Jet Finding as a Real-Time Object Detection Task

Leon Bozianu on behalf of the ATLAS collaboration

MARIE CURIE ACTIONS

REAL-TIME ANALYSIS FOR SCIENCE AND INDUSTRY

5.11.2024 - ML4Jets 2024

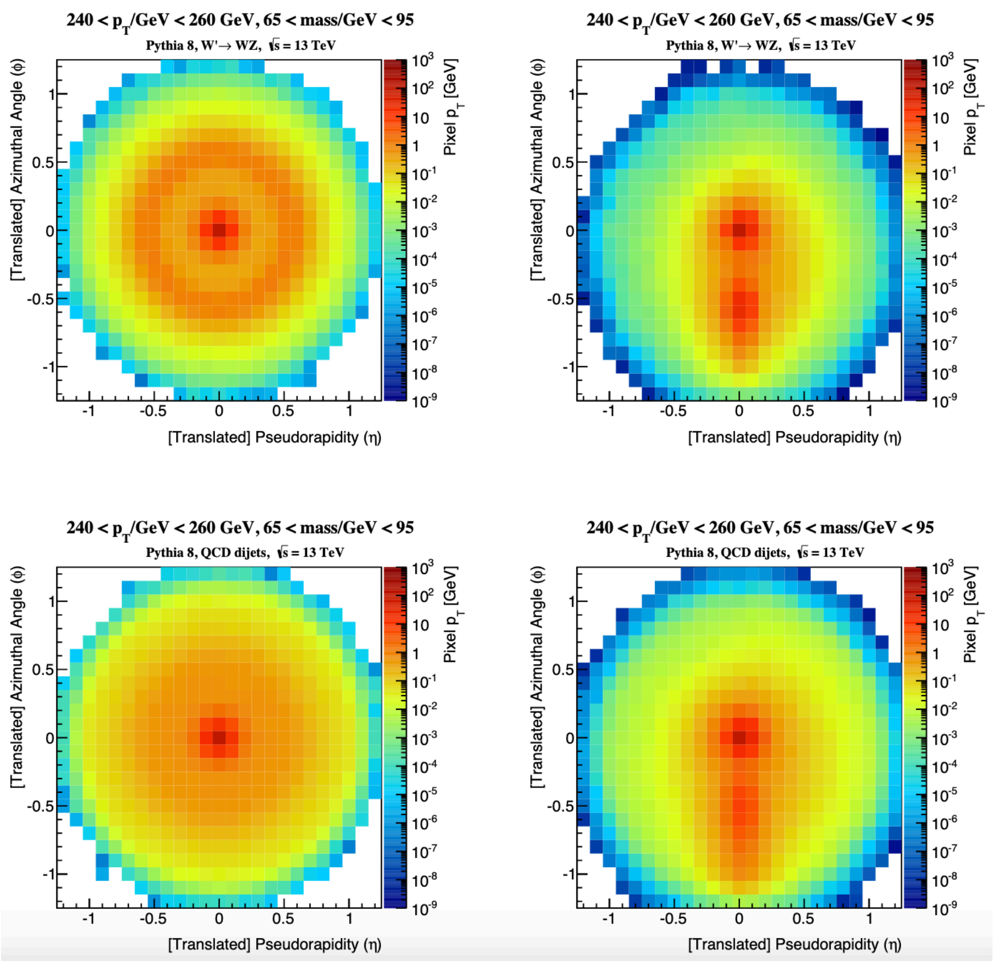
We acknowledge funding by the European Union's Horizon 2020 research and innovation programme, call H2020-MSCA-ITN-2020, under Grant Agreement n. 956086 (SMARTHEP).

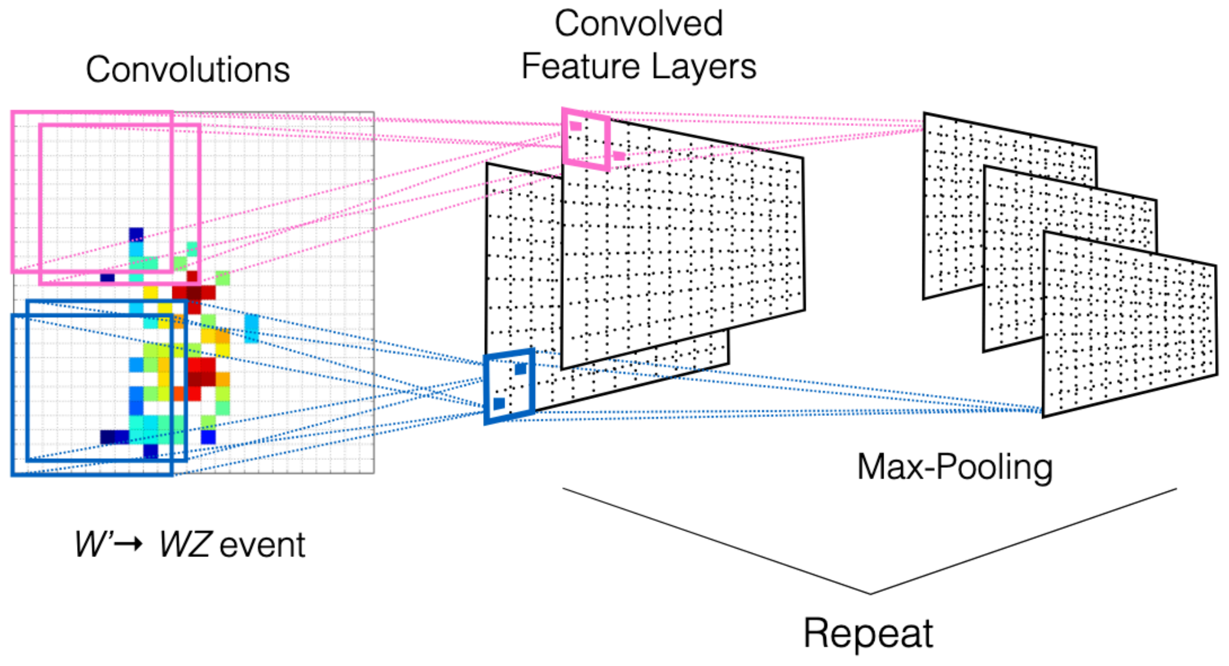
Introduction

- At the HL-LHC ATLAS **trigger** will be required to deal with more data and larger event sizes.
- **Current** jet preselection relies on **sequential, iterative** methods whose computational cost scales with the activity in the event.
- Can we approximate **jets** directly from **calorimeter cells**?
- Forego calorimeter clustering + jet reconstruction then use these primitive “cell jets” as a fast **calorimeter-only preselection** for jet triggers.
- Needs to be fast, flexible and robust to pile-up.
- Idea: Use a **CNN** to “detect” jets based on calorimeter energy deposits.

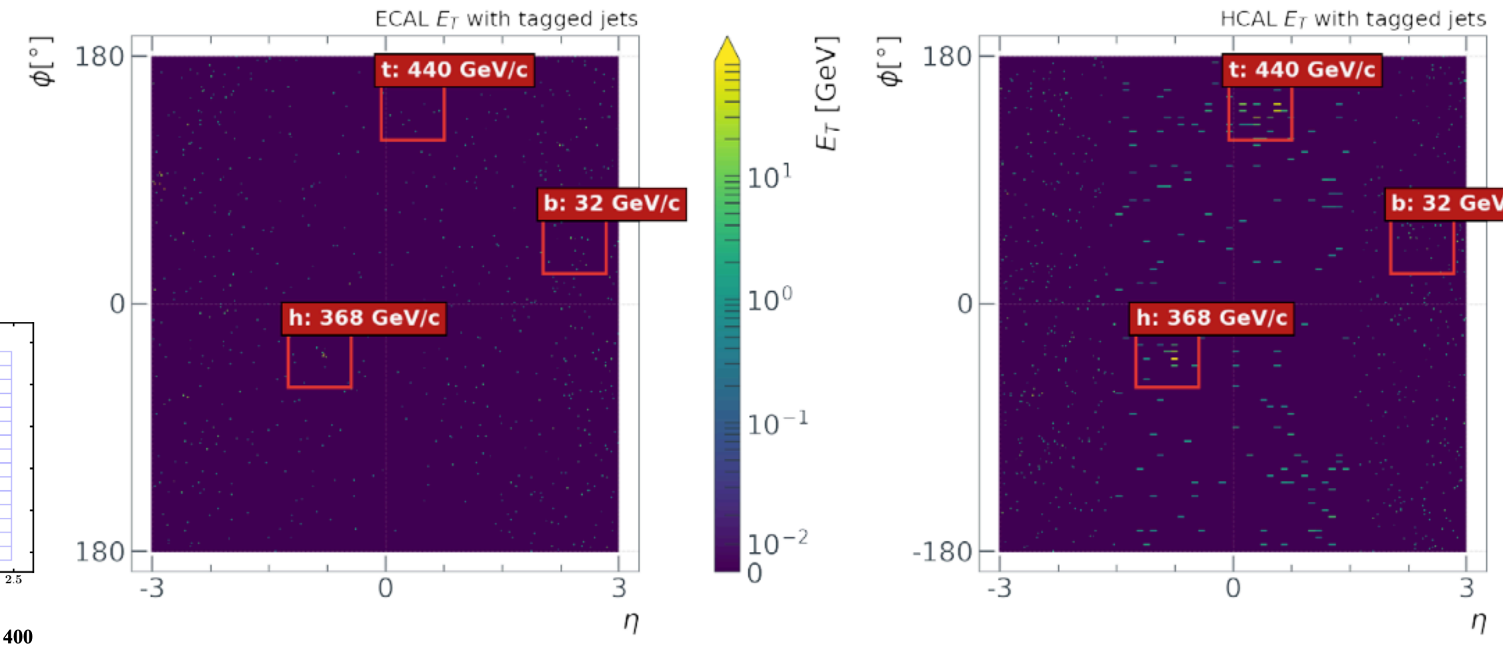
CNNs and Jets

- There is a lot of history treating **jets** as **images**.
- Previously many deep learning taggers have been proposed using CNNs.
- Exploit the **translational invariance** of CNNs + **local** spatial **correlations**.
- Most efforts focus on **classification** or **regression** tasks.
- In this work we consider the entire **event** at once.

