensure continuity of tracking performance at the HL-LHC!



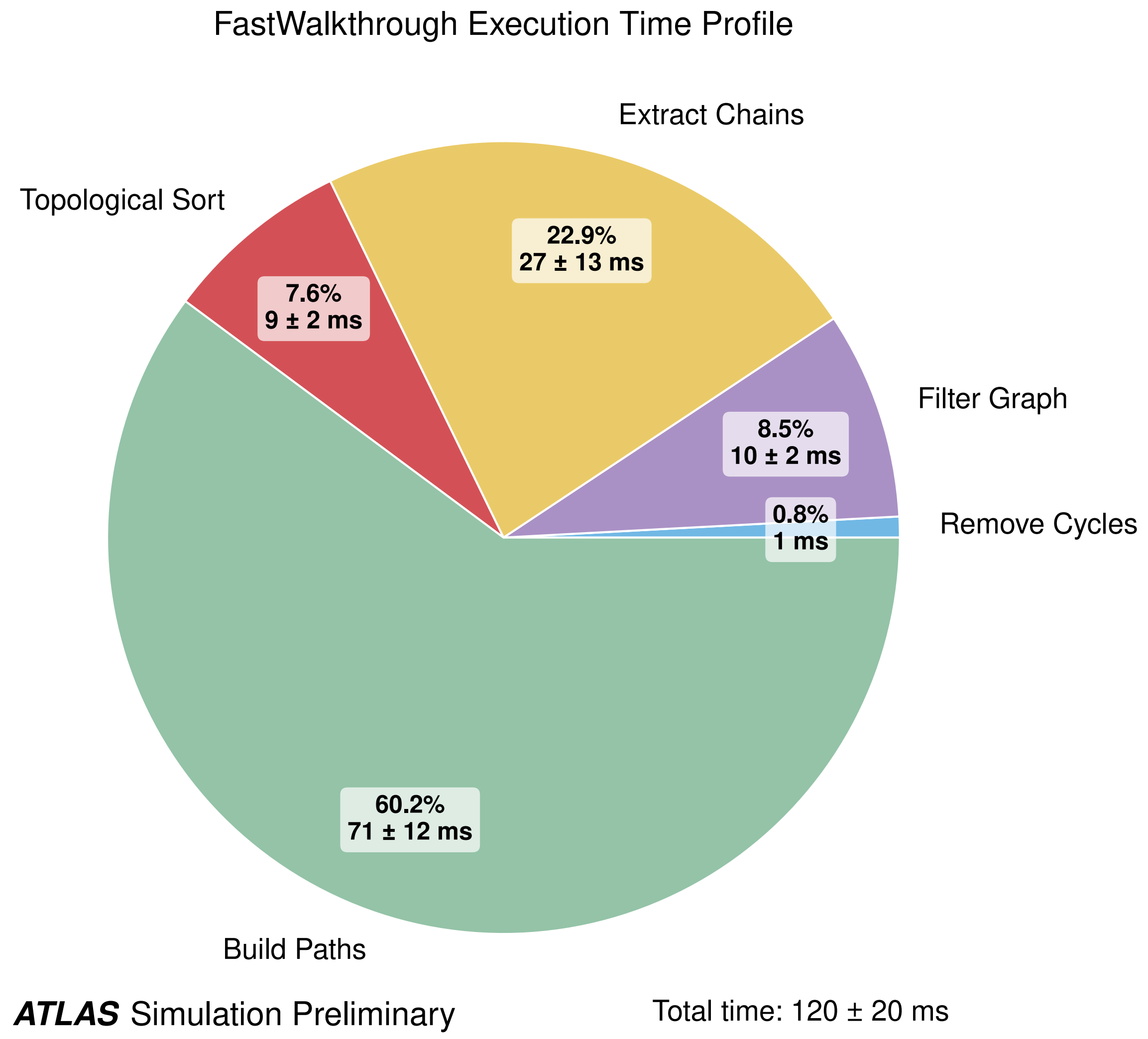
HL-LHC challenge

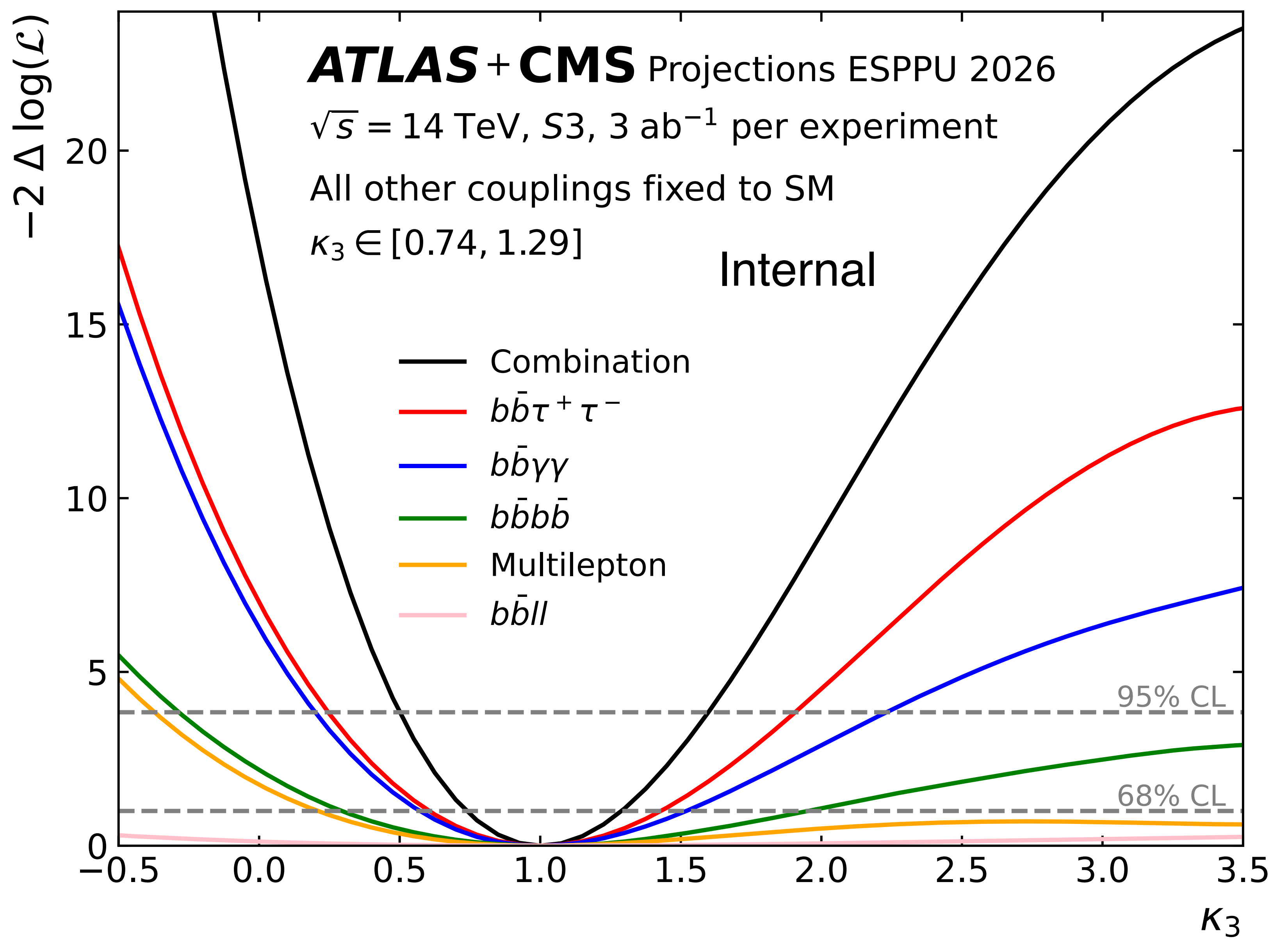
- Recent HL-LHC projections provide promising sensitivity to Higgs self-coupling and beyond, and assume sustained or improved tracking performance; critical to ensure continuity of tracking performance at the HL-LHC!
- Current track finding algorithms, while excellent, are projected to increase dramatically in computational cost at the HL-LHC



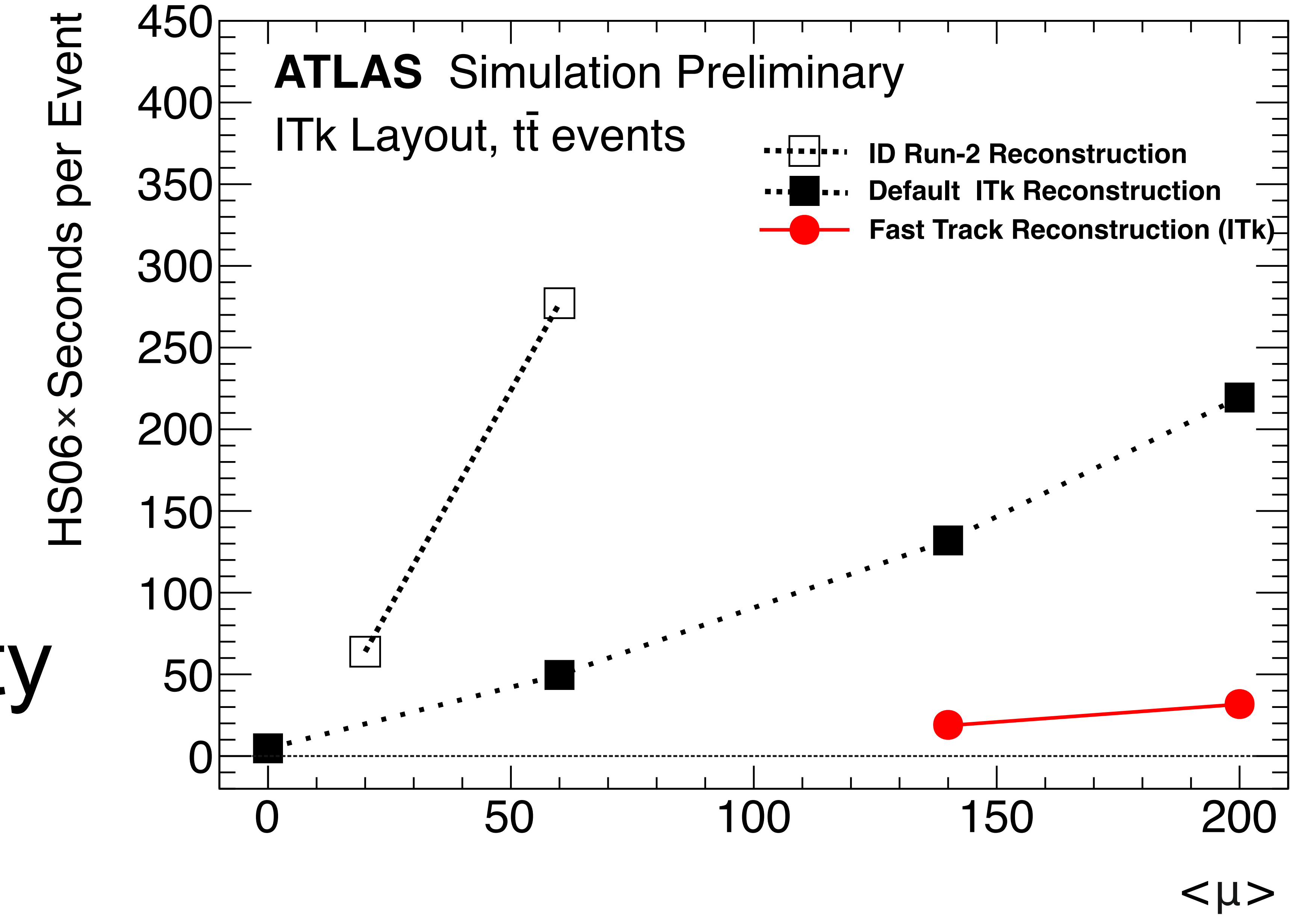


HL-LHC challenge

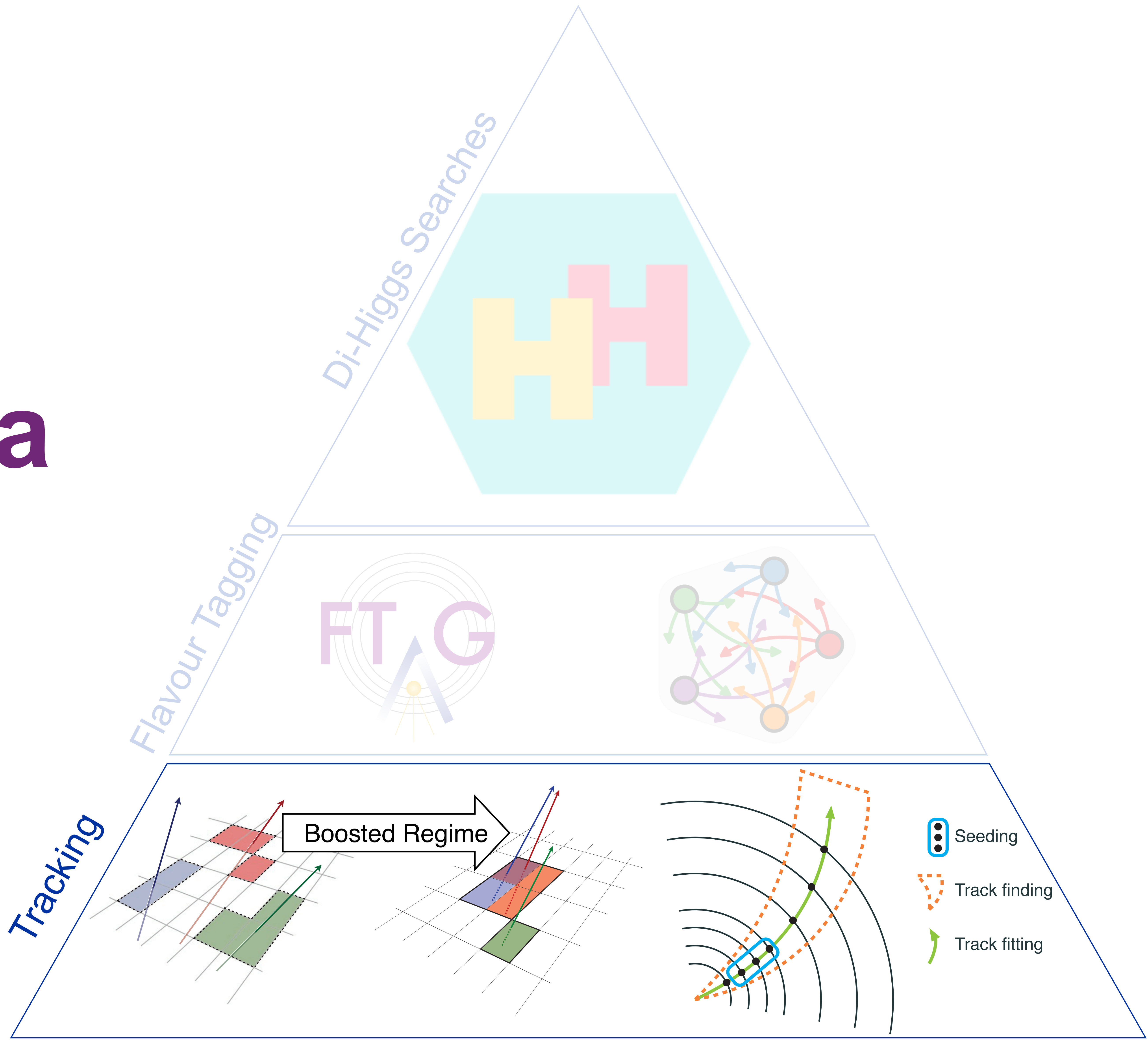




- Recent HL-LHC projections provide promising sensitivity to Higgs self-coupling and beyond, and assume sustained or improved tracking performance; critical to ensure continuity of tracking performance at the HL-LHC!
- Current track finding algorithms, while excellent, are projected to increase dramatically in computational cost at the HL-LHC
- Tremendous work already carried out to ensure continuity of tracking performance in harsh HL-LHC conditions

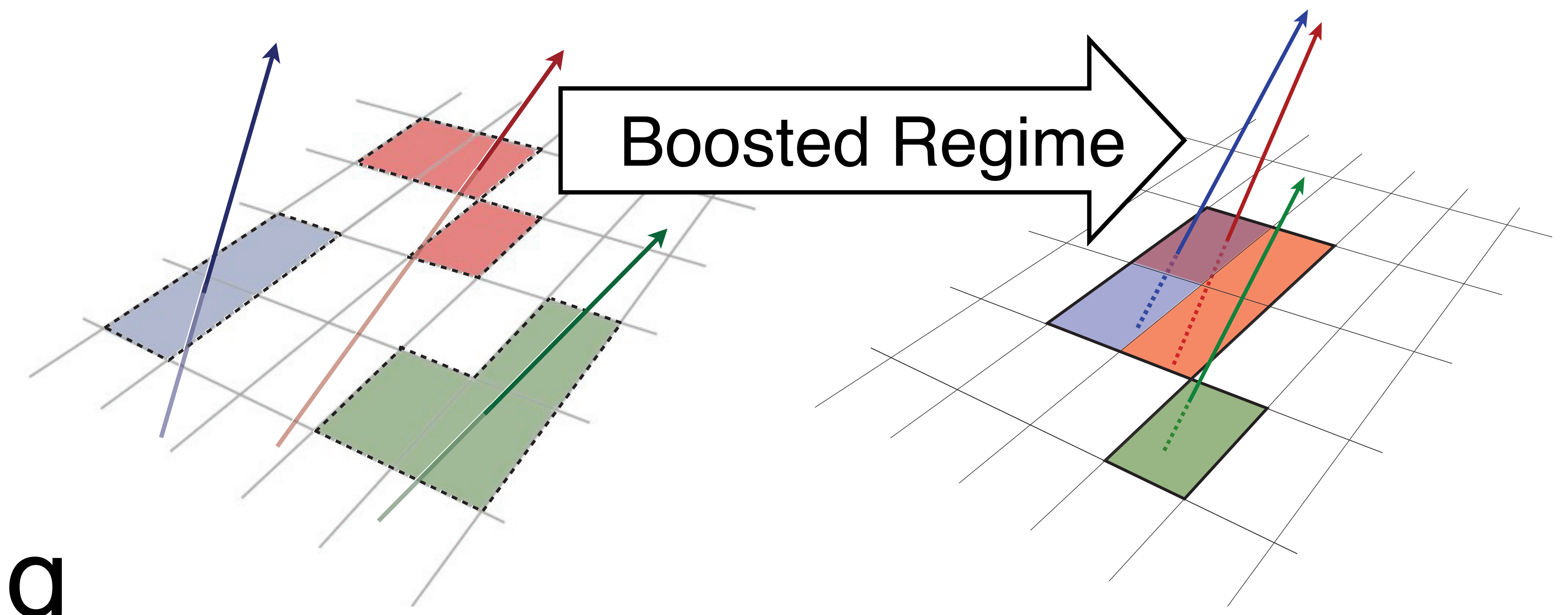


How can we make the most of the data we have now? (including Run 3 data)

