UCSD

JAVIER DUARTE

LISHEP SESSION C

JULY 6, 2021

# MACHINE LEARNING FOR (EXPERIMENTAL) HIGH ENERGY PHYSICS

# INTRODUCTION

Computer Physics Communications 49 (1988) 429–448 429
North-Holland, Amsterdam

**NEURAL NETWORKS AND CELLULAR AUTOMATA IN EXPERIMENTAL HIGH ENERGY PHYSICS**

B. DENBY

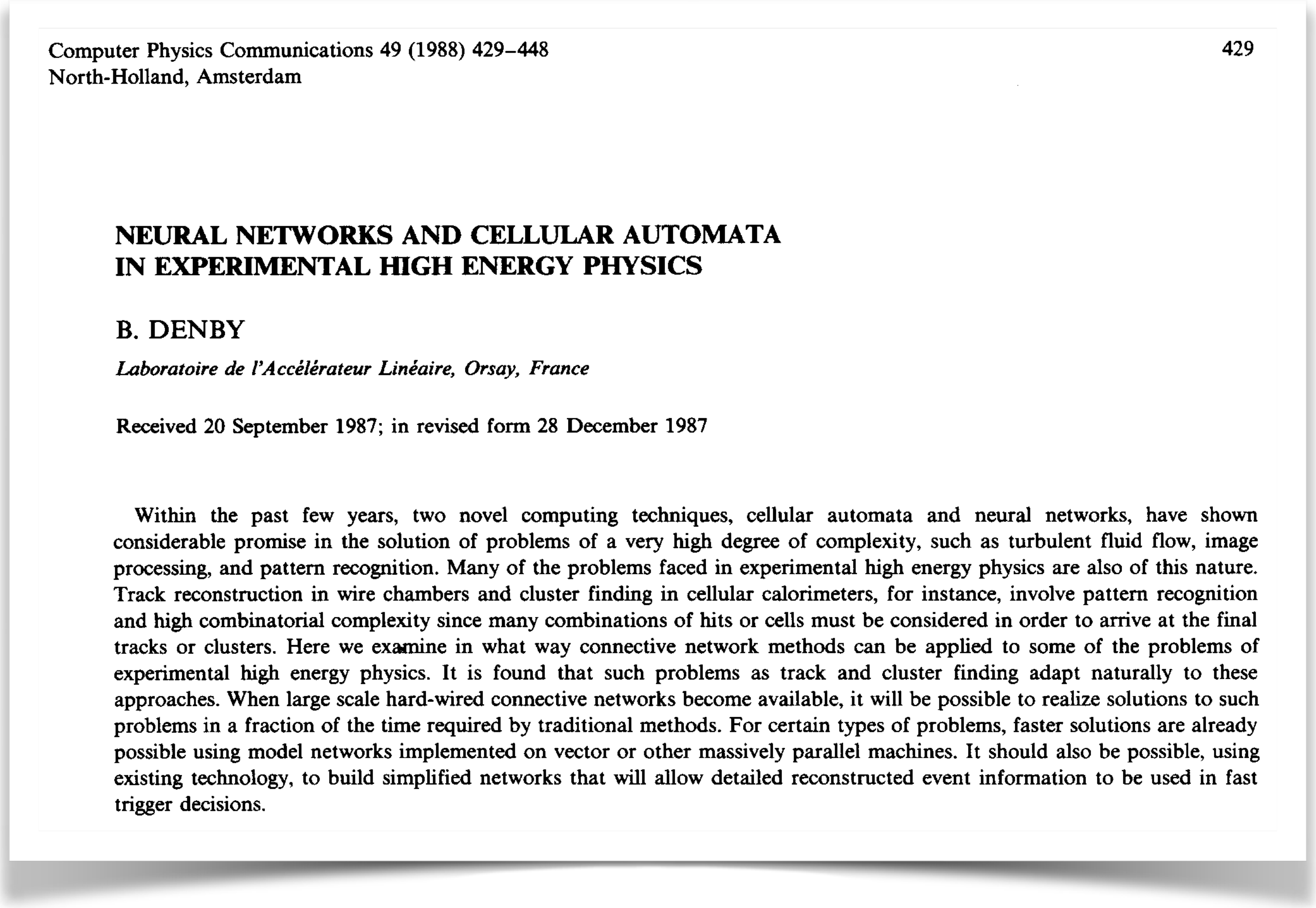
*Laboratoire de l'Accélérateur Linéaire, Orsay, France*

Received 20 September 1987; in revised form 28 December 1987

Within the past few years, two novel computing techniques, cellular automata and neural networks, have shown considerable promise in the solution of problems of a very high degree of complexity, such as turbulent fluid flow, image processing, and pattern recognition. Many of the problems faced in experimental high energy physics are also of this nature. Track reconstruction in wire chambers and cluster finding in cellular calorimeters, for instance, involve pattern recognition and high combinatorial complexity since many combinations of hits or cells must be considered in order to arrive at the final tracks or clusters. Here we examine in what way connective network methods can be applied to some of the problems of experimental high energy physics. It is found that such problems as track and cluster finding adapt naturally to these approaches. When large scale hard-wired connective networks become available, it will be possible to realize solutions to such problems in a fraction of the time required by traditional methods. For certain types of problems, faster solutions are already possible using model networks implemented on vector or other massively parallel machines. It should also be possible, using existing technology, to build simplified networks that will allow detailed reconstructed event information to be used in fast trigger decisions.

# INTRODUCTION

- Particle physics has been linked to machine learning and neural networks since the 1980s!



Computer Physics Communications 49 (1988) 429–448 429
North-Holland, Amsterdam

NEURAL NETWORKS AND CELLULAR AUTOMATA
IN EXPERIMENTAL HIGH ENERGY PHYSICS

B. DENBY

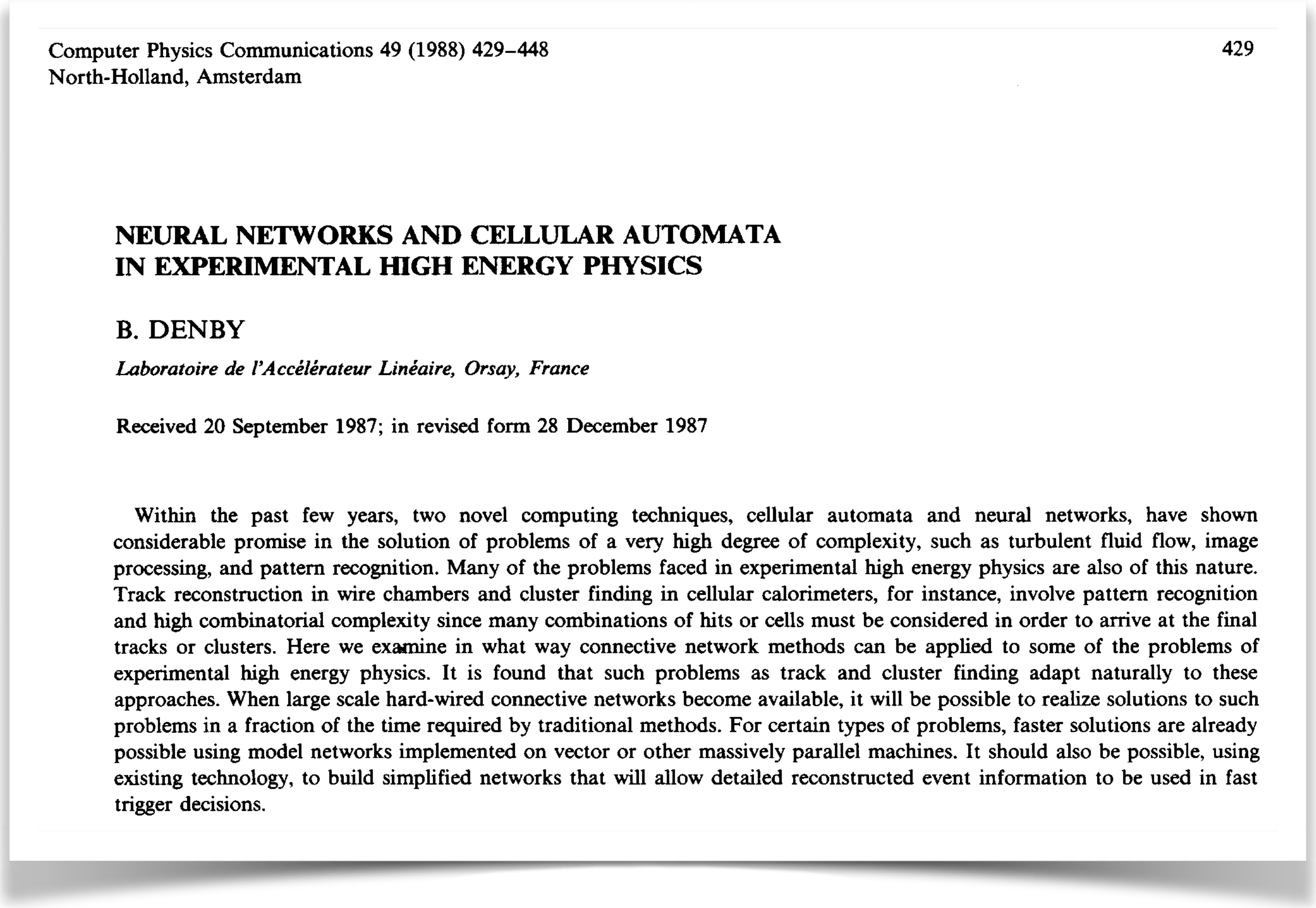
*Laboratoire de l'Accélérateur Linéaire, Orsay, France*

Received 20 September 1987; in revised form 28 December 1987

Within the past few years, two novel computing techniques, cellular automata and neural networks, have shown considerable promise in the solution of problems of a very high degree of complexity, such as turbulent fluid flow, image processing, and pattern recognition. Many of the problems faced in experimental high energy physics are also of this nature. Track reconstruction in wire chambers and cluster finding in cellular calorimeters, for instance, involve pattern recognition and high combinatorial complexity since many combinations of hits or cells must be considered in order to arrive at the final tracks or clusters. Here we examine in what way connective network methods can be applied to some of the problems of experimental high energy physics. It is found that such problems as track and cluster finding adapt naturally to these approaches. When large scale hard-wired connective networks become available, it will be possible to realize solutions to such problems in a fraction of the time required by traditional methods. For certain types of problems, faster solutions are already possible using model networks implemented on vector or other massively parallel machines. It should also be possible, using existing technology, to build simplified networks that will allow detailed reconstructed event information to be used in fast trigger decisions.

# INTRODUCTION

- Particle physics has been linked to machine learning and neural networks since the 1980s!
- In recent years, the use of ML has expanded into new territory



Computer Physics Communications 49 (1988) 429–448
North-Holland, Amsterdam 429

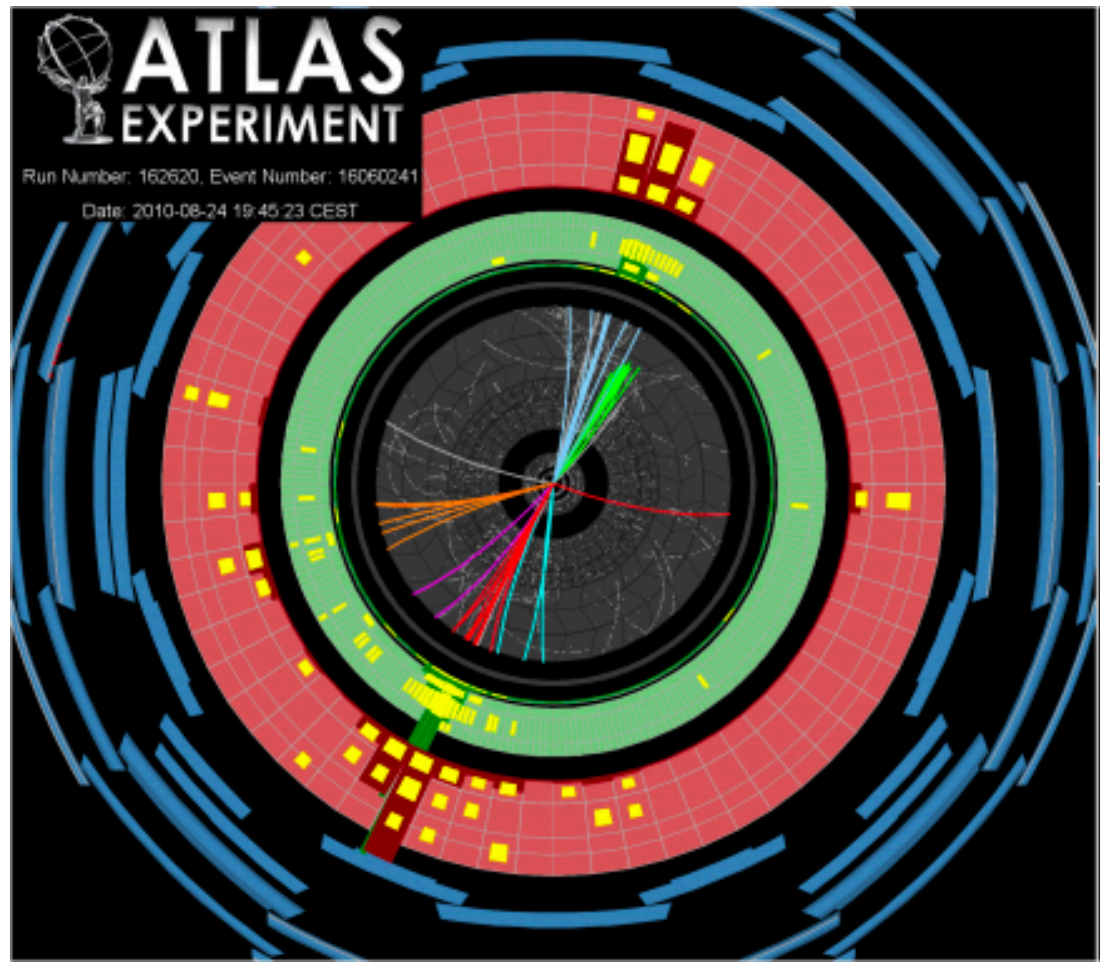
NEURAL NETWORKS AND CELLULAR AUTOMATA
IN EXPERIMENTAL HIGH ENERGY PHYSICS

B. DENBY
*Laboratoire de l'Accélérateur Linéaire, Orsay, France*

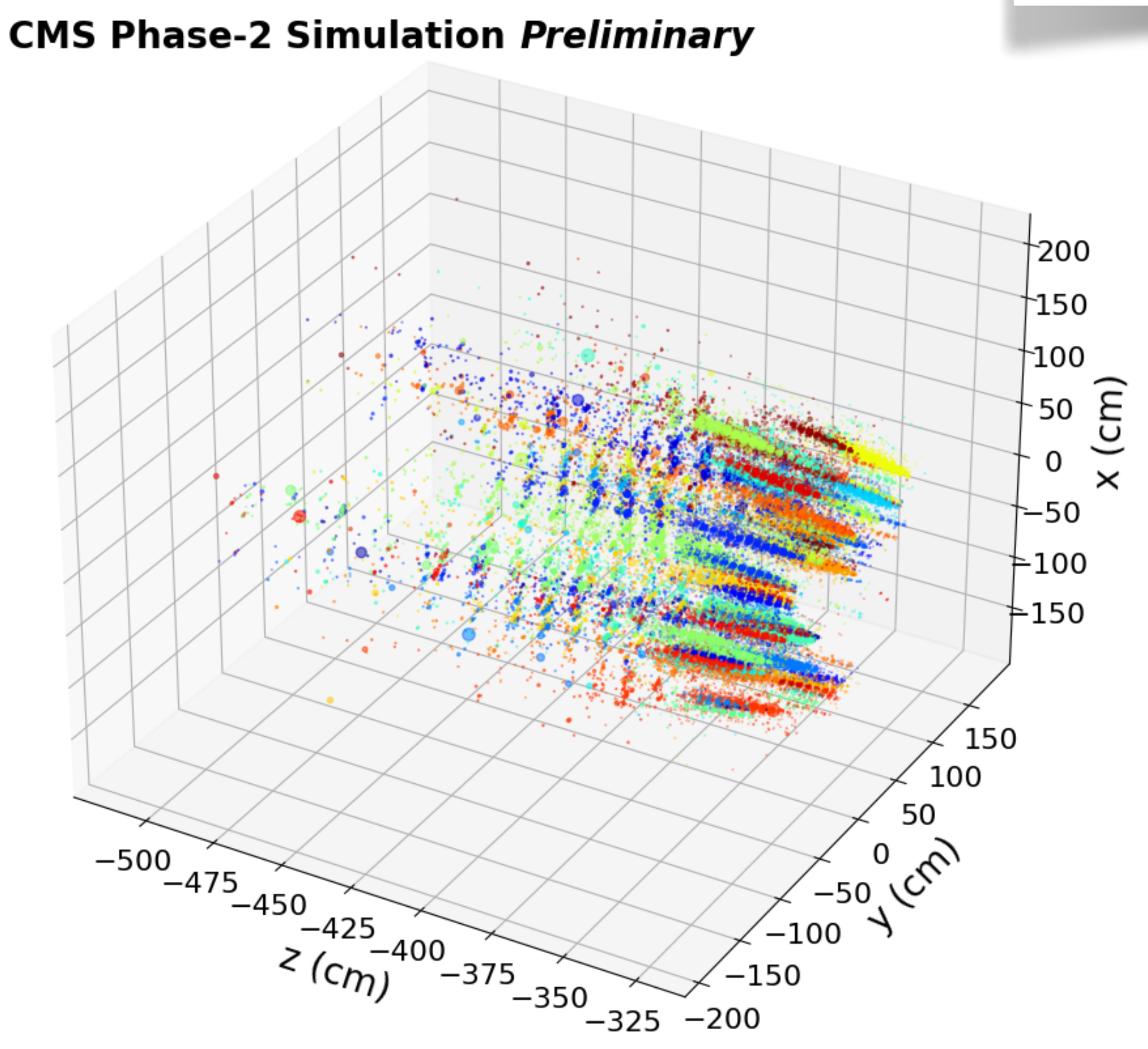
Received 20 September 1987; in revised form 28 December 1987

Within the past few years, two novel computing techniques, cellular automata and neural networks, have shown considerable promise in the solution of problems of a very high degree of complexity, such as turbulent fluid flow, image processing, and pattern recognition. Many of the problems faced in experimental high energy physics are also of this nature. Track reconstruction in wire chambers and cluster finding in cellular calorimeters, for instance, involve pattern recognition and high combinatorial complexity since many combinations of hits or cells must be considered in order to arrive at the final tracks or clusters. Here we examine in what way connective network methods can be applied to some of the problems of experimental high energy physics. It is found that such problems as track and cluster finding adapt naturally to these approaches. When large scale hard-wired connective networks become available, it will be possible to realize solutions to such problems in a fraction of the time required by traditional methods. For certain types of problems, faster solutions are already possible using model networks implemented on vector or other massively parallel machines. It should also be possible, using existing technology, to build simplified networks that will allow detailed reconstructed event information to be used in fast trigger decisions.

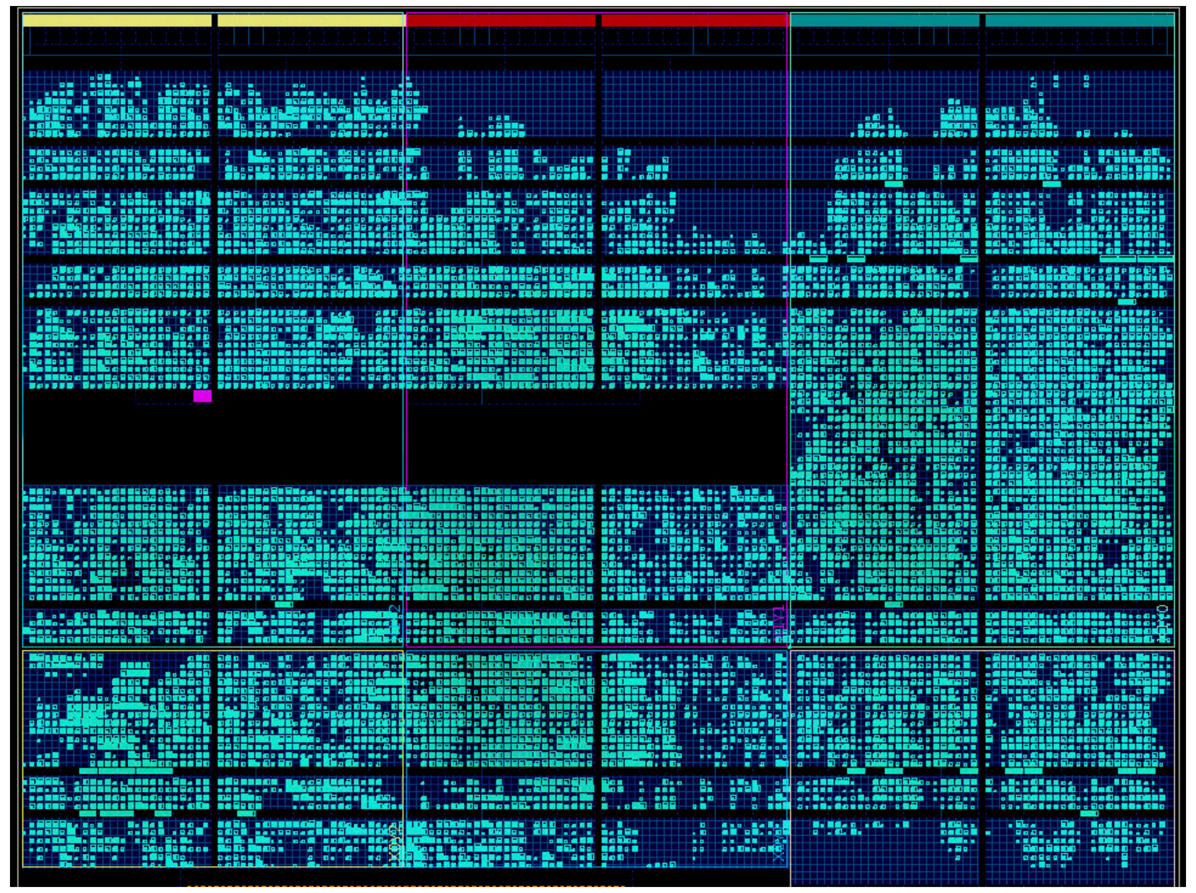
ML for "jet tagging"



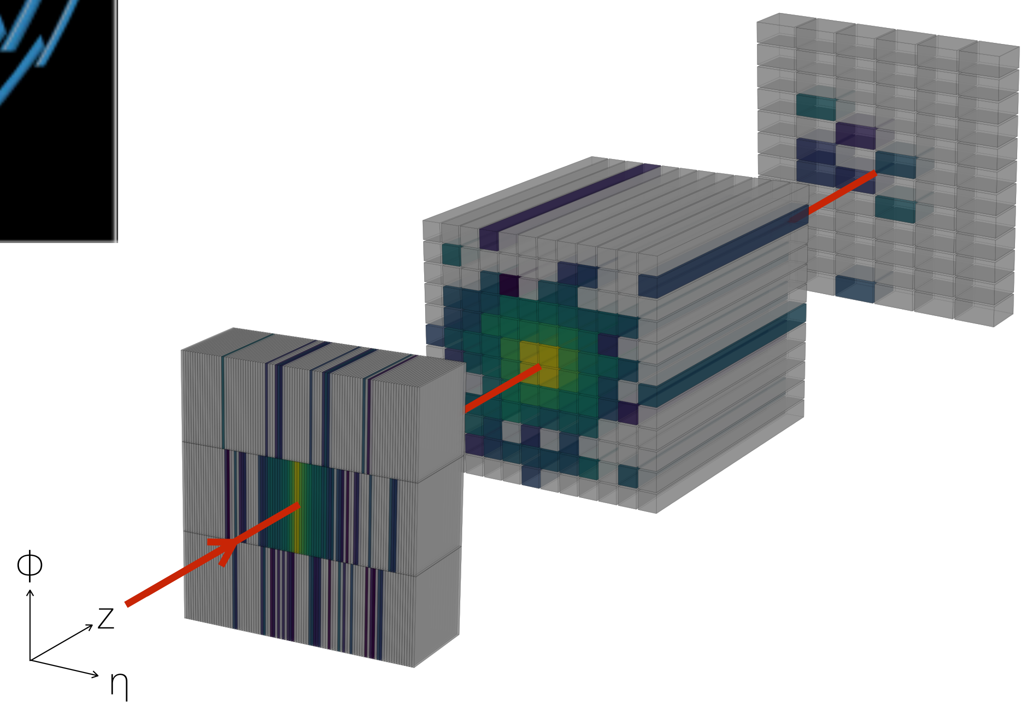

ML for reconstruction




Fast ML for trigger



ML for generation/simulation



# INTRODUCTION

- Particle physics has been linked to machine learning and neural networks since the 1980s!
- In recent years, the use of ML has expanded into new territory
- Broadly speaking, we're interested in two advantages from ML: sensitivity to physics and computational performance

Computer Physics Communications 49 (1988) 429–448
North-Holland, Amsterdam 429

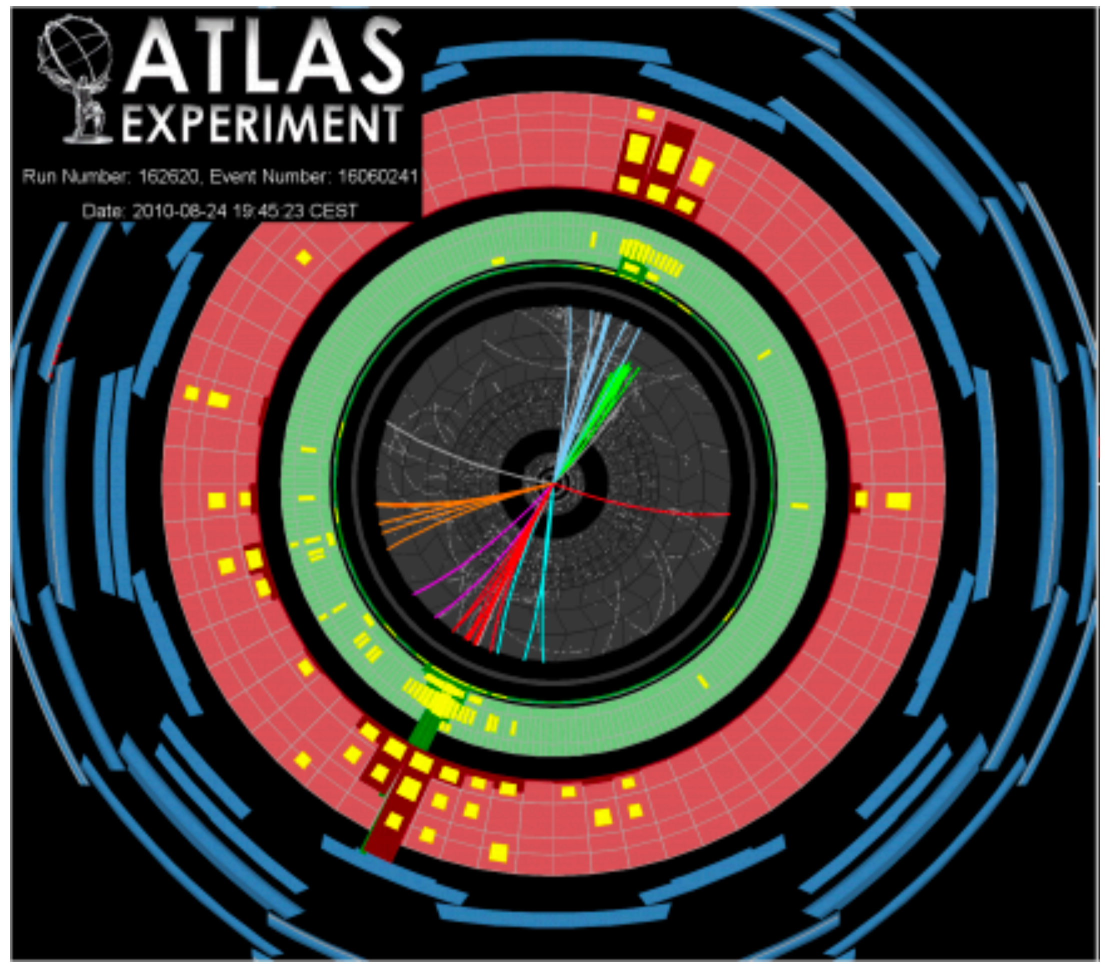
**NEURAL NETWORKS AND CELLULAR AUTOMATA IN EXPERIMENTAL HIGH ENERGY PHYSICS**

B. DENBY
*Laboratoire de l'Accélérateur Linéaire, Orsay, France*

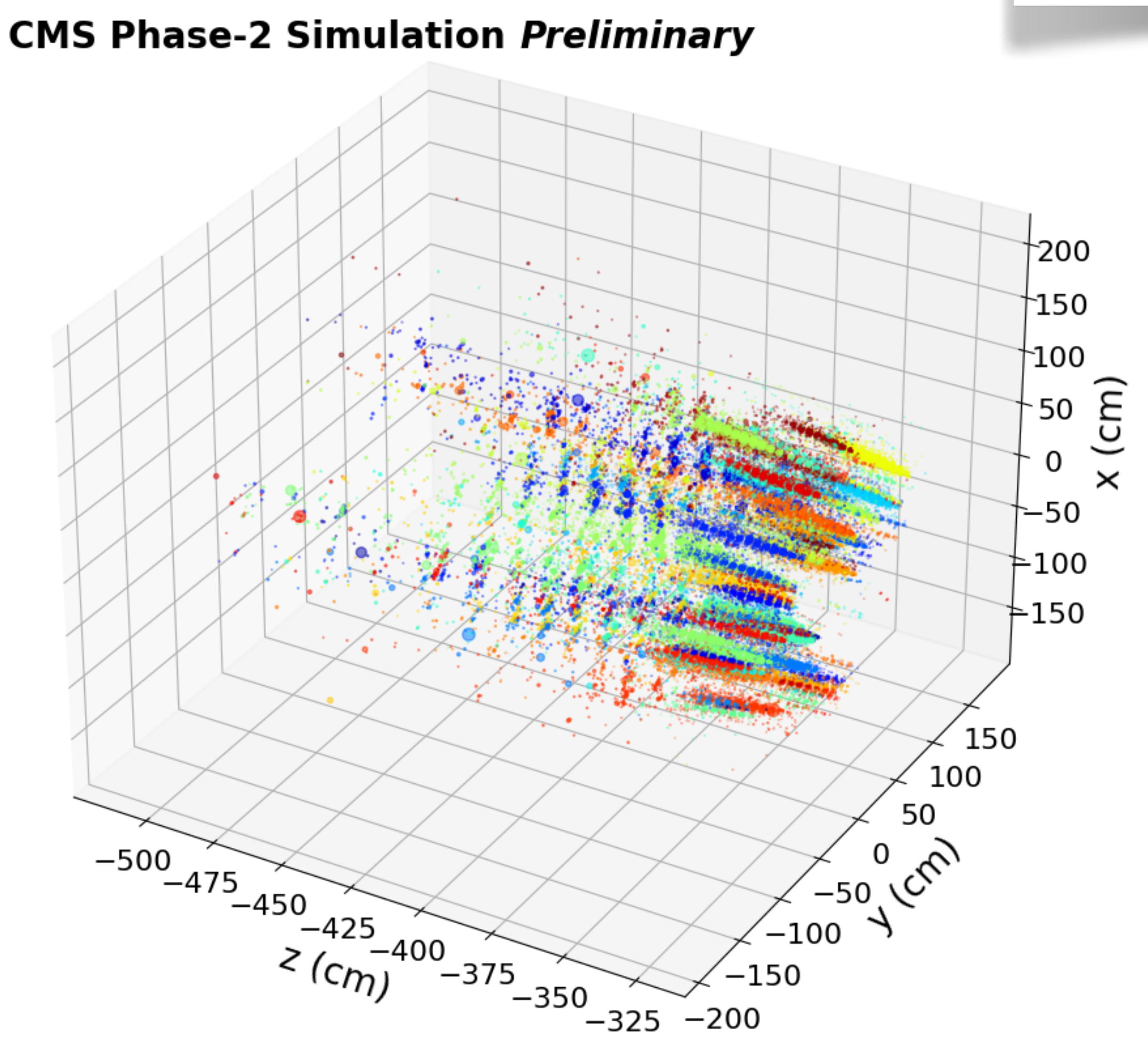
Received 20 September 1987; in revised form 28 December 1987

Within the past few years, two novel computing techniques, cellular automata and neural networks, have shown considerable promise in the solution of problems of a very high degree of complexity, such as turbulent fluid flow, image processing, and pattern recognition. Many of the problems faced in experimental high energy physics are also of this nature. Track reconstruction in wire chambers and cluster finding in cellular calorimeters, for instance, involve pattern recognition and high combinatorial complexity since many combinations of hits or cells must be considered in order to arrive at the final tracks or clusters. Here we examine in what way connective network methods can be applied to some of the problems of experimental high energy physics. It is found that such problems as track and cluster finding adapt naturally to these approaches. When large scale hard-wired connective networks become available, it will be possible to realize solutions to such problems in a fraction of the time required by traditional methods. For certain types of problems, faster solutions are already possible using model networks implemented on vector or other massively parallel machines. It should also be possible, using existing technology, to build simplified networks that will allow detailed reconstructed event information to be used in fast trigger decisions.

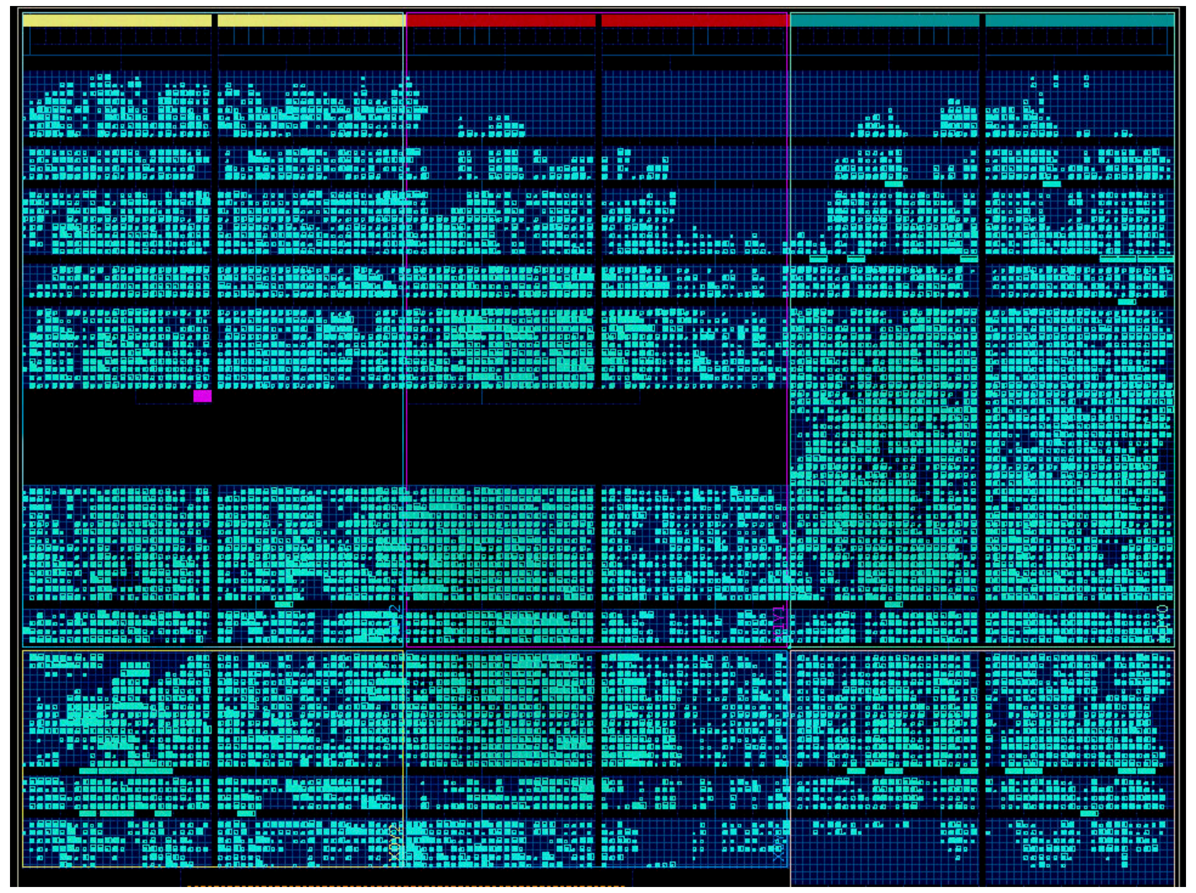
ML for "jet tagging"



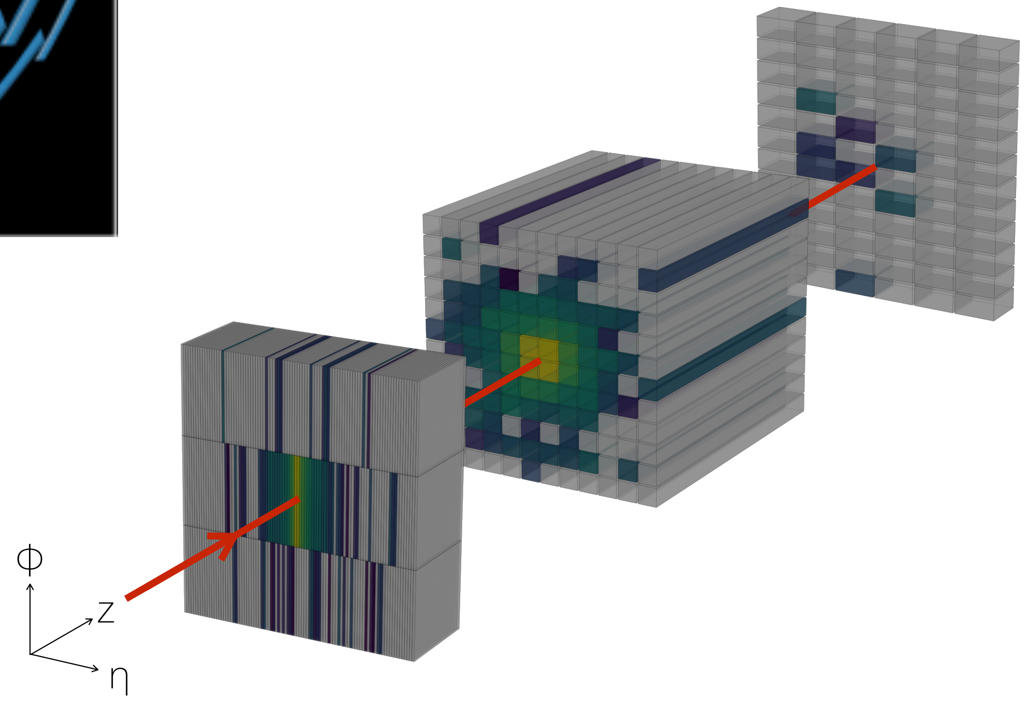

ML for reconstruction




Fast ML for trigger



ML for generation/simulation



# WHY MACHINE LEARNING?

# WHY MACHINE LEARNING?



- Deep (machine) learning is the use of **structured neural networks** with **many hidden layers** as generic functions to approximate the optimal solution for a given task

# WHY MACHINE LEARNING?



- Deep (machine) learning is the use of **structured neural networks** with **many hidden layers** as generic functions to approximate the optimal solution for a given task
- Why use deep neural networks in particle physics?

# WHY MACHINE LEARNING?



- Deep (machine) learning is the use of **structured neural networks** with **many hidden layers** as generic functions to approximate the optimal solution for a given task
- Why use deep neural networks in particle physics?
  - They work!

# WHY MACHINE LEARNING?



- Deep (machine) learning is the use of **structured neural networks** with **many hidden layers** as generic functions to approximate the optimal solution for a given task
- Why use deep neural networks in particle physics?
  - They work!
  - Lots of data & simulation to train them

# WHY MACHINE LEARNING?



- Deep (machine) learning is the use of **structured neural networks** with **many hidden layers** as generic functions to approximate the optimal solution for a given task
- Why use deep neural networks in particle physics?
  - They work!
  - Lots of data & simulation to train them
  - Fast ways to train them

TensorFlow


Keras PyTorch

# WHY MACHINE LEARNING?



- Deep (machine) learning is the use of **structured neural networks** with **many hidden layers** as generic functions to approximate the optimal solution for a given task
- Why use deep neural networks in particle physics?
  - They work!
  - Lots of data & simulation to train them
  - Fast ways to train them
  - Possibly gain new insights…