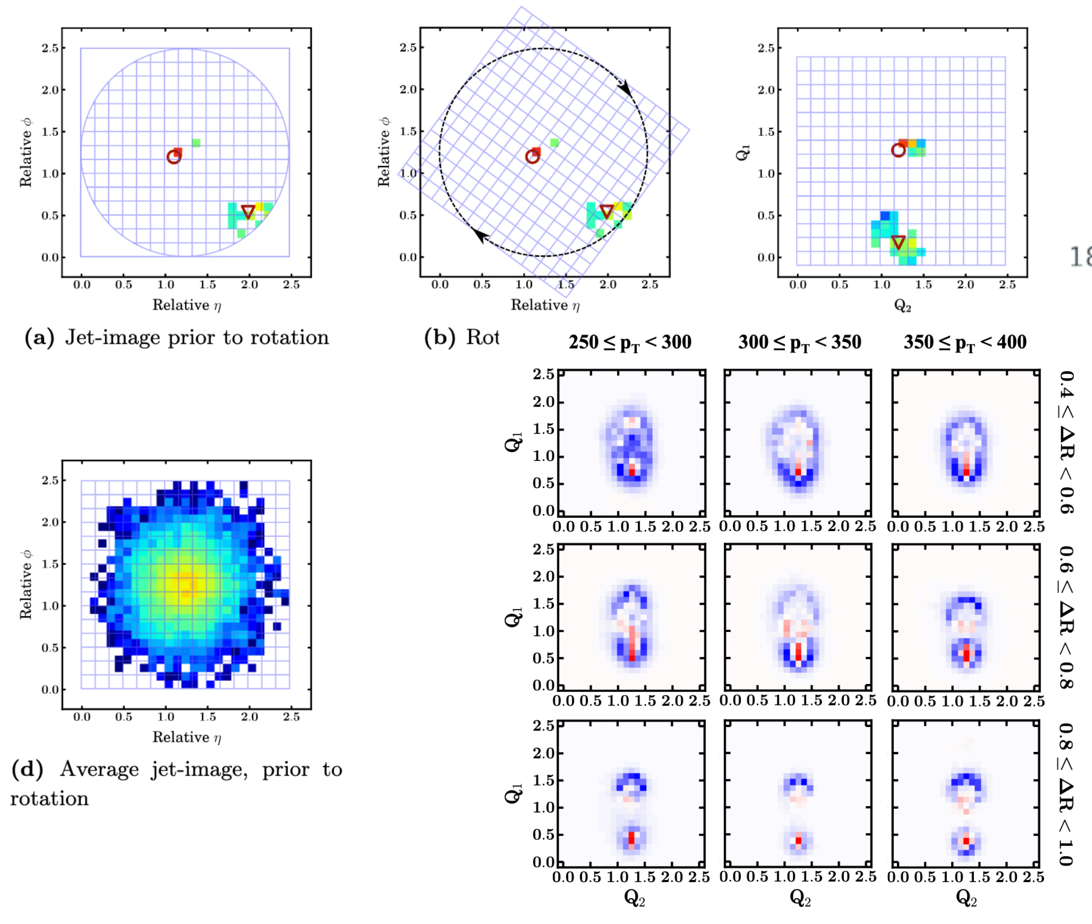




arxiv.org/abs/1511.05190v2



cds.cern.ch/record/2766650

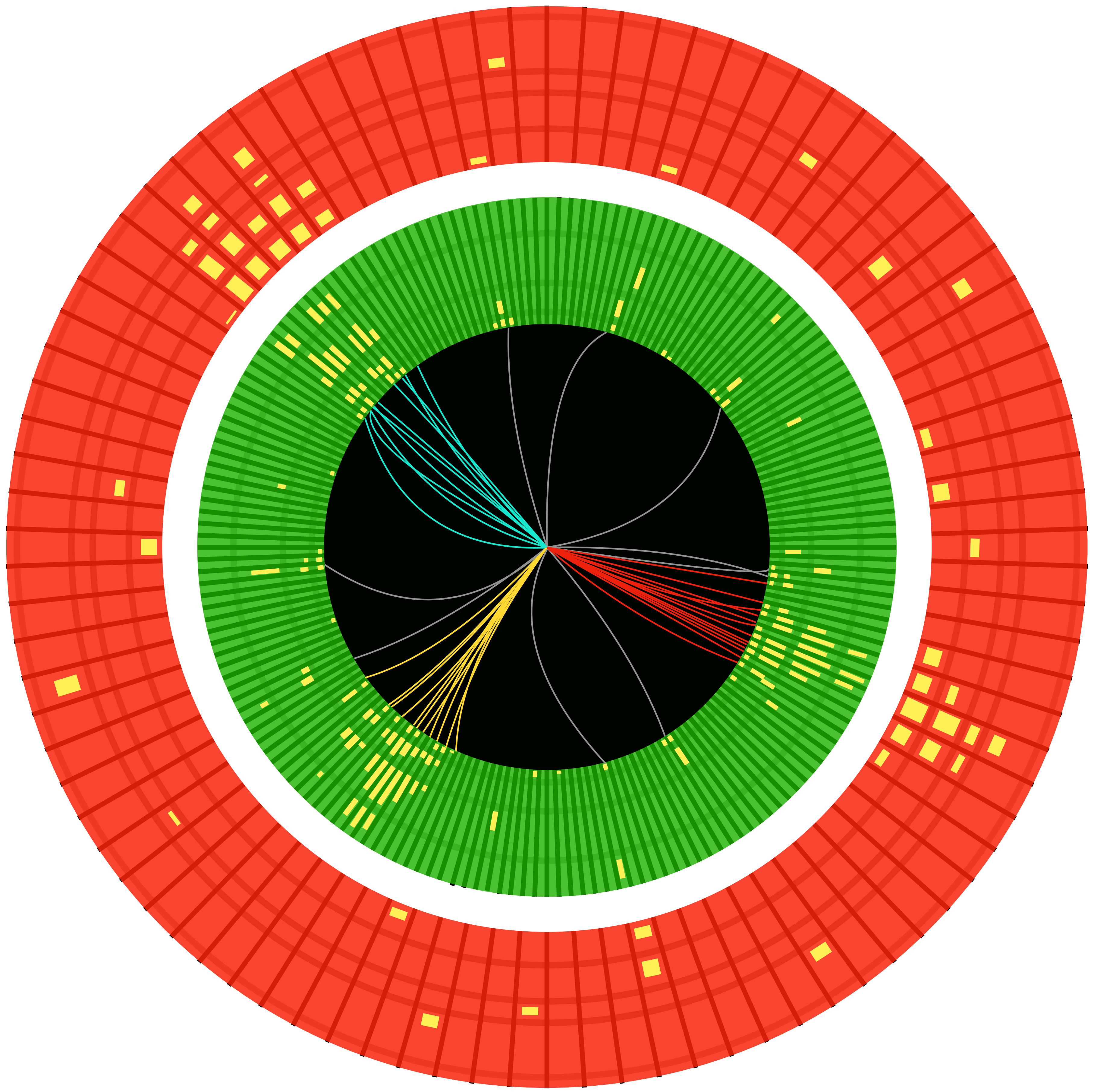


arxiv.org/abs/1407.5675

Calorimeter Data Preparation

Jet Finding $\Longleftrightarrow$ Object Detection

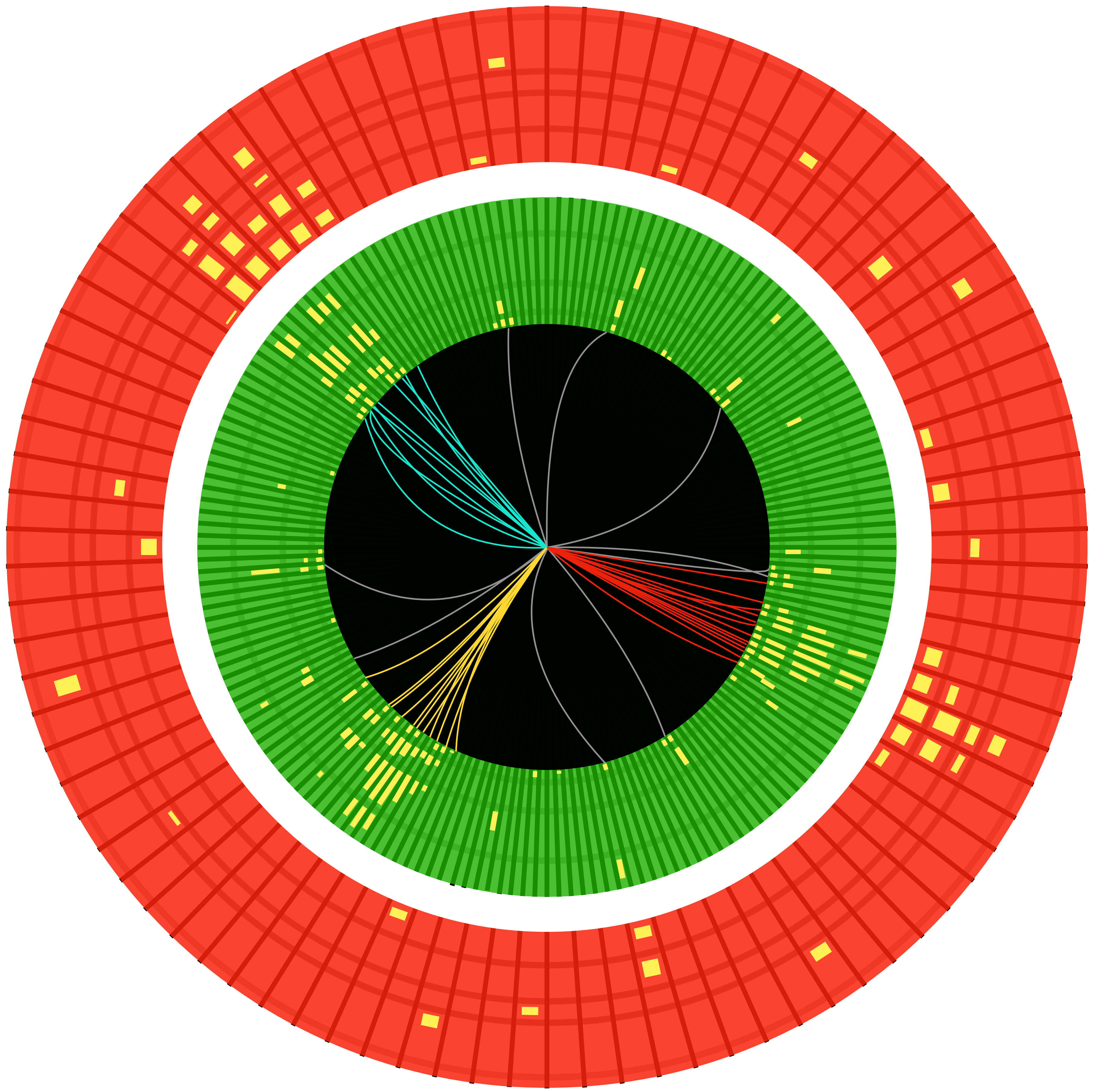
- Use a **CNN** to identify **jets** from energy depositions in the calorimeter **cells**.
- Return a series of **object proposals**, to use & interpret in simple jet triggers.
- Compare these calorimeter jets to existing, iterative methods used in the trigger.
- **Accelerate** CNN inference using **GPU**. Explore timing constraints of ATLAS trigger for deployment.



Calorimeter Data Preparation

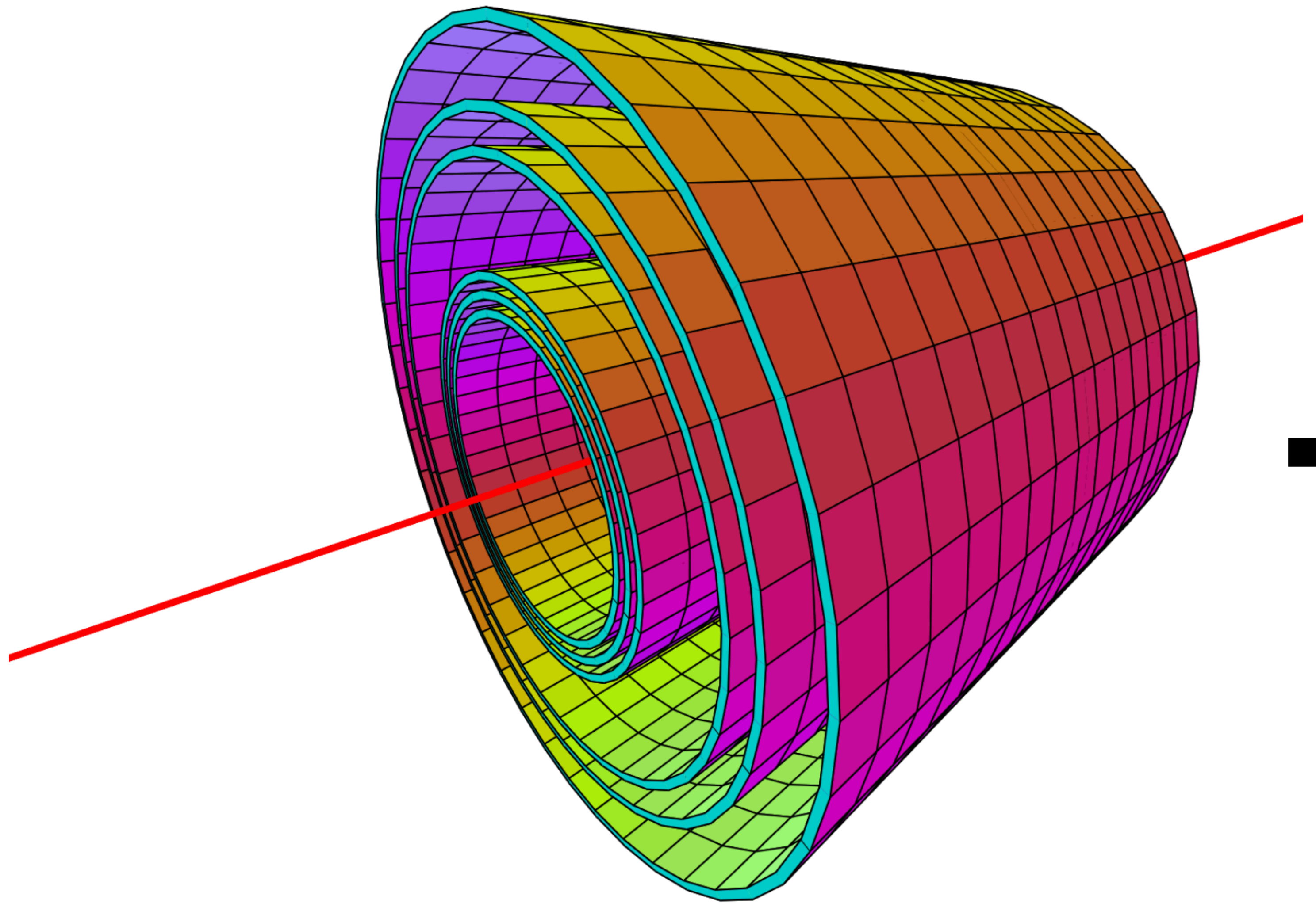
Jet Finding ⟺ Object Detection

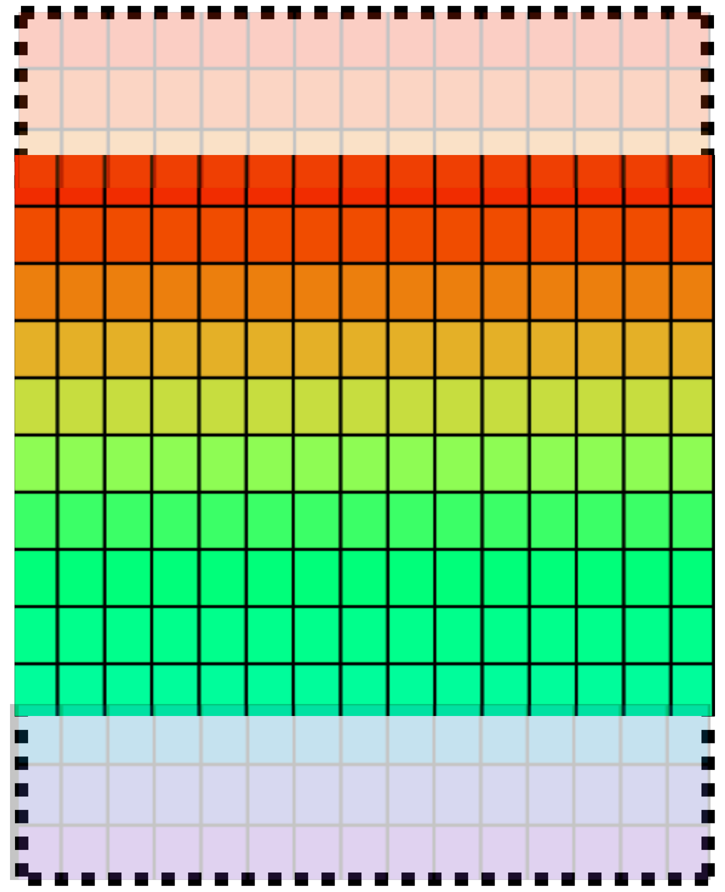
- Use a **CNN** to identify **jets** from energy depositions in the calorimeter **cells**.
- Return a series of **object proposals**, to use & interpret in simple jet triggers.
- Compare these calorimeter jets to existing, iterative methods used in the trigger.
- **Accelerate** CNN inference using **GPU**. Explore timing constraints of ATLAS trigger for deployment.
- **How can we make a regular 2d representation from a highly complex, non-uniform and sparse set of calorimeter cells?**