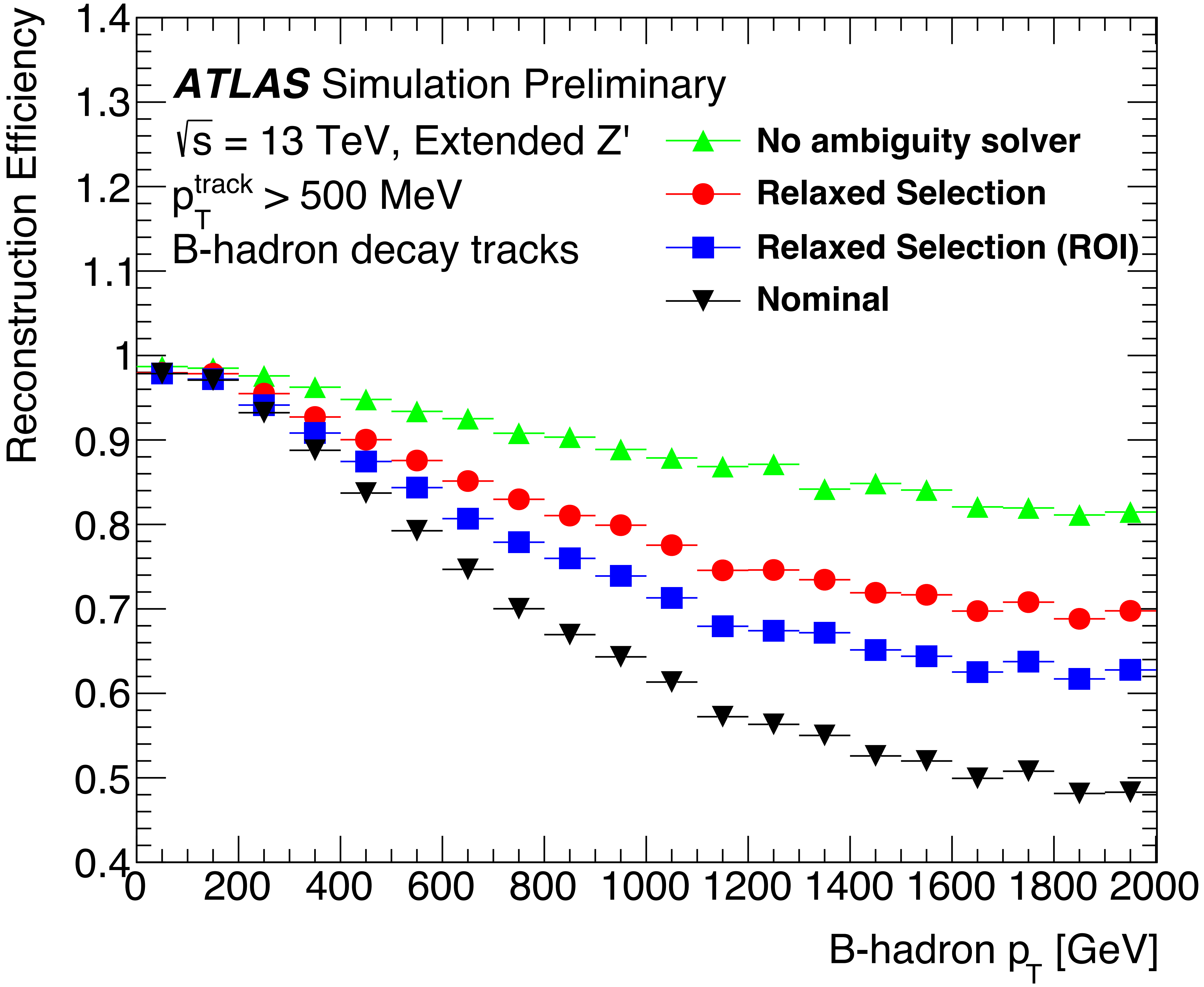


Tracking in dense environments is still not perfect

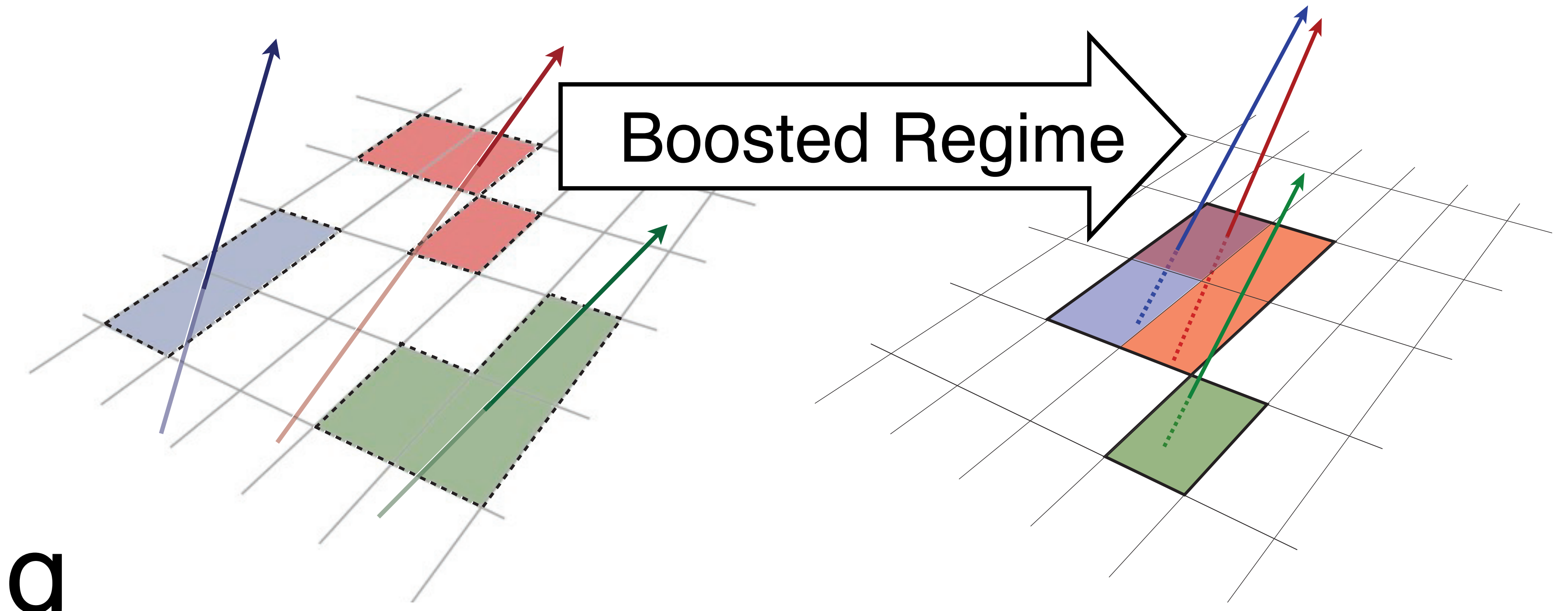
Tracking in dense environments is still not perfect



- Even with latest developments (e.g. pixel cluster splitting MDN, dedicated ambiguity resolution), clear room for improvement

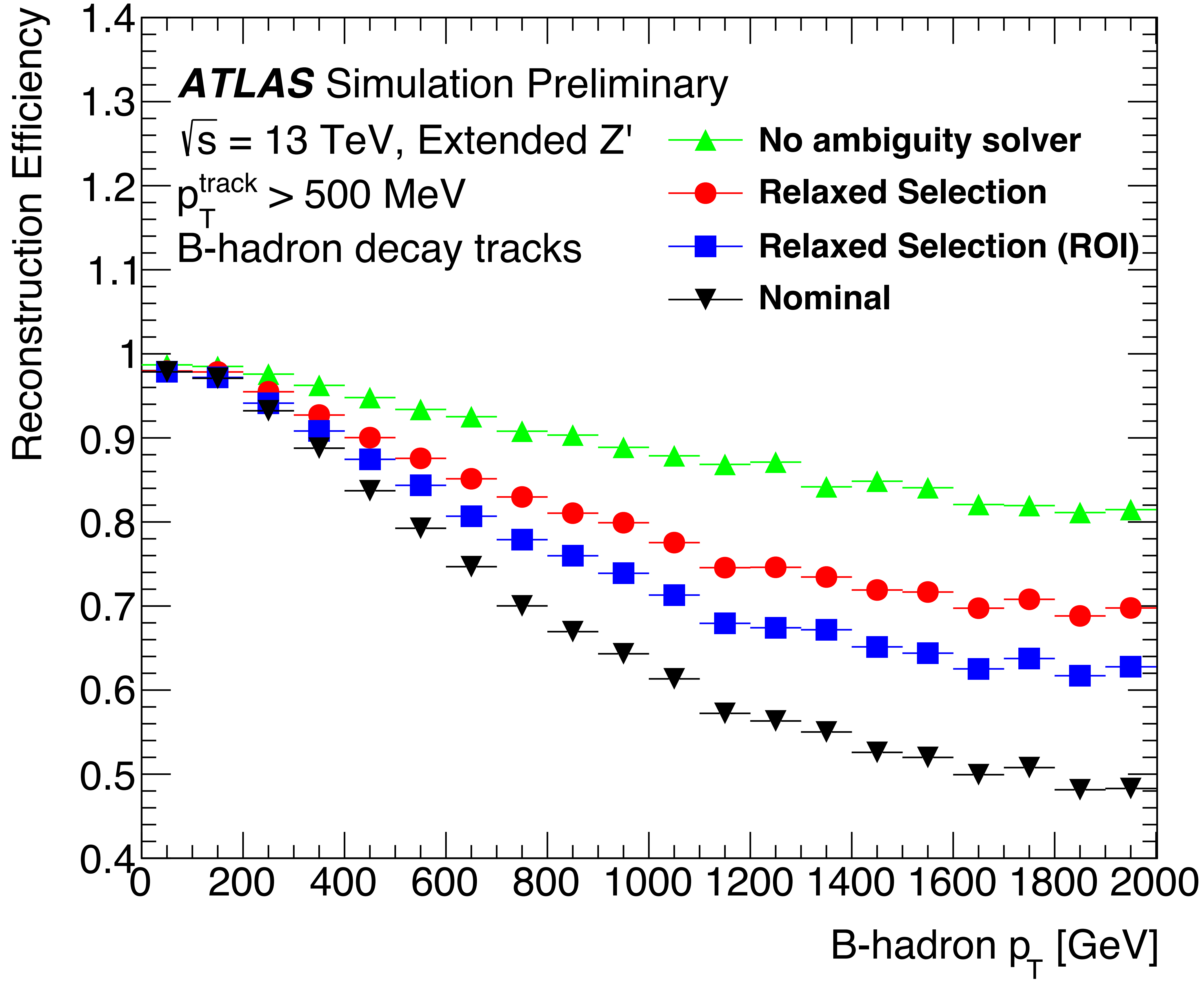


Tracking in dense environments is still not perfect

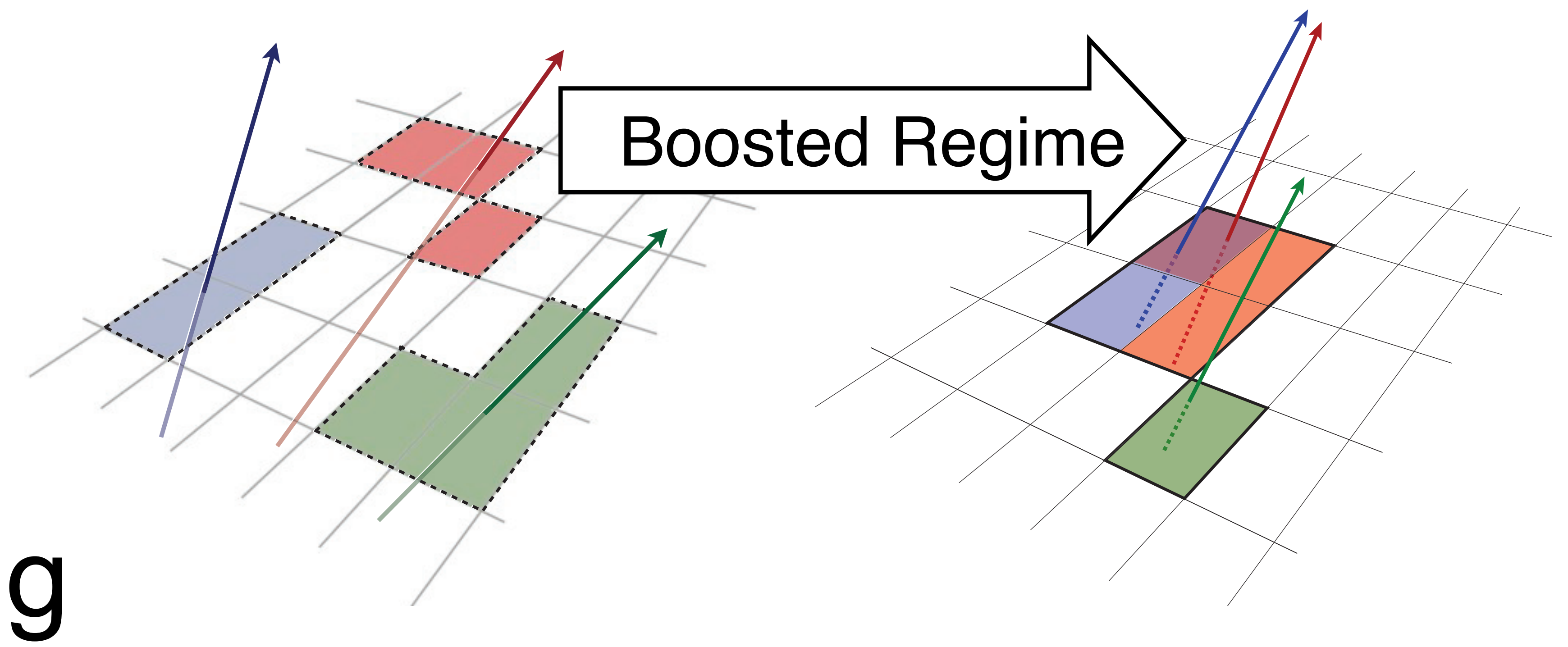


- Even with latest developments (e.g. pixel cluster splitting MDN, dedicated ambiguity resolution), clear room for improvement

- Two main culprits are:

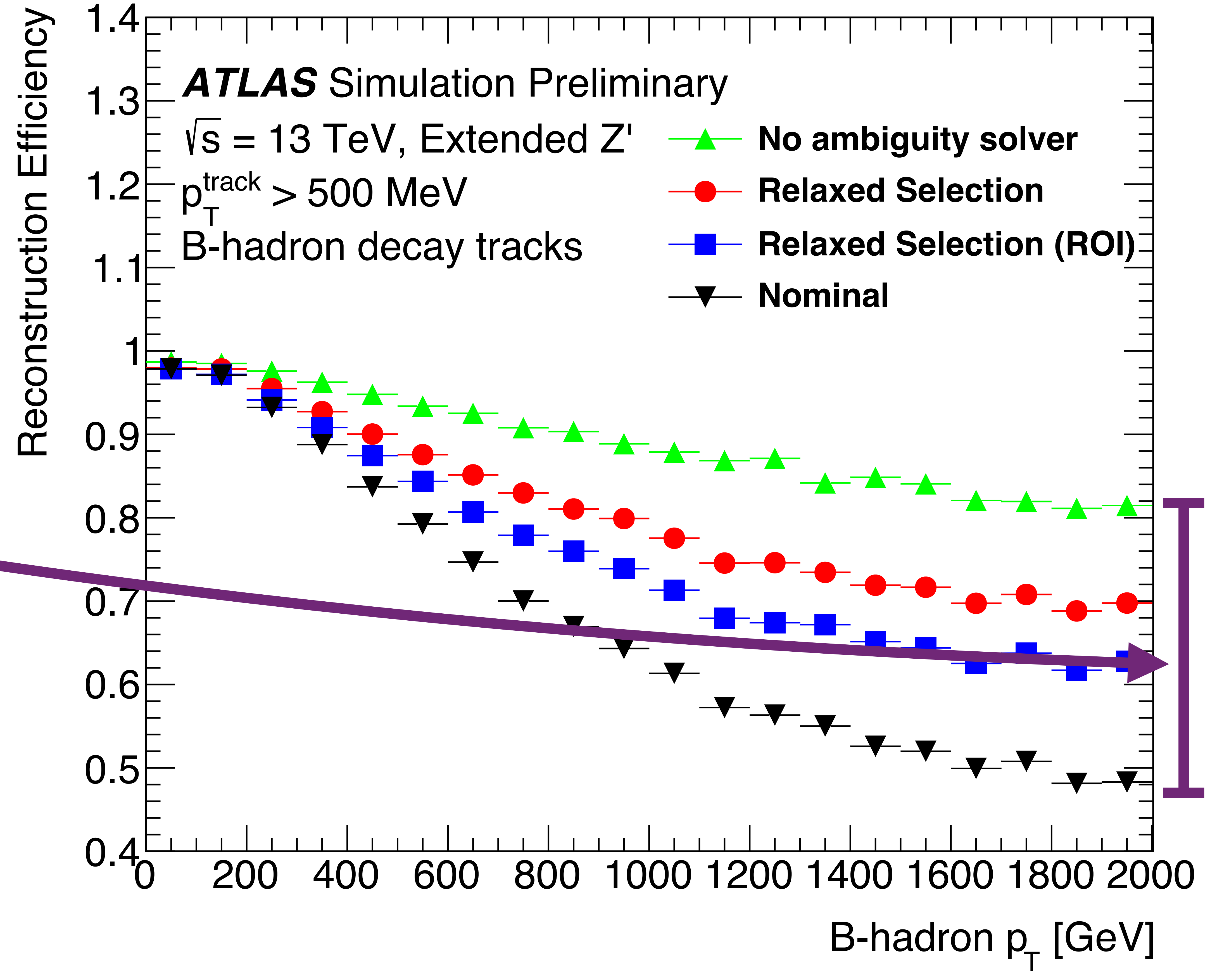


Tracking in dense environments is still not perfect

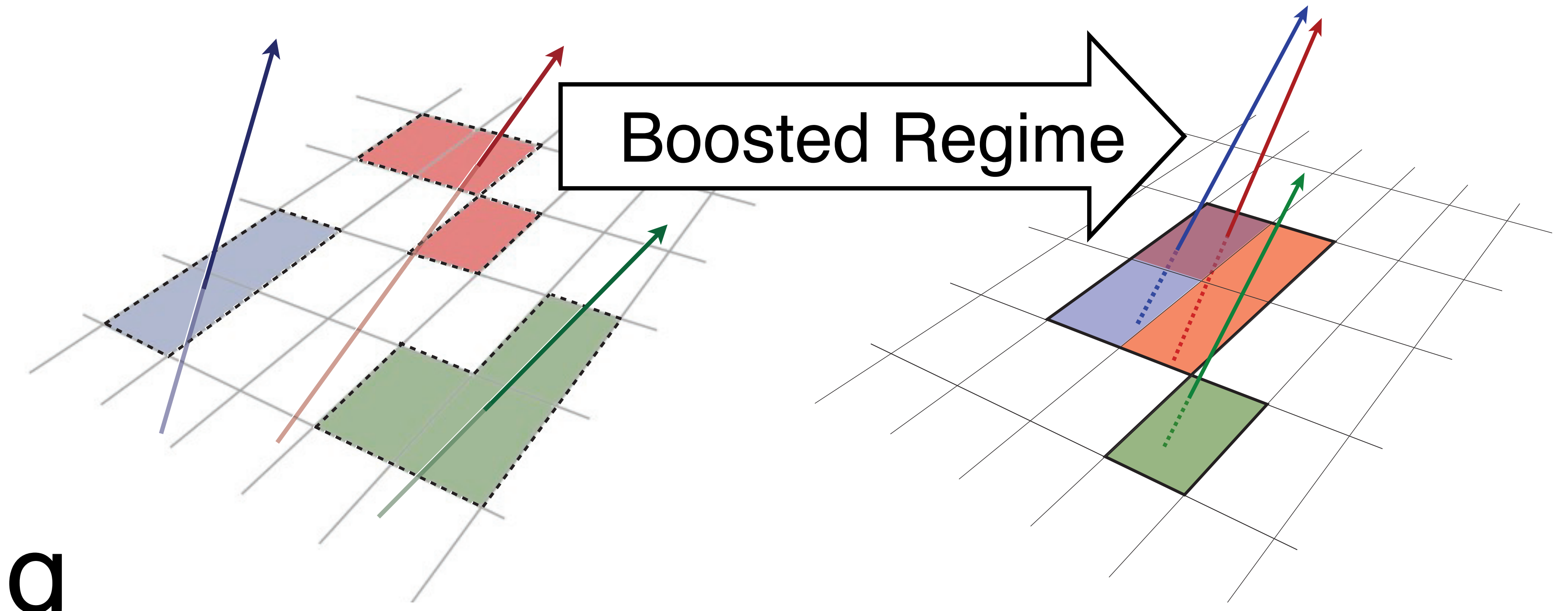


- Even with latest developments (e.g. pixel cluster splitting MDN, dedicated ambiguity resolution), clear room for improvement

- Two main culprits are:
  - Ambiguity resolution

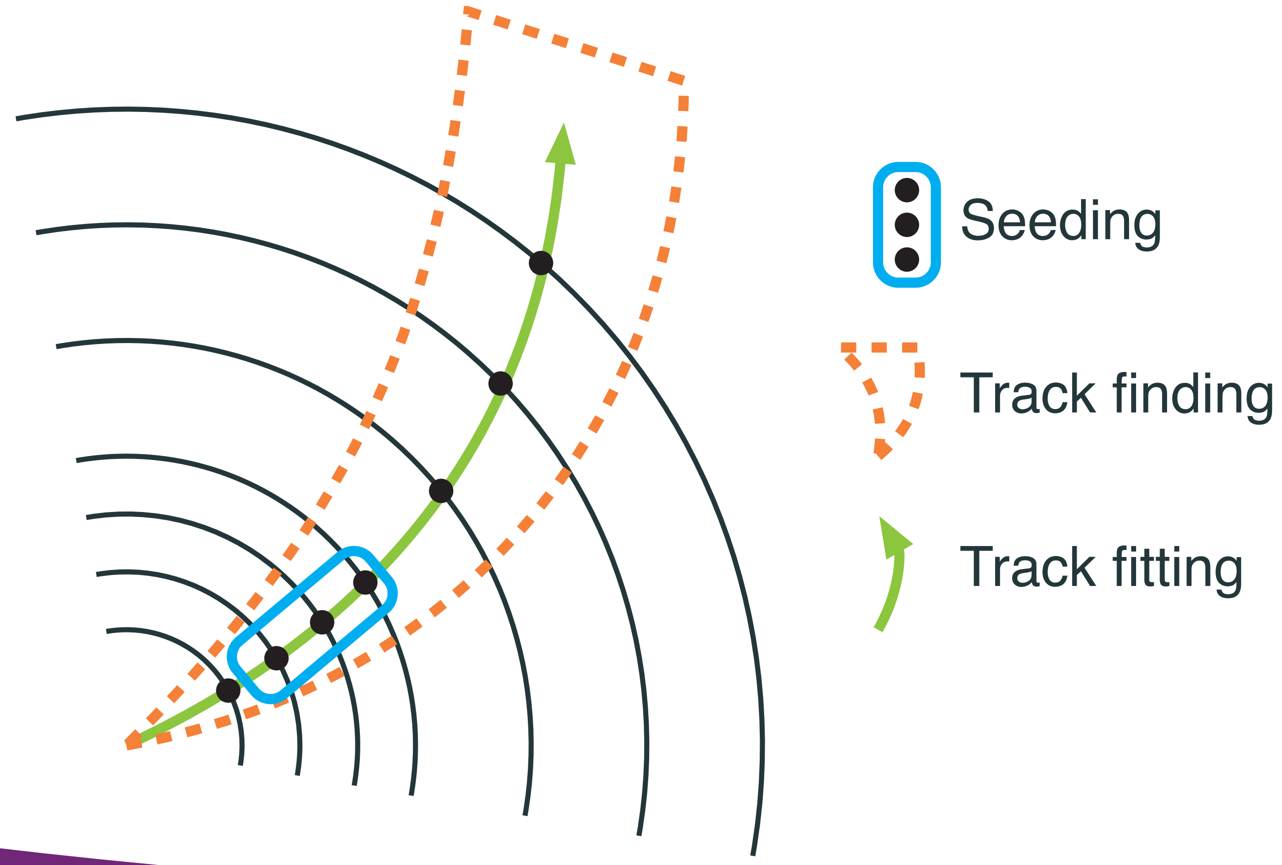


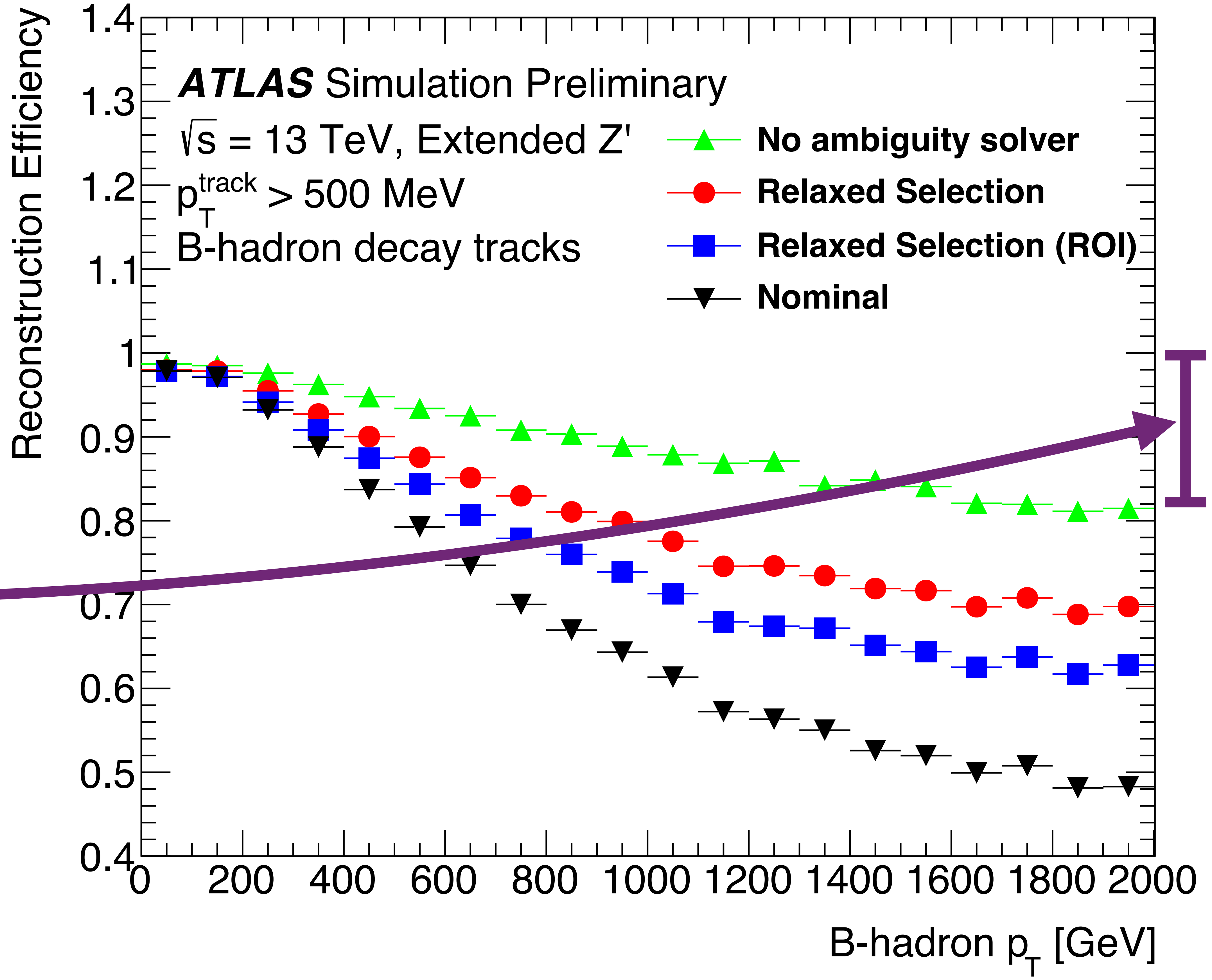
Tracking in dense environments is still not perfect



