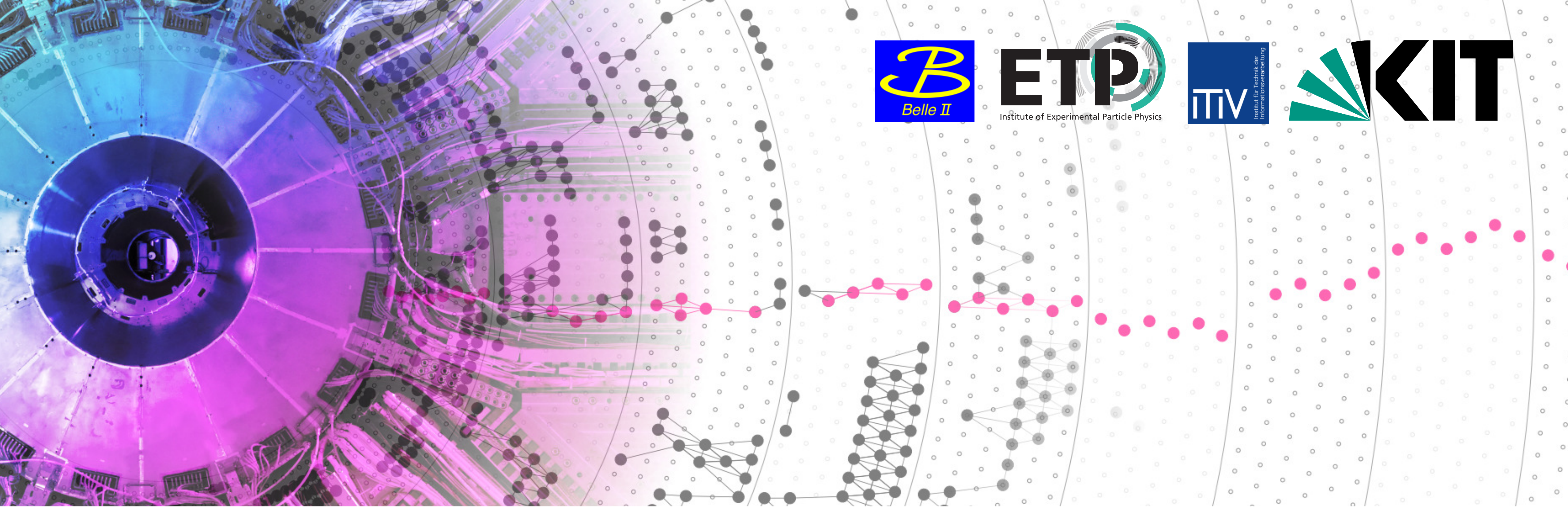


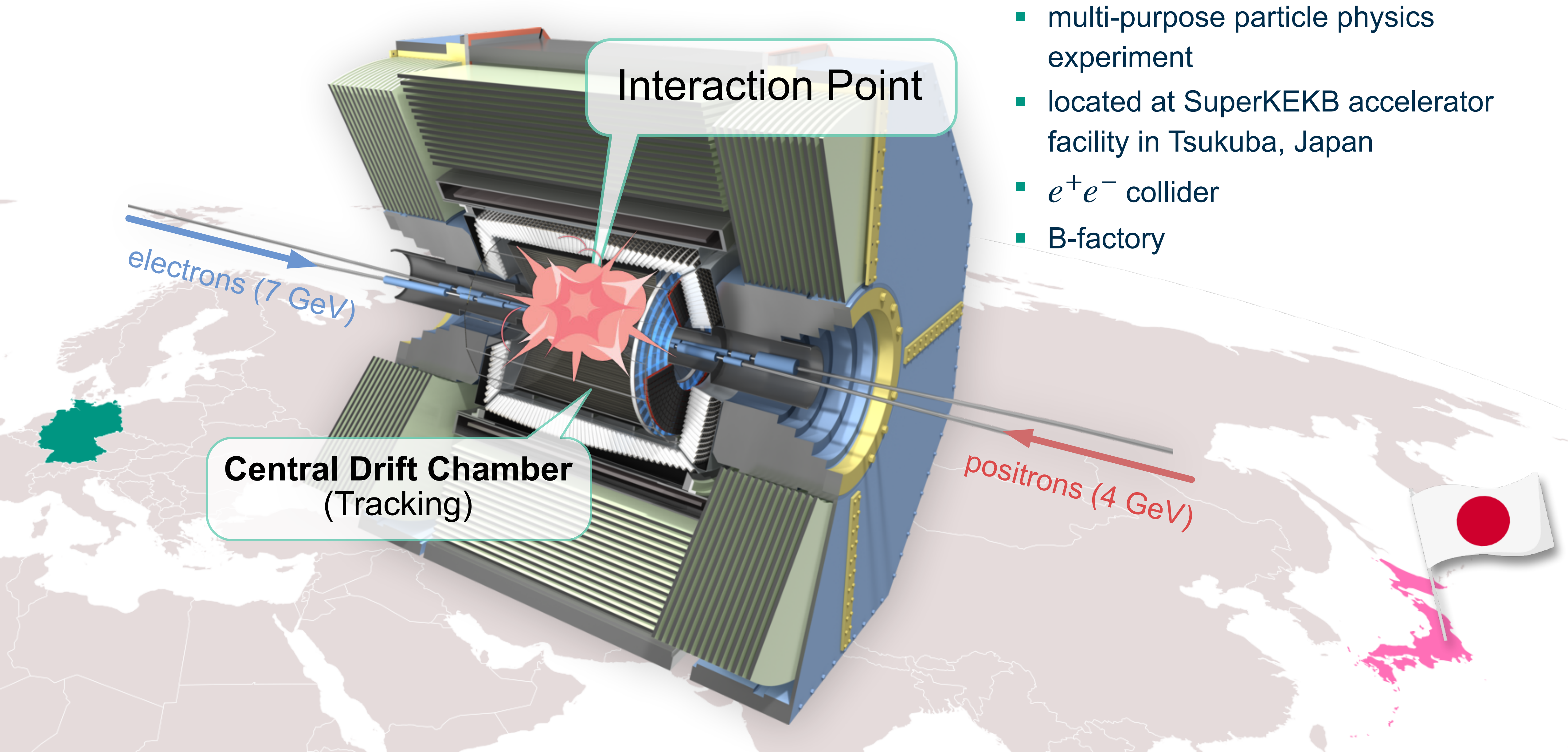


IPE Seminar

Hardware-Aware GNN Co-Design for Low-Latency FPGA Implementation at Belle II

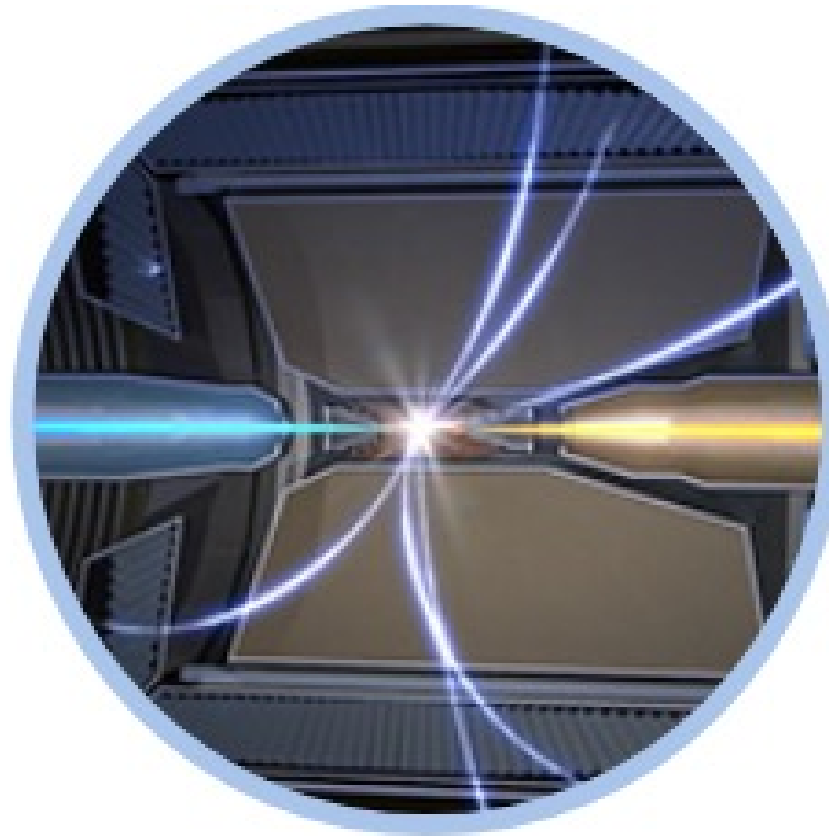
Greta Heine, Fabio Mayer, Marc Neu, Giacomo De Pietro Jürgen Becker, Torben Ferber | November 27, 2025

Belle II Experiment



- multi-purpose particle physics experiment
- located at SuperKEKB accelerator facility in Tsukuba, Japan
- e^+e^- collider
- B-factory

The Belle II Trigger System



interesting physics
<3kHz



Bunch Crossings:
250 MHz

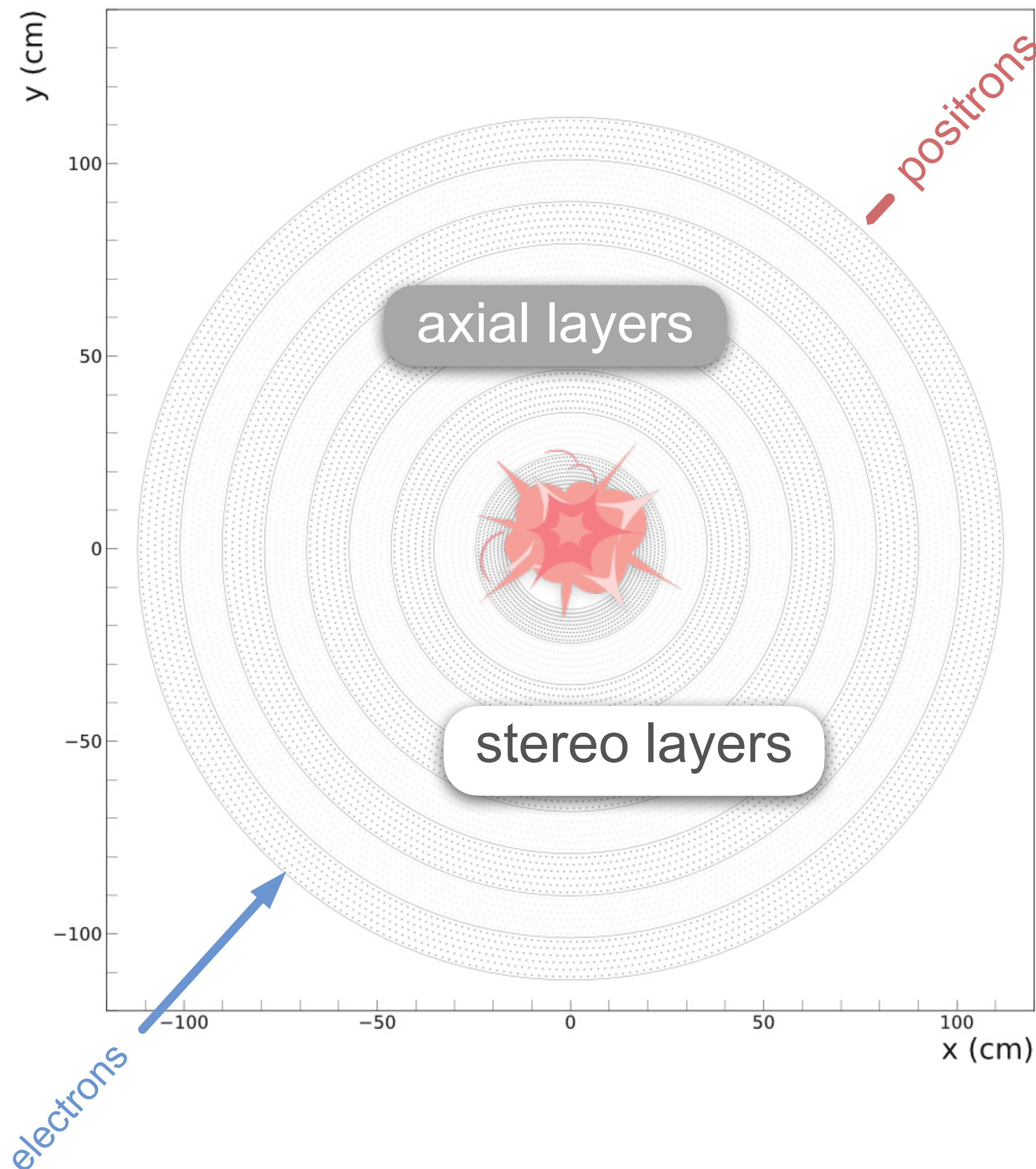
Level 1 Trigger:
30 kHz, <5 μ s latency

High Level Trigger:
10 kHz, 1.8 Gb/s

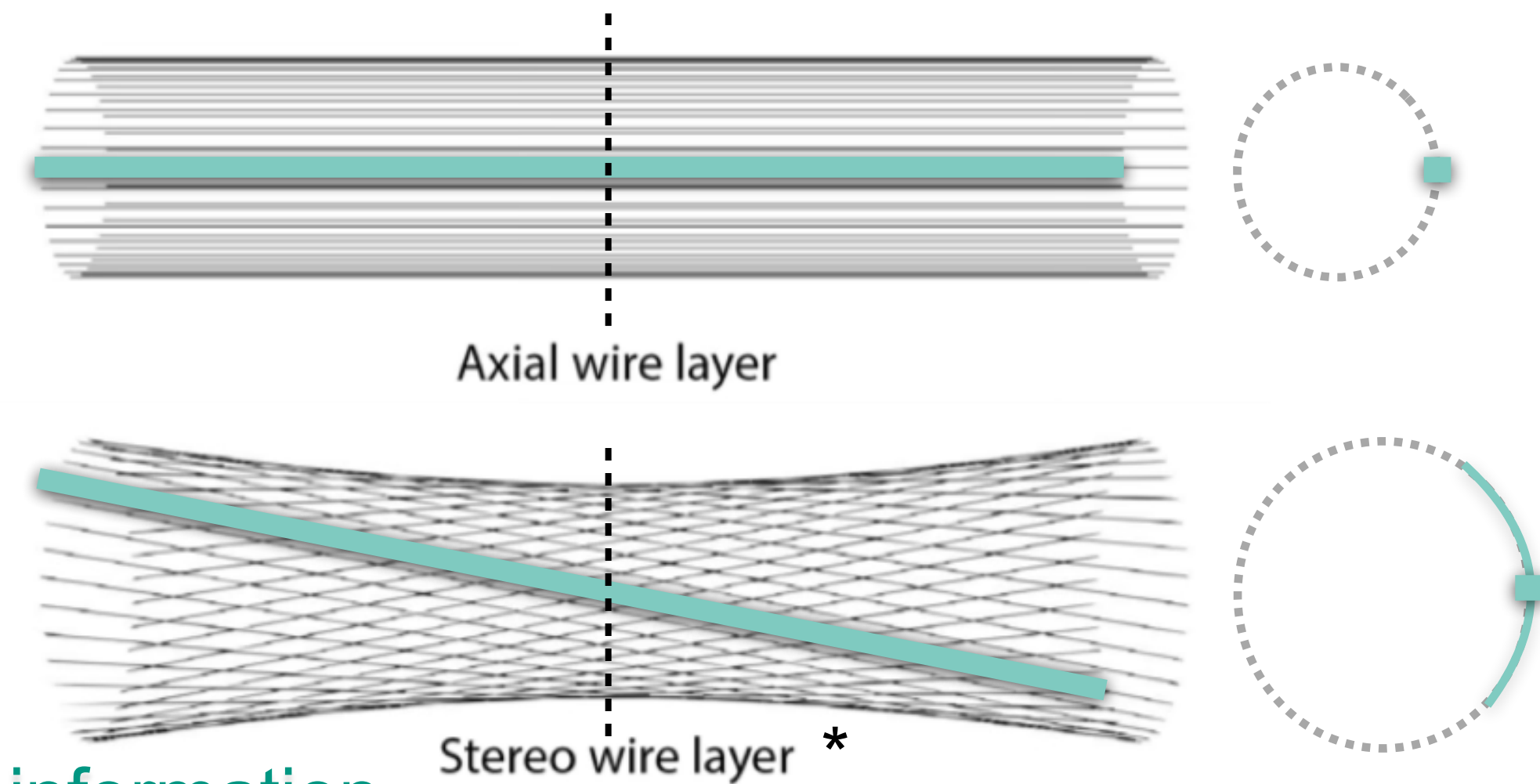
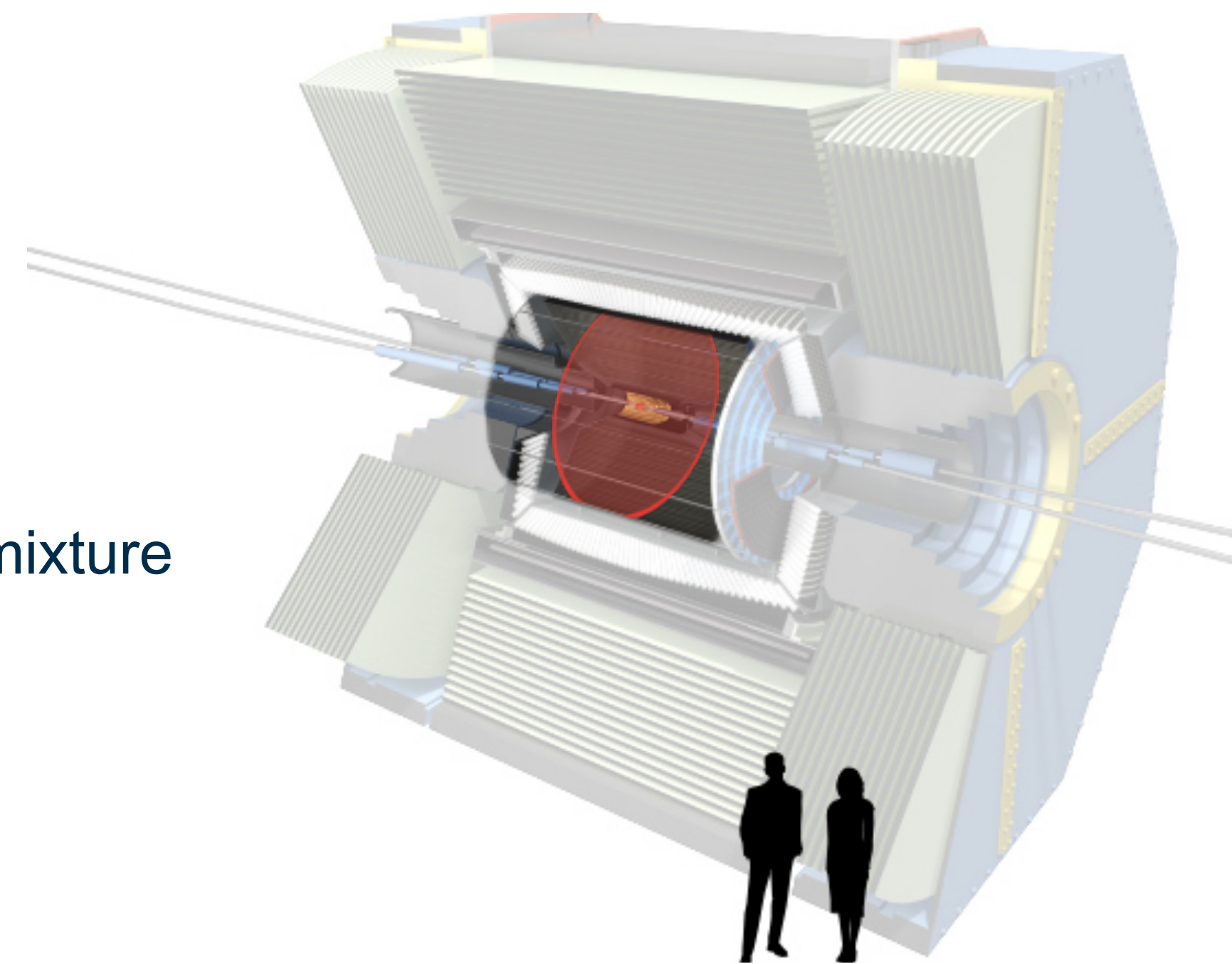
graphic taken from Isabel Haide

- different trigger levels have to reduce the data stream to match DAQ limitations
- needs fast and efficient background suppression to avoid losing physics signals
- L1 trigger is the first line of defense against backgrounds

Central Drift Chamber (CDC)



- gaseous tracking detector
- filled with helium-based gas mixture
- xy projection
- $\approx 15\,000$ sense wires
- 9 super layers
- alternating axial and stereo super layers



3D information

*rotation is hugely exaggerated for illustration

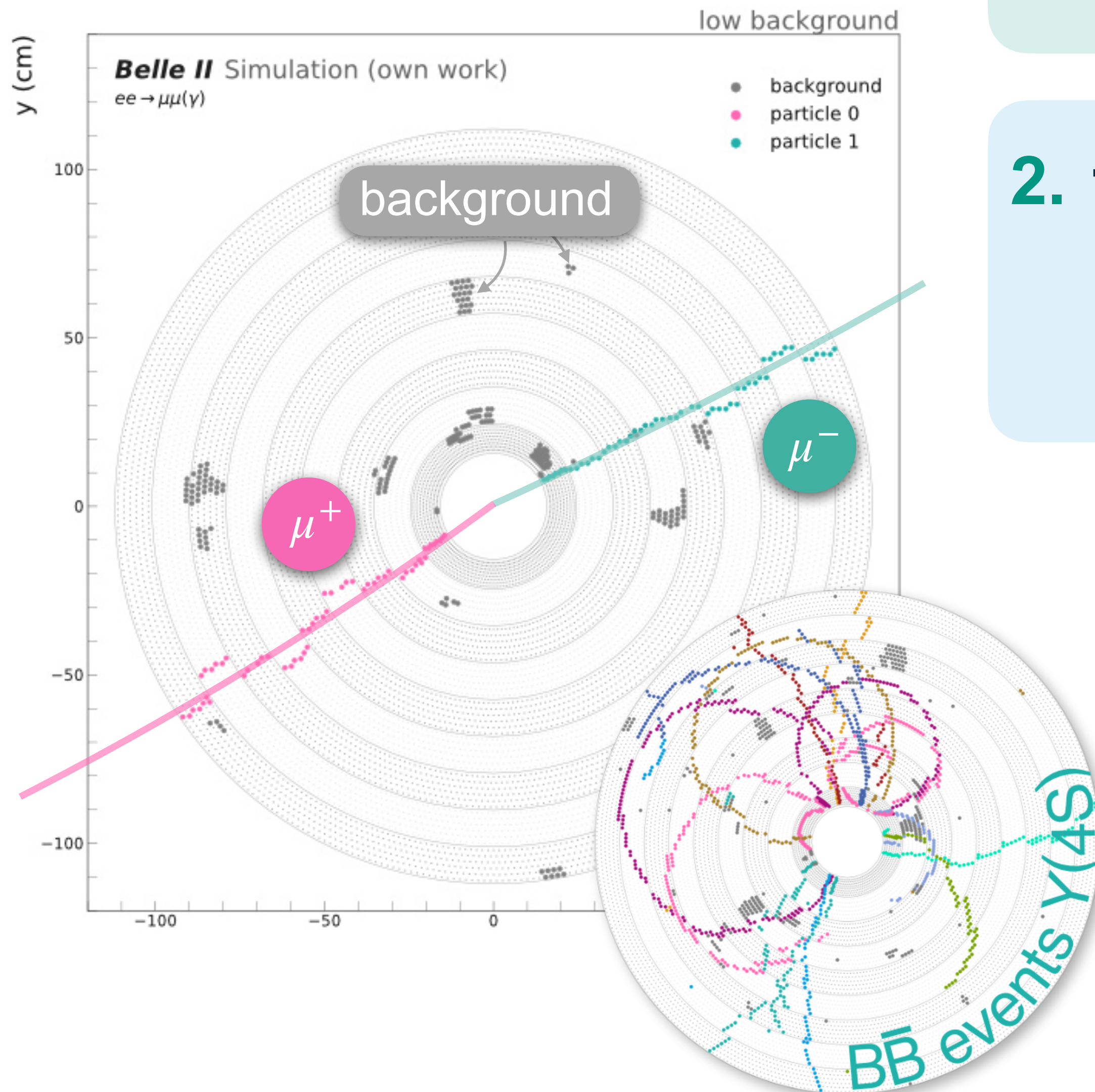
CDC Track Trigger

1. track finding:
identify and assign signal hits to tracks

2. track fitting:
reconstruct track parameters
(curvature $\propto 1/p_T$, z position, ...)