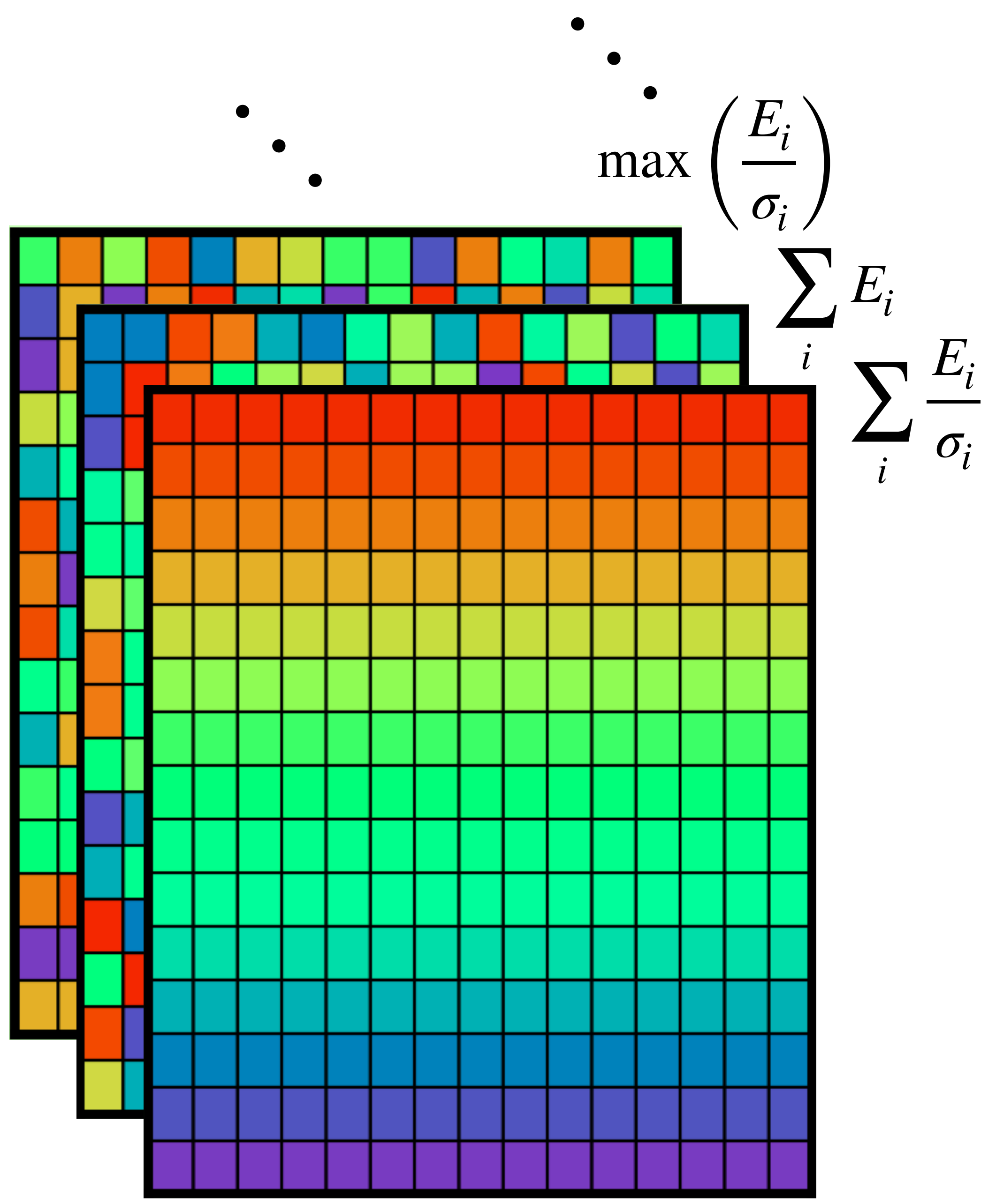


Calorimeter Data Preparation

Preprocessing for CNNs







Focus on central "barrel" and project in $\eta - \phi$

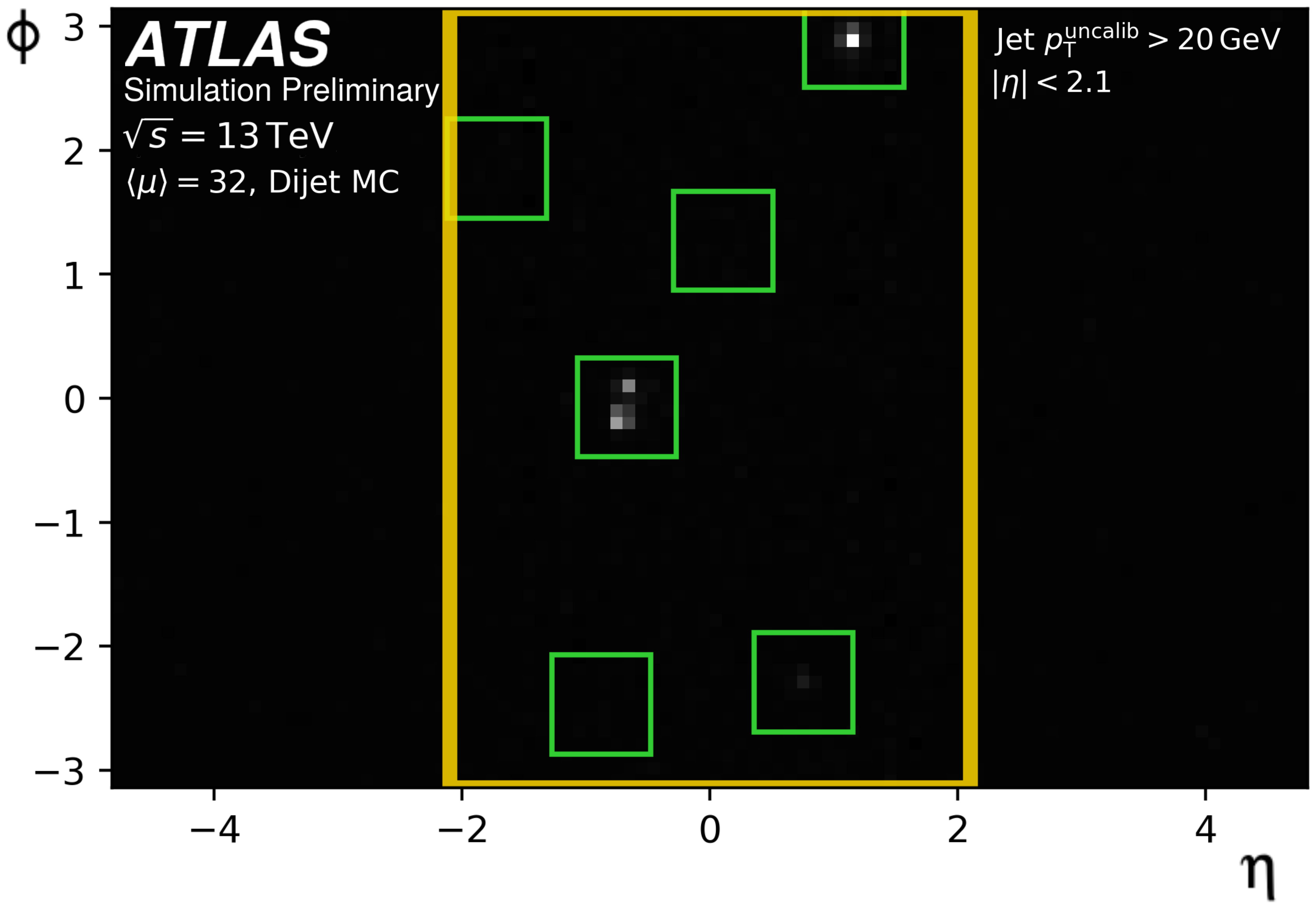
"Wrap" boundary regions

Calculate separate channels using cell information

Calorimeter Data Preparation

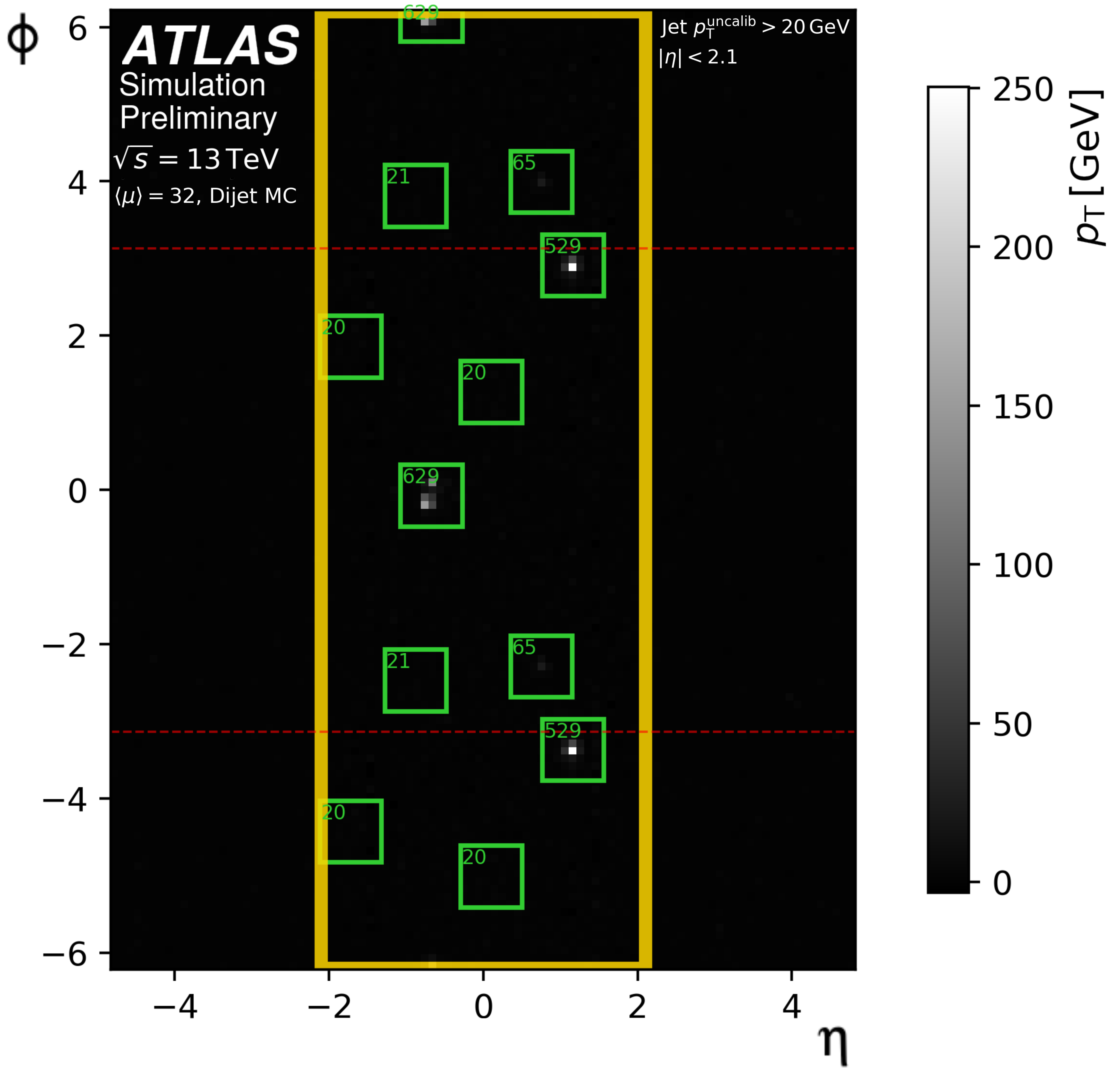
Anti-k_t jets as targets

What we "see" in the calorimeter:



*Note: All the jets are **uncalibrated** (considered at the jet constituent scale) AND **central** ($|\eta| < 2.1$)

What we pass to the network:

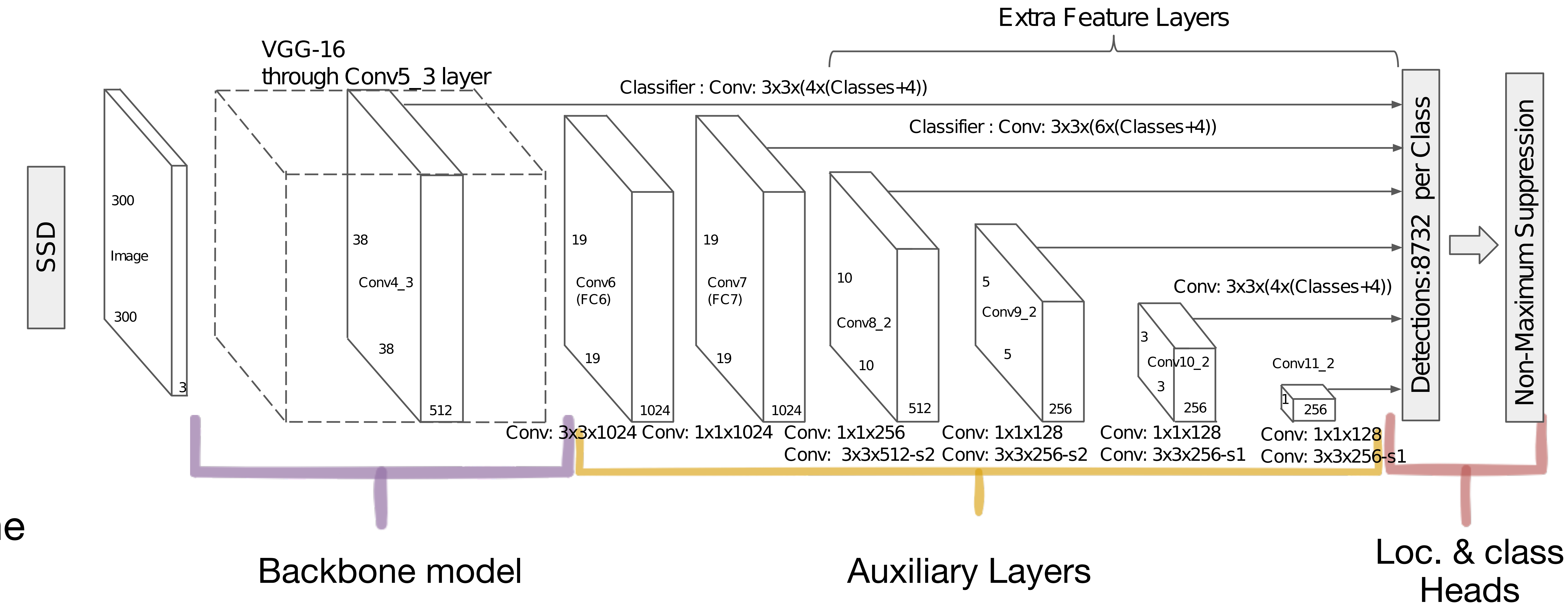


Network Architecture

Network Architecture

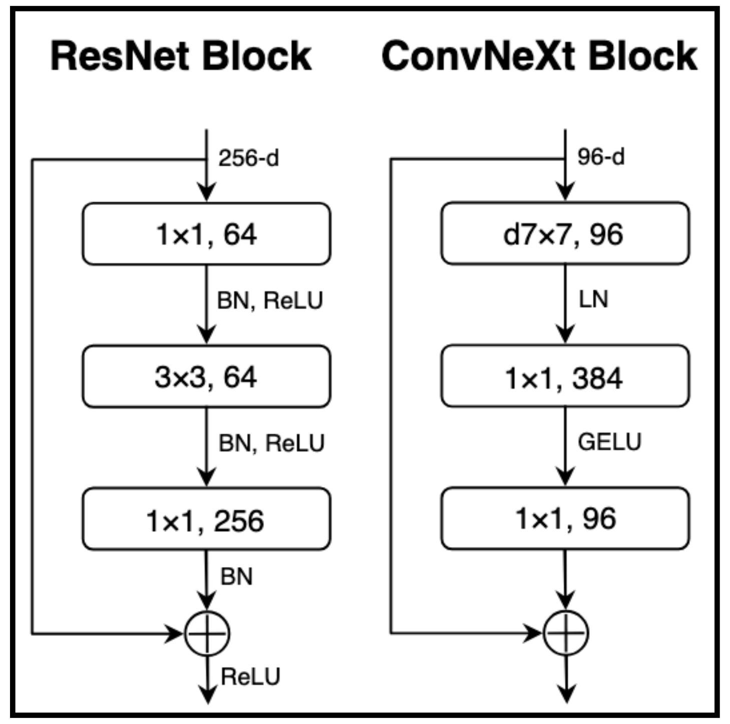
Original SSD architecture

- **Backbone**
  - VGG16 architecture used as feature extractor
  - 35 million parameters, large + relatively old
- **6 Additional Feature Layers**
  - Capture objects of different scales
- **Residual connections** between the layers and outputs
- Two **output heads**, regression + classification
- Total **learnable parameters**: 35,641,826