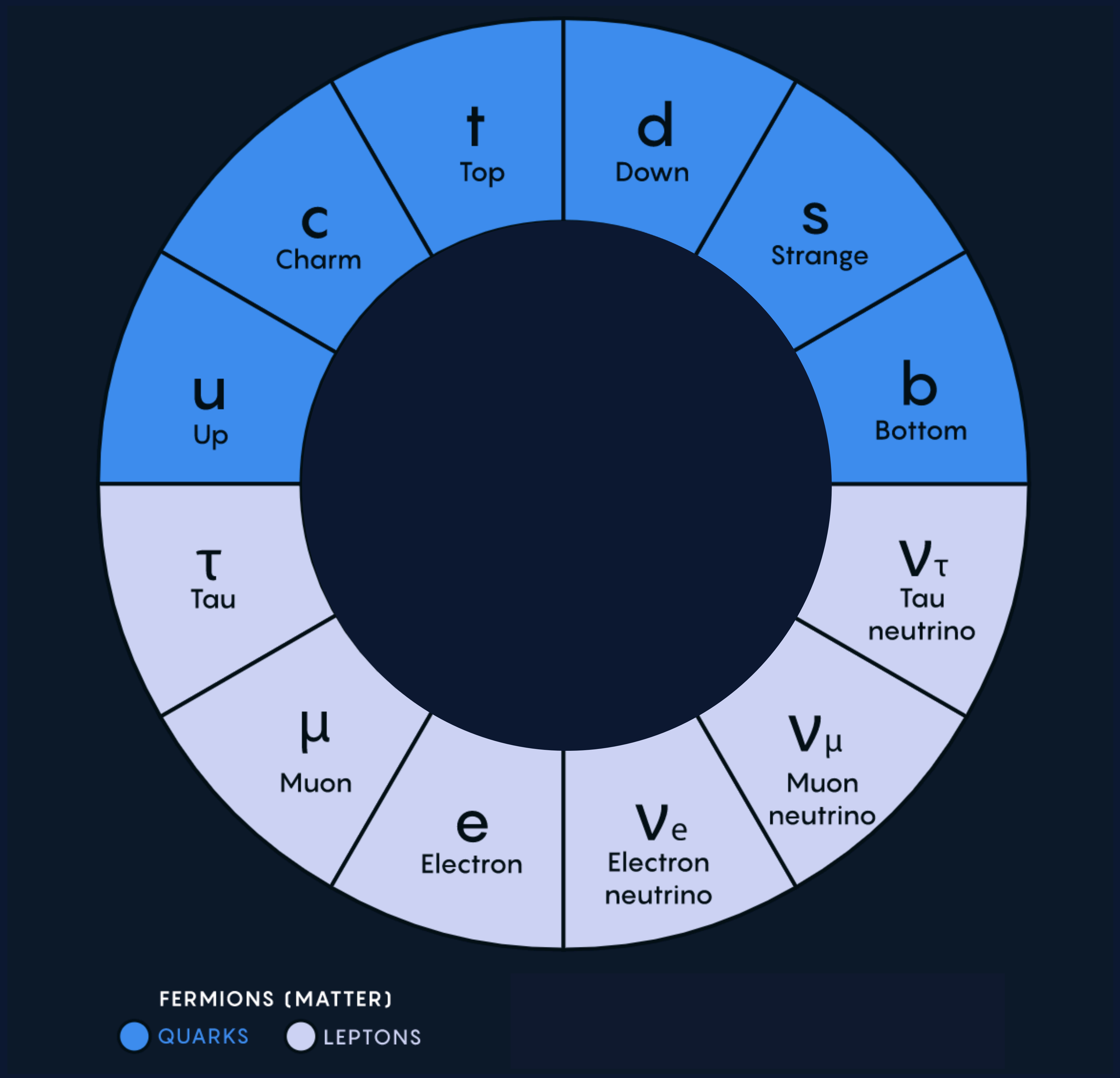
TensorFlow


Keras PyTorch

# ML FOR JET TAGGING
# ML FOR GEN/SIM
# FAST ML FOR TRIGGER

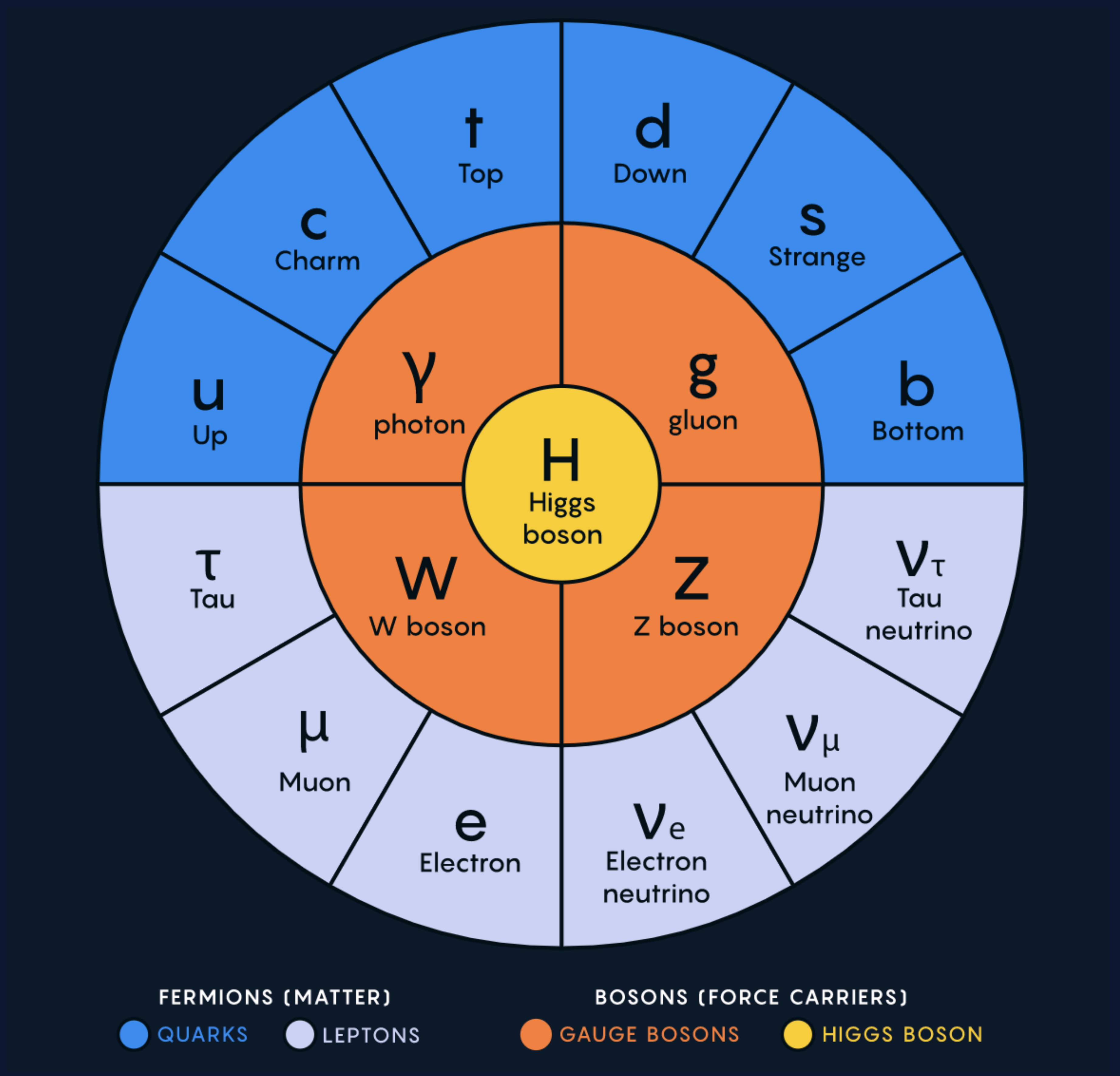
# HIGGS BOSON IN THE STANDARD MODEL





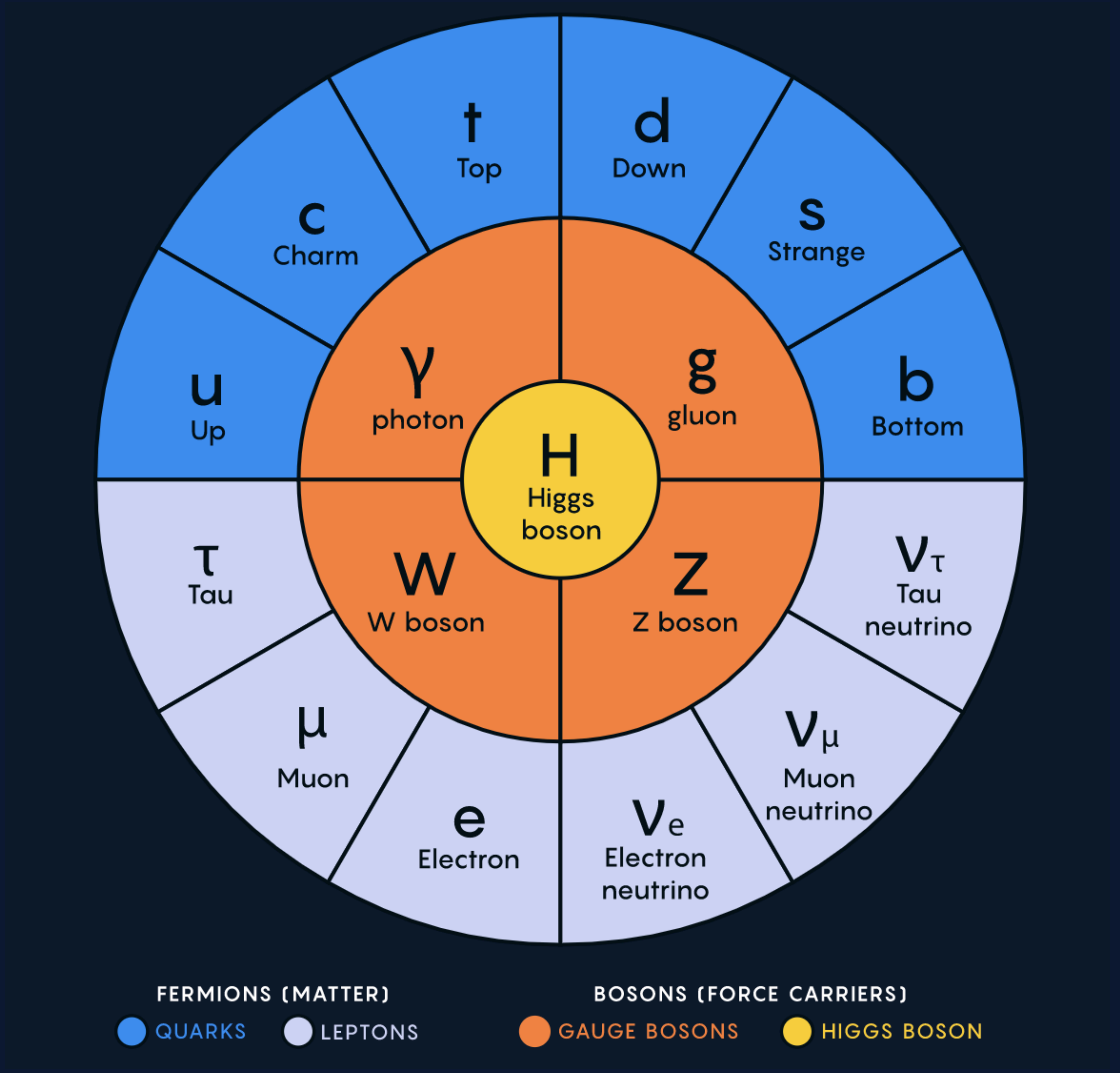
# HIGGS BOSON IN THE STANDARD MODEL

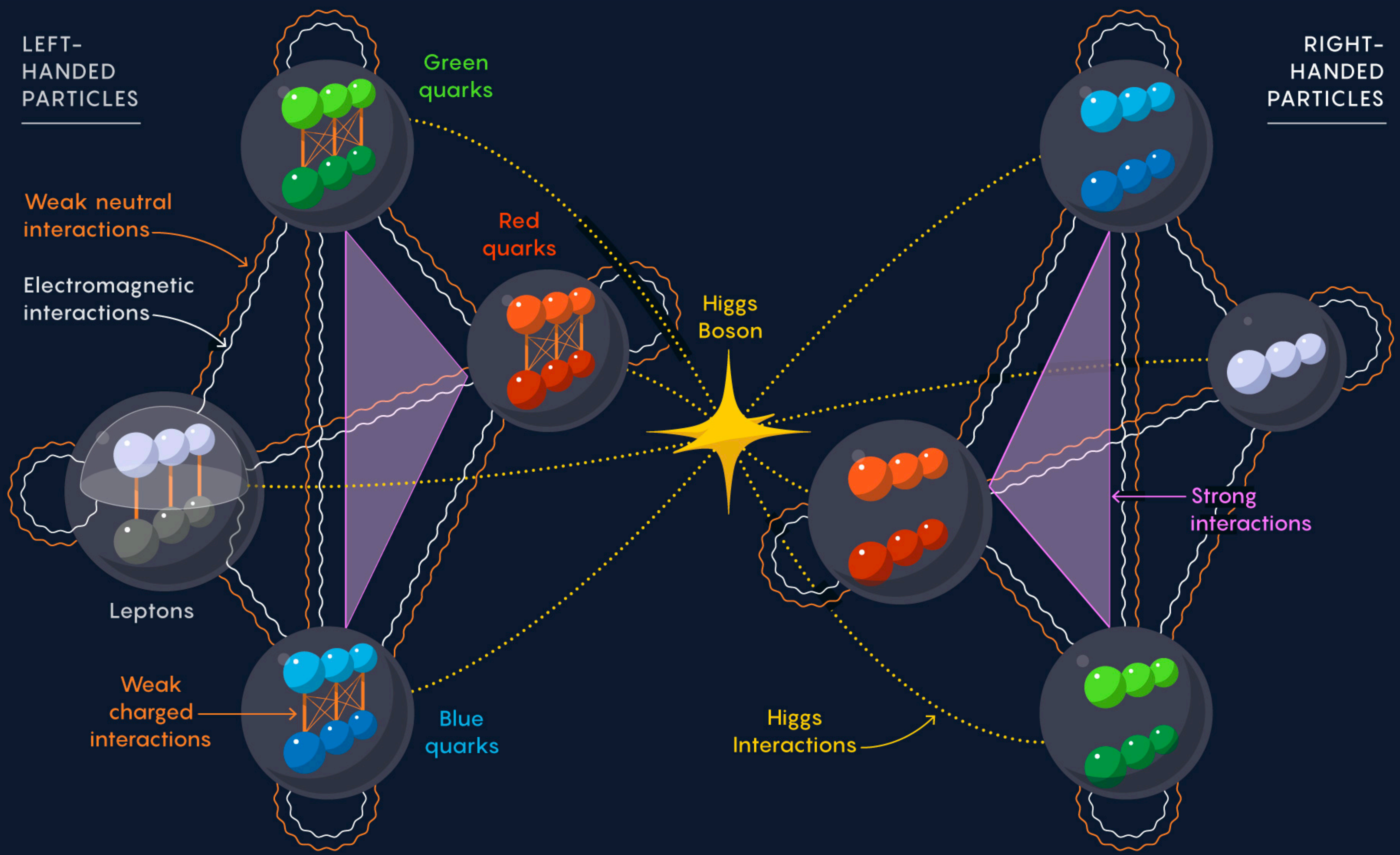




# HIGGS BOSON IN THE STANDARD MODEL

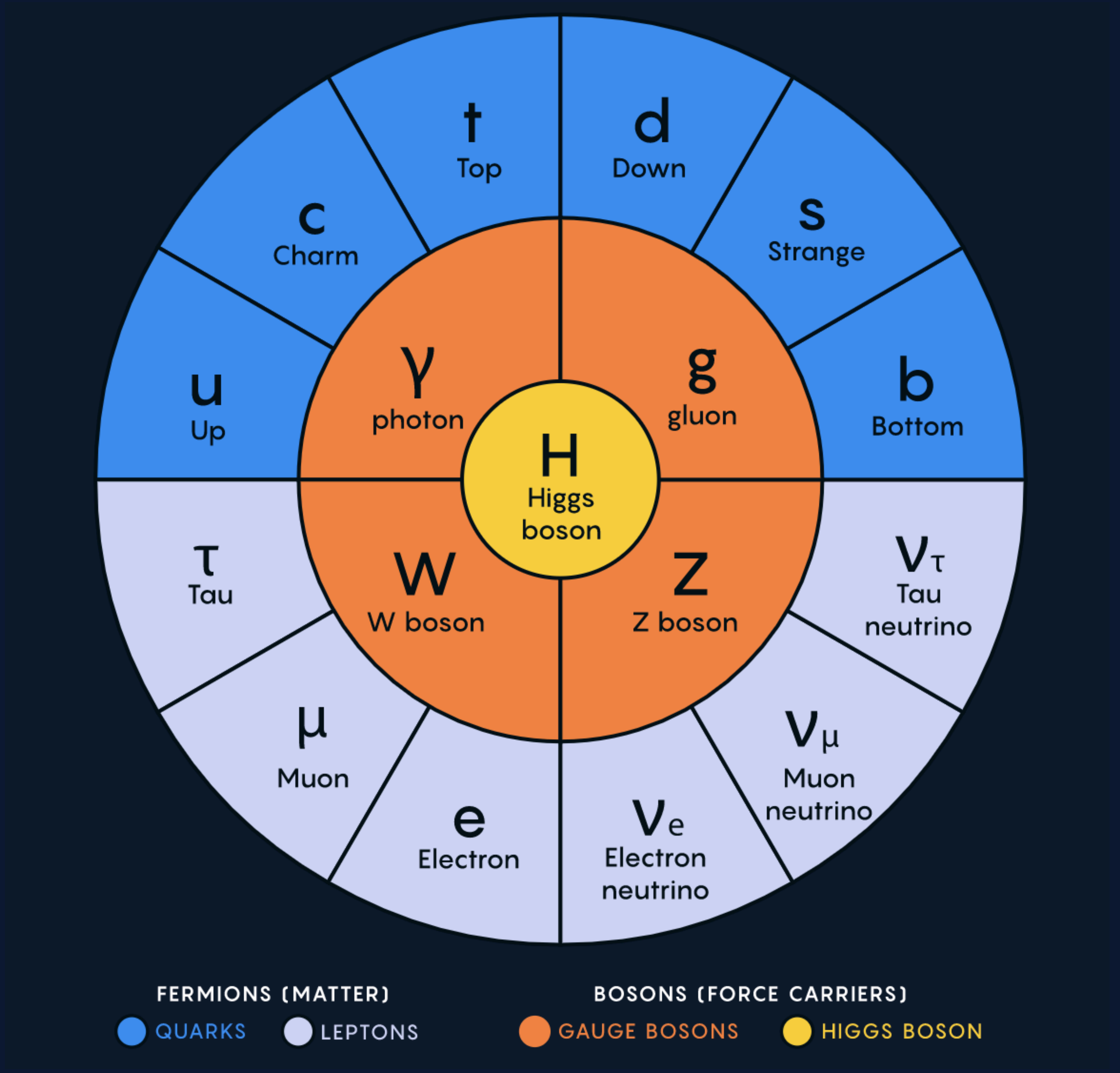


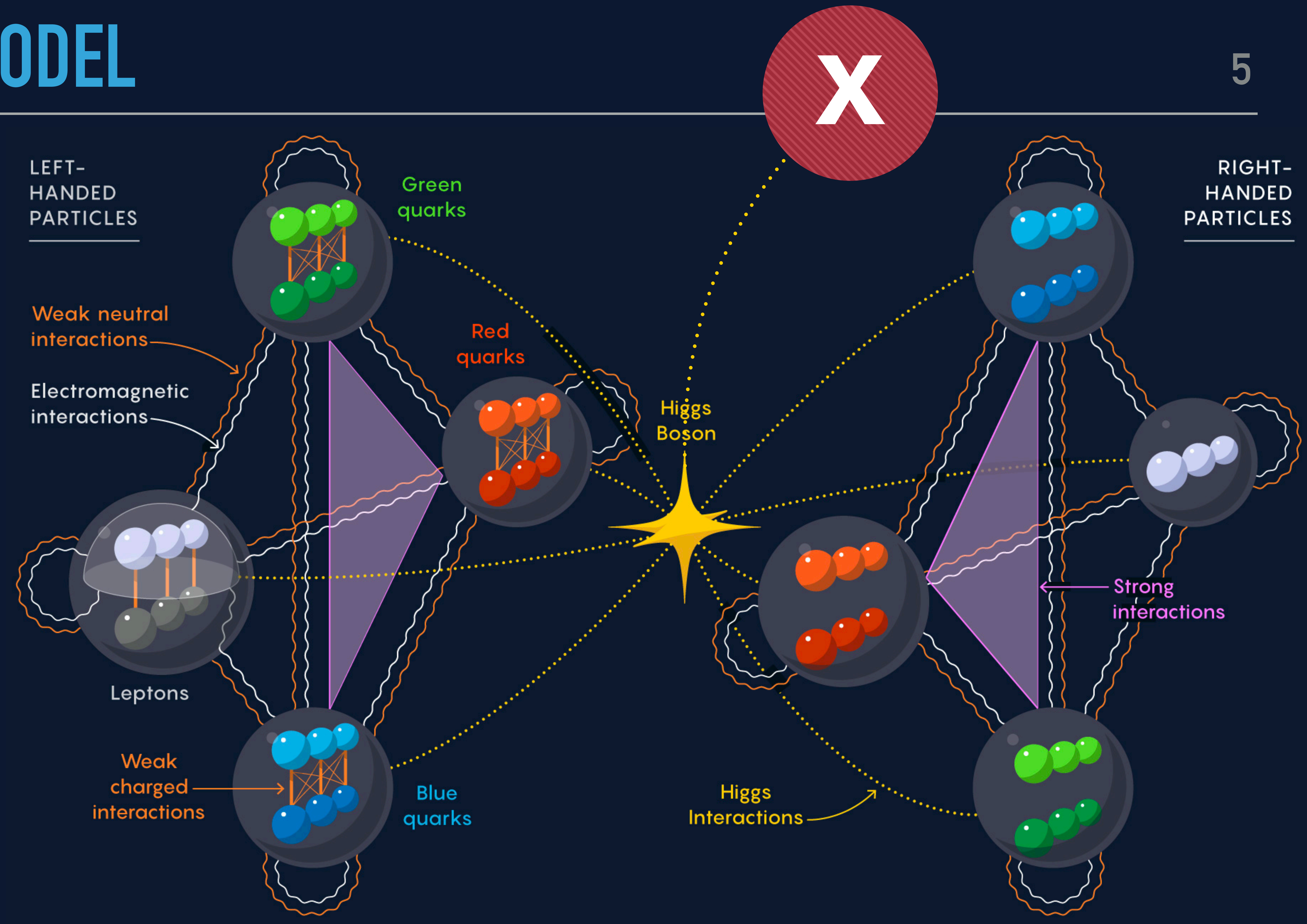


# HIGGS BOSON IN THE STANDARD MODEL

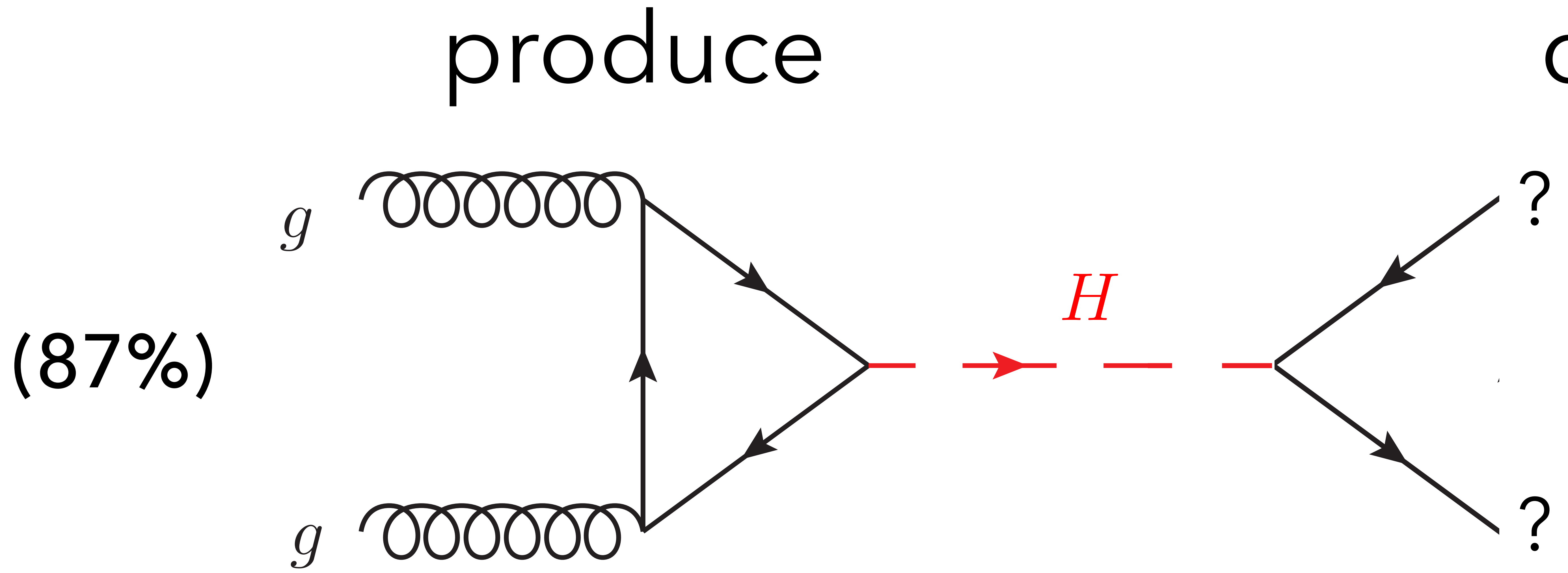


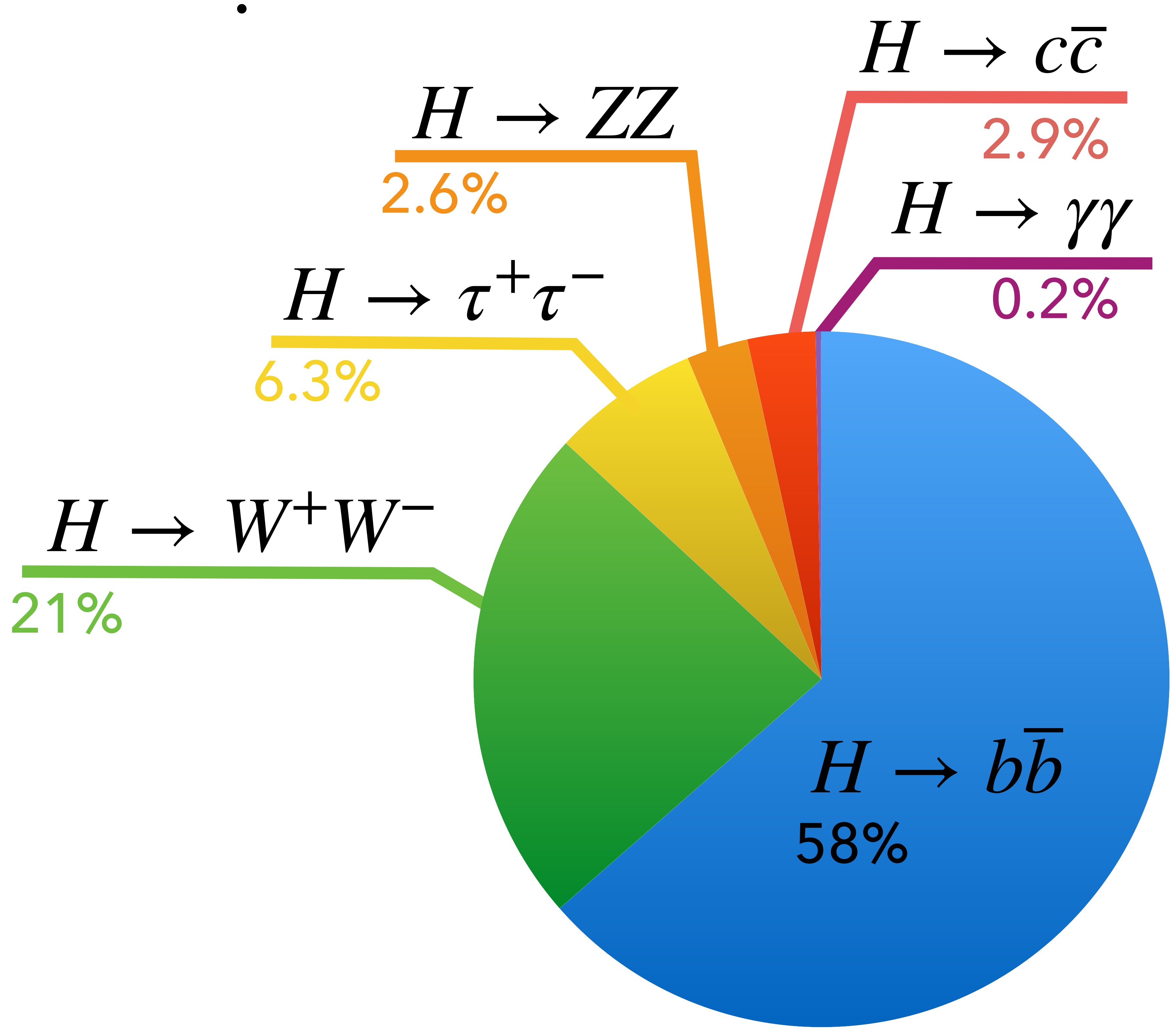



- Higgs boson is the ***centerpiece:*** all particles interact with it
- May be a link to new particles or interactions

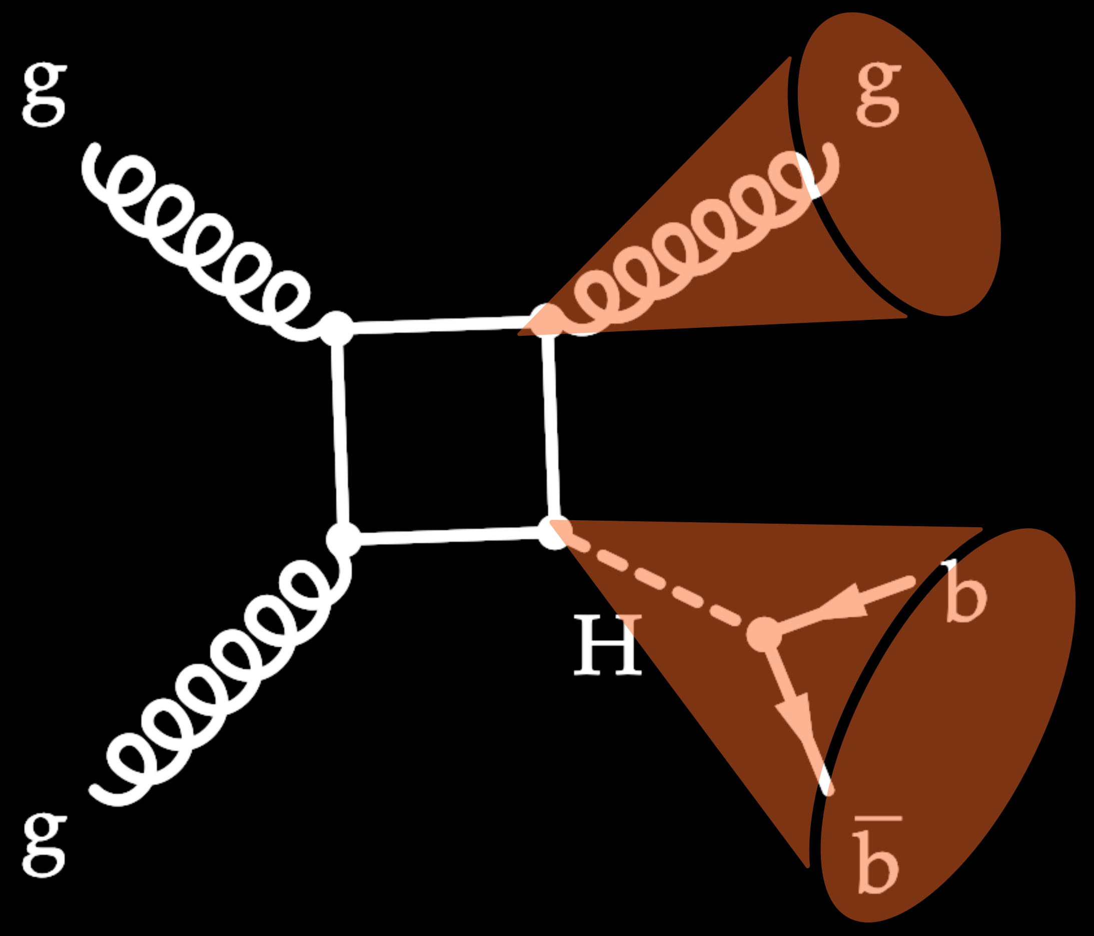
# DETECTING HIGGS BOSON DECAYS

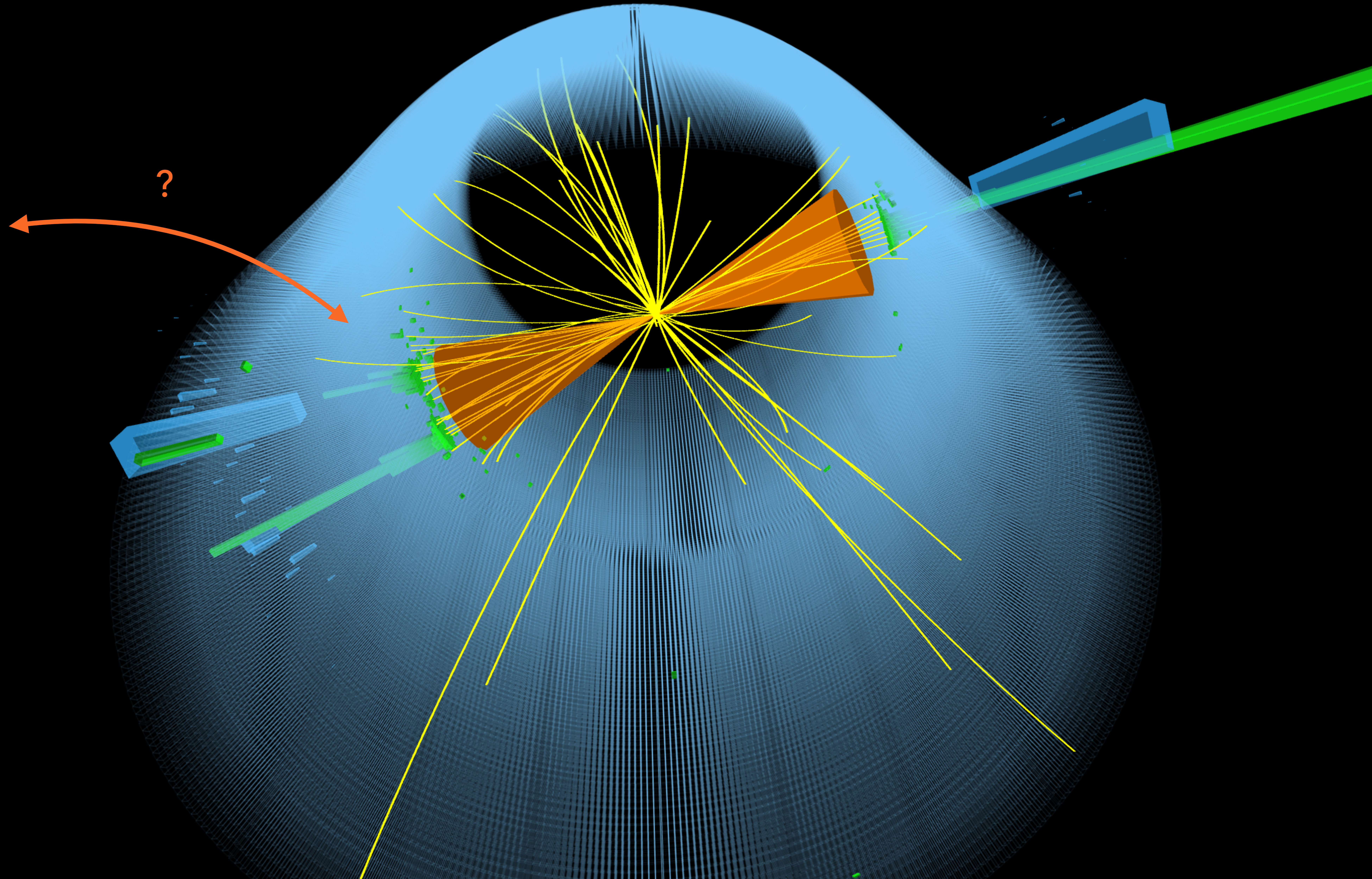


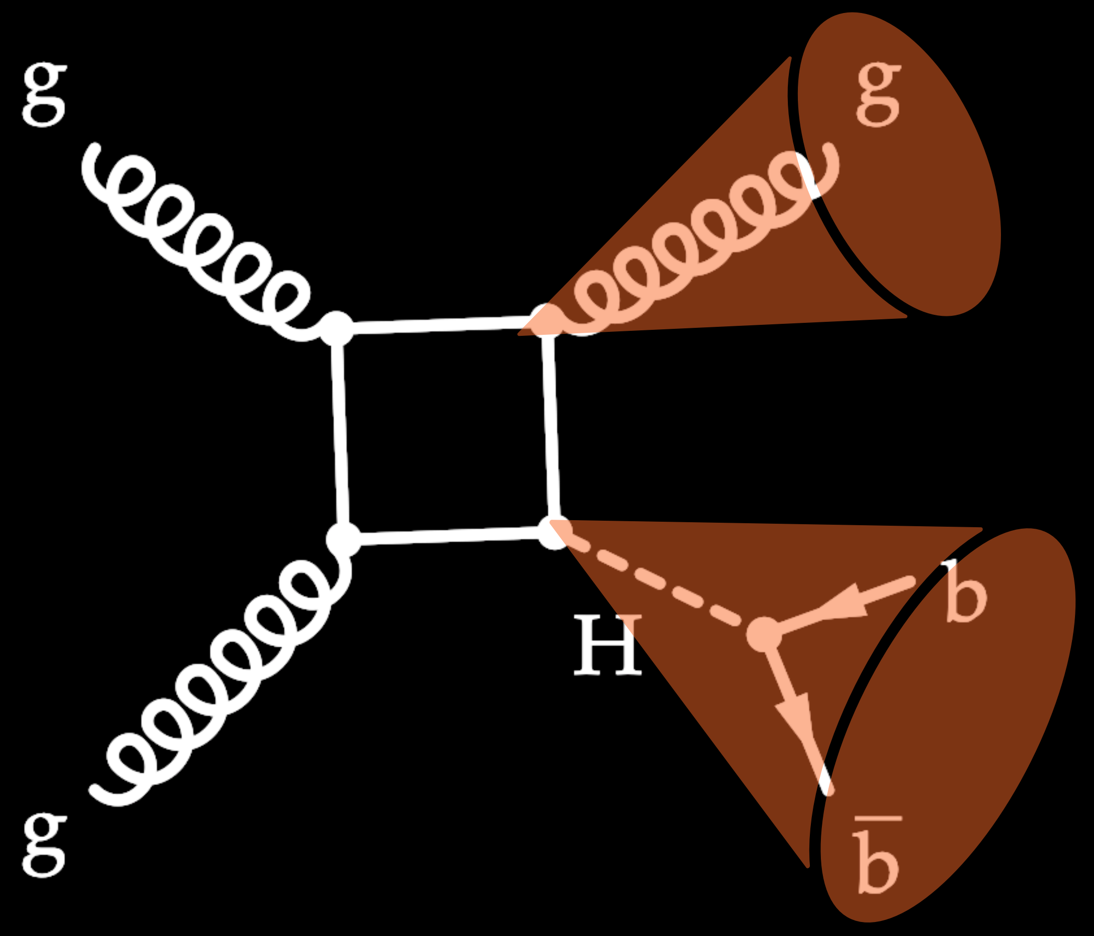

- H→γγ and H→ZZ have small rate but are very "clean"
- H→bb is large, but more difficult due to large backgrounds

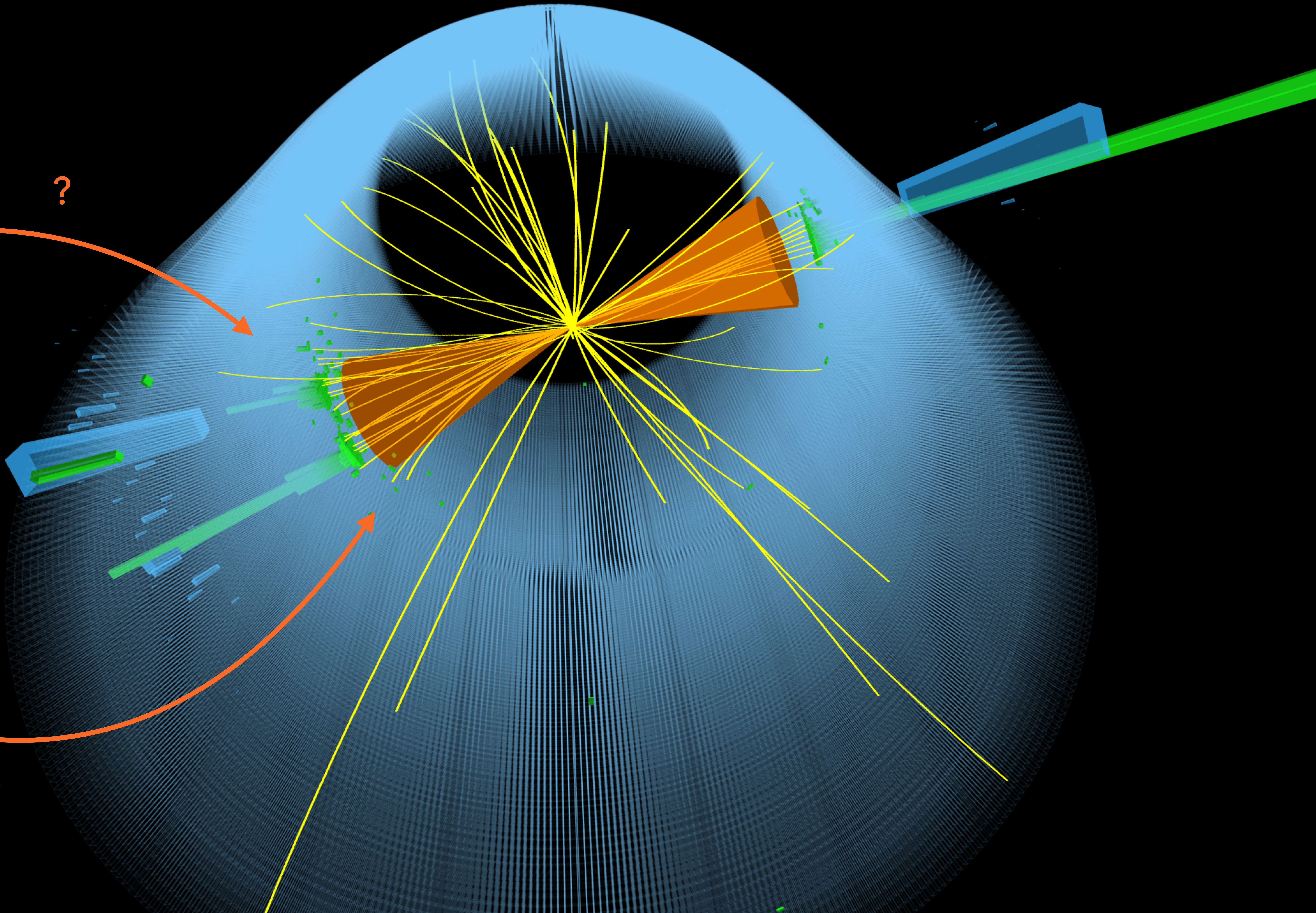
Signal:



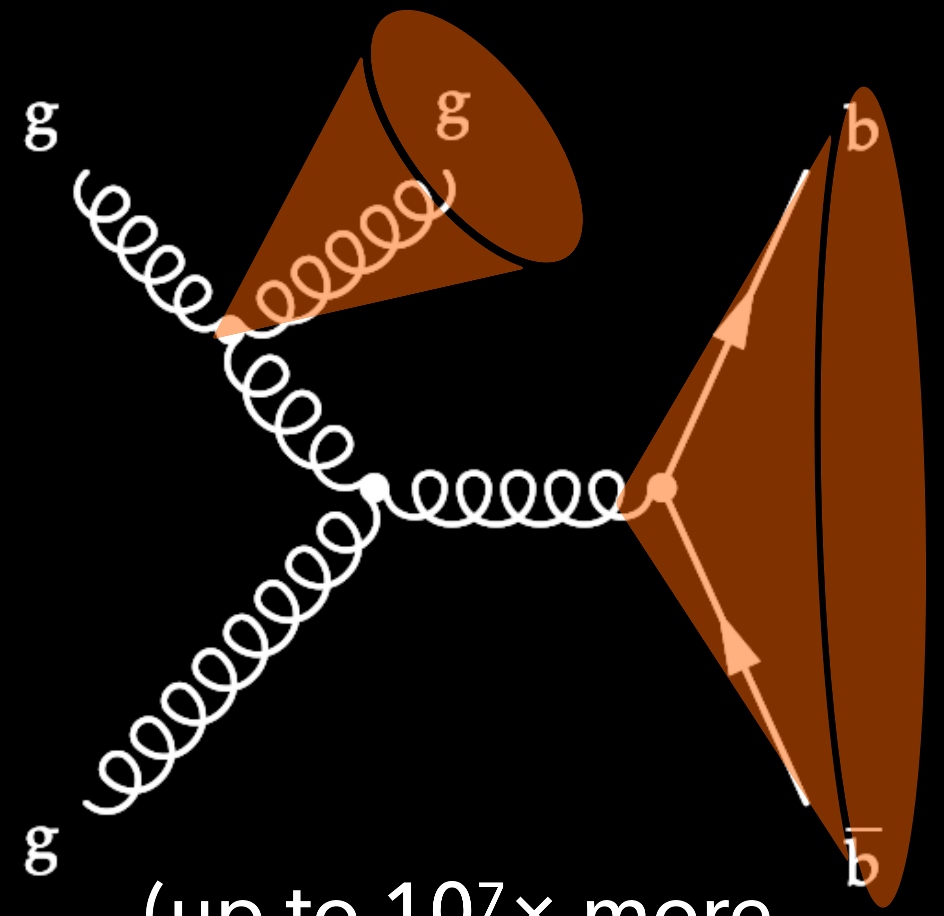



Signal:



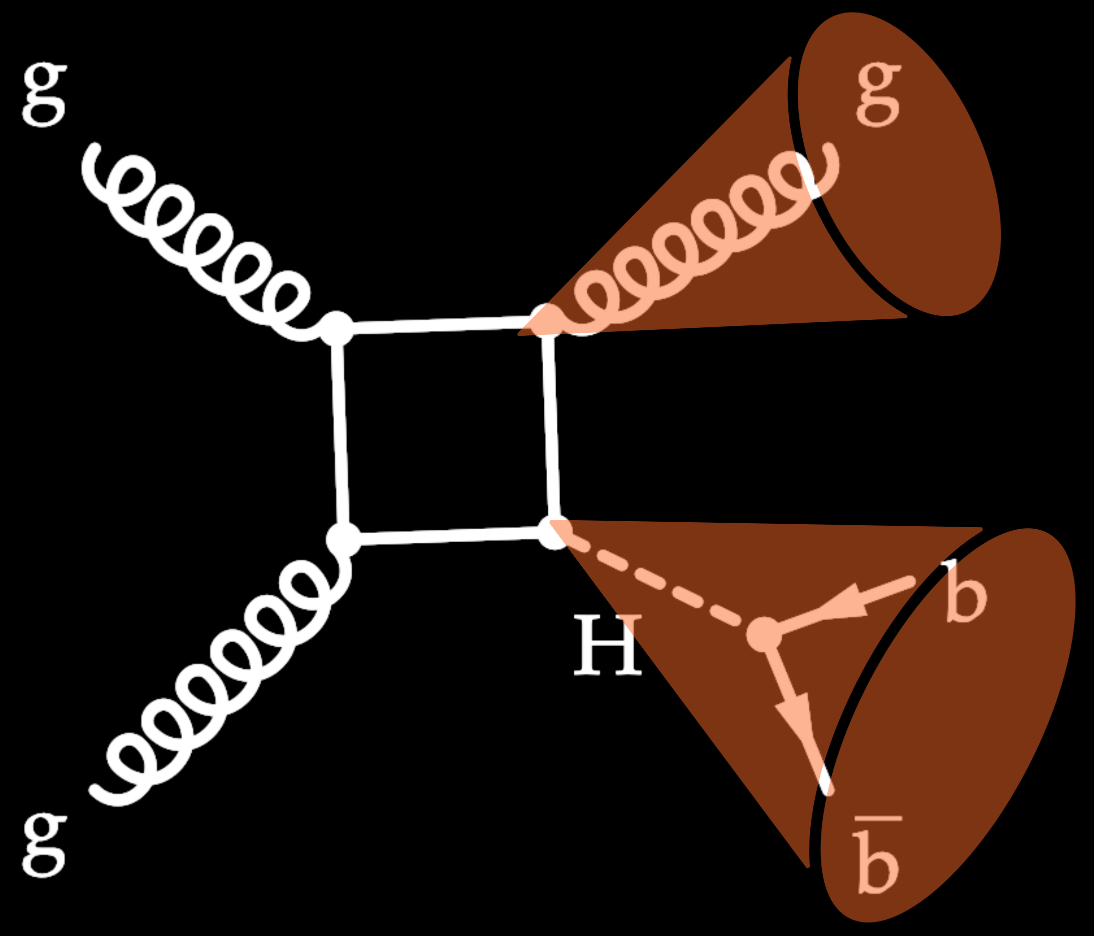

Background:



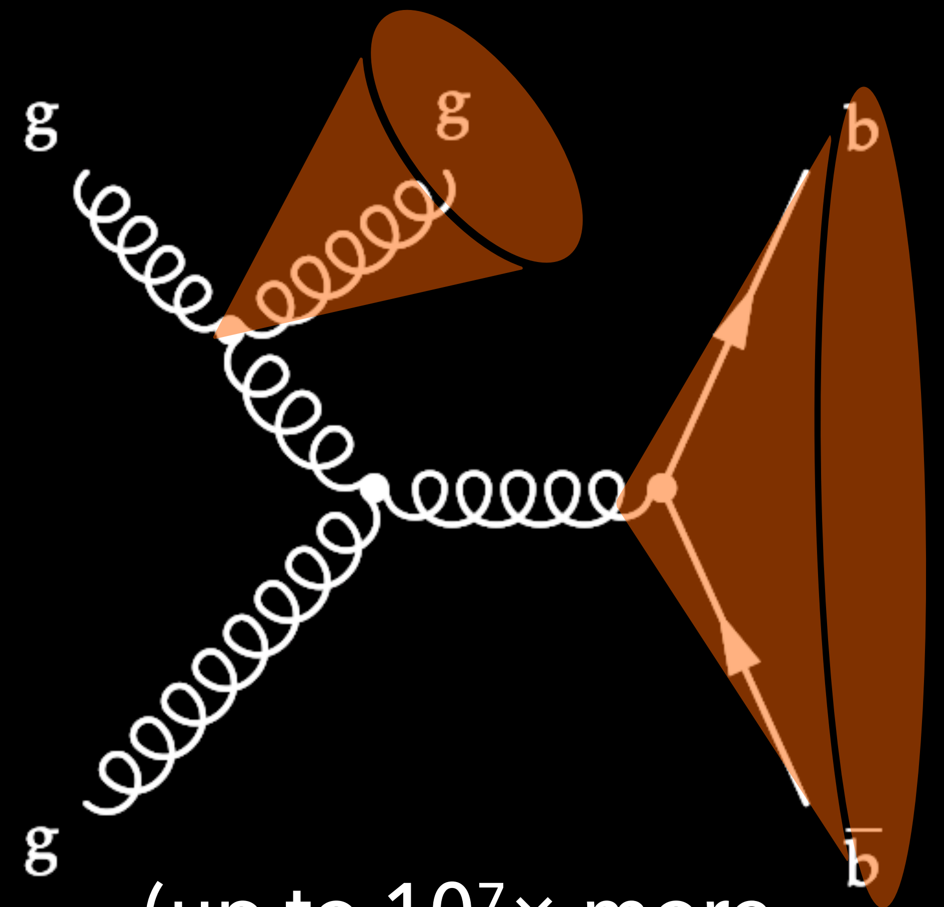

(up to $10^7\times$ more frequent)

# Can machine learning help us?

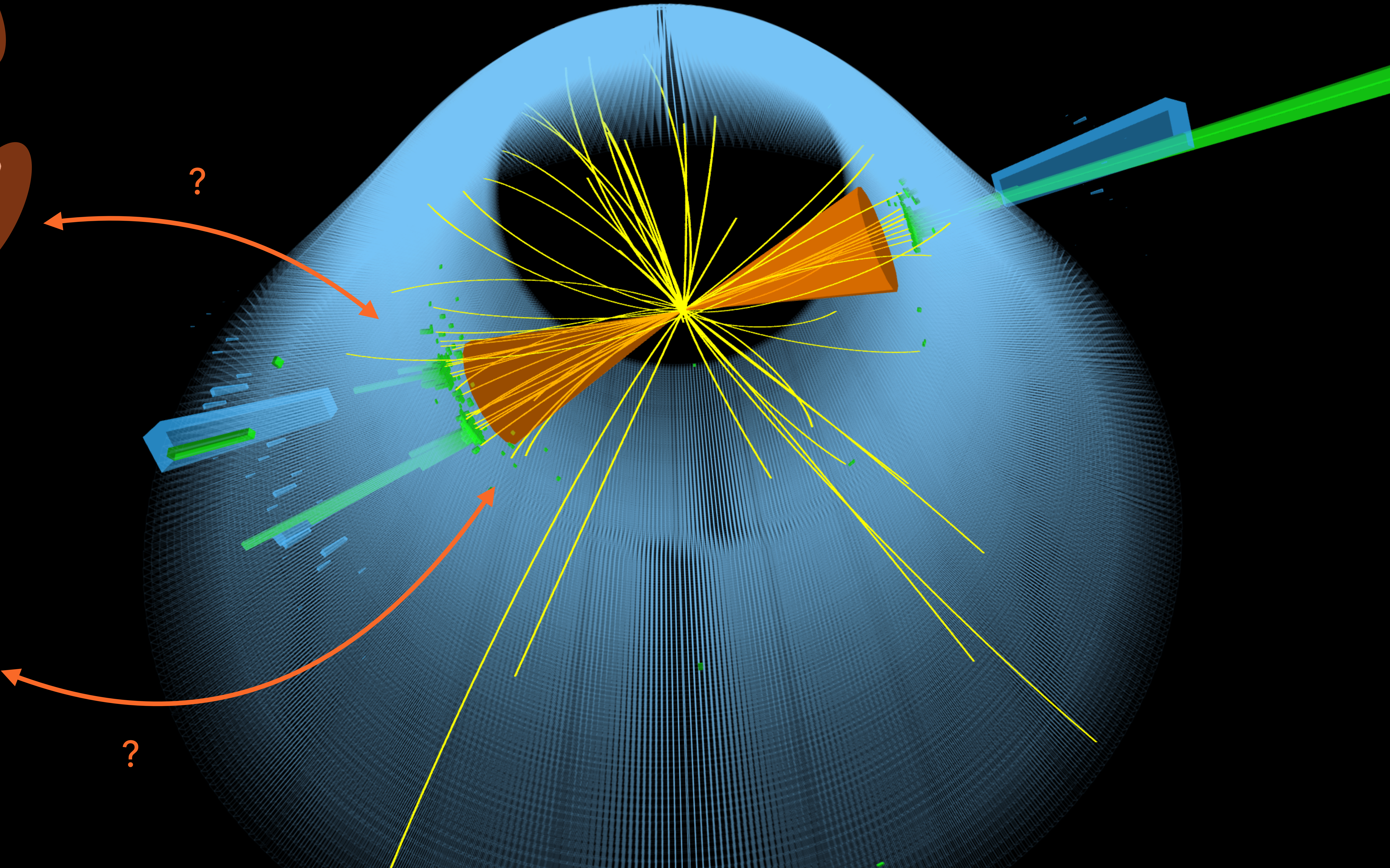
Signal:




Background:



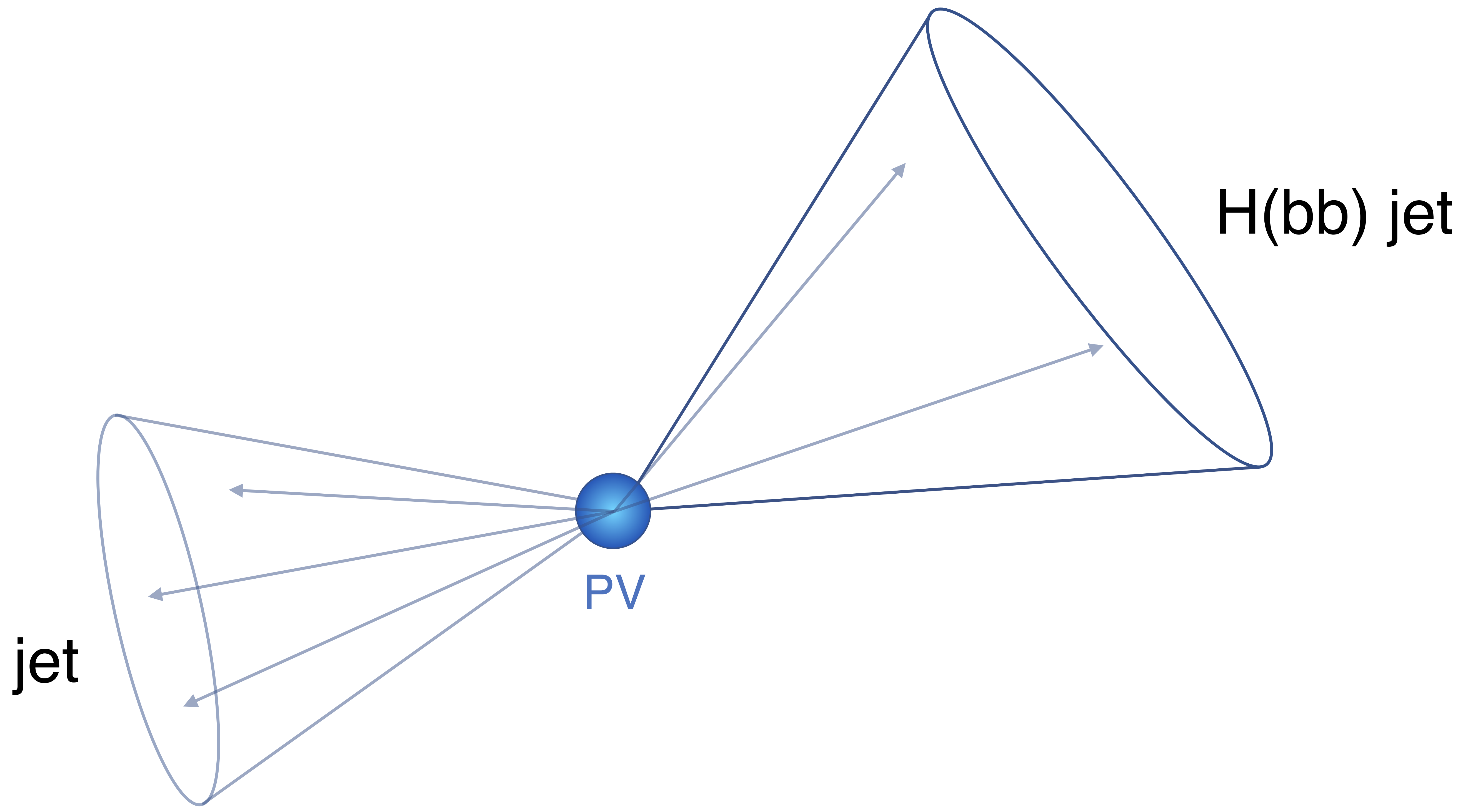

(up to $10^7\times$ more frequent)



# BASICS OF HIGGS (DOUBLE-B) TAGGING



b hadrons have long lifetimes:
travel O(mm) before decay!