Challenges:

- beam-induced background
up to wire 20% wire occupancy
- suppression of background tracks
tracks that are not from the interaction point at (0,0,0)
→ requires good z resolution
- strict latency budget $< 5 \mu\text{s}$

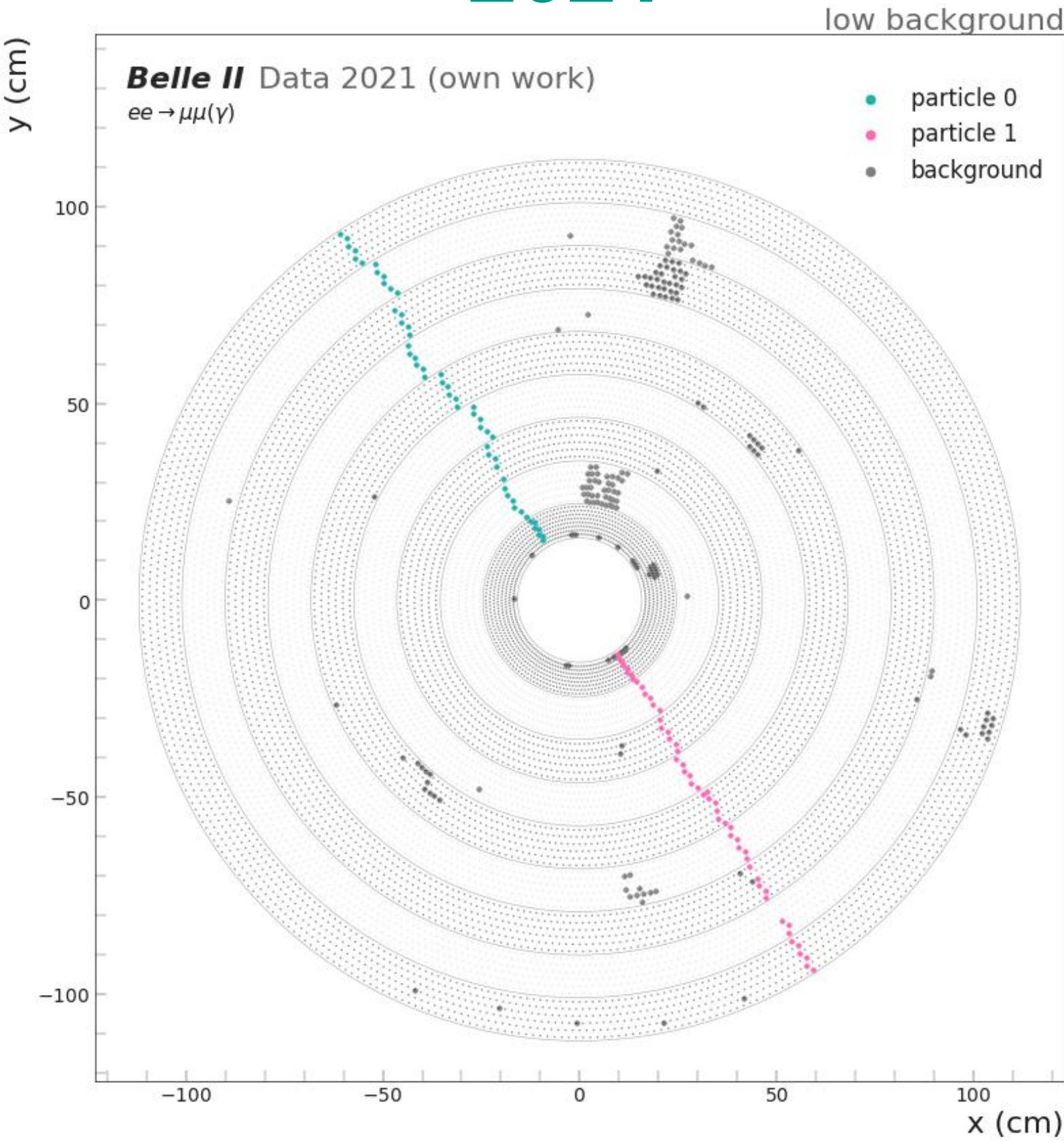




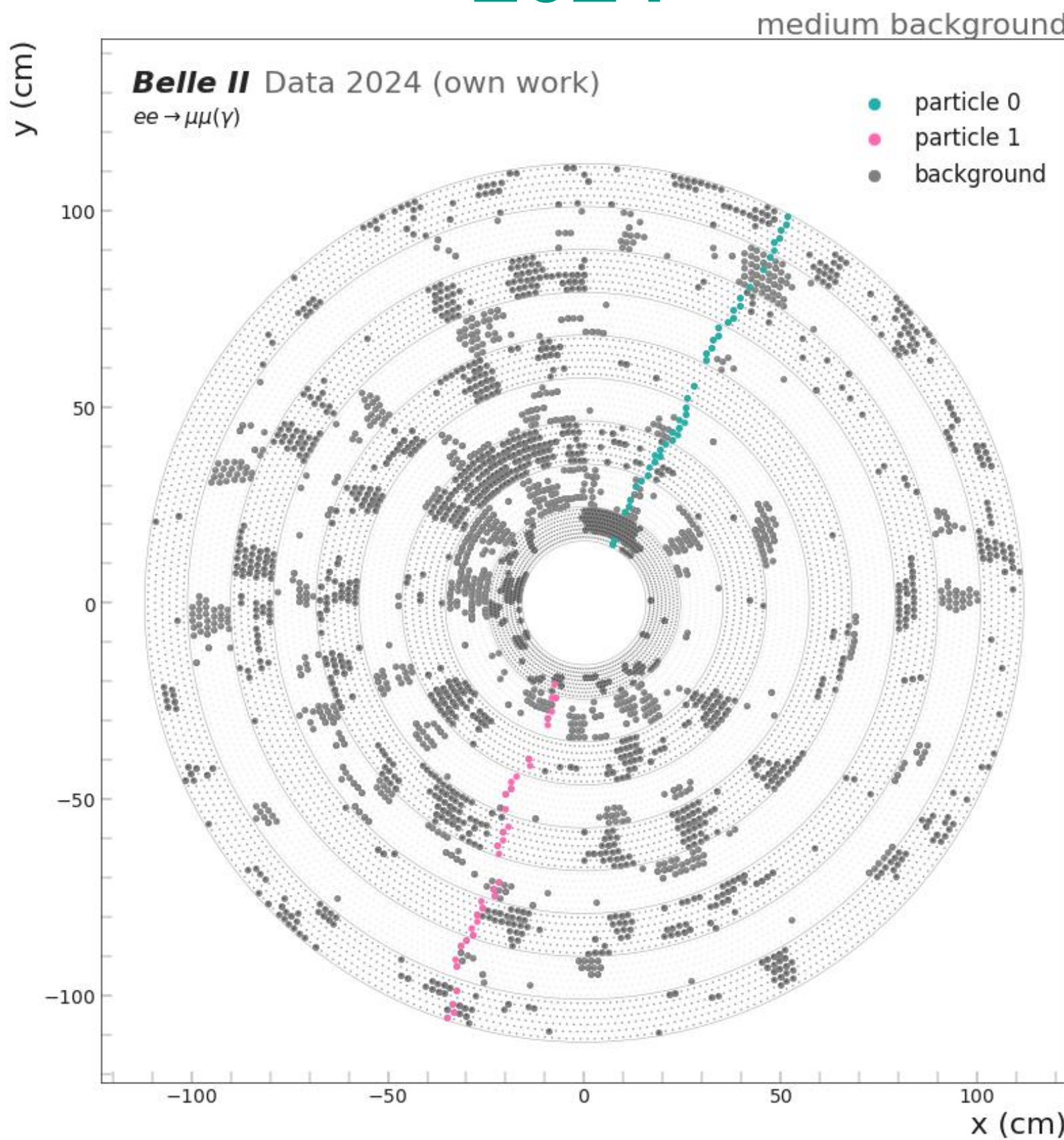
The Mission:
Gotta catch 'em all!™

Evolving Background Levels

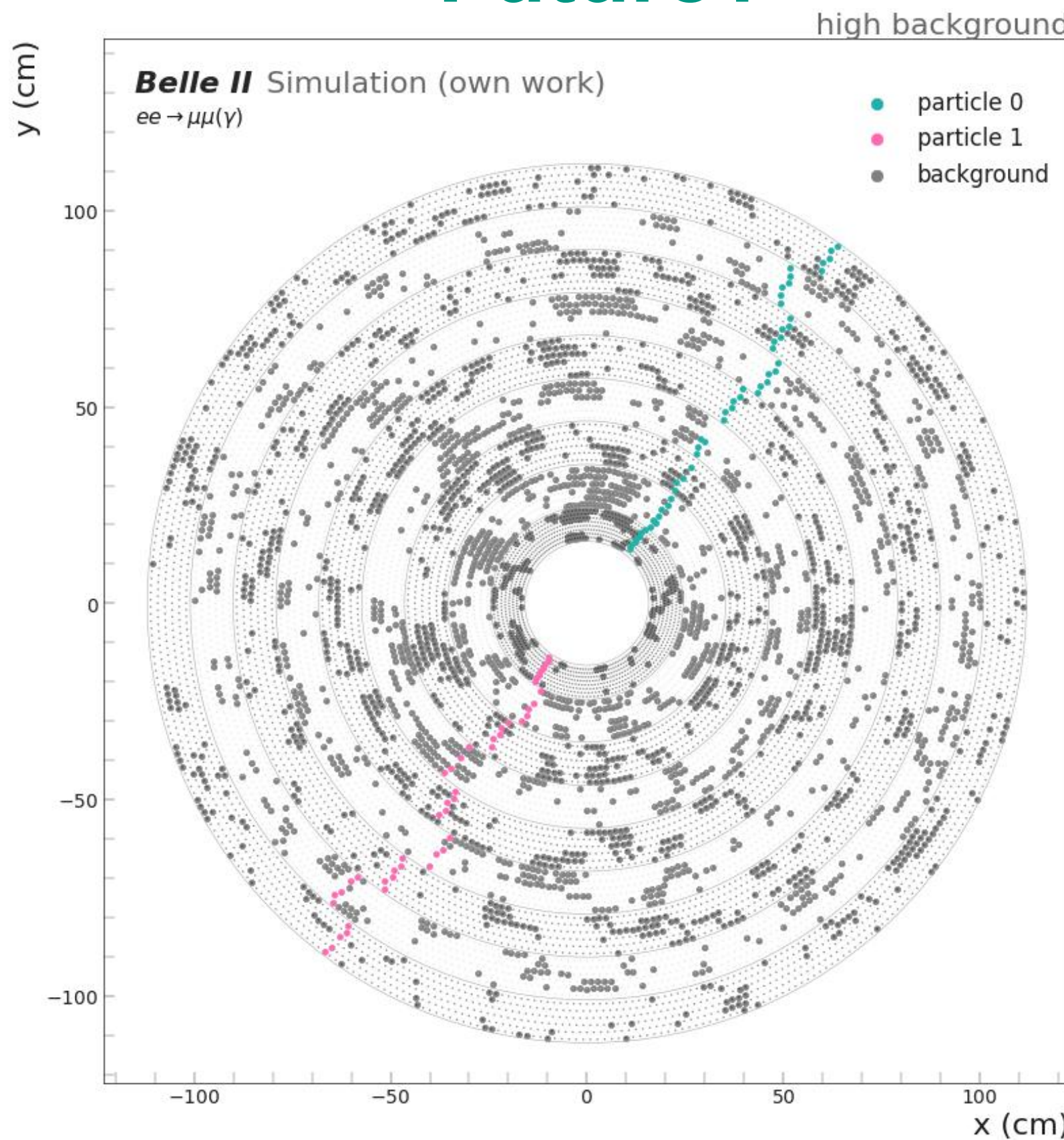
2021



2024



Future?



exp 22 (2021)
≈ 200 hits

exp 35 (2024)
≈ 1900 hits

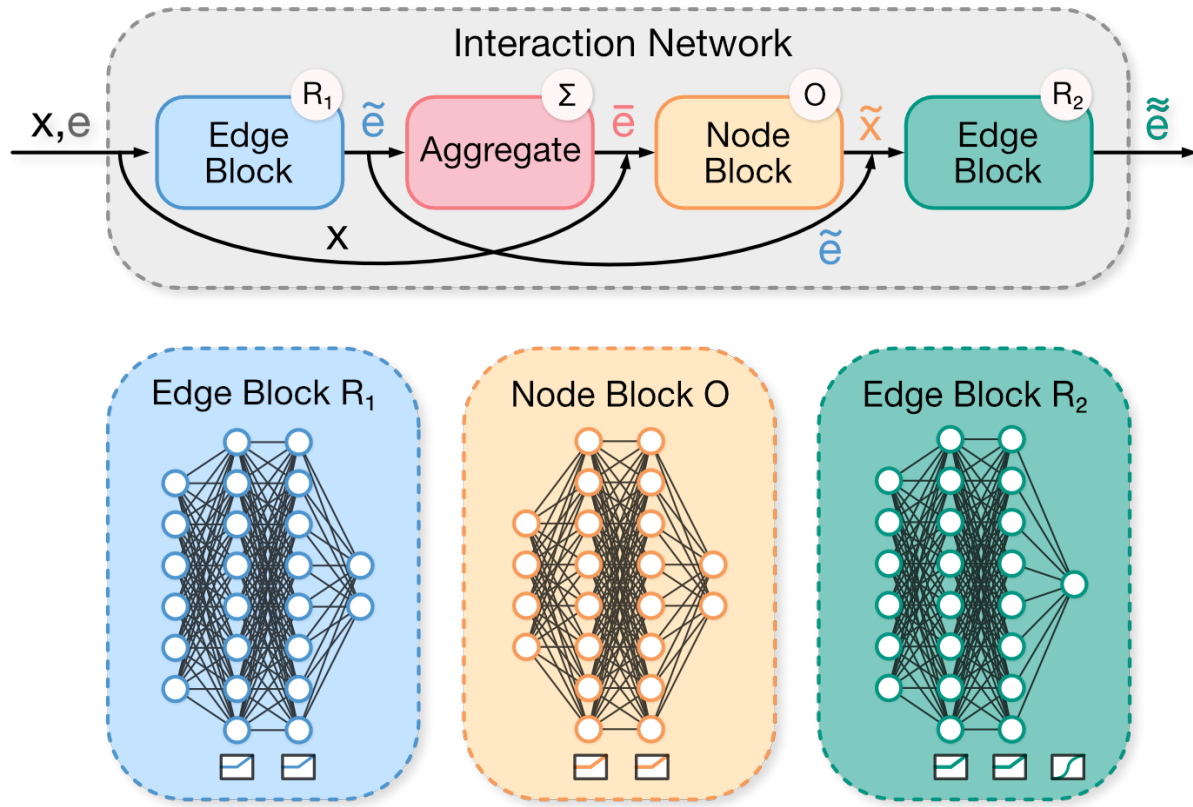
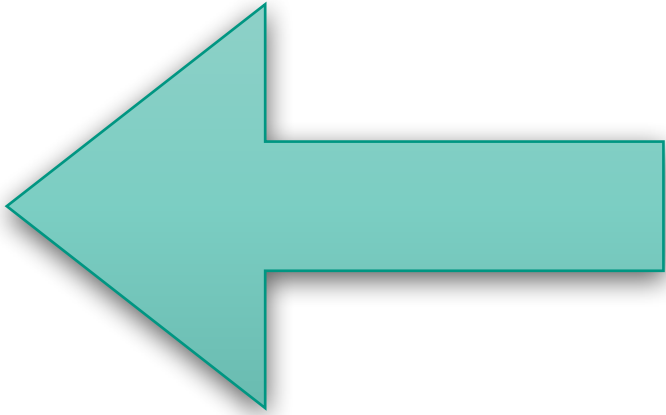
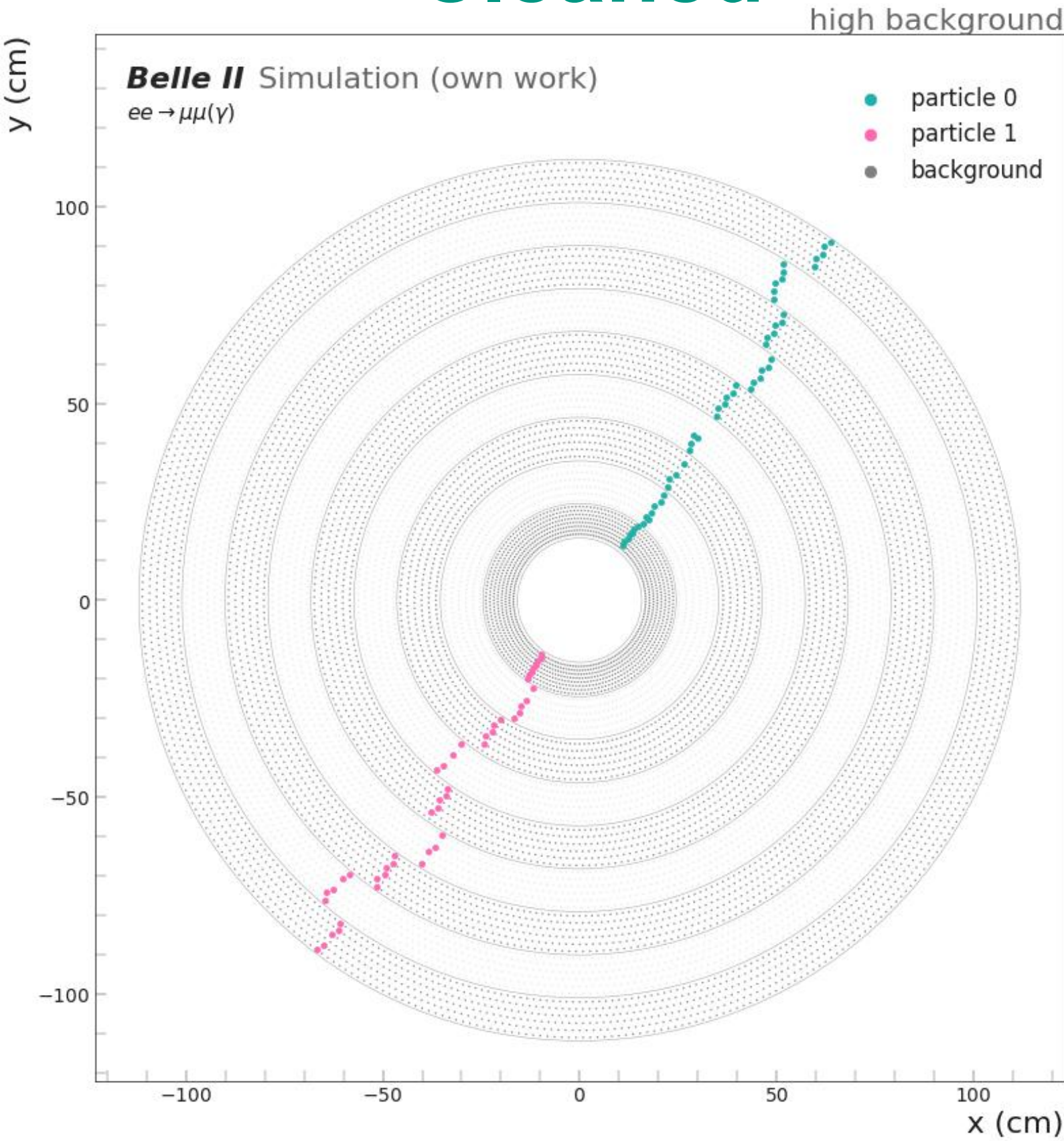
≈2800 hits

Future

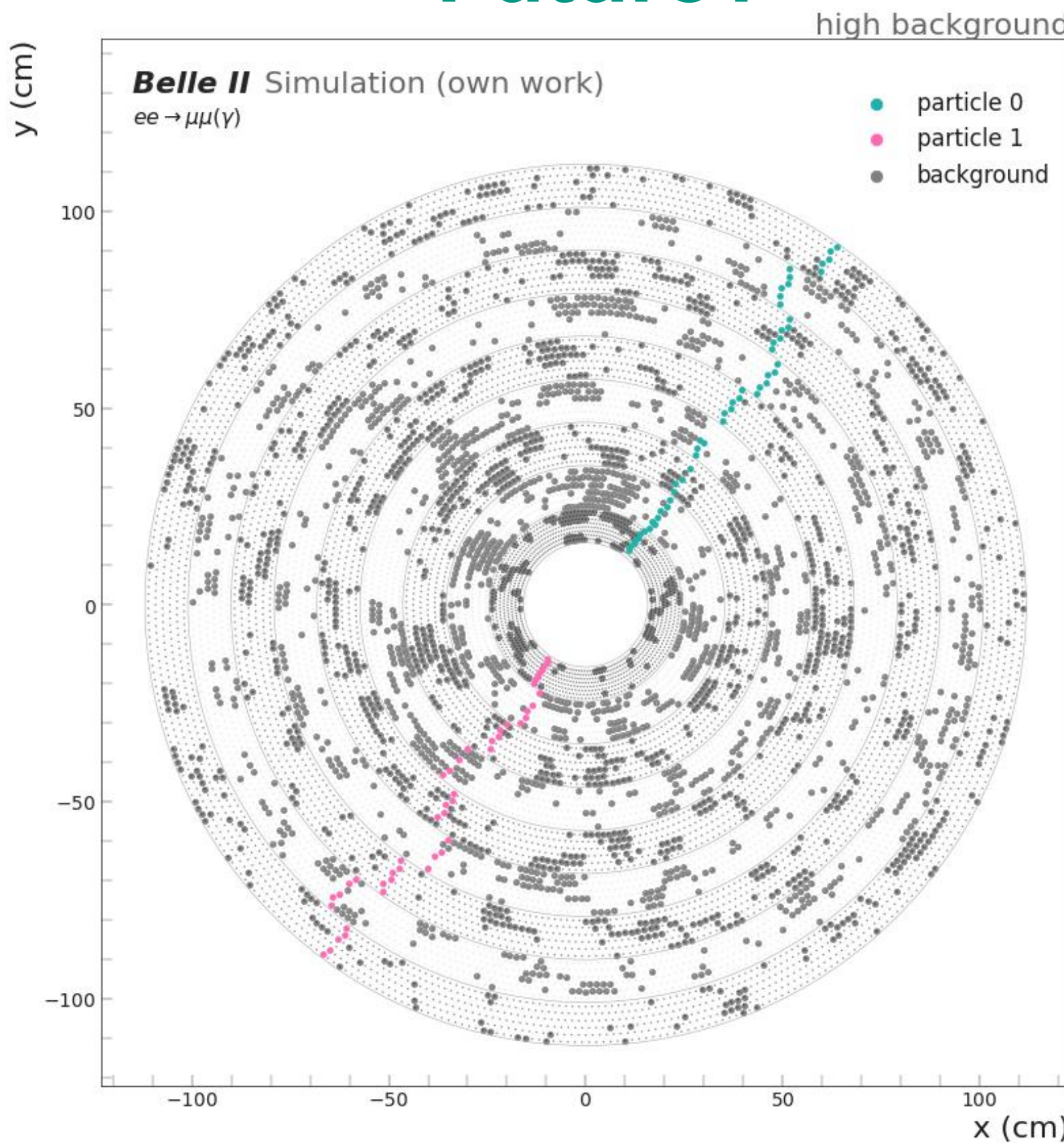
Background Level

Evolving Background Levels

Cleaned



Future?