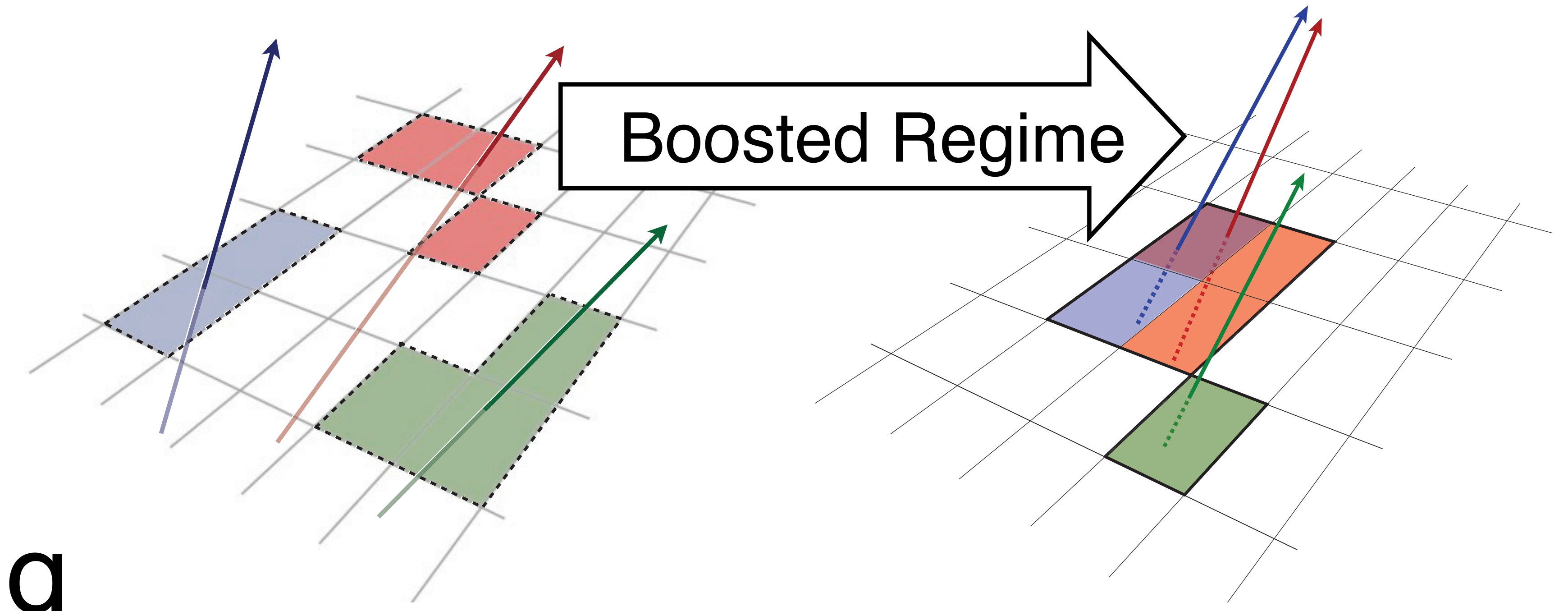
- Even with latest developments (e.g. pixel cluster splitting MDN, dedicated ambiguity resolution), clear room for improvement

- Two main culprits are:
  - Ambiguity resolution
  - Seeding

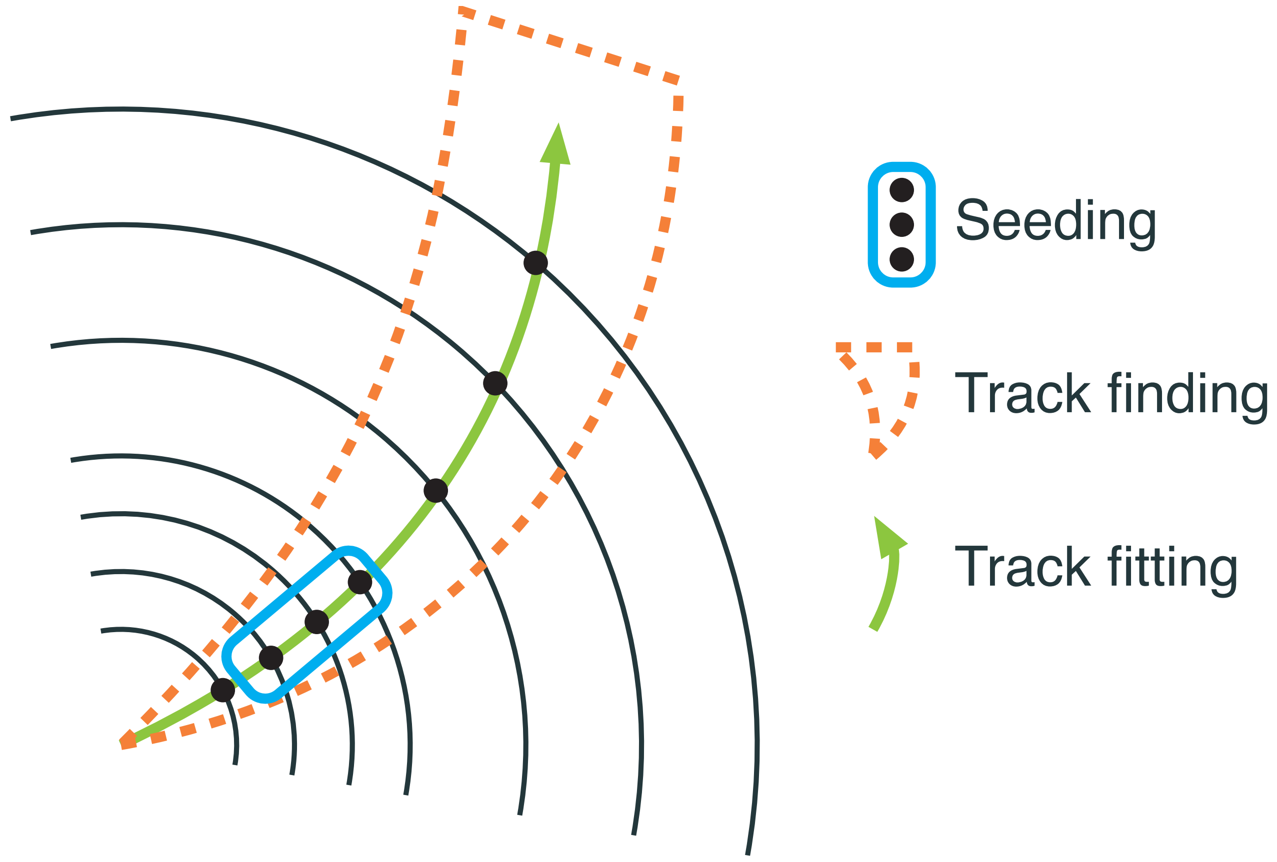




Tracking in dense environments is still not perfect

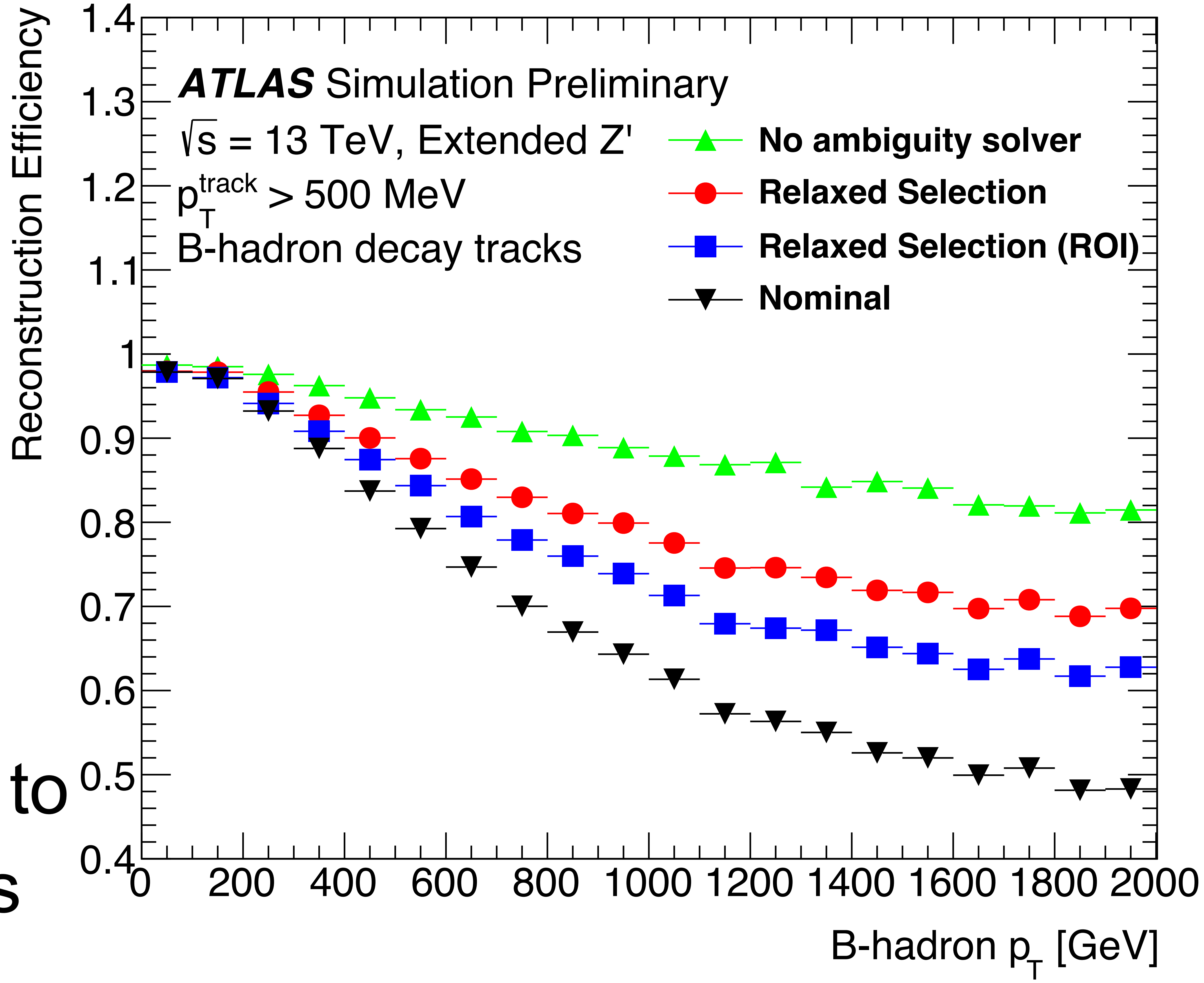


- Even with latest developments (e.g. pixel cluster splitting MDN, dedicated ambiguity resolution), clear room for improvement



- Two main culprits are:
  - Ambiguity resolution
  - Seeding

- Improving reconstruction efficiency for B-hadron tracks directly improves flavour tagging, leading to enhanced di-Higgs searches and measurements



Can we exploit advances in machine learning to improve these two facets of tracking at the same time?

MaskFormer is the current state of the art for image segmentation [2304.02643]

MaskFormer is the current state of the art for image segmentation [2304.02643]

- Semantic segmentation; identifying *stuff*, e.g. this pixel is a dog or a human or a mountain etc.

MaskFormer is the current state of the art for image segmentation [2304.02643]

- Semantic segmentation; identifying *stuff*, e.g. this pixel is a dog or a human or a mountain etc.
- Instance segmentation; identifying countable *things*, e.g. this pixel belongs to dog #1, dog #2, human #1, etc.

MaskFormer is the current state of the art for image segmentation [2304.02643]

- Semantic segmentation; identifying *stuff*, e.g. this pixel is a dog or a human or a mountain etc.
- Instance segmentation; identifying countable *things*, e.g. this pixel belongs to dog #1, dog #2, human #1, etc.

MaskFormer unifies semantic and instance segmentation to provide a **many-to-many mapping** from M input pixels to N output masks

MaskFormer is the current state of the art for image segmentation [2304.02643]

- Semantic segmentation; identifying *stuff*, e.g. this pixel is a dog or a human or a mountain etc.
- Instance segmentation; identifying countable *things*, e.g. this pixel belongs to dog #1, dog #2, human #1, etc.

MaskFormer unifies semantic and instance segmentation to provide a **many-to-many mapping** from M input pixels to N output masks

MaskFormer is the current state of the art for image segmentation [2304.02643]

- Semantic segmentation; identifying *stuff*, e.g. this pixel is a dog or a human or a mountain etc.
- Instance segmentation; identifying countable *things*, e.g. this pixel belongs to dog #1, dog #2, human #1, etc.

MaskFormer unifies semantic and instance segmentation to provide a **many-to-many mapping** from M input pixels to N output masks

MaskFormer is the current state of the art for image segmentation [2304.02643]

- Semantic segmentation; identifying *stuff*, e.g. this pixel is a dog or a human or a mountain etc.
- Instance segmentation; identifying countable *things*, e.g. this pixel belongs to dog #1, dog #2, human #1, etc.

MaskFormer unifies semantic and instance segmentation to provide a **many-to-many mapping** from M input pixels to N output masks

Identify objects in images by learning binary masks over input pixels

MaskFormer for particle physics 

MaskFormer for particle physics

Identify objects in images by learning binary masks over input pixels

MaskFormer for particle physics

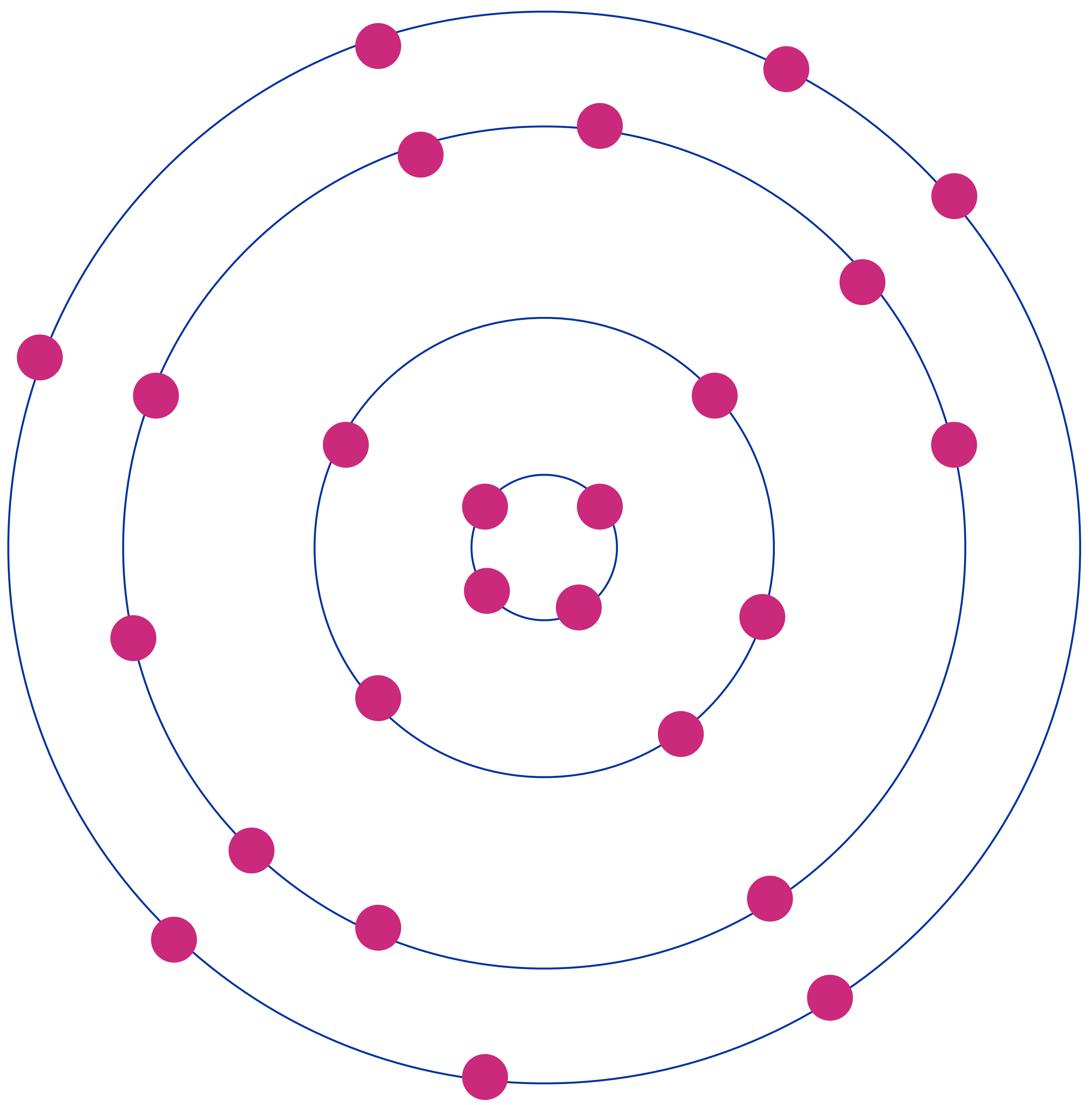
Identify objects in images by learning binary masks over input pixels

Reconstruct tracks in events by learning binary masks over input hits

MaskFormer for particle physics

Identify objects in images by learning binary masks over input pixels

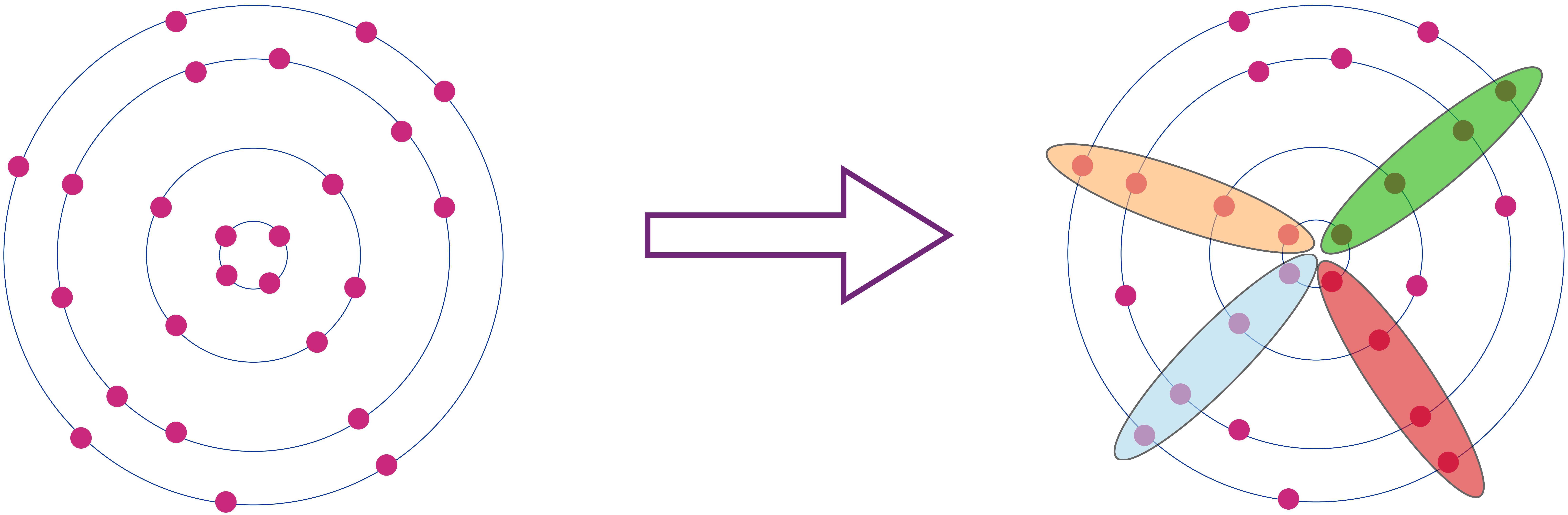
Reconstruct tracks in events by learning binary masks over input hits