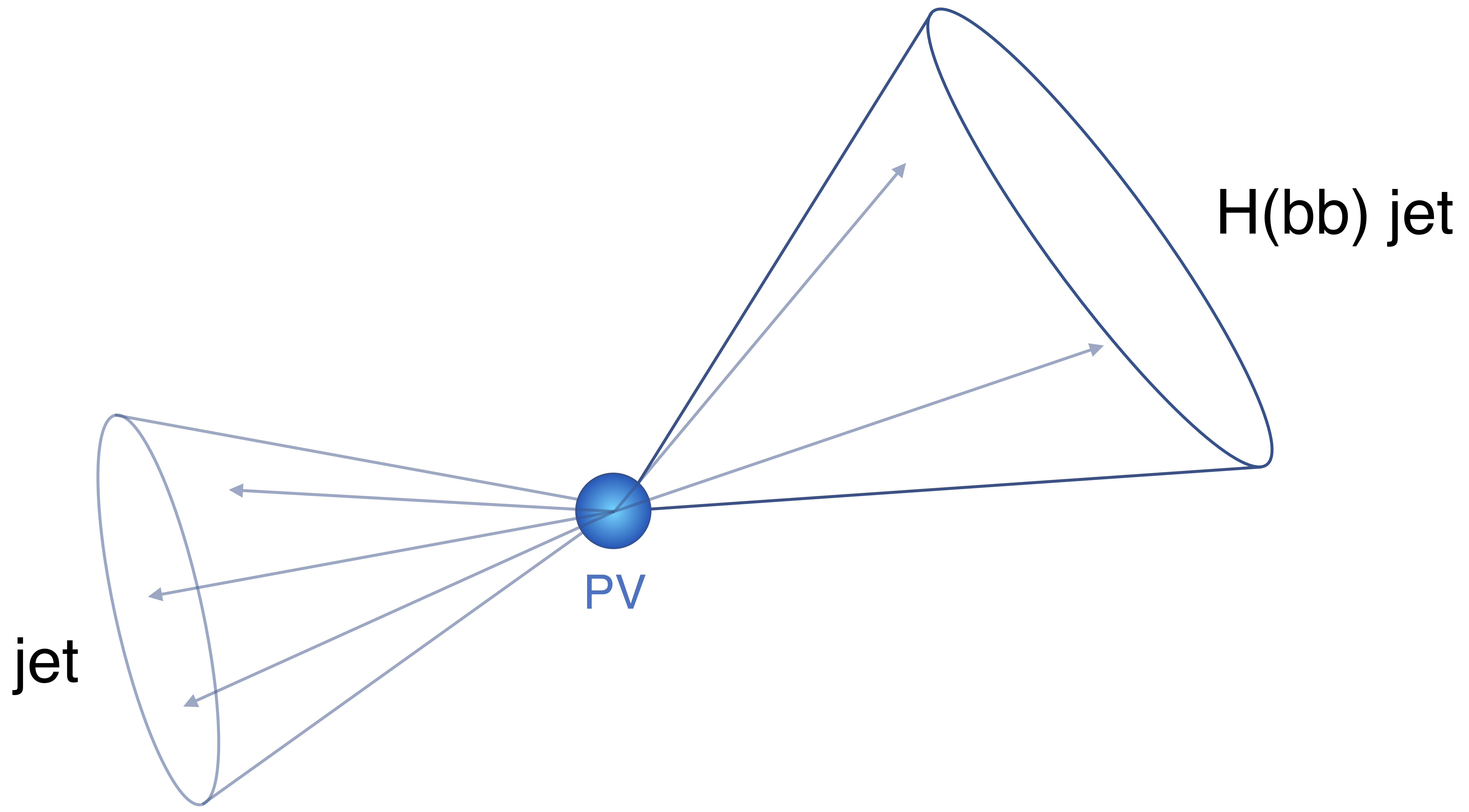


# BASICS OF HIGGS (DOUBLE-B) TAGGING



b hadrons have long lifetimes:
travel O(mm) before decay!

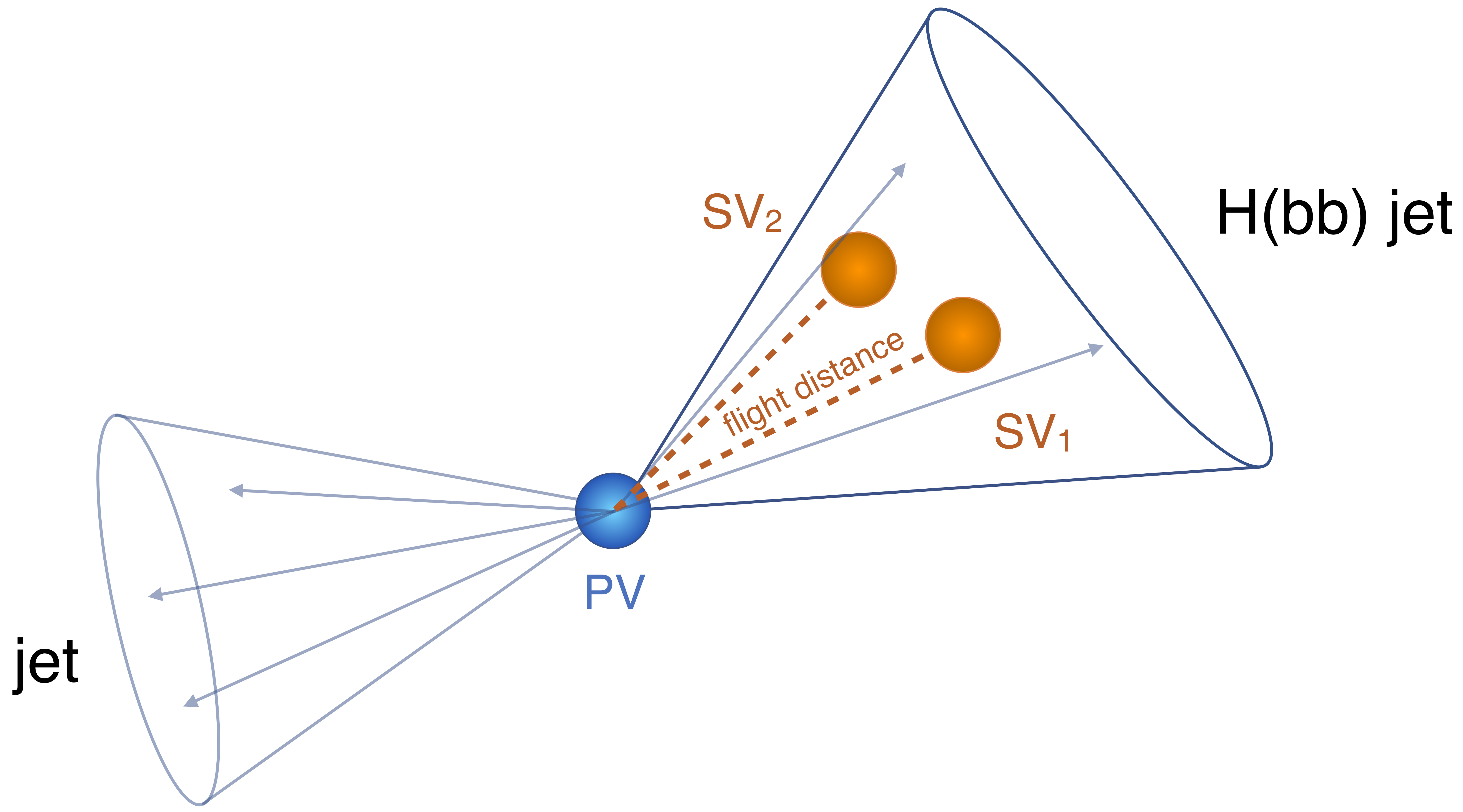
- Handles:



# BASICS OF HIGGS (DOUBLE-B) TAGGING

b hadrons have long lifetimes:
travel O(mm) before decay!

- Handles:
  - secondary vertices

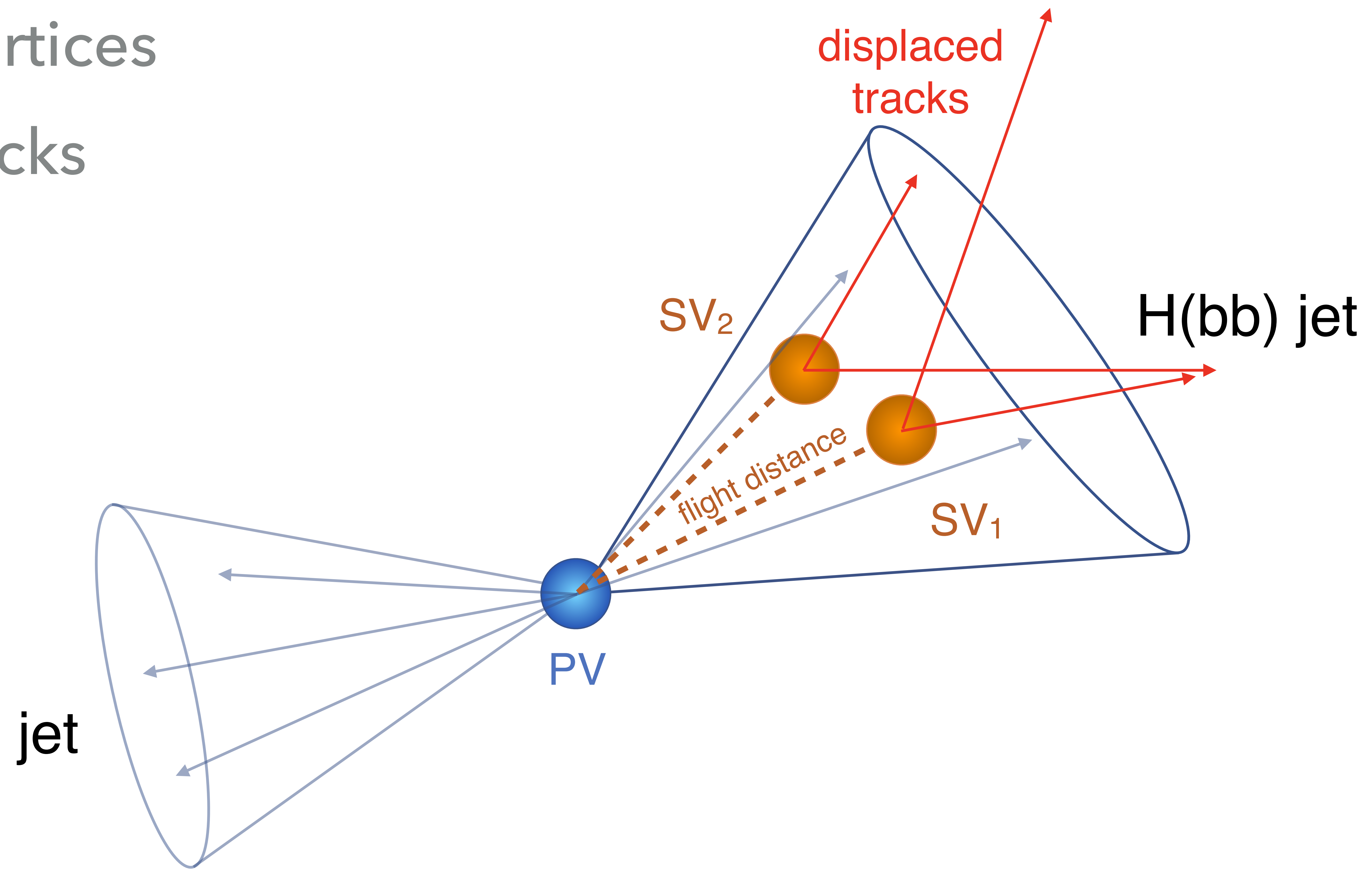


# BASICS OF HIGGS (DOUBLE-B) TAGGING



b hadrons have long lifetimes:
travel O(mm) before decay!

- Handles:
  - secondary vertices
  - displaced tracks

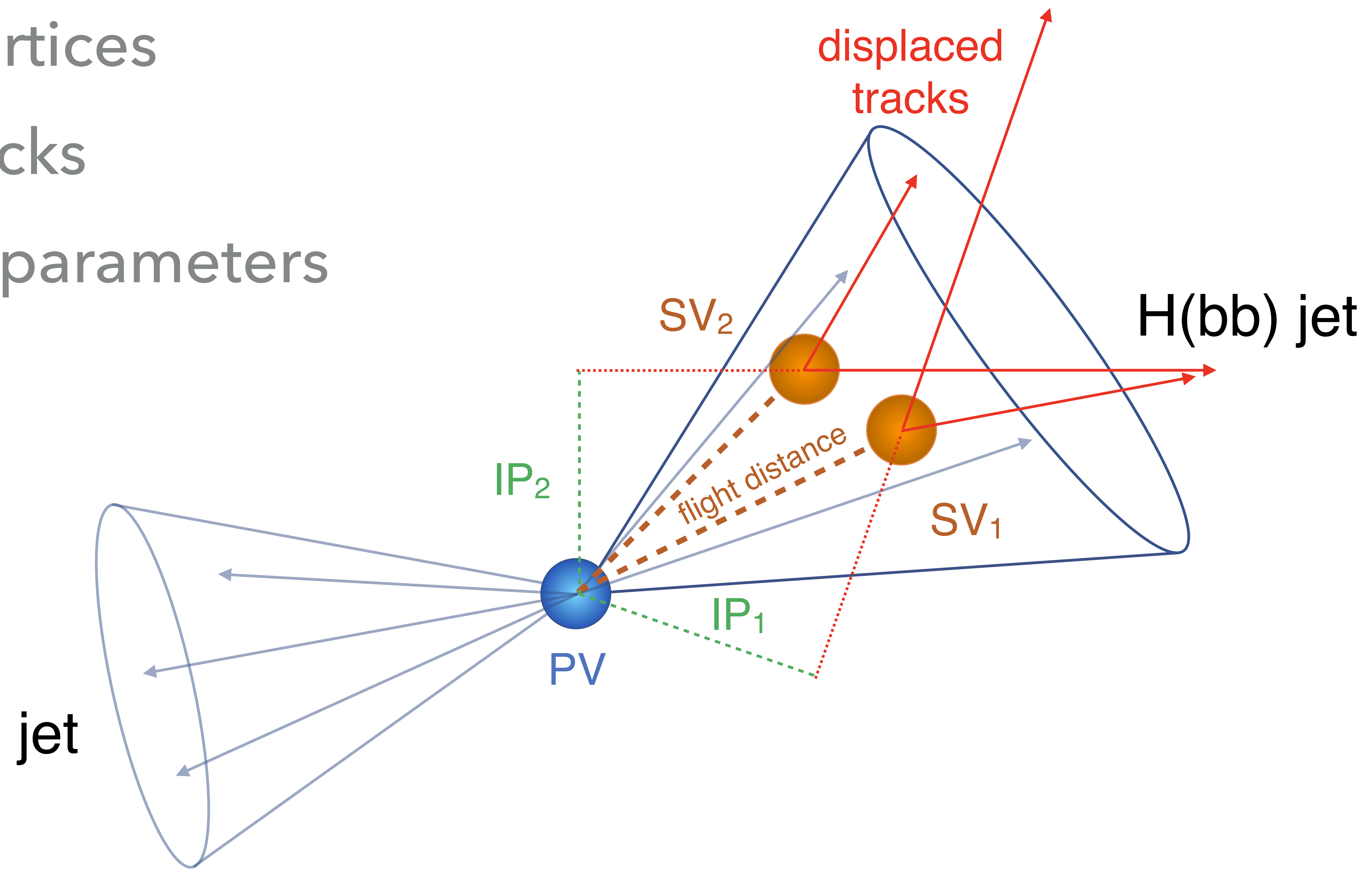


# BASICS OF HIGGS (DOUBLE-B) TAGGING



b hadrons have long lifetimes:
travel O(mm) before decay!

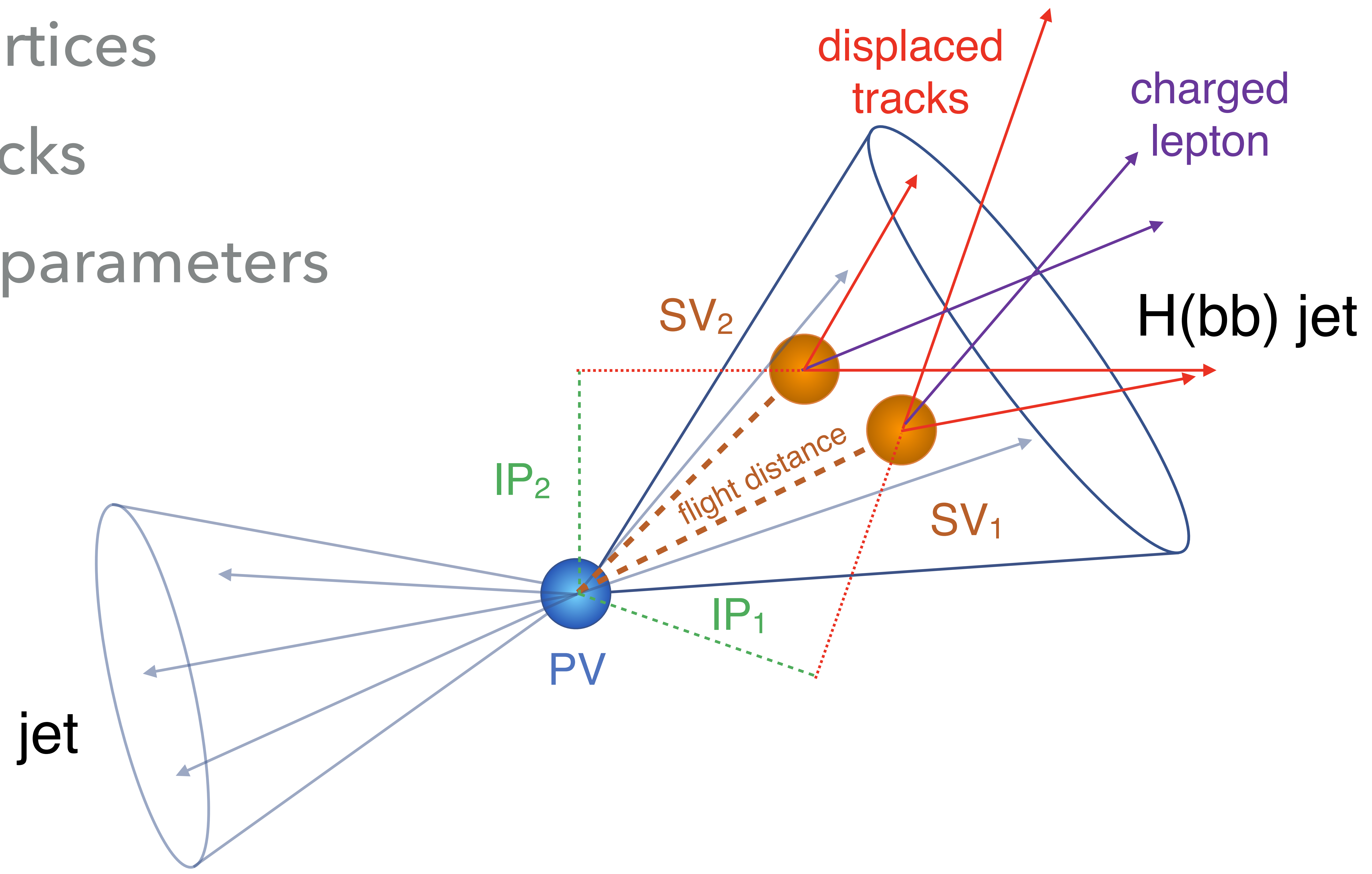
- Handles:
  - secondary vertices
  - displaced tracks
  - large impact parameters



# BASICS OF HIGGS (DOUBLE-B) TAGGING

b hadrons have long lifetimes:
travel O(mm) before decay!

- Handles:
  - secondary vertices
  - displaced tracks
  - large impact parameters
  - soft leptons

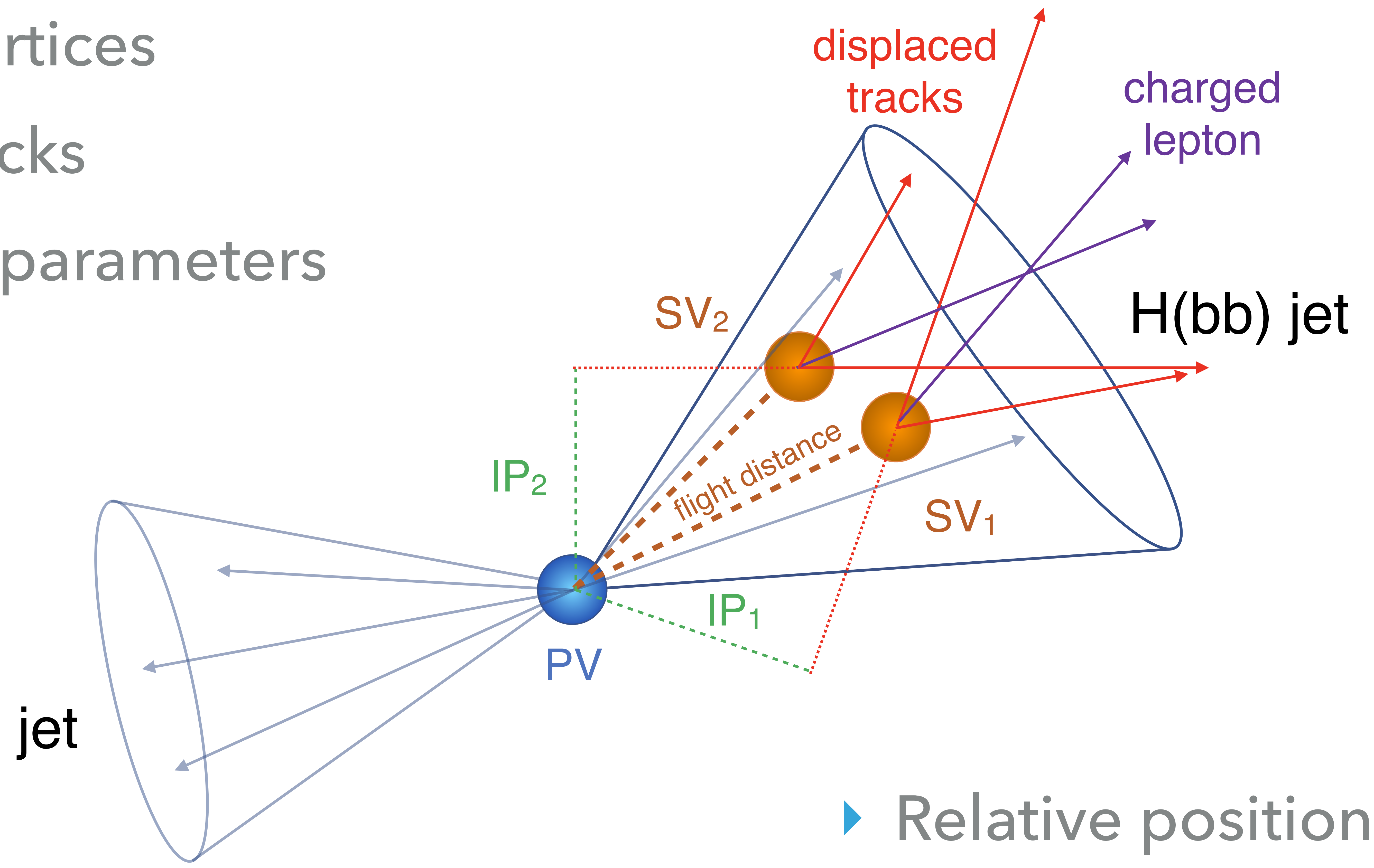


# BASICS OF HIGGS (DOUBLE-B) TAGGING



b hadrons have long lifetimes:
travel O(mm) before decay!

- Handles:
  - secondary vertices
  - displaced tracks
  - large impact parameters
  - soft leptons




- Relative positions of SVs

# HOW CAN WE REPRESENT JETS?

# HOW CAN WE REPRESENT JETS?

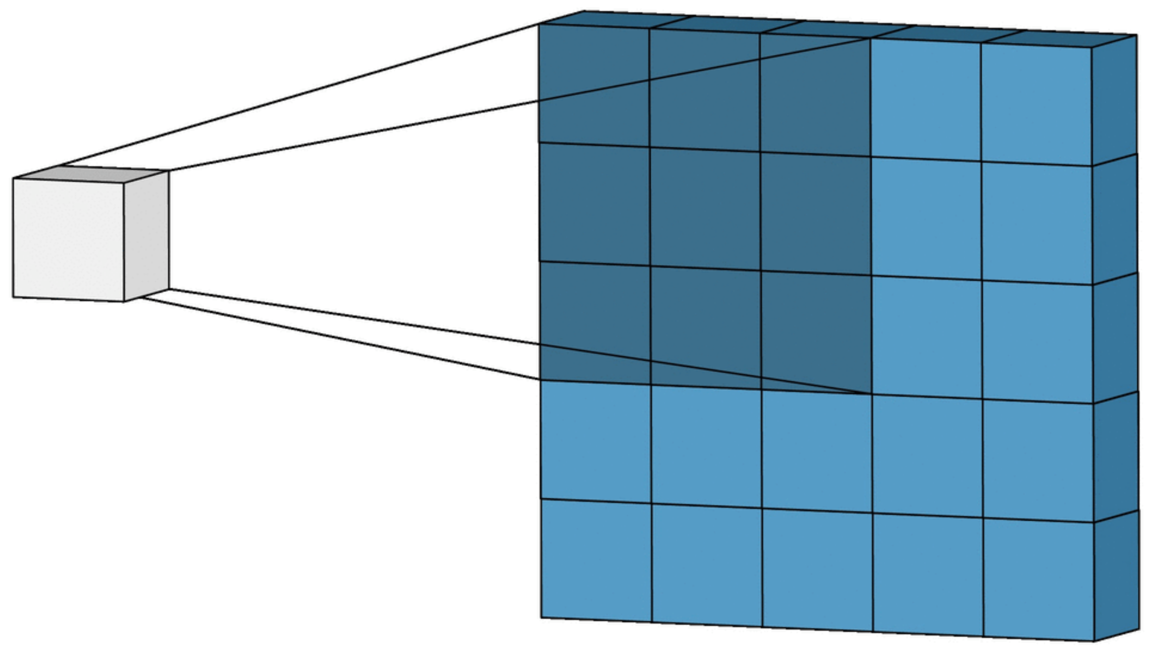


- In deep learning, tailoring algorithms to the structure (and symmetries) of the data has led to groundbreaking performance

# HOW CAN WE REPRESENT JETS?



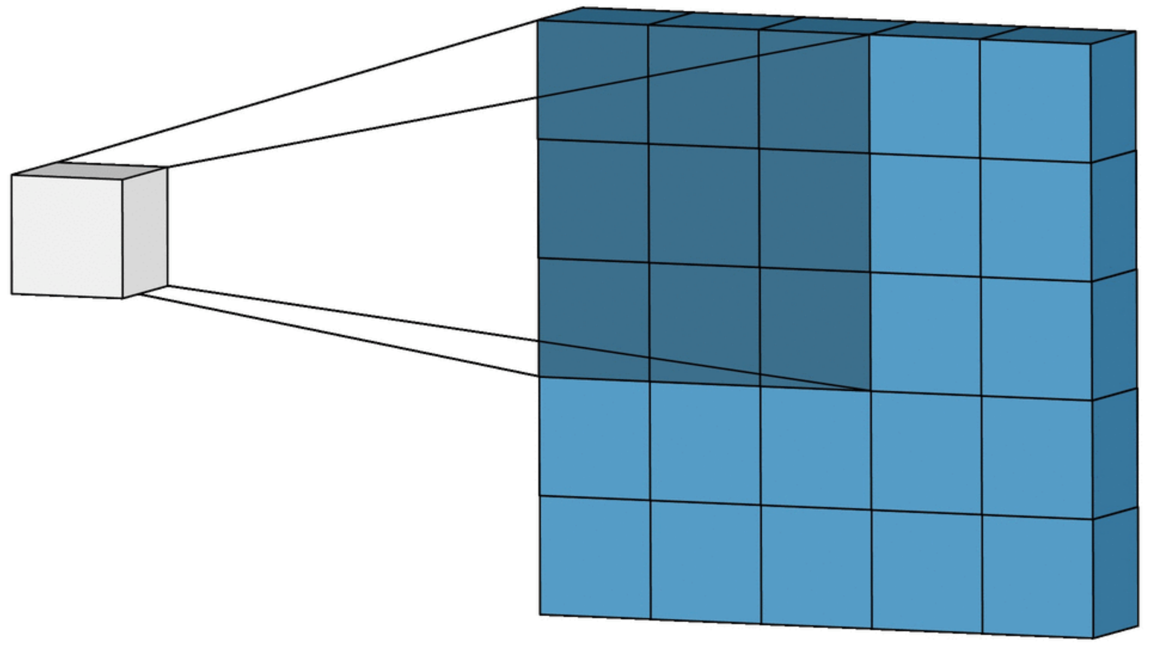
- In deep learning, tailoring algorithms to the structure (and symmetries) of the data has led to groundbreaking performance
  - CNNs for images



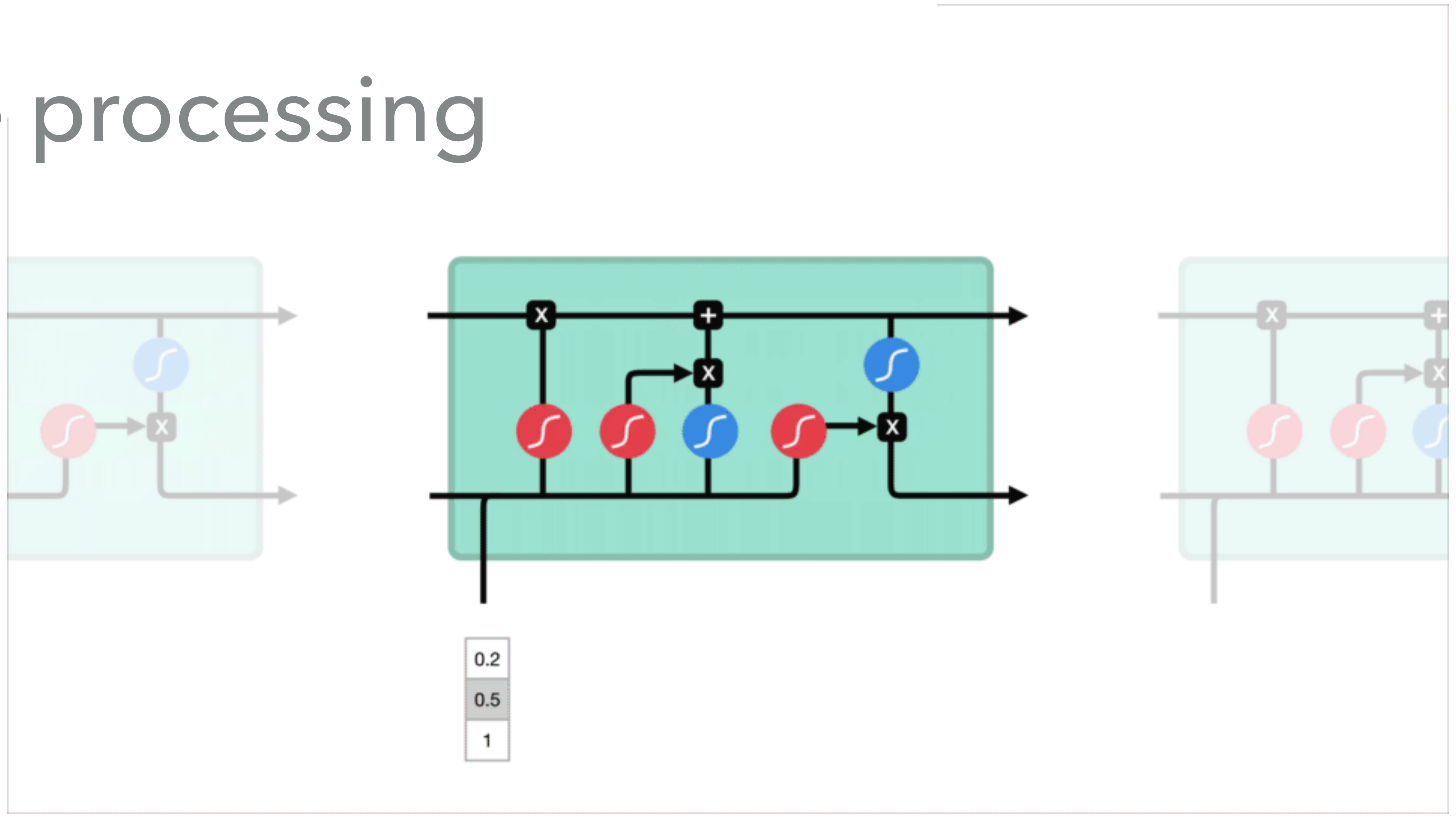
# HOW CAN WE REPRESENT JETS?



- In deep learning, tailoring algorithms to the structure (and symmetries) of the data has led to groundbreaking performance
  - CNNs for images



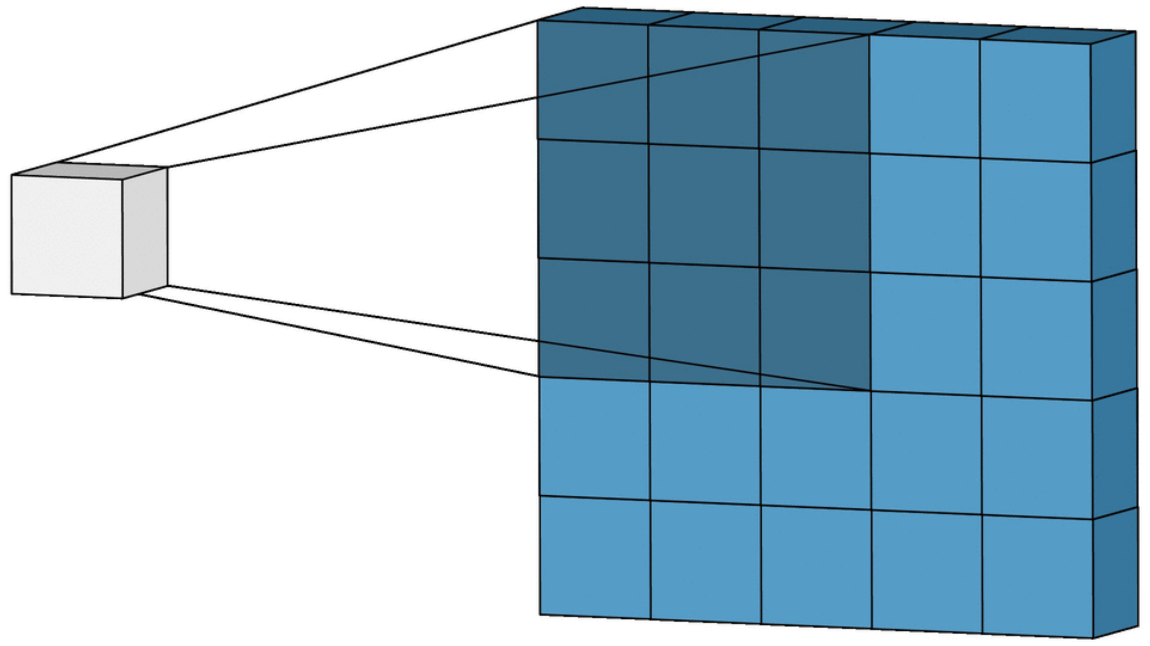
  - RNNs for language processing



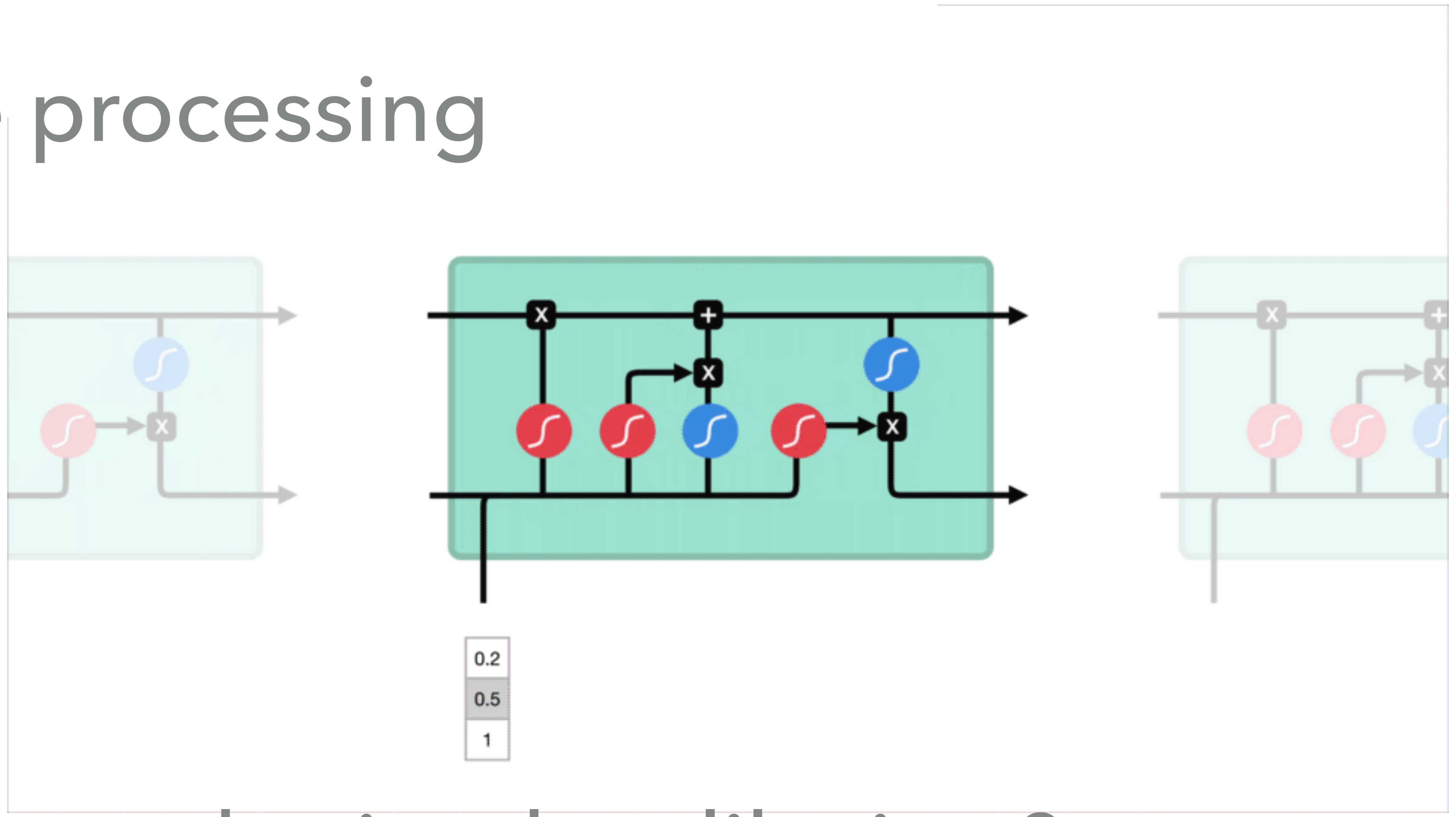
# HOW CAN WE REPRESENT JETS?



- In deep learning, tailoring algorithms to the structure (and symmetries) of the data has led to groundbreaking performance
  - CNNs for images



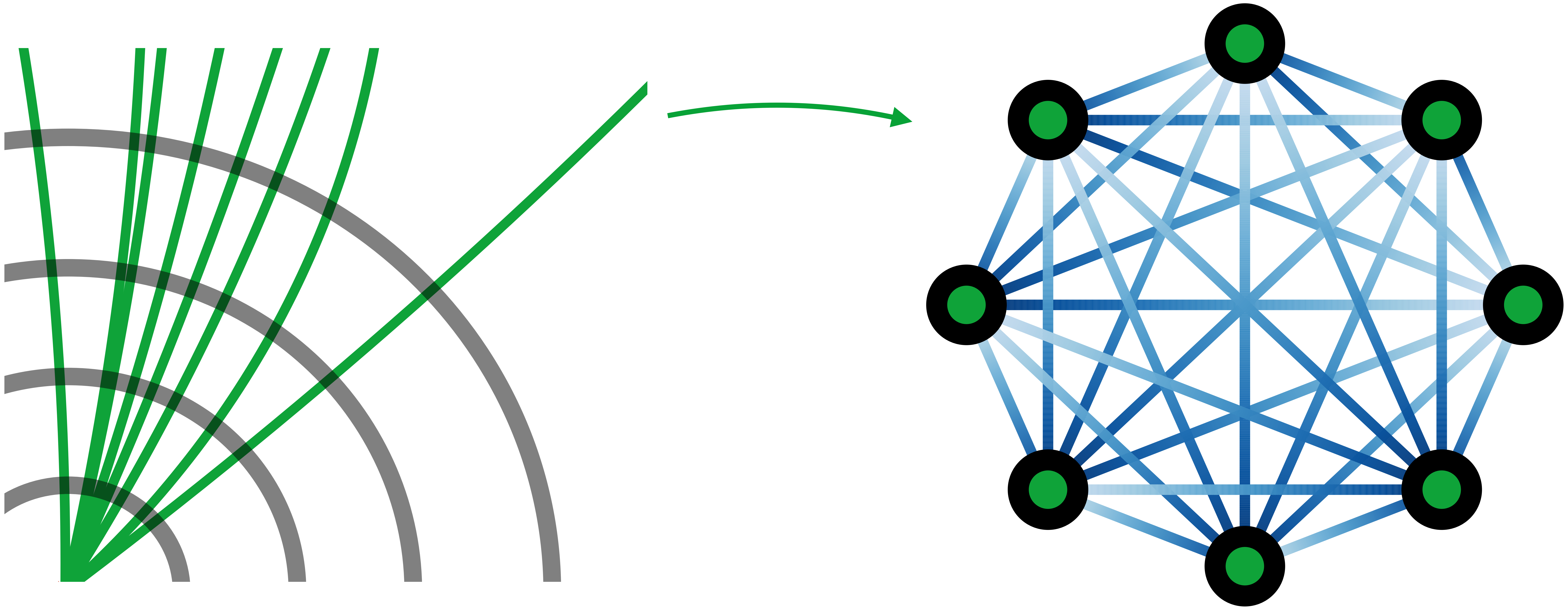
  - RNNs for language processing




- What about high energy physics data like jets?