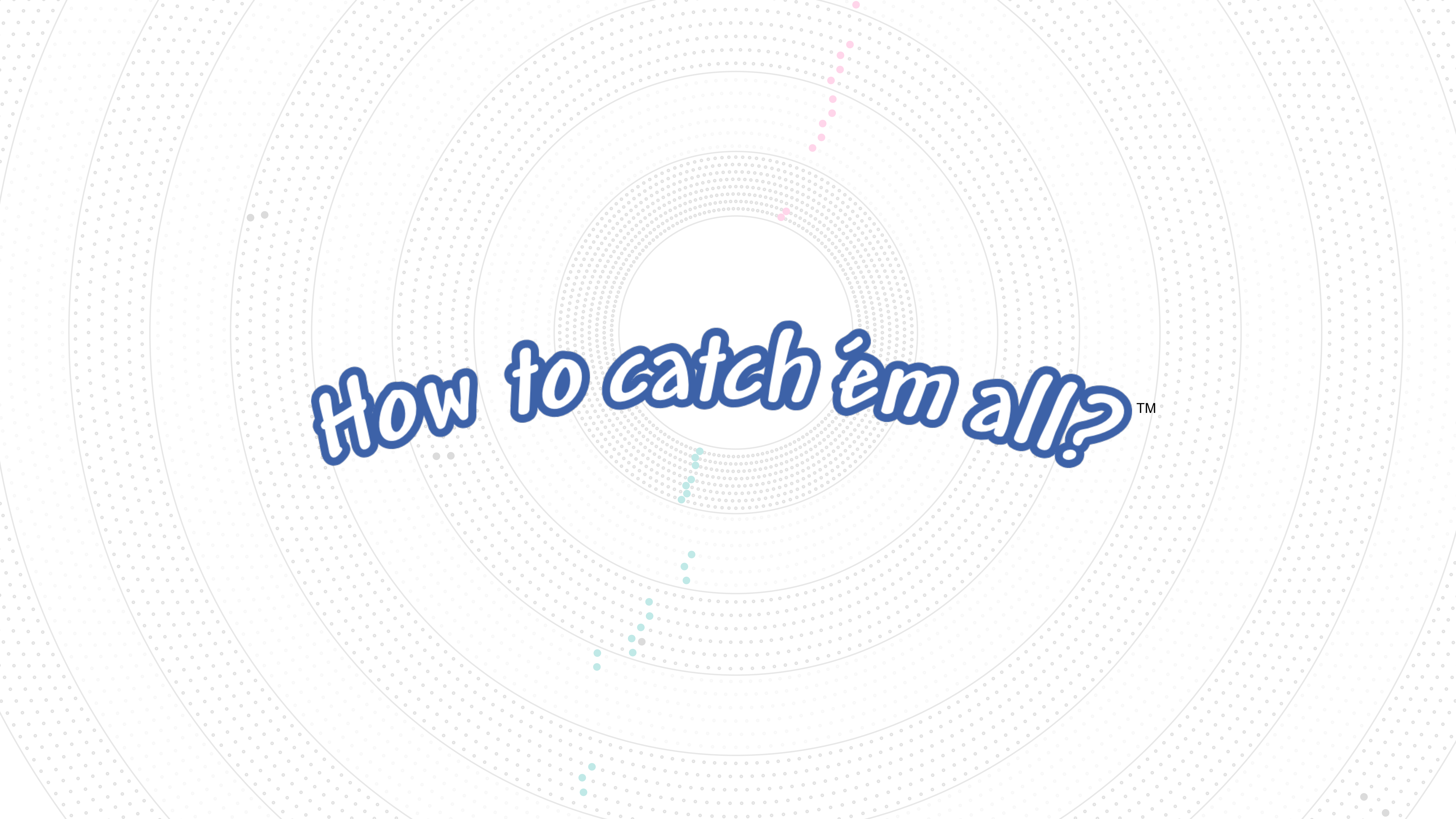
exp 22 (2021)
≈ 200 hits

exp 35 (2024)
≈ 1900 hits

≈ 2800 hits

Future

Background Level



How to catch 'em all?™

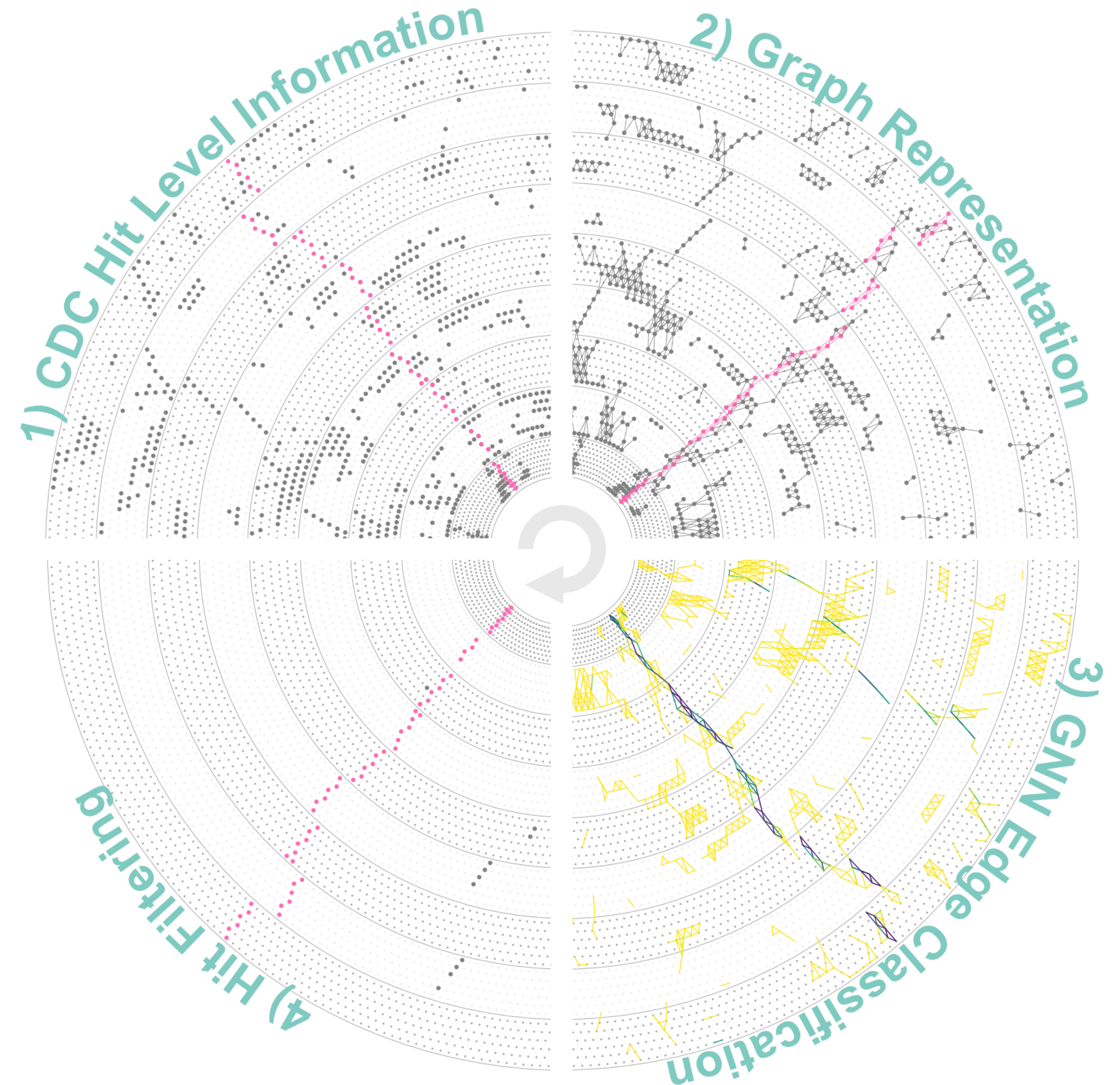
Graph Neural Network (GNN) Hit Clean-up

Hit filtering steps:

1. extract and pre-filter CDC hits
2. graph representation of hits
3. GNN edge classification on graphs
4. hit filtering

track finding

track fitting



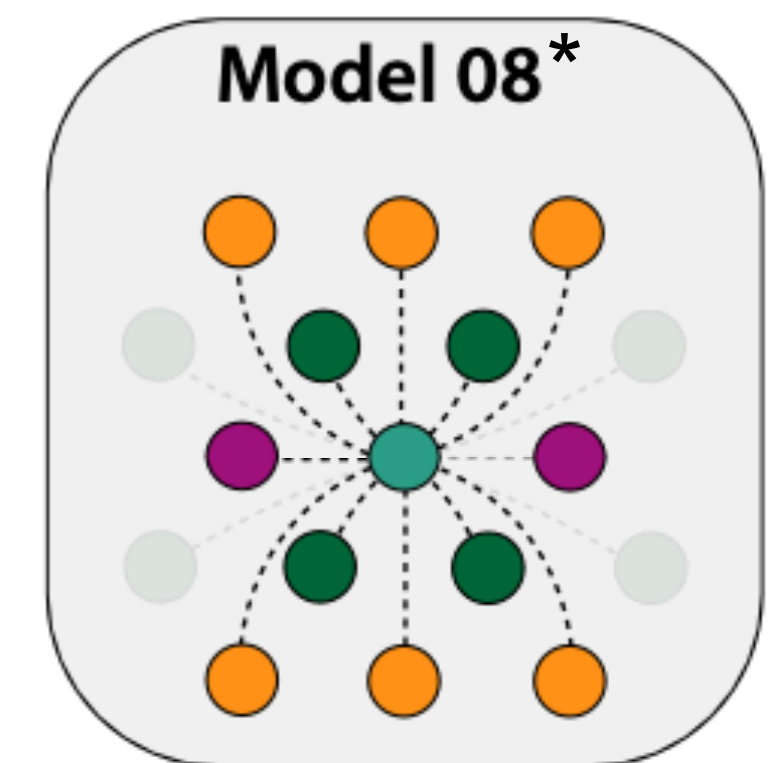
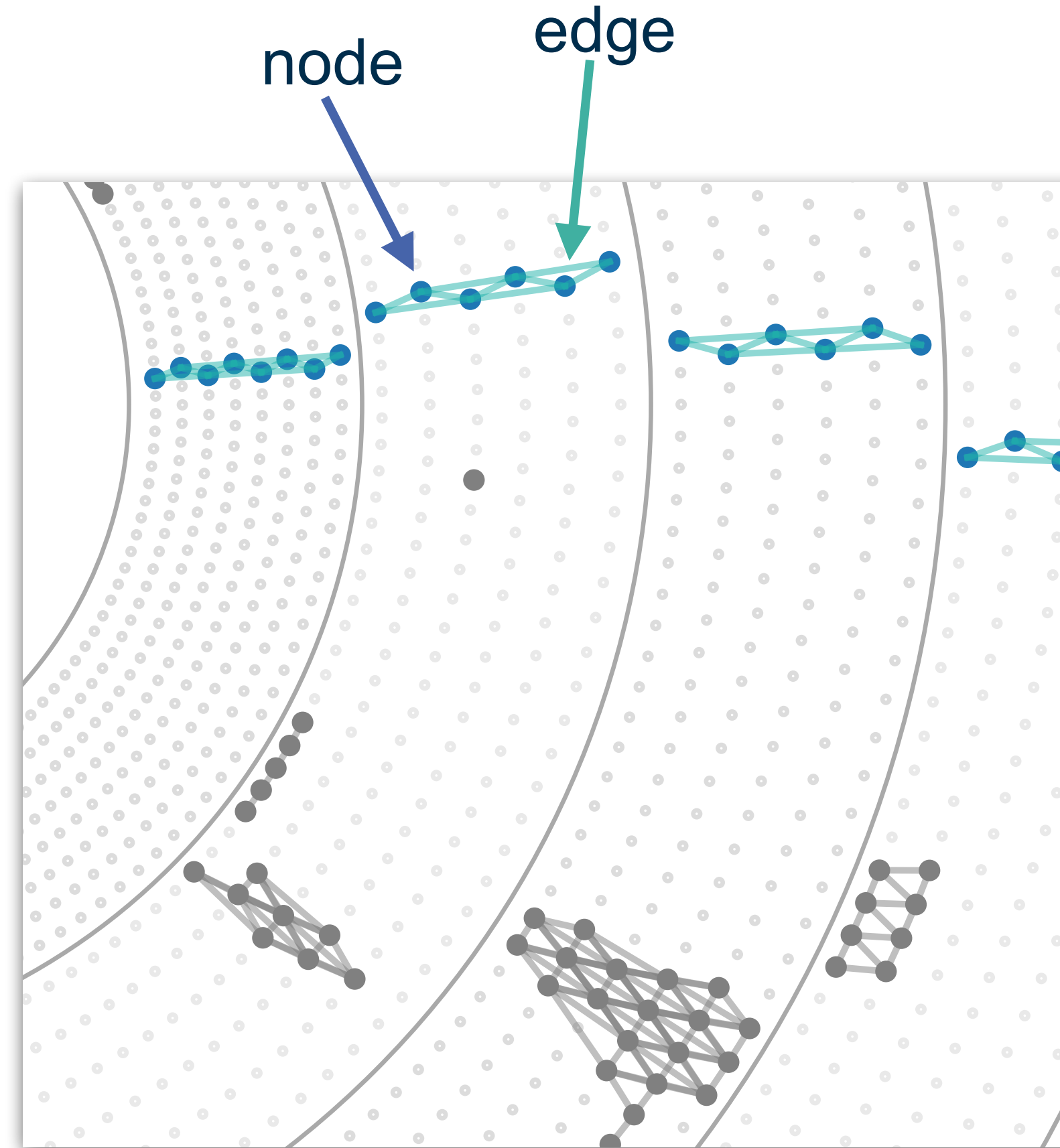
Graph Encoding

node attributes

- x coordinate ← from lookup table
- y coordinate ← from lookup table
- ADC (charge)

edge attributes

- Δr (radial distance) ← from lookup table
- $\Delta \phi$ (azimuthal angle) ← from lookup table
- ΔTDC (time)
- Truth label for training from offline reconstruction



See also

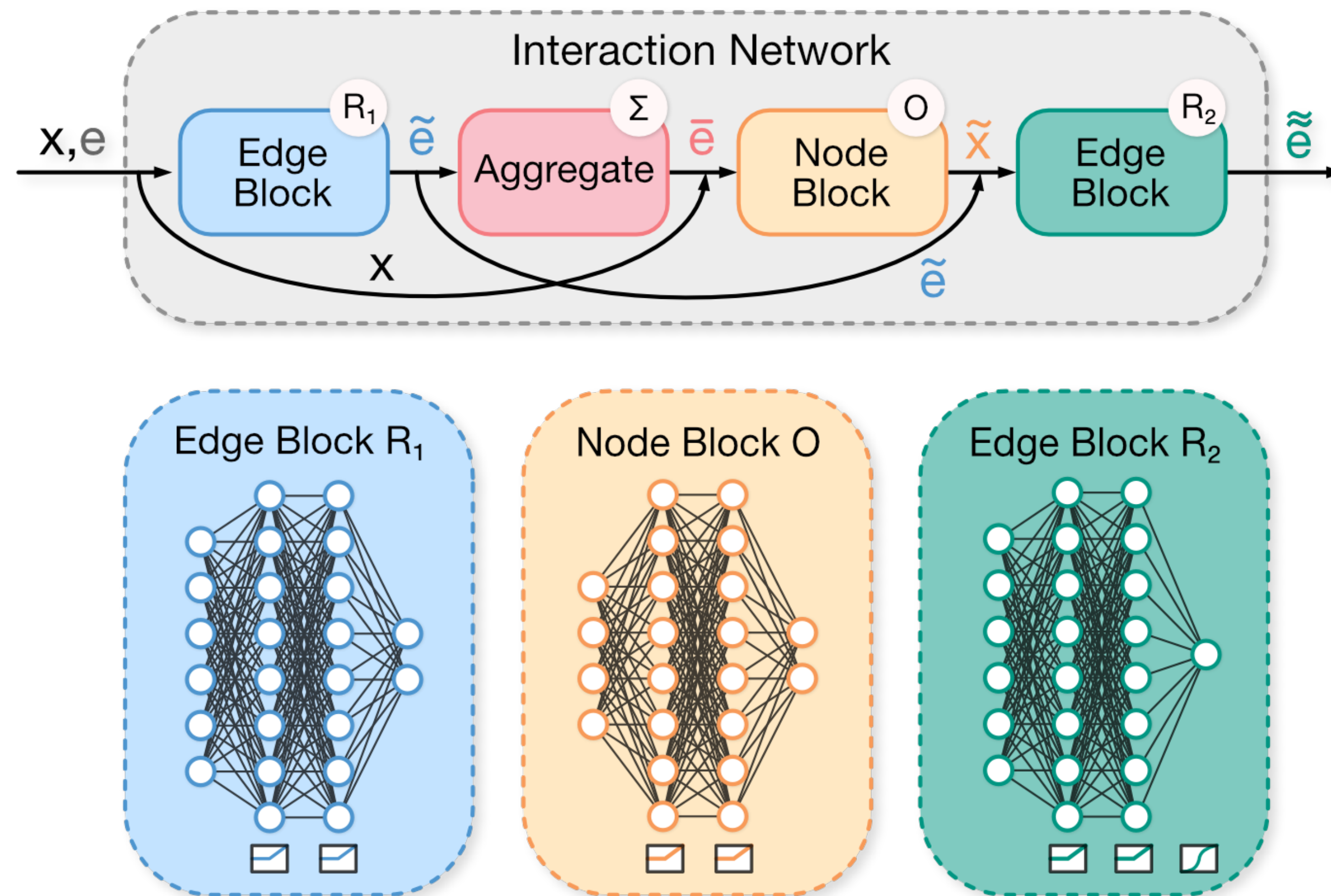
[arXiv:2307.07289](https://arxiv.org/abs/2307.07289)

(graph building with lookup tables)

*[Master thesis „Graph-Building and Input Feature Analysis for Edge Classification in the CDC at Belle II“ of Philipp Dorwarth, May 2023](#)

Interaction Network (GNN)

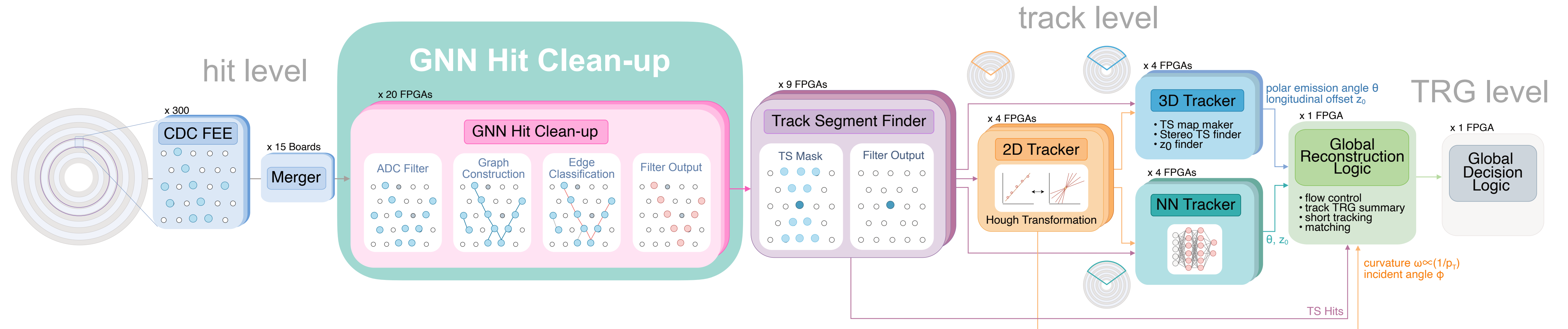
node inputs = x
edge inputs = e



- **interaction network** model by Battaglia et al.
- **multilayer perceptron blocks** chained together
- **light-weight** by design (less than 700 parameters for full precision offline model implementation)
- **local pattern recognition** (up to 2 nodes distance)

[arXiv:1612.00222](https://arxiv.org/abs/1612.00222)

Towards L1 Trigger Application



Constraints

Scope

- up to 14336 sense wires
- spread on 20 FPGA boards



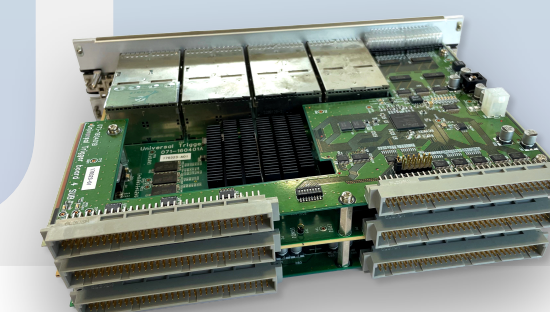
Trigger Input Signal

- up to 978 nodes + 4545 edges per FPGA
- 1 bit hit + 4 bit timing + 4 bit ADC

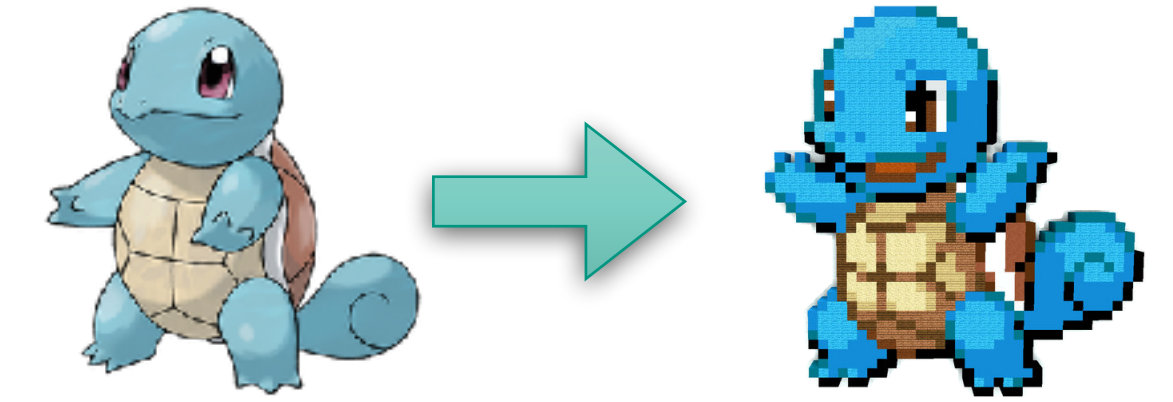


Hardware Requirements

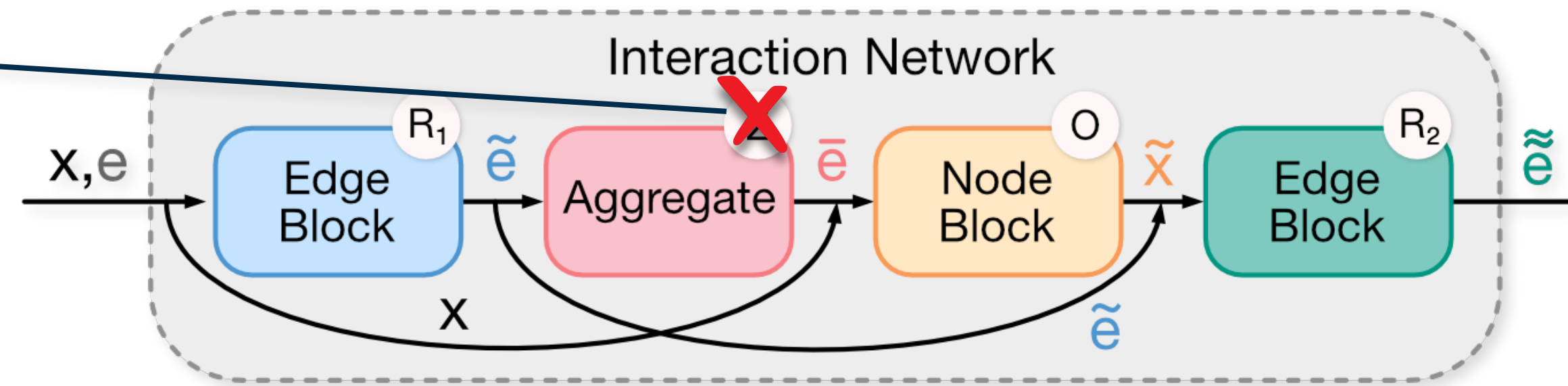
- ≈ 500 ns for Clean-up
- throughput: 32 mio events/s