MaskFormer for particle physics

Identify objects in images by learning binary masks over input pixels

Reconstruct tracks in events by learning binary masks over input hits



MaskFormer for particle physics

Identify objects in images by learning binary masks over input pixels

Reconstruct tracks in events by learning binary masks over input hits

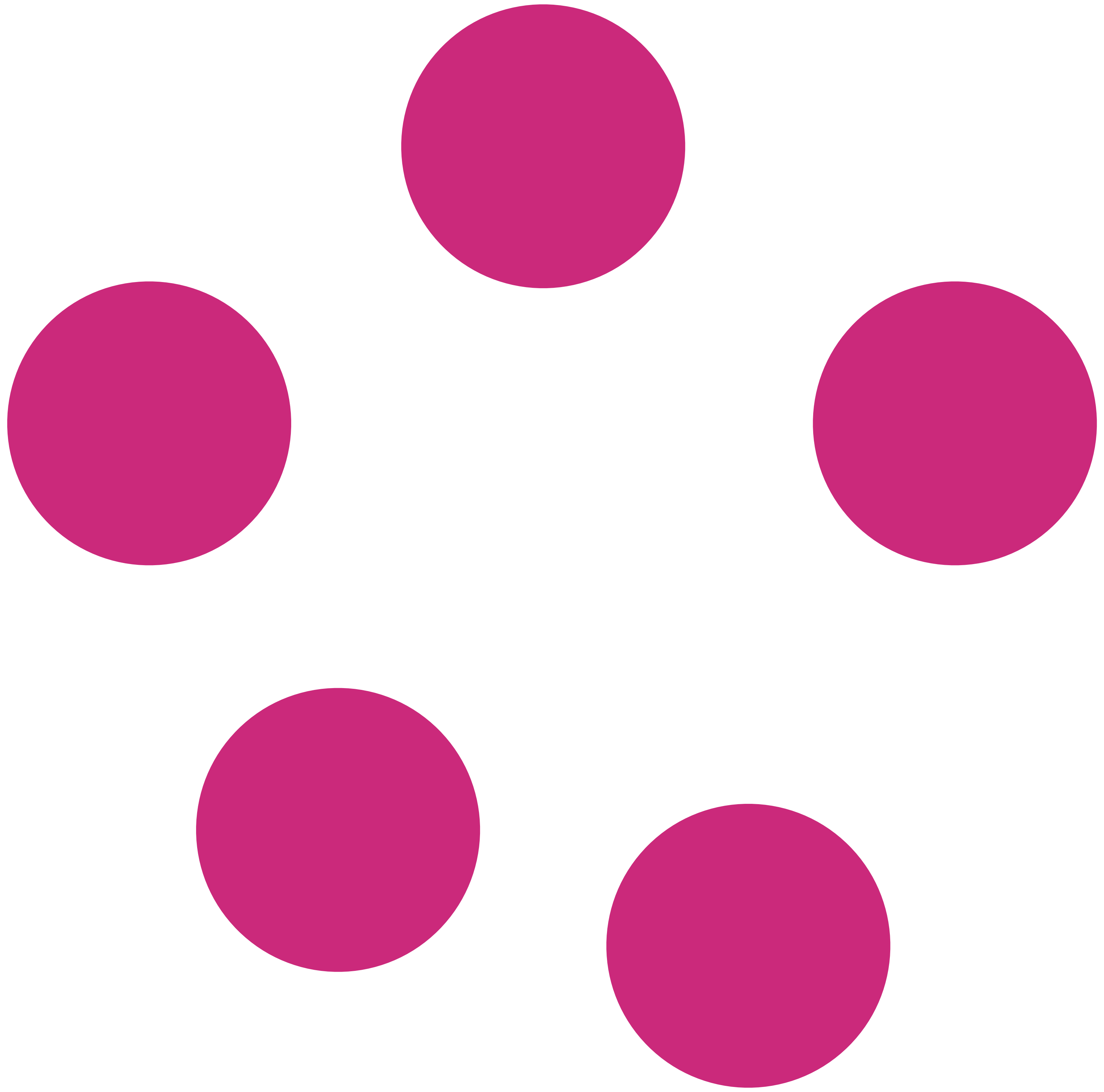
Reconstruct vertices in jets by learning binary masks over input tracks

MaskFormer for particle physics

Identify objects in images by learning binary masks over input pixels

Reconstruct tracks in events by learning binary masks over input hits

Reconstruct vertices in jets by learning binary masks over input tracks

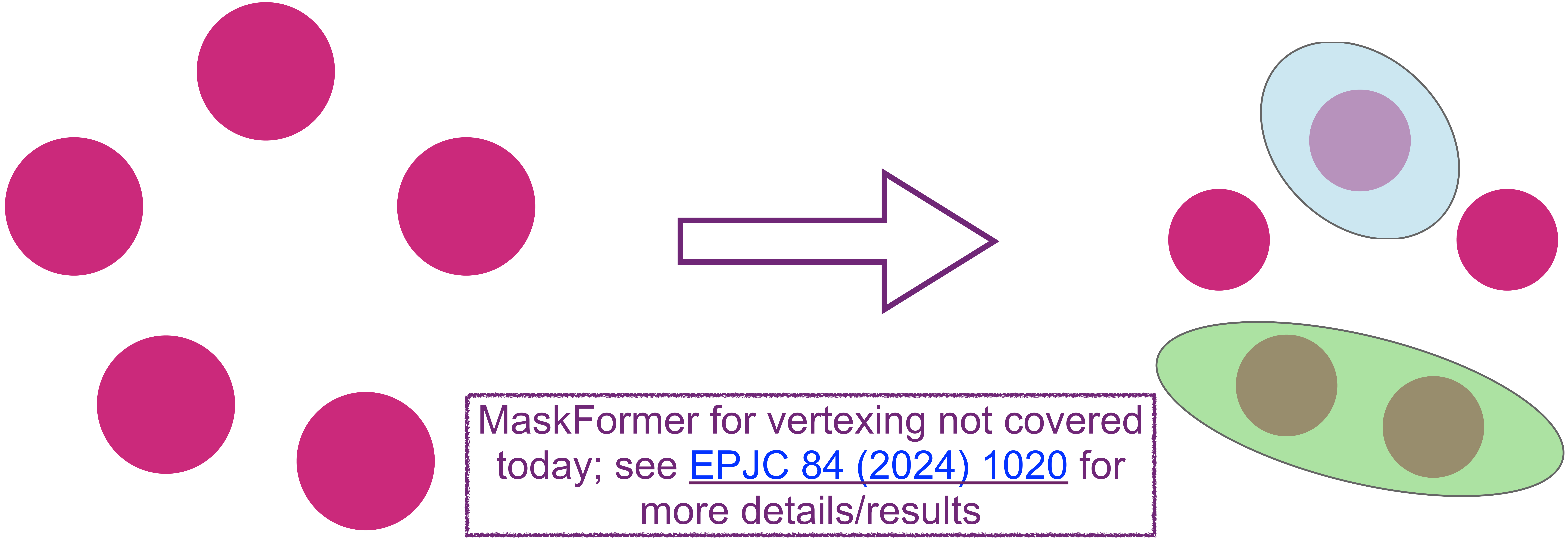


MaskFormer for particle physics

Identify objects in images by learning binary masks over input pixels

Reconstruct tracks in events by learning binary masks over input hits

Reconstruct vertices in jets by learning binary masks over input tracks



MaskFormer for vertexing not covered today; see EPJC 84 (2024) 1020 for more details/results

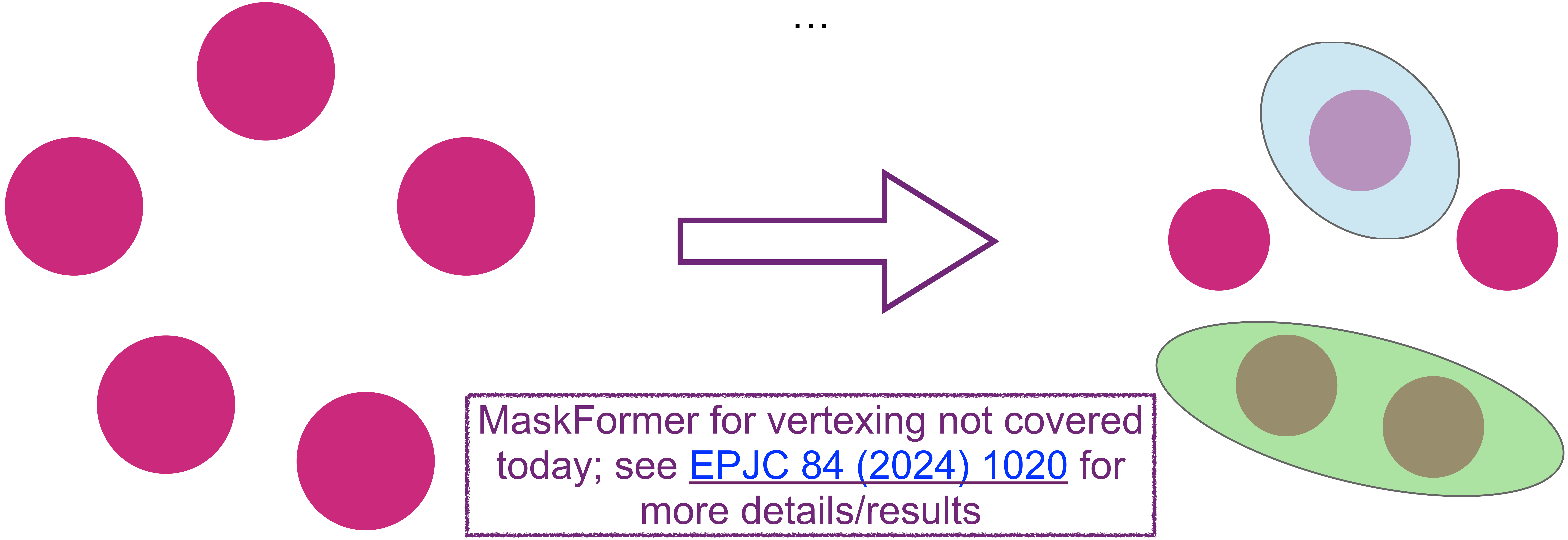
MaskFormer for particle physics

Identify objects in images by learning binary masks over input pixels

Reconstruct tracks in events by learning binary masks over input hits

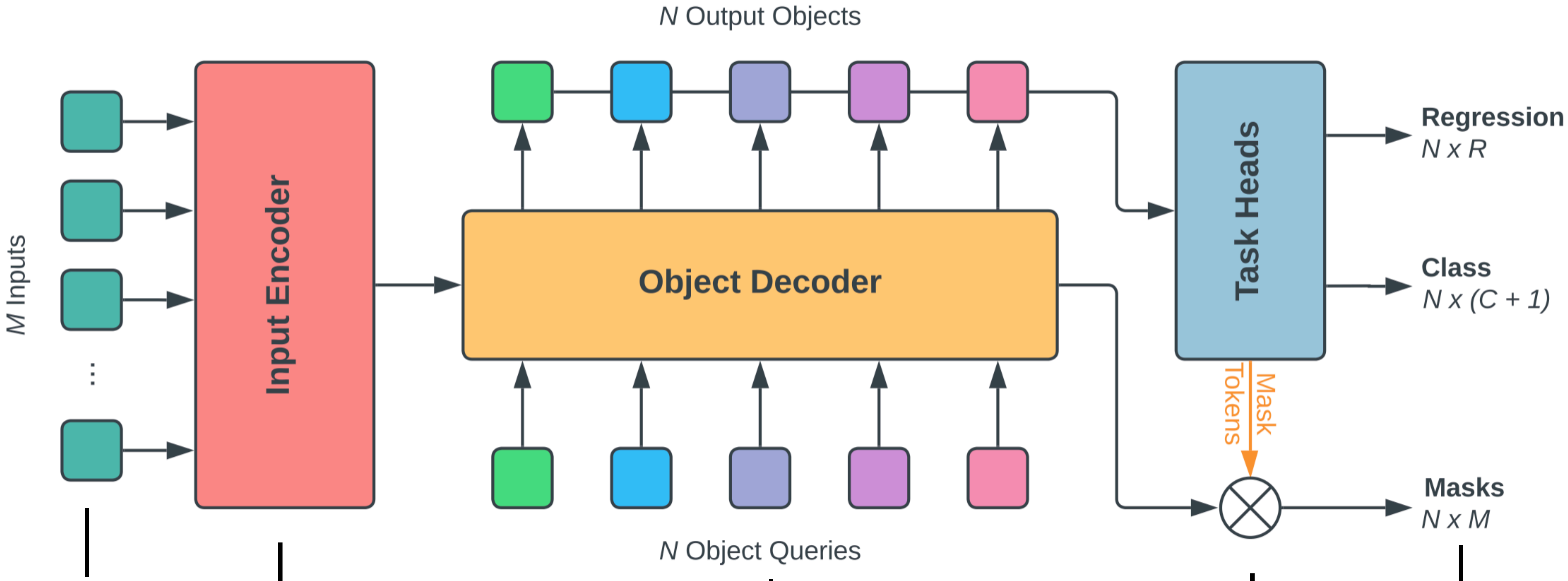
Reconstruct vertices in jets by learning binary masks over input tracks

…



MaskFormer for vertexing not covered today; see [EPJC 84 (2024) 1020]() for more details/results

MaskFormer architecture



Input *hits*

Transformer encoder

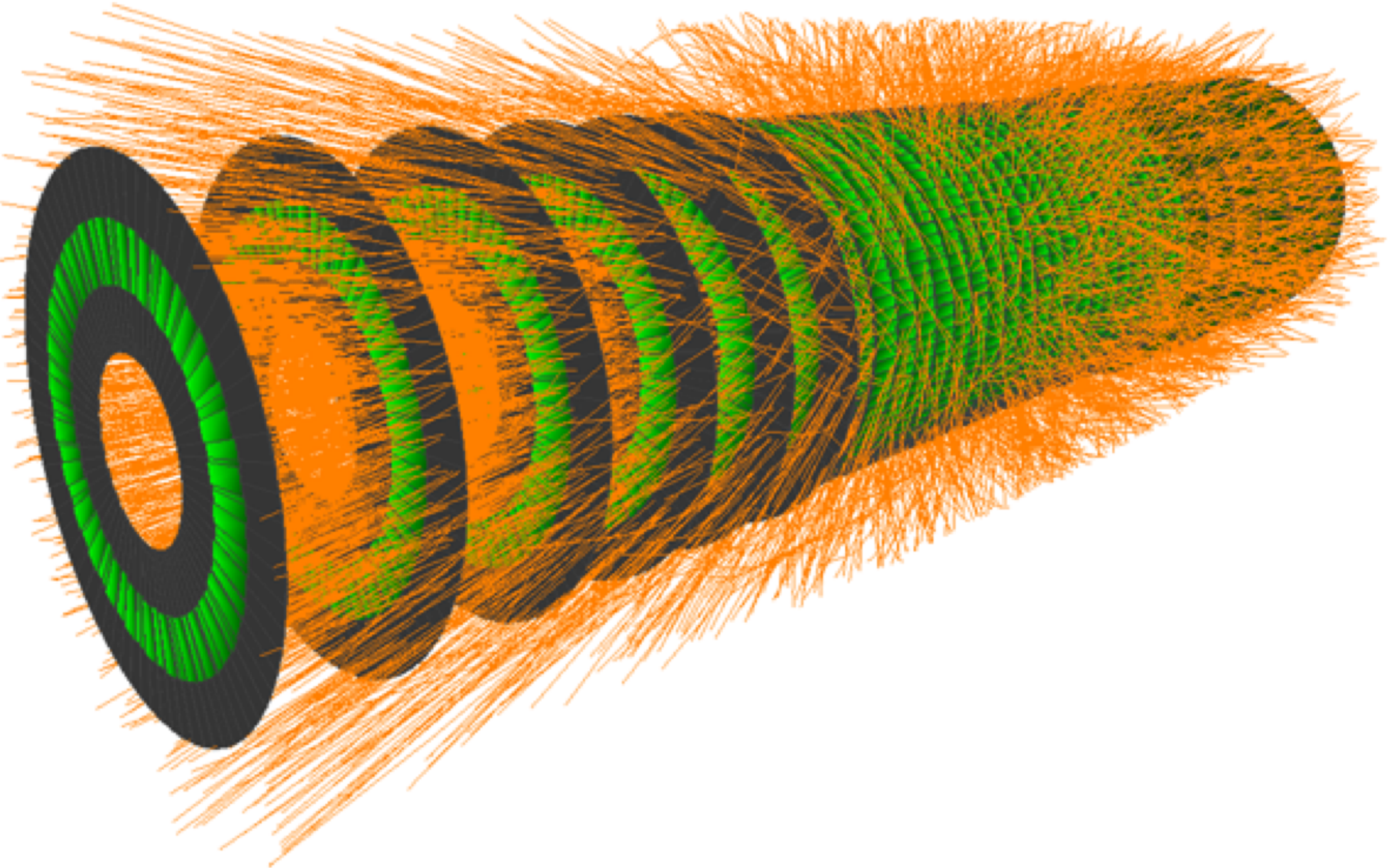
Transformer decoder with bidirectional cross attention

Mask prediction

Output *tracks*

Starting point: trackML

TrackML challenge

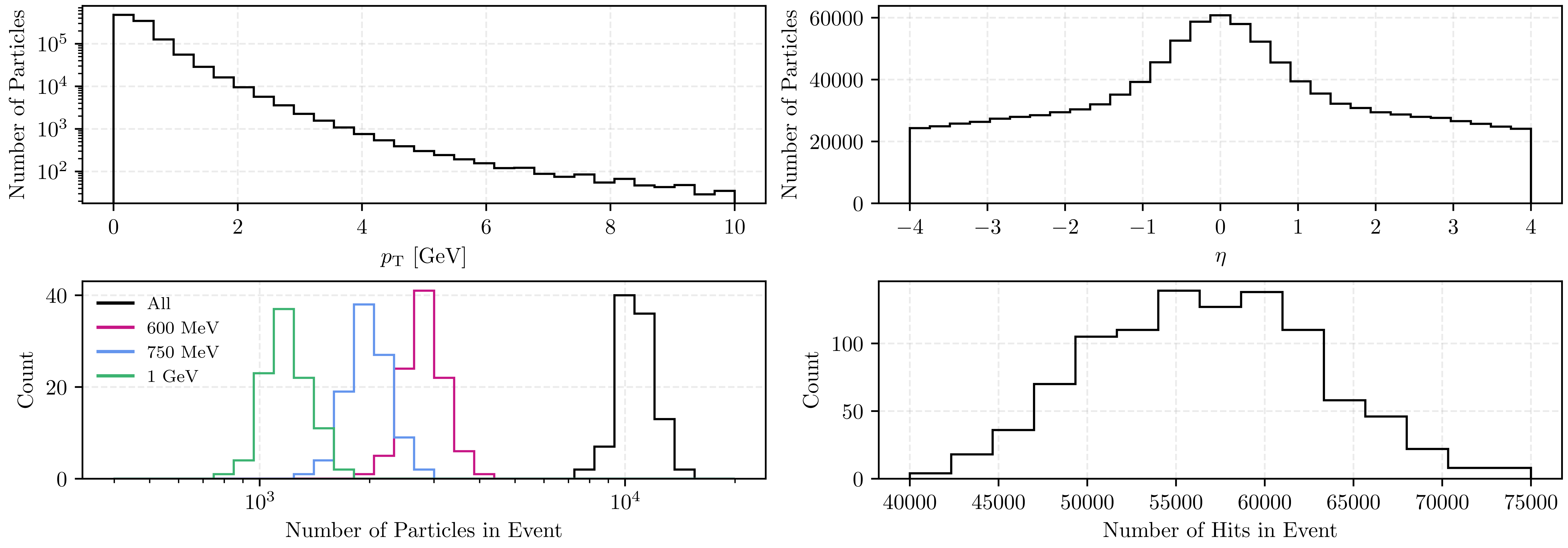


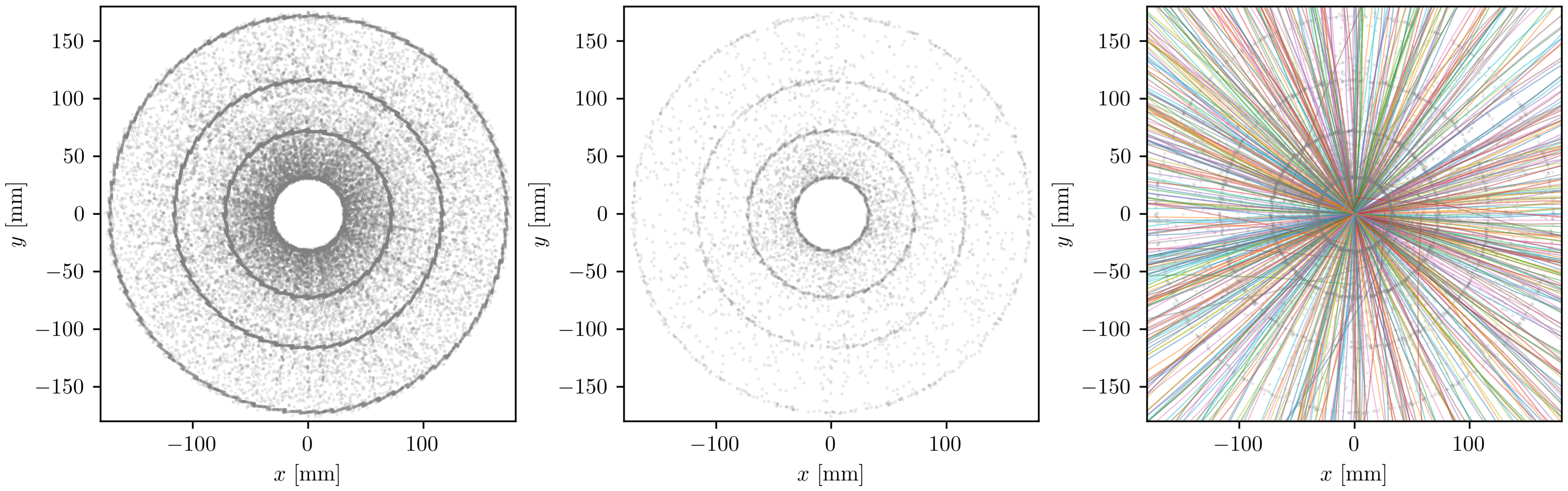
- Starting with the trackML dataset to benchmark performance
- Focus on accuracy phase first, then move to throughput phase
- $\mathcal{O}(100\text{k})$ hits, $\mathcal{O}(10\text{k})$ particles