arxiv.org/abs/1512.02325

arxiv.org/abs/2201.03545



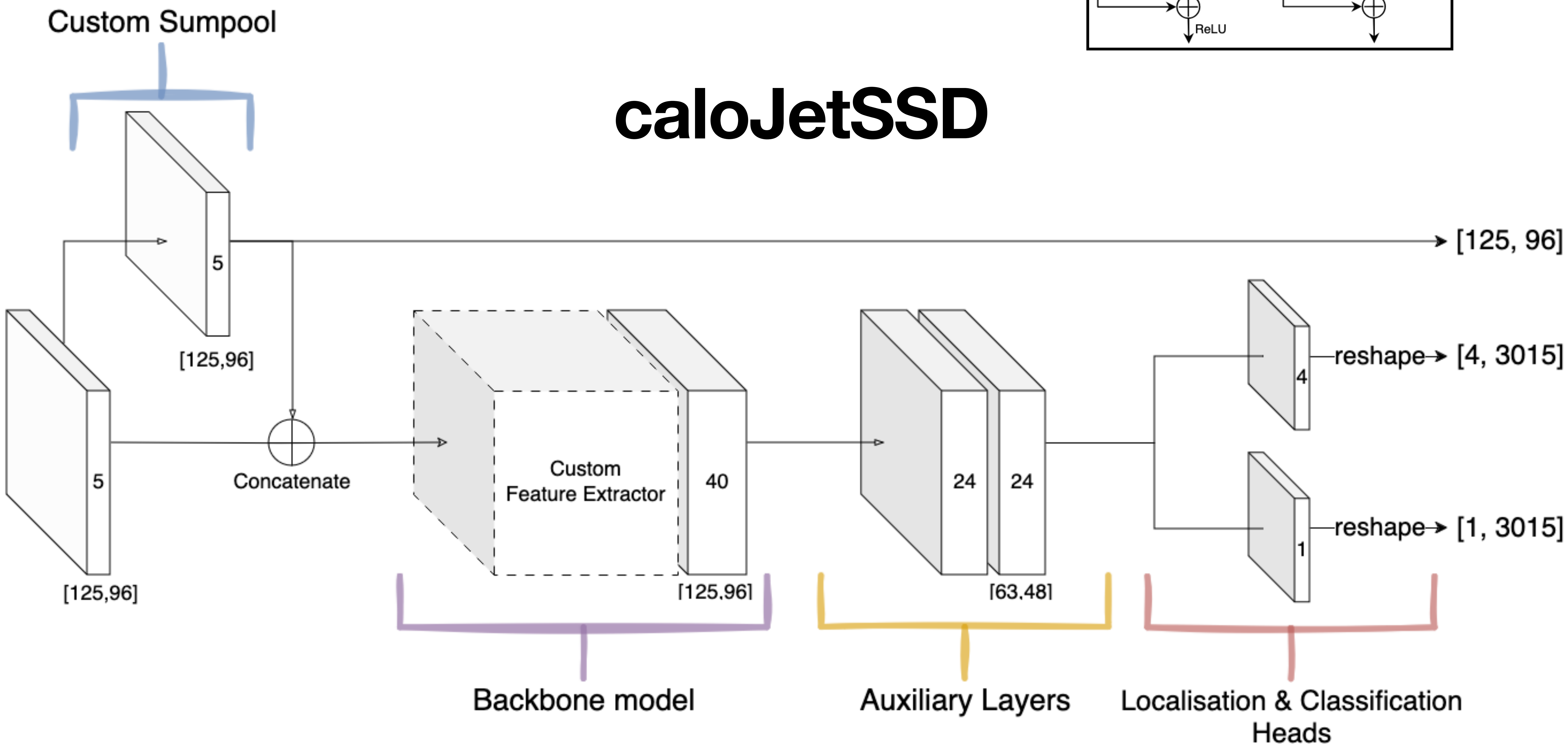
Network Architecture

Modernising SSD + feature extractor network

- **Backbone**
  - Very aggressively reduced the size and depth of the backbone
  - Adapted ConvNeXt blocks
  - >10m → 30k learnable params
- **One Additional Feature Layer**
  - Reduced the # kernels and channels in the auxiliary layer
- **Output heads**
  - Decreased number of prior boxes and shape of output (factor ~2)
  - Introduced "sumpool" output array for "quick" p_T estimation
- Total **learnable parameters: 50,841**

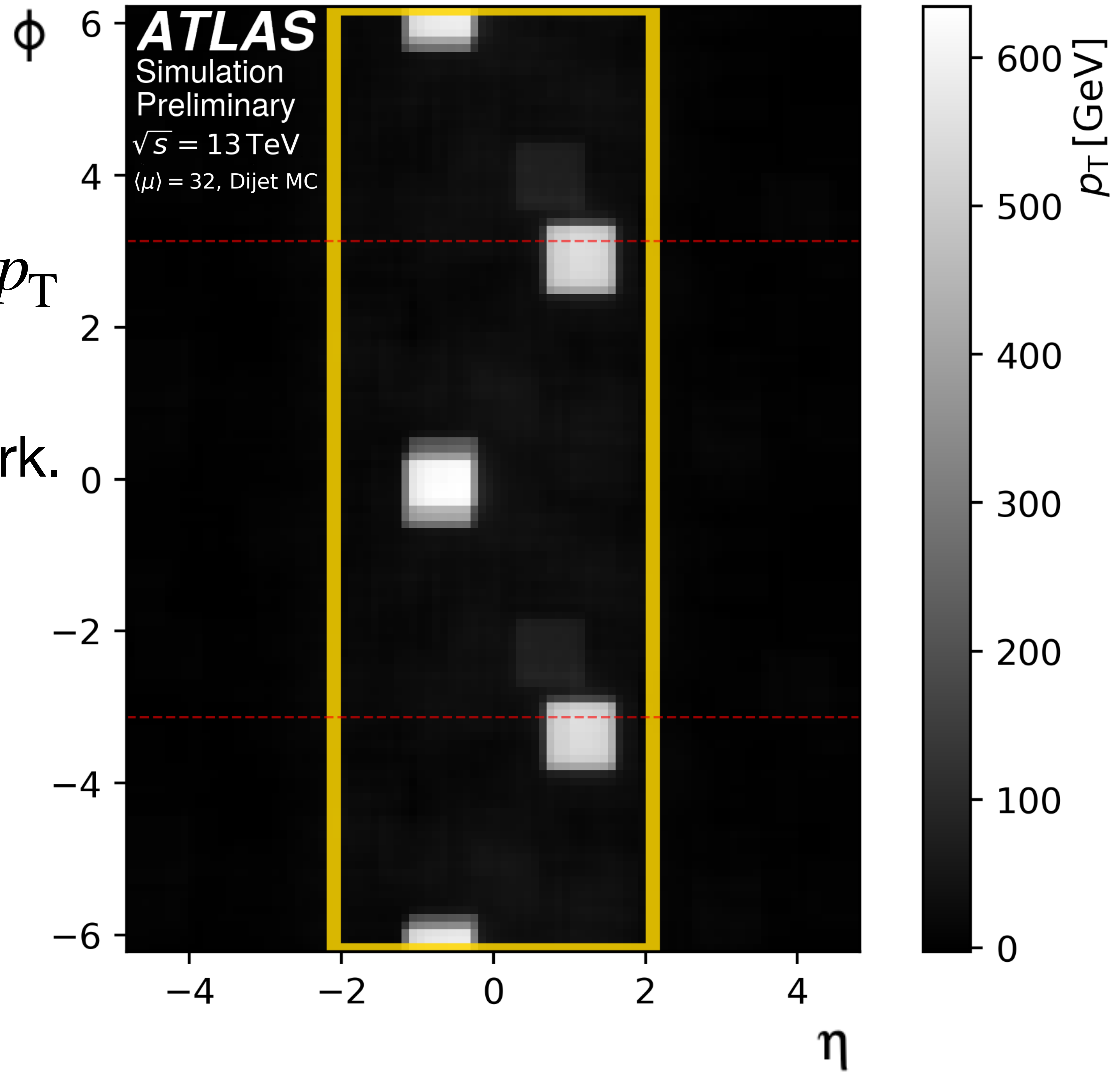
700 times fewer!

caloJetSSD



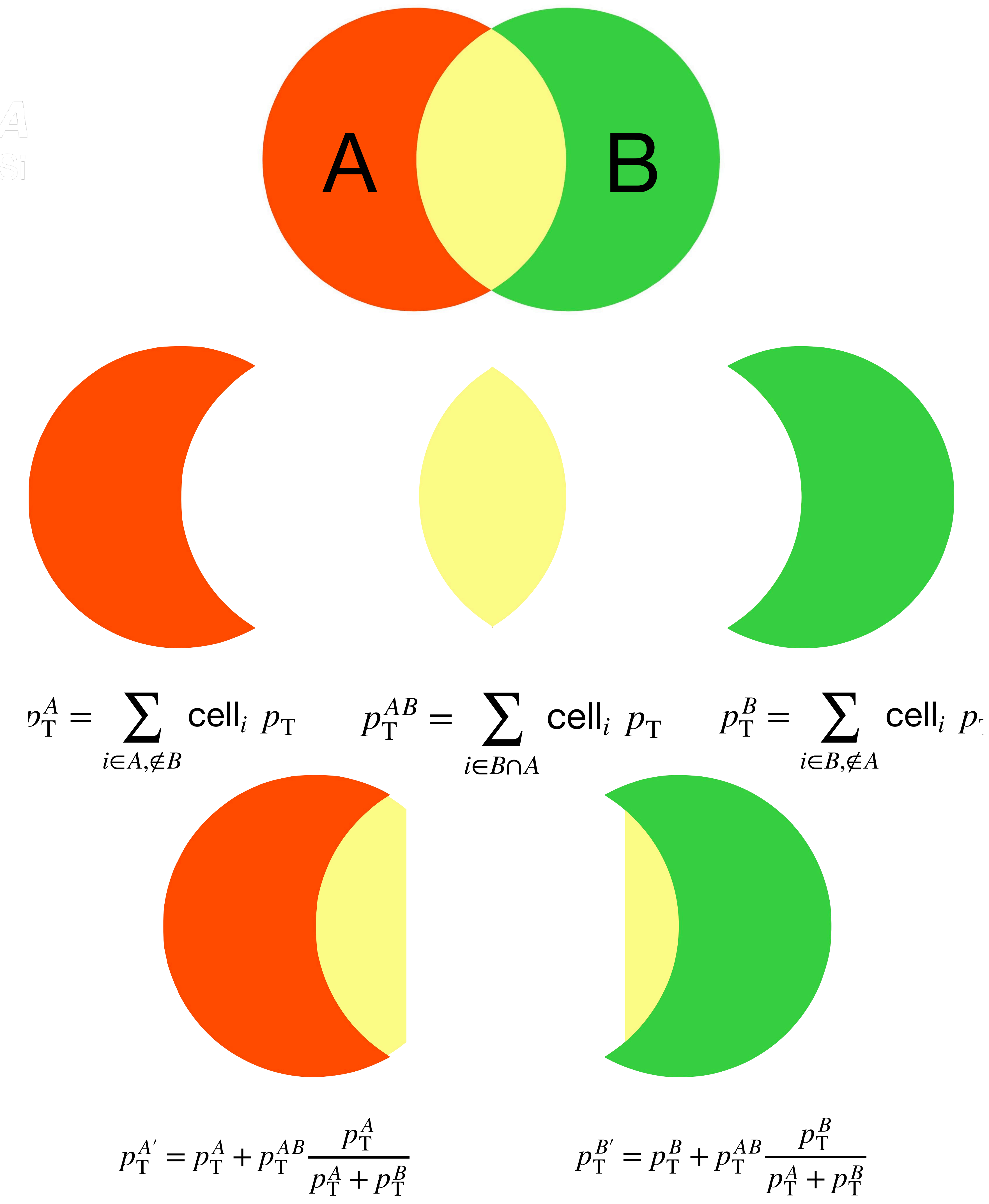
Transverse Momentum Estimation

- Object detection finds the location of the jets.
- To evaluate trigger decisions we estimate the p_{T} of the jet predictions.
- **Direct method**: Sumpool output of the network.
    - Sum pixels in 9x9 kernel or window.
    - Location of prediction determines $\sum p_{\mathrm{T}}$ value.



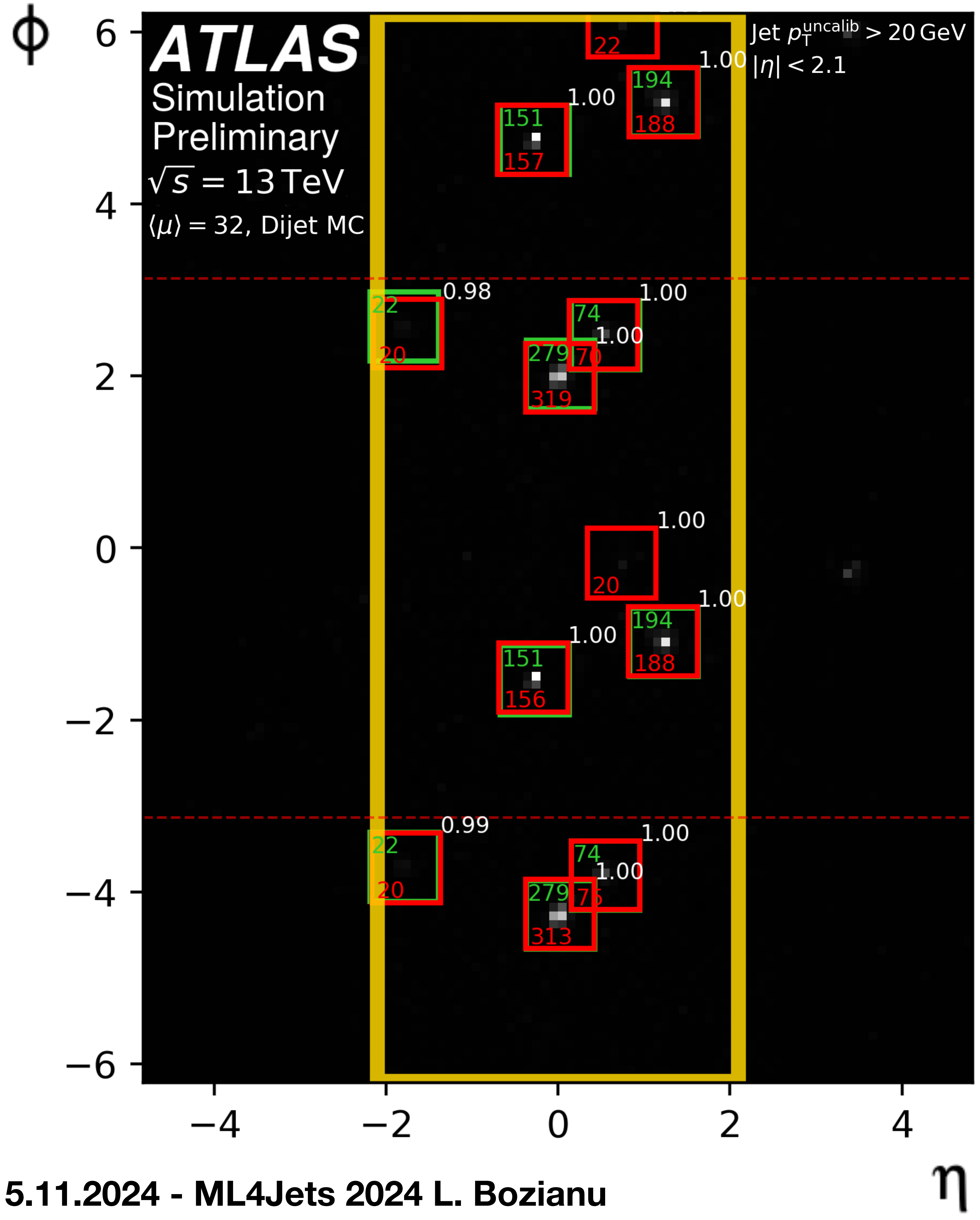
Transverse Momentum Estimation

- Object detection finds the location of the jets.
- To evaluate trigger decisions we estimate the p_{T} of the jet predictions.
- **Direct method**: Sumpool output of the network.
- **Iterative method**: Weighted circle.
    - Retrieve cells in $R = 0.4$ circle centred on each prediction.
    - Share p_{T} among overlapping predictions.