# REPRESENTING A JET AS A GRAPH (OR “PARTICLE CLOUD”)



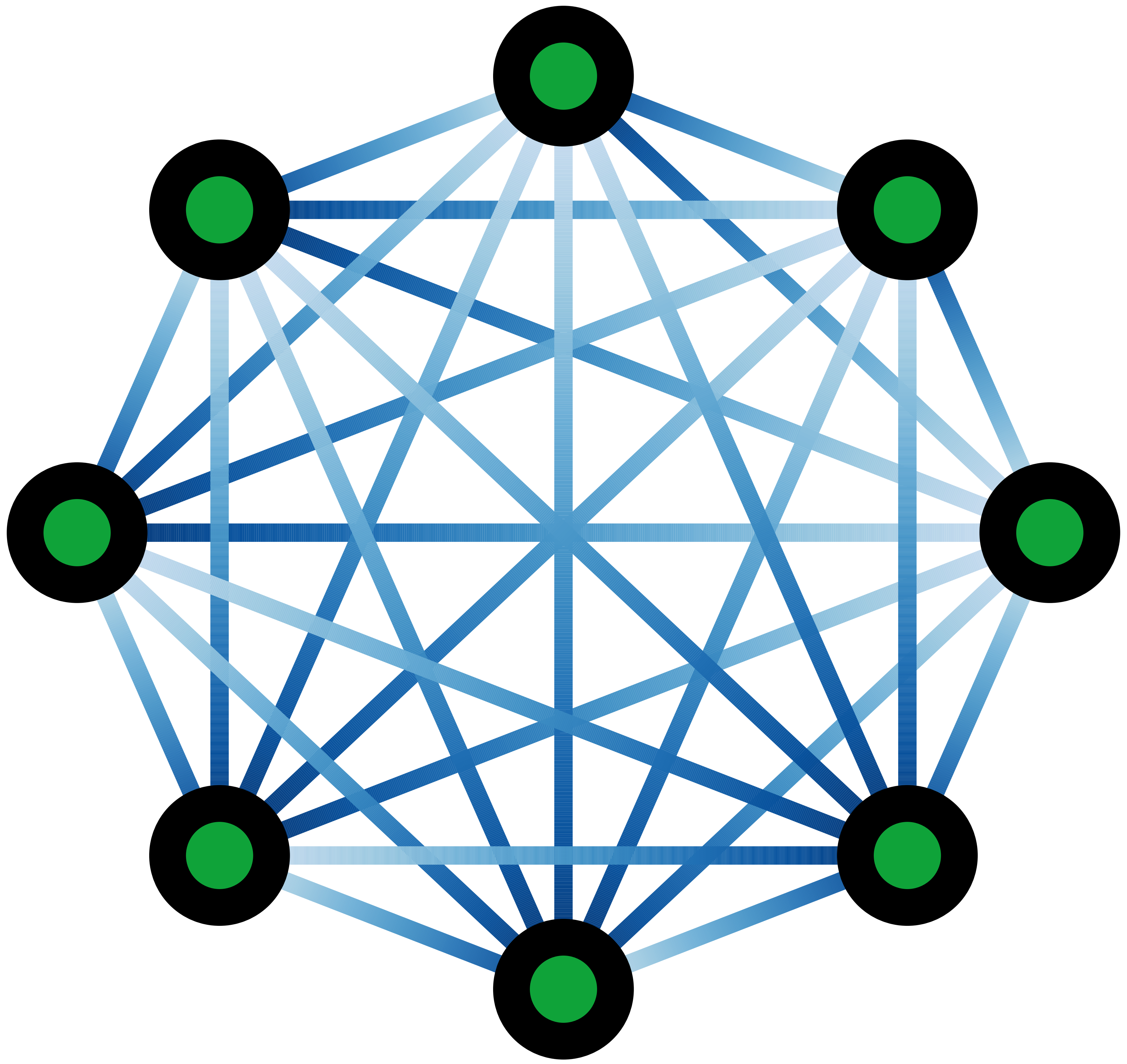


# REPRESENTING A JET AS A GRAPH (OR "PARTICLE CLOUD")



- Node features $\mathbf{v}_i$: particle 4-momentum

$$p = [E, p_x, p_y, p_z] \equiv [p_\mathrm{T}, \eta, \phi, m]$$



# REPRESENTING A JET AS A GRAPH (OR “PARTICLE CLOUD”)

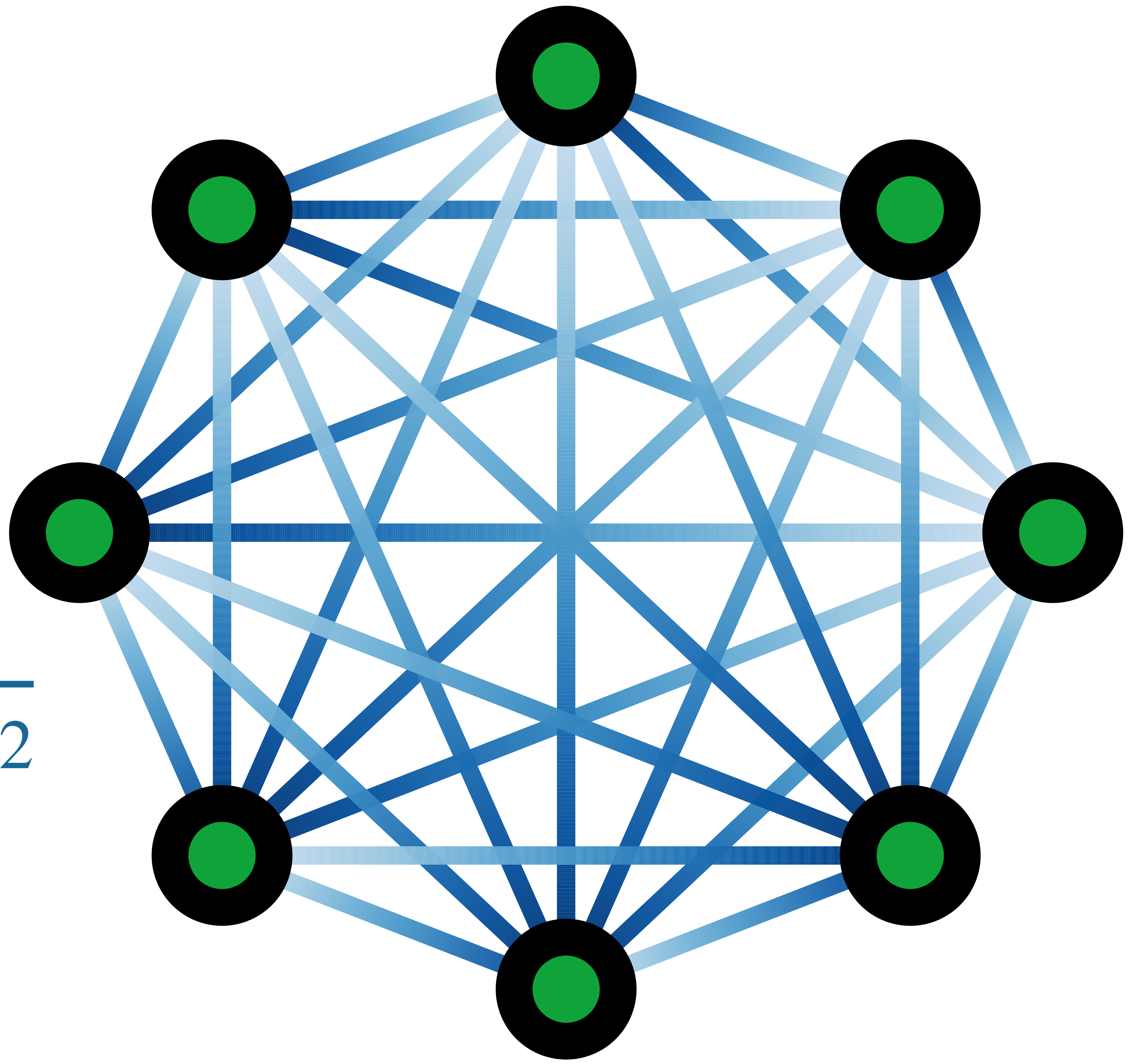


- Node features $\mathbf{v}_i$: particle 4-momentum

$$p = [E, p_x, p_y, p_z] \equiv [p_\mathrm{T}, \eta, \phi, m]$$

- Edge features $\mathbf{e}_k$: pseudoangular distance between particles

$$\Delta R = \sqrt{\Delta\eta^2 + \Delta\phi^2}$$

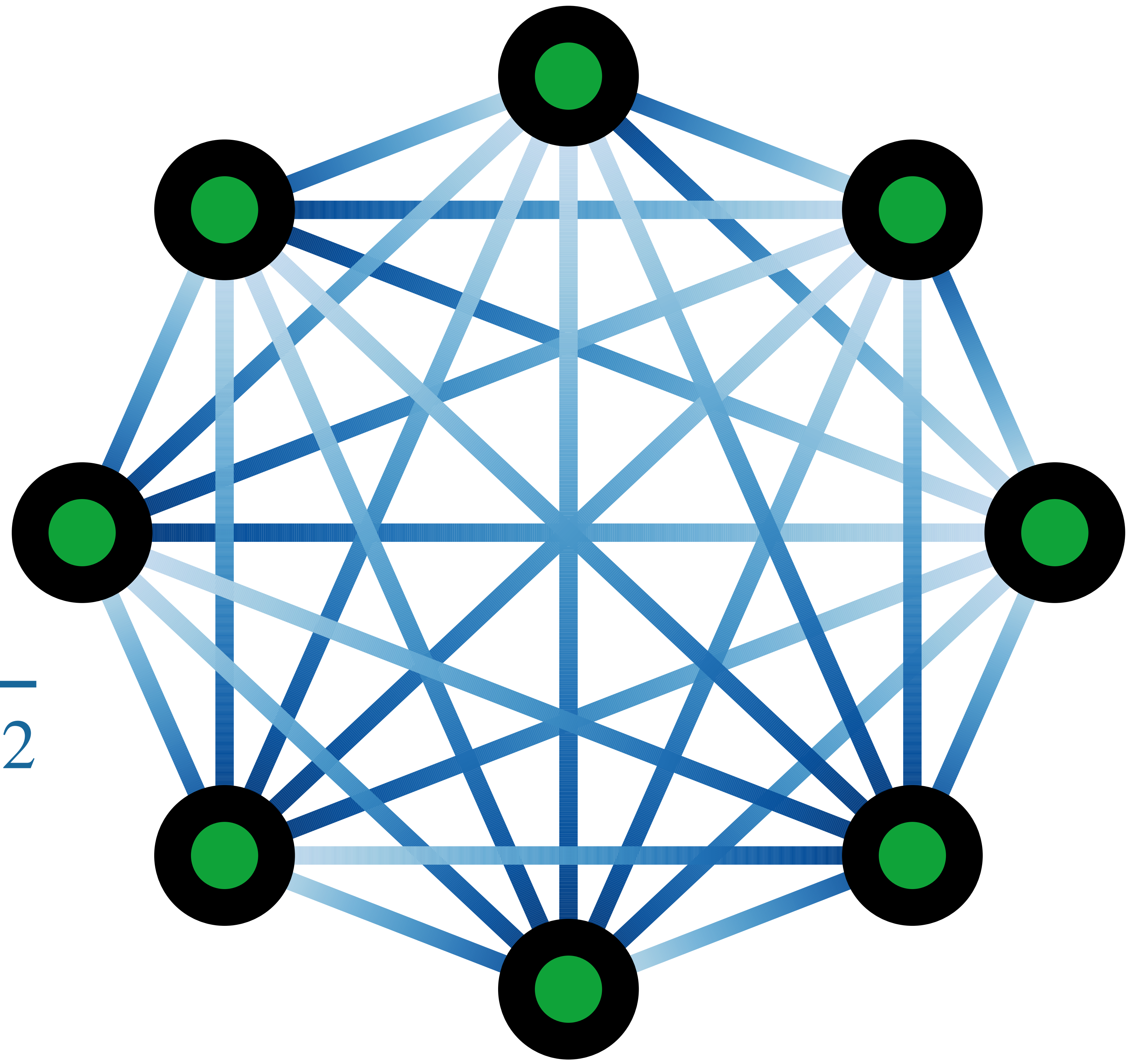


# REPRESENTING A JET AS A GRAPH (OR "PARTICLE CLOUD")



- Node features $\mathbf{v}_i$: particle 4-momentum

$$p = [E, p_x, p_y, p_z] \equiv [p_\mathrm{T}, \eta, \phi, m]$$



- Edge features $\mathbf{e}_k$: pseudoangular distance between particles

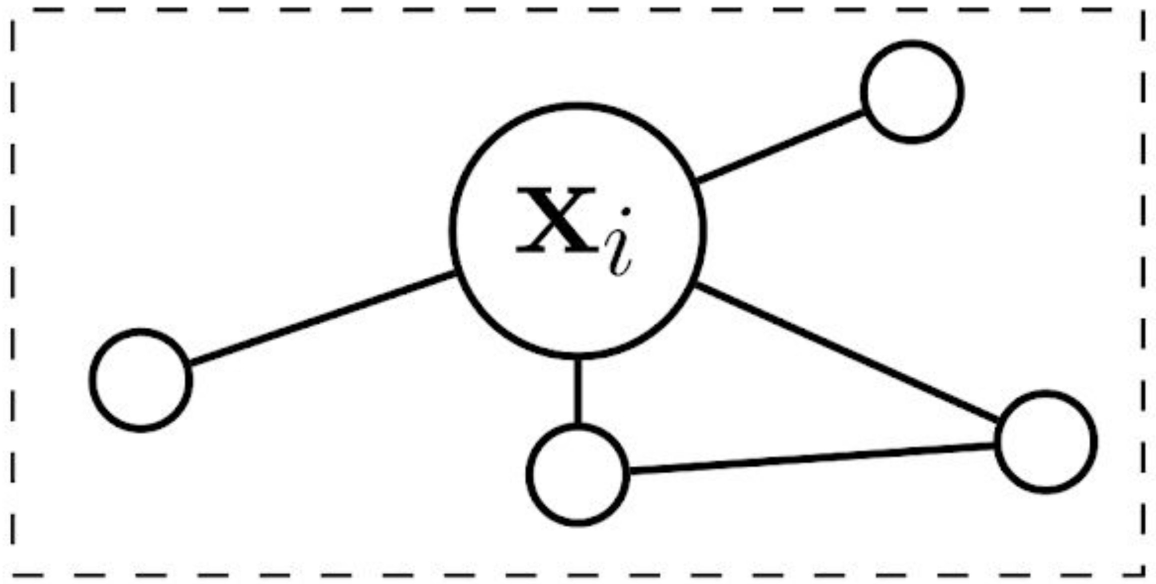
$$\Delta R = \sqrt{\Delta\eta^2 + \Delta\phi^2}$$

- Graph (global) features $\mathbf{u}$: jet mass

$$m = \sqrt{\sum_{i \in \mathrm{jet}} E_i^2 - p_{x,i}^2 - p_{y,i}^2 - p_{z,i}^2}$$

# GRAPH NEURAL NETWORKS



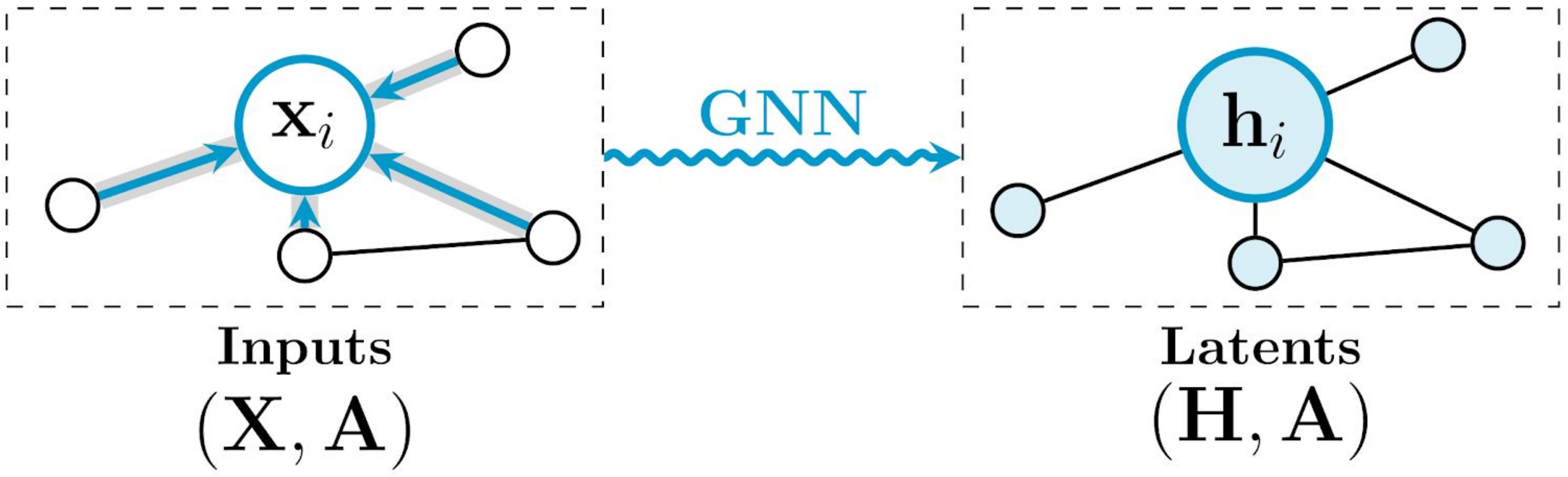



**Inputs**

$(\mathbf{X}, \mathbf{A})$

# GRAPH NEURAL NETWORKS

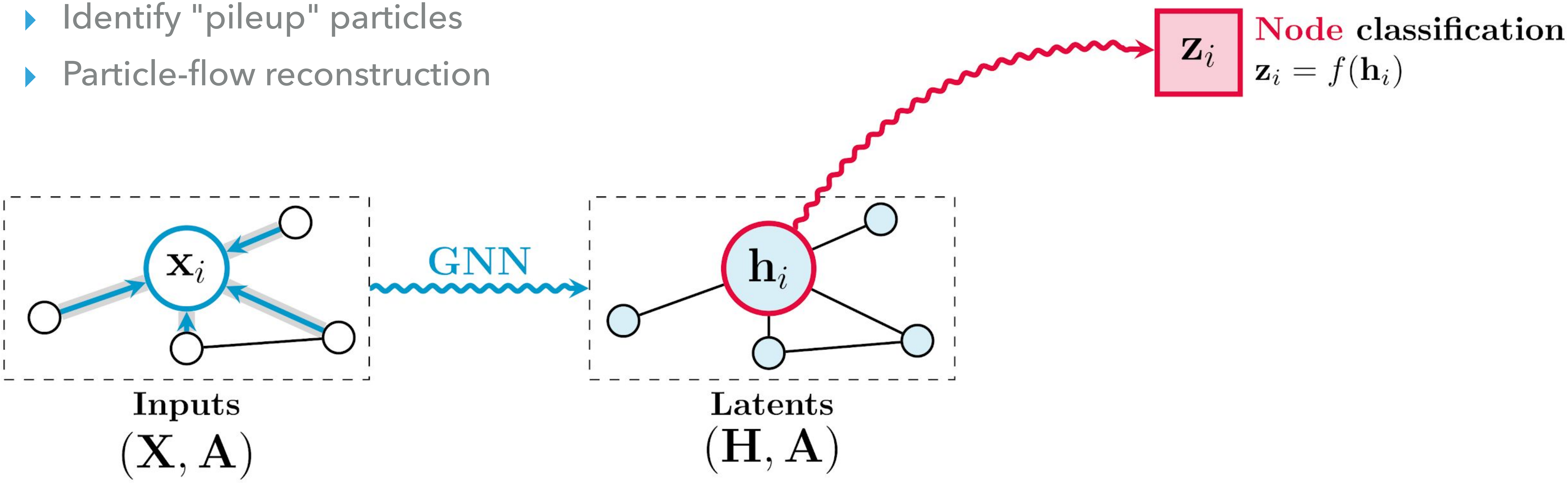




# GRAPH NEURAL NETWORKS

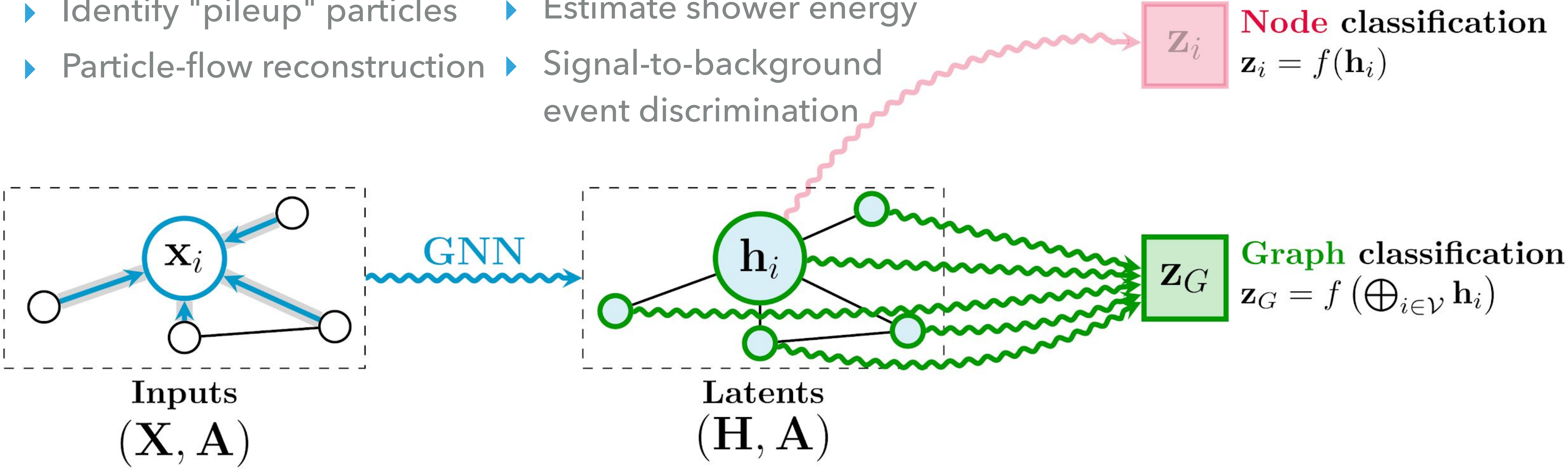


- Node-level tasks
  - Correct cluster energies
  - Identify "pileup" particles
  - Particle-flow reconstruction



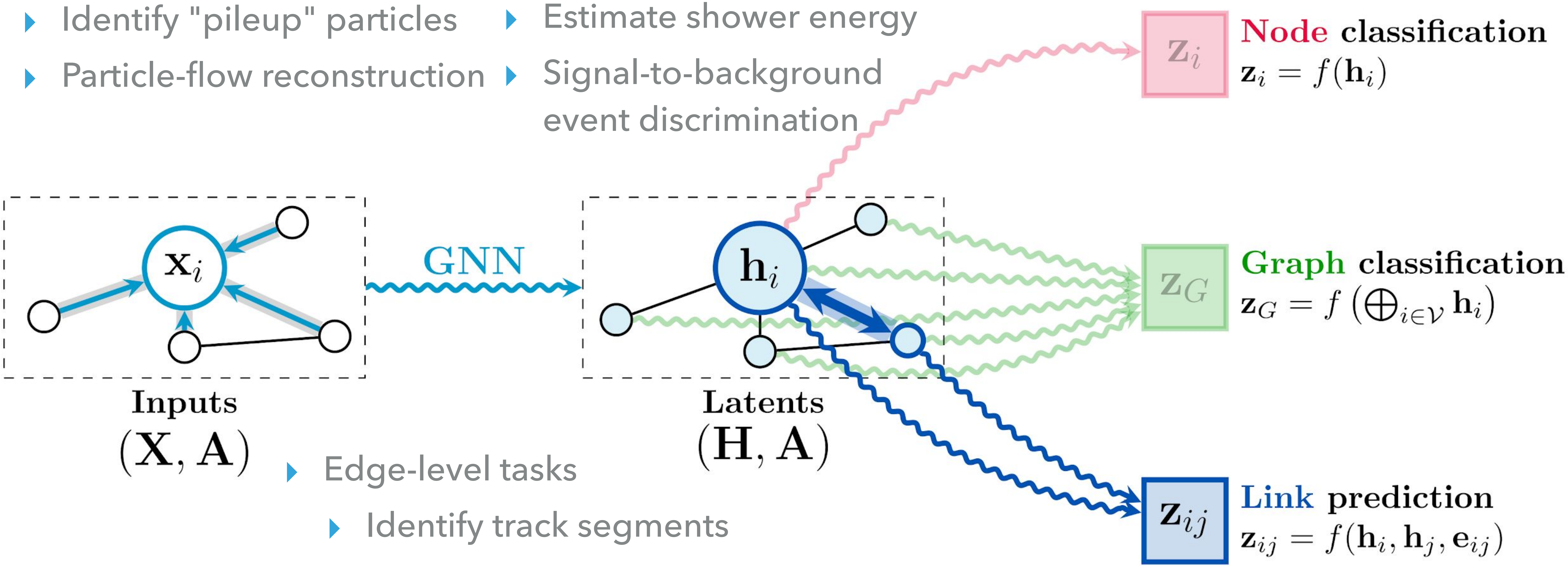
# GRAPH NEURAL NETWORKS

- Node-level tasks
  - Correct cluster energies
  - Identify "pileup" particles
  - Particle-flow reconstruction
- Graph-level tasks
  - **Jet tagging**
  - Estimate shower energy
  - Signal-to-background event discrimination



Source: https://youtu.be/uF53xsT7mjc 

- Node-level tasks
  - Correct cluster energies
  - Identify "pileup" particles
  - Particle-flow reconstruction
- Graph-level tasks
  - **Jet tagging**
  - Estimate shower energy
  - Signal-to-background event discrimination

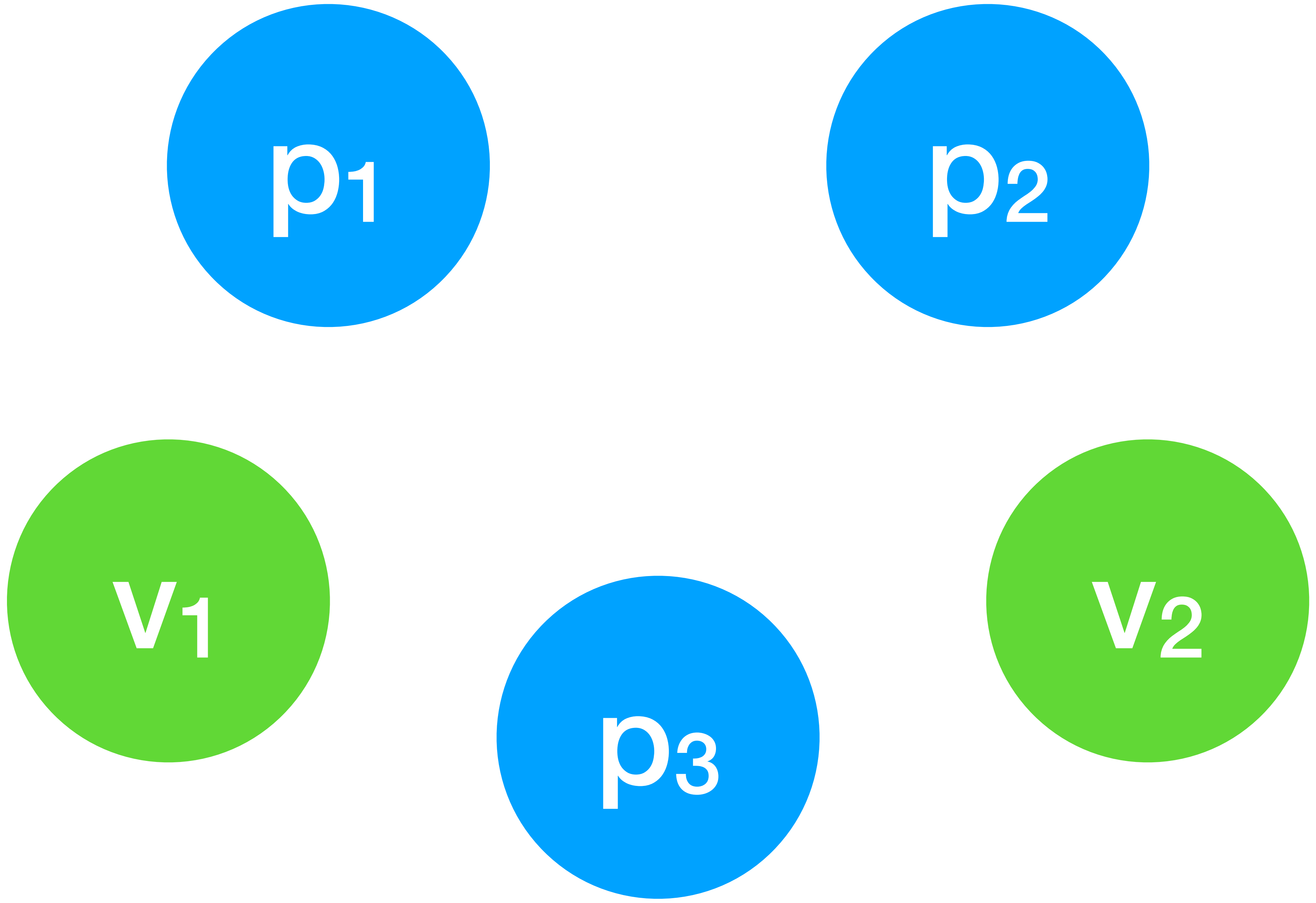



- Edge-level tasks
  - Identify track segments
  - Estimate track parameters
  - Secondary vertex reconstruction

# PARTICLES AND SECONDARY VERTICES: TWO INPUT GRAPHS



$$p_i = [p_\mathrm{T}^\mathrm{rel}, \phi^\mathrm{rel}, \eta^\mathrm{rel}, \ldots, d_\mathrm{3D}, \mathrm{cov}(p_\mathrm{T}, p_\mathrm{T}), \ldots]$$



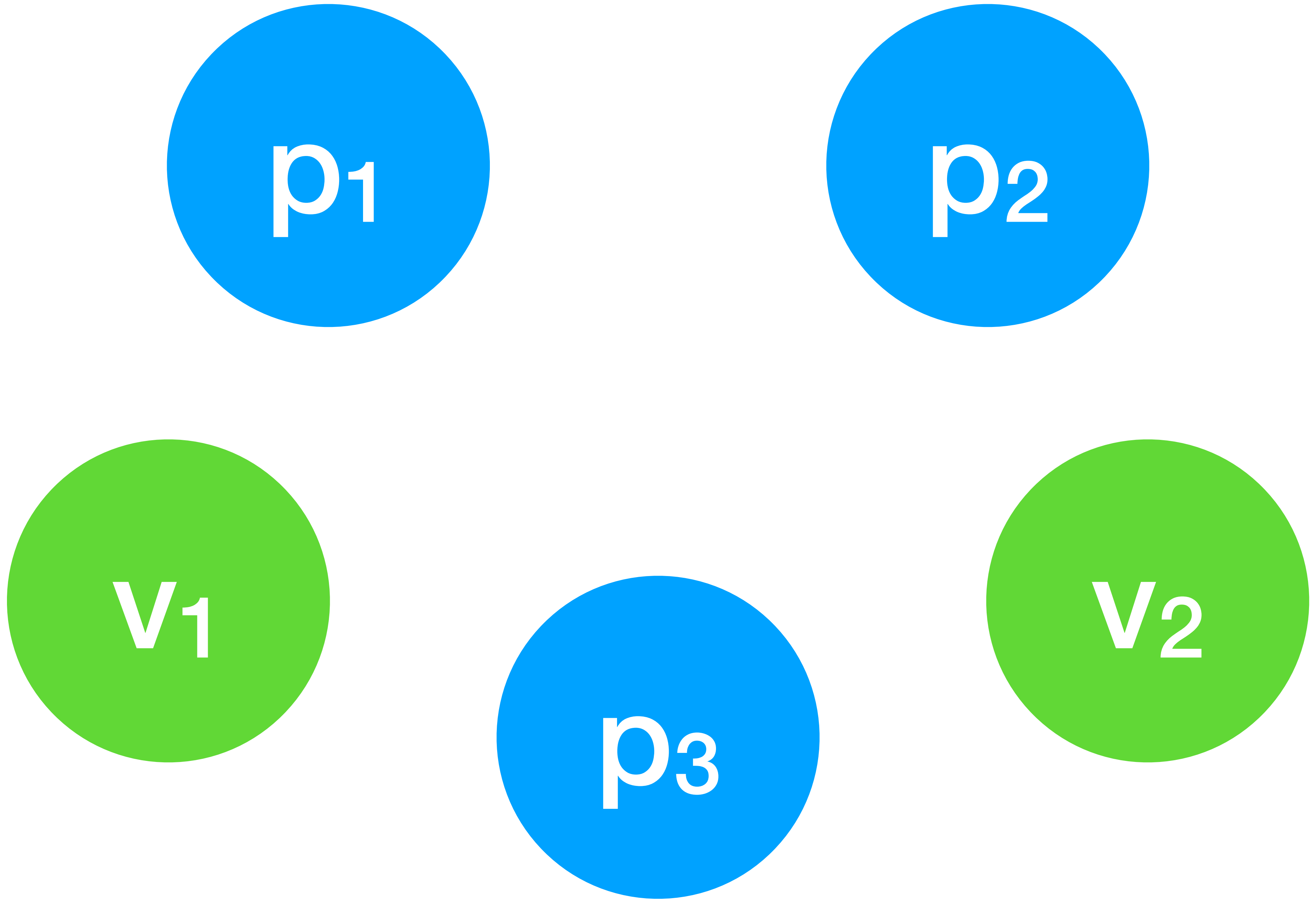

$$v_i = [p_\mathrm{T}^\mathrm{rel}, \phi^\mathrm{rel}, \eta^\mathrm{rel}, \ldots, n_\mathrm{tracks}, \cos\theta_\mathrm{PV}, \ldots]$$

# PARTICLES AND SECONDARY VERTICES: TWO INPUT GRAPHS



- Particles (i.e. tracks) and vertices are two separate inputs with different feature vectors (*heterogenous graph*)

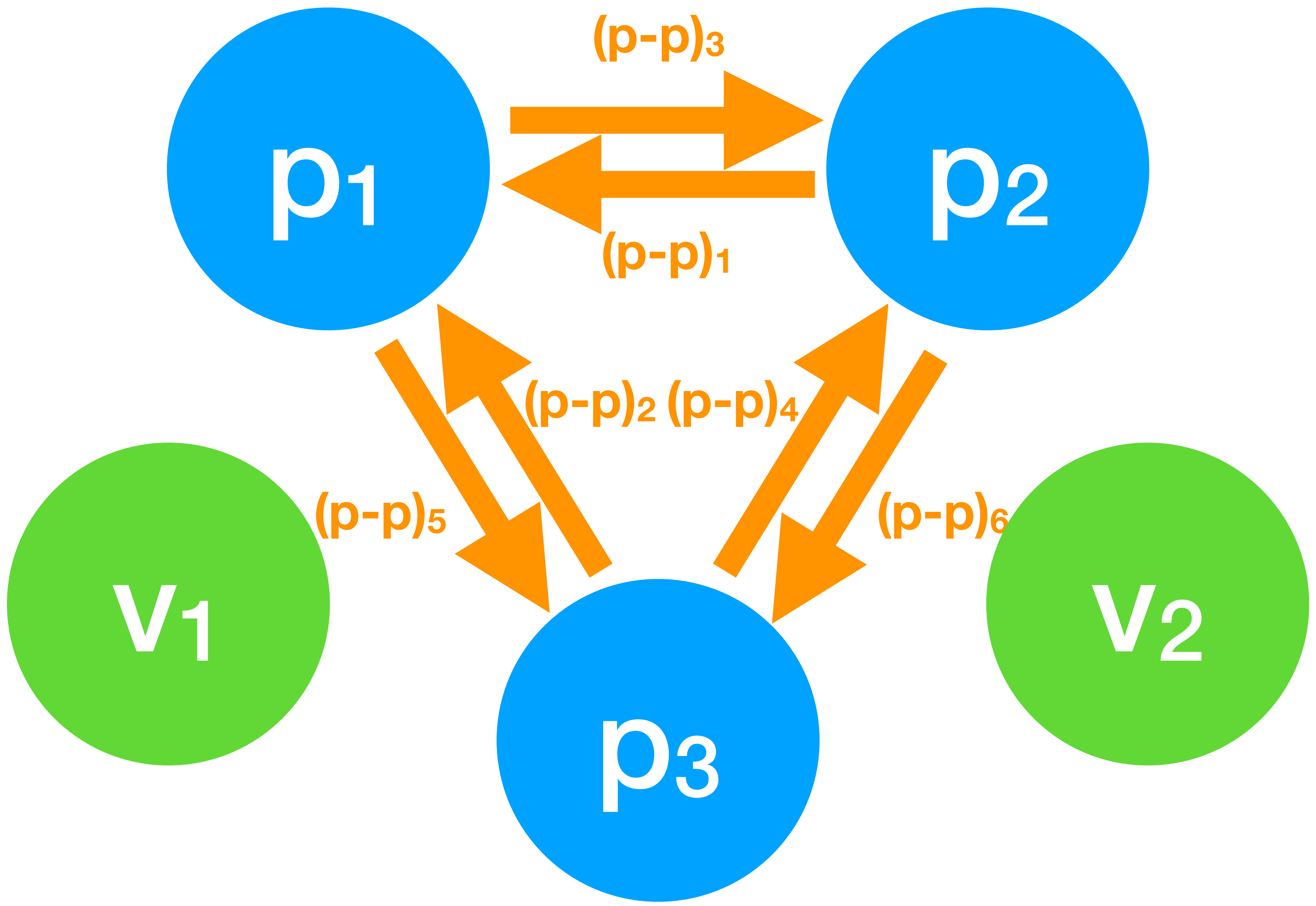
$$p_i = [p_{\mathrm{T}}^{\mathrm{rel}}, \phi^{\mathrm{rel}}, \eta^{\mathrm{rel}}, \ldots, d_{3\mathrm{D}}, \mathrm{cov}(p_{\mathrm{T}}, p_{\mathrm{T}}), \ldots]$$




$$v_i = [p_{\mathrm{T}}^{\mathrm{rel}}, \phi^{\mathrm{rel}}, \eta^{\mathrm{rel}}, \ldots, n_{\mathrm{tracks}}, \cos\theta_{\mathrm{PV}}, \ldots]$$

- Particles (i.e. tracks) and vertices are two separate inputs with different feature vectors (*heterogenous graph*)
- GNNs typically consider a *homogenous graph* (e.g. particle-particle graph)

$p_i = [p_T^{rel}, \phi^{rel}, \eta^{rel}, \ldots, d_{3D}, \mathrm{cov}(p_T, p_T), \ldots]$

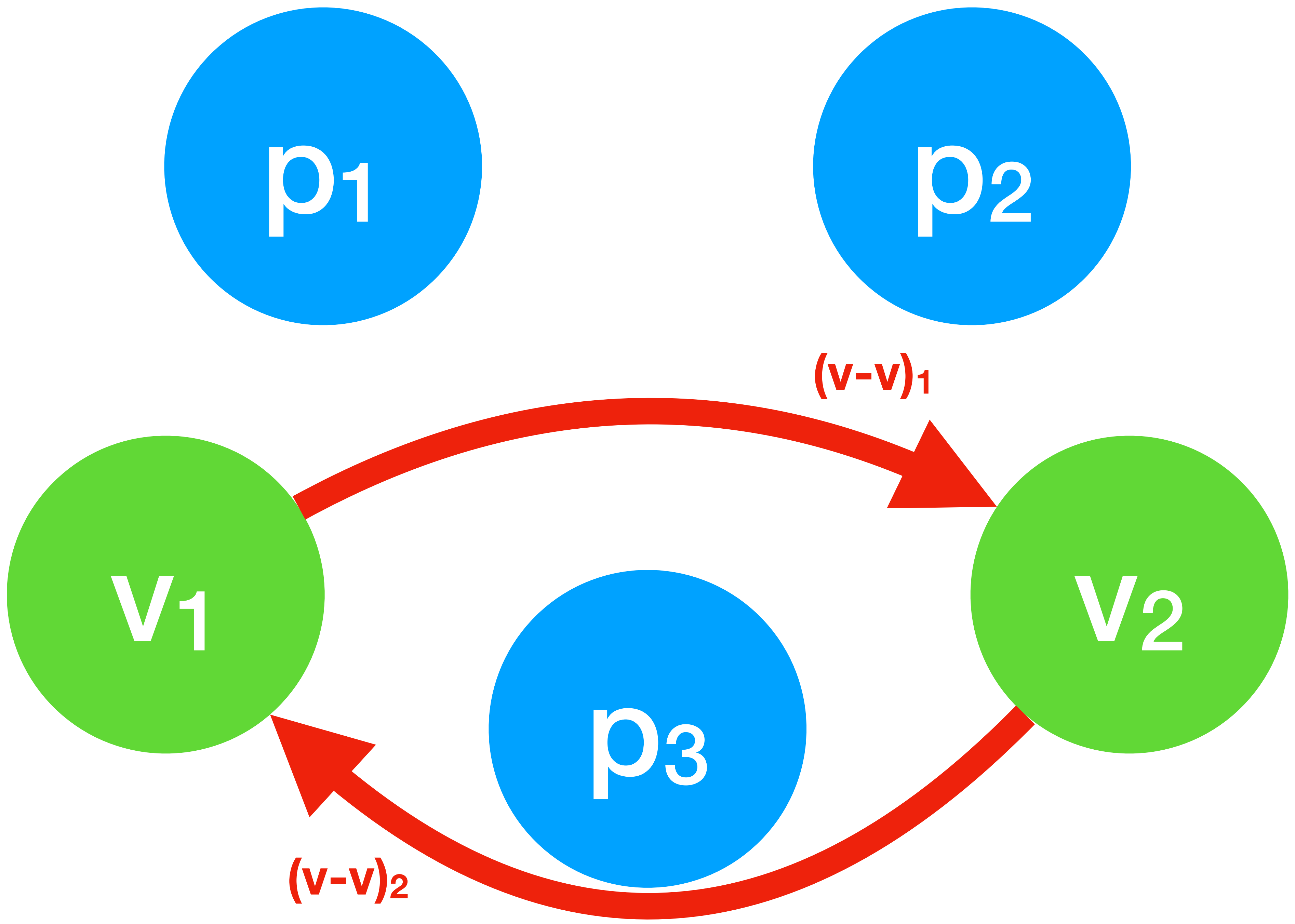



$v_i = [p_T^{rel}, \phi^{rel}, \eta^{rel}, \ldots, n_{tracks}, \cos\theta_{PV}, \ldots]$

# PARTICLES AND SECONDARY VERTICES: TWO INPUT GRAPHS

- Particles (i.e. tracks) and vertices are two separate inputs with different feature vectors (*heterogenous graph*)
- GNNs typically consider a *homogenous graph* (e.g. particle-particle graph)
- Vertex-vertex graph can also be considered

$p_i = [p_{\mathrm{T}}^{\mathrm{rel}}, \phi^{\mathrm{rel}}, \eta^{\mathrm{rel}}, \ldots, d_{3\mathrm{D}}, \mathrm{cov}(p_{\mathrm{T}}, p_{\mathrm{T}}), \ldots]$



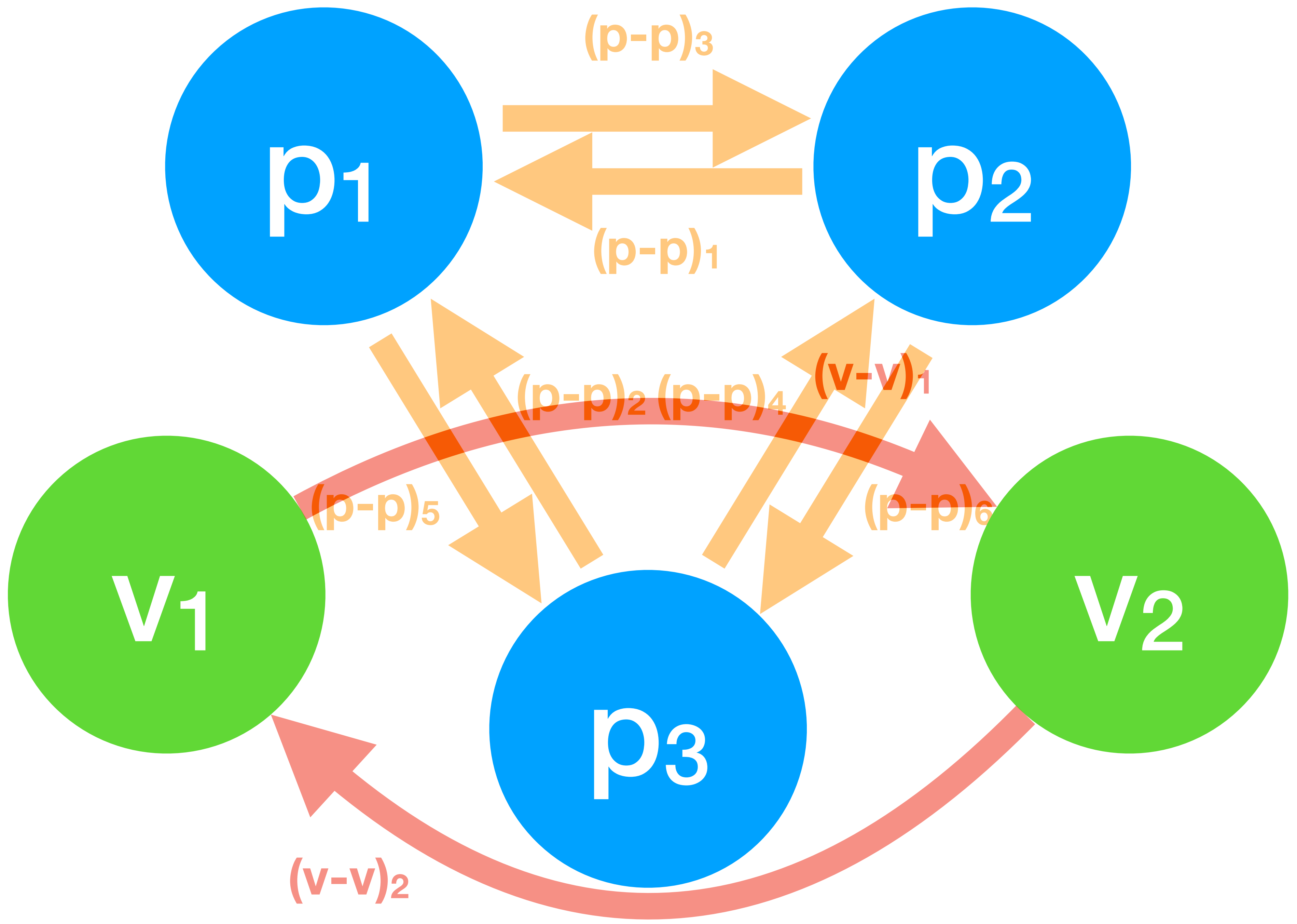

$v_i = [p_{\mathrm{T}}^{\mathrm{rel}}, \phi^{\mathrm{rel}}, \eta^{\mathrm{rel}}, \ldots, n_{\mathrm{tracks}}, \cos\theta_{\mathrm{PV}}, \ldots]$