Network Quantisation and Pruning



max aggregation
instead of sum



pruning

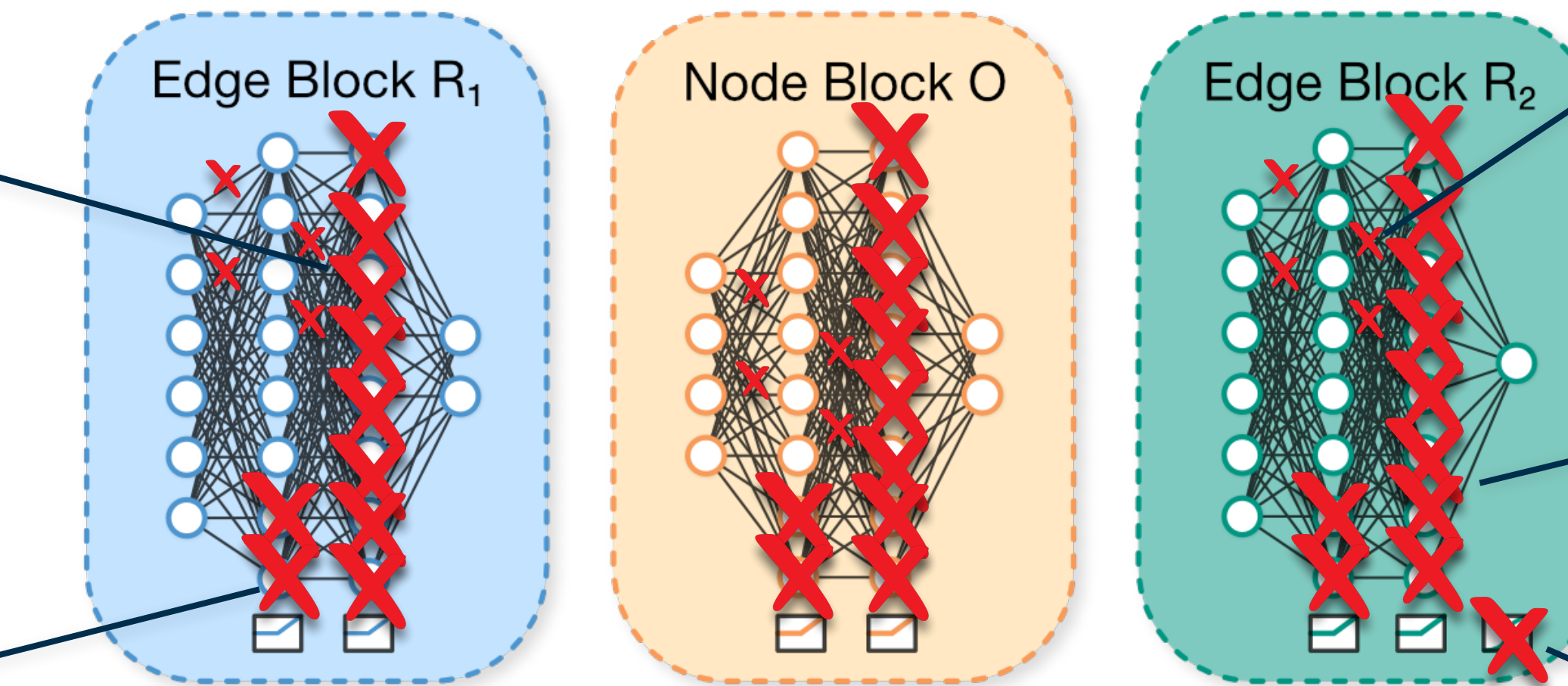
- pruning of weights to up to 50% sparsity

lower hidden depth

- remove hidden layers

hidden size reduction

- remove hidden nodes



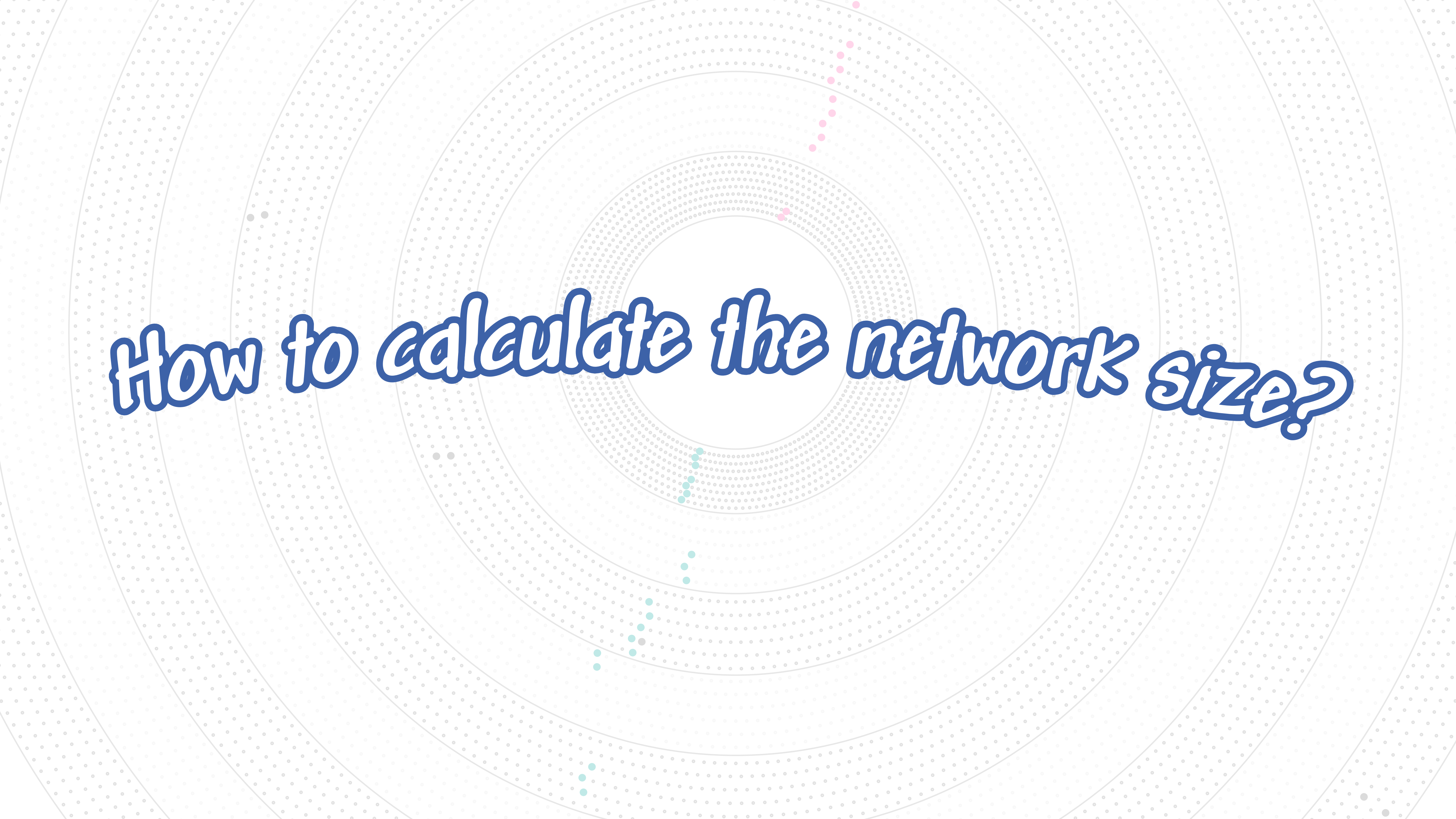
610 trainable parameters

211 trainable parameters

Brevitas
quantisation

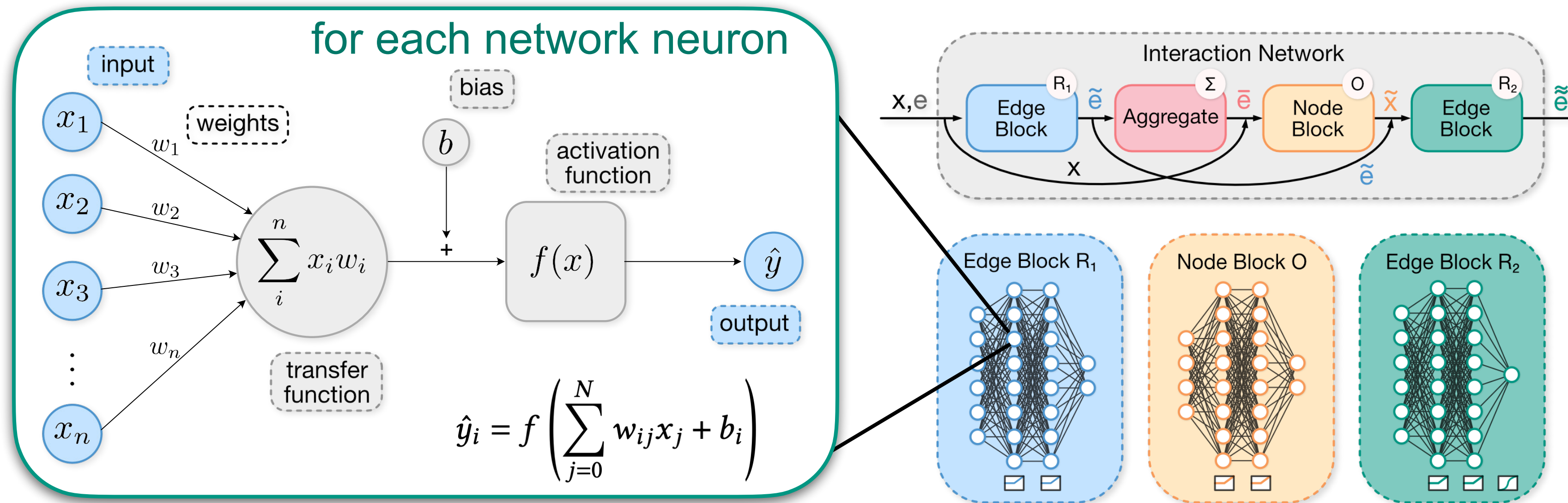
- ultra-low precision
- 4-bit weights

replace sigmoid
by linear mapping

The background features a series of concentric circles centered on the page. Each circle is composed of a dense pattern of small dots. Some of these dots are highlighted in different colors: a cluster of pink dots in the upper right, a group of teal dots in the lower left, and a few grey dots scattered elsewhere. The overall effect is a subtle, technical-looking pattern.

How to calculate the network size?

Calculating the „network size“



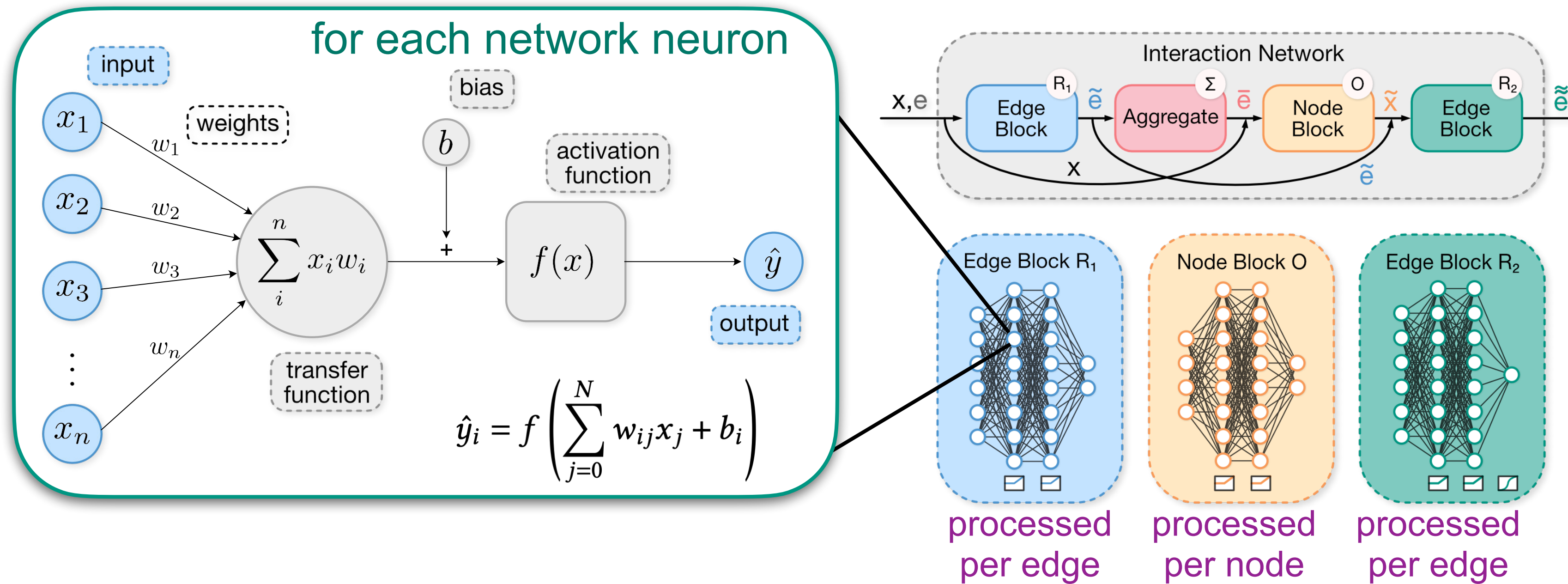
- **trainable parameters:** weights and biases
 - number of weights per layer: #inputs x #neurons
 - number of biases per layer: #neurons
 - e.g. first layer (9 inputs, 8 neurons): $9 \times 8 = 72$ weights and 8 biases
- the number of **trainable parameters** is a good starting point, but
 1. number of **multiplications** more important than number of parameters
 2. effect of **quantization** reduction not included

InteractionNetwork
(node_dim: 3, edge_dim: 3, hidden_size: 8)

Modules	Parameters
R1.layers.0.weight	72
R1.layers.0.bias	8
R1.layers.3.weight	64
R1.layers.3.bias	8
R1.layers.6.weight	24
R1.layers.6.bias	3
O.layers.0.weight	48
O.layers.0.bias	8
O.layers.3.weight	64
O.layers.3.bias	8
O.layers.6.weight	24
O.layers.6.bias	3
R2.layers.0.weight	72
R2.layers.0.bias	8
R2.layers.3.weight	64
R2.layers.3.bias	8
R2.layers.6.weight	8
R2.layers.6.bias	1

Total Trainable Params: 495

Calculating the „network size“



problems:

1. number of **multiplications** ✓
2. effect of **quantization** ✗

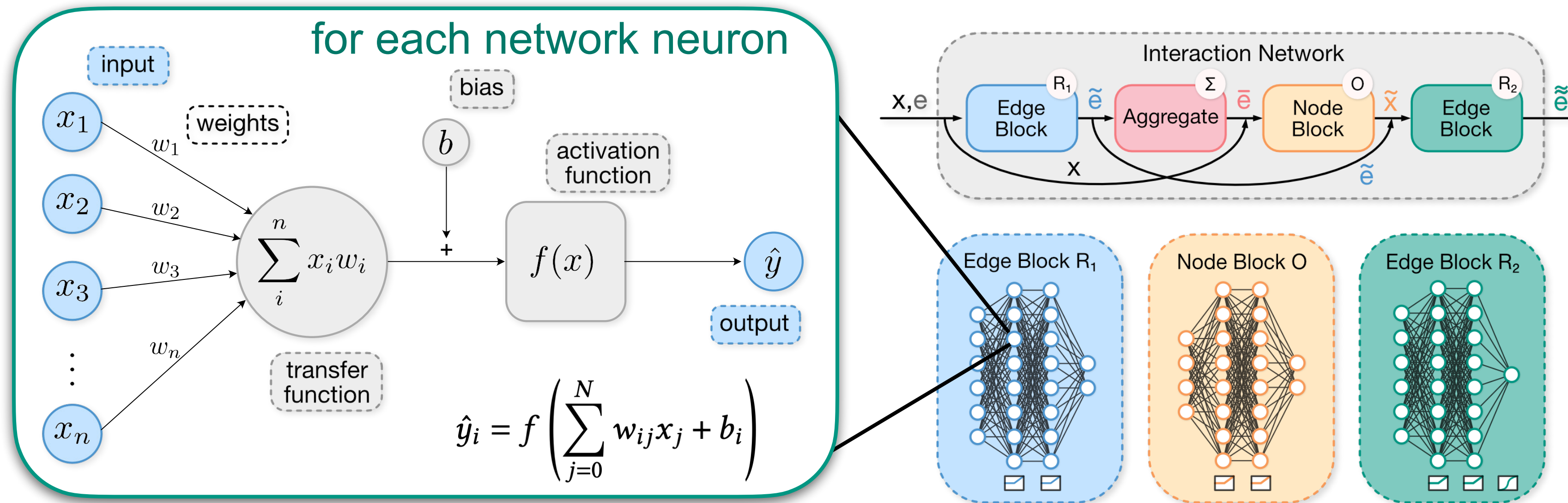
InteractionNetwork
(node_dim: 3, edge_dim: 3, hidden_size: 8)

Modules	Parameters	
R1.layers.0.weight	72	✗ edges
R1.layers.0.bias	8	
R1.layers.3.weight	64	
R1.layers.3.bias	8	
R1.layers.6.weight	24	
R1.layers.6.bias	3	
0.layers.0.weight	48	✗ nodes
0.layers.0.bias	8	
0.layers.3.weight	64	
0.layers.3.bias	8	
0.layers.6.weight	24	
0.layers.6.bias	3	
R2.layers.0.weight	72	✗ edges
R2.layers.0.bias	8	
R2.layers.3.weight	64	
R2.layers.3.bias	8	
R2.layers.6.weight	8	
R2.layers.6.bias	1	

Total Trainable Params: 495

- number of **multiply-accumulate (MAC) operations**:
 - takes into account input graph sizes
 - e.g. for a graph with 978 nodes and 4545 edges: 1.7Mio MAC operations

Calculating the „network size“



problems:

1. number of **multiplications** ✗
2. effect of **quantization** ✓

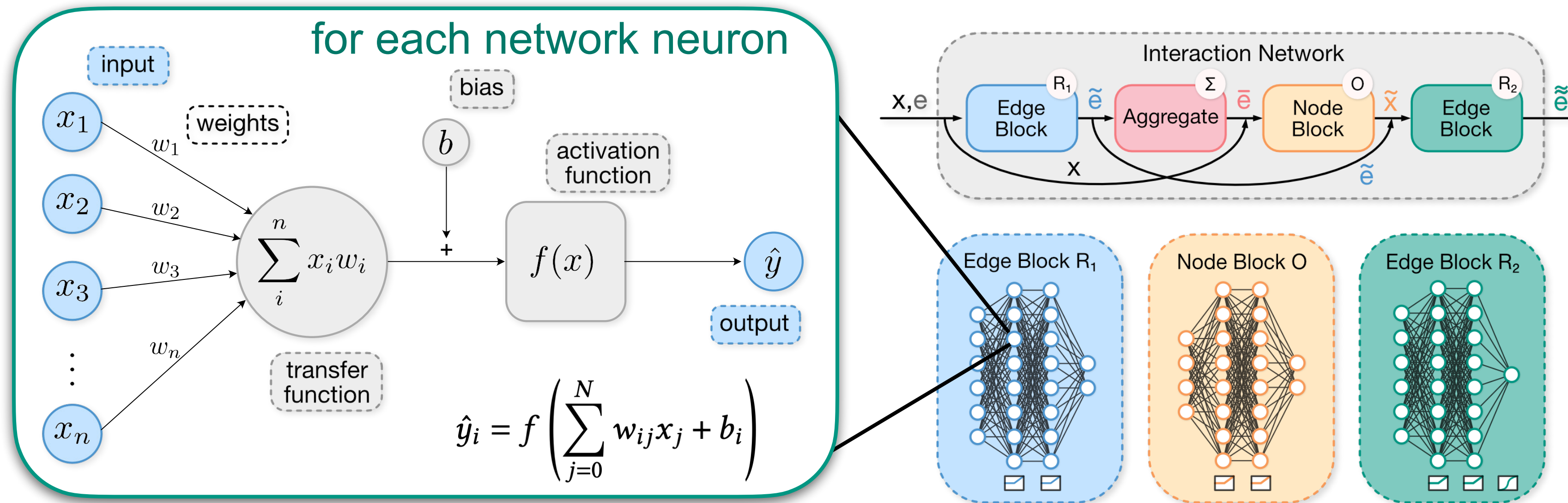
- number of **bits**:
 - quantization gives each weight/bias a reduced bit width
 - measure this by **counting the total number of bits**
 - e.g. first layer: $72 \times 4\text{bits} = 288 \text{ bits}$
- however, this is far from accurate due to
 - multiplication operations of weights: $N_w \cdot b_w \rightarrow N_w \cdot (b_{in} \cdot b_w)$
 - accumulation effects (due to summing of partial products)

InteractionNetwork
(node_dim: 3, edge_dim: 3, hidden_size: 8)

Modules	Parameters
R1.layers.0.weight	72
R1.layers.0.bias	8
R1.layers.3.weight	64
R1.layers.3.bias	8
R1.layers.6.weight	24
R1.layers.6.bias	3
O.layers.0.weight	48
O.layers.0.bias	8
O.layers.3.weight	64
O.layers.3.bias	8
O.layers.6.weight	24
O.layers.6.bias	3
R2.layers.0.weight	72
R2.layers.0.bias	8
R2.layers.3.weight	64
R2.layers.3.bias	8
R2.layers.6.weight	8
R2.layers.6.bias	1

Total Trainable Params: 610

Calculating the „network size“



problems:

1. number of **multiplications** ✓
2. effect of **quantization** ✓

- solution: number of **bit operations (BOPs)** that takes into account

- multiplications
- reduced precision due to quantization

- per layer: <https://arxiv.org/pdf/1804.10969>

$$\text{BOPs} = N_w b_w b_{in} + N_{bias} b_{bias} + N_w b_{acc}$$

where $b_{acc} = b_{in} + b_w + \log_2(N_{in}) \approx b_{in} + b_w$

N_w = number of weights
 N_{bias} = number of biases
 N_{in} = number of input features
 b_w = bit width of weights
 b_{bias} = bit width of biases
 b_{in} = bit width of input

InteractionNetwork
(node_dim: 3, edge_dim: 3, hidden_size: 8)

Modules	Parameters	Bit width
R1.layers.0.weight	72	4
R1.layers.0.bias	8	16
R1.layers.3.weight	64	4
R1.layers.3.bias	8	16
R1.layers.6.weight	24	4
R1.layers.6.bias	3	16
O.layers.0.weight	48	4
O.layers.0.bias	8	16
O.layers.3.weight	64	4
O.layers.3.bias	8	16
O.layers.6.weight	24	4
O.layers.6.bias	3	16
R2.layers.0.weight	72	4
R2.layers.0.bias	8	16
R2.layers.3.weight	64	4
R2.layers.3.bias	8	16
R2.layers.6.weight	8	8
R2.layers.6.bias	1	16

Total Trainable Params: 610

Total Sum of Bits: 1144

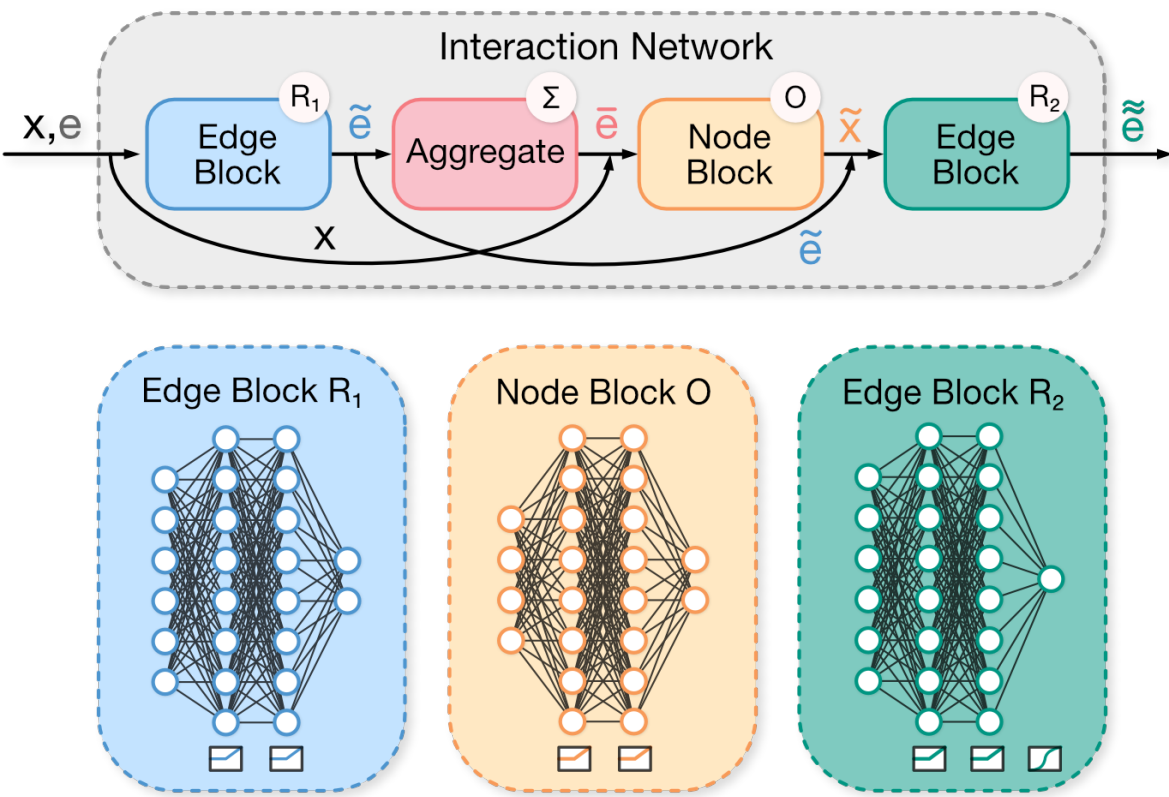
- e.g. for a graph with 978 nodes and 4545 edges: 42Mio BOPs

Network size as design target

from an initial study from Fabio Mayer (ITIV):