	$p_{\mathrm{T}}^{\mathrm{min}}$	max $\lvert\eta\rvert$	Layers Used	Preprocessing	Postprocessing
GNN4ITK [29]	1 GeV	4.0	Pixel + strip	Edge classification	Graph traversal
HGNN [30]	1 GeV	4.0	Pixel + strip	Edge classification	GMPool [30]
OC [31]	900 MeV	4.0	Pixel	Edge classification	Clustering
This Work	600 MeV	2.5	Pixel	Hit filtering	None

Tracking with MaskFormer

- Simplify problem in the first instance; focus on innermost pixel detector (no strips for now)
- Two-step approach used:
  1. Filter hits with transformer
  2. Run tracking on remaining hits with MaskFormer





Nikhef

Training setup

$p_T^{\min}$	Hits (Pre)	Hits (Post)	Particles	Object Queries
1 GeV	57k	6k	800	1100
750 MeV	57k	8k	1300	1800
600 MeV	57k	12k	1800	2100

- Target tracks:
  - At least 3 hits in pixel detector, $|\eta| < 2.5$
  - p_T > threshold (600 MeV, 750 MeV, 1 GeV depending on model)
- Thresholds chosen to explore trade-offs between model complexity, inference time, and performance
- Each setup trains dedicated hit filtering (HF) and tracking Maskformer (MF)
  - HF-600 MeV and HF-750 MeV: ~8M trainable parameters
  - HF-1 GeV: ~5M trainable parameters (optimized to minimize inference times)
  - MF models: ~22M trainable parameters

```yaml
inputs:
  hit:
    - "x"
    - "y"
    - "z"
    - "r"
    - "s"
    - "eta"
    - "phi"
    - "u"
    - "v"
    - "charge_frac"
    - "leta"
    - "lphi"
    - "lx"
    - "ly"
    - "lz"
    - "geta"
    - "gphi"
```

Training setup

p_T^{min}	Hits (Pre)	Hits (Post)	Particles	Object Queries
1 GeV	57k	6k	800	1100
750 MeV	57k	8k	1300	1800
600 MeV	57k	12k	1800	2100

- Target tracks:
  - At least 3 hits in pixel detector, $|\eta| < 2.5$
  - p_T > threshold (600 MeV, 750 MeV, 1 GeV depending on model)
- Thresholds chosen to e een model complexity, inference ti
- Each setup trains dedic nd tracking Maskformer (MF)
  - HF-600 MeV and HF-7 arameters
  - HF-1 GeV: ~5M trainable parameters (optimized to minimize inference times)
  - MF models: ~22M trainable parameters

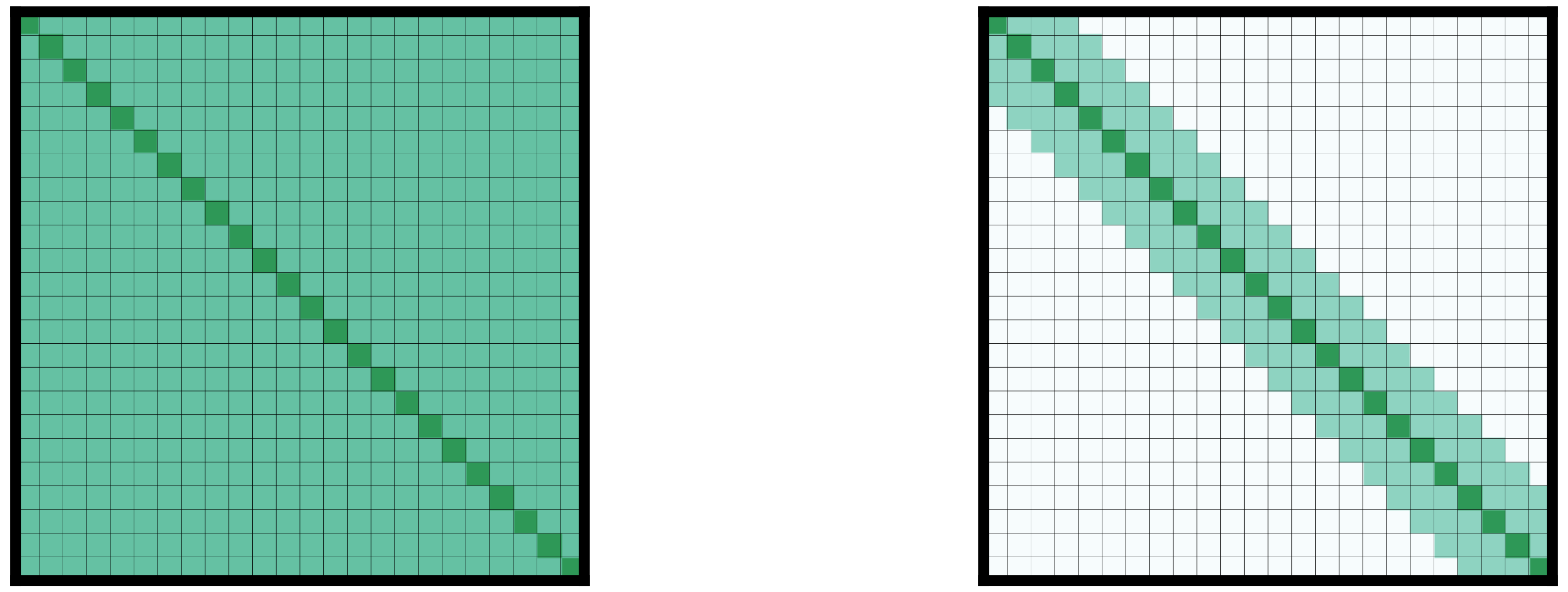
Trained on a single NVIDIA A100 GPU for 30 epochs (batch size of 1)

Hit filtering: ~10 hours

Tracking: 20-60 hours, depending on p_T threshold

```
inputs:
  hit:
    - "x"
    - "y"
    - "z"
    - "r"
    - "s"
    - "eta"
    - "phi"
    - "u"
    - "v"
    - "charge_frac"
    - "leta"
    - "lphi"
    - "lx"
    - "ly"
    - "lz"
    - "geta"
    - "gphi"
```

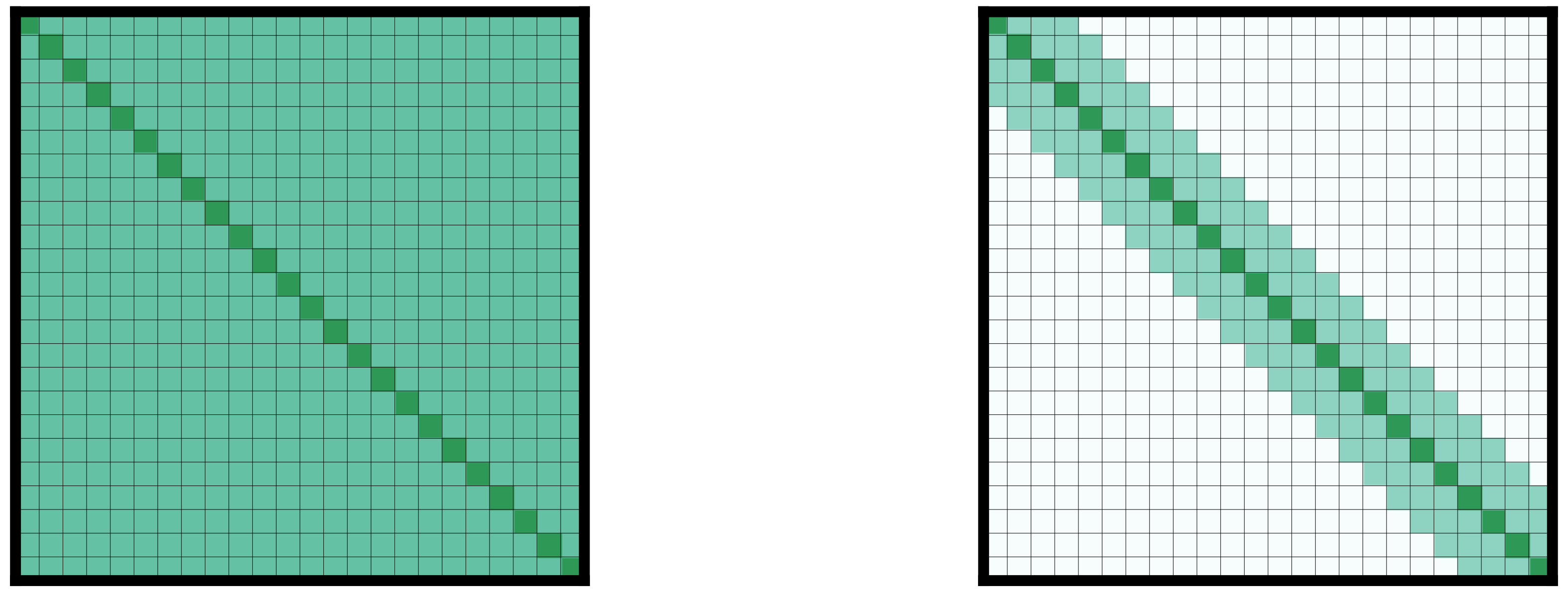
Sliding window attention



(a) Full M^2 attention (b) Sliding window attention

- Typical transformer architectures have $\mathcal{O}(M^2)$ complexity due to self-attention
- Assume hits only need to attend to nearby hits in the azimuthal angle ϕ
- Ordering hits in ϕ and assuming ϕ-locality allows us to use sliding window attention which scales like $\mathcal{O}(M \times w)$, where w is the width of the sliding window (i.e. linearly in number of input hits M)
- Note: append first $w/2$ hits to the end of the sequence and vice-versa to allow hits to communicate around the $\pm\pi$ boundary
- `FlashAttention-2` and `SwiGLU` activation improve performance

Sliding window attention



(a) Full M^2 attention (b) Sliding window attention

- Typical transformer architectures have $\mathcal{O}(M^2)$ complexity due to self-attention
- Assume hits only nee… the azimuthal angle ϕ
- Ordering hits in ϕ and… us to use sliding window attention whic… here w is the width of the sliding window (i.e… hits M)
- Note: append first $w/2$ hits to the end of the sequence and vice-versa to allow hits to communicate around the $\pm\pi$ boundary
- FlashAttention-2 and SwiGLU activation improve performance

From the SwiGLU paper:
"We offer no explanation as to why these architectures seem to work; we attribute their success, as all else, to divine benevolence."

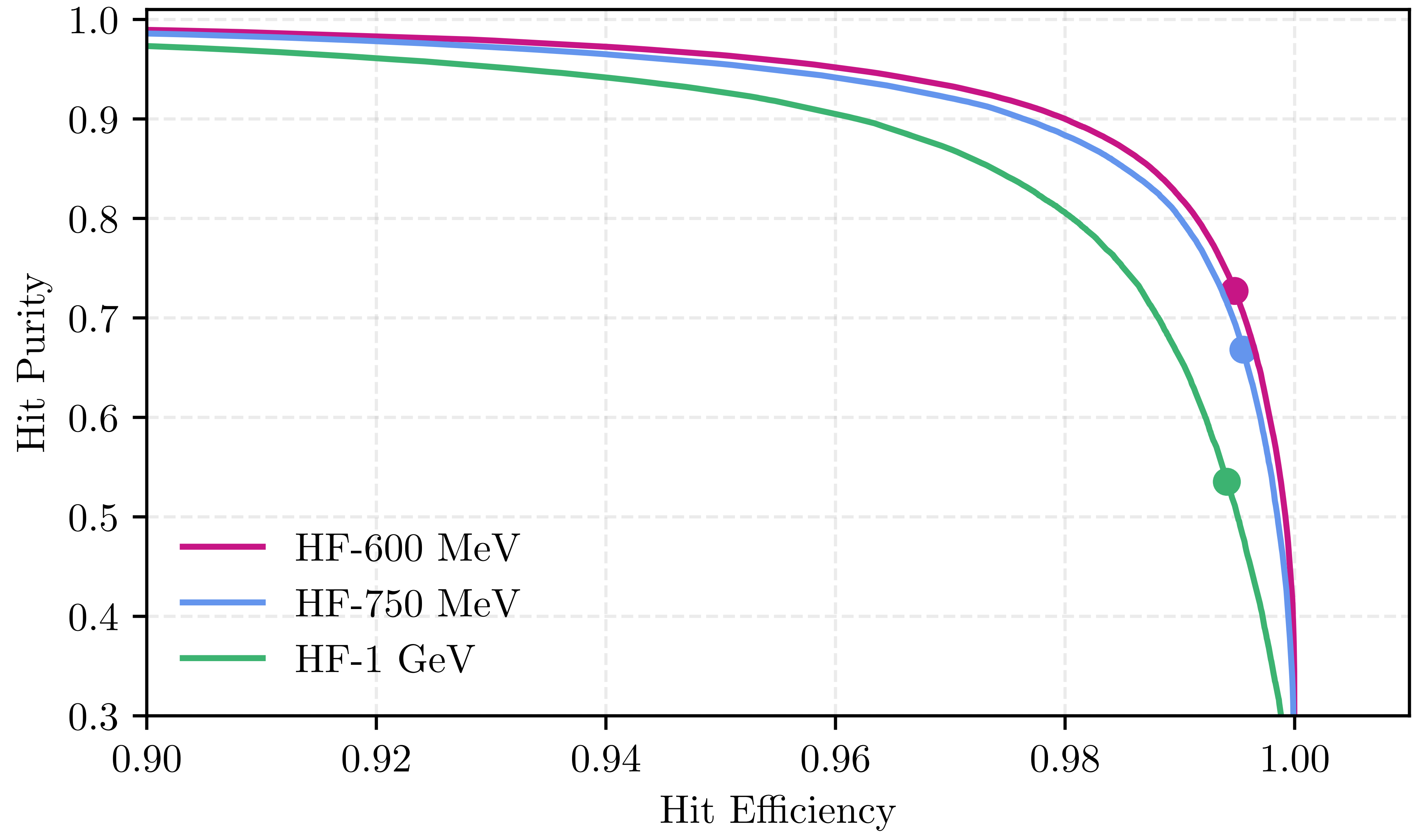
Hit filtering

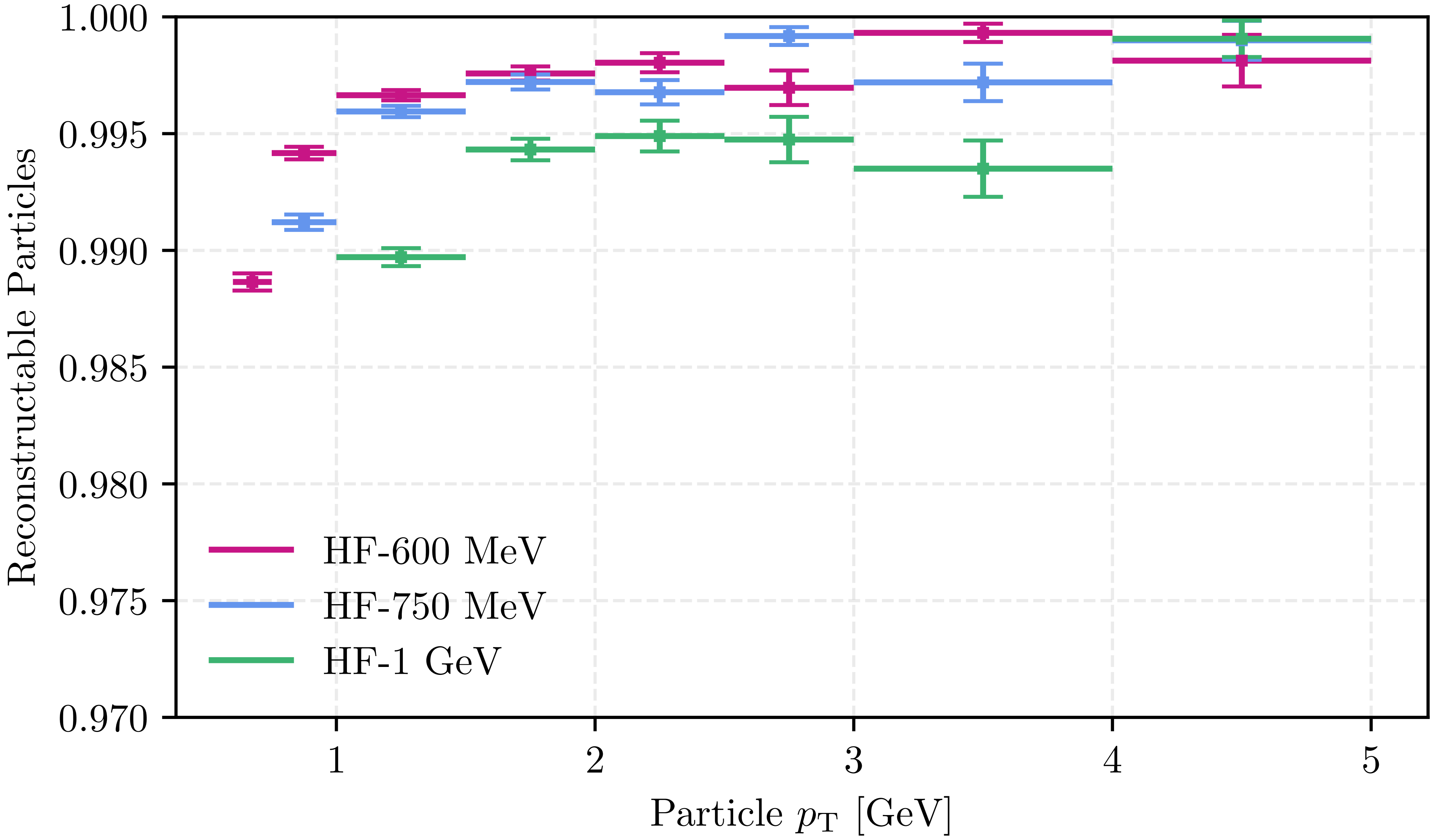
- Exploit Transformer-based hit filtering model to reduce the input hit multiplicities by predicting whether each hit is noise or signal
- Signal: hit belonging to a reconstructable particle
- Noise: hits belonging to particles that we do not wish to reconstruct (e.g. particle with p_T below threshold or outside η acceptance) and intrinsic noise hits not belonging to any particle
- Use hit input features to represent hit in $d = 256$-dimensional latent space
- Window size $w = 1024$ used for sliding window attention
- Output embeddings are classified using a dense network with three hidden layers

```
inputs:
  hit:
    - "x"
    - "y"
    - "z"
    - "r"
    - "s"
    - "eta"
    - "phi"
    - "u"
    - "v"
    - "charge_frac"
    - "leta"
    - "lphi"
    - "lx"
    - "ly"
    - "lz"
    - "geta"
    - "gphi"
```

Hit filtering results

- Hit efficiency: fraction of signal hits retained after filtering
- Hit purity: fraction of retained hits that are signal hits
- Markers show efficiency and purity at the chosen threshold of 0.1

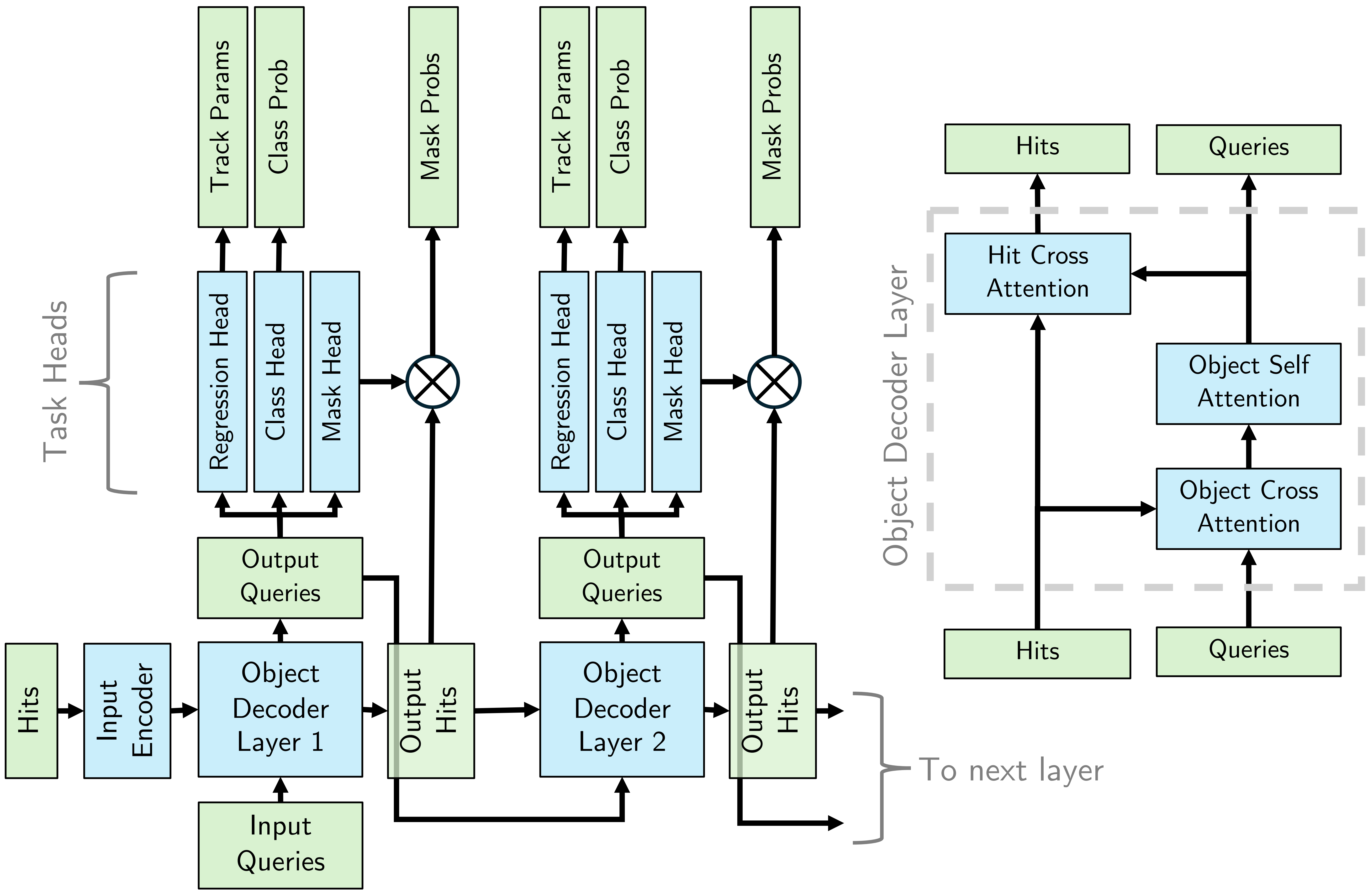




Model	Initial Purity ($\lvert\eta\rvert < 2.5$)	Filter Efficiency	Filter Purity	Reconstructable
HF-1 GeV	6.8% (13.3%)	99.4%	53.6%	99.1%
HF-750 MeV	11.1% (21.9%)	99.6%	66.8%	99.4%
HF-600 MeV	15.6% (30.6%)	99.5%	72.7%	99.3%