# PARTICLES AND SECONDARY VERTICES: TWO INPUT GRAPHS



- Particles (i.e. tracks) and vertices are two separate inputs with different feature vectors (*heterogenous graph*)
- GNNs typically consider a *homogenous graph* (e.g. particle-particle graph)
- Vertex-vertex graph can also be considered
- Combined GNN can consider both by constructing two separate graphs

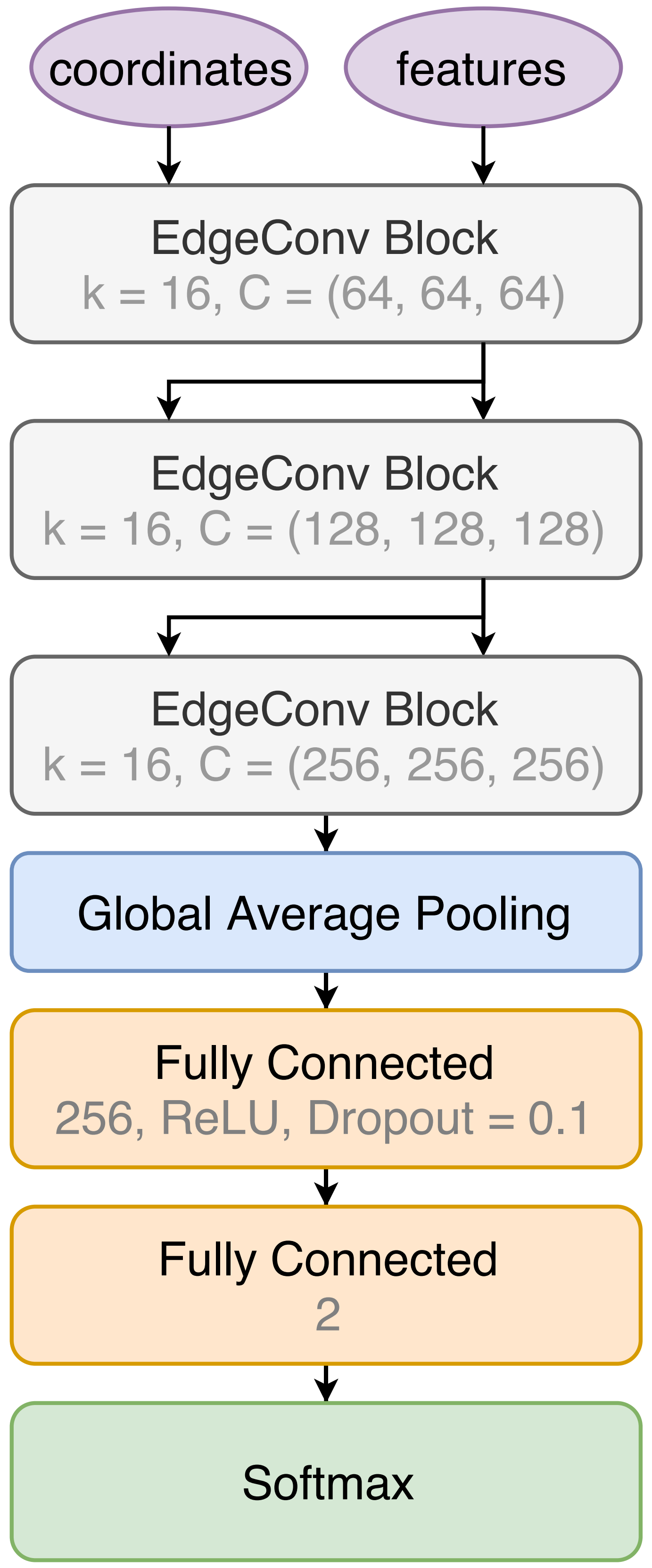
$p_i = [p_T^{rel}, \phi^{rel}, \eta^{rel}, \ldots, d_{3D}, \mathrm{cov}(p_T, p_T), \ldots]$




$v_i = [p_T^{rel}, \phi^{rel}, \eta^{rel}, \ldots, n_{tracks}, \cos\theta_{PV}, \ldots]$

# GNN FOR TAGGING H(BB) IN CMS

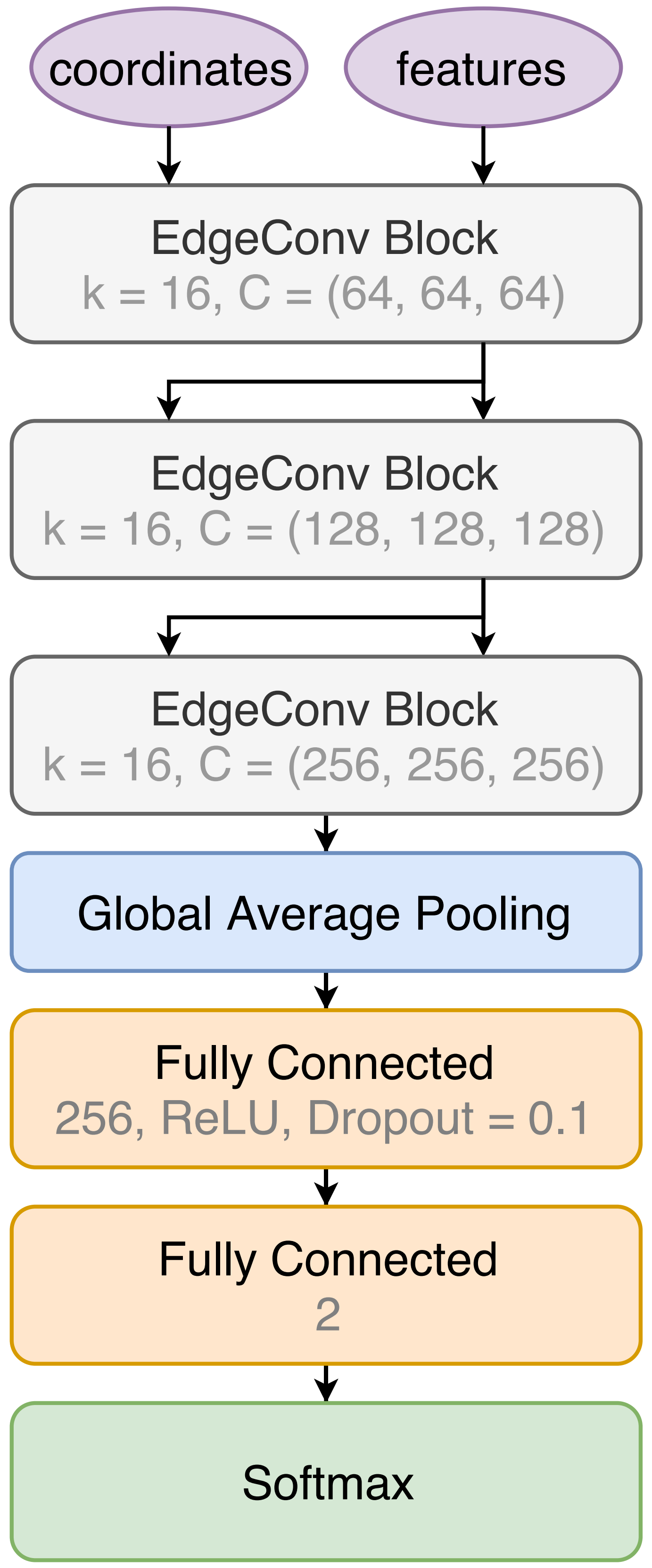




# GNN FOR TAGGING H(BB) IN CMS



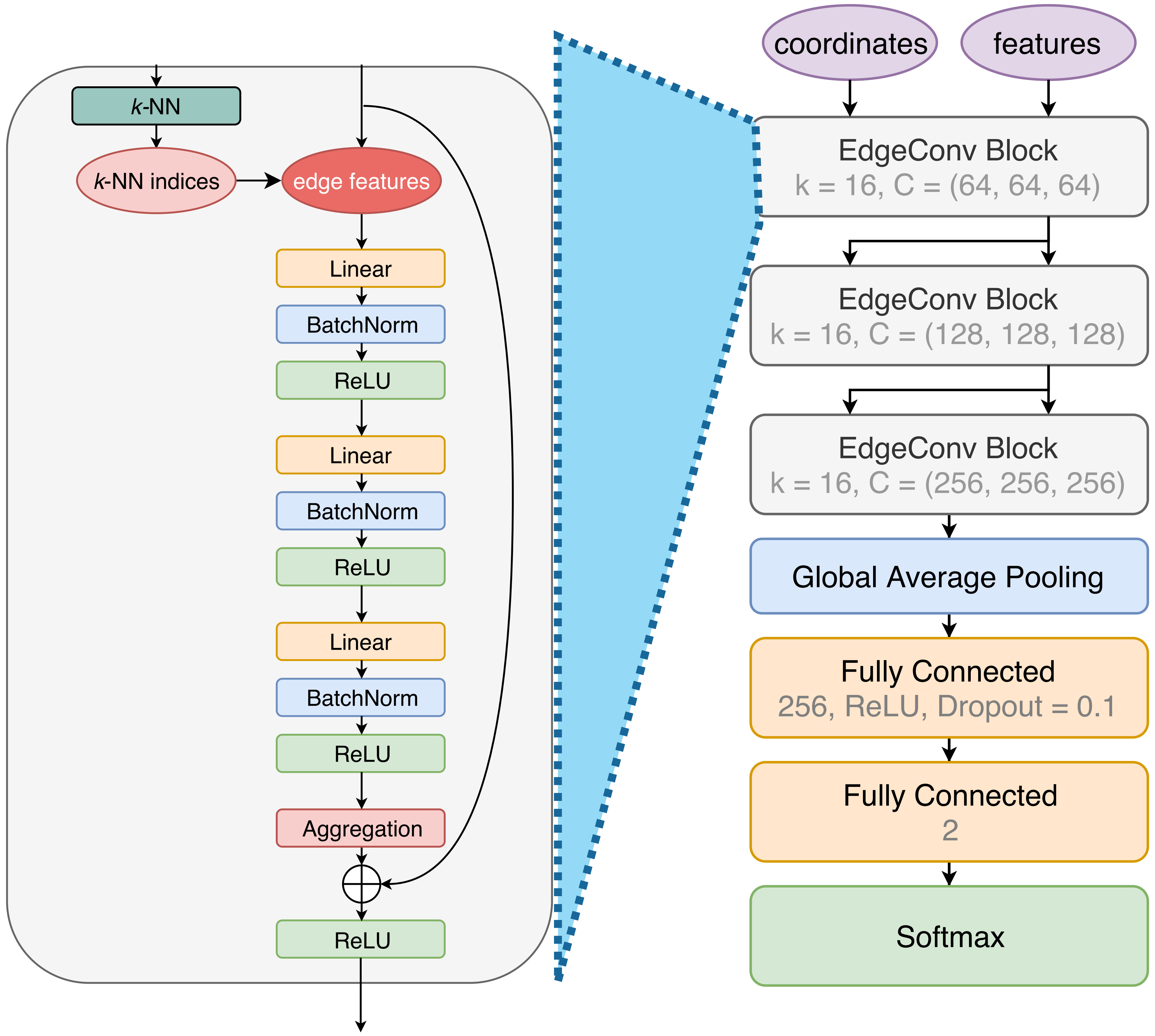
- ParticleNet, using "dynamic edge convolutions:" graph is constructed based on "closeness" in a latent space



# GNN FOR TAGGING H(BB) IN CMS

[arXiv:1902.08570](arXiv:1902.08570)
[CMS-DP-2020-002](CMS-DP-2020-002)

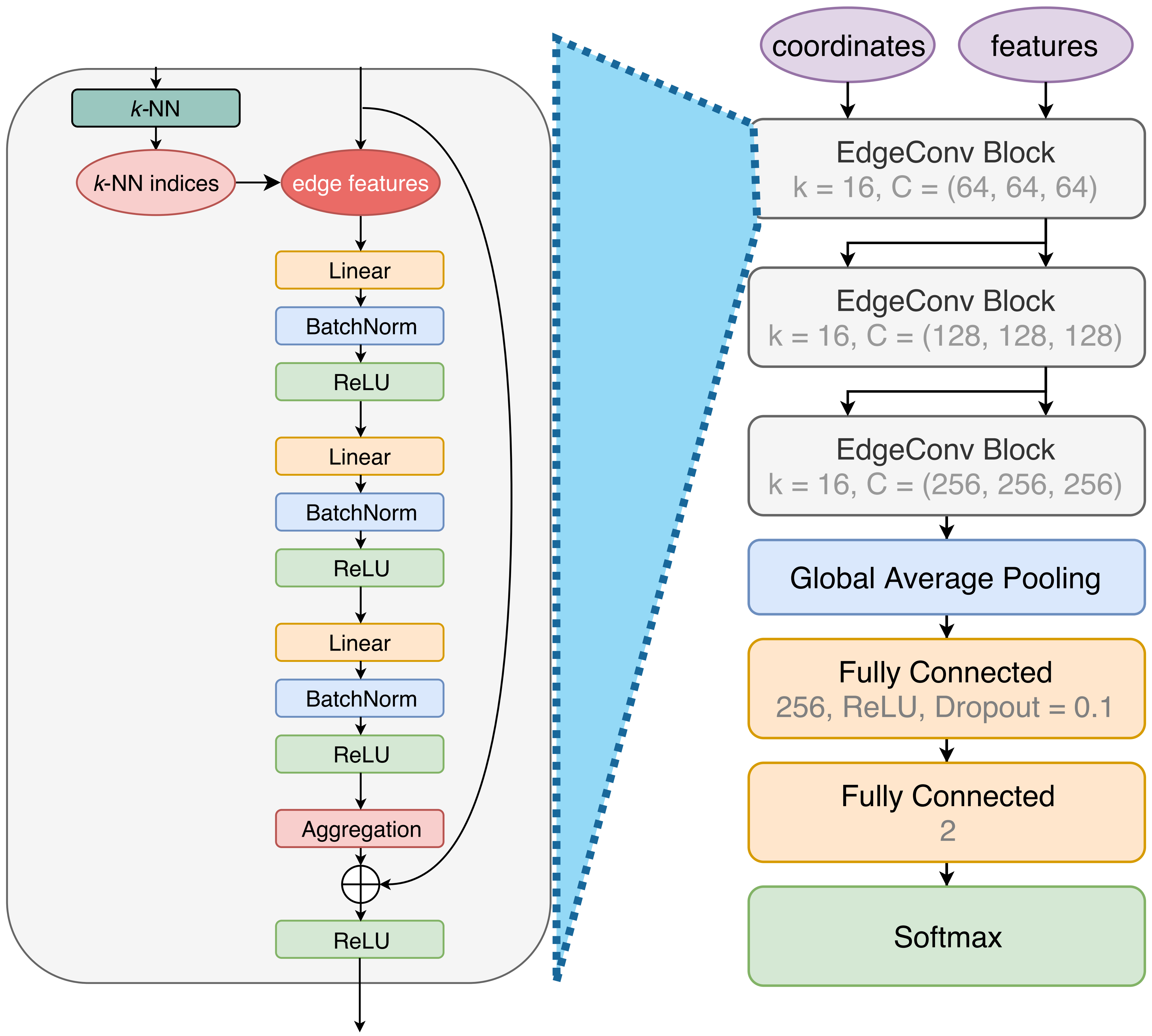

- ParticleNet, using "dynamic edge convolutions:" graph is constructed based on "closeness" in a latent space

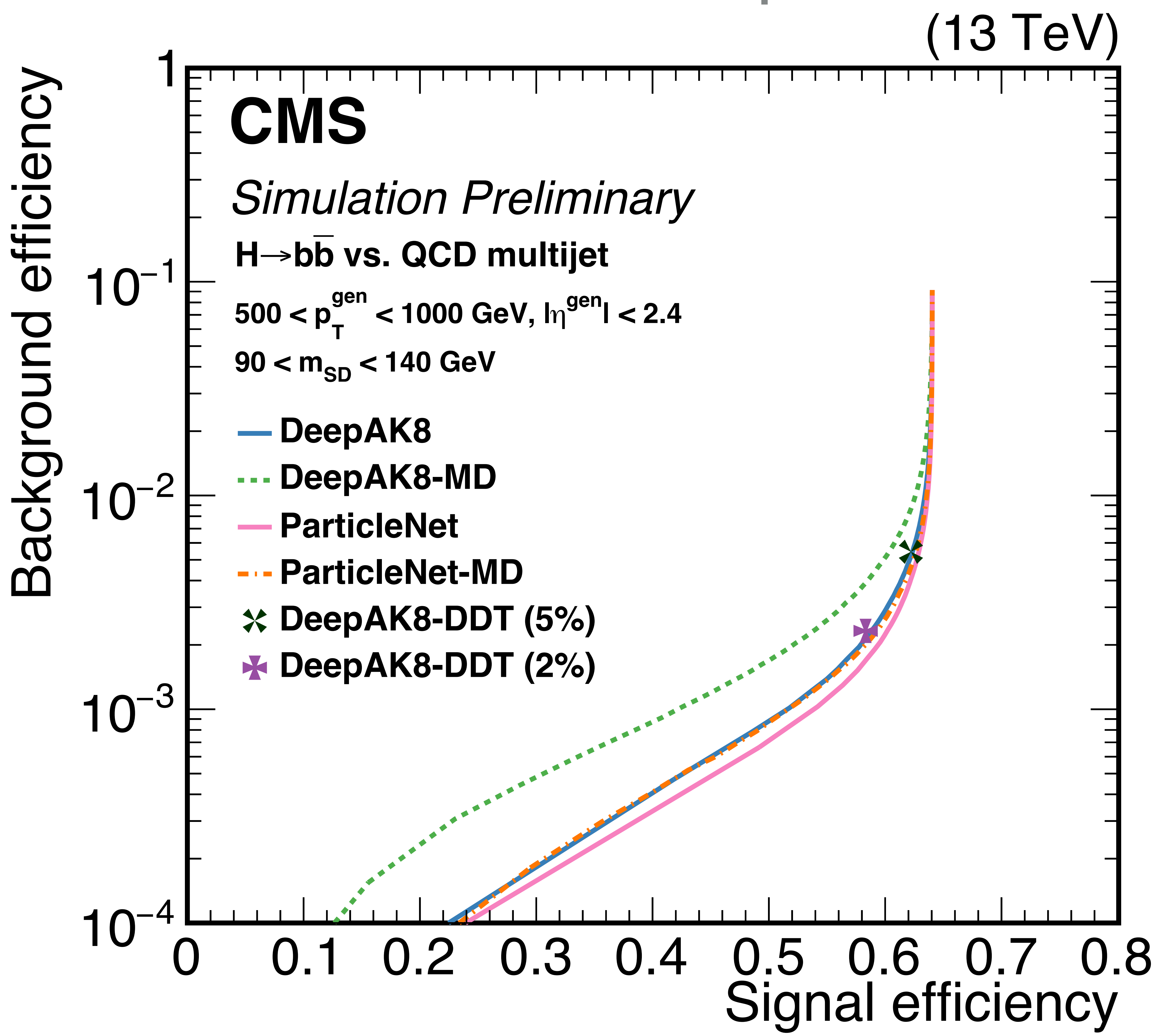


# GNN FOR TAGGING H(BB) IN CMS



- ParticleNet, using "dynamic edge convolutions:" graph is constructed based on "closeness" in a latent space
- Identifies H(bb) with a true positive rate of over 50% and a false positive rate of 0.1%



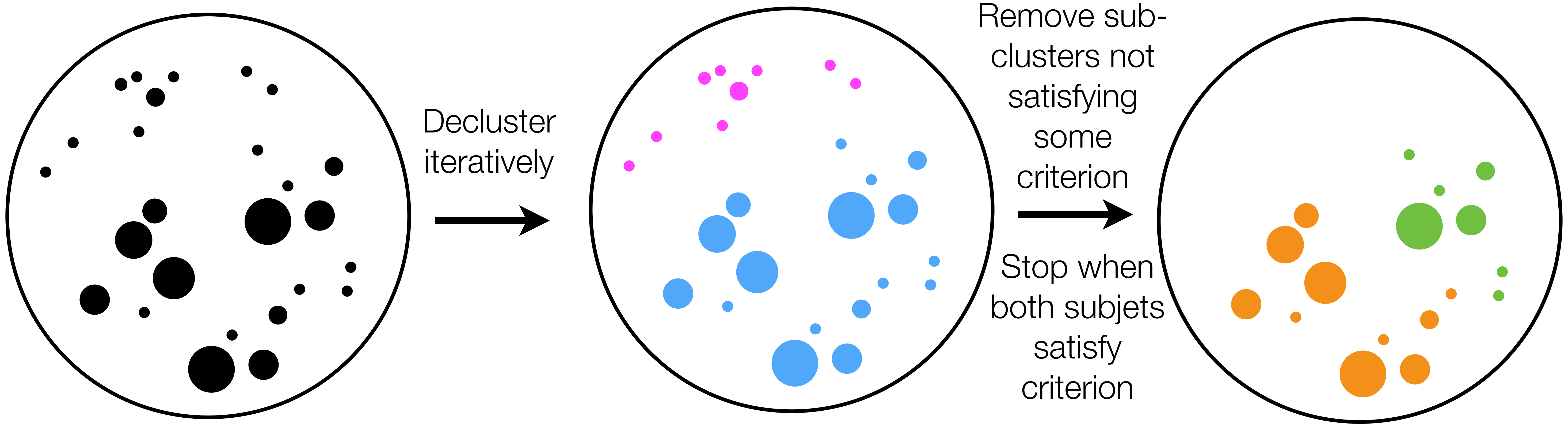
# BEYOND JET TAGGING: JET MASS

arXiv:1307.0007

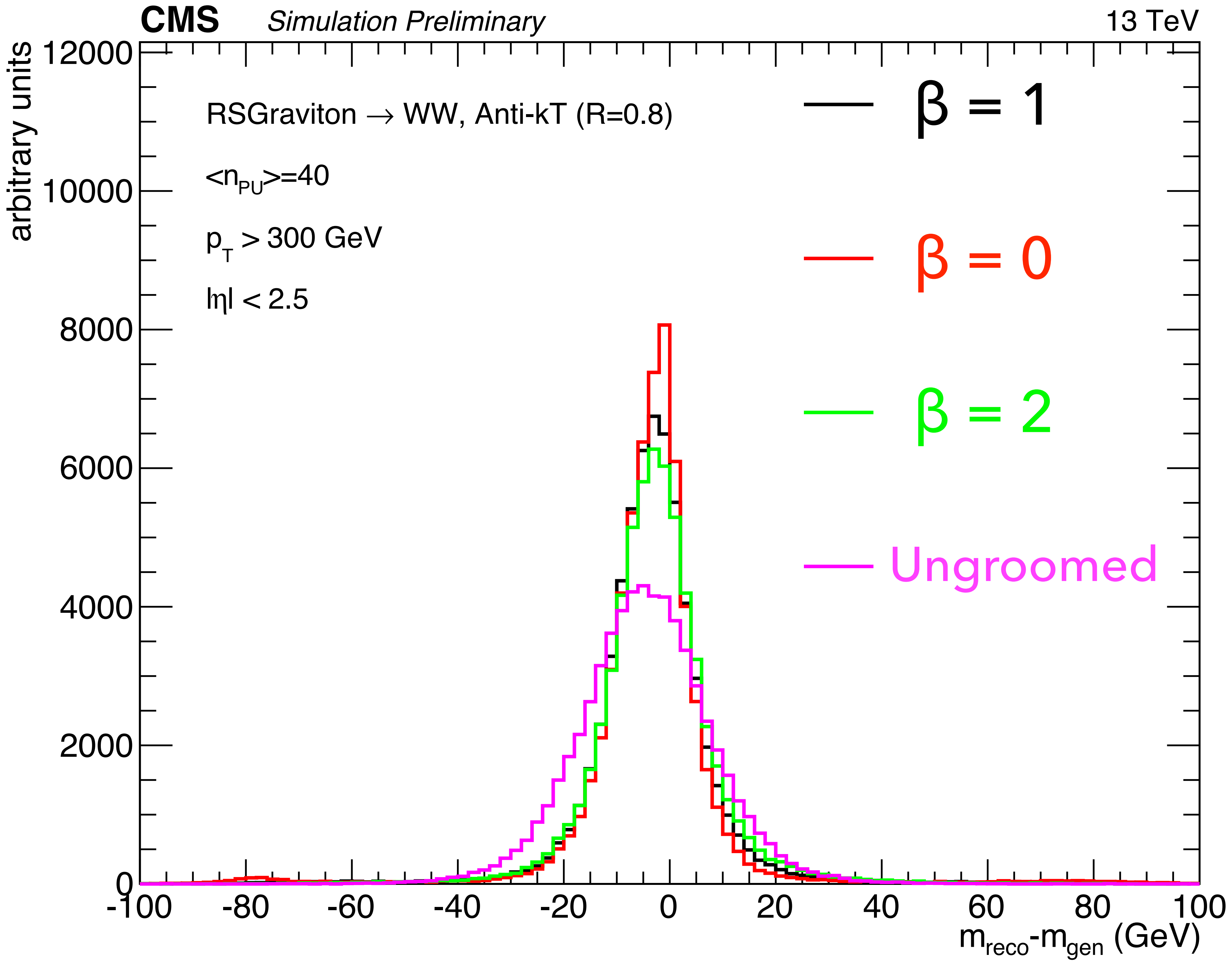
arXiv:1402.2657



- An important property used to analyze Higgs boson jets is the invariant mass
  - Provides good separation between W/Z/H-jets and q/g jets
  - Grooming removes soft and wide-angle radiation (soft drop is CMS standard)
  - Can we do better with ML?




$$\text{Soft Drop Condition:} \quad \frac{\min(p_{T1}, p_{T2})}{p_{T1} + p_{T2}} > z_{\text{cut}} \left( \frac{\Delta R_{12}}{R_0} \right)^{\beta}$$




CMS: $z_{cut} = 0.1$, $\beta = 0$

# GNN FOR MASS REGRESSION IN CMS

# GNN FOR MASS REGRESSION IN CMS



- Reuse ParticleNet architecture with a target of the “true” jet mass
- Special training samples incorporate X→bb, X→cc, X→qq with varying X mass in [15, 250] GeV

$$M_{\text{target}} = \begin{cases} M_{\text{SD}}^{\text{gen}} & \text{if jet is QCD} \\ m_{\text{X}} \in [15{,}250]\,\text{GeV} & \text{otherwise} \end{cases}$$

- Minimize loss function:

$$L(y, y^p) = \sum_{i=1}^{n} \log\cosh(y_i^p - y_i)$$

# GNN FOR MASS REGRESSION IN CMS

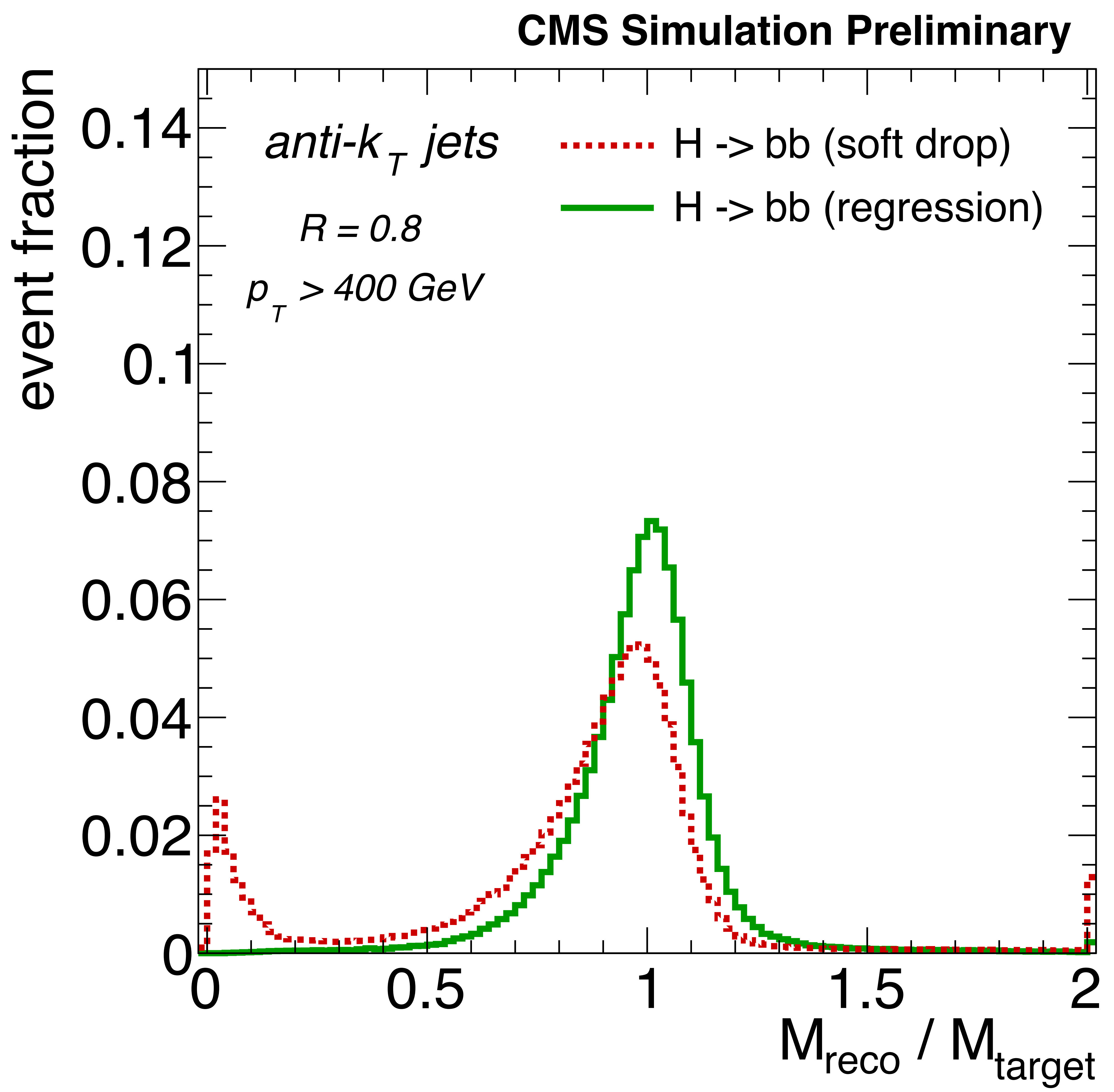
- Reuse ParticleNet architecture with a target of the "true" jet mass
- Special training samples incorporate X→bb, X→cc, X→qq with varying X mass in [15, 250] GeV

$$M_{\text{target}} = \begin{cases} M_{\text{SD}}^{\text{gen}} & \text{if jet is QCD} \\ m_{\text{X}} \in [15{,}250]\,\text{GeV} & \text{otherwise} \end{cases}$$

- Minimize loss function:

$$L(y, y^p) = \sum_{i=1}^{n} \log\cosh(y_i^p - y_i)$$

- Substantial scale and resolution improvement



# GNN FOR MASS REGRESSION IN CMS

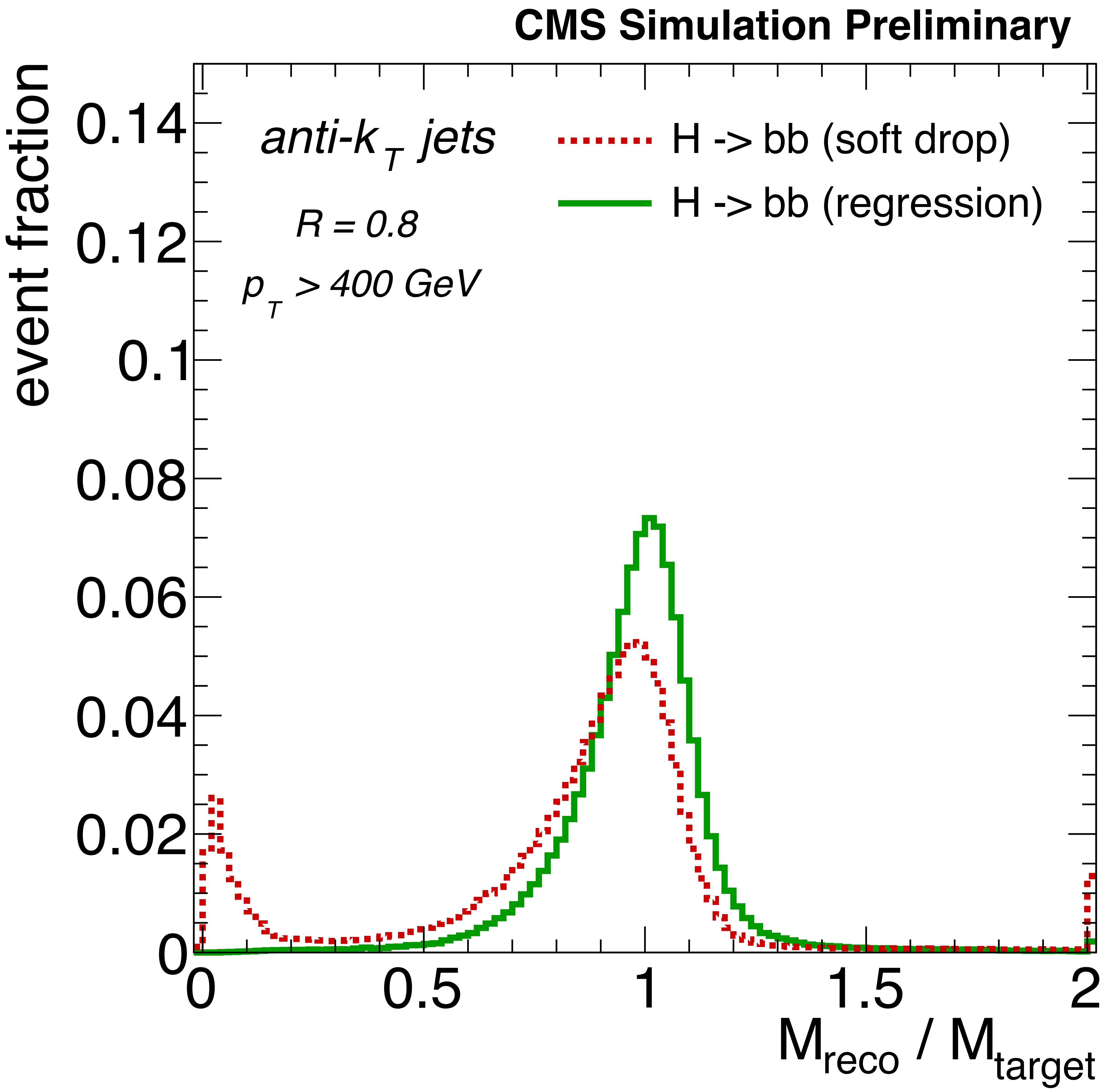
- Reuse ParticleNet architecture with a target of the "true" jet mass
- Special training samples incorporate X→bb, X→cc, X→qq with varying X mass in [15, 250] GeV

$$M_{\text{target}} = \begin{cases} M_{\text{SD}}^{\text{gen}} & \text{if jet is QCD} \\ m_{\text{X}} \in [15{,}250]\,\text{GeV} & \text{otherwise} \end{cases}$$

- Minimize loss function:

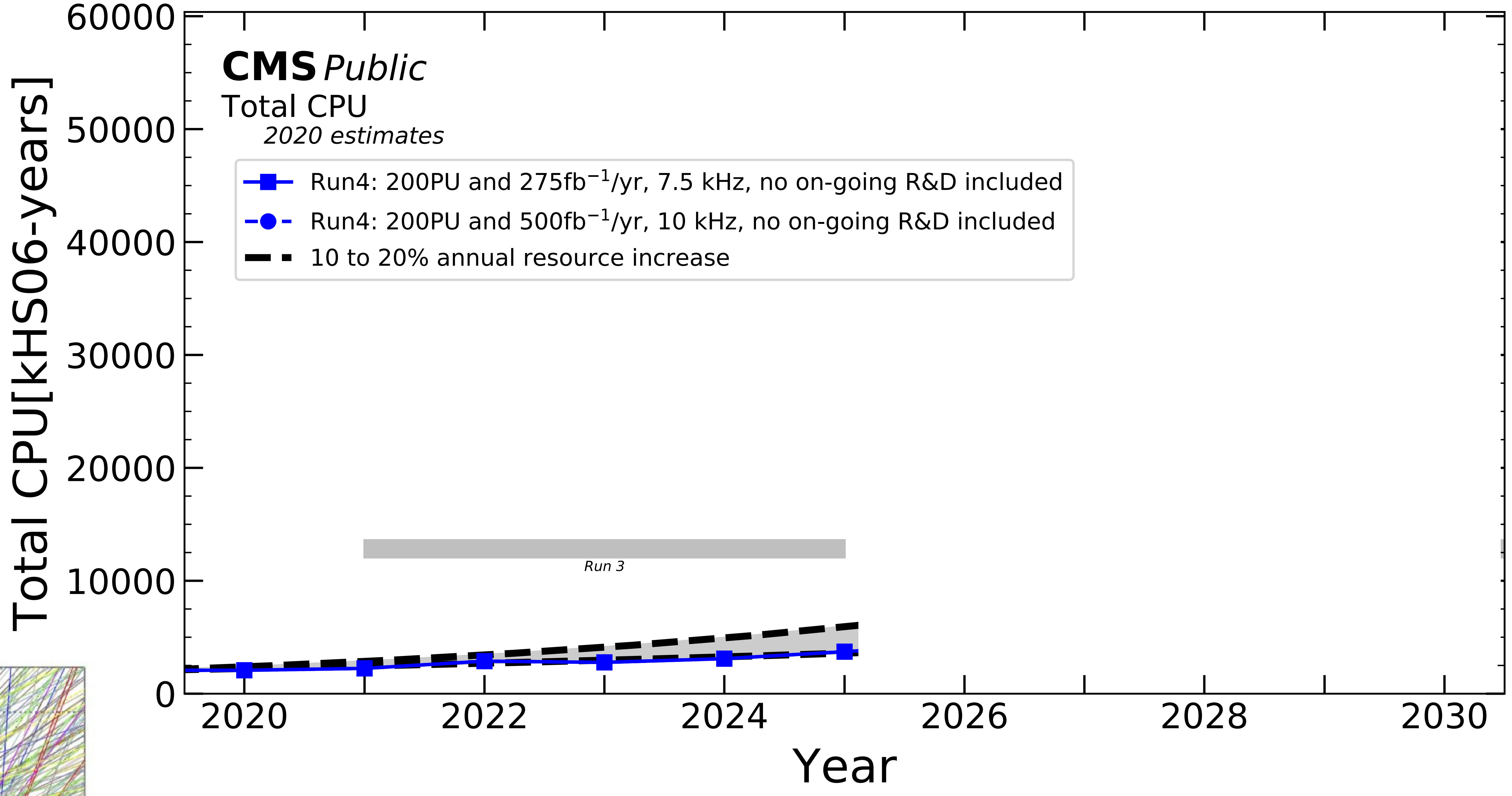
$$L(y, y^p) = \sum_{i=1}^{n} \log\cosh(y_i^p - y_i)$$

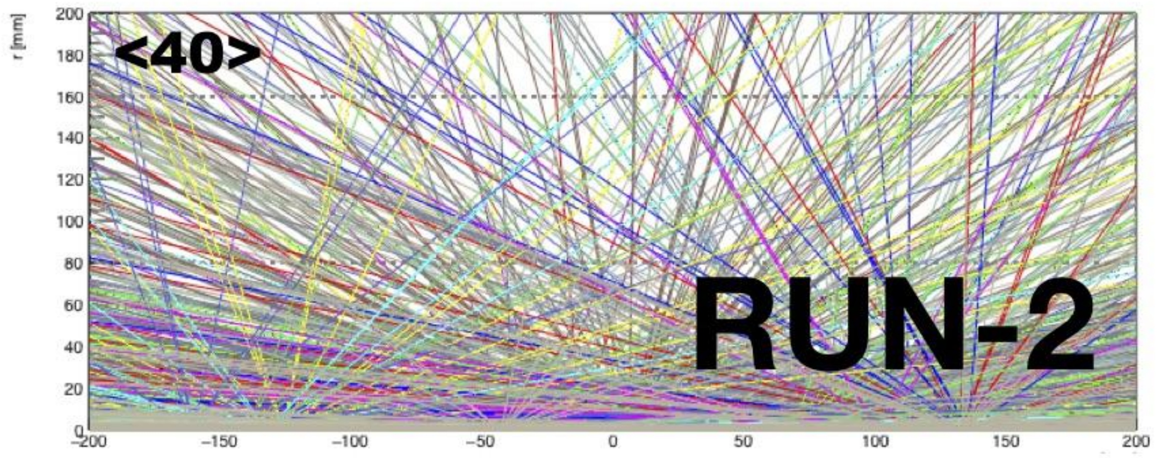
- Substantial scale and resolution improvement
  - Can increase sensitivity by 20-25% to rare Higgs boson signals like HH, VBF, …