- a network with
 - 4 node features
 - 5 edge features
 - hidden depth = 2
 - hidden size = 8
 - weight sparsity = 80%
- for a graph with
 - 978 nodes
 - 4545 edges
- with a reuse factor of 8 ($f_{sys} = 256\text{MHz}$ and $f_{cdc} = 32\text{MHz}$)
- would fit on the given FPGA (Virtex Ultrascale 160)

target BOPs = 2265 x edges + 1000 x nodes = 1.4Mio operations



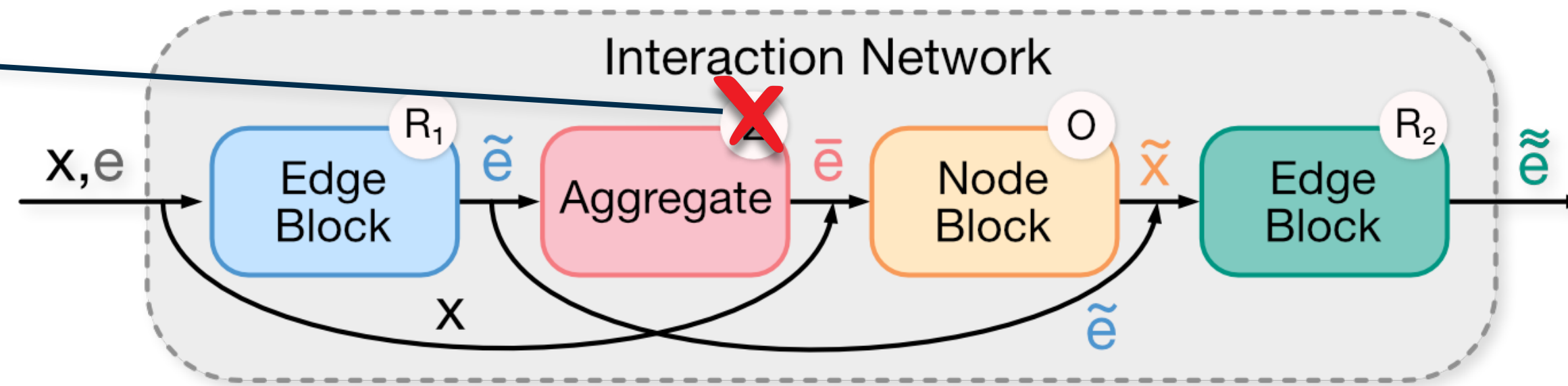
Reference InteractionNetwork
(node_dim: 4, edge_dim: 5, hidden_size: 8, sparsity: 80%)

Modules	Parameters	Bit width	BOPs
R1.layers.0.weight	104	4	582
R1.layers.0.bias	8	4	32
R1.layers.3.weight	64	4	345
R1.layers.3.bias	8	4	32
R1.layers.6.weight	40	4	216
R1.layers.6.bias	5	4	20
O.layers.0.weight	72	4	403
O.layers.0.bias	8	4	32
O.layers.3.weight	64	4	345
O.layers.3.bias	8	4	32
O.layers.6.weight	32	4	172
O.layers.6.bias	4	4	16
R2.layers.0.weight	104	4	582
R2.layers.0.bias	8	4	32
R2.layers.3.weight	64	4	345
R2.layers.3.bias	8	4	32
R2.layers.6.weight	8	4	43
R2.layers.6.bias	1	4	4

Total Trainable Params: 610
 Total Sum of Bits: 674
 Total BOPs: 3265 (Edge: 2265, Node: 1000)

Network Quantisation and Pruning

max aggregation
instead of sum

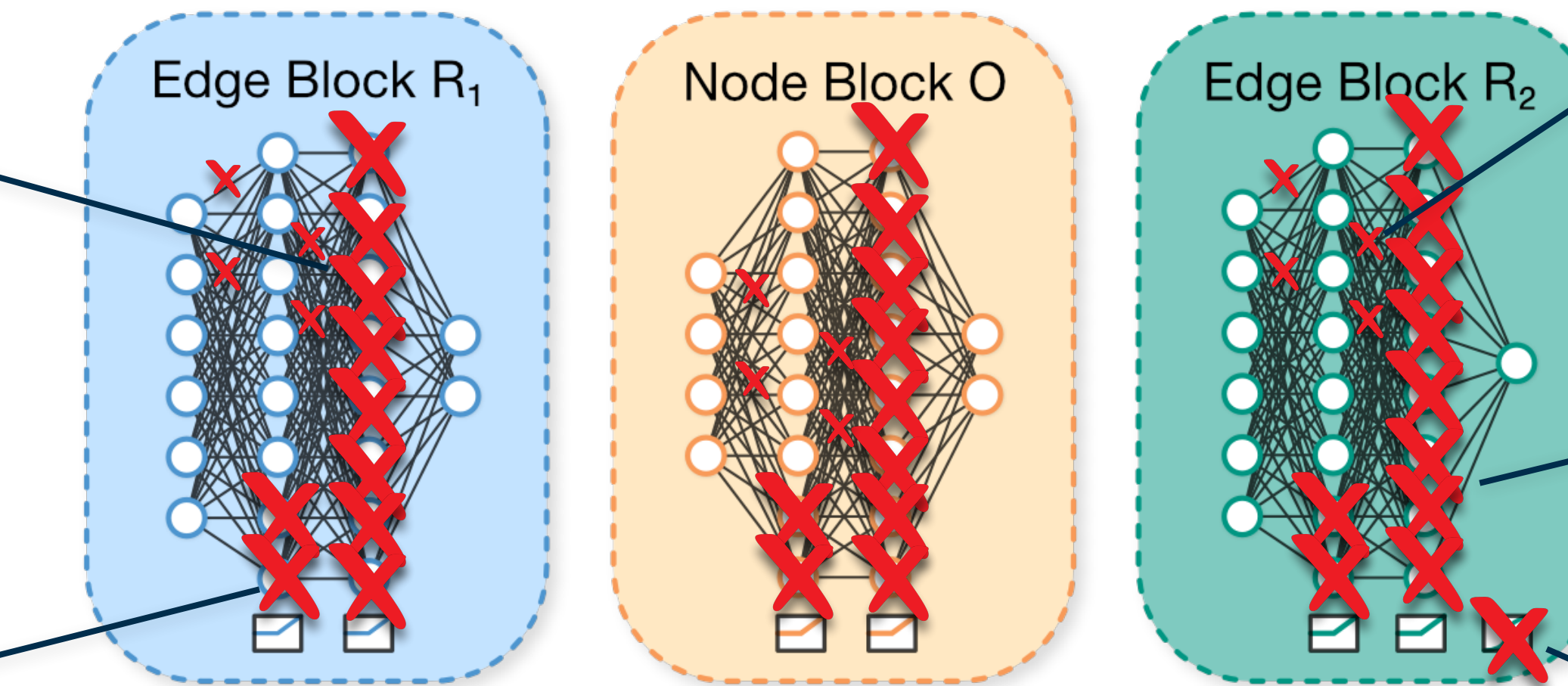


lower hidden depth

- remove hidden layers

hidden size reduction

- remove hidden nodes



610 trainable parameters

<1.4 Mio?

211 trainable parameters

pruning

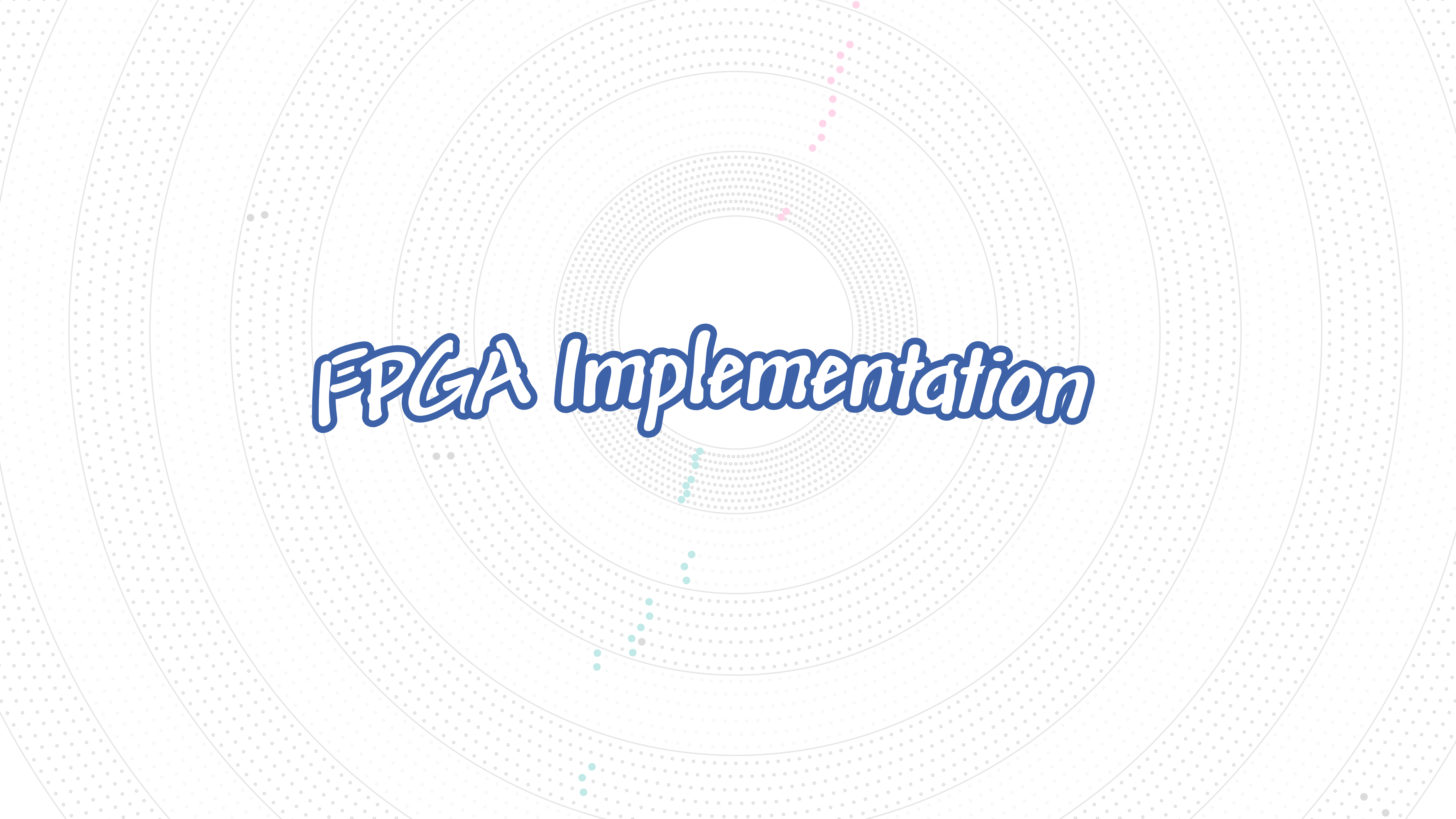
- pruning of weights to up to 50% sparsity

Brevitas
quantisation

- ultra-low precision
- 4-bit weights

replace sigmoid
by linear mapping

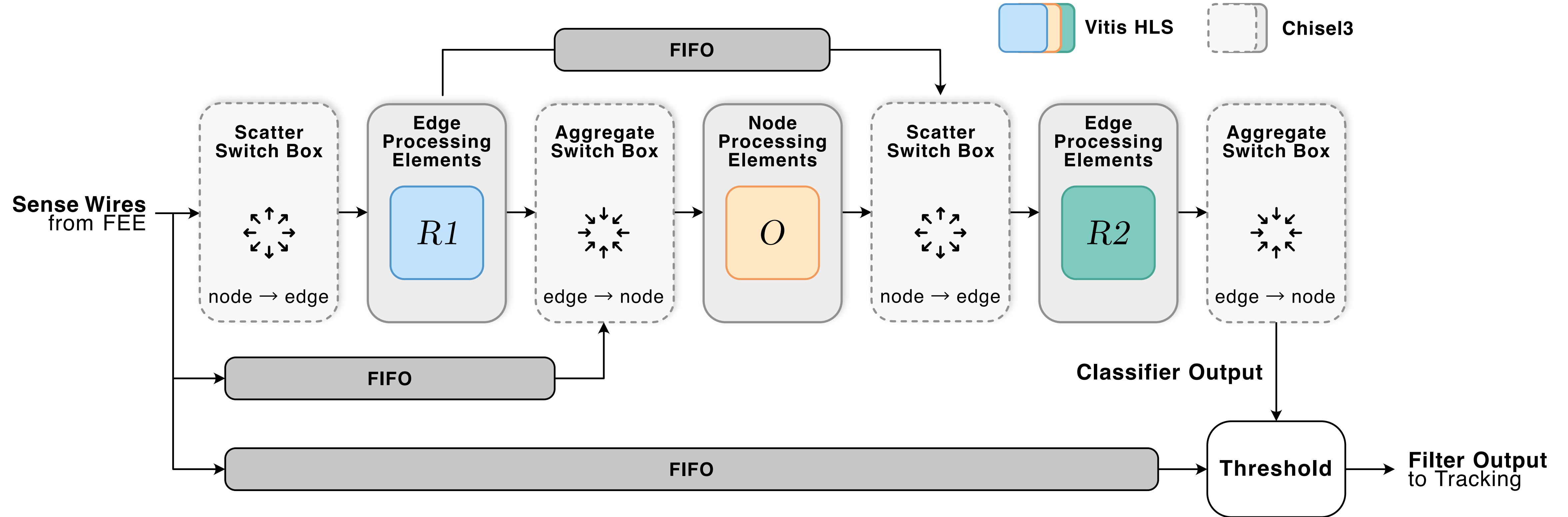


The background of the image features a series of concentric circles centered on the text. These circles are composed of a dense pattern of small, light gray dots. Scattered throughout the image are several larger, colored dots in shades of pink, teal, and gray, some of which are arranged in small, curved sequences. The overall effect is a technical, grid-like aesthetic.

FPGA Implementation

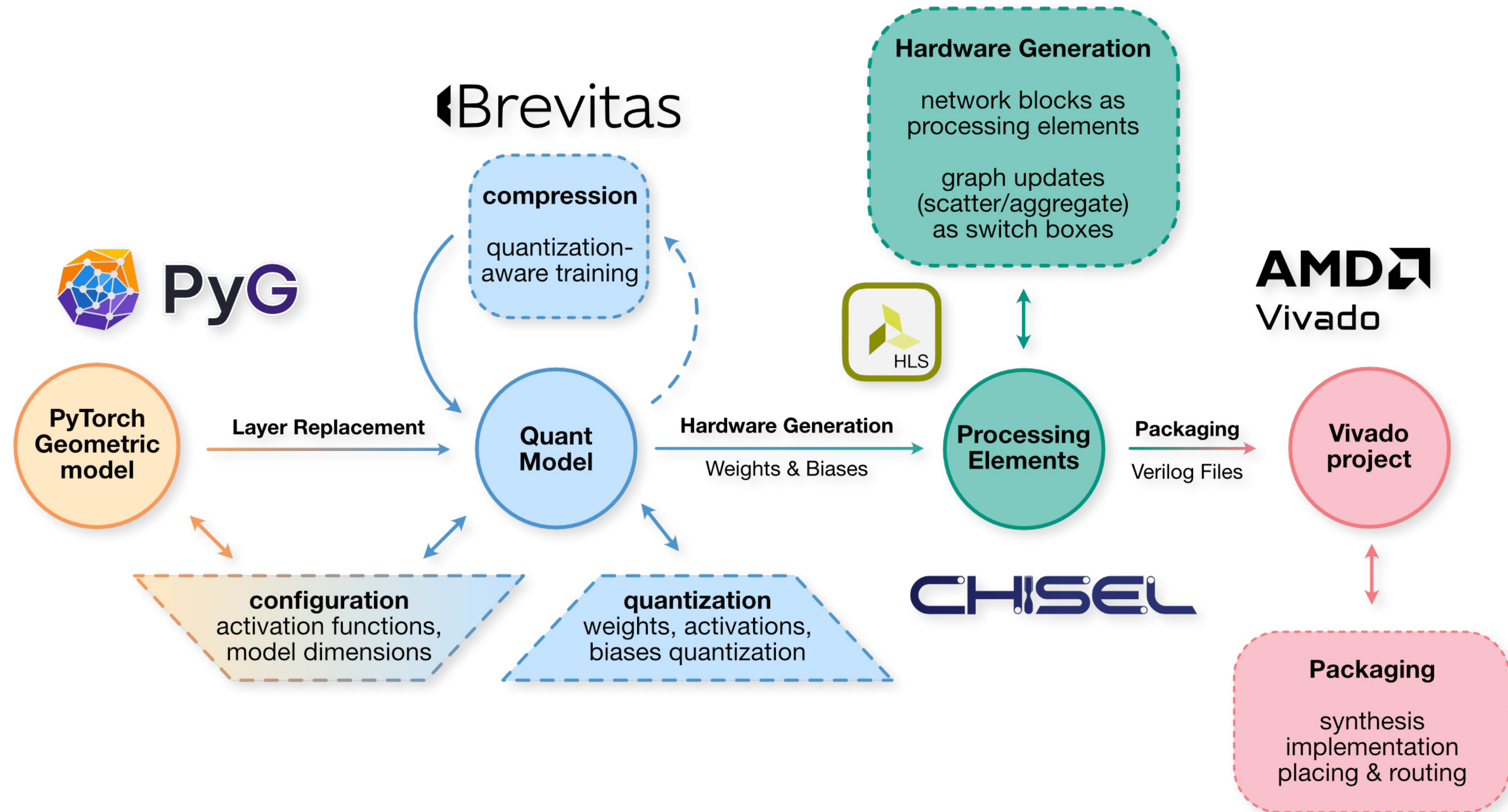
Hardware architecture

done by Marc Neu & Fabio Mayer (ITIV)



- quantised **node** and **edge** blocks embedded as black boxes (processing elements) in Chisel3
- utilises static pre-defined graphs and **updates** nodes or edges in each step by **switch boxes**
- this design allows for drop-in replacement of the network blocks for other network architectures

FPGA implementation Workflow

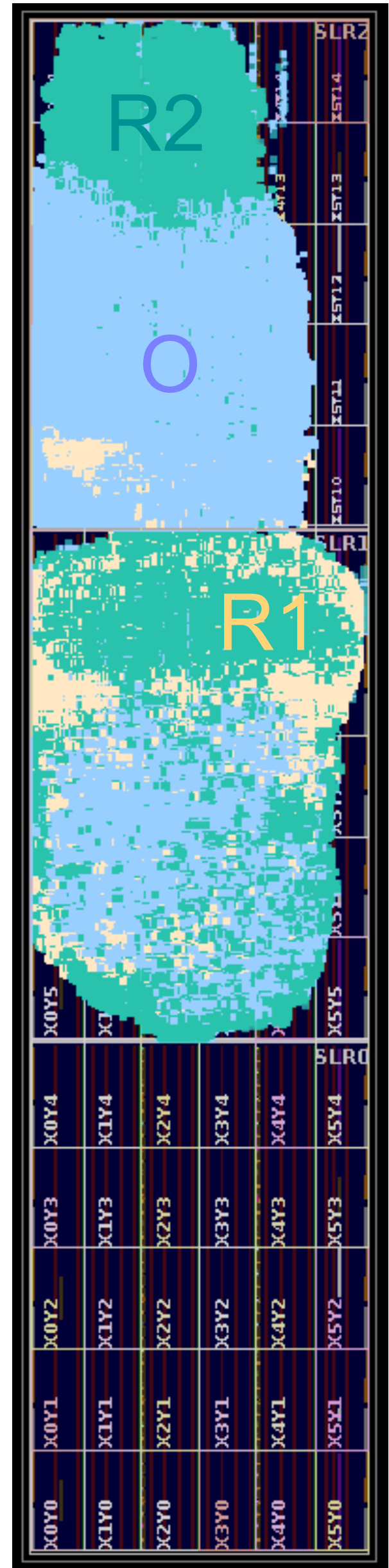
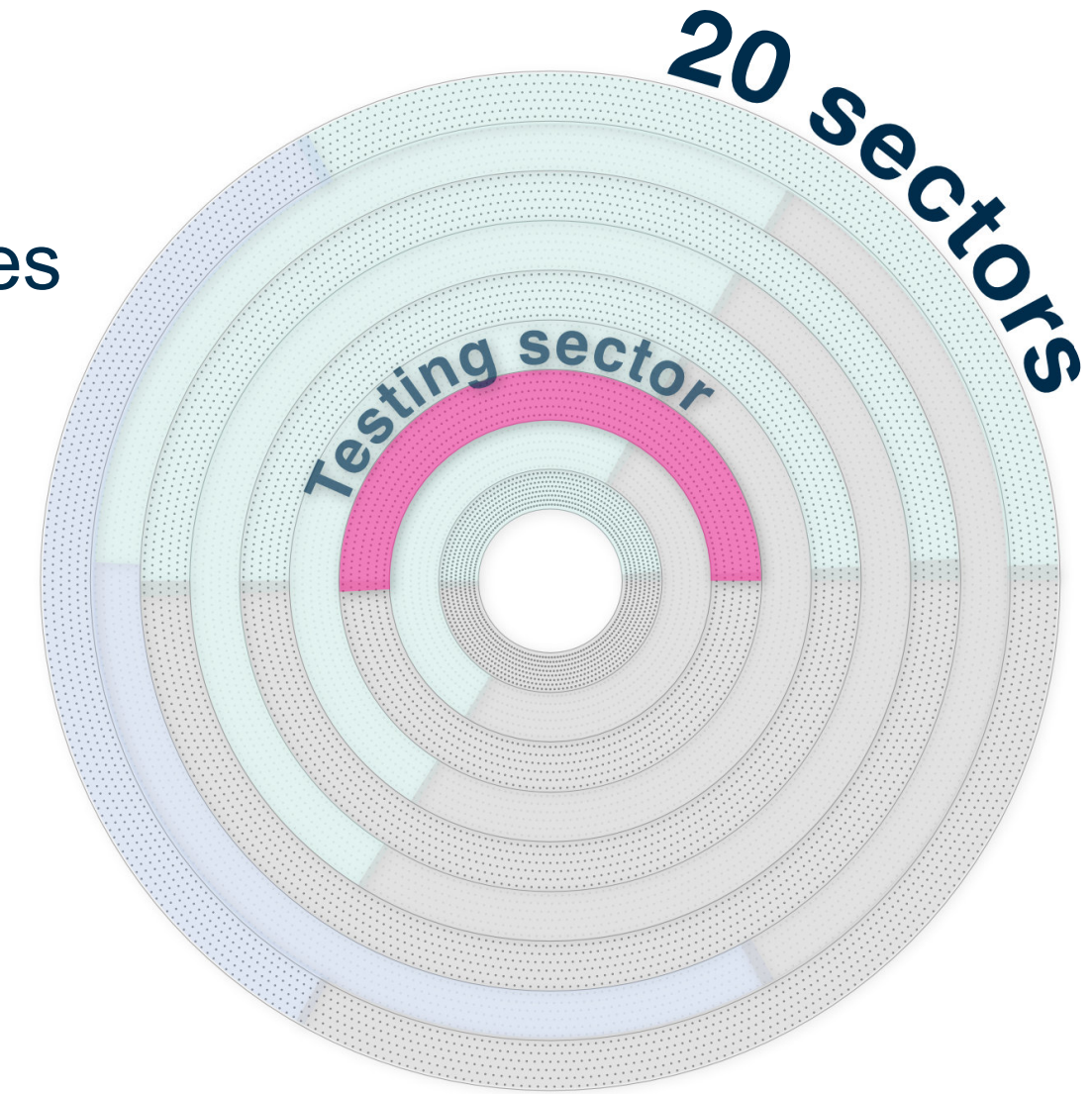