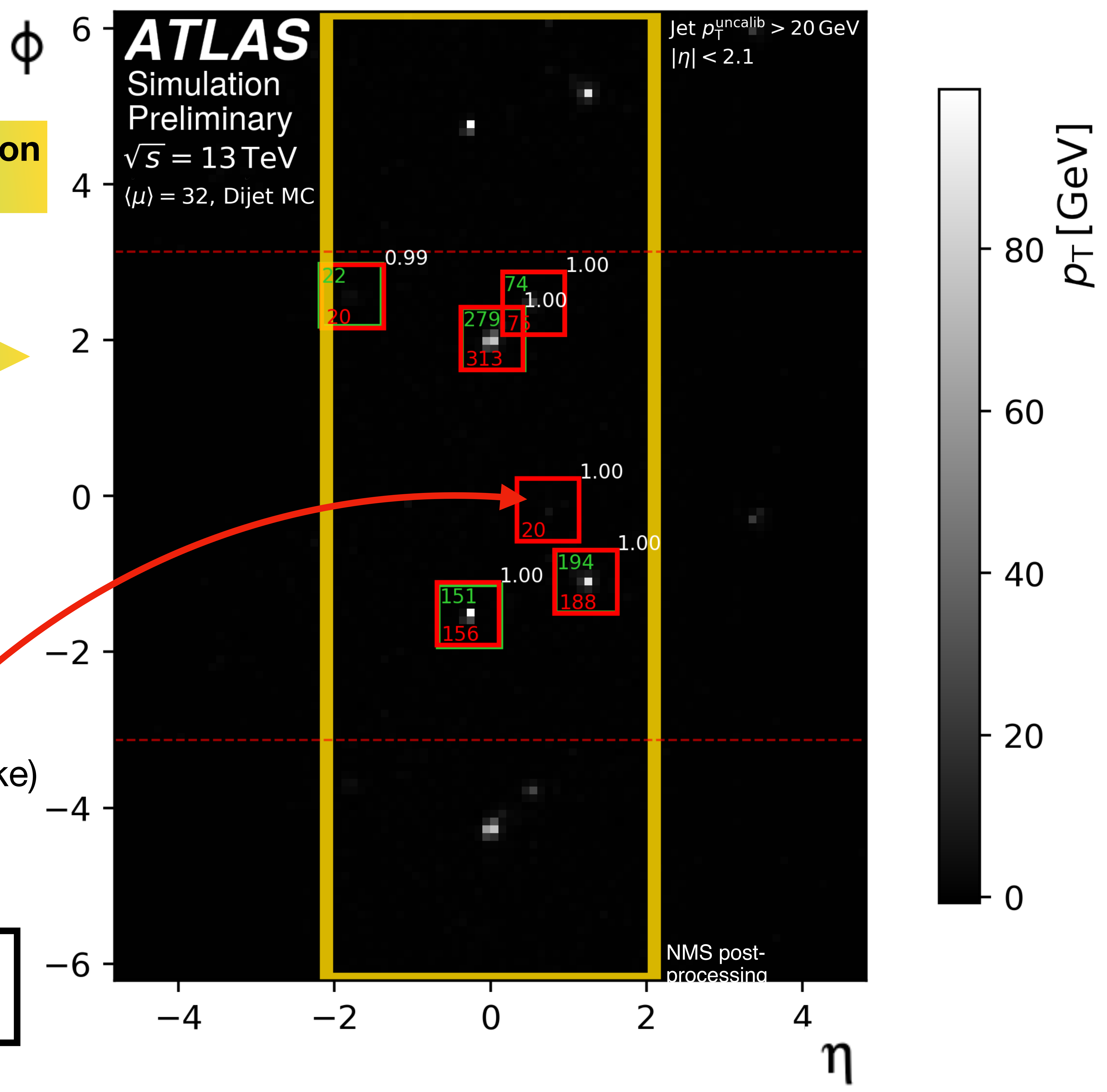


Performance Results

Jet Detection for a <u>single</u> event with $\langle \mu \rangle = 32$



Non-Maximum Suppression post-processing



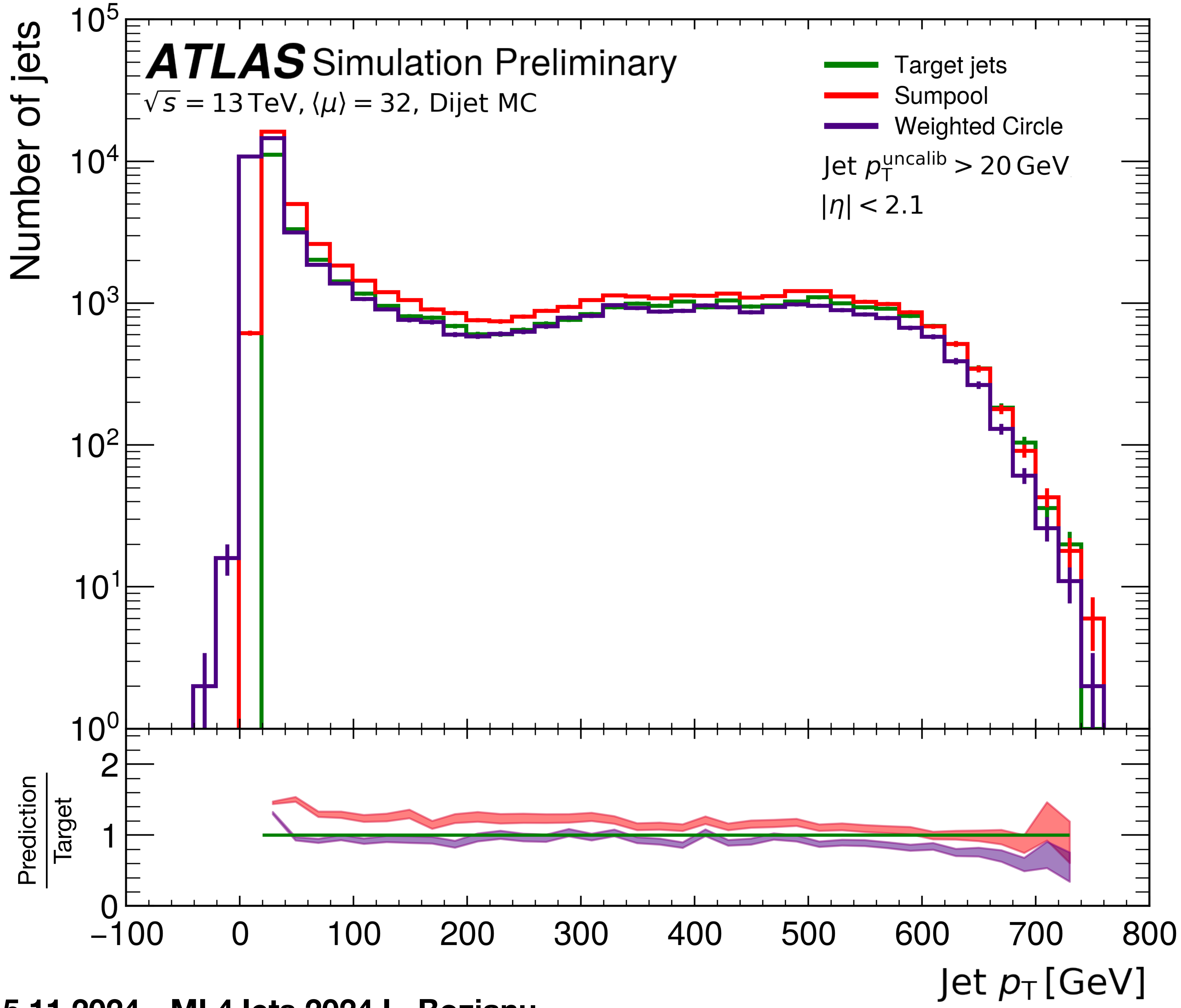
Unmatched prediction (fake)

Anti-k_t jet p_T overlaid (green)
Sumpool p_T overlaid (red)

Jet Detection for a single event with $\langle\mu\rangle = 200$

Reconstructing jet p_T

*LHC **Run 2-like** conditions*



*HL-LHC **high pile-up** conditions*

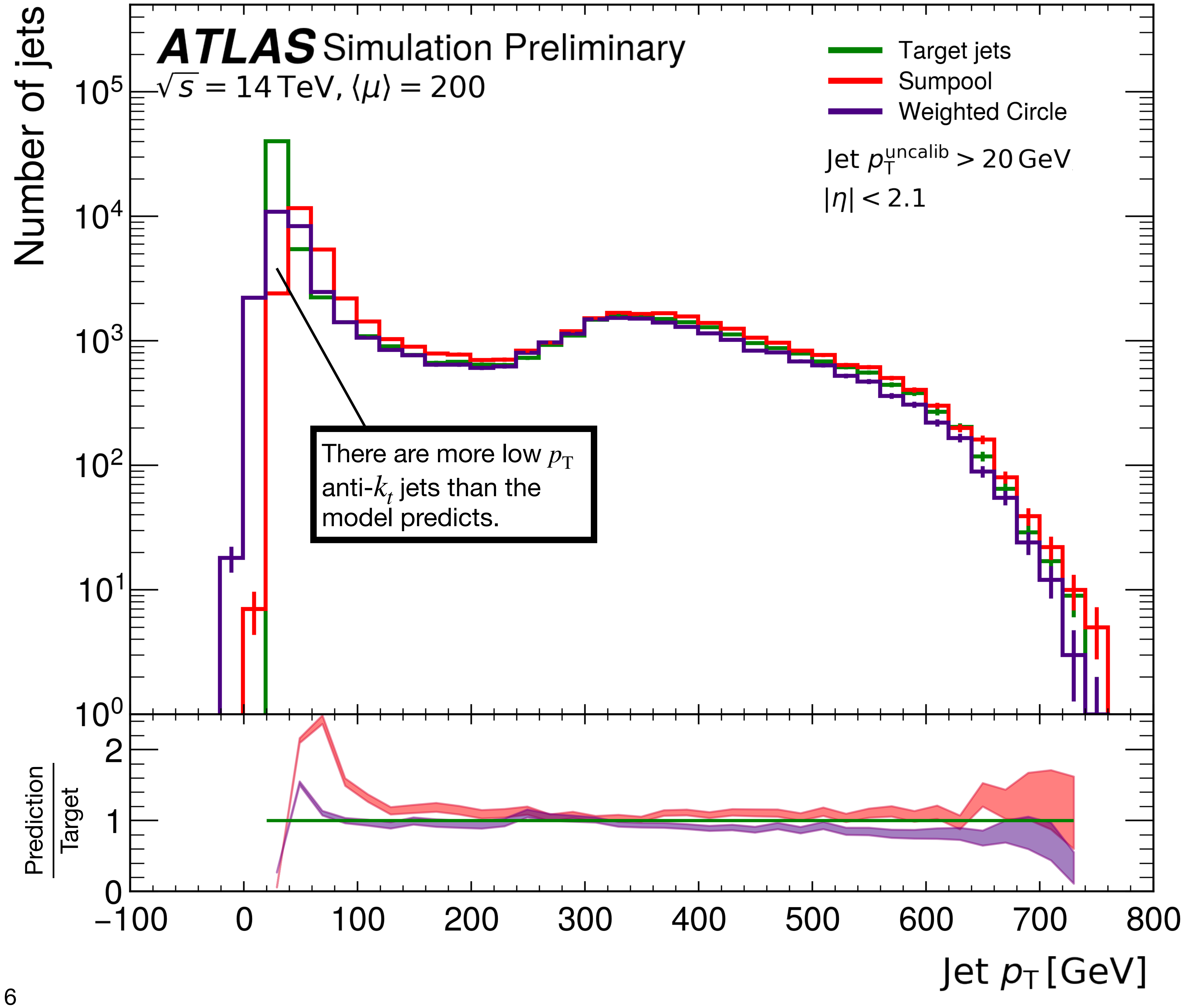
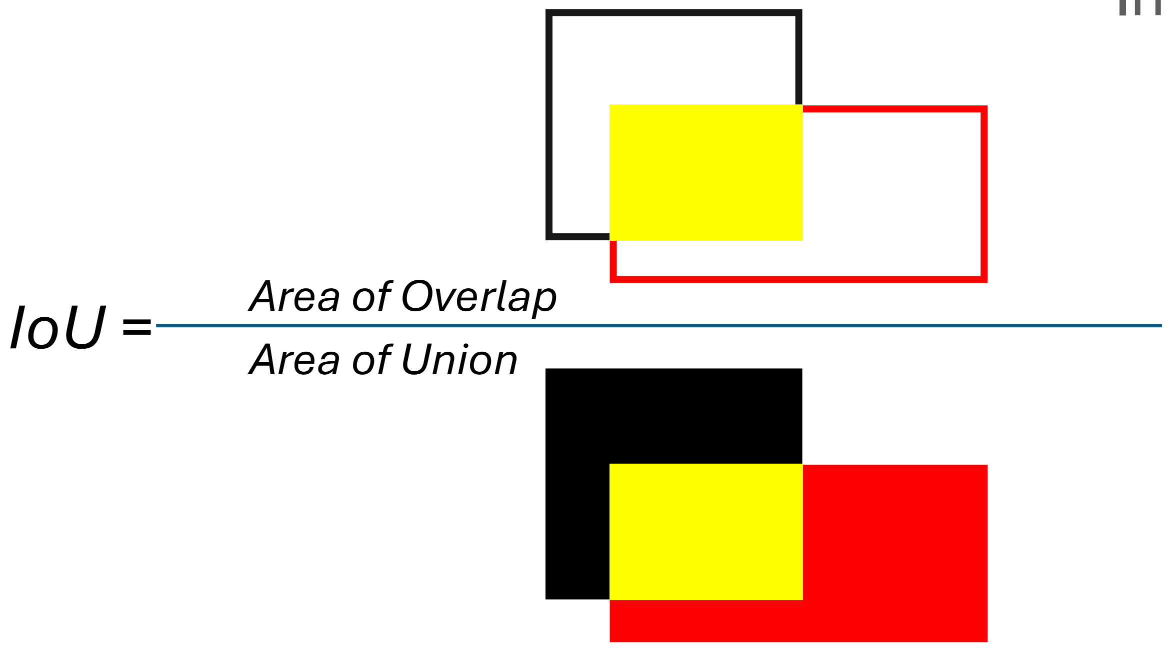


Image: M. Moors

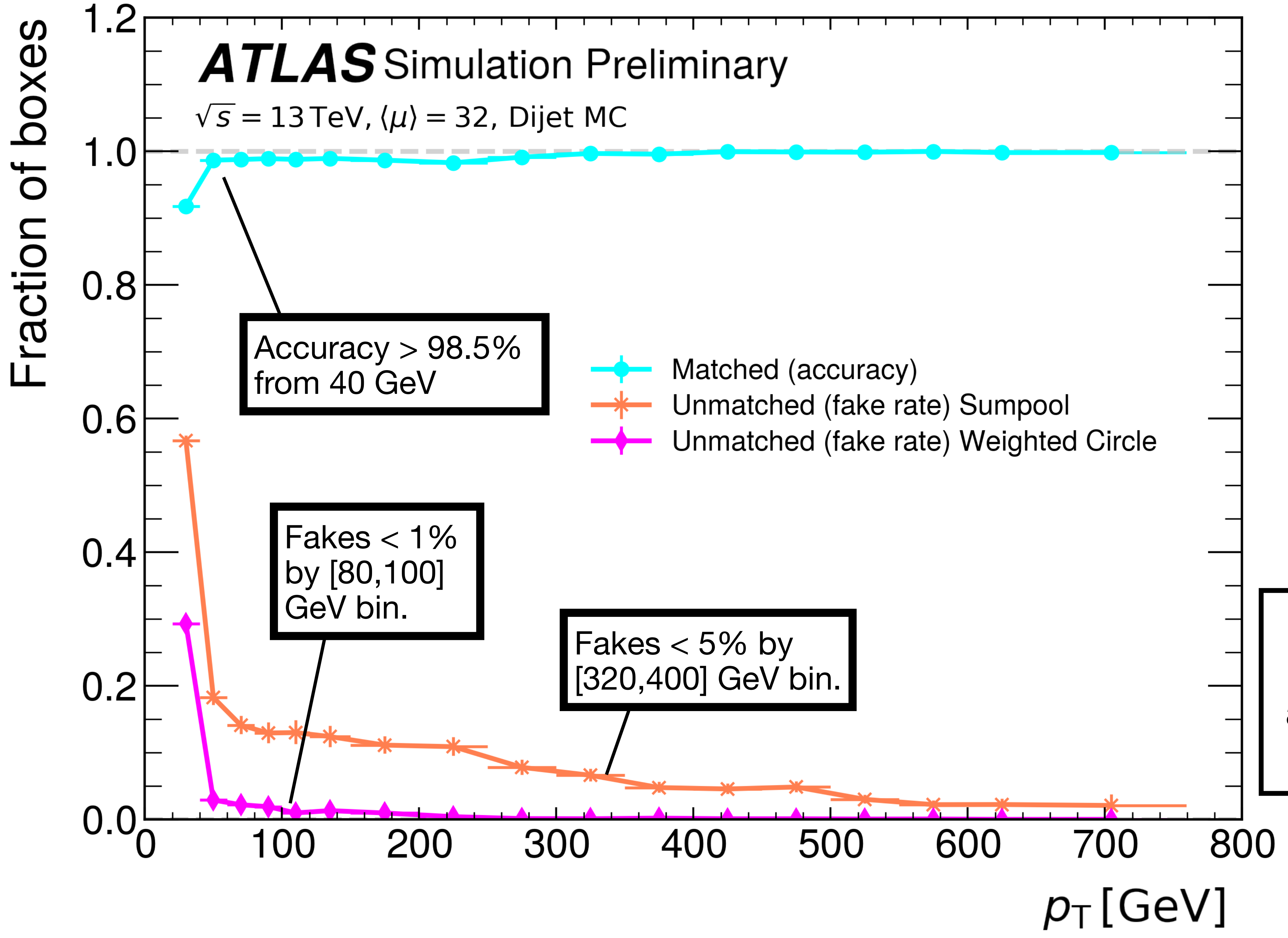
Performance across p_T



Detection accuracy vs fake rate:

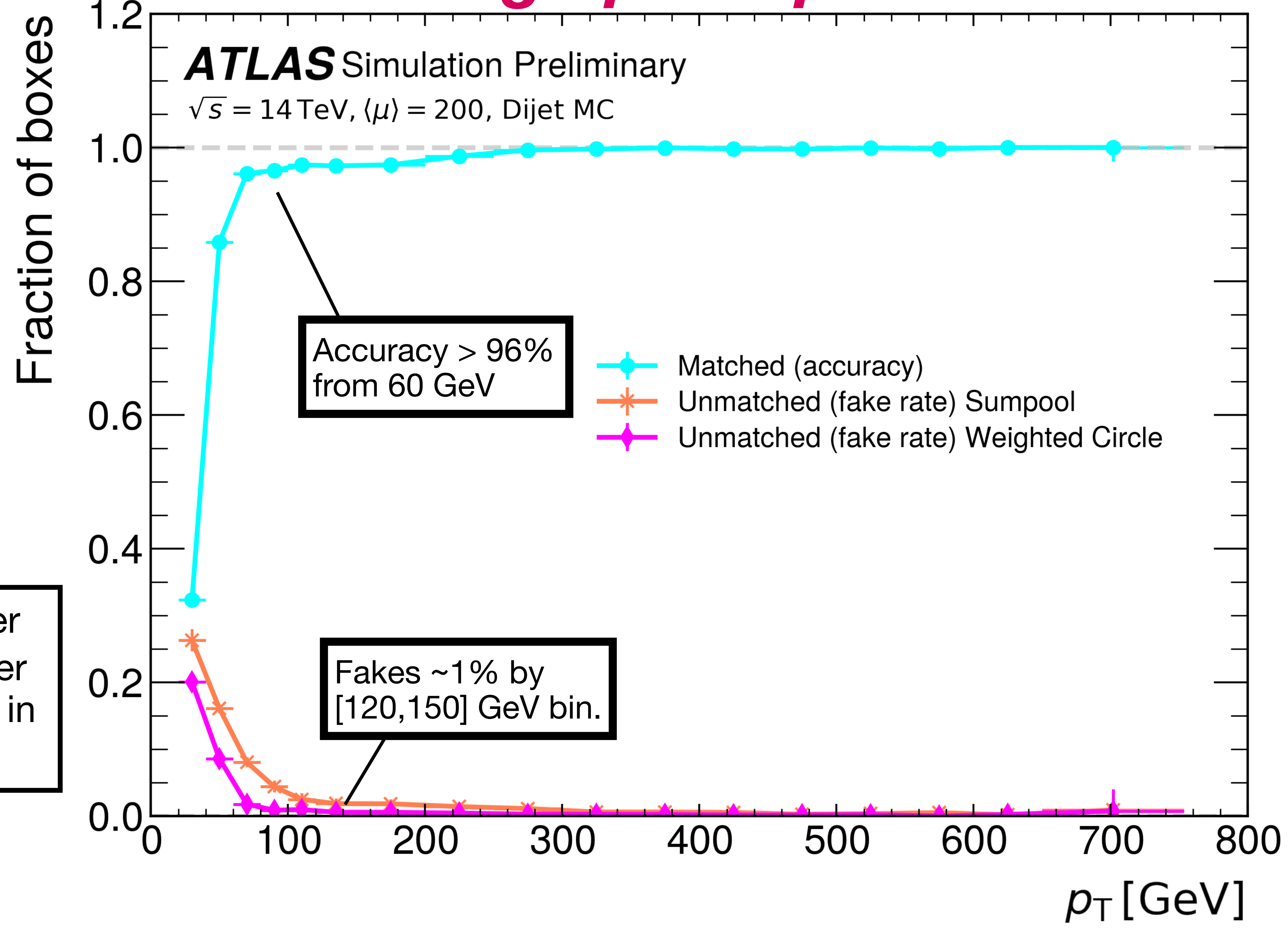
- % matched - Target jets *found* with intersection over union (IoU) > 0.5 with any prediction.
- % unmatched - Predictions with no corresponding target jet, *or* predictions that overlap with a previously matched target.

LHC ***Run 2-like*** *conditions*



Trade-off between fewer fake predictions ↔ lower accuracy. Largest effect in low p_T bins.

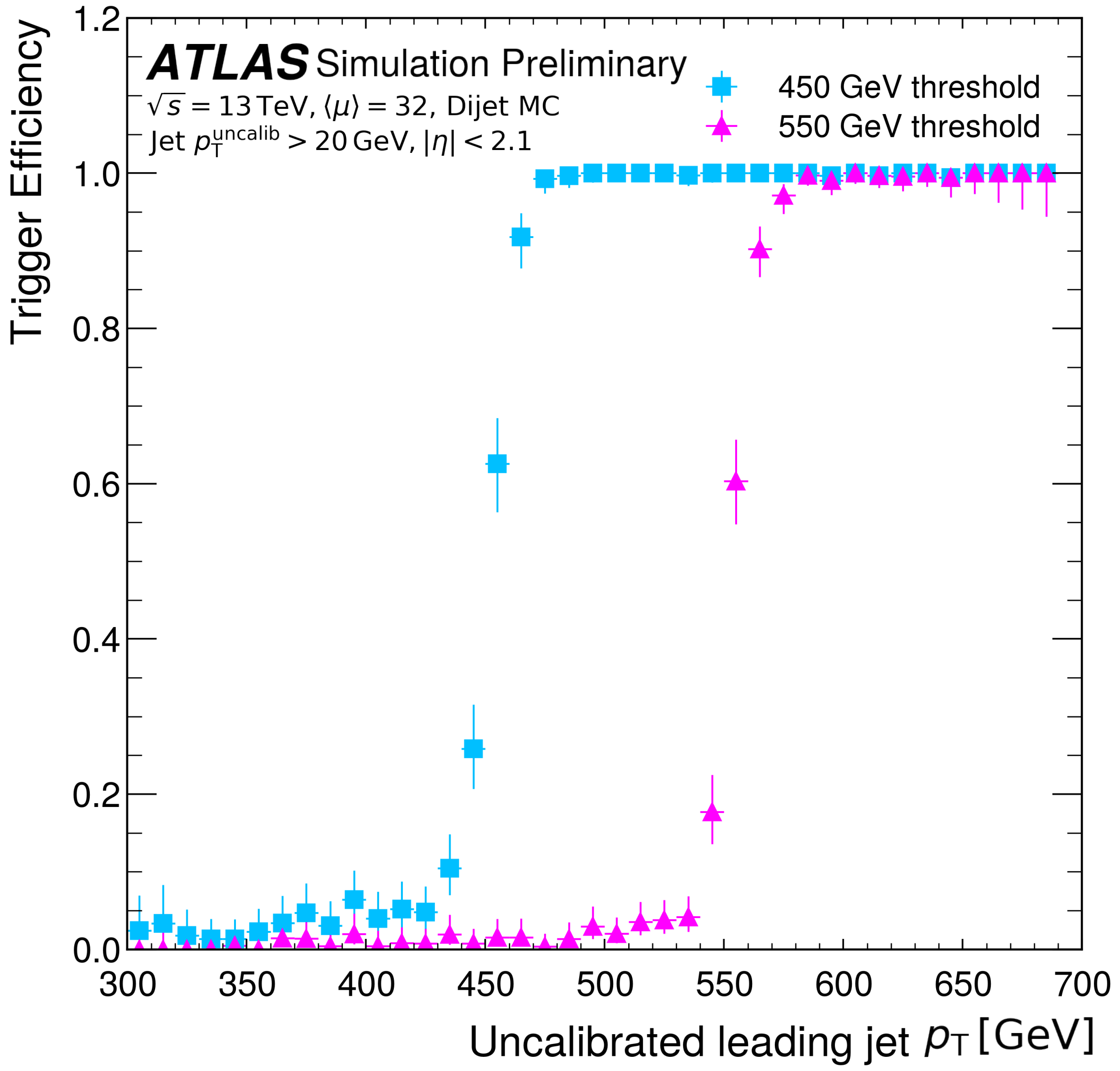
HL-LHC ***high pile-up*** *conditions*

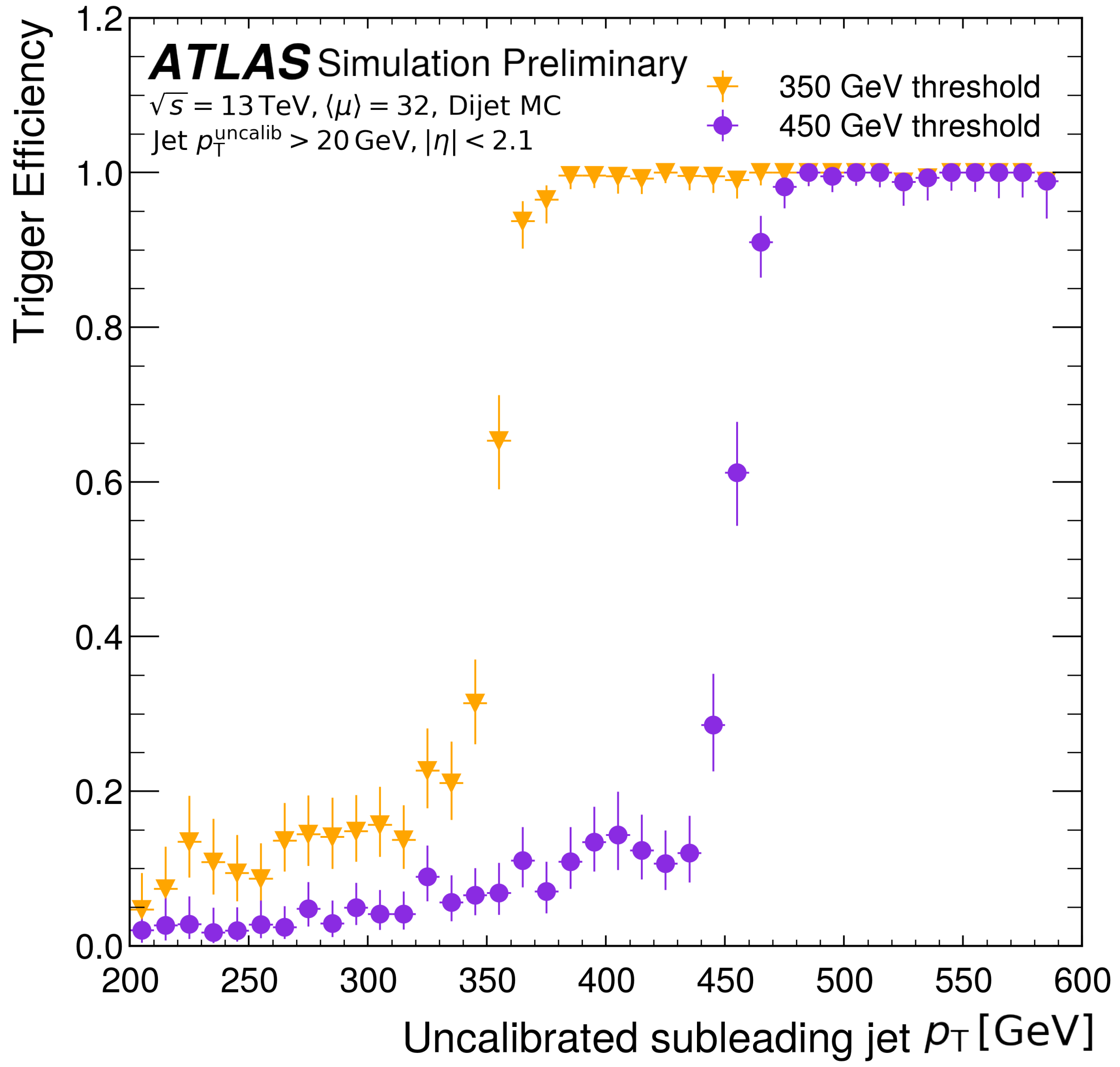


Trigger efficiencies

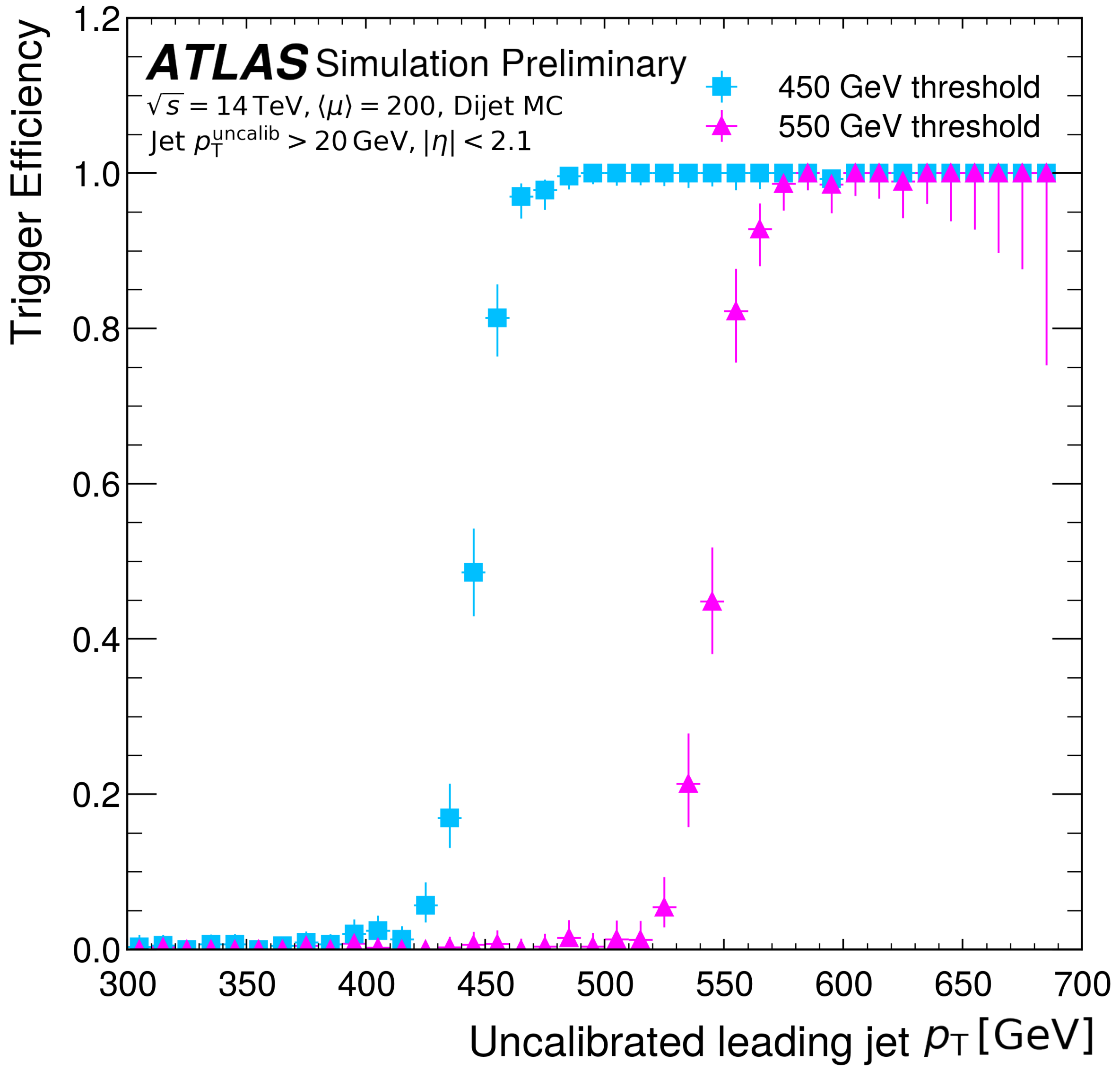
Leading and sub-leading jets using sumpool output