# ML FOR TAGGING
# ML FOR GEN/SIM
# FAST ML FOR TRIGGER

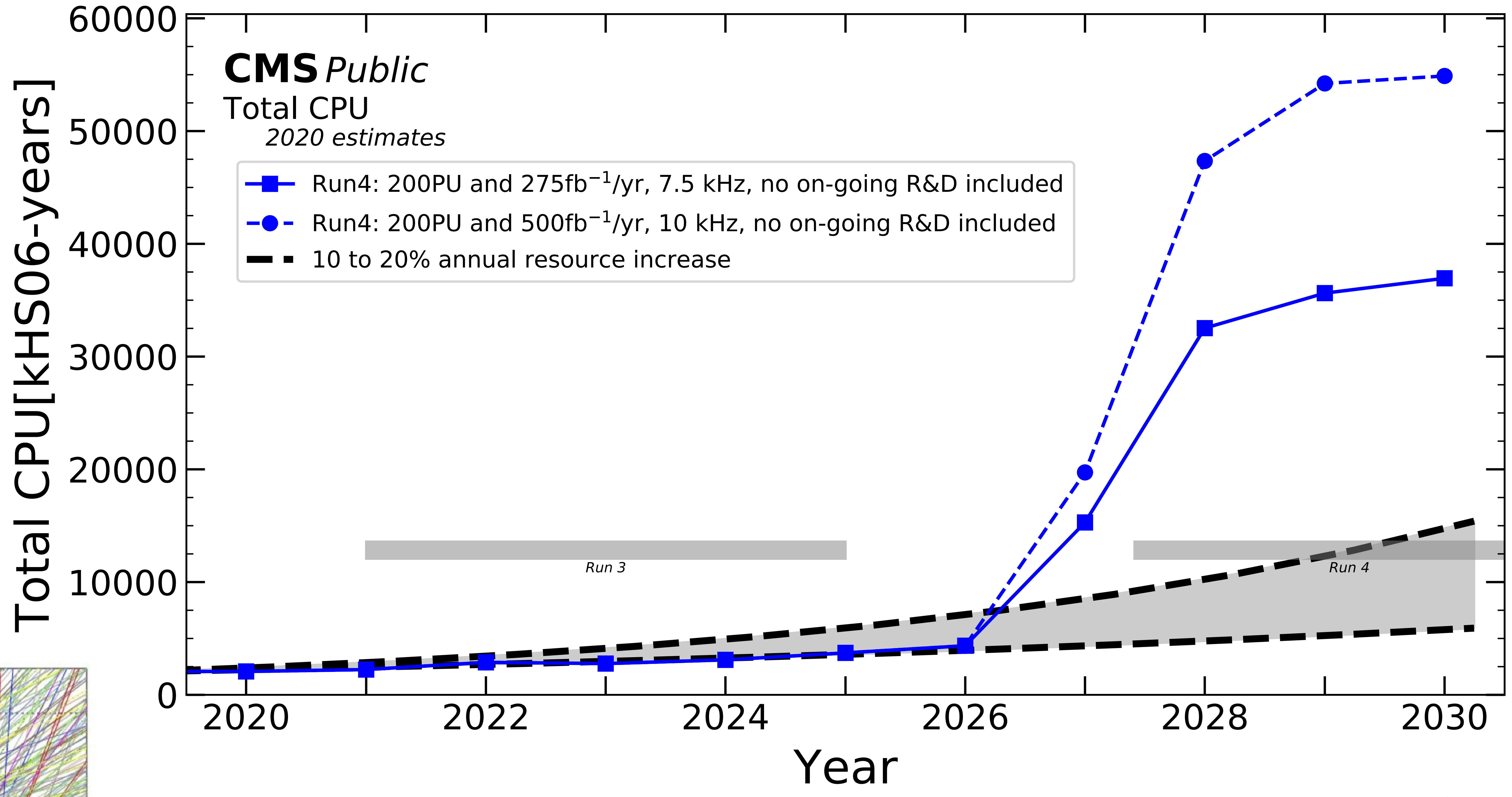
# CPU DEMANDS AT THE UPGRADED LHC

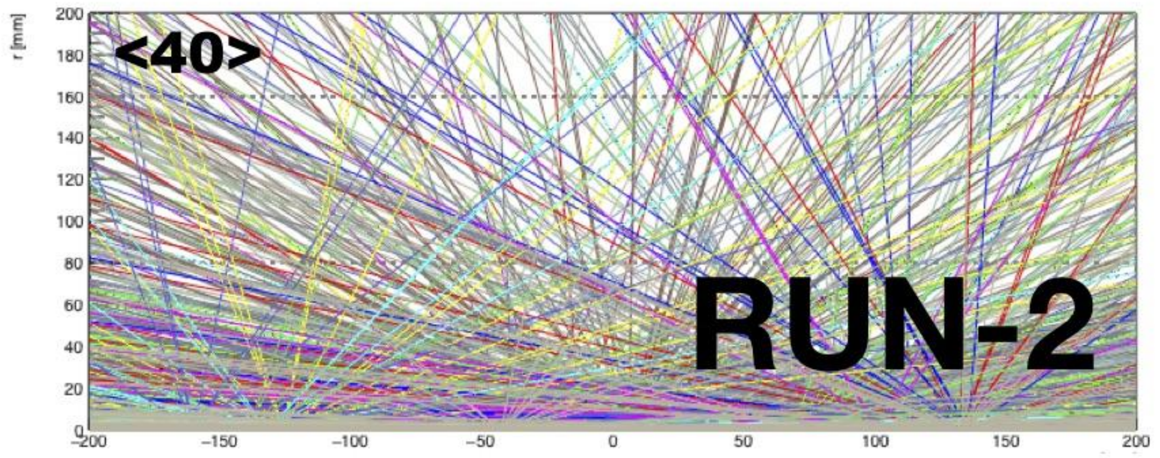


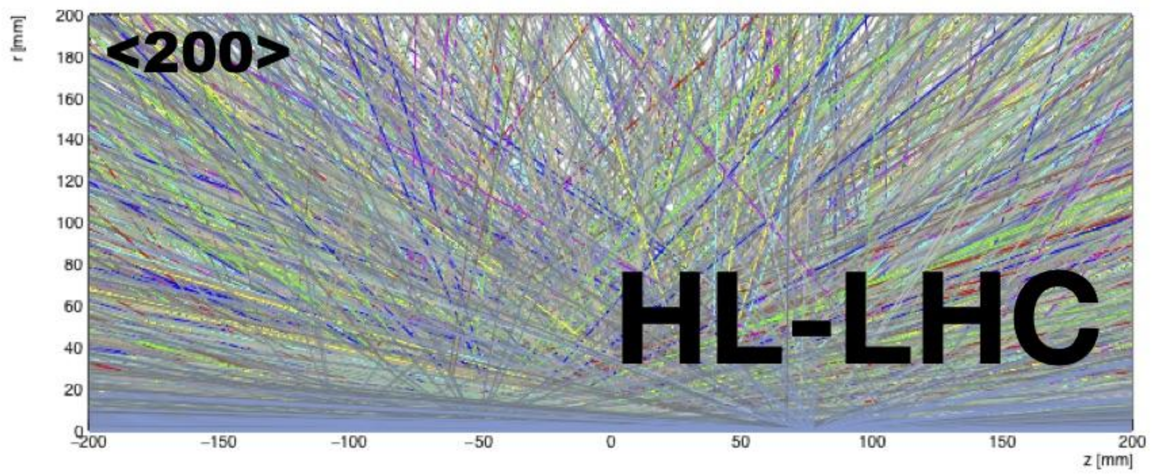
# CPU DEMANDS AT THE UPGRADED LHC





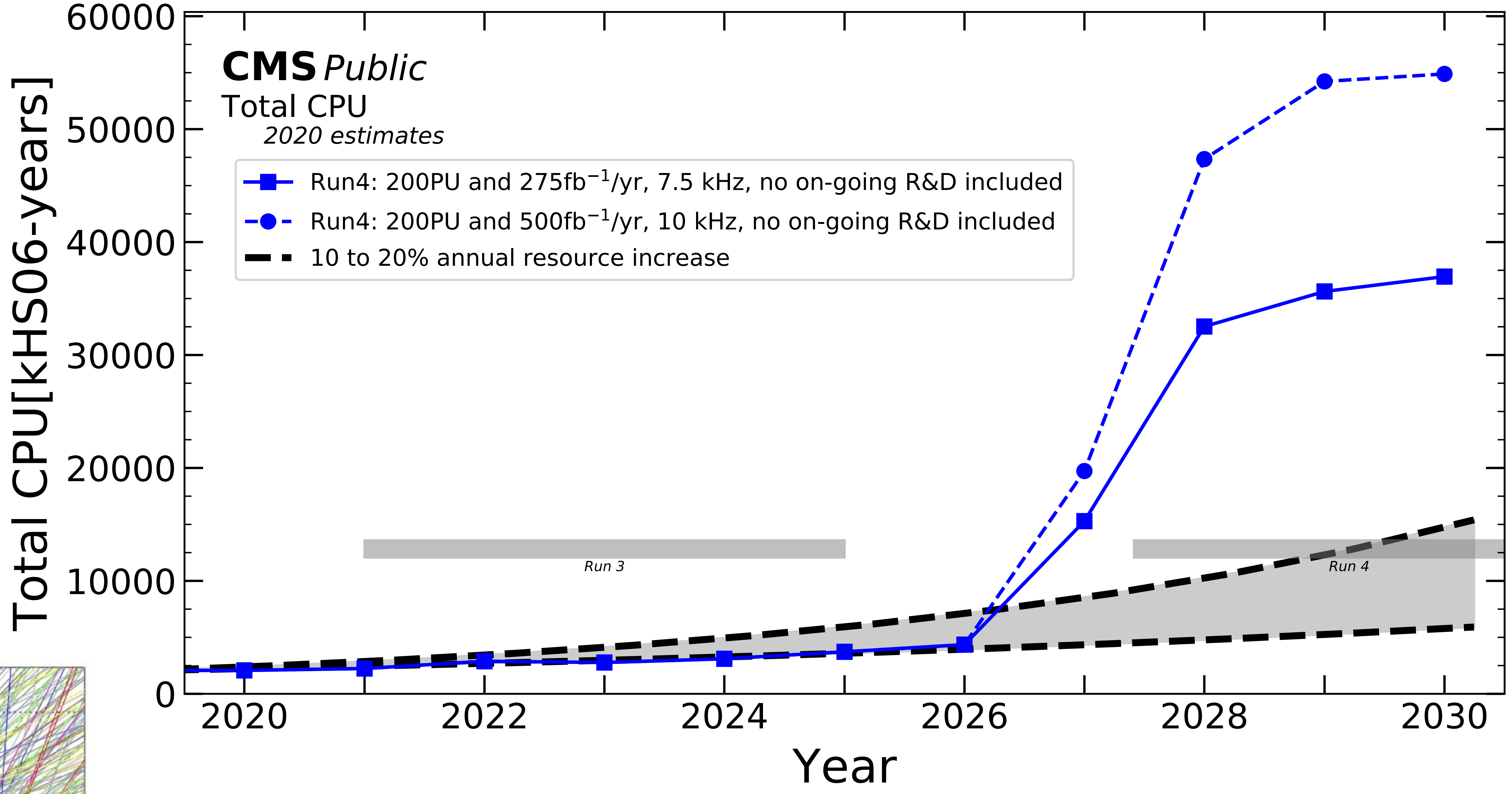

LHC upgrade

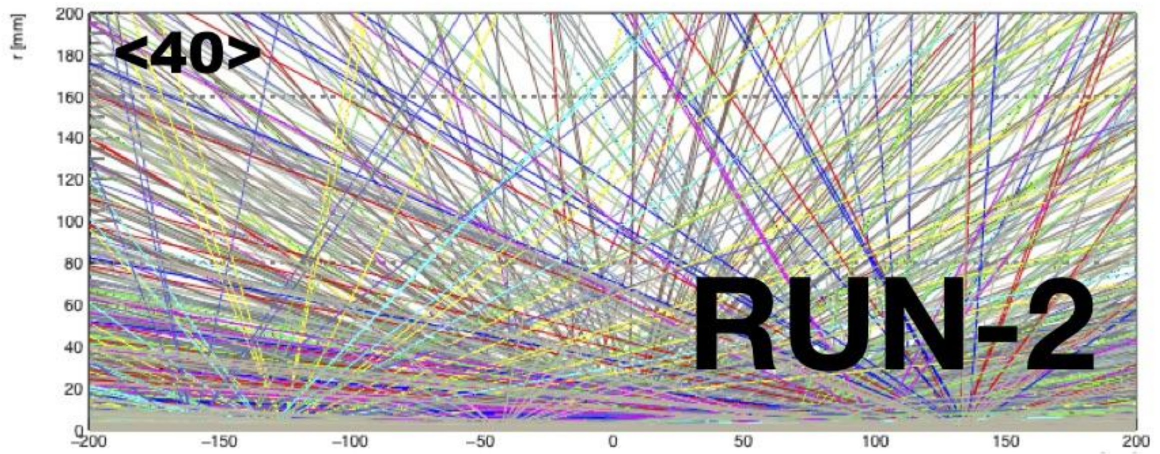



- Computing demands increase nonlinearly with increasing “pileup”

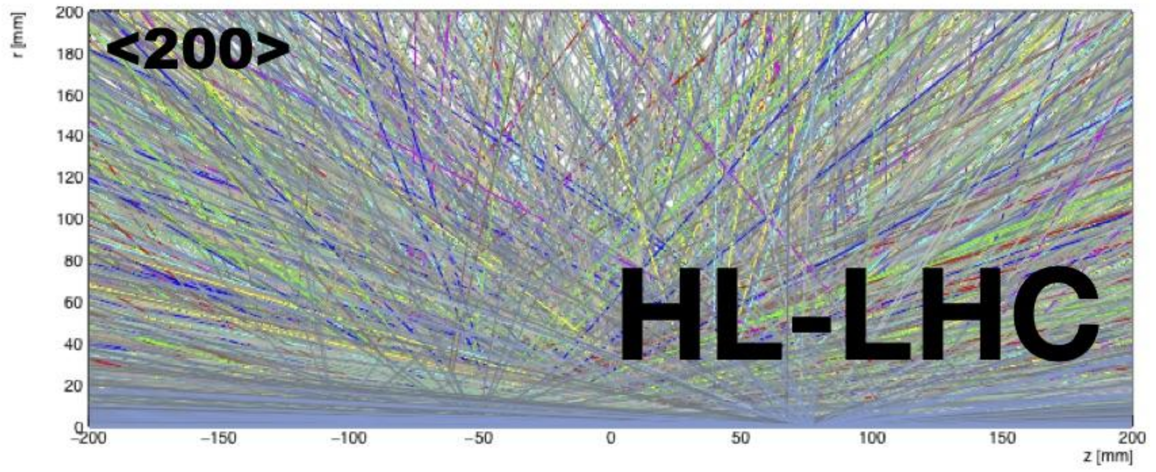
# CPU DEMANDS AT THE UPGRADED LHC





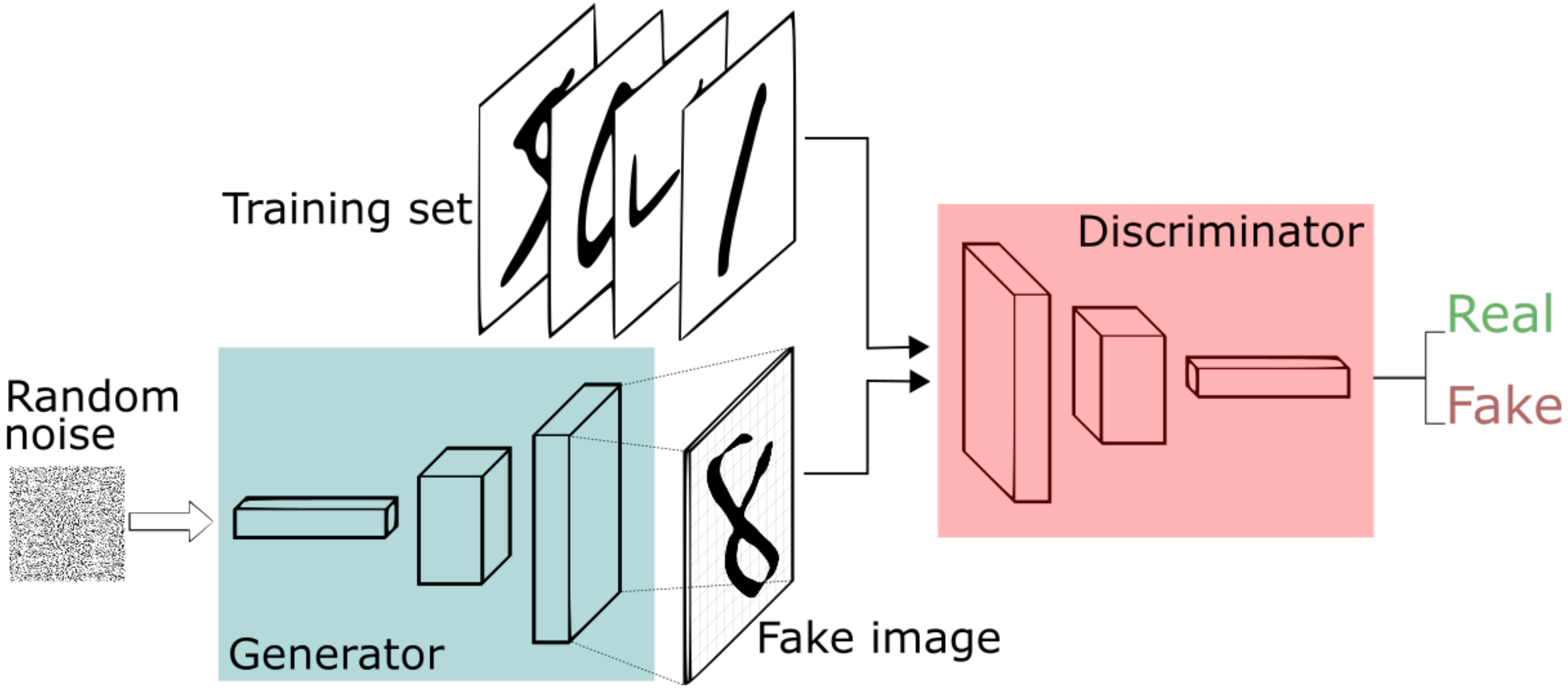

LHC upgrade




- Computing demands increase nonlinearly with increasing “pileup”
- Need more processing power (or smarter algorithms like **deep learning**) to keep up with demands

# GENERATIVE ADVERSARIAL NETWORKS

arXiv:1406.2661
arXiv:1912.04958

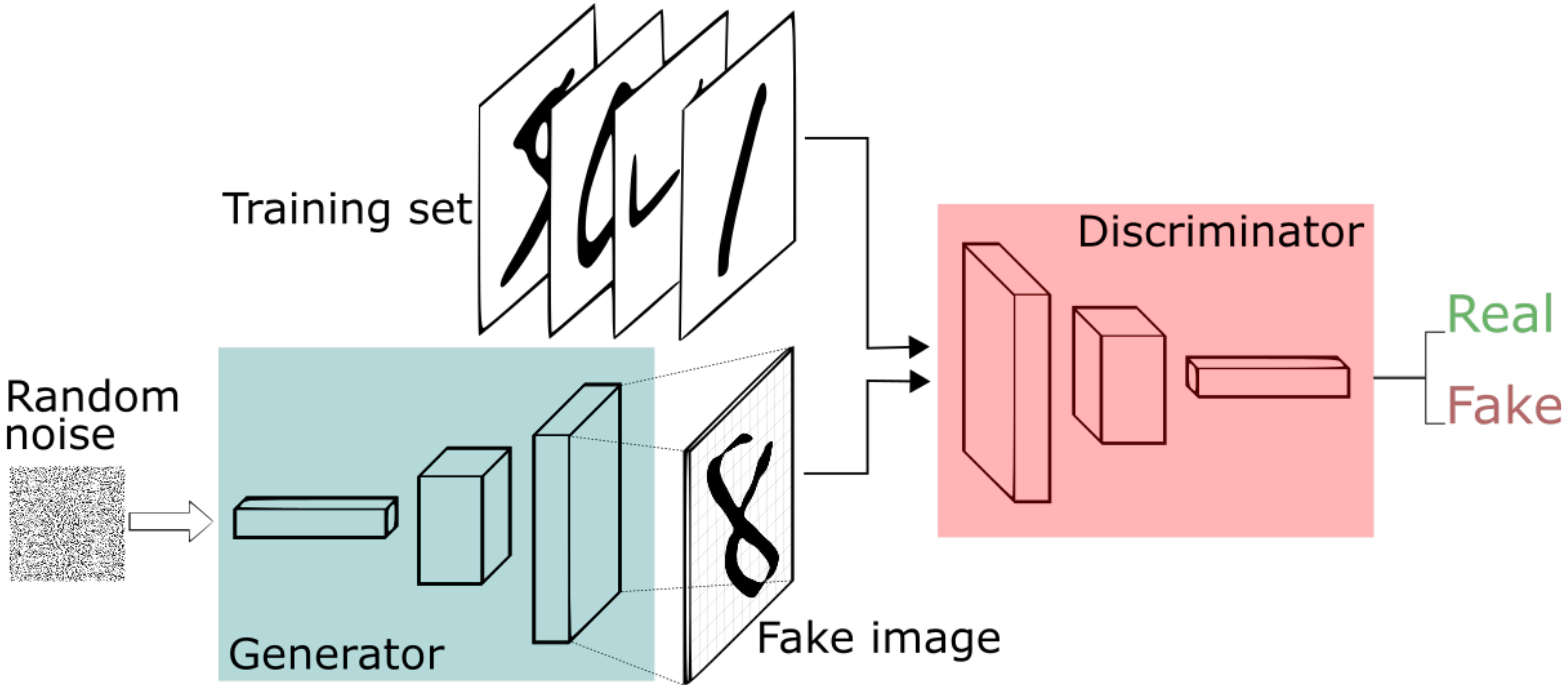



- Train two neural networks in tandem:
  - one to generate realistic “fake” data
  - the other to discriminate “real” from “fake” data

# GENERATIVE ADVERSARIAL NETWORKS

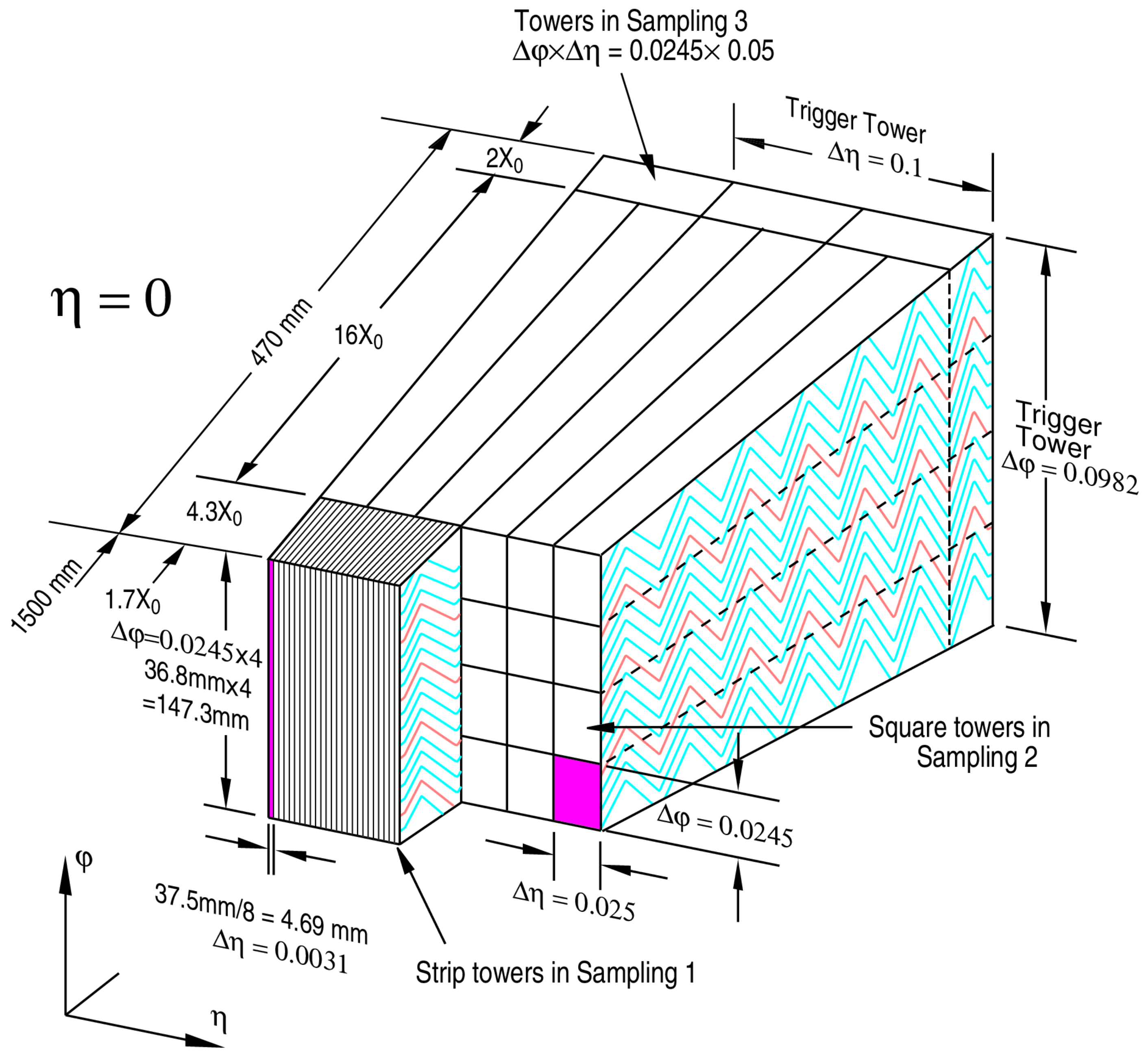
arXiv:1406.2661
arXiv:1912.04958






thispersondoesnotexist.com

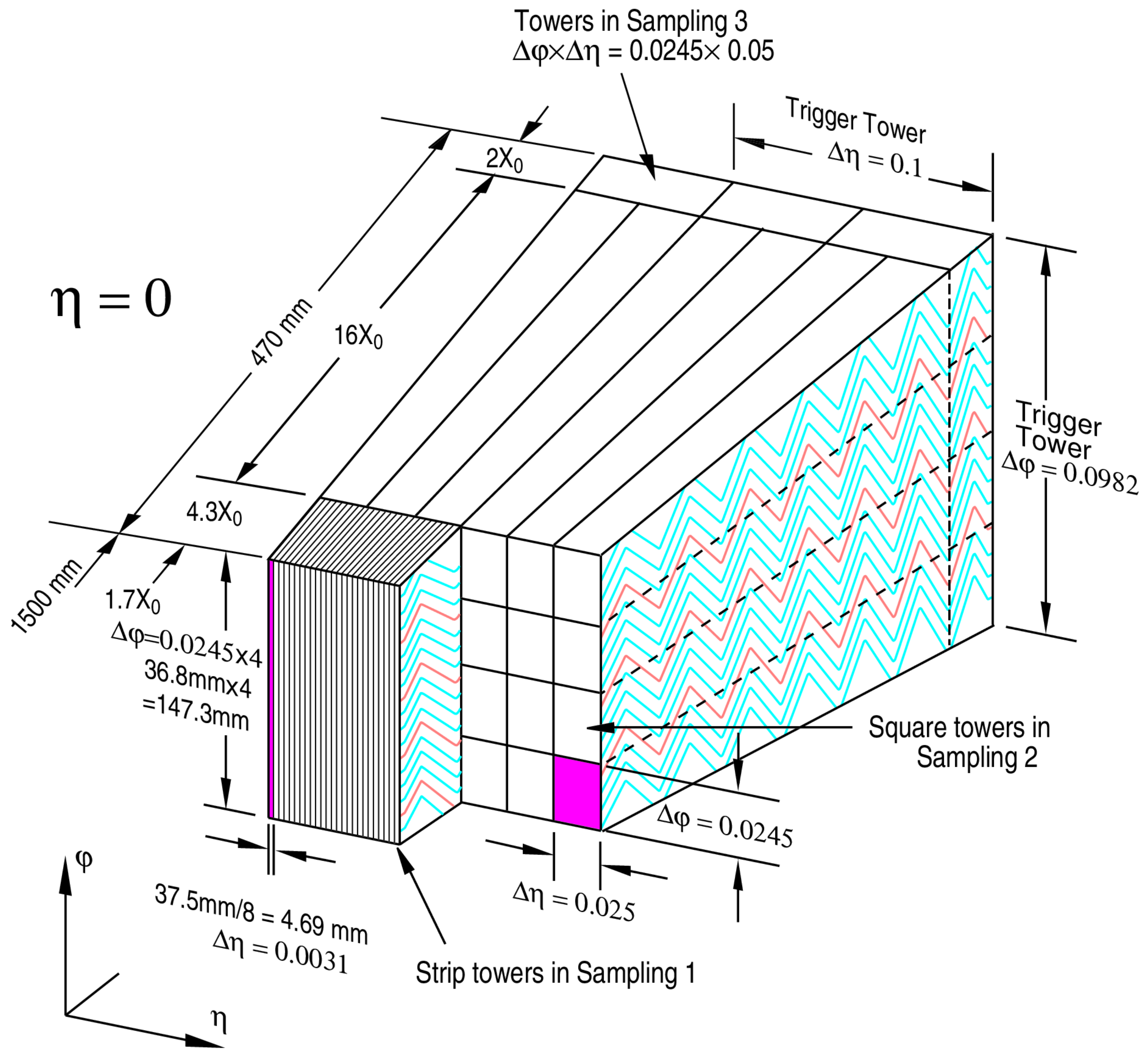
- Train two neural networks in tandem:
  - one to generate realistic “fake” data
  - the other to discriminate “real” from “fake” data



Towers in Sampling 3
Δφ×Δη = 0.0245× 0.05
Trigger Tower
Δη = 0.1
2X0
η = 0
470 mm
16X0
Trigger
Tower
Δφ = 0.0982
4.3X0
1500 mm
1.7X0
Δφ=0.0245x4
36.8mmx4
=147.3mm
Square towers in
Sampling 2
Δφ = 0.0245
φ
Δη = 0.025
37.5mm/8 = 4.69 mm
Δη = 0.0031
Strip towers in Sampling 1
η

# ATLAS FASTCALOGAN

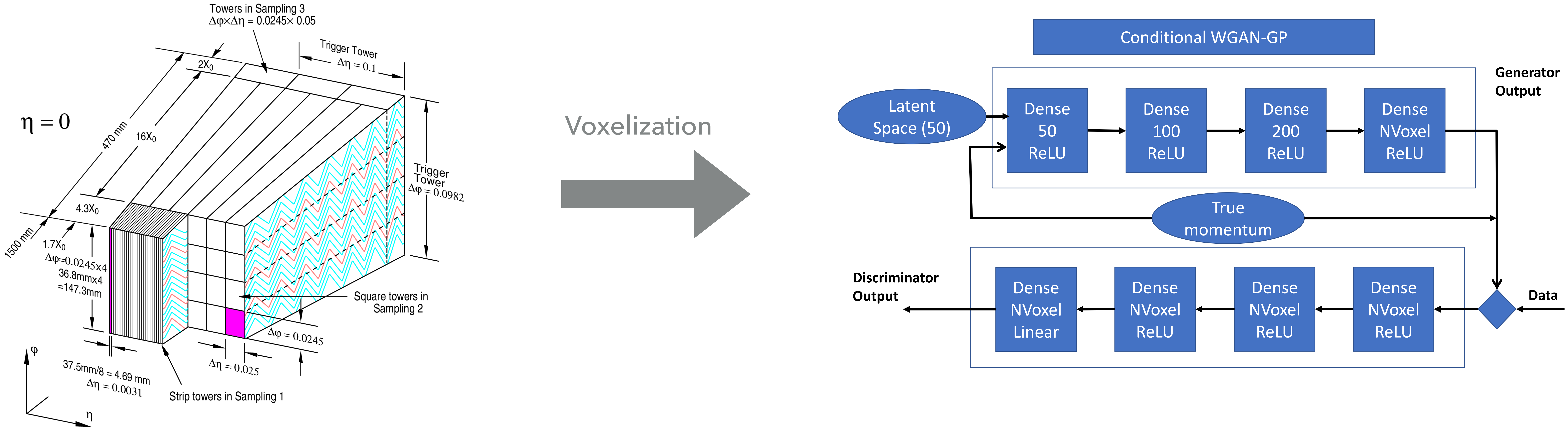





- Geant4-based ATLAS simulation of the full calorimeter is slow; can a GAN replace this?

# ATLAS FASTCALOGAN

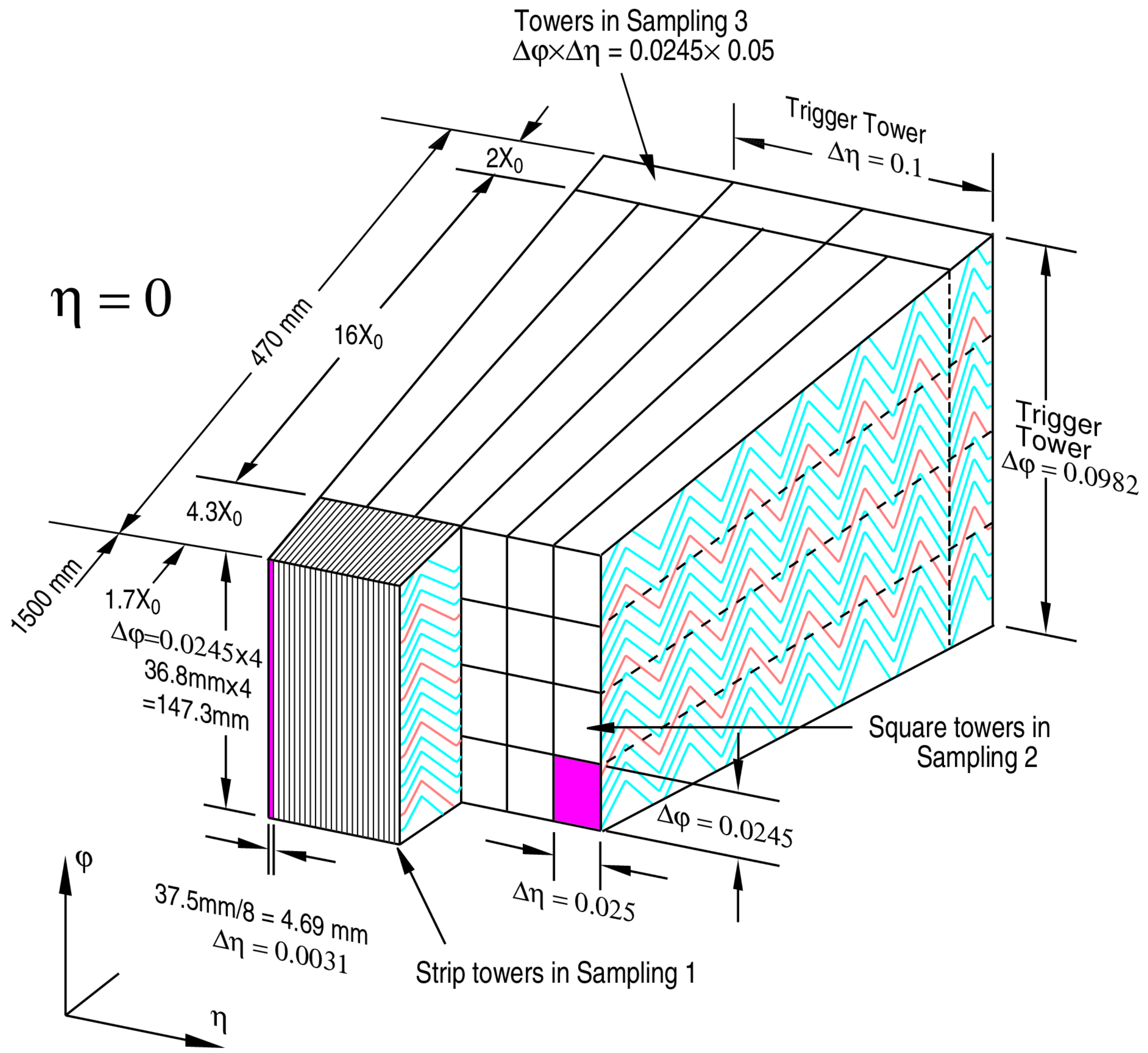




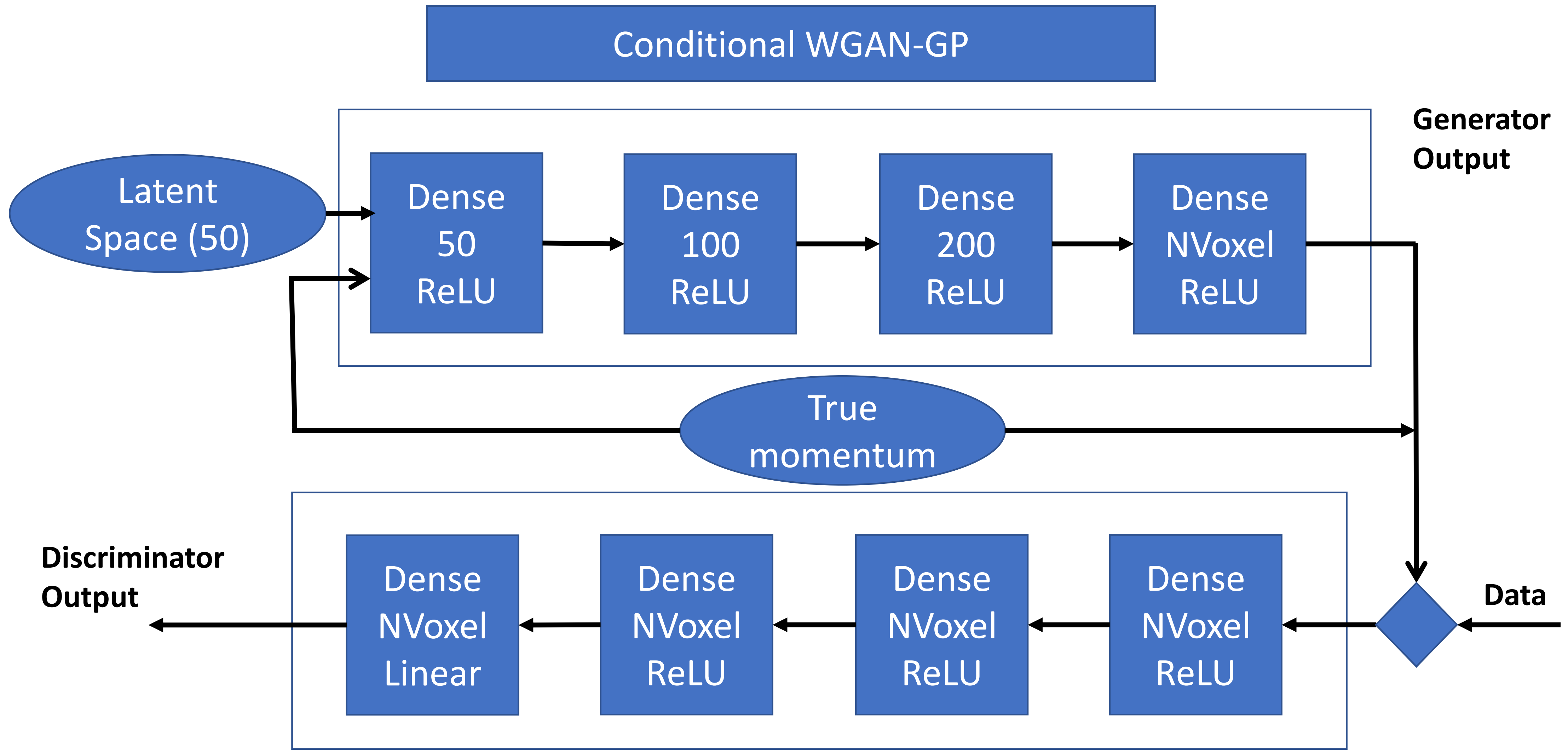

- Geant4-based ATLAS simulation of the full calorimeter is slow; can a GAN replace this?
- 300 GANs trained to parametrize the detector response to photons, electrons and pions

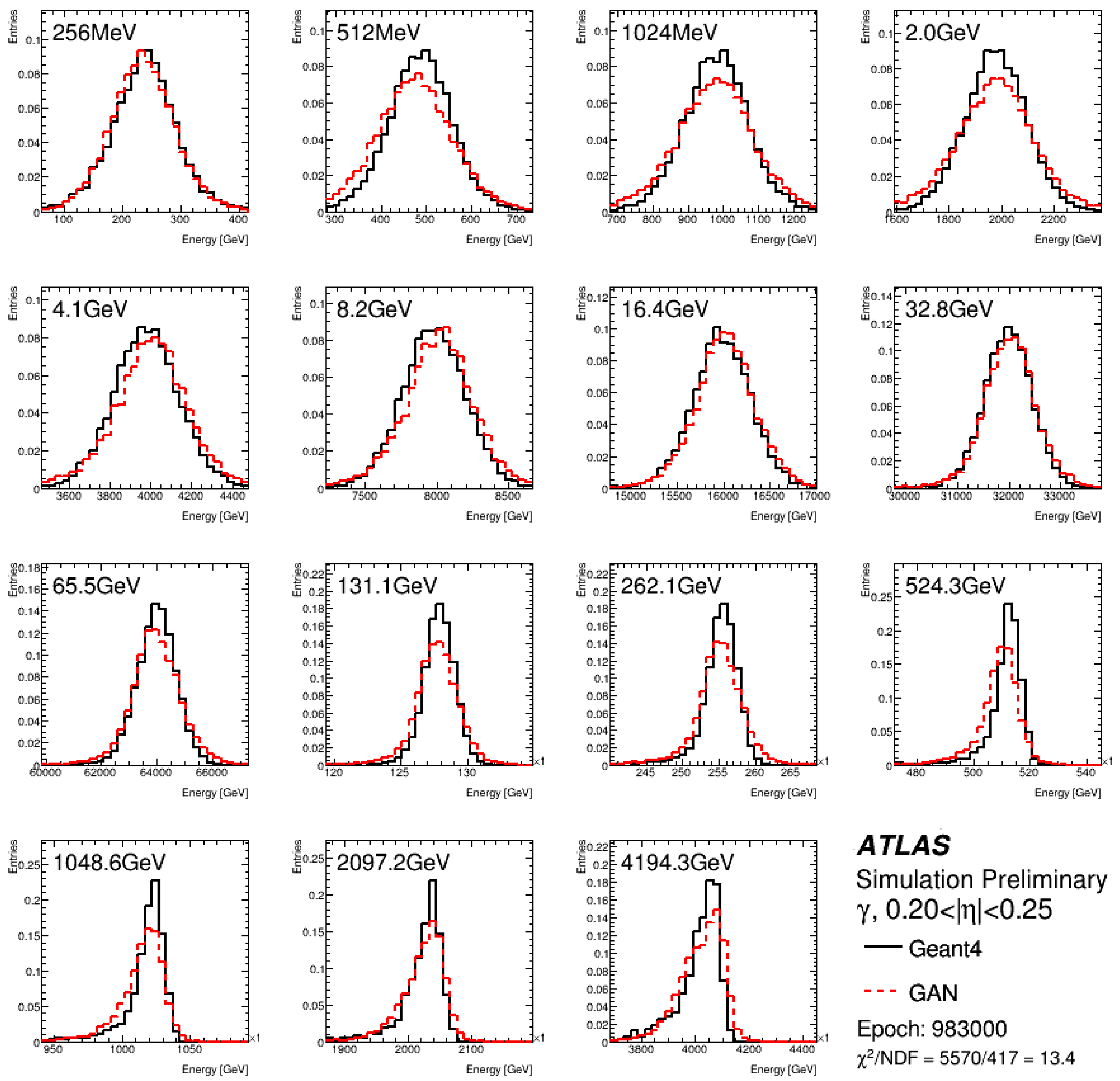
# ATLAS FASTCALOGAN






Voxelization

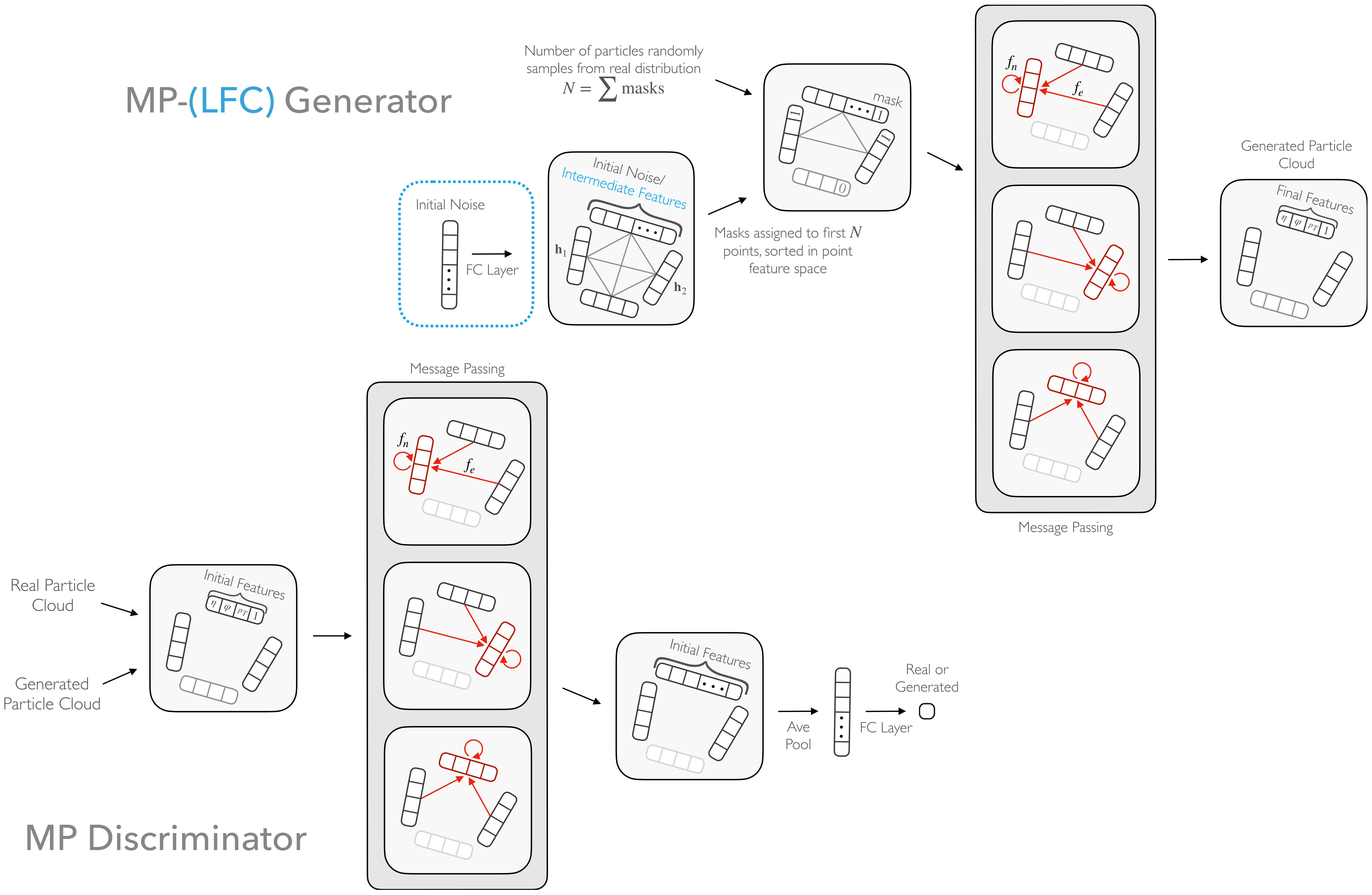



- Geant4-based ATLAS simulation of the full calorimeter is slow; can a GAN replace this?
- 300 GANs trained to parametrize the detector response to photons, electrons and pions
- Good agreement between the GAN and Geant4 both for single-particle showers and jets

# GRAPH-BASED GAN

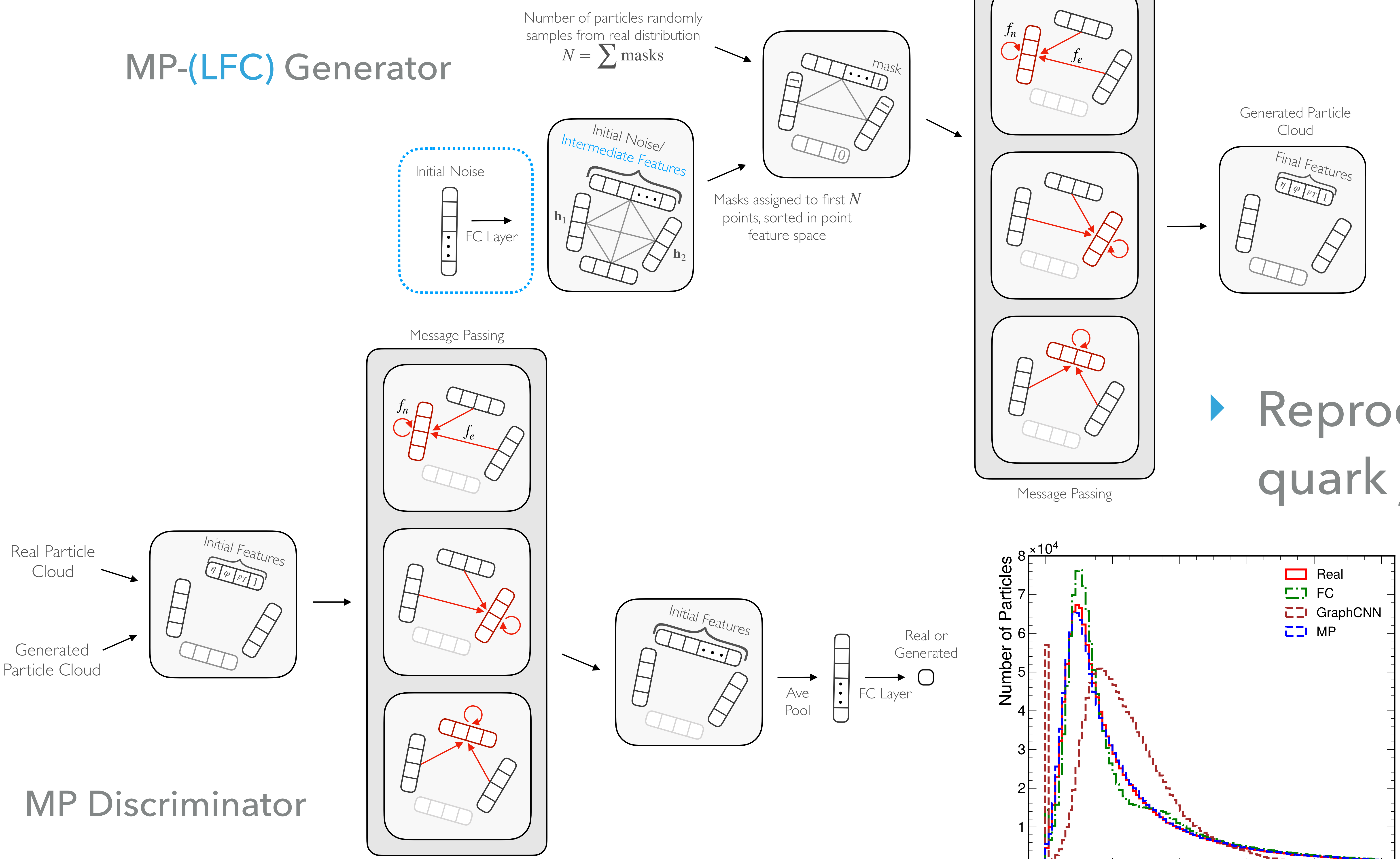
- As an alternative to voxelization, a graph-based GAN can be used to generate jets as particle clouds



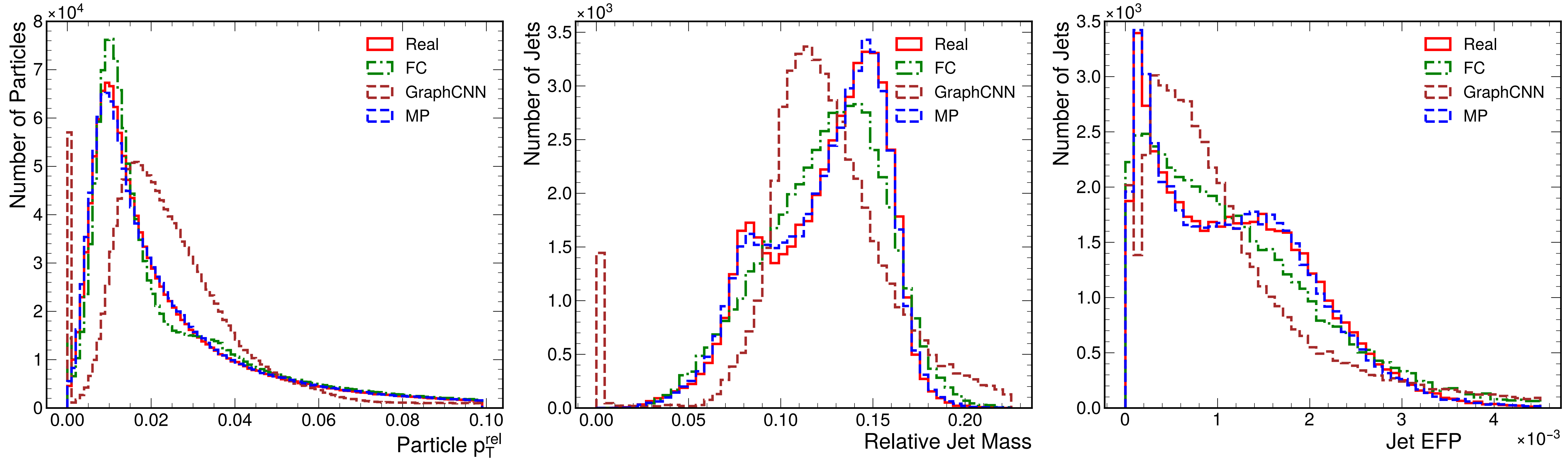
# GRAPH-BASED GAN

arXiv:2012.00173
arXiv:2106.11535

- As an alternative to voxelization, a graph-based GAN can be used to generate jets as particle clouds