Implementation on Virtex Ultrascale 160

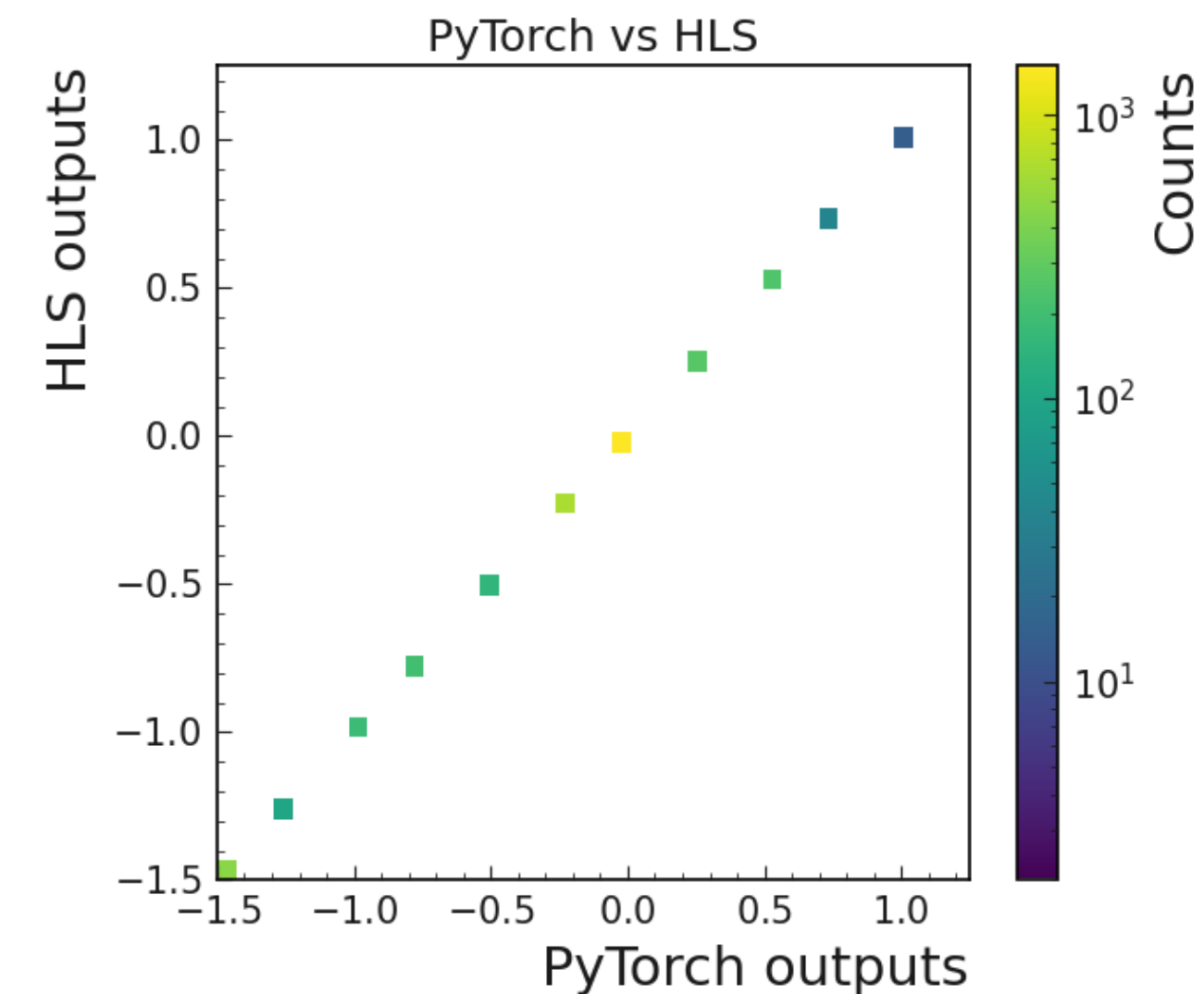
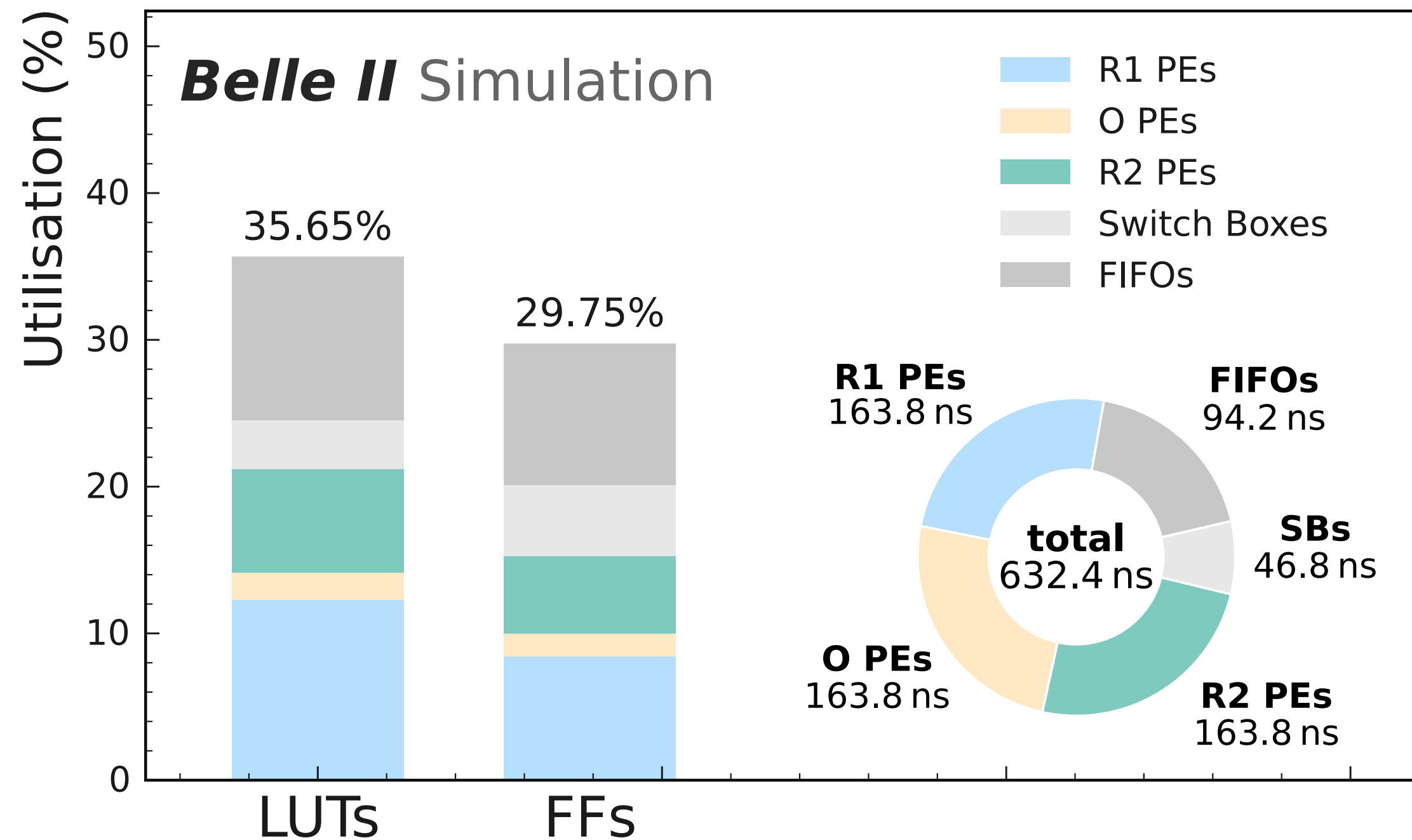
done by Marc Neu & Fabio Mayer (ITIV)

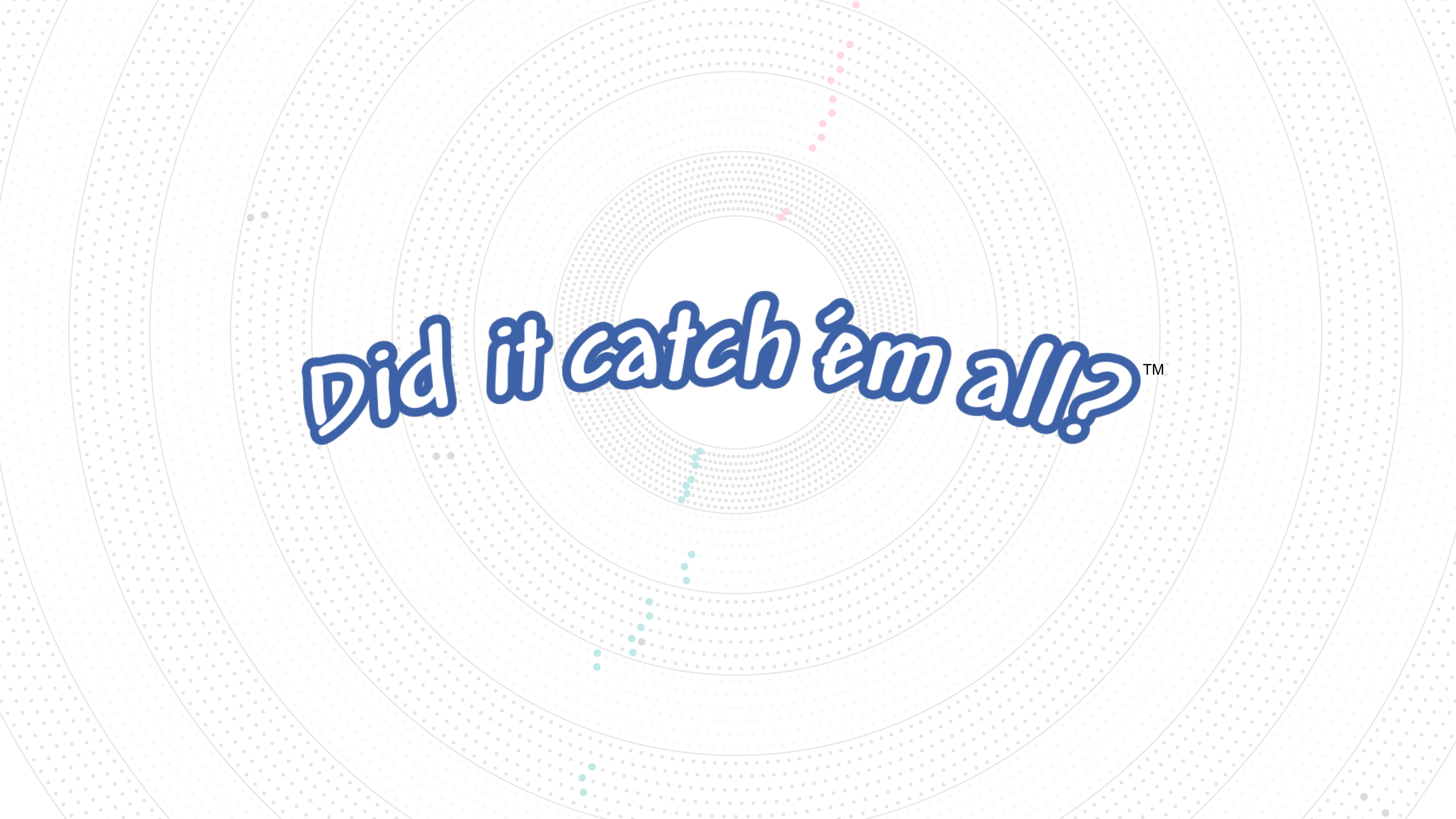
- out-of-context synthesis, place & route for a test sector with 495 nodes and 2163 edges
- functional verification in cycle-accurate simulation using CoCoTb and ModelSim
- system frequency $f_{\text{sys}} = 128 \text{ MHz}$ and CDC readout frequency $f_{\text{cdc}} = 32 \text{ MHz}$:

$$\left\lceil \frac{N_{\text{max}}}{4} \right\rceil = 124 \times \text{Node Block } O \text{ \& } \left\lceil \frac{E_{\text{max}}}{4} \right\rceil = 541 \times \text{Edge Block } R_1 \text{ \& } R_2$$