Track reconstruction

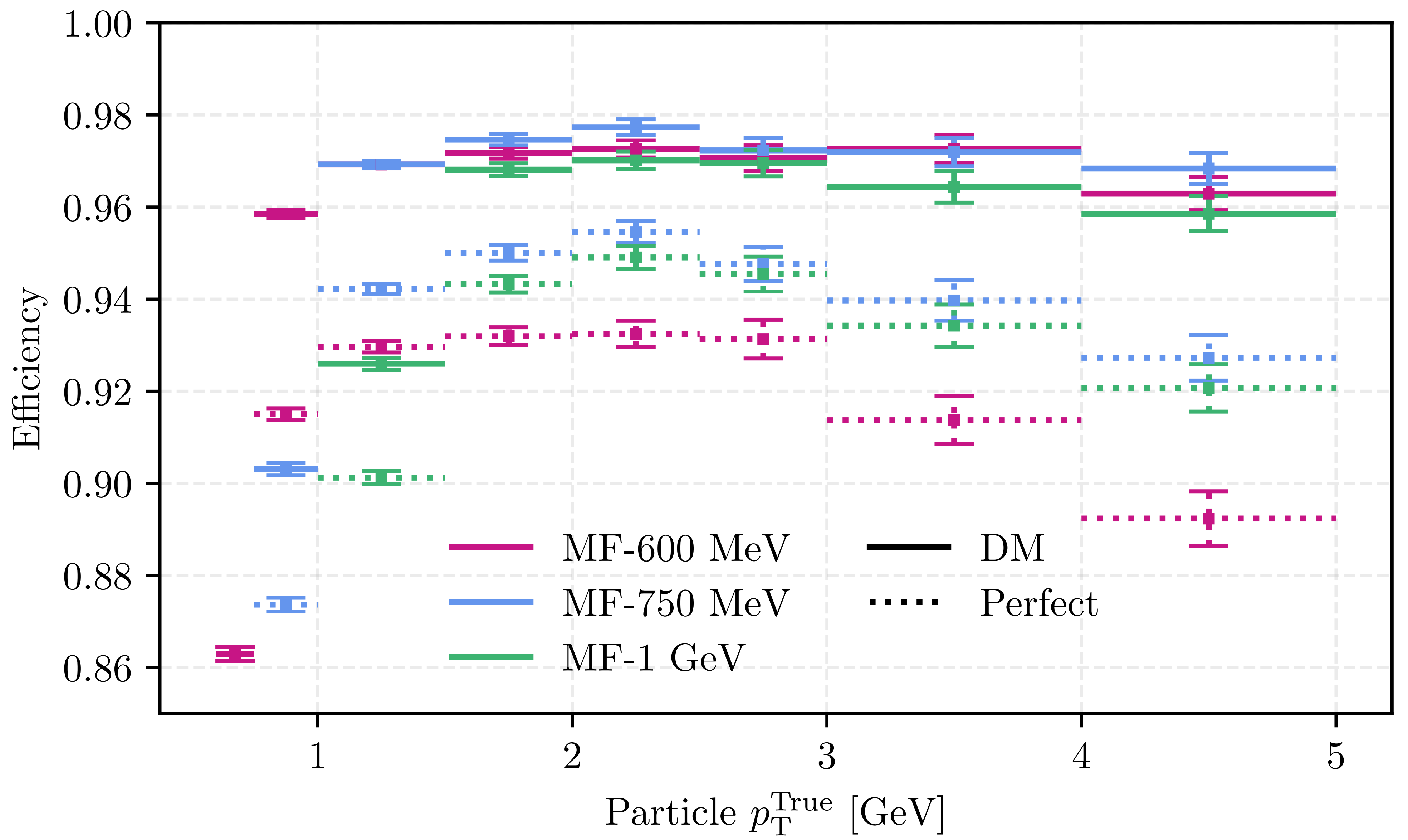
- Window size $w = 512$ to compensate for reduced hit multiplicities after filtering
- Object queries represent possible output tracks (set number of queries to max number of tracks in the training dataset)

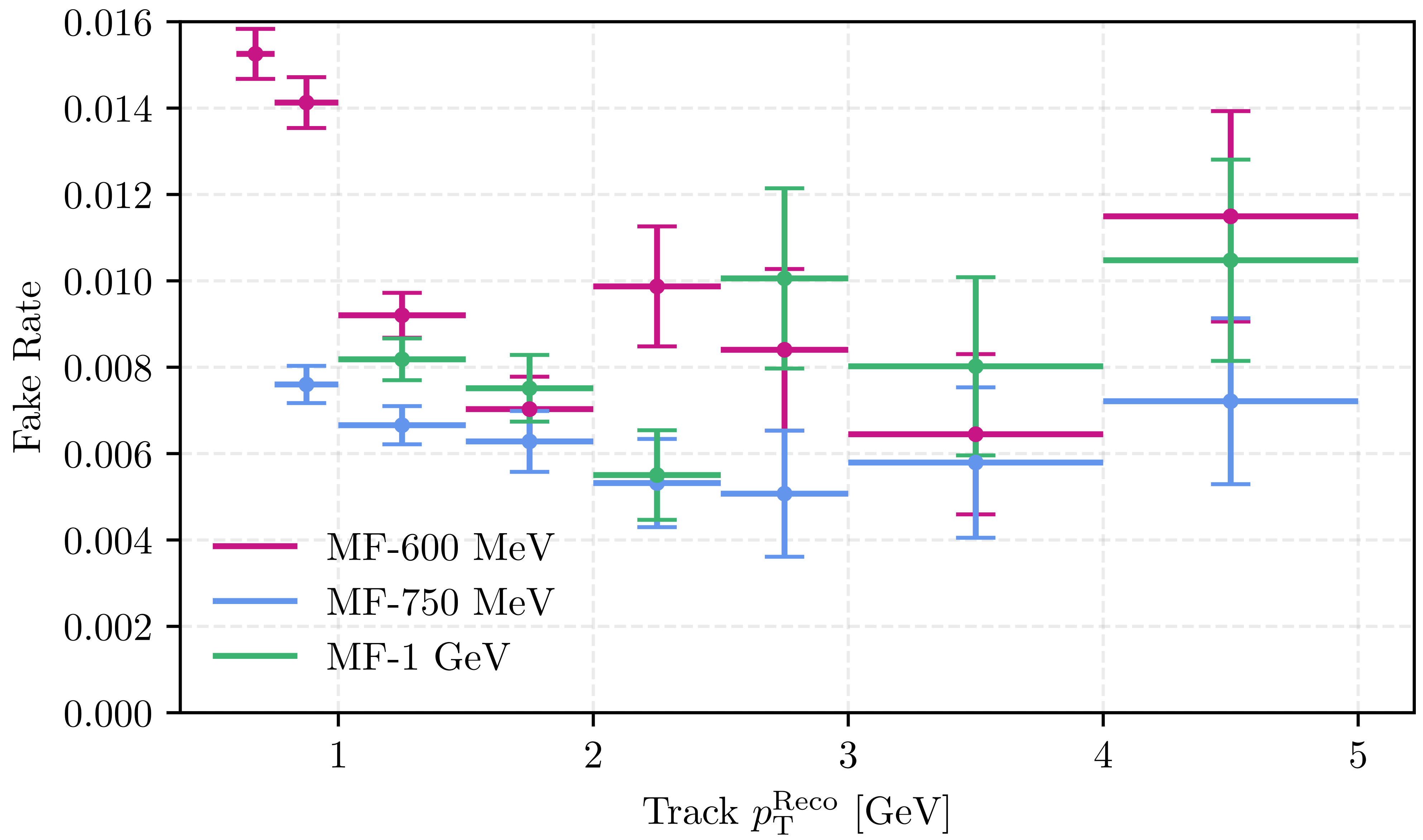


- Three task heads: track class, track-to-hit assignment, track properties
- Use 8 decoder layers
- Regressed track properties: p_x, p_y, p_z, production vertex z position

Tracking results

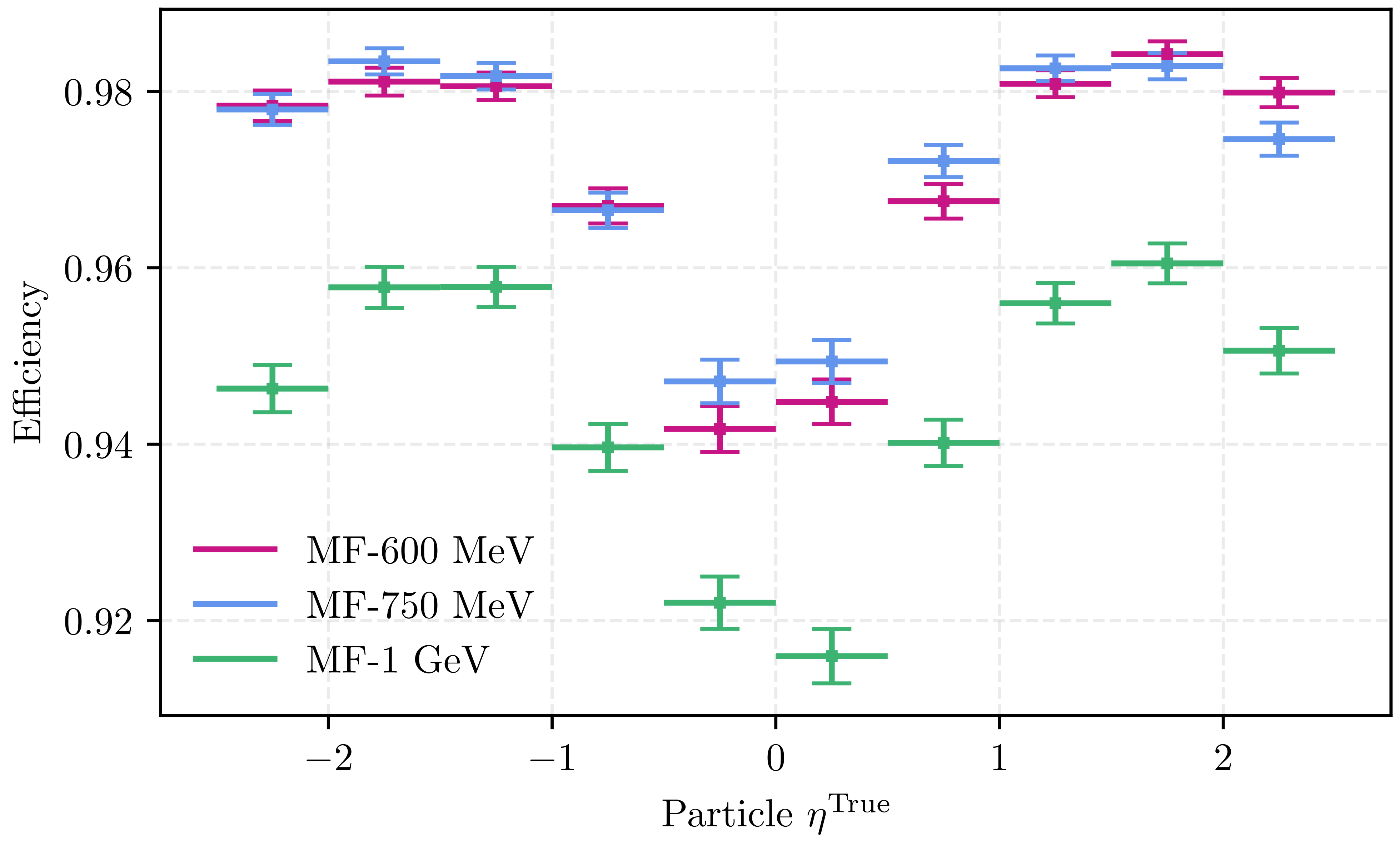
- Efficiency: fraction of reconstructable particles that are correctly matched to a reconstructed track
- Fake rate: fraction of reconstructed tracks that are not well-matched to any reconstructable particle
- Matching criteria:
  - Double majority (DM): match occurs if >50% of the particle's hits are assigned to the track and >50% of the hits on the track are from that particle
  - Perfect matching: match occurs if all of the particle's hits are assigned to the track, and no other hits are assigned
- Each track is matched to the simulated particle that contributes the largest number of hits to the track (particle chosen at random if two particles have the same number of hits on a given track)

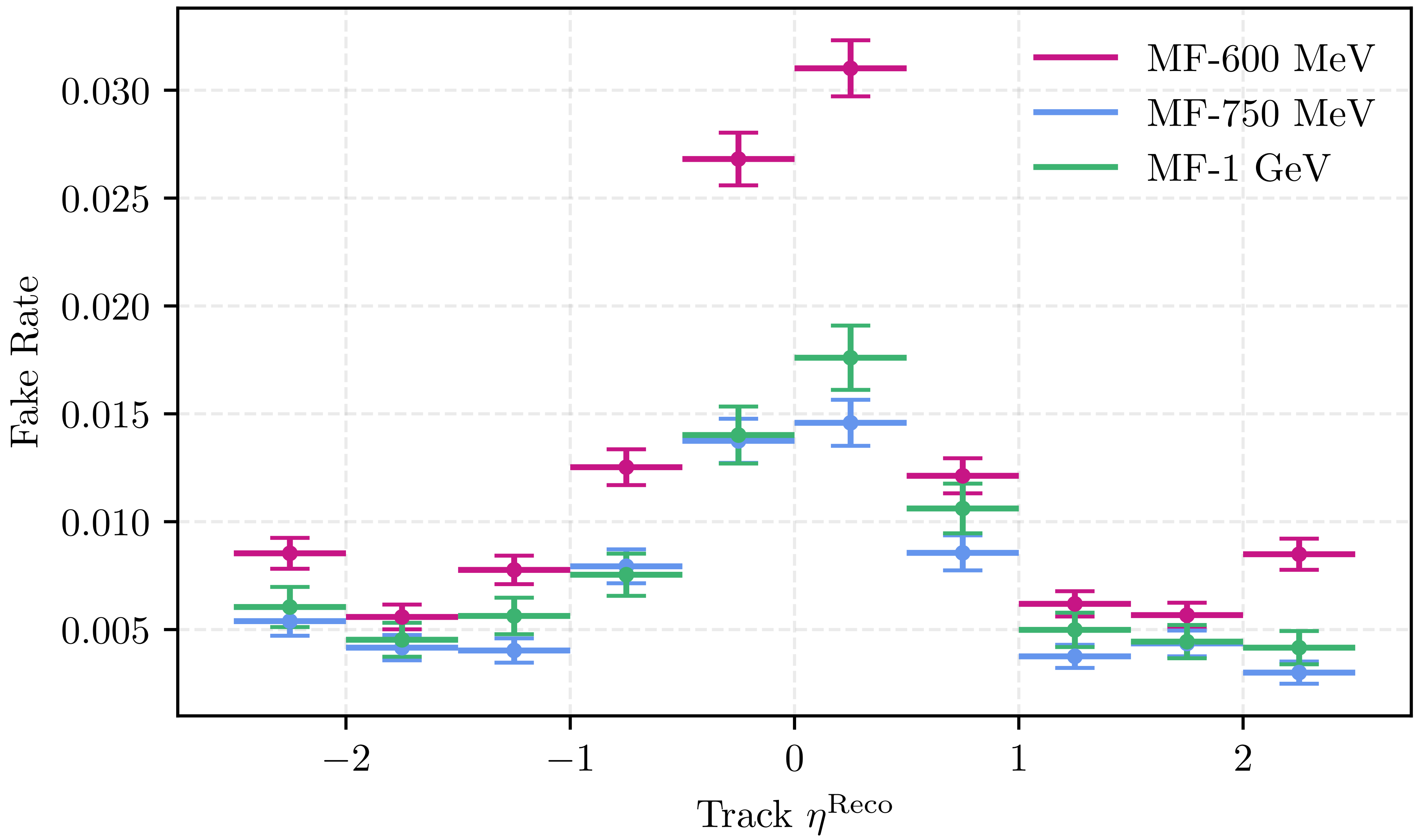




Tracking results

- Efficiency: fraction of reconstructable particles that are correctly matched to a reconstructed track
- Fake rate: fraction of reconstructed tracks that are not well-matched to any reconstructable particle
- Matching criteria:
  - Double majority (DM): match occurs if >50% of the particle's hits are assigned to the track and >50% of the hits on the track are from that particle
  - Perfect matching: match occurs if all of the particle's hits are assigned to the track, and no other hits are assigned
- Each track is matched to the simulated particle that contributes the largest number of hits to the track (particle chosen at random if two particles have the same number of hits on a given track)