- Reproduces nontrivial properties like top quark jet mass and energy-flow polynomials



# ML FOR TAGGING
# ML FOR GEN/SIM
# FAST ML FOR TRIGGER

# COMPUTING HARDWARE ALTERNATIVES



Image: Microsoft

# COMPUTING HARDWARE ALTERNATIVES




Image: [Microsoft]

# COMPUTING HARDWARE ALTERNATIVES




Image: [Microsoft]

# COMPUTING HARDWARE ALTERNATIVES



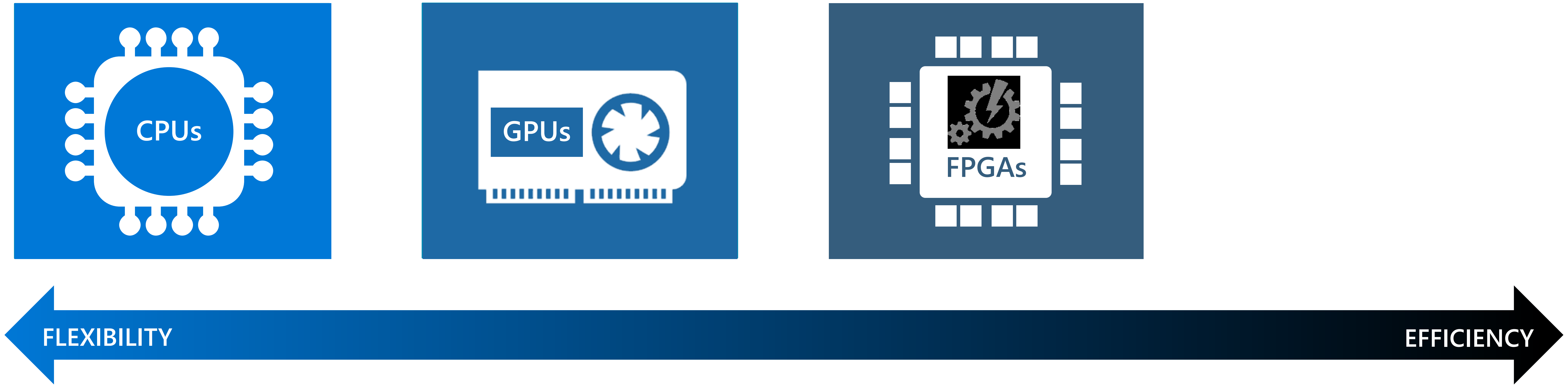



Image: [Microsoft]

# COMPUTING HARDWARE ALTERNATIVES



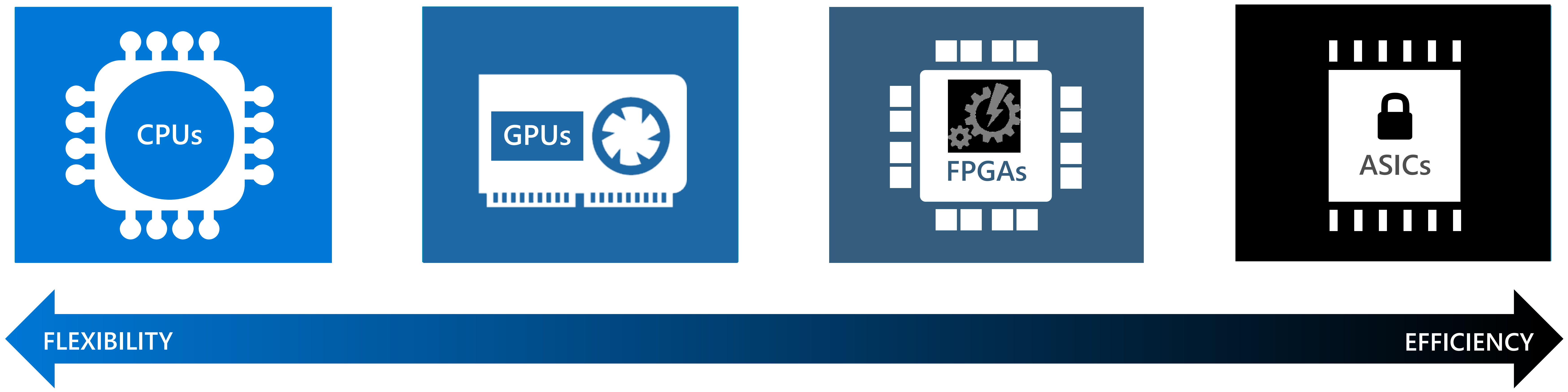



Image: Microsoft

# COMPUTING HARDWARE ALTERNATIVES

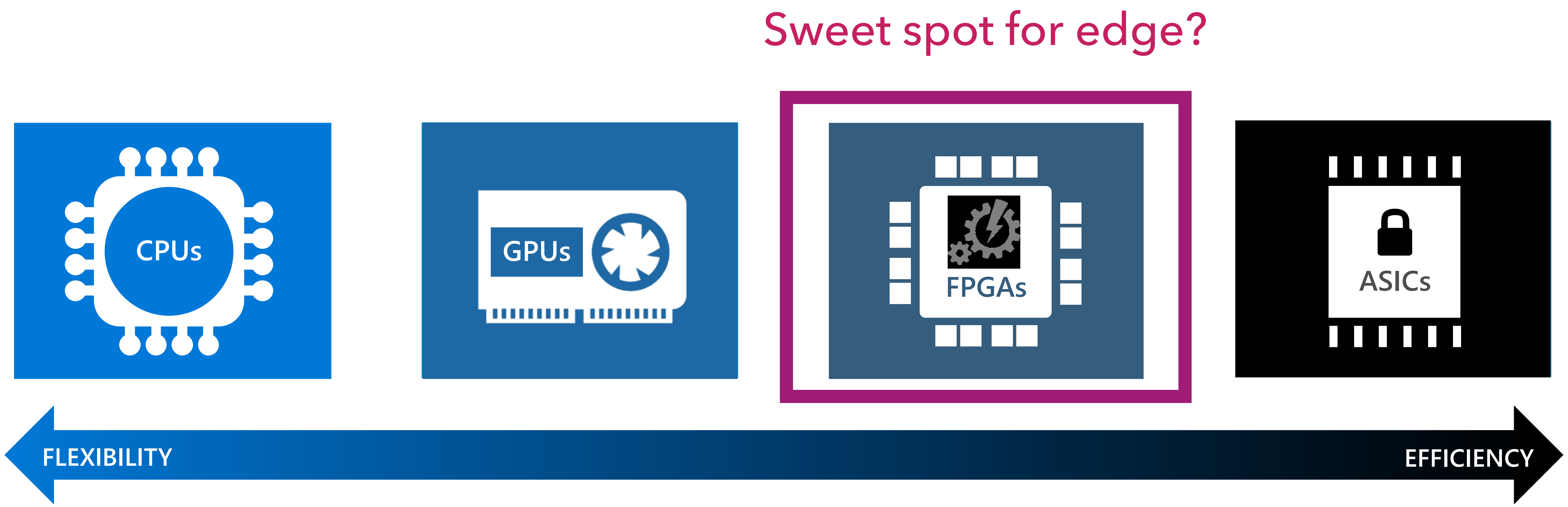


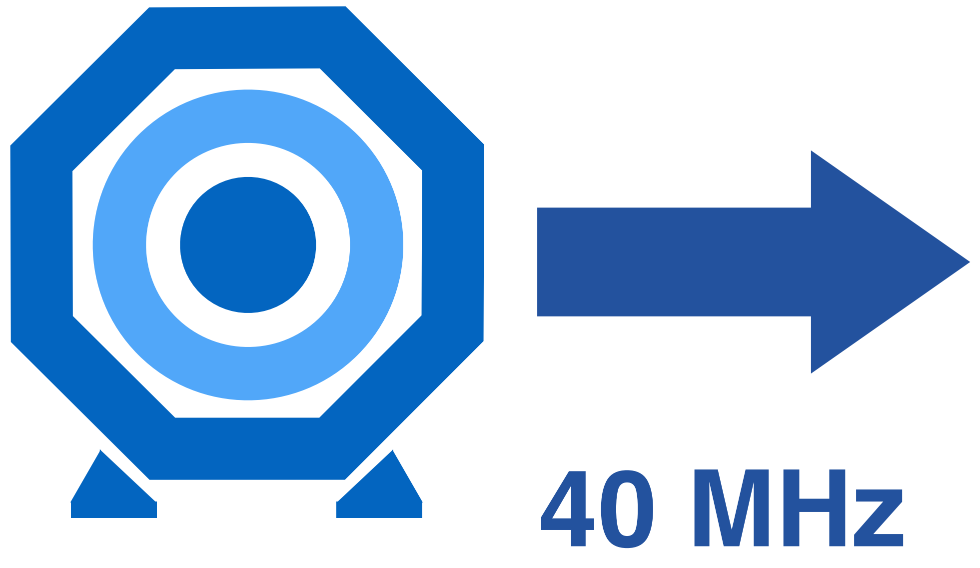

Image: [Microsoft]

# LHC EVENT PROCESSING



Compute Latency

1 ns    1 μs    1 ms    1 s



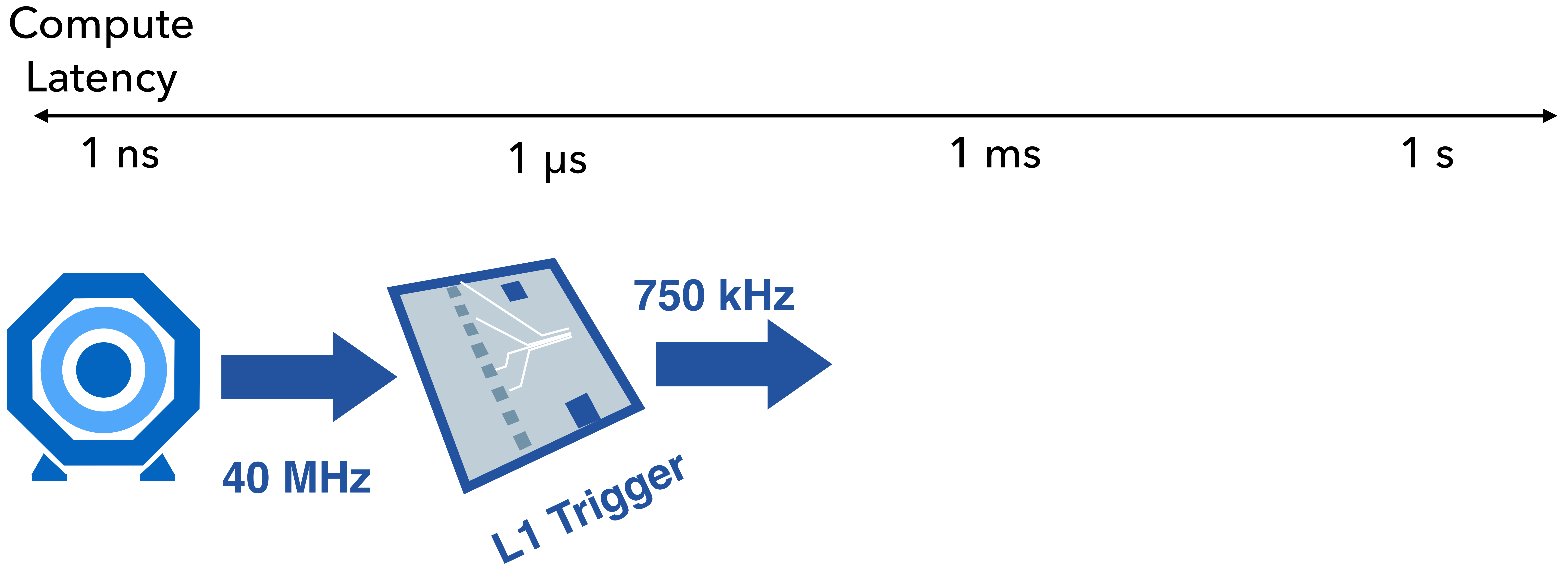

**Challenges:**

Each collision produces $O(10^3)$ particles

The detectors have $O(10^8)$ sensors

Extreme data rates of O(100 TB/s)

# LHC EVENT PROCESSING



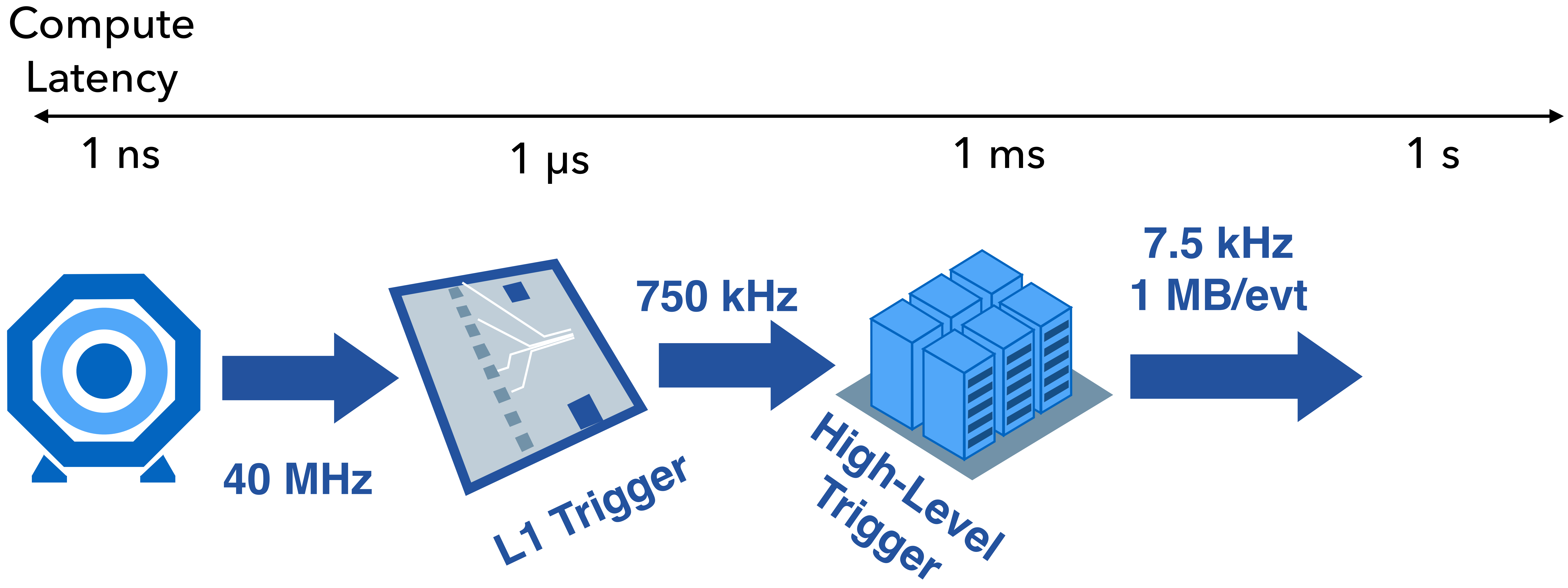

**Challenges:**

Each collision produces $O(10^3)$ particles

The detectors have $O(10^8)$ sensors

Extreme data rates of O(100 TB/s)

# LHC EVENT PROCESSING



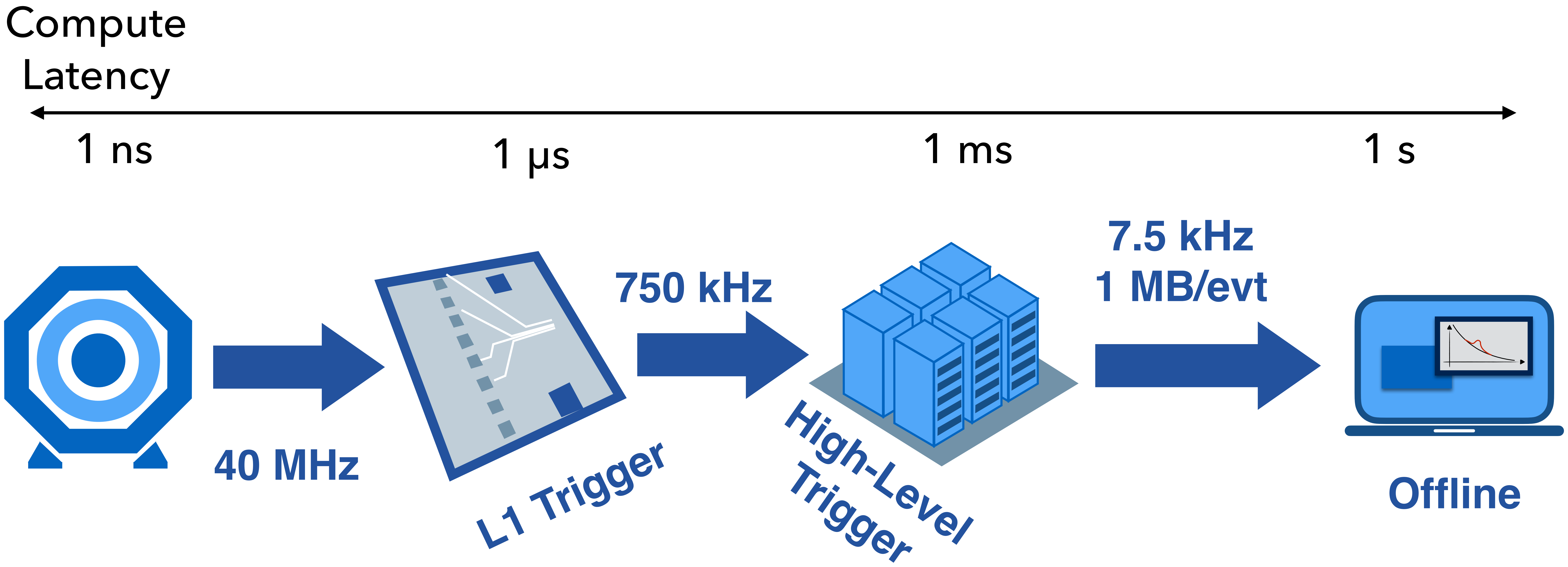

**Challenges:**

Each collision produces $O(10^3)$ particles

The detectors have $O(10^8)$ sensors

Extreme data rates of O(100 TB/s)

# LHC EVENT PROCESSING




**Challenges:**

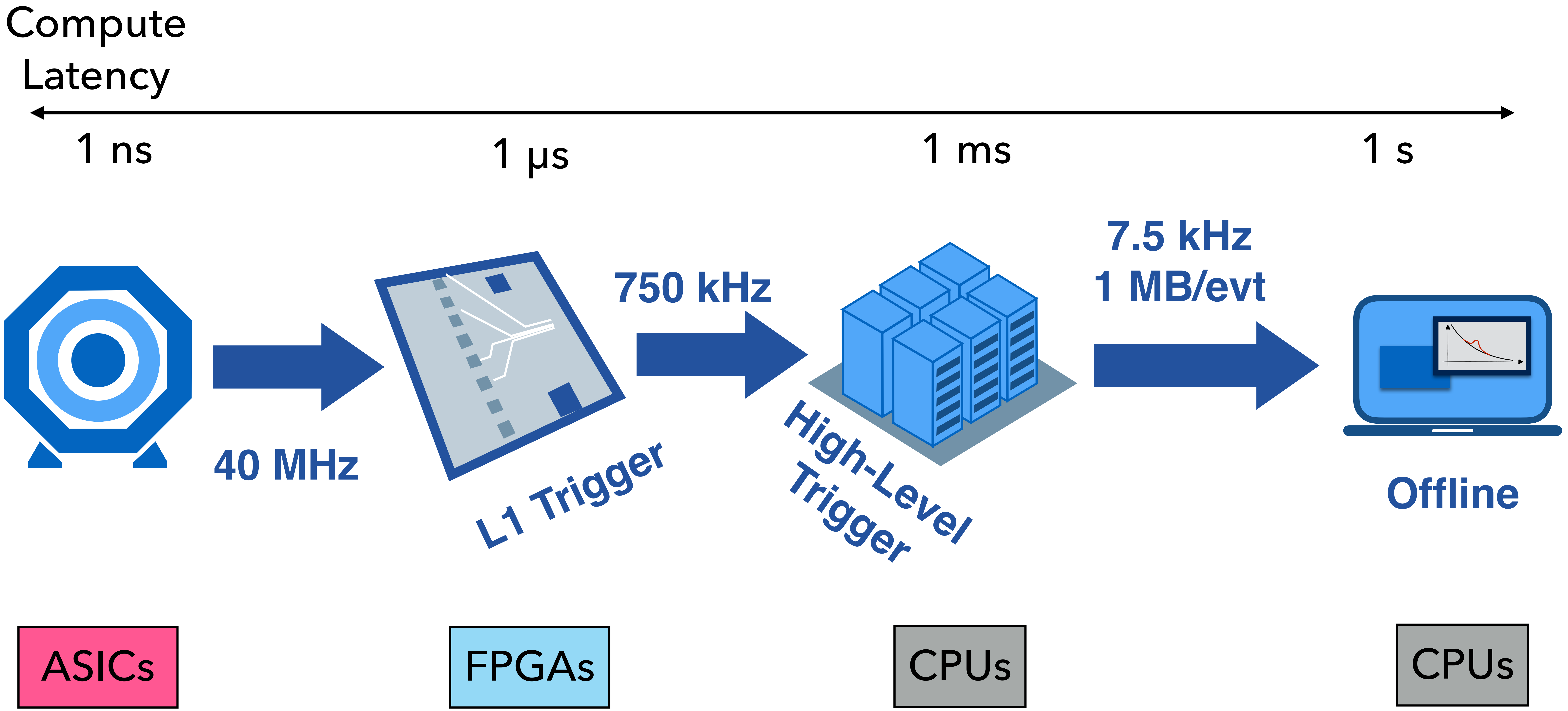
Each collision produces $O(10^3)$ particles

The detectors have $O(10^8)$ sensors

Extreme data rates of O(100 TB/s)

# LHC EVENT PROCESSING





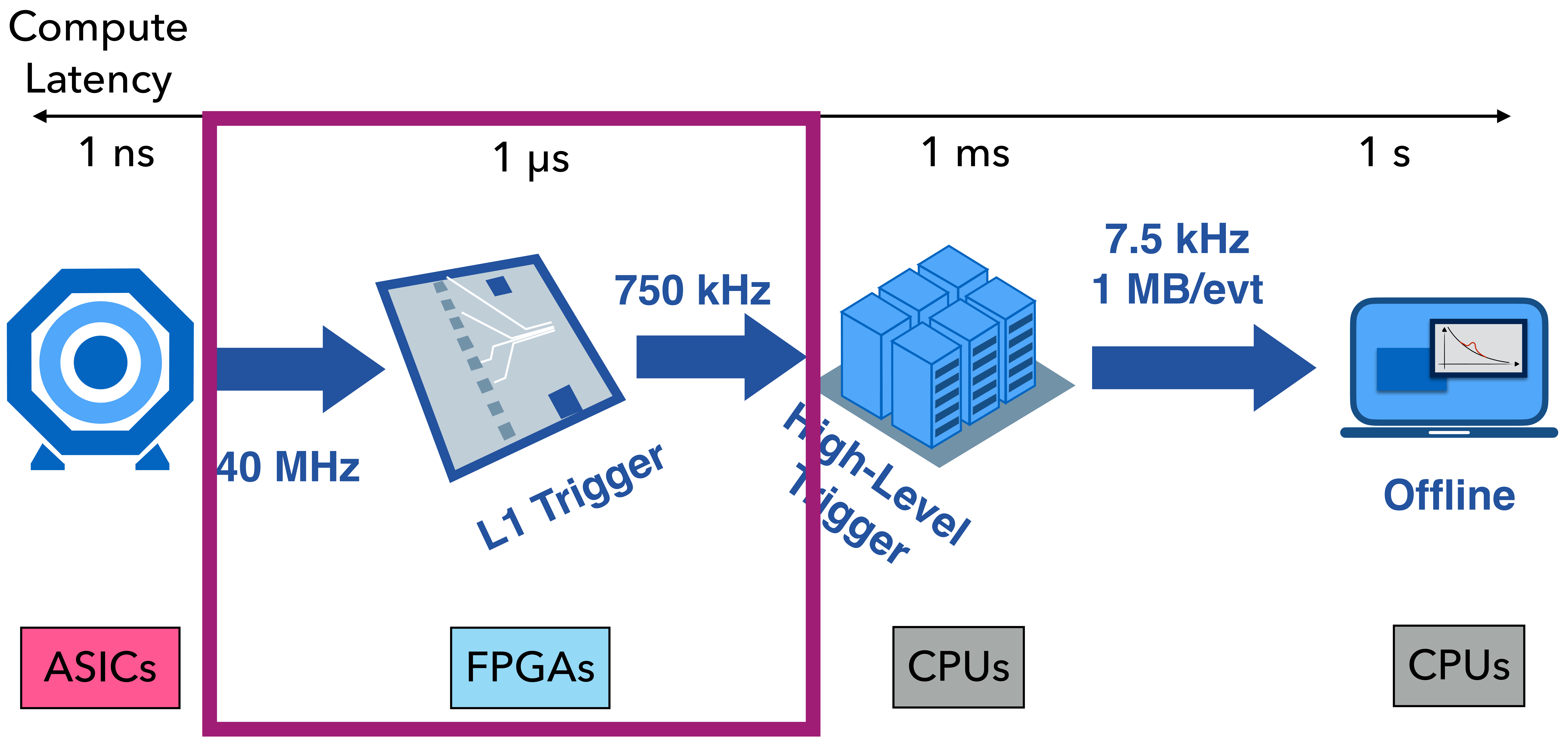

**Challenges:**

Each collision produces $O(10^3)$ particles

The detectors have $O(10^8)$ sensors

Extreme data rates of O(100 TB/s)

# LHC EVENT PROCESSING




**Challenges:**

Each collision produces $O(10^3)$ particles

The detectors have $O(10^8)$ sensors

Extreme data rates of O(100 TB/s)

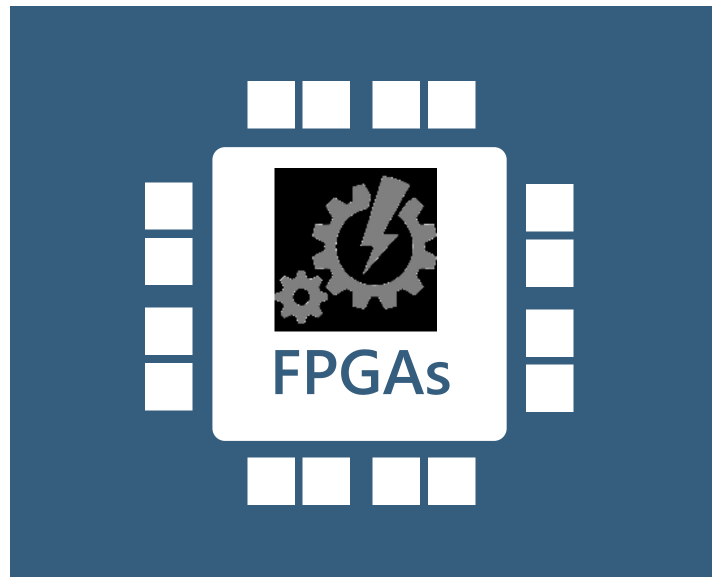
# PROGRAMMING HARDWARE (FPGAS)



- Say you want to program an "adder" function on an FPGA

```
module adder(
    input  wire [4:0] a,
    input  wire [4:0] b,
    output wire [4:0] y
);
    assign y = a + b;

endmodule
```

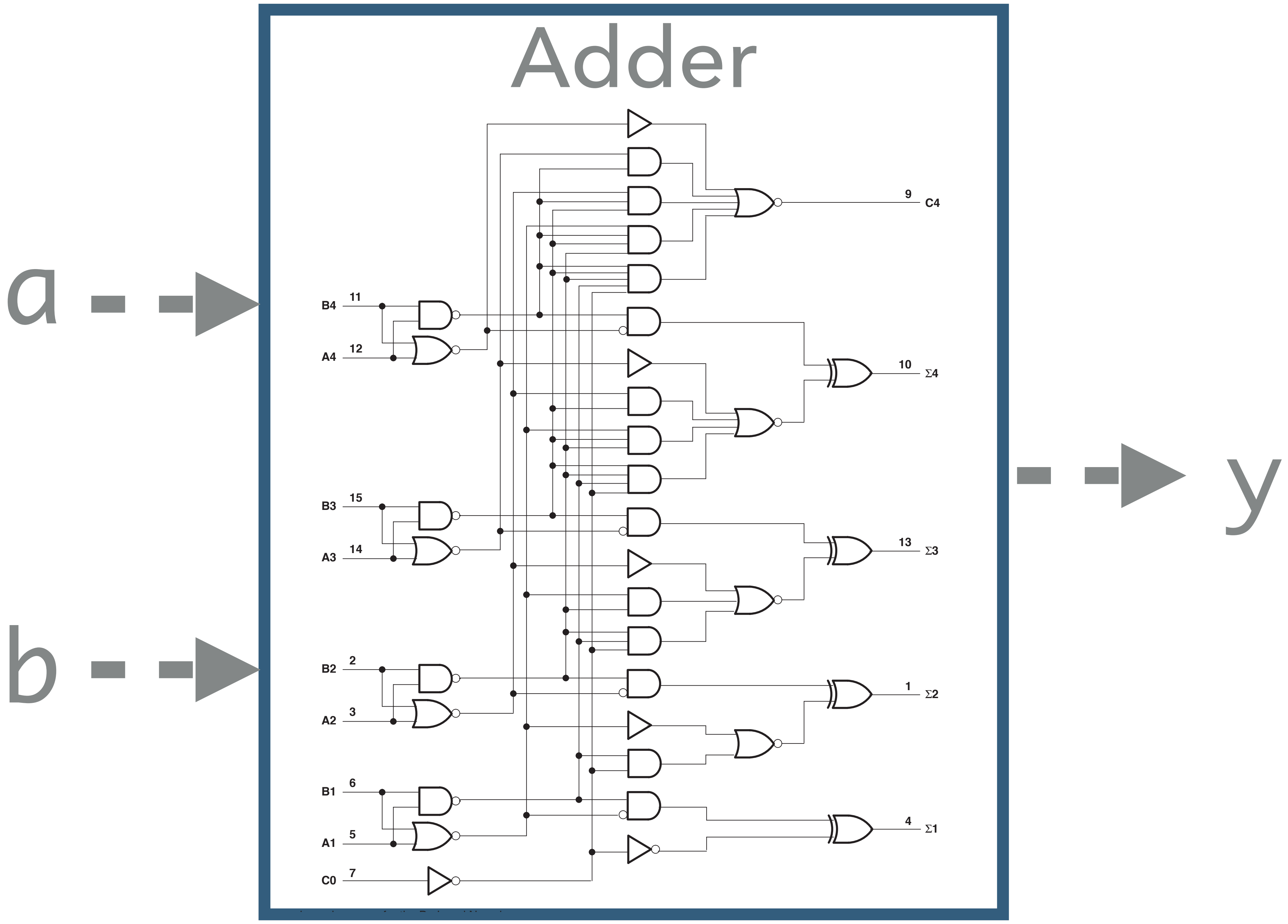



- Register transfer-level (RTL) code is "synthesized" into gates

- Say you want to program an "adder" function on an FPGA

```
module adder(
      input  wire [4:0] a,
      input  wire [4:0] b,
      output wire [4:0] y
);
      assign y = a + b;

endmodule
```



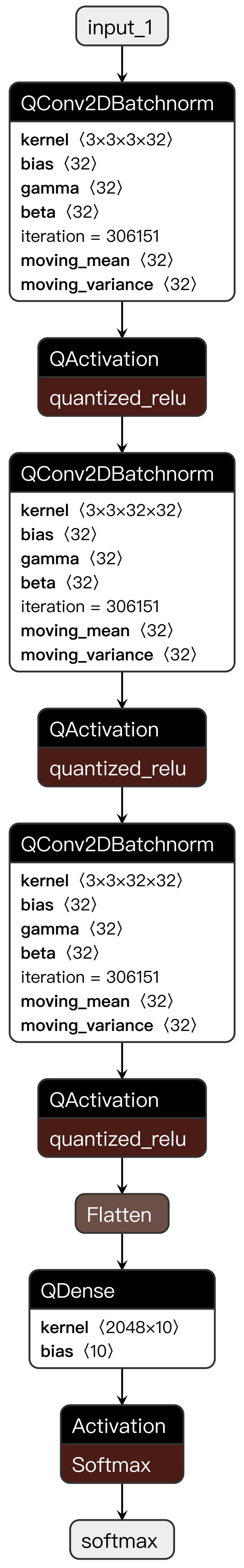

Synthesis

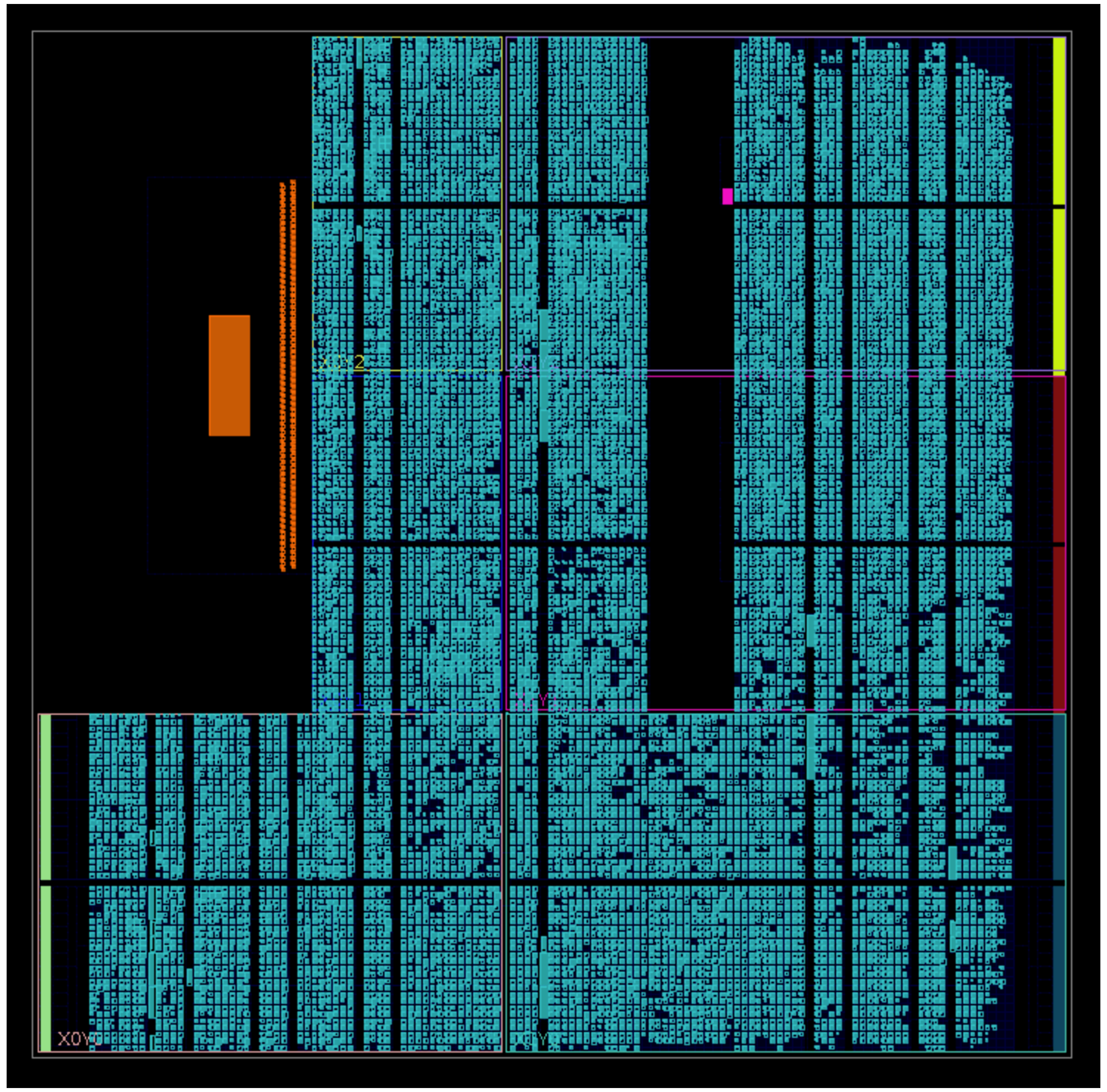
- Register transfer-level (RTL) code is "synthesized" into gates

# PROGRAMMING HARDWARE (FPGAS)



- What if instead we specify an AI model



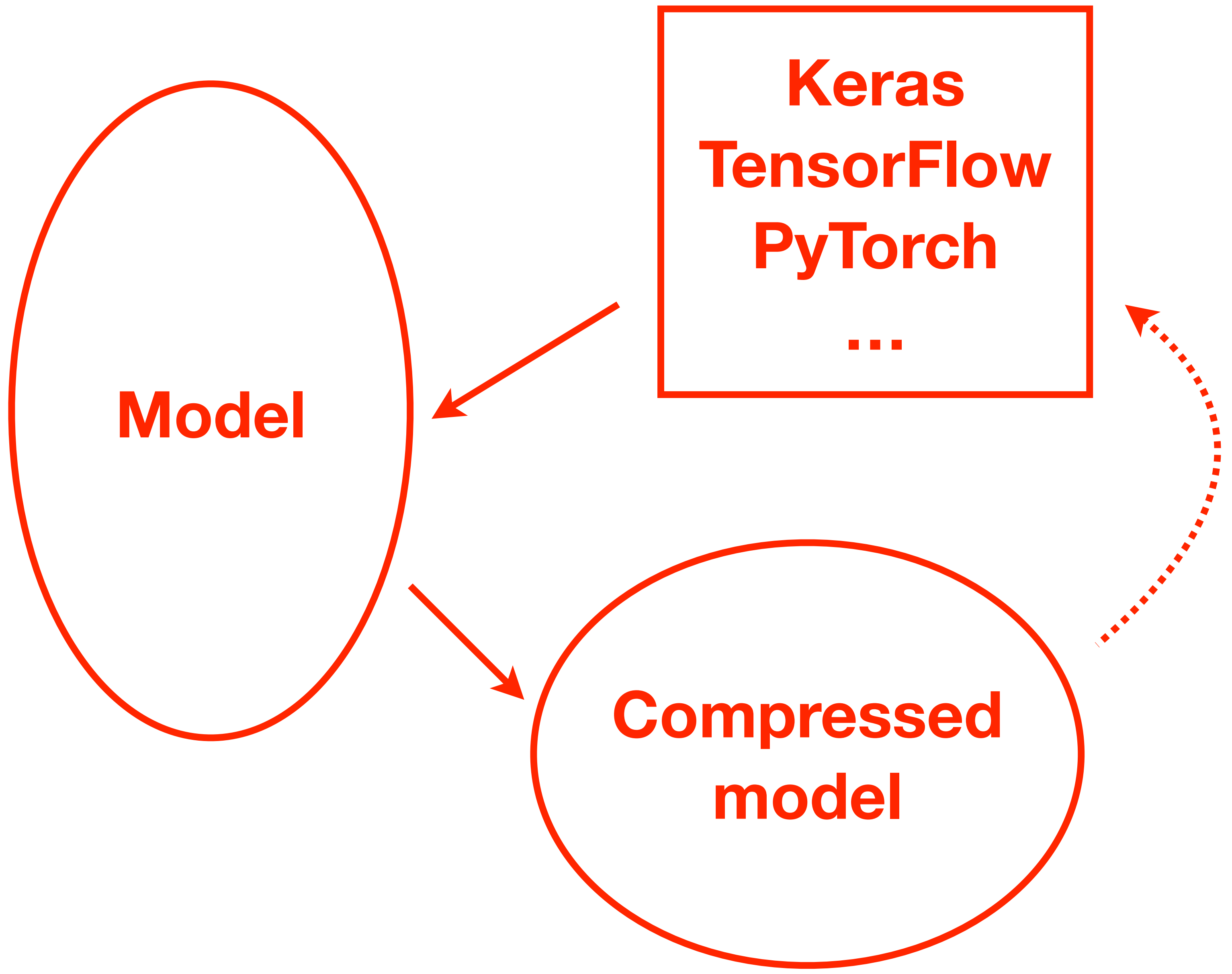



High-Level Synthesis

# DESIGN EXPLORATION WITH HLS4ML



- hls4ml for scientists or ML experts to translate ML algorithms into RTL firmware

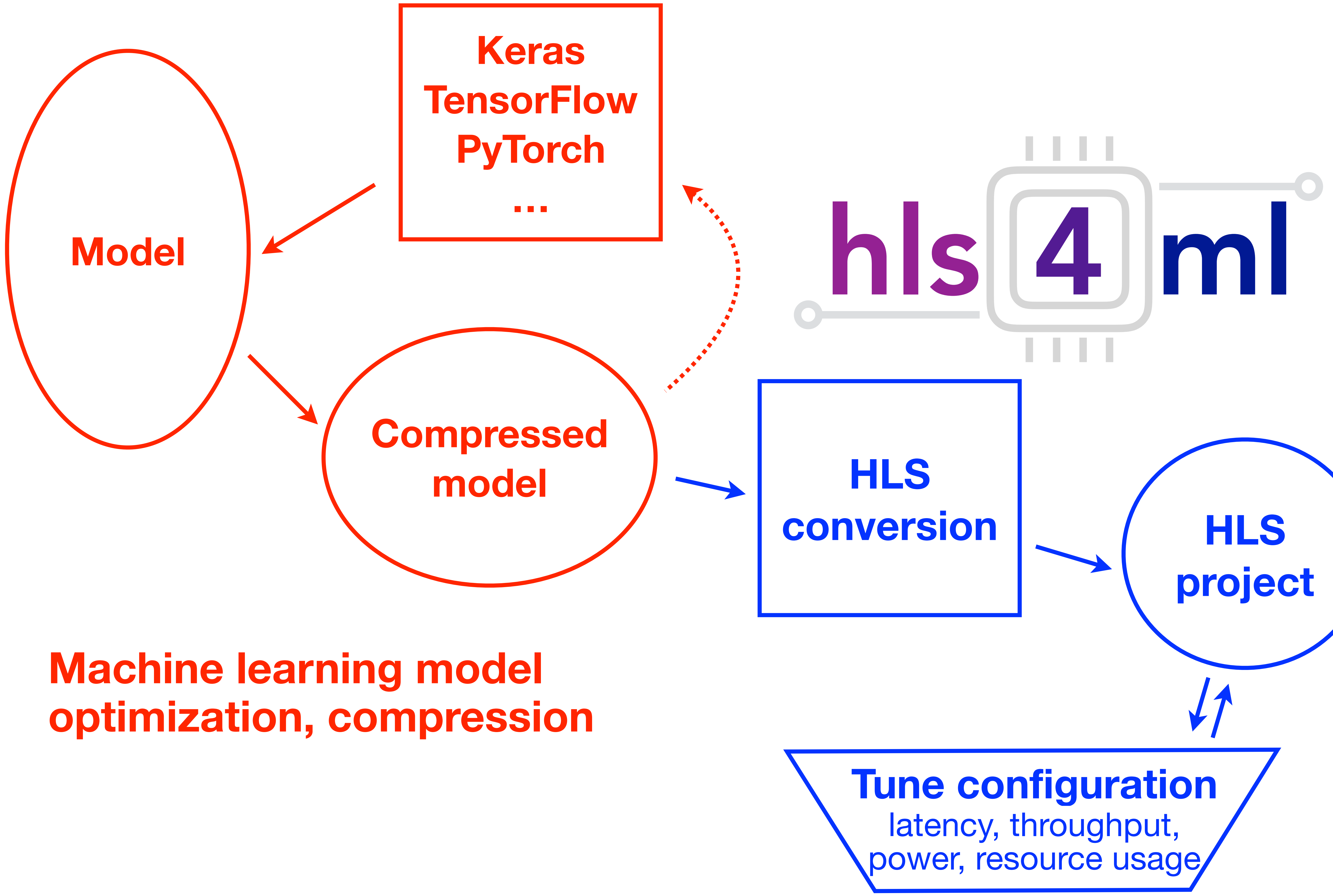



Machine learning model optimization, compression

# DESIGN EXPLORATION WITH HLS4ML

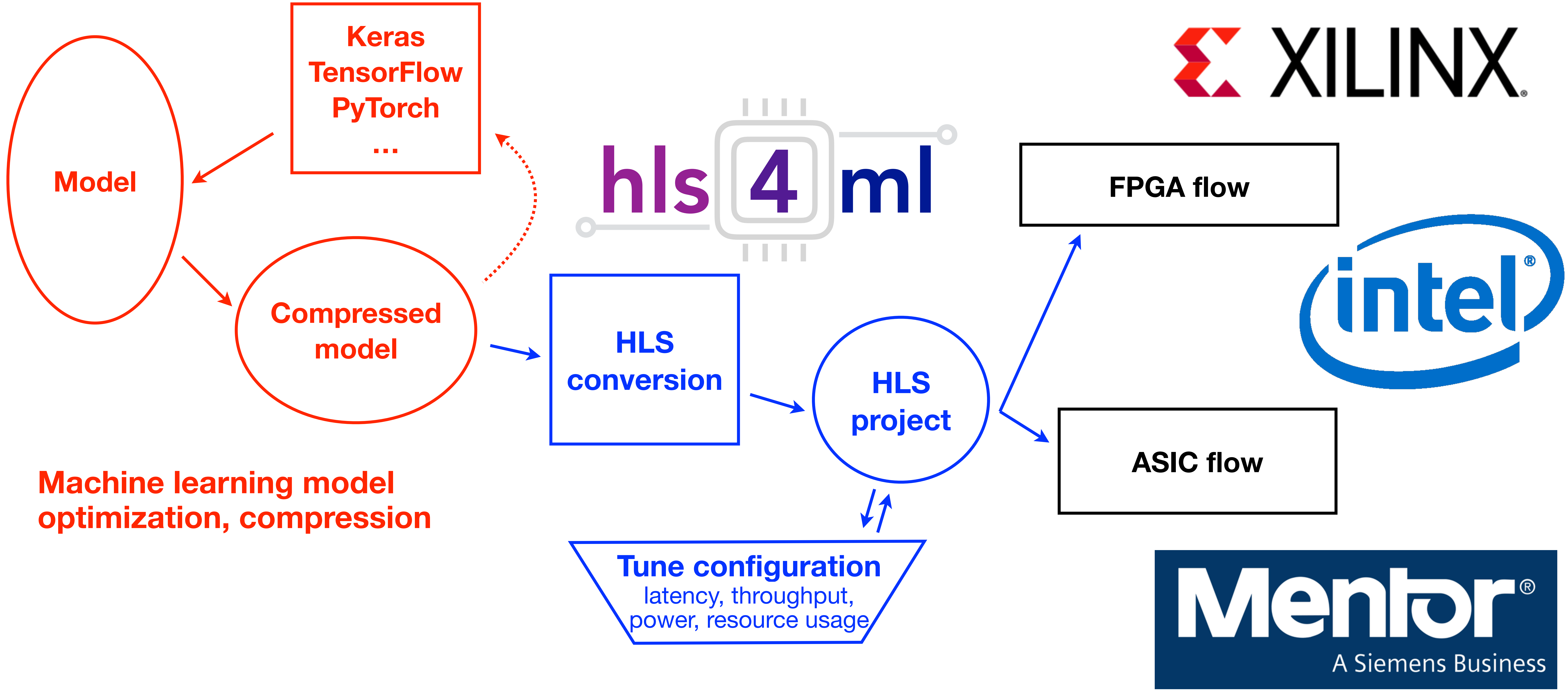


- hls4ml for scientists or ML experts to translate ML algorithms into RTL firmware