on AMD Ultrascale XVCU190

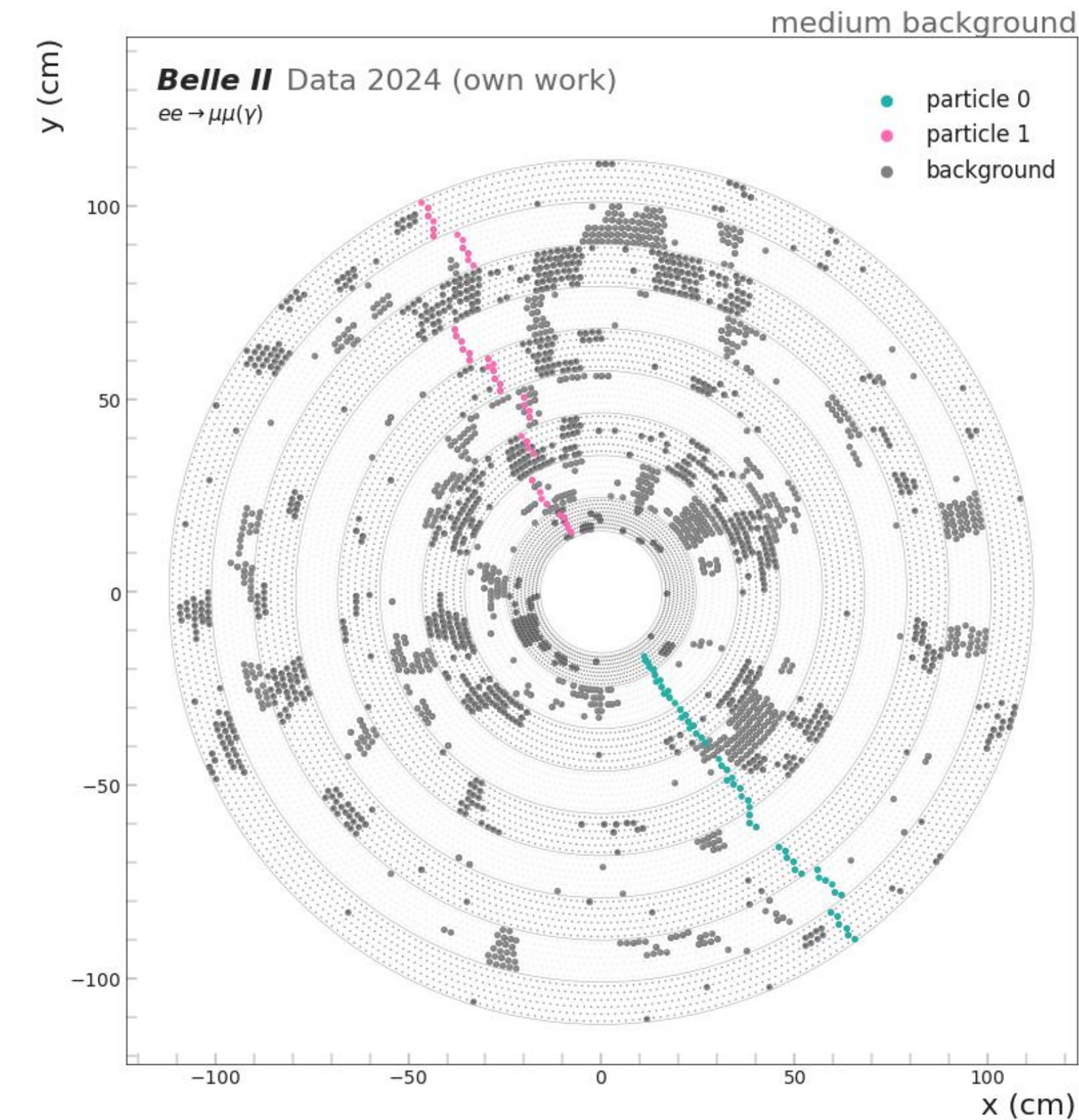




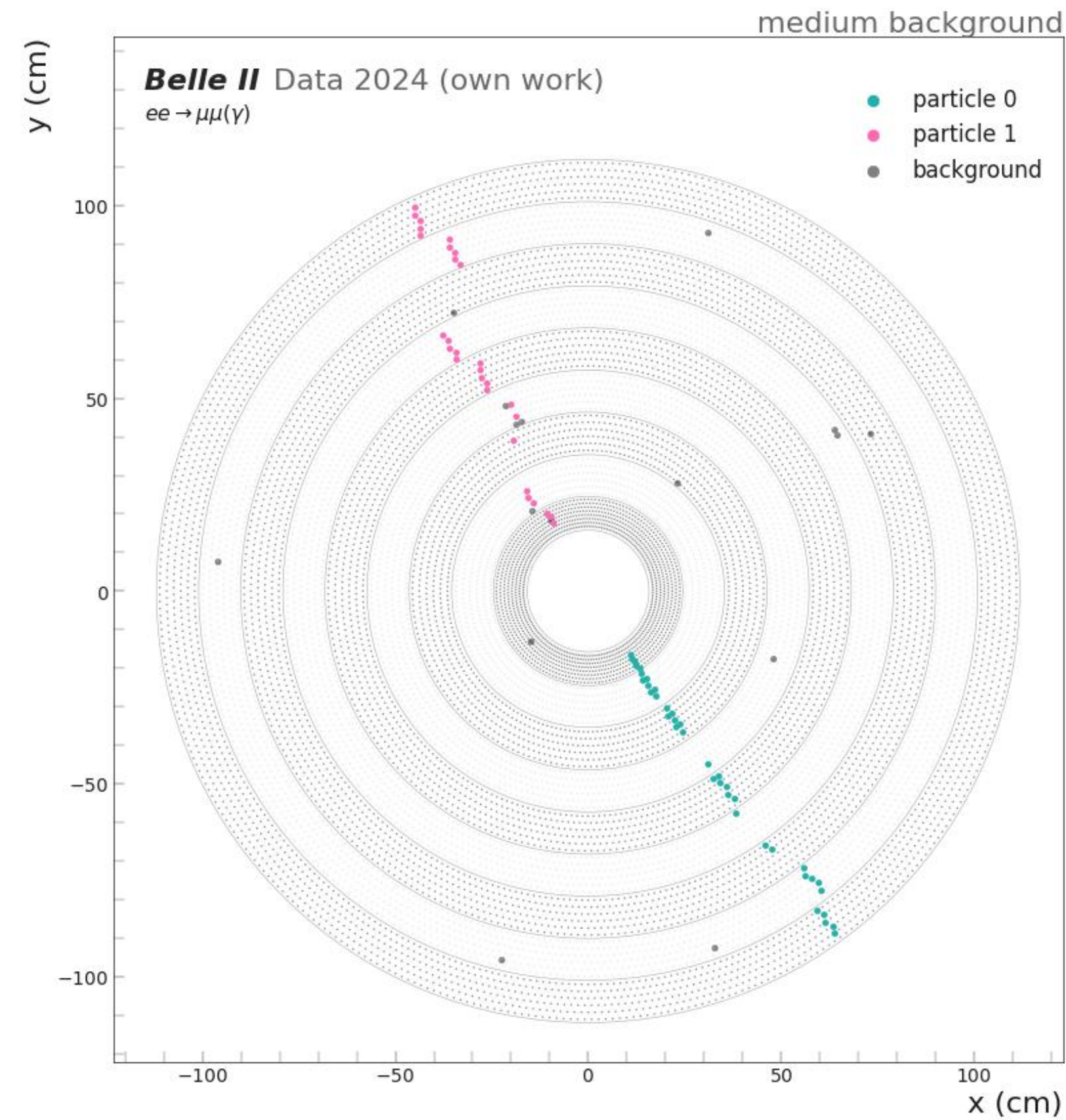
Did it catch 'em all?™

GNN Filtering on Hit Level

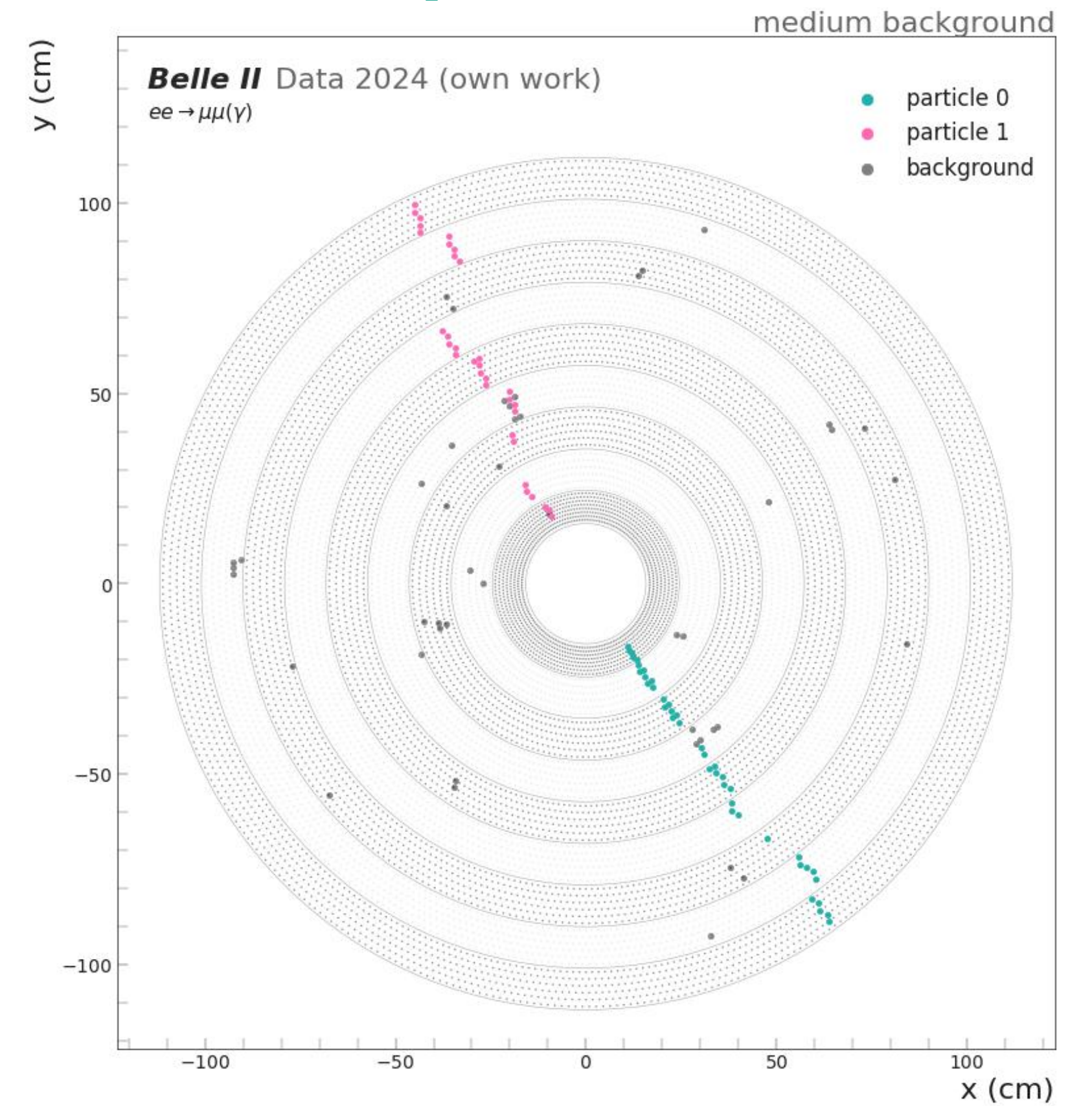
before cleanup



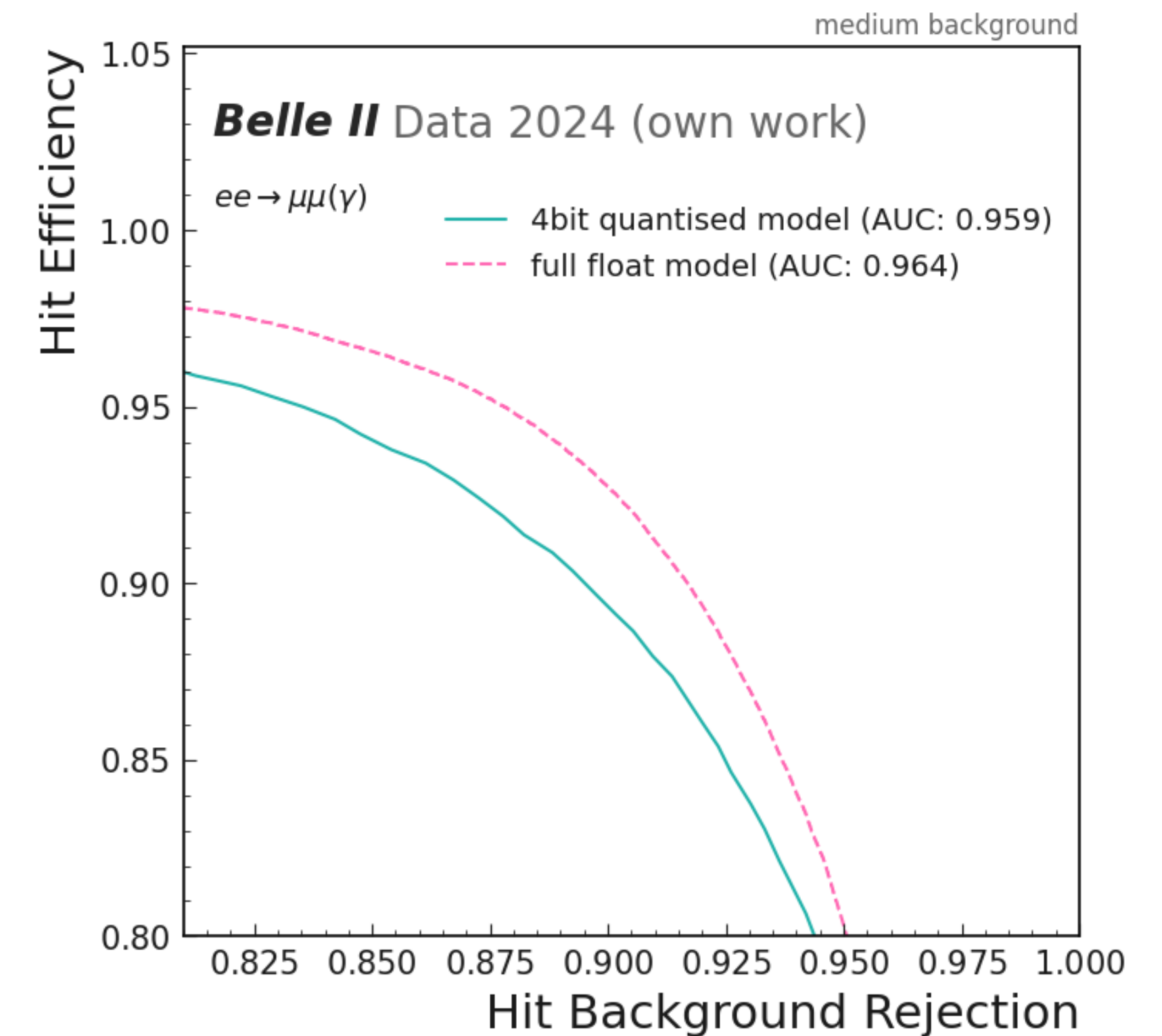
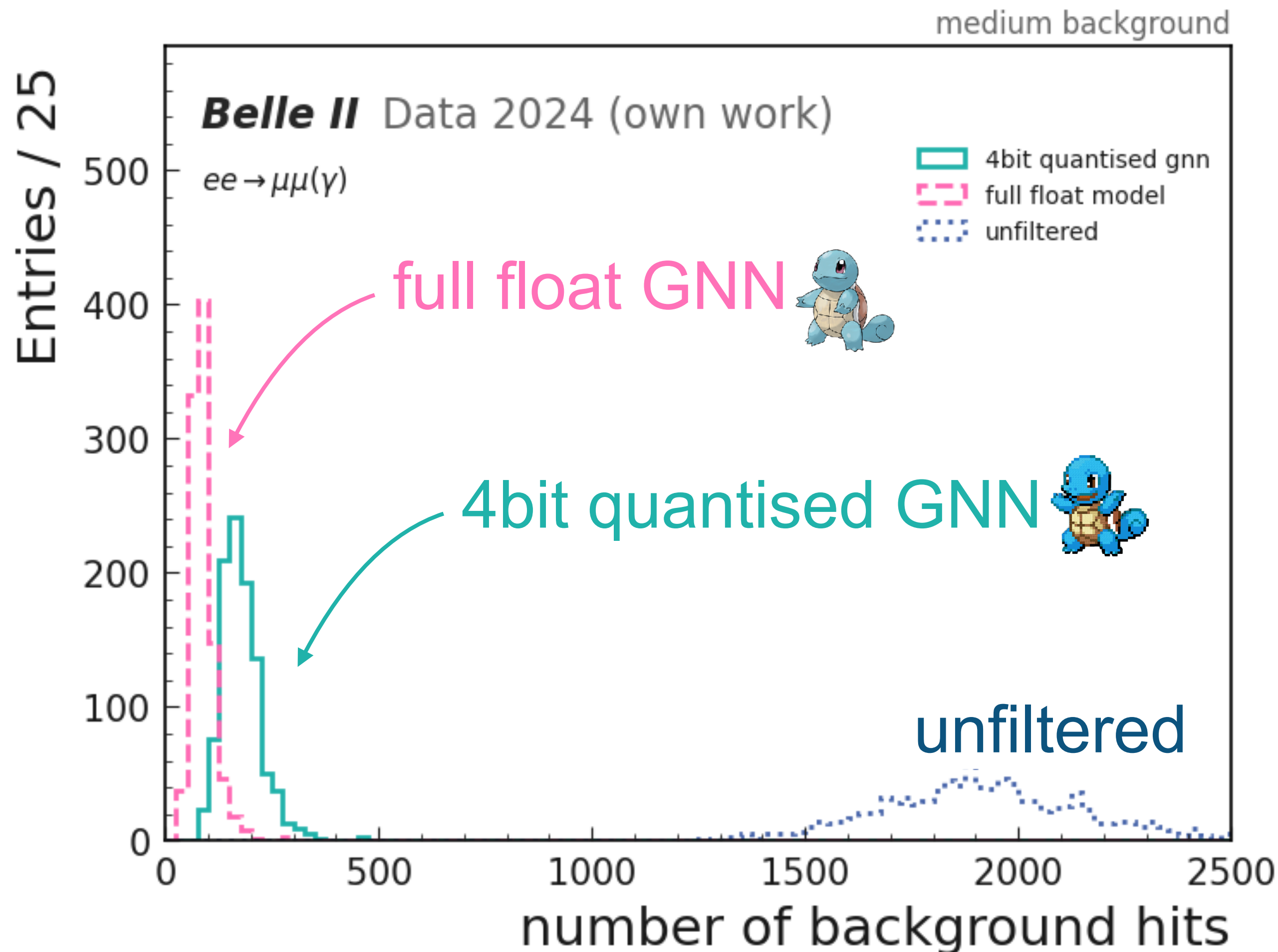
after full float model



after quant model



Hit Filtering @95% Efficiency



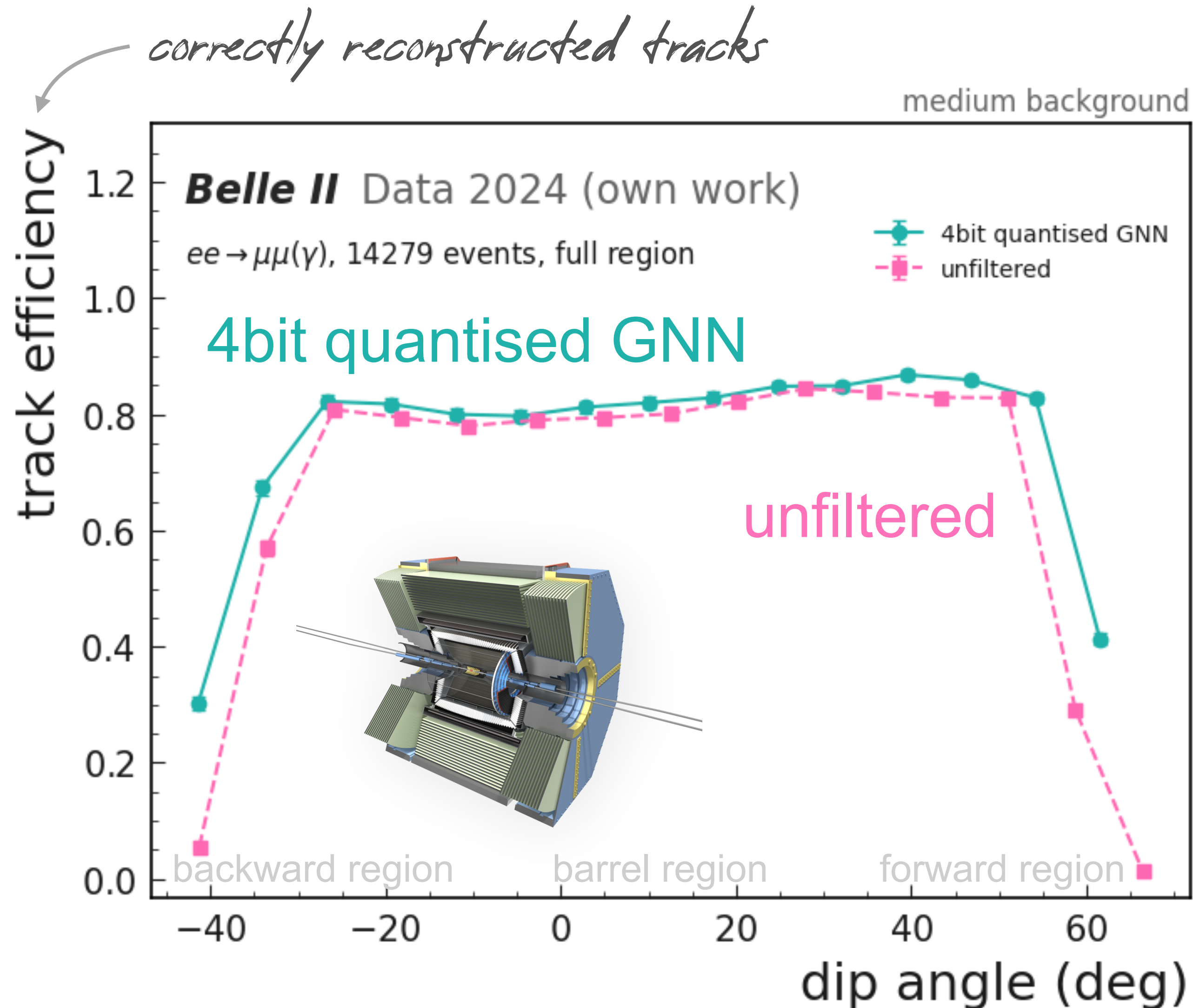
at a hit efficiency of 95%*:

- the **full floating point** model removes **87%** of background hits
- the **4bit quantised** model removes **83%** of background hits

* ratio of retained signal hits over all signal hits**

** signal hit = hit related with offline reconstructed track fulfilling quality criteria in momentum and z offset

Tracking Power



- running the **trigger simulation track finding and fitting** with and without GNN filtering
- this includes optimisation of subsequent module parameters to filtered hits
- the **track fitting efficiency** reaches **71.4%** (unfiltered: 64.0%)
- at a **track fake rate** of 3.38% (unfiltered: 3.38%)
- highest improvement in the endcaps: **+20%** vs. unfiltered

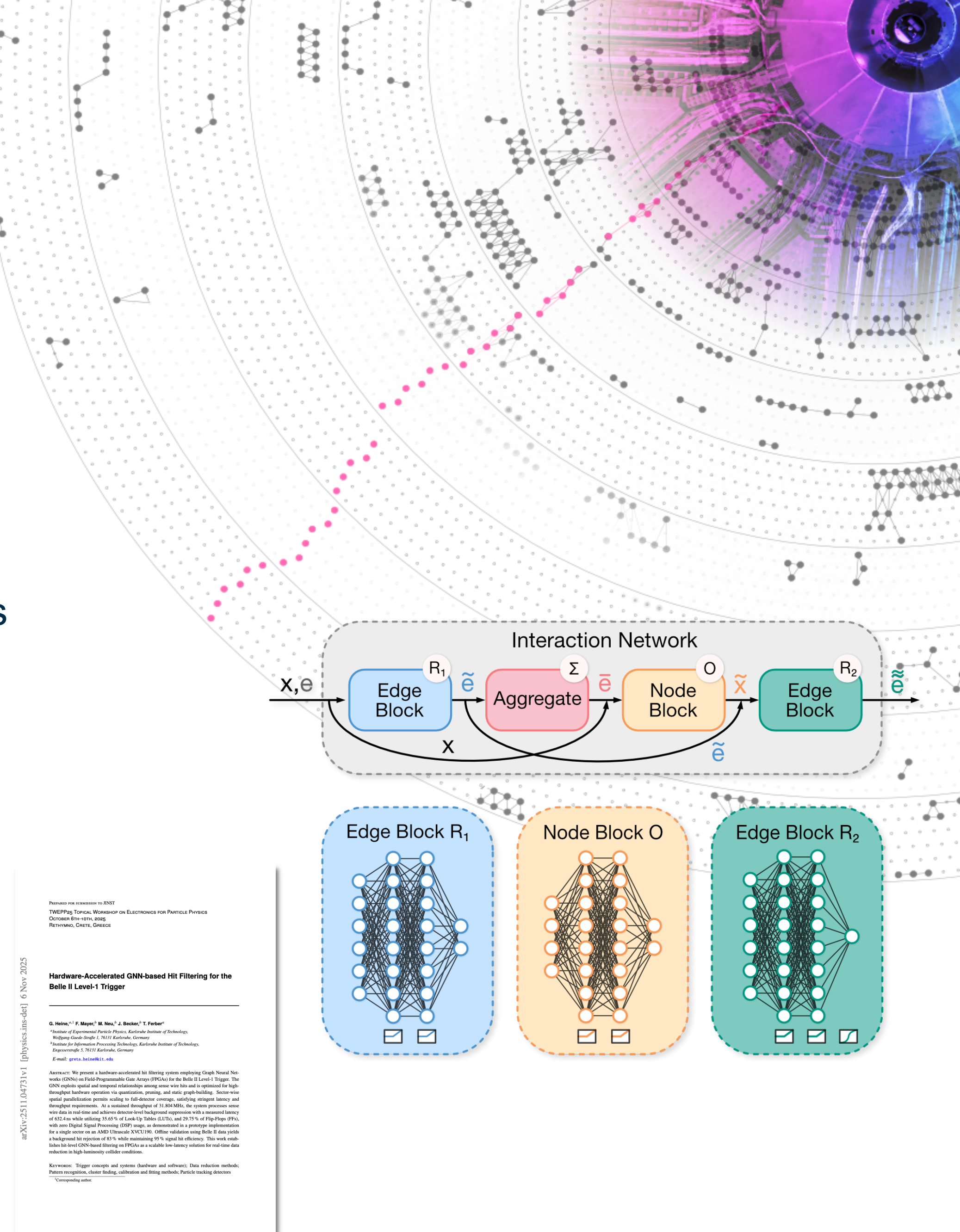
* ratio of correctly reconstructed signal tracks**

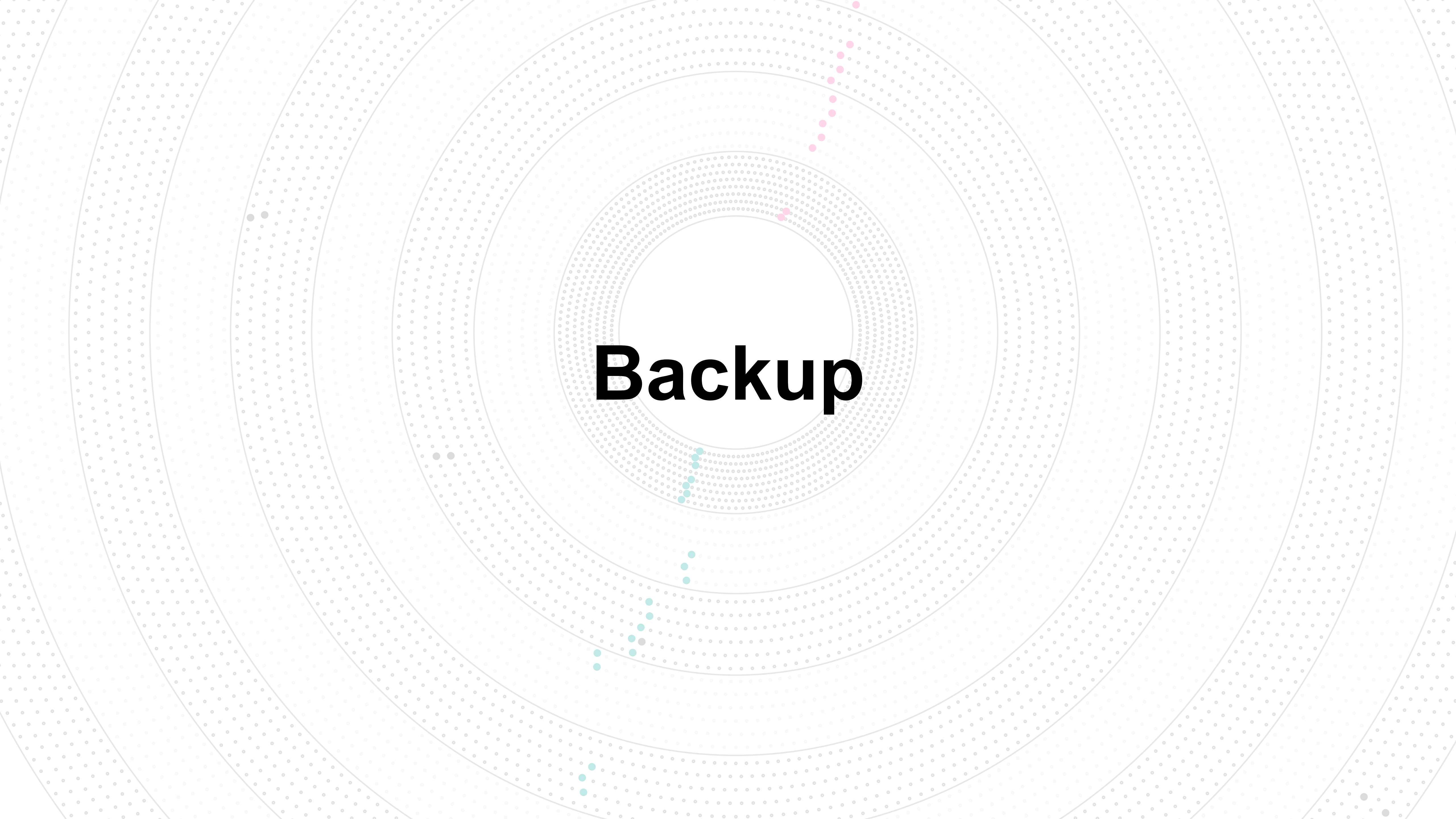
** signal track = offline reconstructed track fulfilling quality criteria in momentum and z offset

Summary and Outlook

- **background levels** increase with luminosity, e.g. by scattering
- this poses huge challenges for the **track trigger** (efficiency vs. background rejection)
- **GNN-based hit filtering** can significantly **improve the track trigger**
 - **hit-level filtering**: retains 95% true hits, removes 83% background hits
 - **track-level efficiency improved**: 64.0% $\rightarrow$ 71.4% (at same fake rate)
- **Implementation on FPGA**
 - **latency**: network inference <650ns at $f_{\text{sys}} = 128\text{MHz}$
 - **end-of-year goal**: real-time prototype running on evaluation board as important step toward in-detector deployment

Hardware-Accelerated GNN-based Hit Filtering for the Belle II Level-1 Trigger
<https://arxiv.org/pdf/2511.04731>