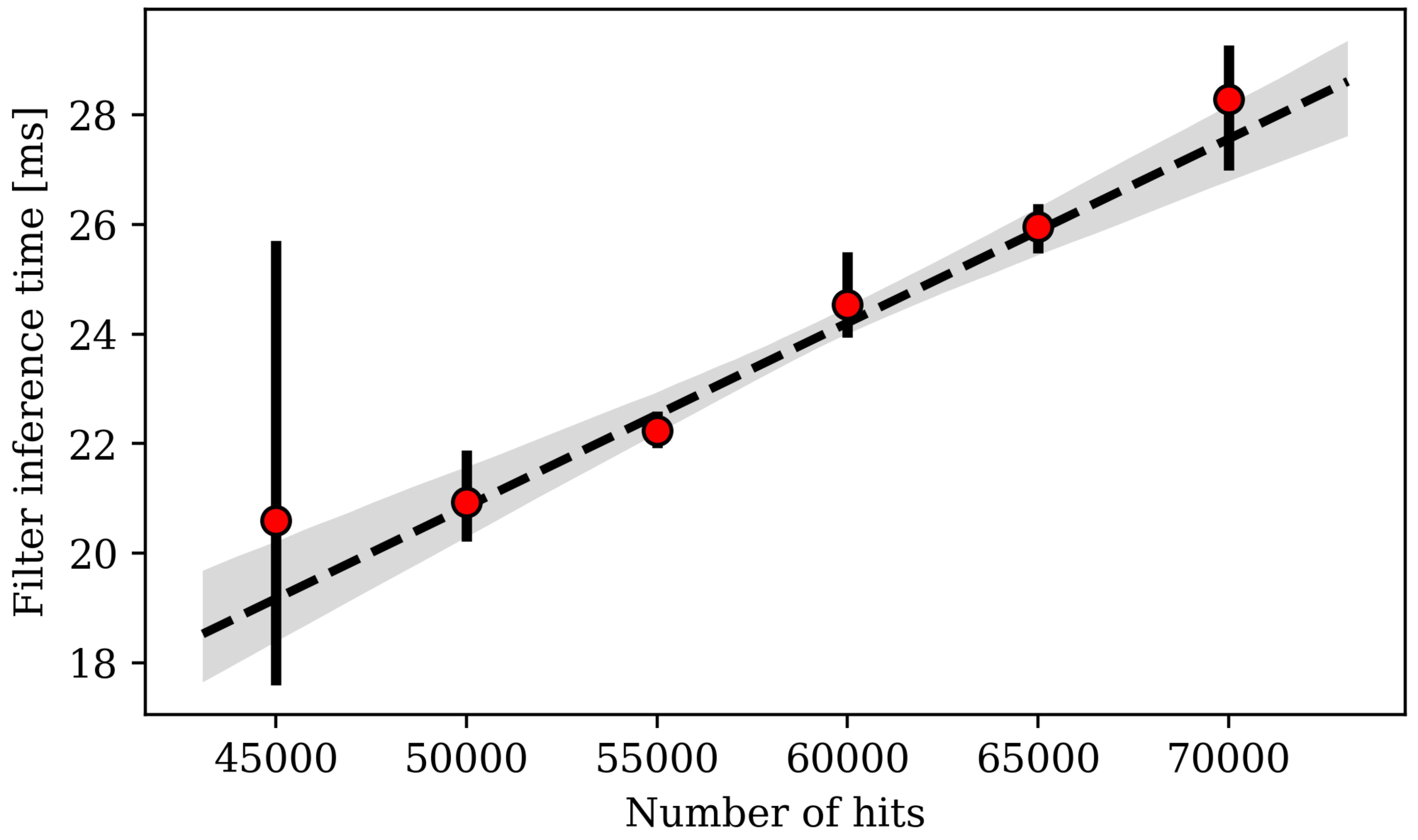


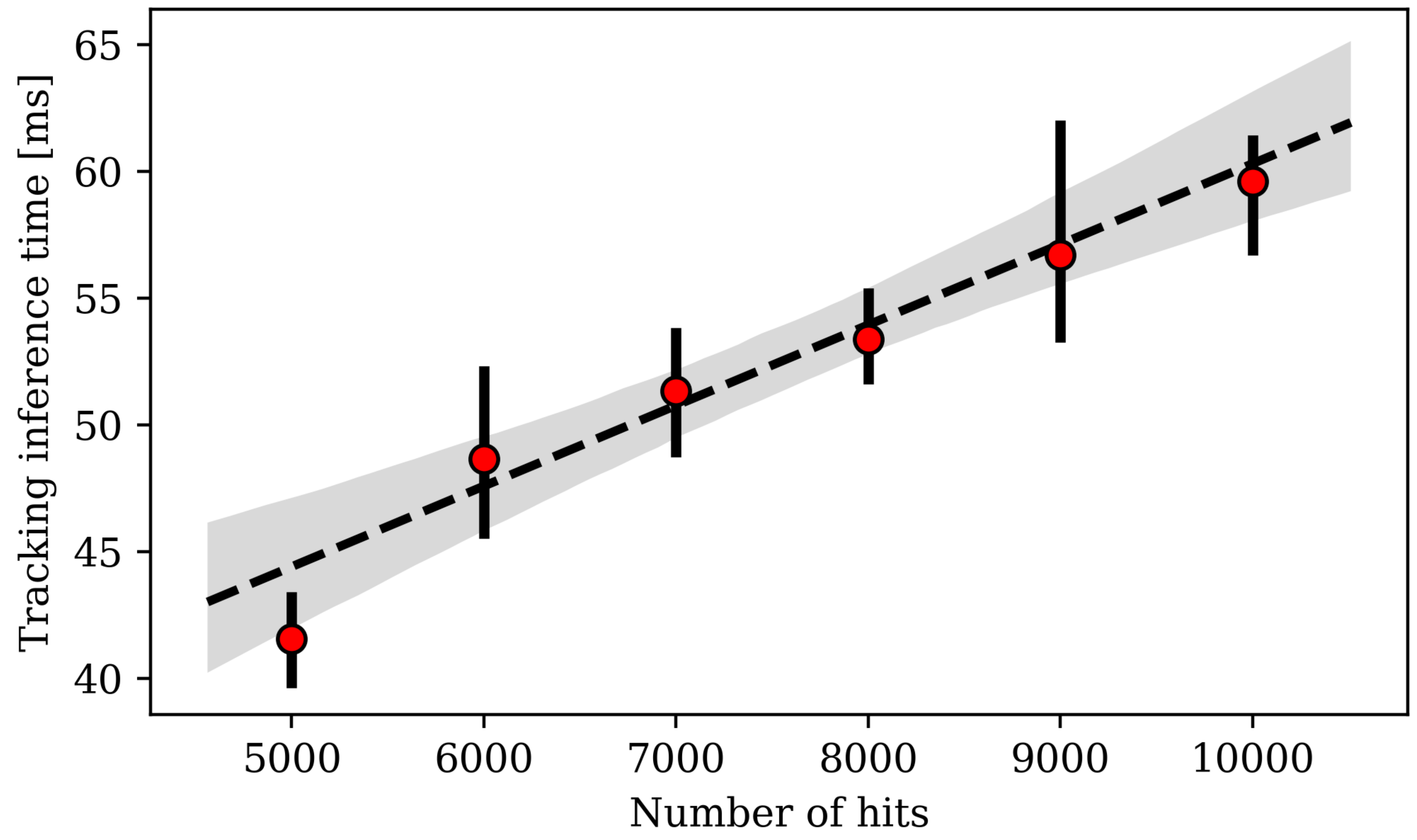


Inference times

- Timing studies carried out on NVIDIA A100 GPU (batch size 1)
- Hit filtering forward pass requires on average 23 ms per event with $\mathcal{O}(60\mathrm{k})$ hits
- Rough extrapolation to ITk with $\mathcal{O}(300\mathrm{k})$ hits provides 390 ms for filter+tracking assuming linear scaling holds and hit filter retains 25% of all hits

	Tracking time [ms]	Filter + tracking time [ms]
MF-1 GeV	50 ± 7	73 ± 8
MF-750 MeV	77 ± 9	100 ± 11
MF-600 MeV	101 ± 12	124 ± 14





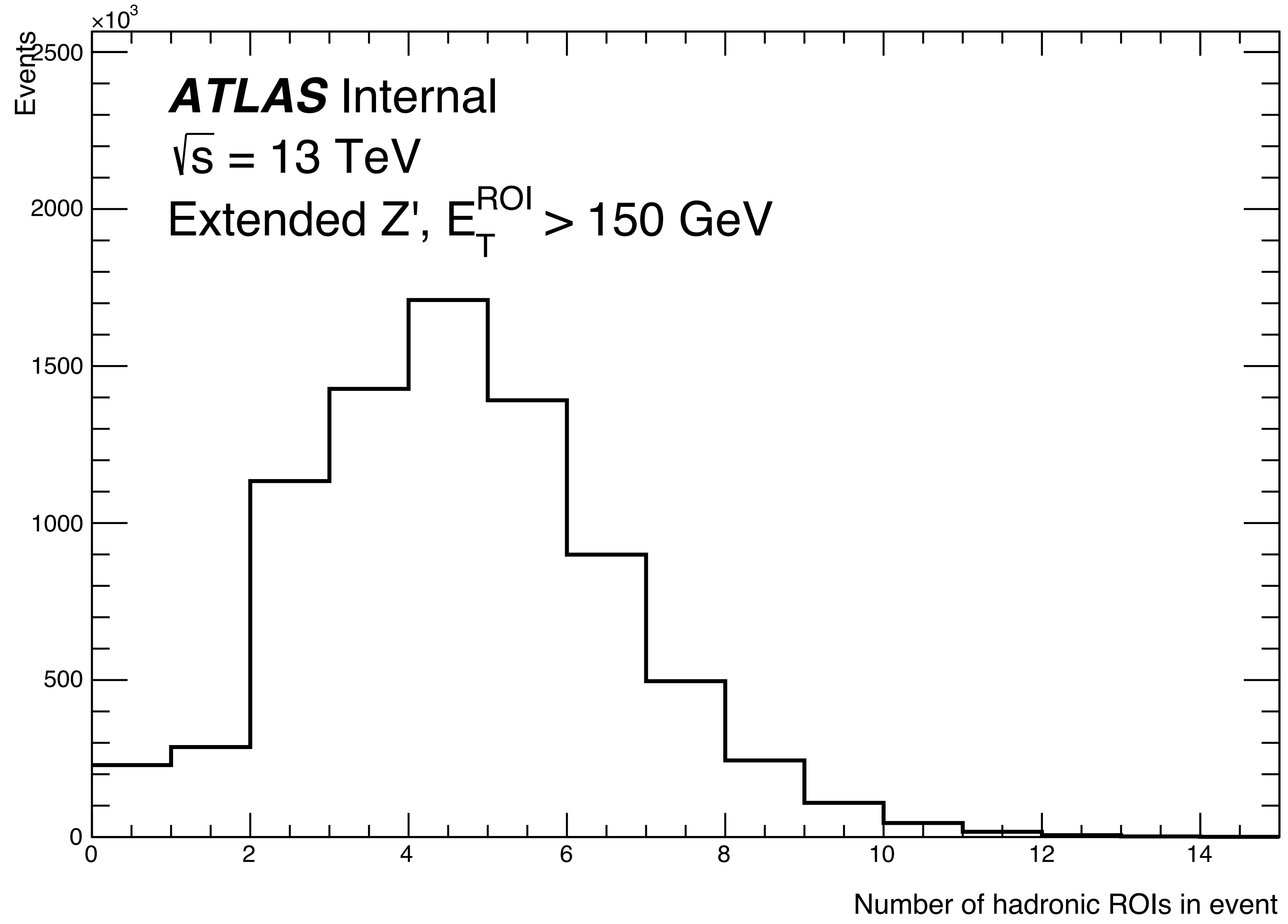
Can MaskFormer help the current ATLAS tracking?

MaskFormer in CTIDE

- Current number/position networks view each cluster individually and have no detailed knowledge of the other clusters on the track
  - Can we exploit the correlation that exists to improve both track assignment and local hit multiplicity/position estimation?
- Different problem than trackML; fewer tracks to reconstruct, fewer hits to deal with
  - Instead of tackling *scale*, can we instead use MaskFormer to tackle *complexity*?

```yaml
fields:
  # Coordinates in global frame
  - r
  - eta
  - phi
  - theta
  - x
  - y
  - z
  - u
  - v
  - s
  # ROI axis location in detector coords
  - roi_eta
  - roi_phi
  - roi_z0
  # Coordinates in ROI frame
  - deta
  - dphi
  - dR
  # Module global orientation
  - mod_norm_phi
  - mod_norm_theta
  # Module coordinates
  - mod_x
  - mod_y
  - mod_z
  - mod_eta
  - mod_phi
  # Module local coordinates
  - mod_loc_x
  - mod_loc_y
  # Other stuff
  - logcharge
  # Pixel specific fields
  - lshift
  - logchargemat
  - pitches
```

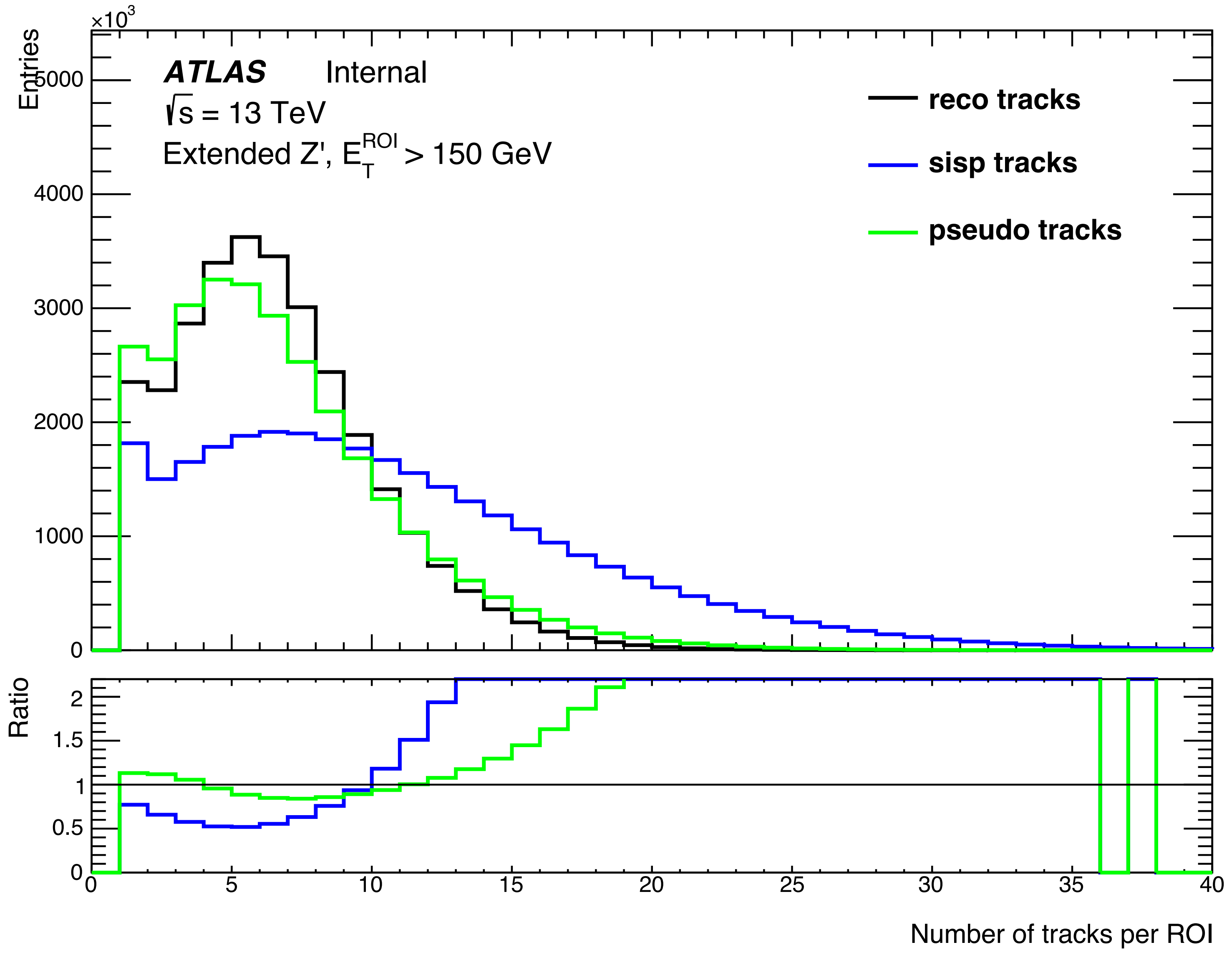
Hadronic ROIs

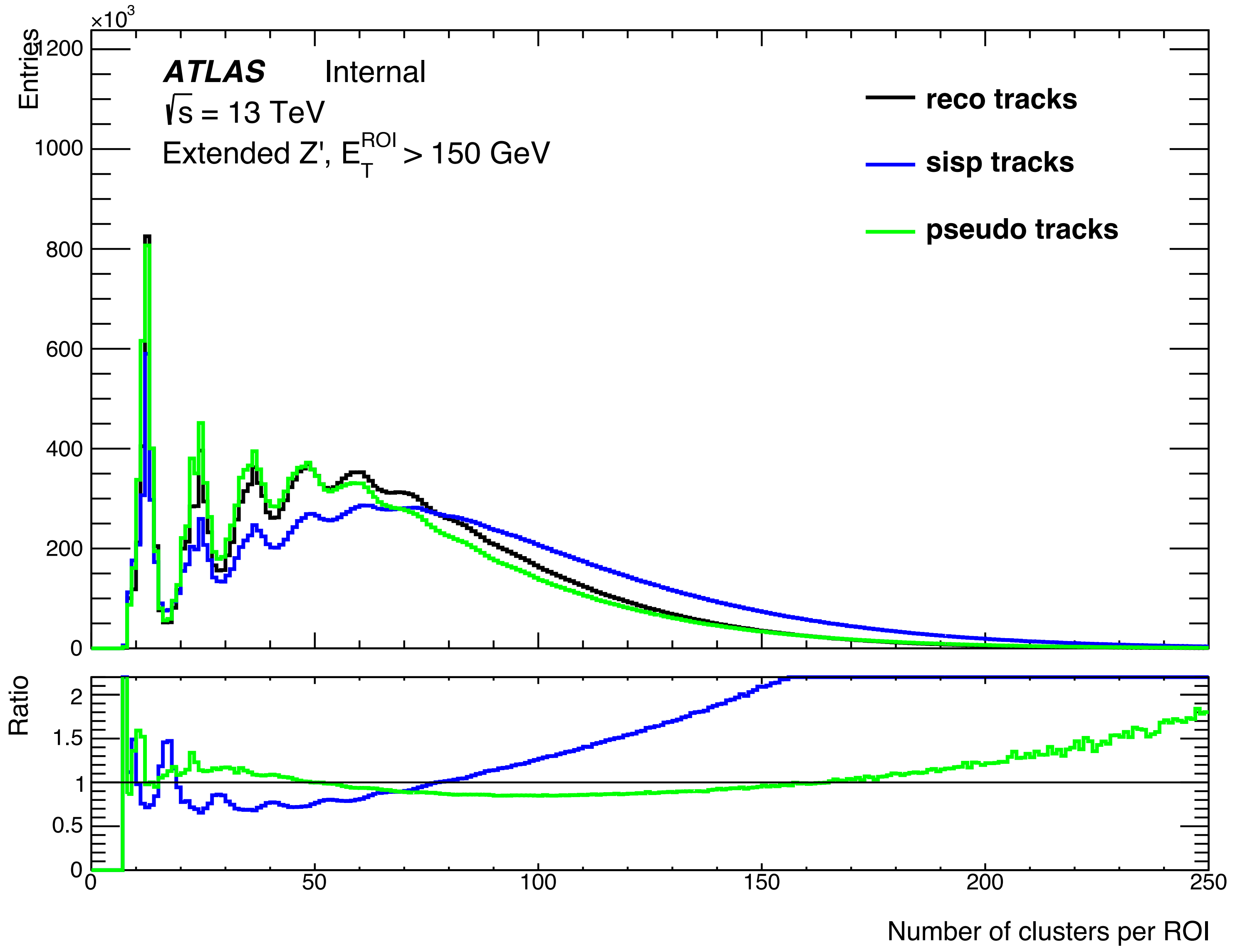


- Instead of full-event, each instance is a single hadronic ROI
  - Good test environment for tracking in dense environments
- Use dumper to get per-ROI `.parquet` files for easy manipulation in ML framework
- Ultimately need to feed this back into Athena to properly test things
  - Currently in the process of exporting model to ONNX
  - Try simply running ambiguity solving twice; once with default reco, once with MaskFormer

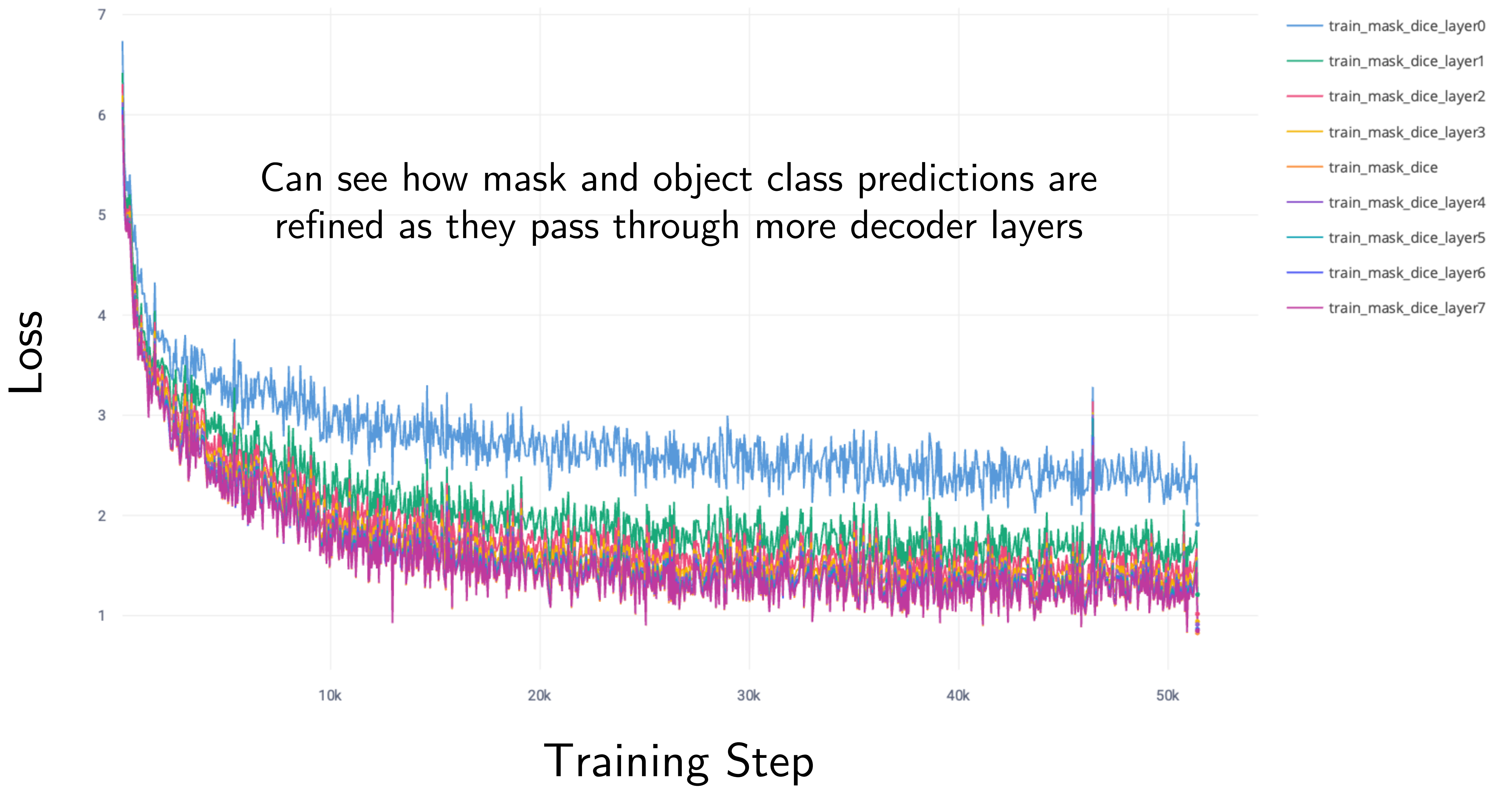
CTIDE MaskFormer

- No hit filtering applied
- Train model to predict:
  - Track-to-hit assignment based on pseudo tracks (primary task)
  - Track parameters (q/p, η, φ, d_0, z_0)
  - Track-hit parameters (local x/y, local φ/θ, energy)
- Number of clusters roughly peaking at multiples of 12 clusters (i.e. number of clusters for a "perfect" track)
  - 4 pixel + 4x2 SCT = 12 clusters