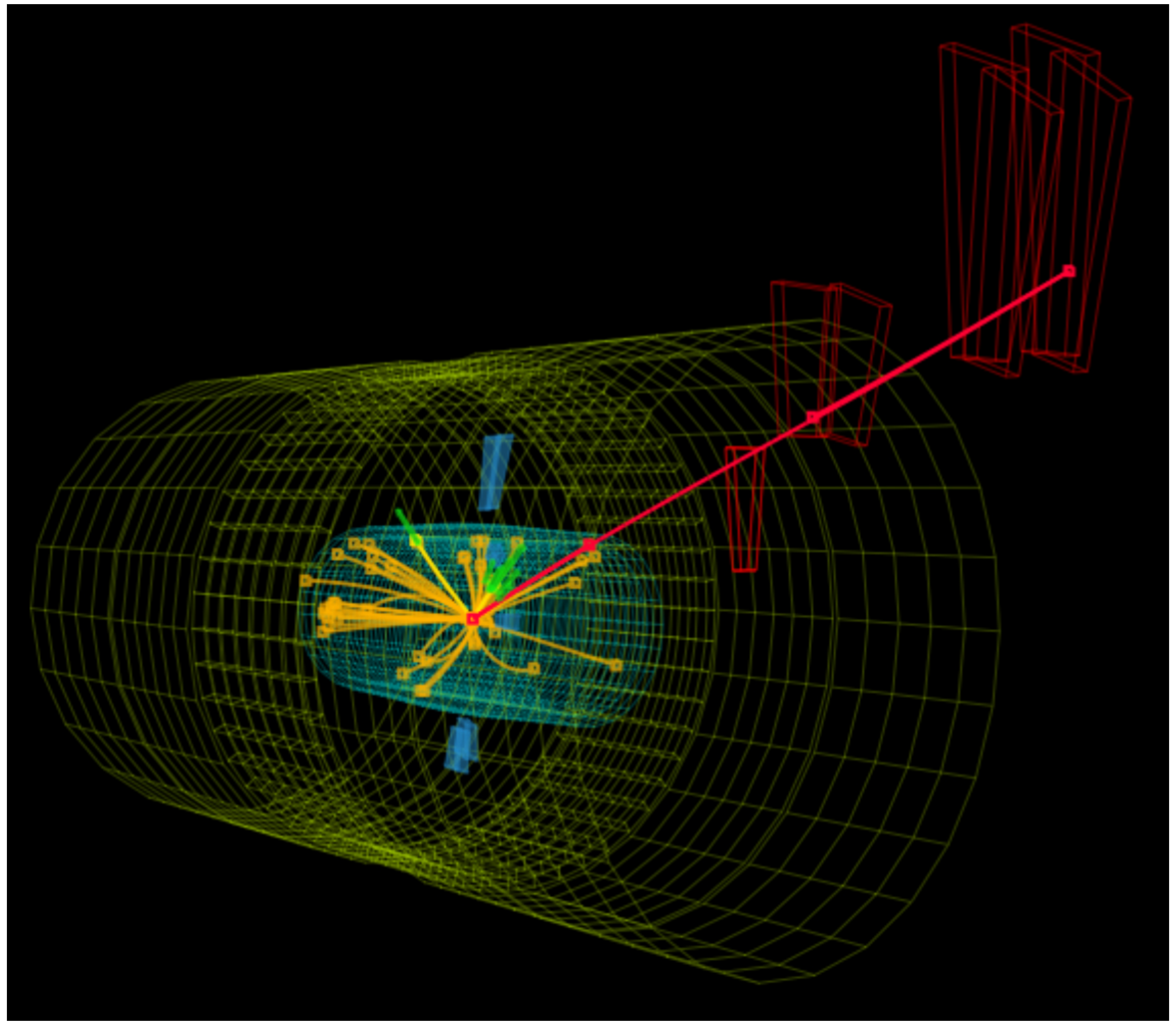
# DESIGN EXPLORATION WITH HLS4ML

- hls4ml for scientists or ML experts to translate ML algorithms into RTL firmware



# EXAMPLE: ENDCAP MUON TRACK FINDER UPGRADE (EMTF++)

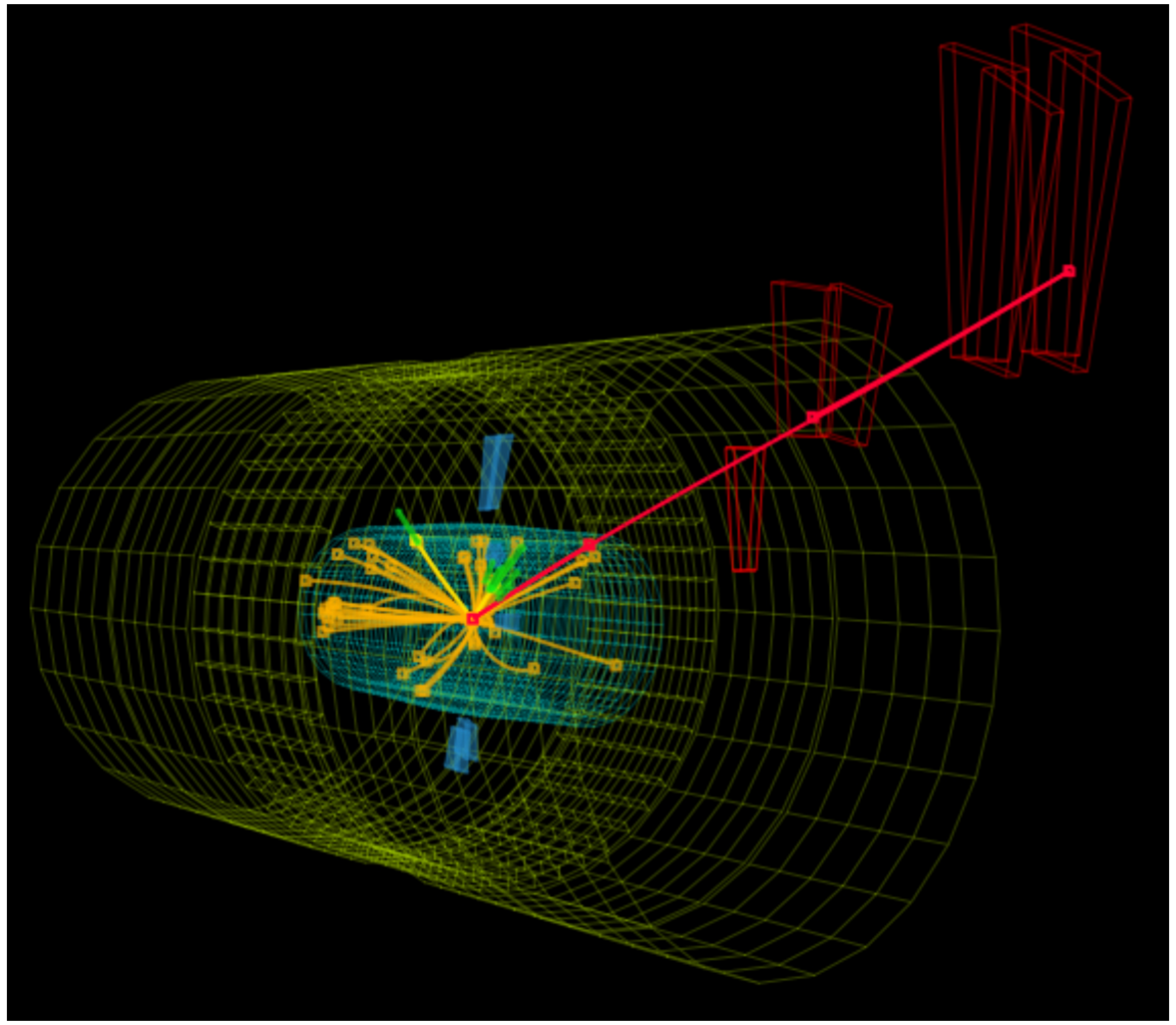
CMS-TDR-021



# EXAMPLE: ENDCAP MUON TRACK FINDER UPGRADE (EMTF++)

CMS-TDR-021

- Goal: determine muon $p_T$ in endcap based on info. available in the L1 trigger

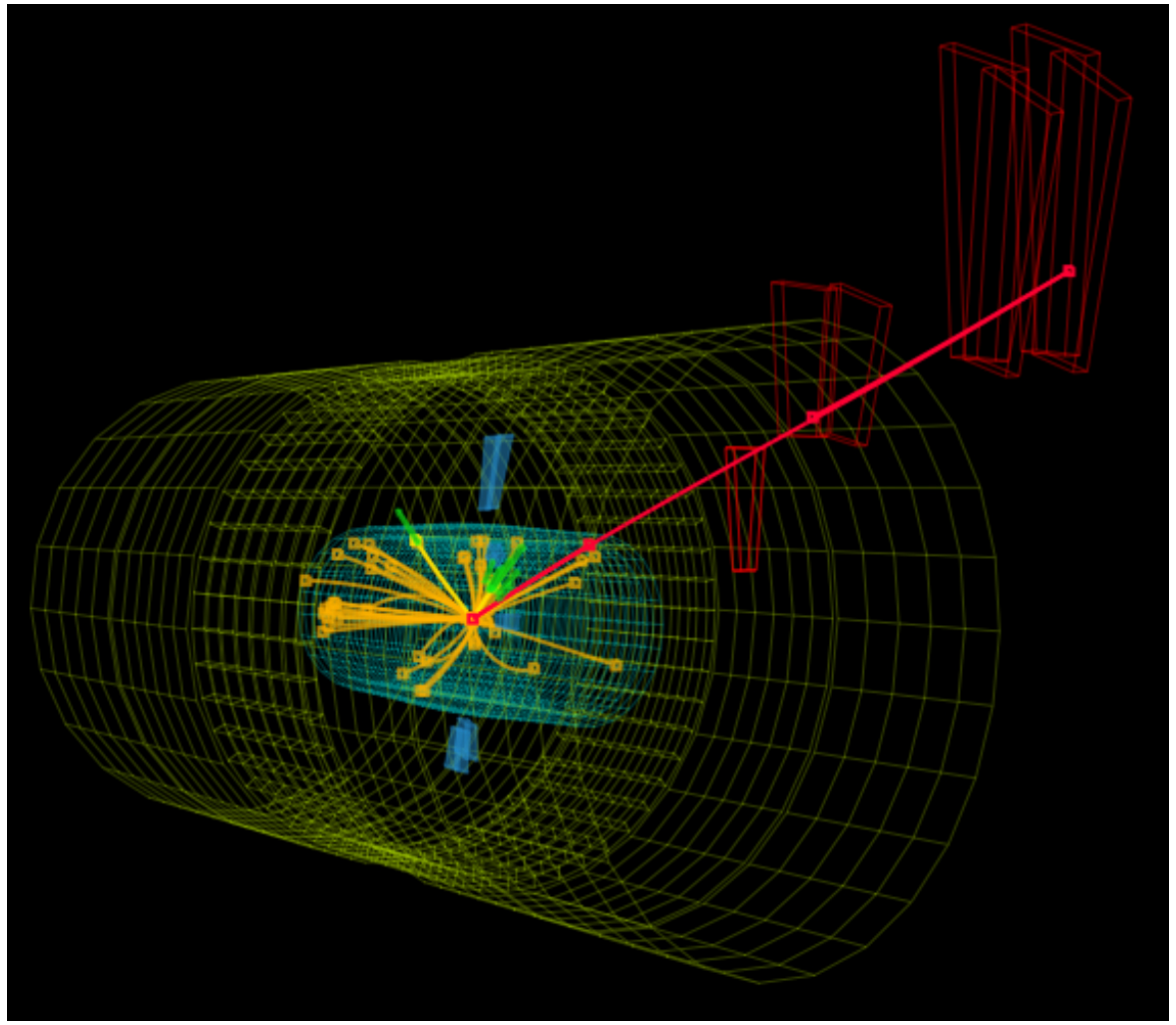


# EXAMPLE: ENDCAP MUON TRACK FINDER UPGRADE (EMTF++)

CMS-TDR-021



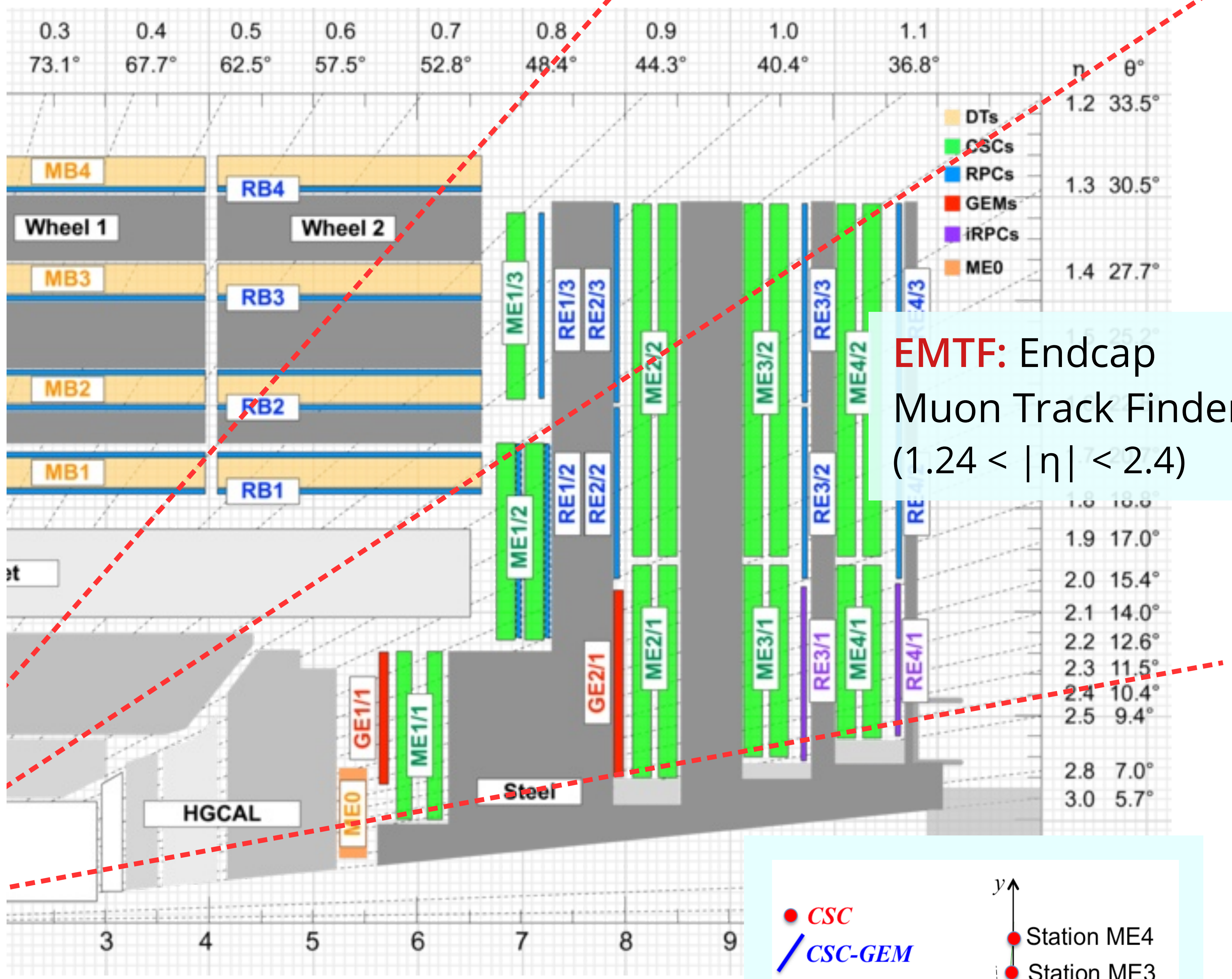
- Goal: determine muon $p_T$ in endcap based on info. available in the L1 trigger



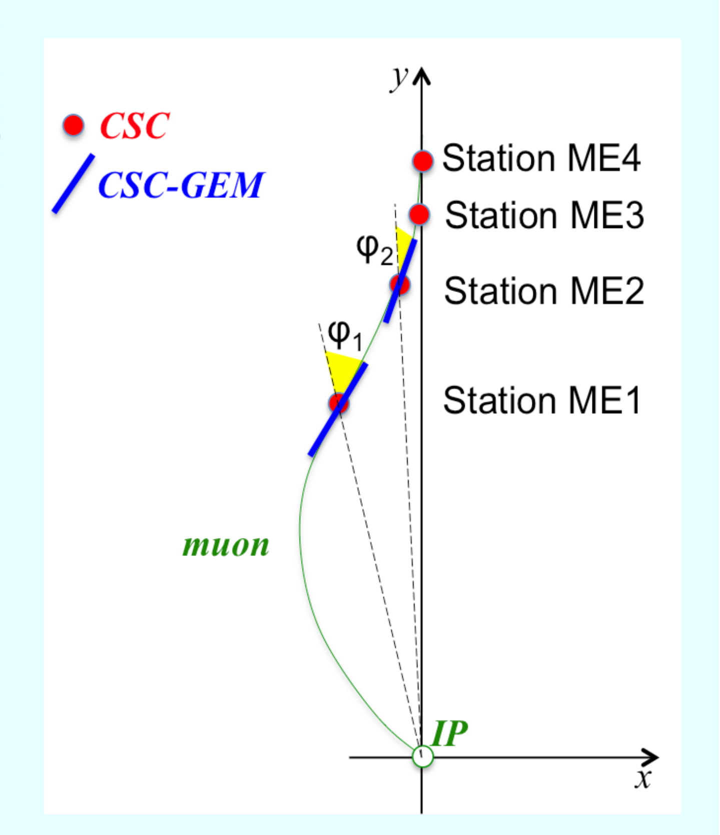
BMTF: Barrel Muon Track Finder ($0 < |\eta| < 0.83$)

OMTF: Overlap Muon Track Finder ($0.83 < |\eta| < 1.24$)

EMTF: Endcap Muon Track Finder ($1.24 < |\eta| < 2.4$)



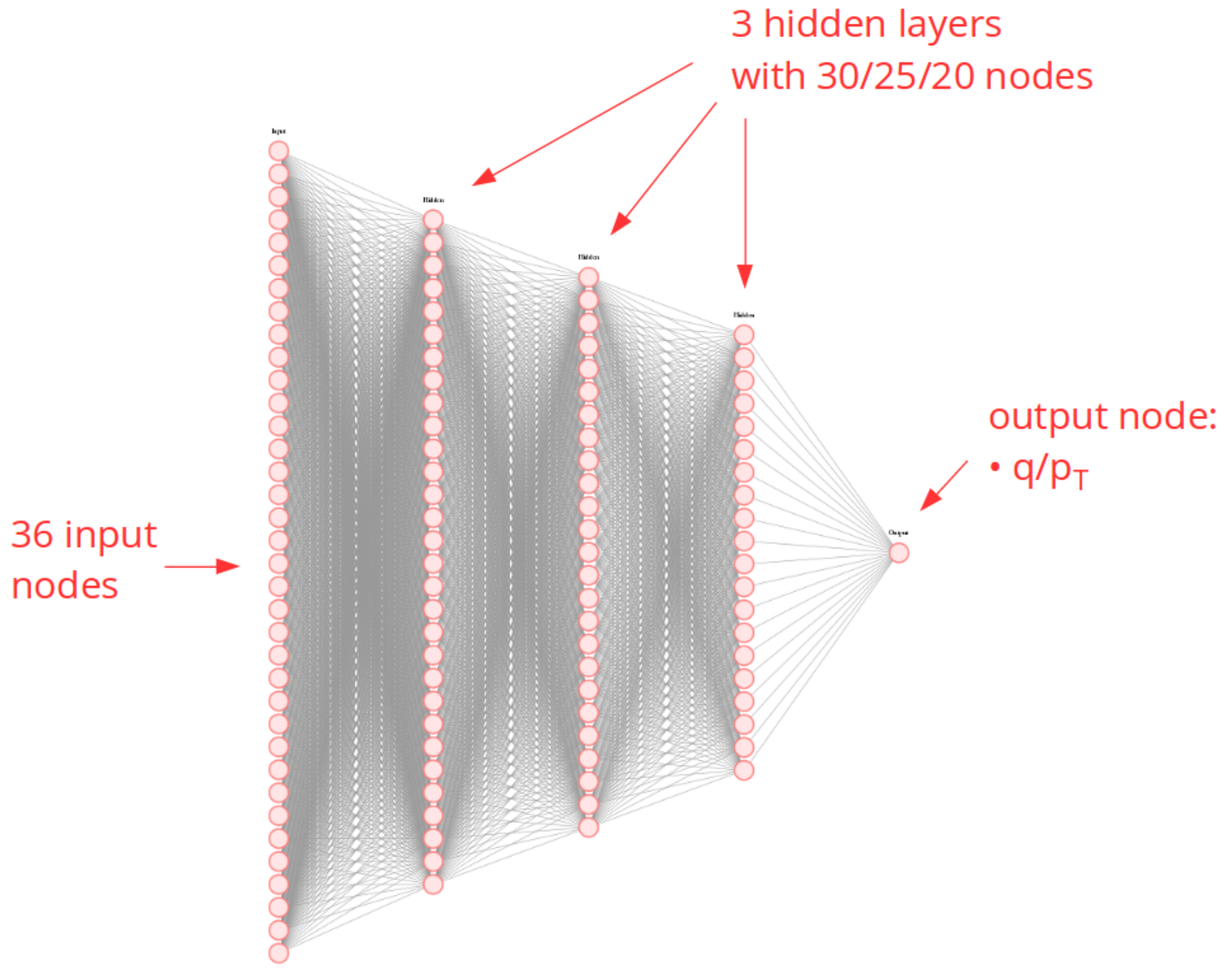

- Challenges:
  - Non-uniform magnetic field with little bending
  - Large background from multiple sources
- EMTF++ has to evolve to
  - Incorporate new muon detectors
  - Improve efficiency, redundancy, $p_T$ resolution, timing
  - Maintain the same trigger threshold at higher pileup




GEM-CSC bending angle improves $p_T$ resolution and reduce trigger rate.

# EMTF++ NETWORK AND PERFORMANCE

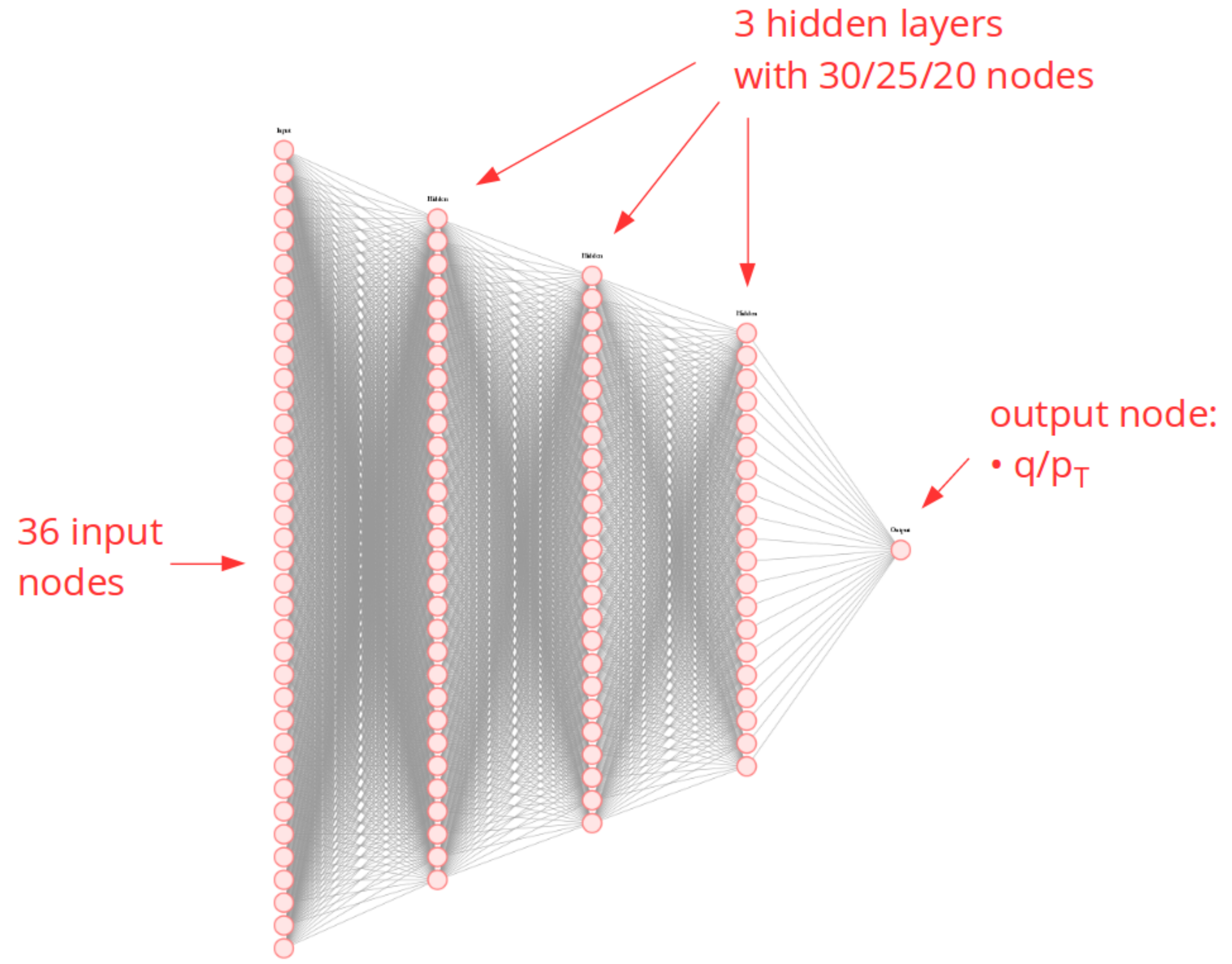
CMS-TDR-021




| | ME1/1 | ME1/2 | ME2 | ME3 | ME4 | RE1 | RE2 | RE3 | RE4 | GE1/1 | GE2/1 | ME0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ϕ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| θ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| bend | ✔ | ✔ | ✔ | ✔ | ✔ | | | | | | | ✔ |
| quality | ✔ | ✔ | ✔ | ✔ | ✔ | | | | | | | ✔ |
| time | | | | | | | | | | | | |

# EMTF++ NETWORK AND PERFORMANCE



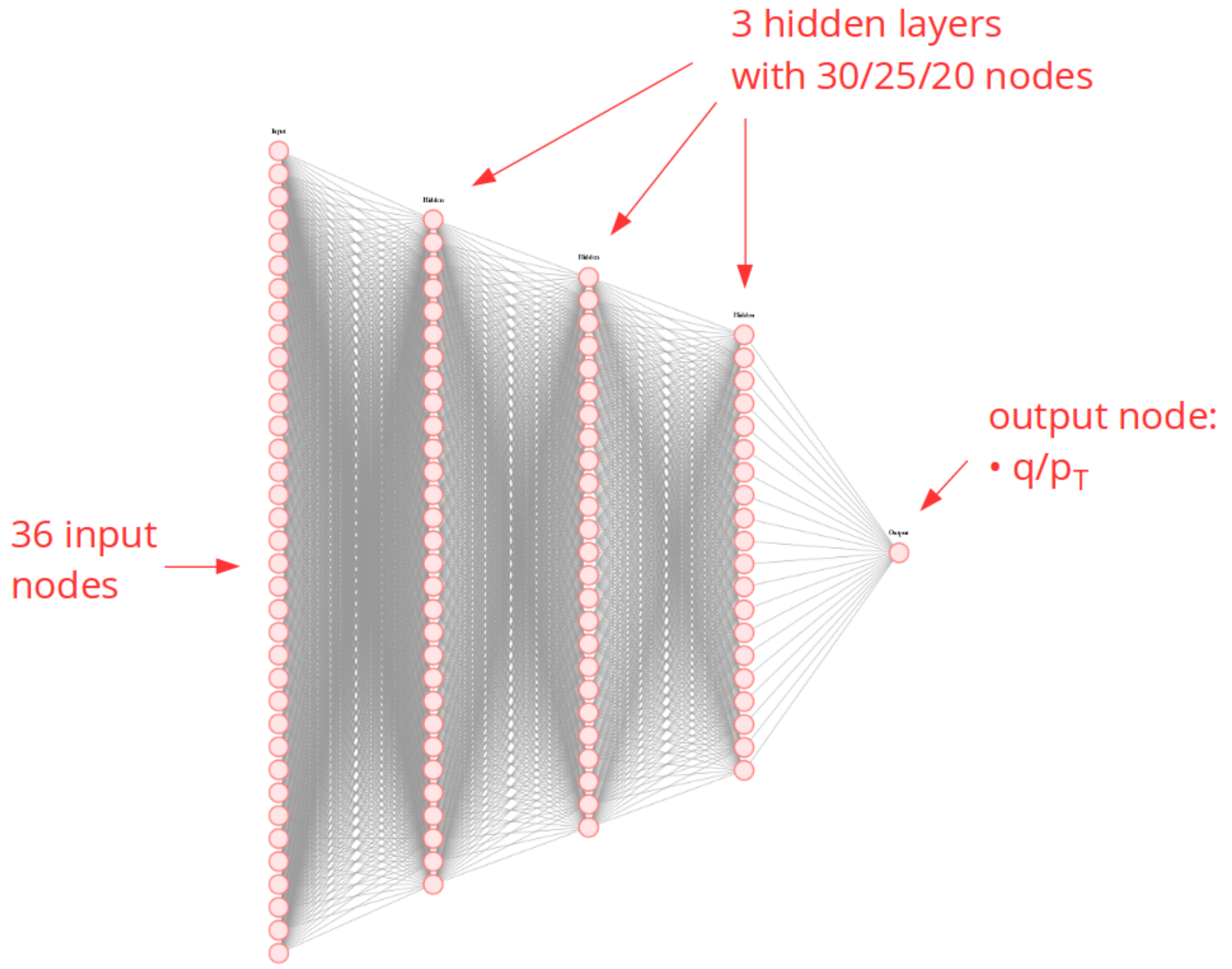



| | ME1/1 | ME1/2 | ME2 | ME3 | ME4 | RE1 | RE2 | RE3 | RE4 | GE1/1 | GE2/1 | ME0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ϕ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| θ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| bend | ✔ | ✔ | ✔ | ✔ | ✔ | | | | | | | ✔ |
| quality | ✔ | ✔ | ✔ | ✔ | ✔ | | | | | | | ✔ |
| time | | | | | | | | | | | | |

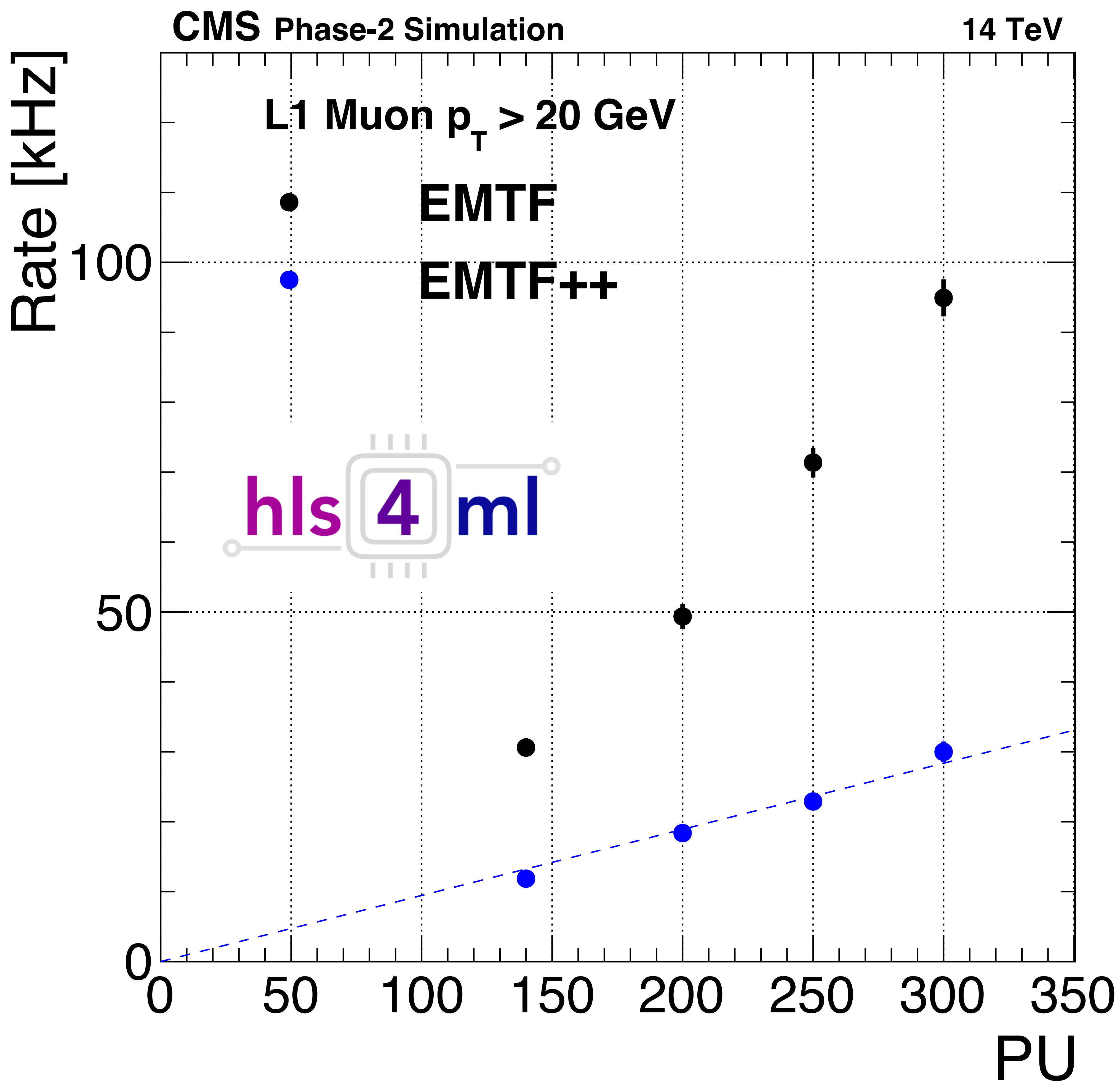
- NN regresses muon $p_T$ based on 36 inputs

# EMTF++ NETWORK AND PERFORMANCE

CMS-TDR-021




| | ME1/1 | ME1/2 | ME2 | ME3 | ME4 | RE1 | RE2 | RE3 | RE4 | GE1/1 | GE2/1 | ME0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ϕ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| θ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| bend | ✔ | ✔ | ✔ | ✔ | ✔ | | | | | | | ✔ |
| quality | ✔ | ✔ | ✔ | ✔ | ✔ | | | | | | | ✔ |
| time | | | | | | | | | | | | |

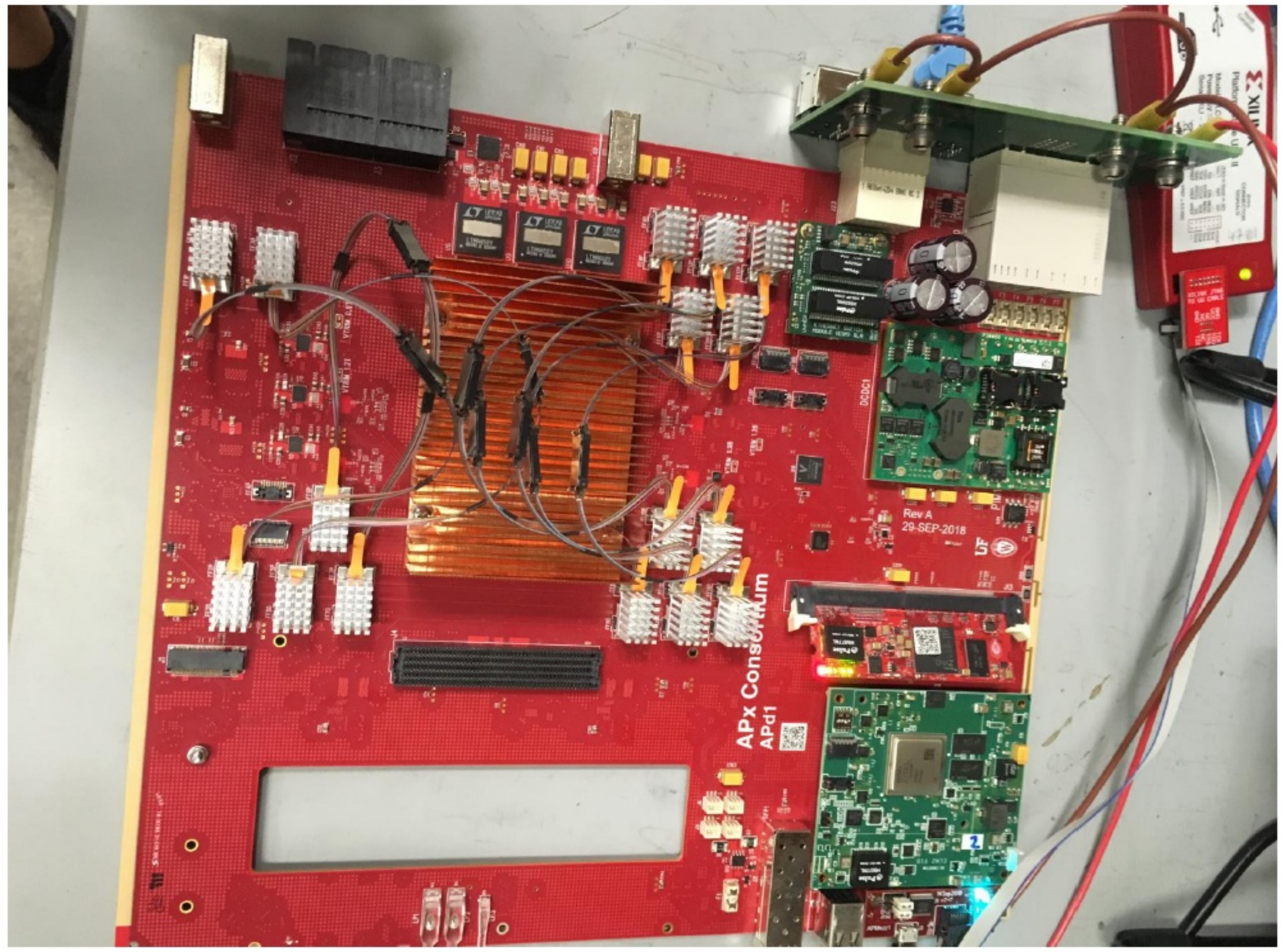


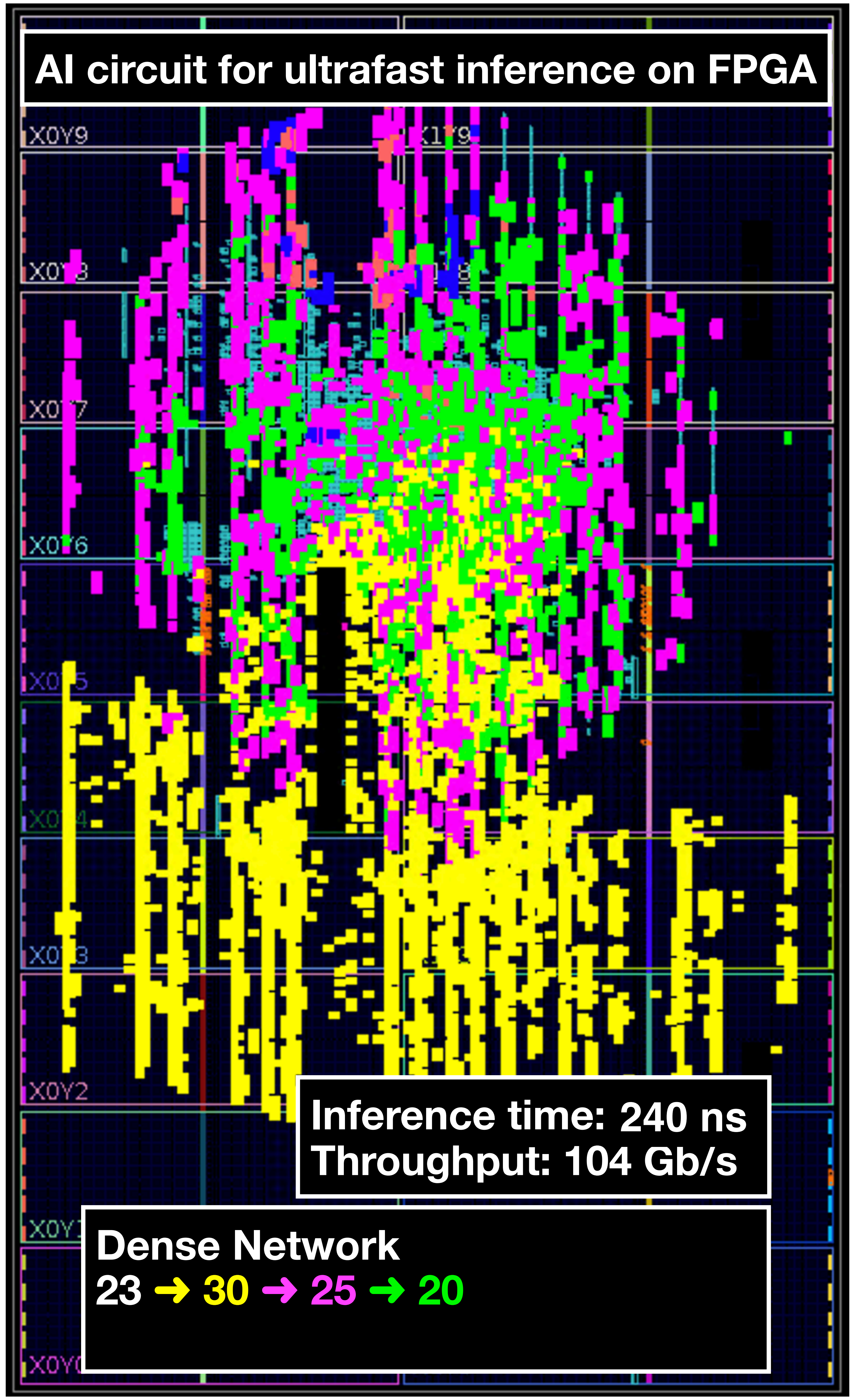

- NN regresses muon $p_T$ based on 36 inputs
- 3× reduction in the trigger rate for NN!

# EMTF++ FPGA IMPLEMENTATION



| Algorithm (target FPGA) | LUT | Flip-flop | Block RAM | DSP |
|---|---|---|---|---|
| NN + EMTF (VU9P) | 28% | 8% | 30% | 30% |






- Fits within L1 trigger latency (240 ns!) and FPGA resource requirements (less then 30%)

# SUMMARY AND OUTLOOK

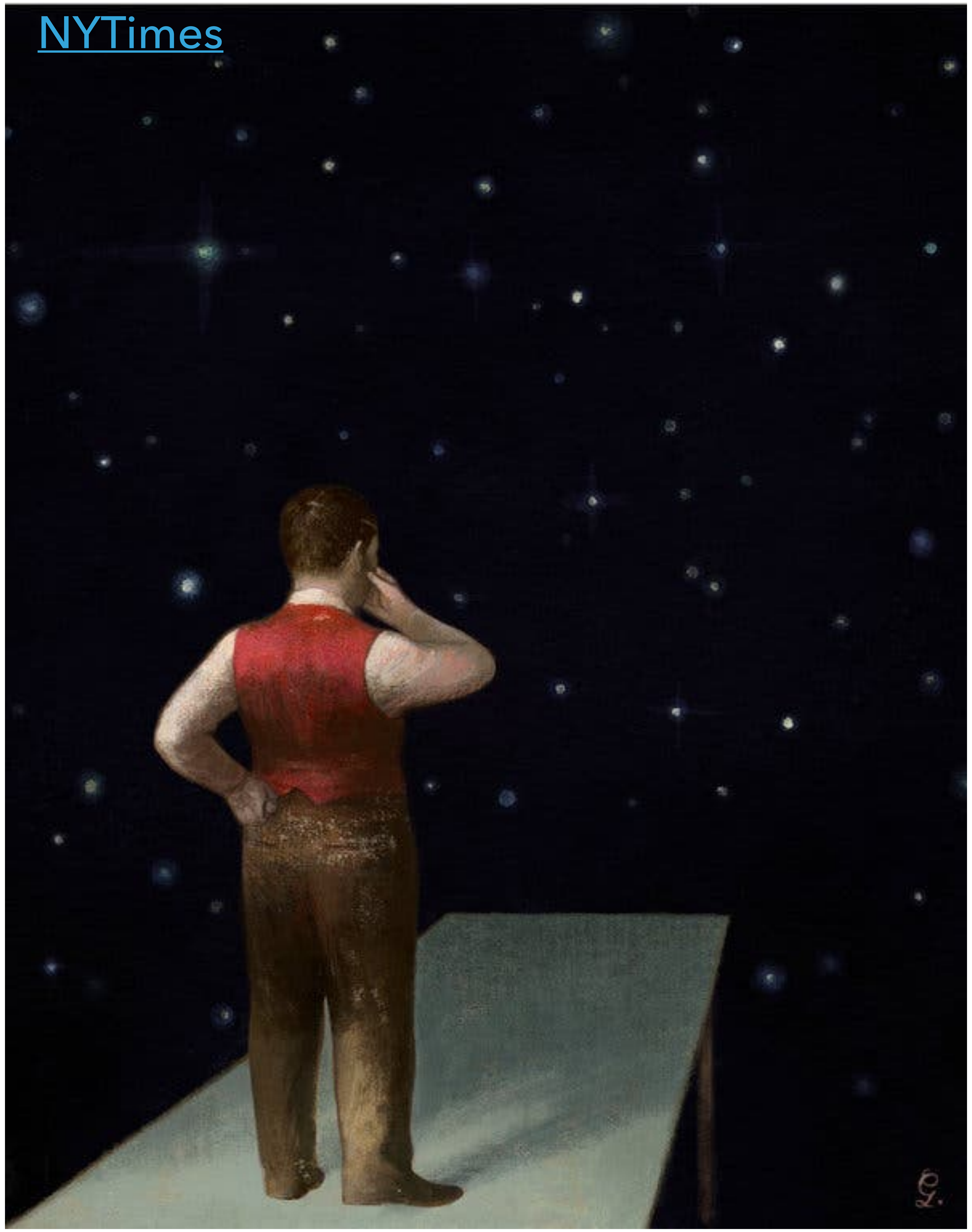
# SUMMARY AND OUTLOOK



- Modern ML is the latest tool in the arsenal of HEP that has a wide range of applications
  - Jet tagging/regression, event reconstruction, anomaly detection, trigger, data compression, generation/simulation
- We have only scratched the surface of what is possible in the future with ML
  - Improvements in physics sensitivity, detector design, automatic calibrations, reducing time/cost of data analysis

# SUMMARY AND OUTLOOK

- Modern ML is the latest tool in the arsenal of HEP that has a wide range of applications
  - Jet tagging/regression, event reconstruction, anomaly detection, trigger, data compression, generation/simulation
- We have only scratched the surface of what is possible in the future with ML
  - Improvements in physics sensitivity, detector design, automatic calibrations, reducing time/cost of data analysis
- With upcoming data at the LHC and beyond, we will explore the **edge** of the unknown in particle physics with cutting-edge ML



JAVIER DUARTE

LISHEP SESSION C

JULY 6, 2021

# BACKUP