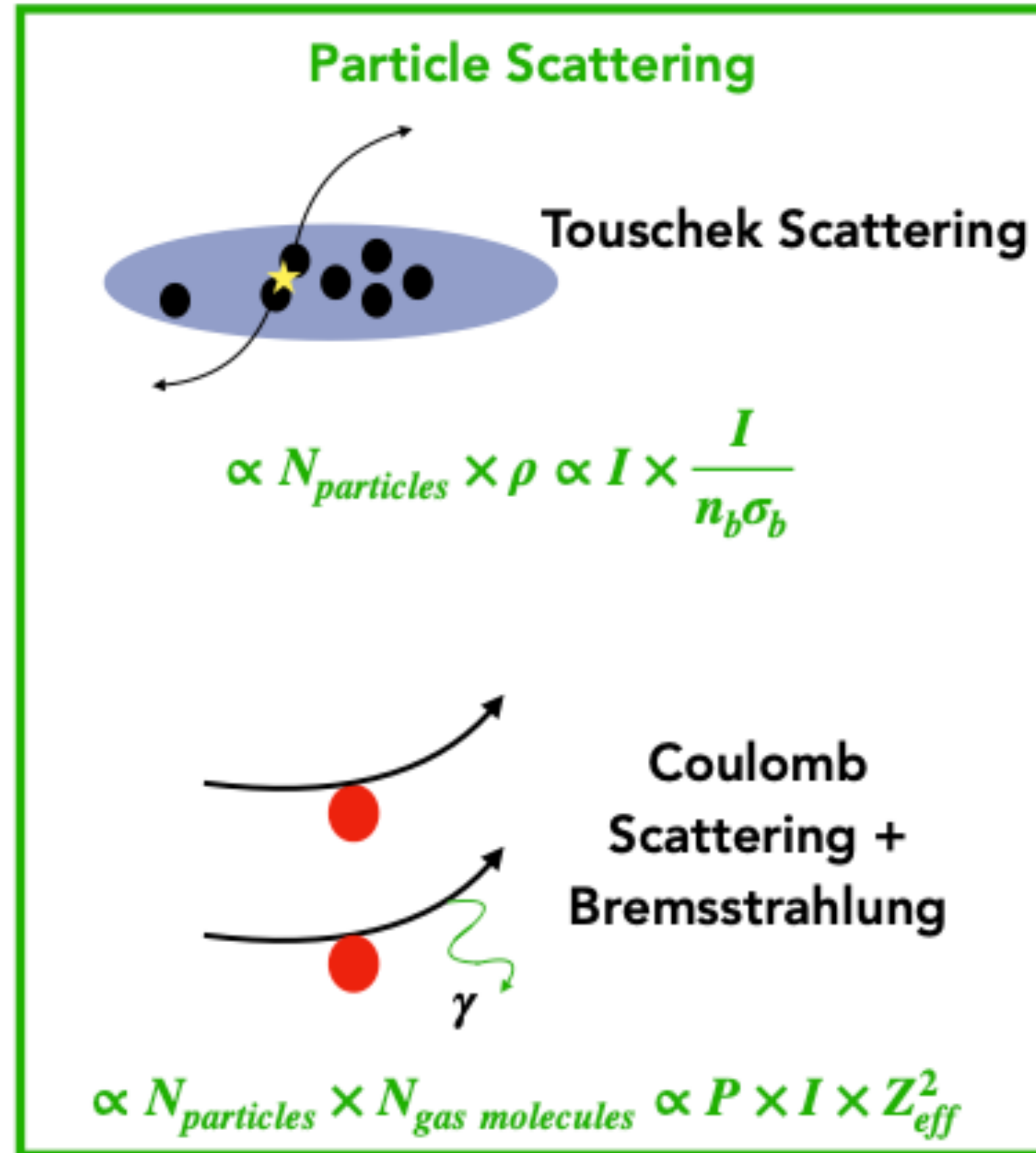




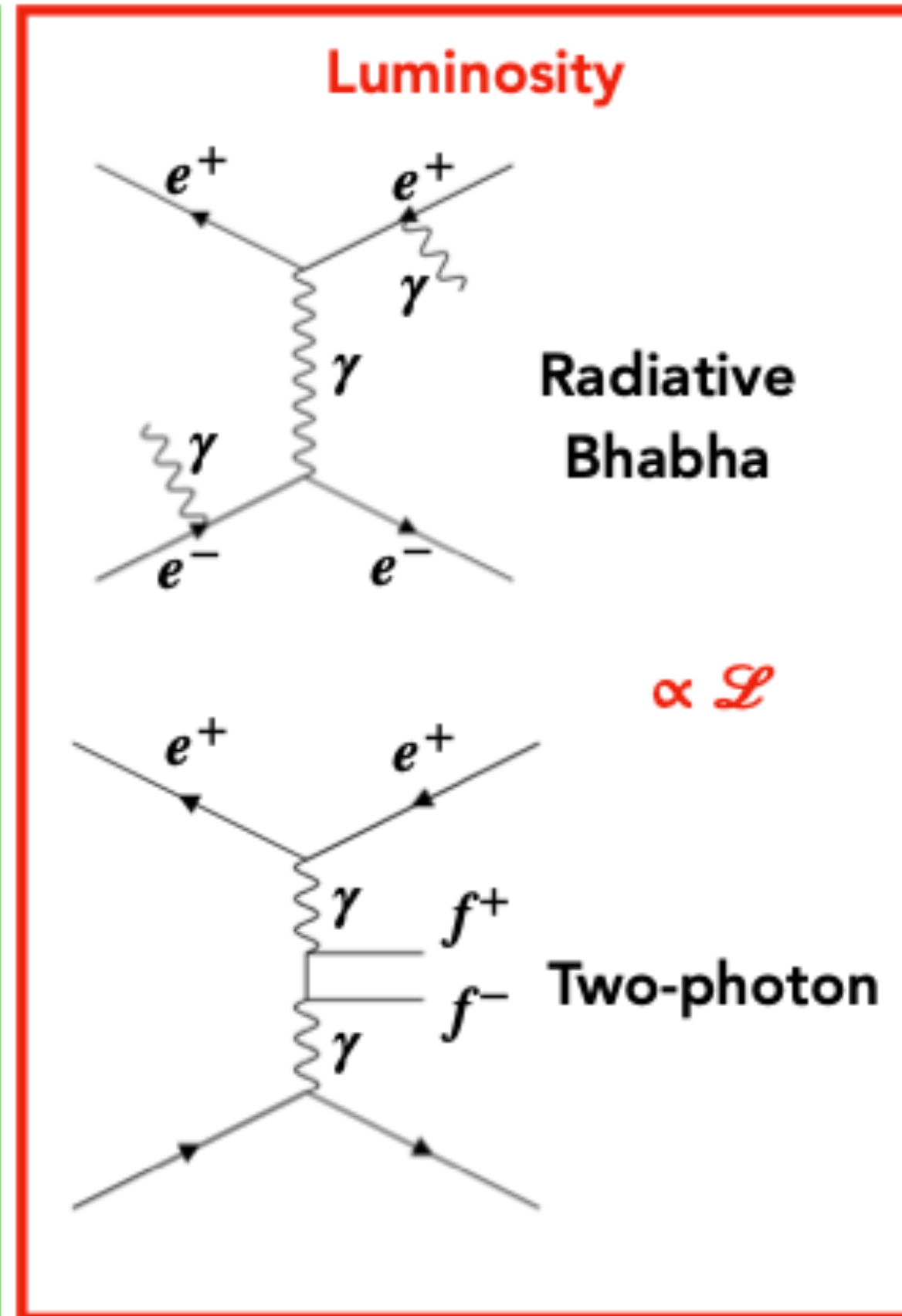
Backup

Belle II Backgrounds

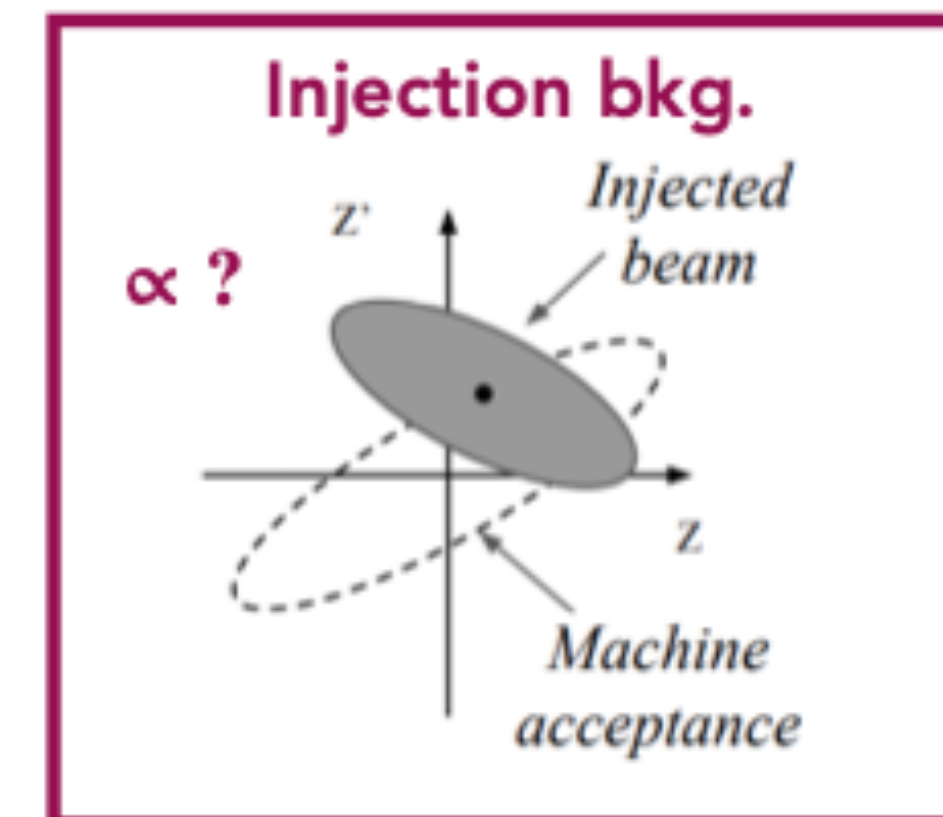
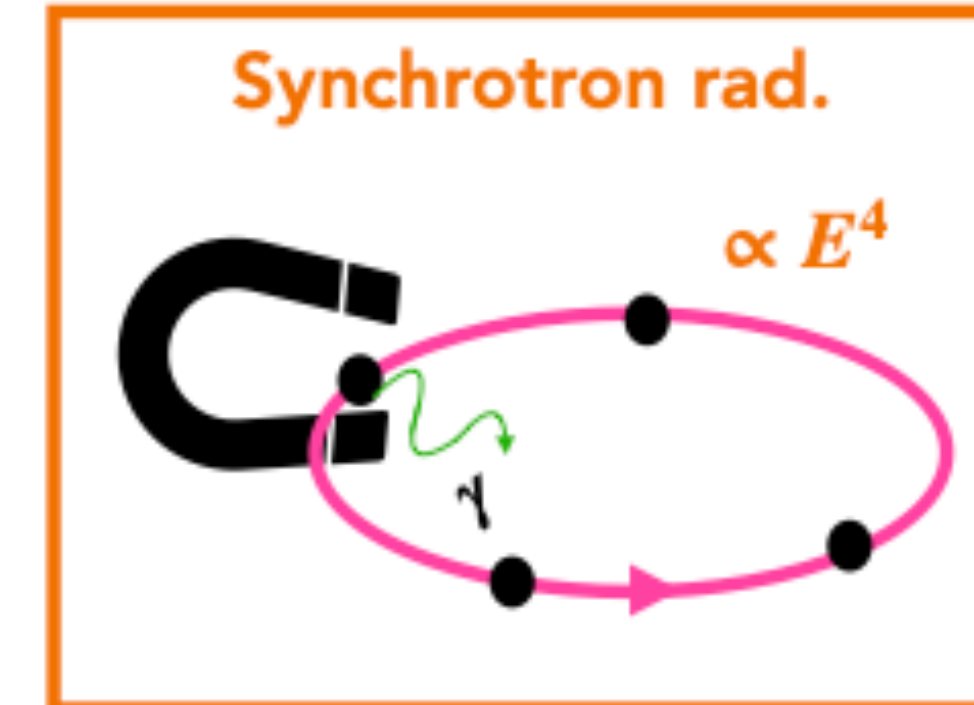
Background Processes



Slavomira Stefkova, slavomira.stefkova@kit.edu



3

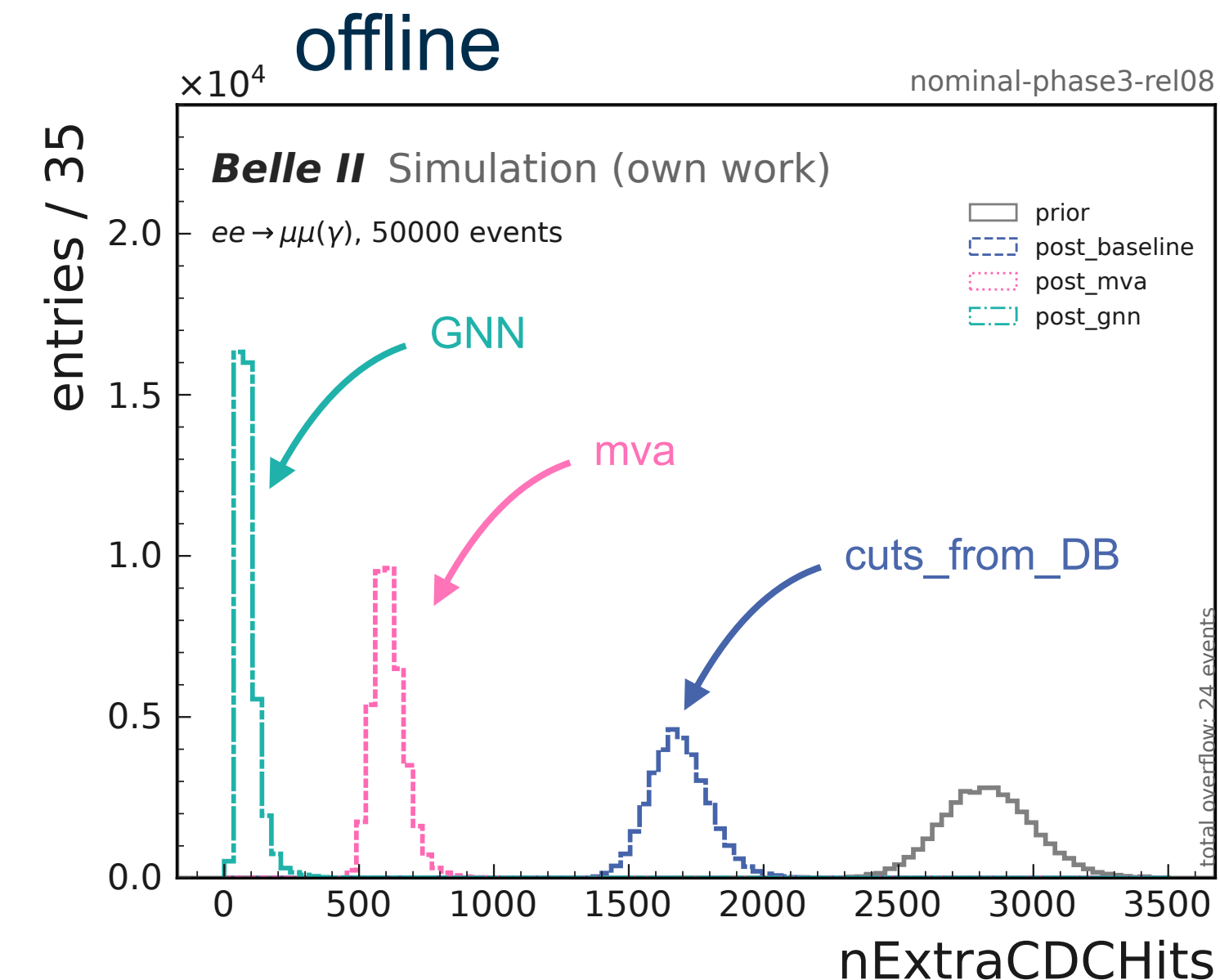
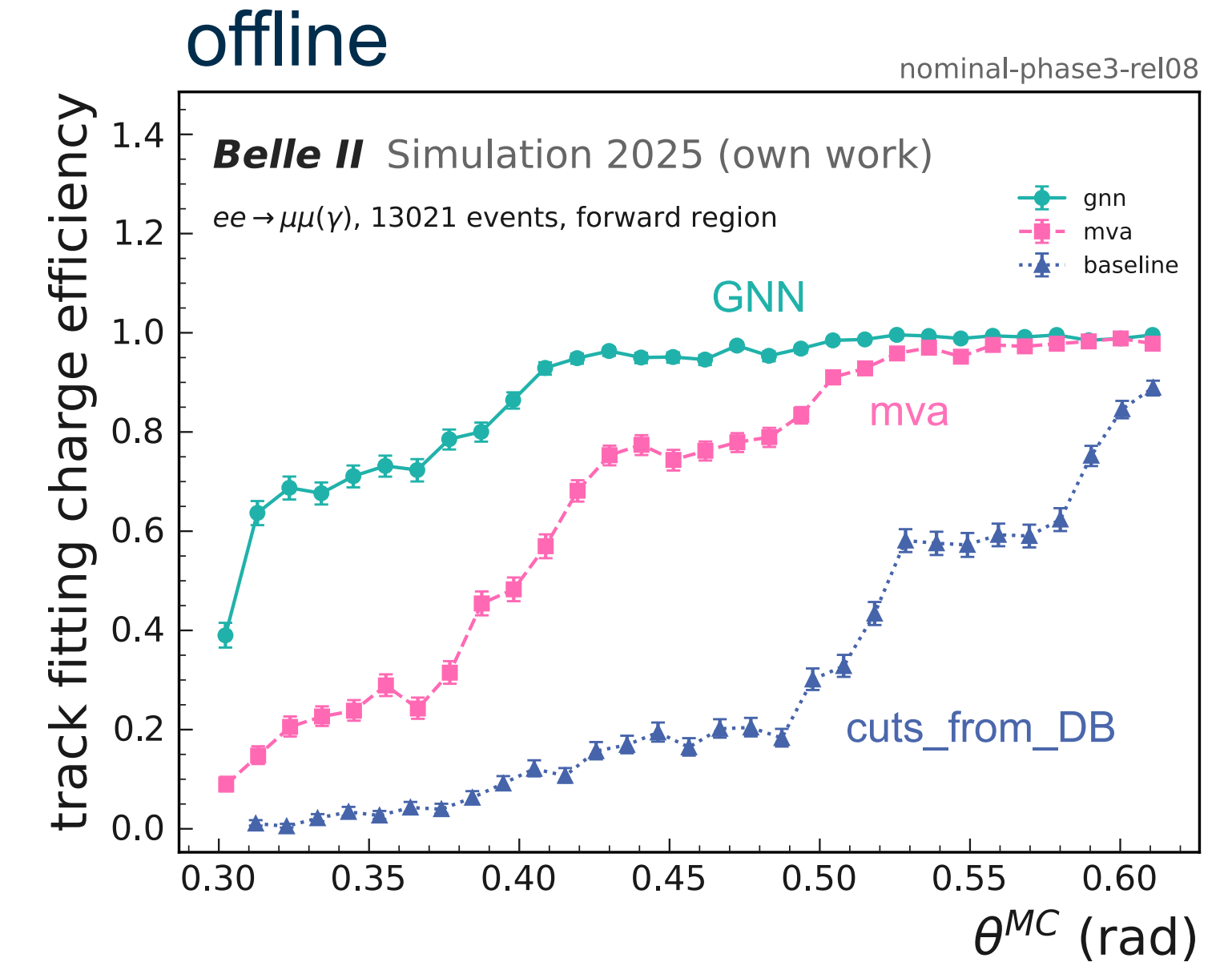


Belle II Group Meeting 

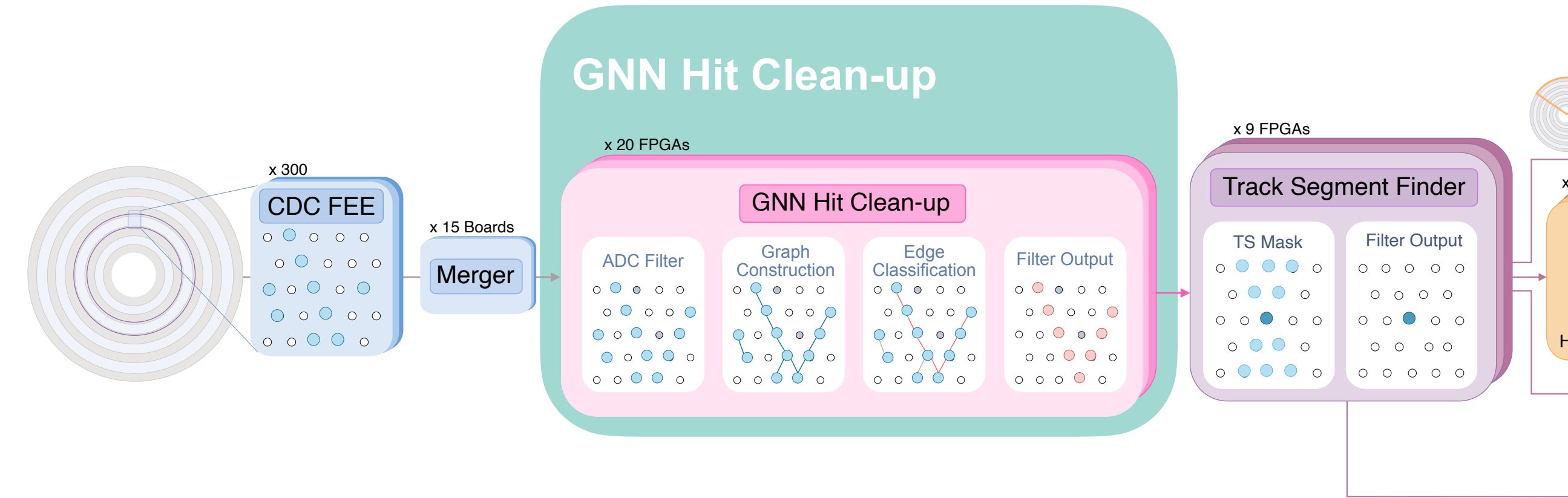
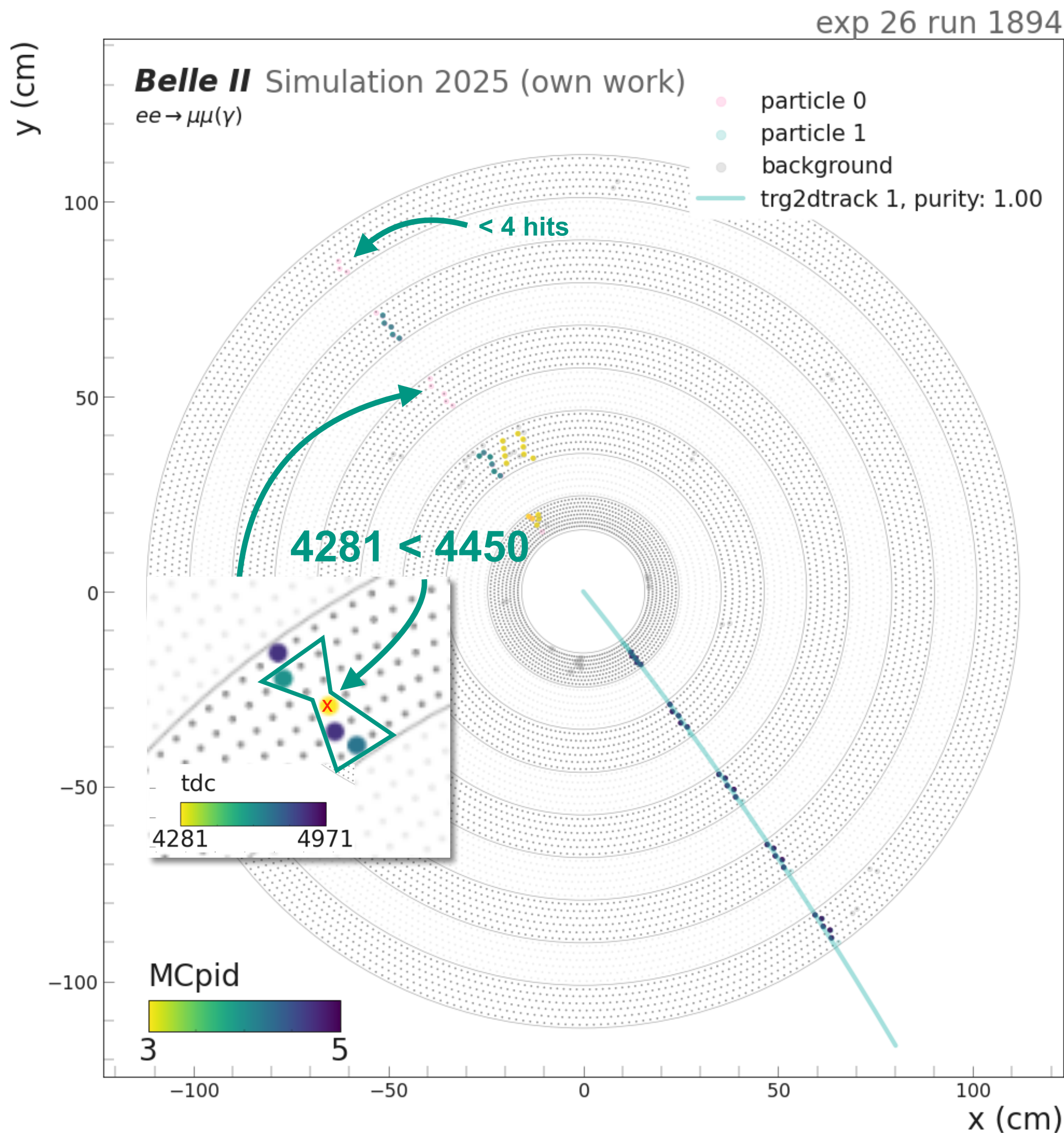
GNN Hit Clean-up Offline Results

- **Background levels** increase with luminosity, e.g. by scattering
- This affects **tracking** accuracy and computational cost ($\propto n_{\text{hits}}^2$) offline and online
- **GNN-based hit filtering** significantly **improves offline tracking*** (e.g. for future high background muon pairs vs. current Belle II baseline):
 - Hit background rejection 40.2% $\rightarrow$ 97.0%
 - Track fitting charge efficiency 67.1% $\rightarrow$ 93.0%
 - Track fitting fake rate 8.2% $\rightarrow$ 5.9%
 - Track finding and fitting time 543 ms/call $\rightarrow$ 100 ms/call

* track reconstruction (not on TRG level) with higher resolution inputs, full floating point GNN model and more sophisticated algorithms than on TRG level



Missing Track Segments



- sometimes tracks are not found due to **missing track segments**
 - CDCHits are there after filtering
 - but no track segment is being built
- the subsequent track finder asks for >6 track segments in order to suppress background
- with GNN hit filtering this number can and needs to be reduced
- **reasons for missing track segments:**
 - in some cases <4 hits within pattern shape of priority wire (at least 4 hits required by track segment finder)
 - TDC value outside of 500ns window

Training Setup

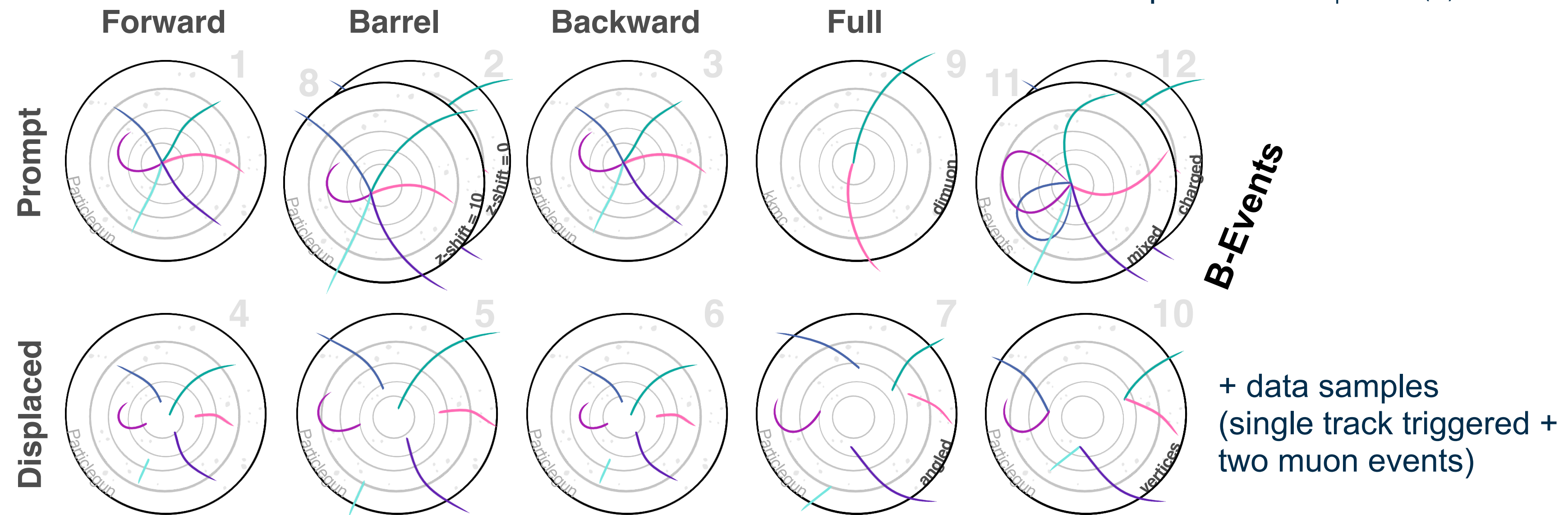
technical sample mix

- MC particle gun tracks**
 - prompt
 - z-shifted
 - displaced (& non-pointing)
 - non-pointing displaced vertices
 - + enriched with low p_T tracks*
- MC physics**
 - KKMC $\mu\mu(\gamma)$ samples
 - EvtGen $B\bar{B}$ samples
- data samples
 - low-bias triggered events
 - two muon events

- technical details
 - using a learning rate scheduler
 - AdamW optimiser
 - with early stopping
 - binary cross entropy with positive samples weight
 - dropout

Particle Gun Details

particles: muons
nTracks: 1 to 6 (uniform)
momentum: 0.05 to 6 GeV (uniform)
low_mom: 0.05 to 0.4 GeV (uniform)
low pT enrichment: poisson(1)



* additional poisson distributed low momentum tracks for curlers

** with background overlay from data

Sample mix inspired by [arXiv:2411.13596](https://arxiv.org/abs/2411.13596)

(End-to-End Multi-Track Reconstruction using Graph Neural Networks at Belle II)

Resource Constraints

- Assumption: **CDC segmentation into 20 sectors** (with overlaps)
- Latency Goal: 500ns
- The biggest of these sectors (SL 6) has ([Neu et al. 2024](#)):
 - $N_{\max} = 978$ nodes
 - $E_{\max} = 4545$ edges
- For each of the nodes and edges the respective Node and Edge Blocks of the network needs to be processed in parallel
- Due to the system frequency $f_{\text{sys}} = 256$ MHz and the CDC readout frequency $f_{\text{cdc}} = 32$ MHz we get the following **number of instances for each block**:

$$- \left\lceil \frac{N_{\max}}{8} \right\rceil = 123 \times \text{Node Block } O$$

$$- \left\lceil \frac{E_{\max}}{8} \right\rceil = 569 \times \text{Edge Block } R_1 \text{ \& Edge Block } R_2$$