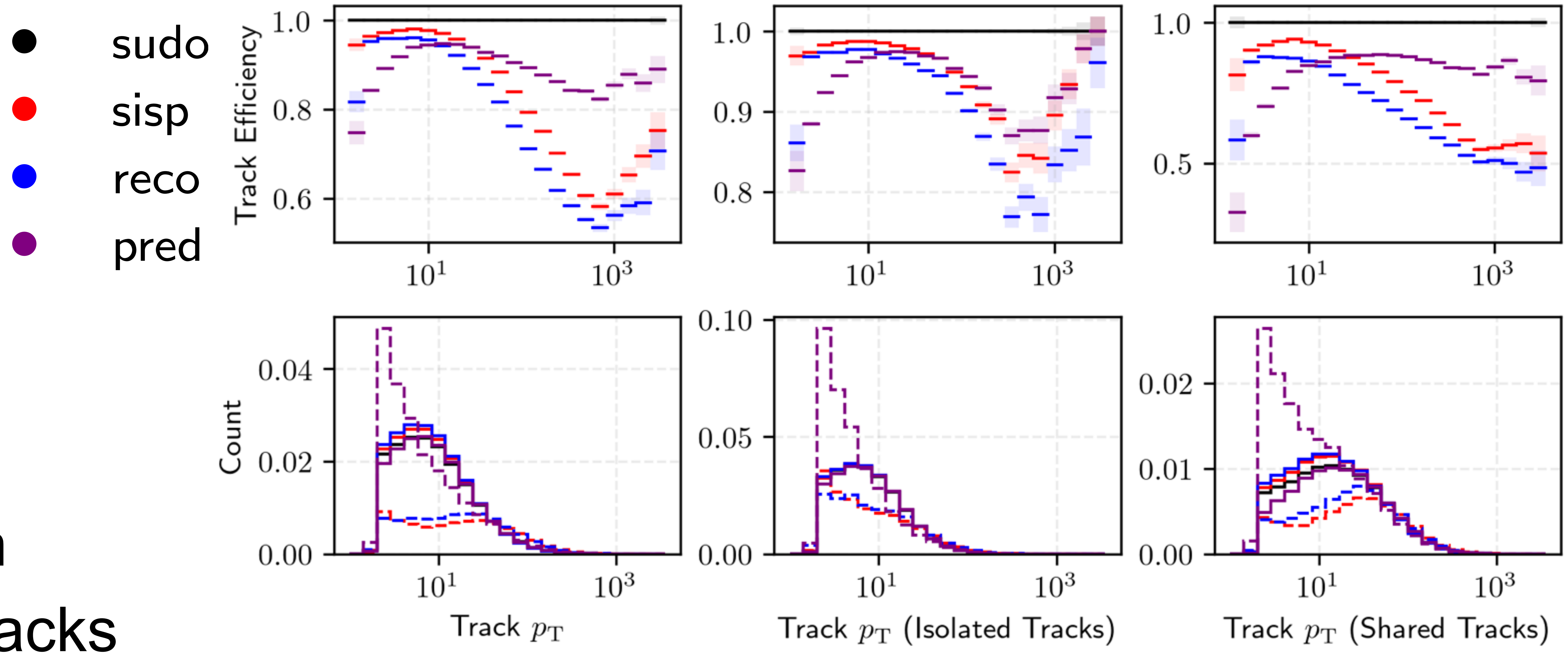


Training setup

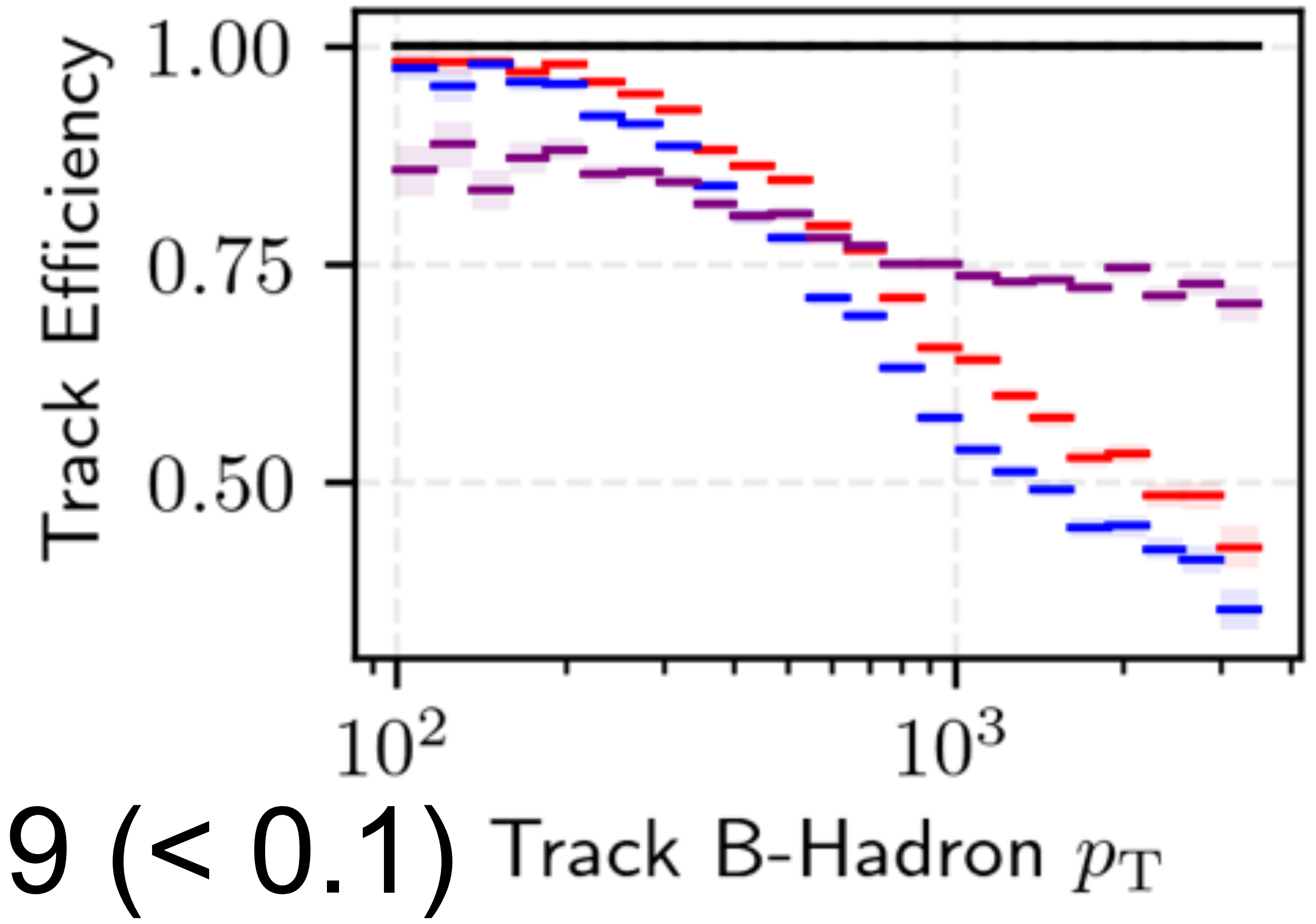


- Use 1M ROIs, maximum 32 pseudo tracks in ROI
- 100k ROIs used for testing
- Require pseudo tracks to have p_T > 500 MeV, ROI $|\eta| < 4$, all sharing and noise hits allowed
- Also regress track p_T, ROI relative η and ϕ
- Use a $d = 256$-dimensional latent space
- Pixel and SCT hits fed through 40-layer transformer encoder
- These are concatenated and fed through an 8-layer object decoder

Tracking results



- Predicted tracks are uniquely matched to pseudo tracks to maximize overall TMP score
- Matching requirement between predicted tracks and pseudo tracks is TMP >= 0.75
- Efficiency: fraction of pseudo tracks that are correctly matched to a predicted track
- Low-p_T performance expected as we train on high-p_T ROIs
- “Shared tracks” refer to a given track with >=1 shared hit
- Solid (dashed) lines show distributions for tracks with TMP > 0.9 (< 0.1)

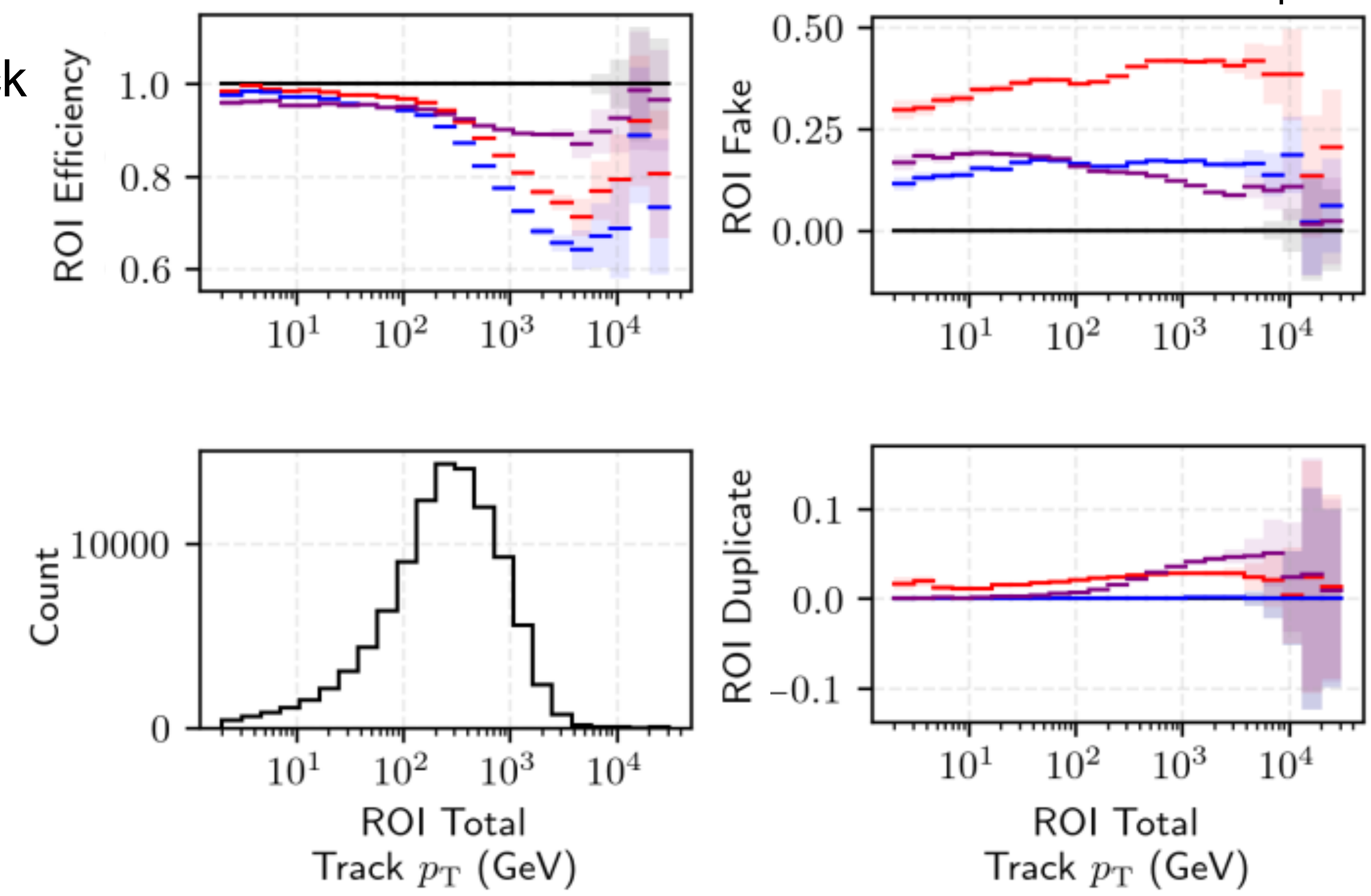


Nikhef

Tracking results (continued)

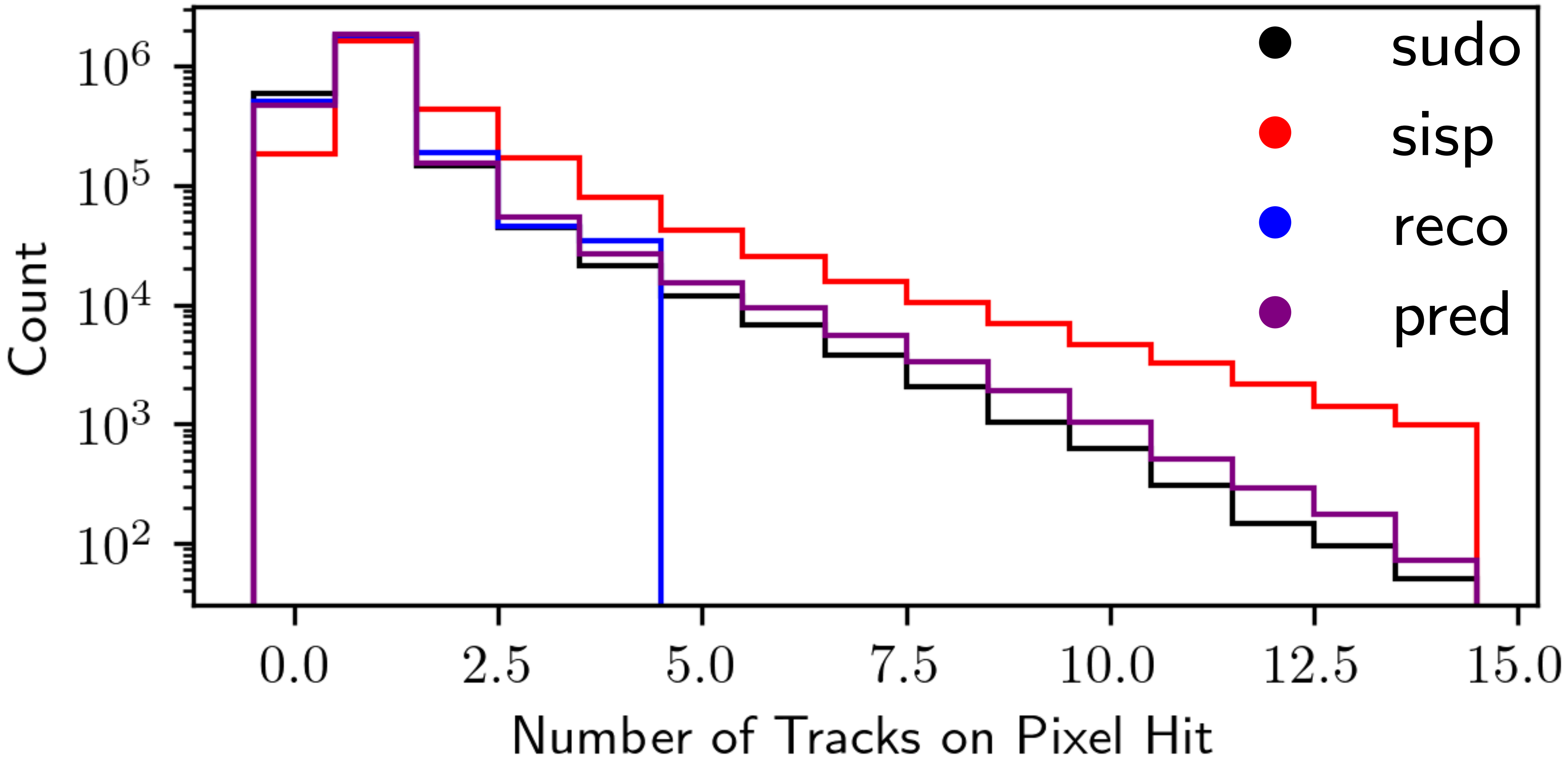
Reminder:
We are restricting ourselves to hadronic ROIs with p_T > 150 GeV (which is why e.g. nominal reco fake rate is higher than typical plots)

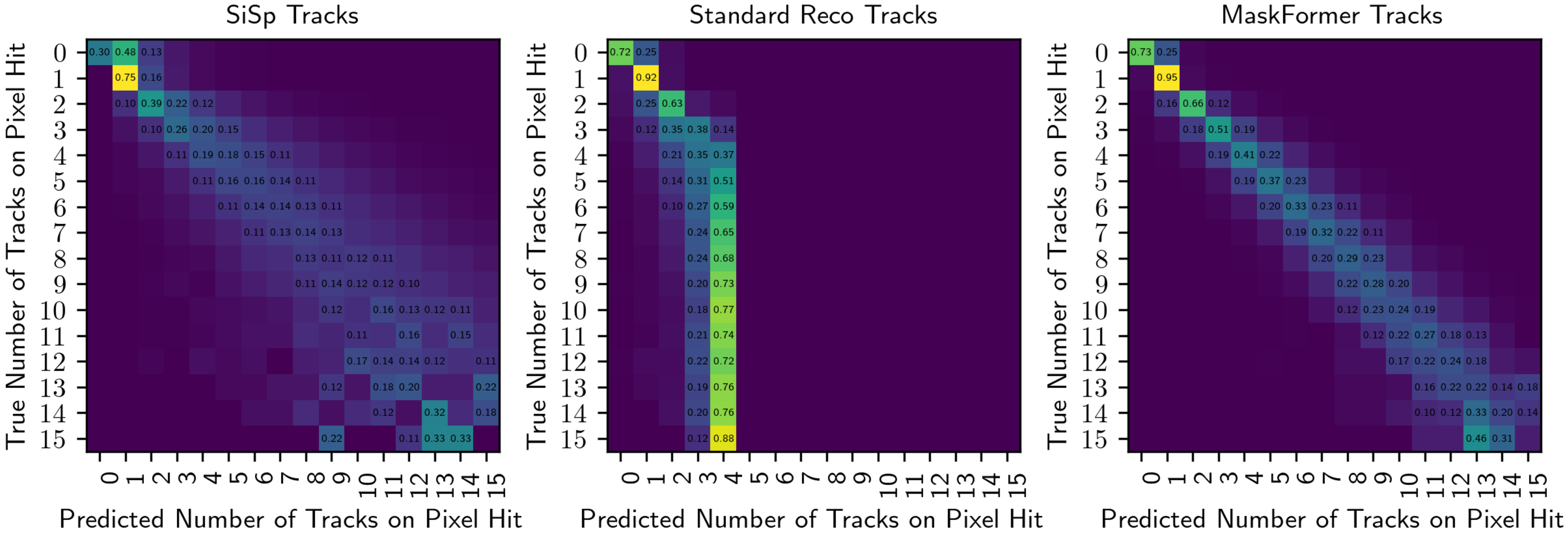
- Fake rate: fraction of predicted tracks that are not matched to any pseudo track
- Duplicate rate: fraction of predicted tracks that are matched to more than one pseudo track
- Note: fake/duplicate rate plotted as a function of total ROI track p_T to avoid caveats related to plots vs. predicted track p_T
- Slightly better fake rate than nominal reco, but significantly higher duplicate rate (sum of fake and duplicate rate similar)



Pixel splitting performance

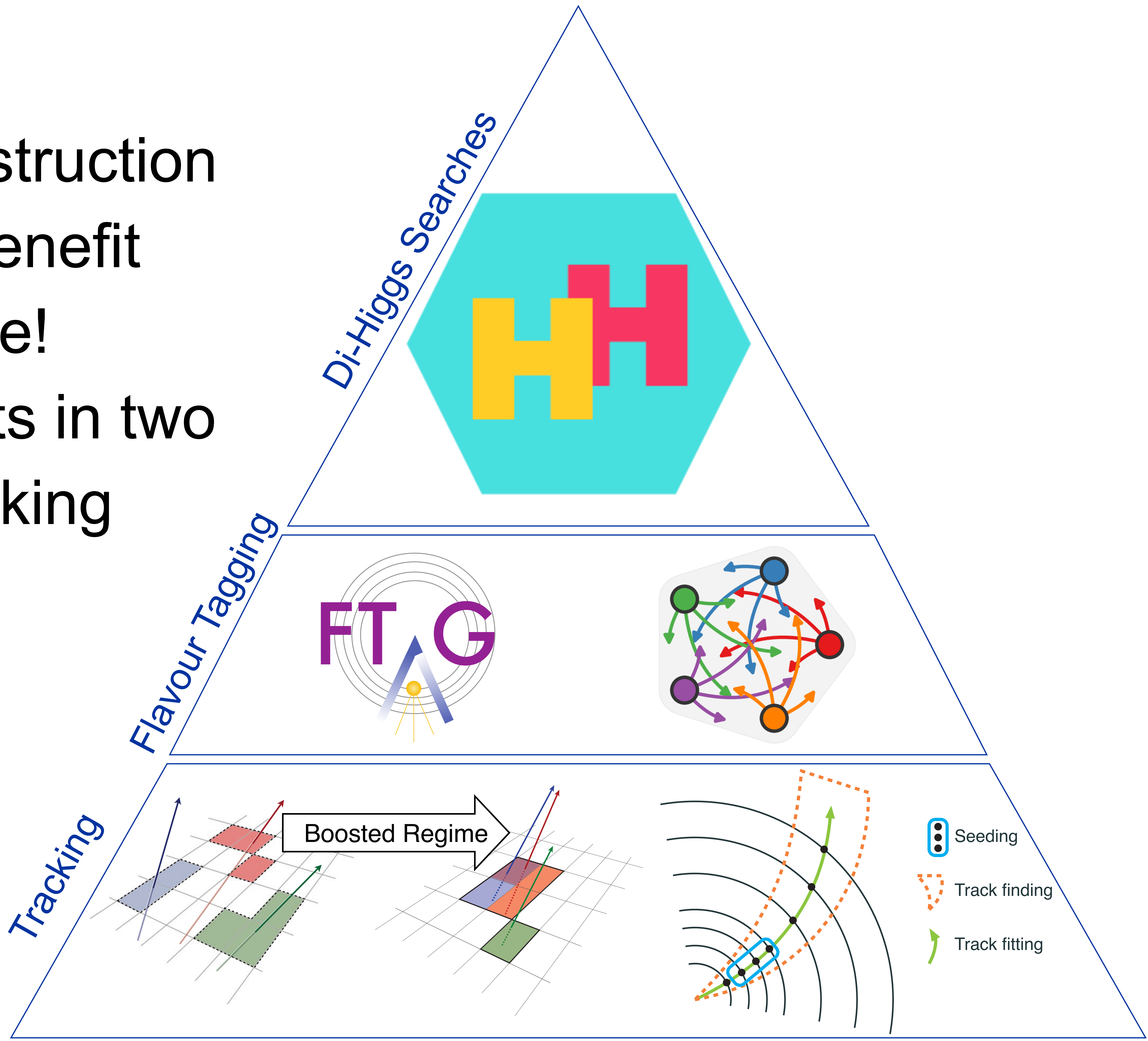
- Model handles sharing so implicitly replaces the pixel splitting network
- Able to recover shared hits that are killed in the nominal reconstruction pipeline





Summary & Outlook

- Tracking underpins the entire reconstruction chain; improvements here directly benefit the entire ATLAS physics programme!
- MaskFormer shows promising results in two distinct environments: full-event tracking and CTIDE
- Next steps:
  - Integration into Athena/ACTS
  - Start looking into ITk samples
  - Improve model efficiency/capability
  - Integration with particle flow



Nikhef

Thank you!
Merci!

Questions?

Truth match probability (TMP)

$$\text{Score(track)} = \sum_{\text{hits}} \text{Weight(hit)}$$

Weights: Pixel = 10, SCT = 5, TRT = 1

$$\text{TMP} = \frac{\text{Score(track)}}{\text{Score(truth)}}$$

Pseudo tracks

- Pseudo tracks (PT) are reconstructed with ideal pattern recognition, i.e. use truth information instead of pattern recognition and the ambiguity solver; correct clusters, reco hit positions
- At the moment, PT are only available for extended Z’ sample, hence focusing on high-p_T regime
- Previous studies show PT greatly improve the GNN performance