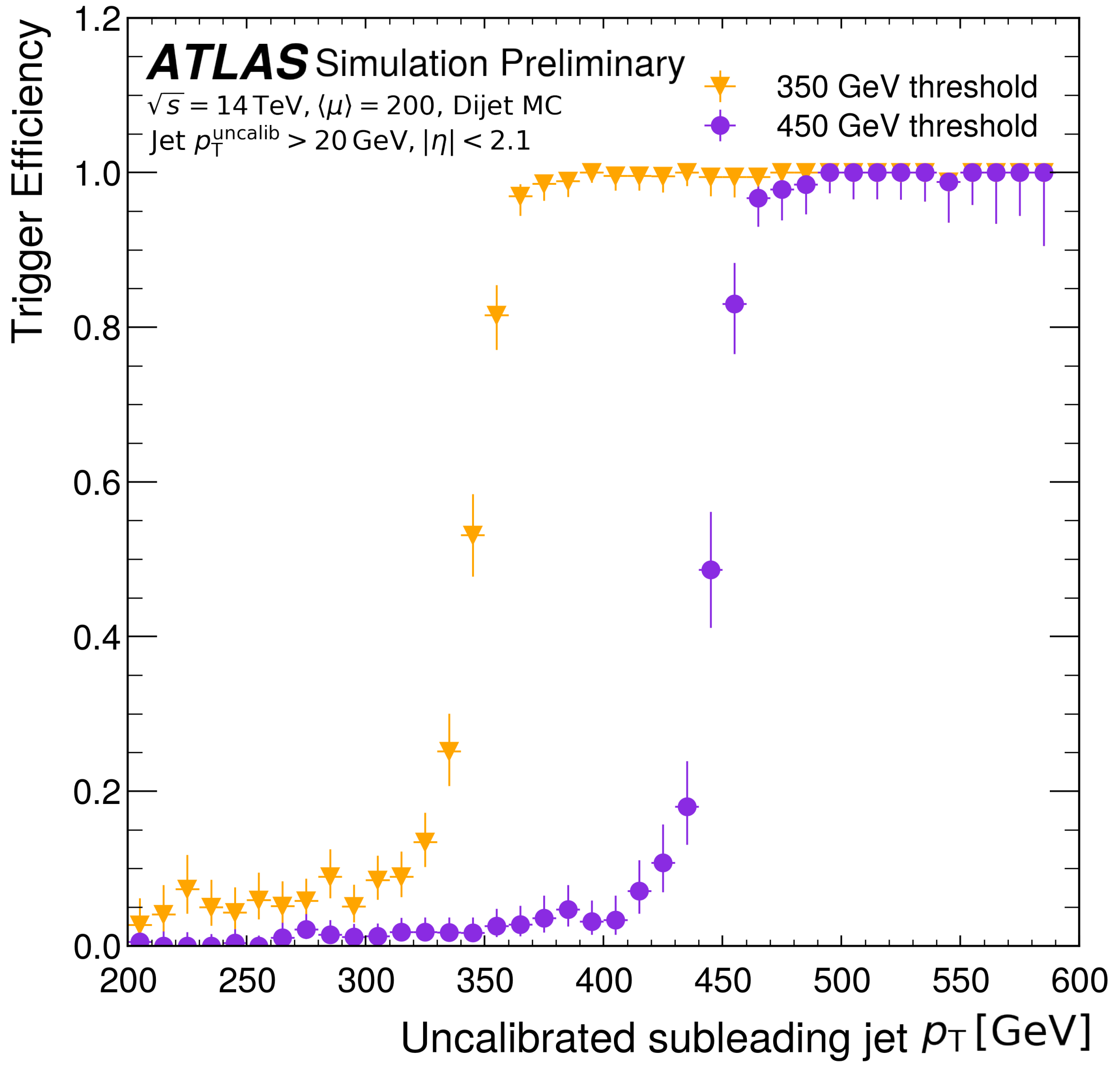
*LHC **Run 2-like** conditions*





*HL-LHC **high pile-up** conditions*





Sharp turn on with the plateau approaching 100% in both cases!

Timing evaluation

- Pre- and *optional* post-processing executed on a single CPU (AMD EPYC 7742 CPU).
- Model inference on a single NVidia RTX 2080 Ti GPU.
- The current, iterative calorimeter preselection jet reconstruction takes $\mathcal{O}(100\,\mathrm{ms}) \implies$ **caloJetSSD** is an **order of magnitude faster**.

Preprocessing

8.1 ± 4.3ms

Model inference*

4.4 ± 1.5ms

(*Optional* post-processing)

Weighted circle method

11.3 ± 4.9ms

*including data transfer + decoding

Conclusion

Conclusion

- We can use CNNs to approximate jets in the calorimeter.
- The complexity of the model can be reduced significantly, with respect to the SSD literature, without a loss in performance.
  - We don't need to use million-parameter models! caloJetSSD *700* times smaller.
- Promising trigger efficiencies for simple jet hypotheses.
- Robust against pile-up, still performant in HL-LHC conditions.
- Order of magnitude speed-up over current iterative methods.

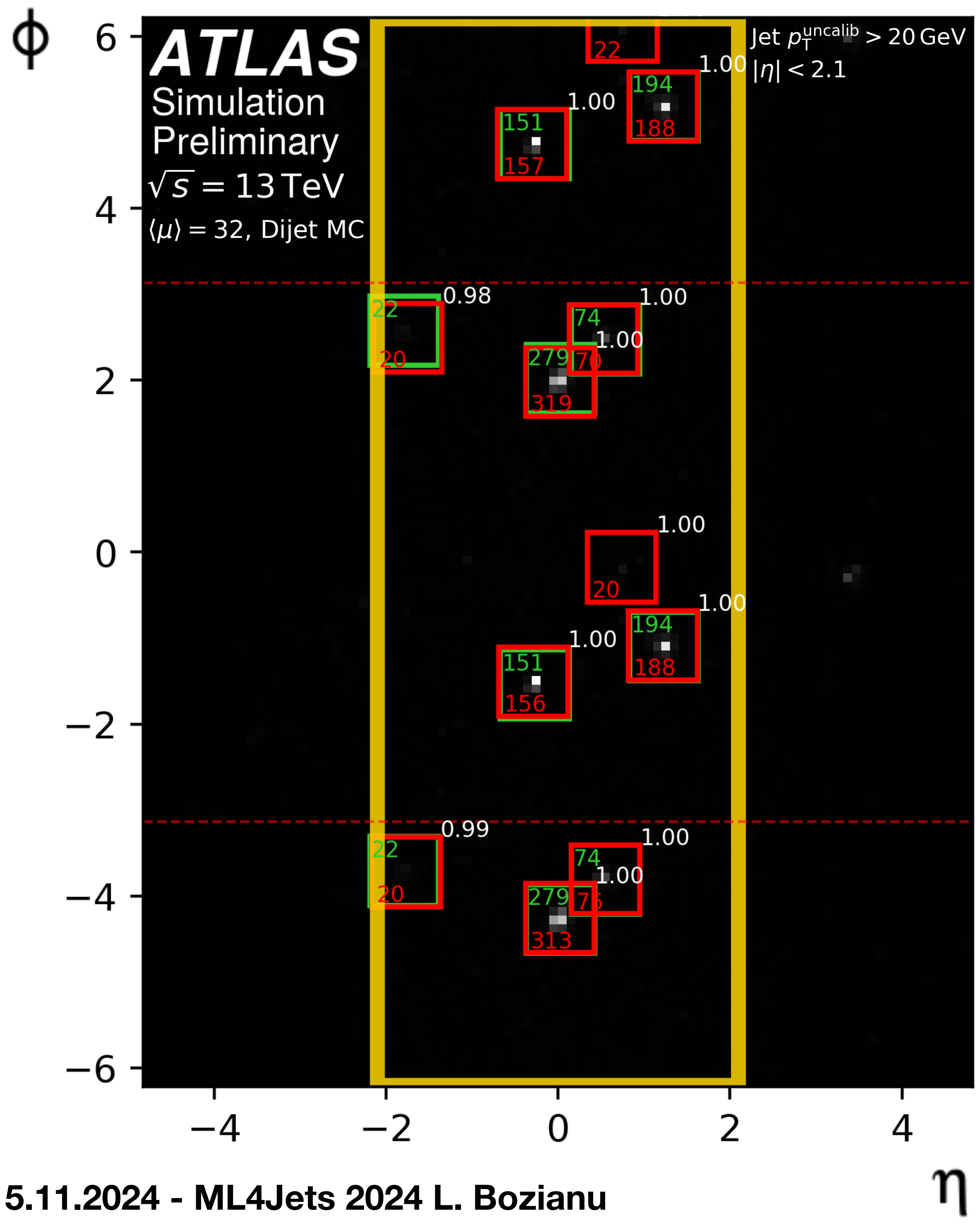
Conclusion

- We can use CNNs to approximate jets in the calorimeter.
- The complexity of the model can be reduced significantly, with respect to the SSD literature, without a loss in performance.
  - We don’t need to use million-parameter models! caloJetSSD *700* times smaller.
- Promising trigger efficiencies for simple jet hypotheses.
- Robust against pile-up, still performant in HL-LHC conditions.
- Order of magnitude speed-up over current iterative methods.

Thanks for your attention

Backup

Jet Detection for a single event with $\langle\mu\rangle = 32$



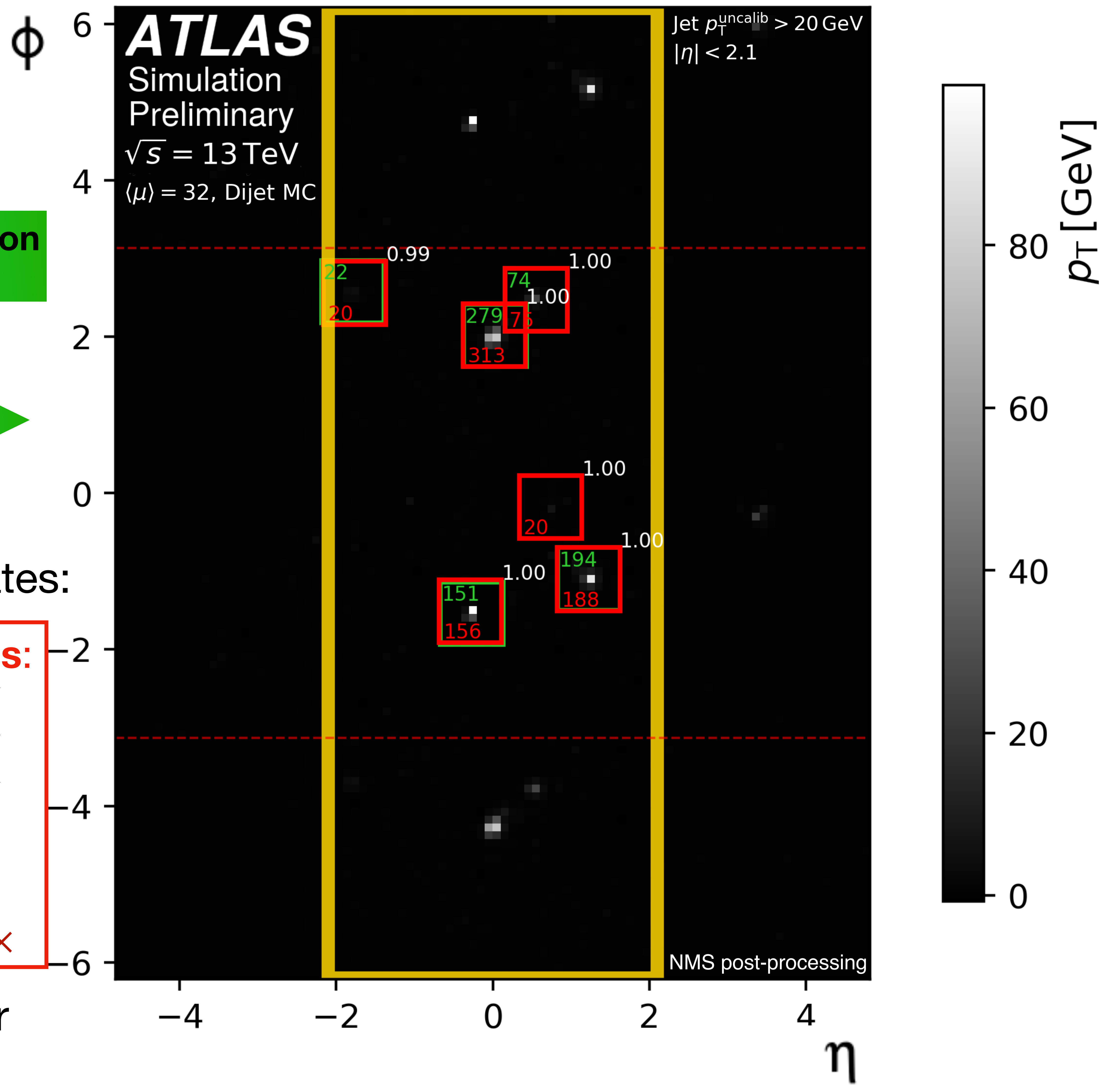
Non-Maximum Suppression post-processing

Uncalibrated p_T estimates:

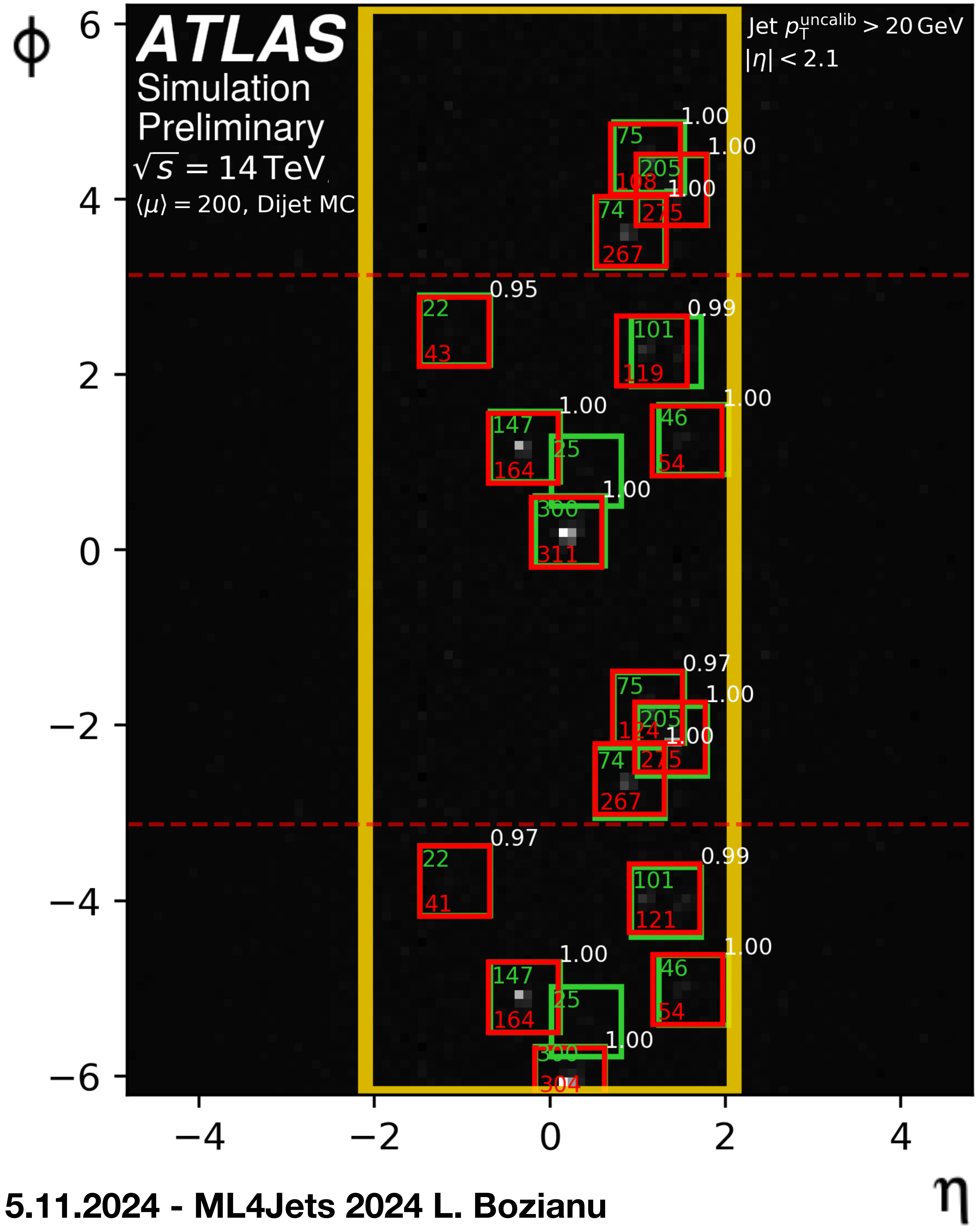
Targets:
- 279 GeV
- 194 GeV
- 151 GeV
- 74 GeV
- 22 GeV

Predictions:
- 313 GeV
- 188 GeV
- 156 GeV
- 75 GeV
- 20 GeV
- 20 GeV ×

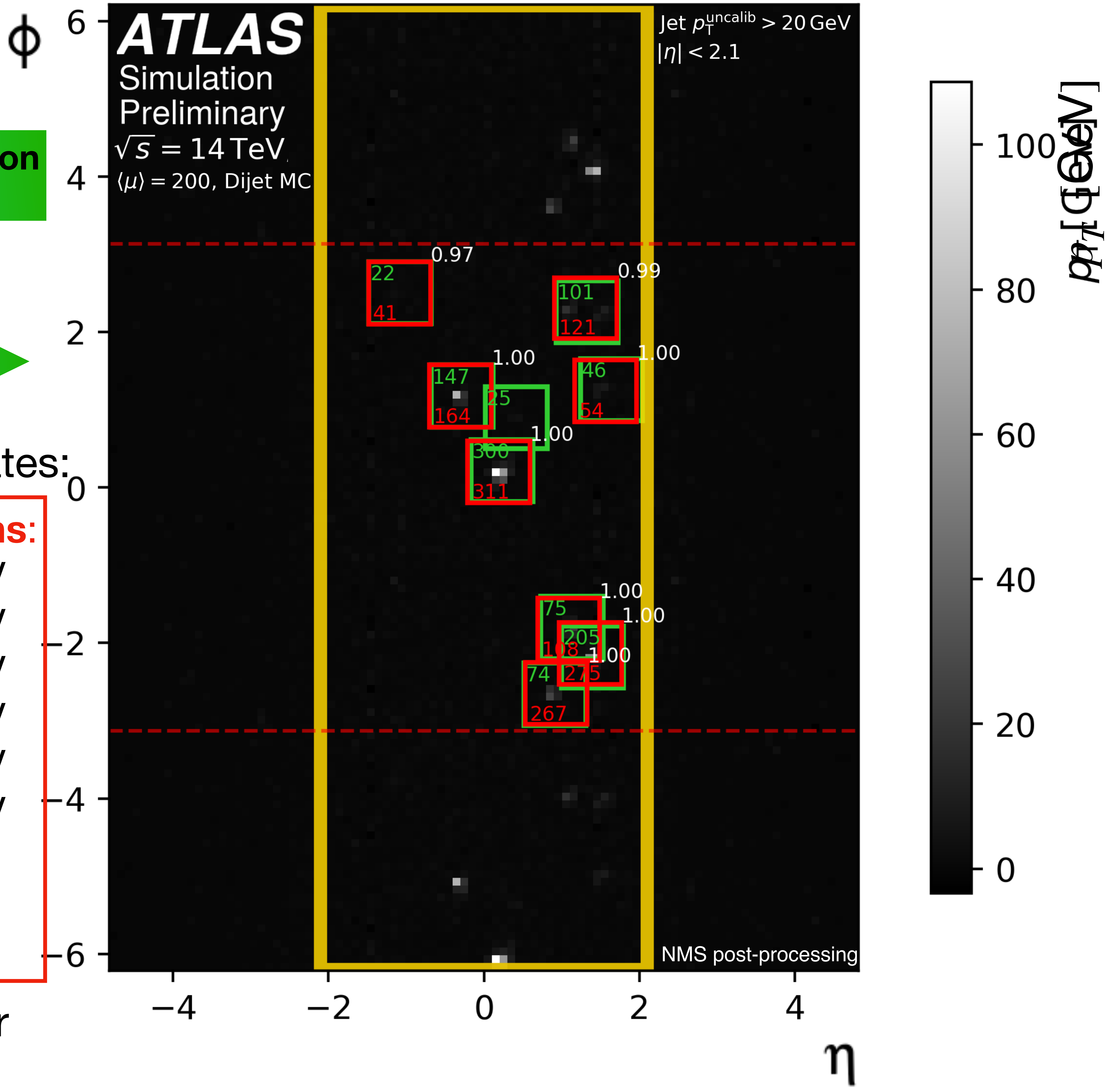
Using **sumpool** layer



Jet Detection for a single event with $\langle \mu \rangle = 200$



Non-Maximum Suppression post-processing



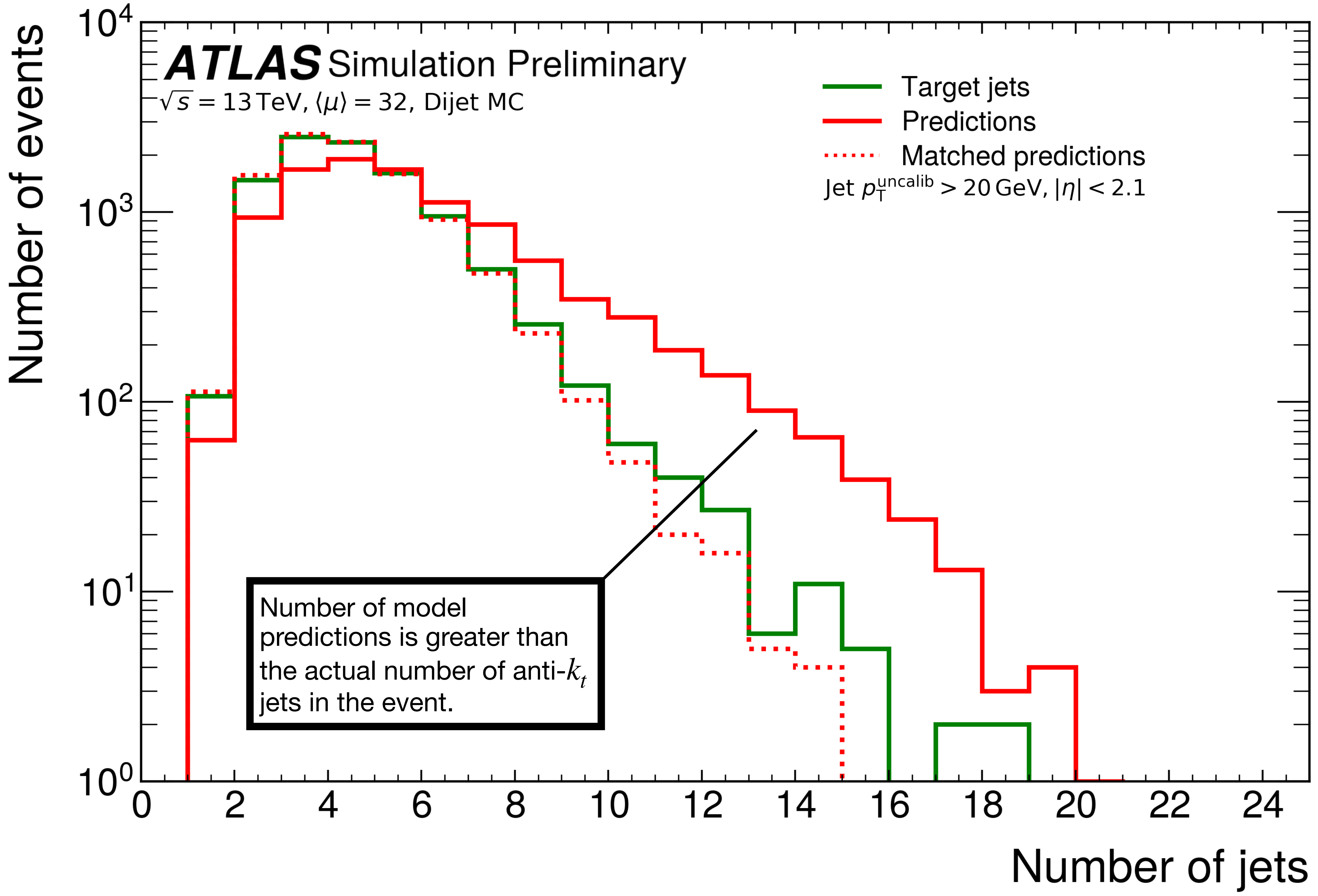
Uncalibrated p_T estimates:

Targets:	**Predictions:**
300 GeV	311 GeV
205 GeV	275 GeV
147 GeV	164 GeV
101 GeV	121 GeV
75 GeV	108 GeV
74 GeV	267 GeV
46 GeV	54 GeV
25 GeV	×
22 GeV	41 GeV

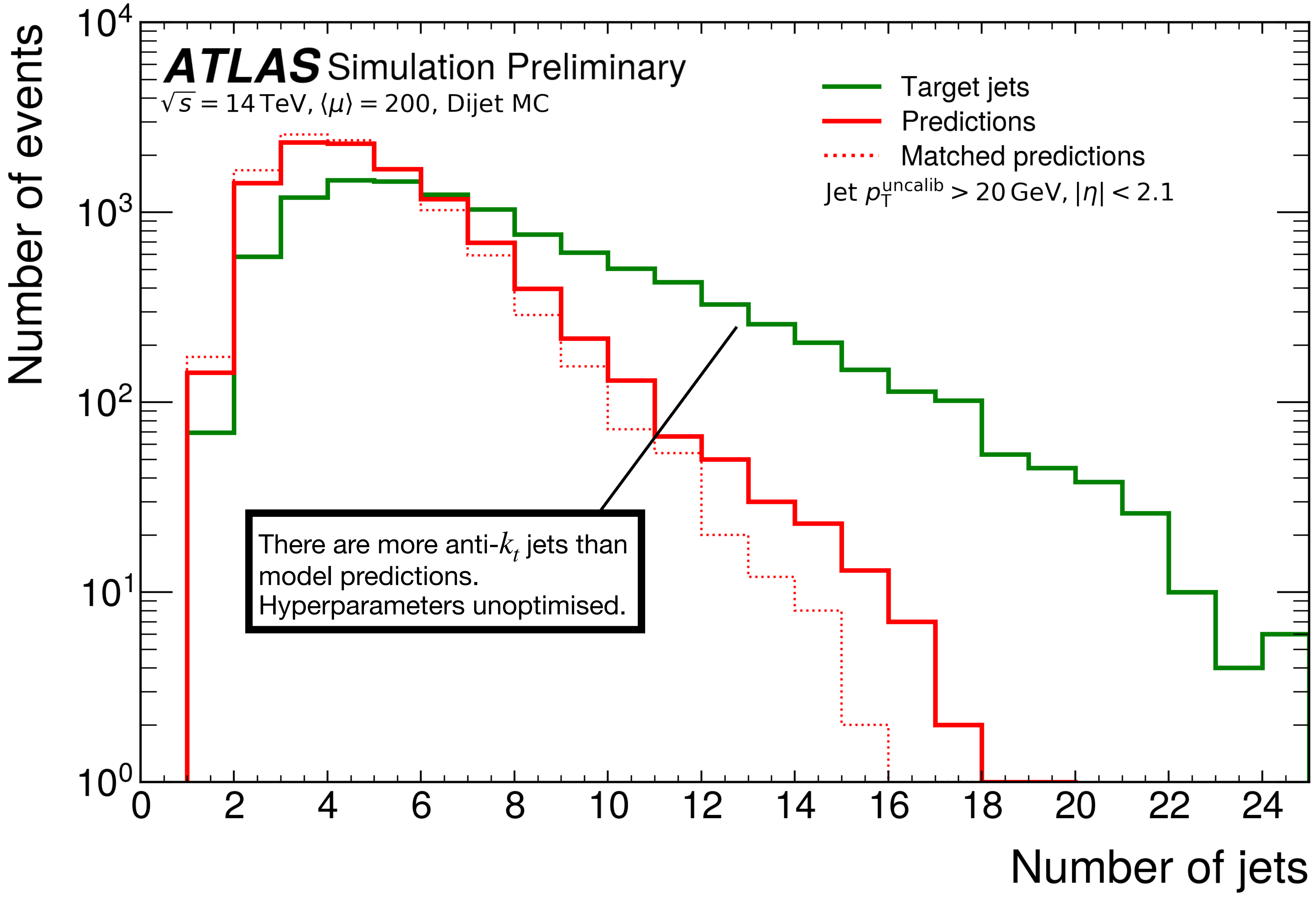
Using **sumpool** layer

Jet & Prediction Multiplicities

*LHC **Run 2-like** conditions*



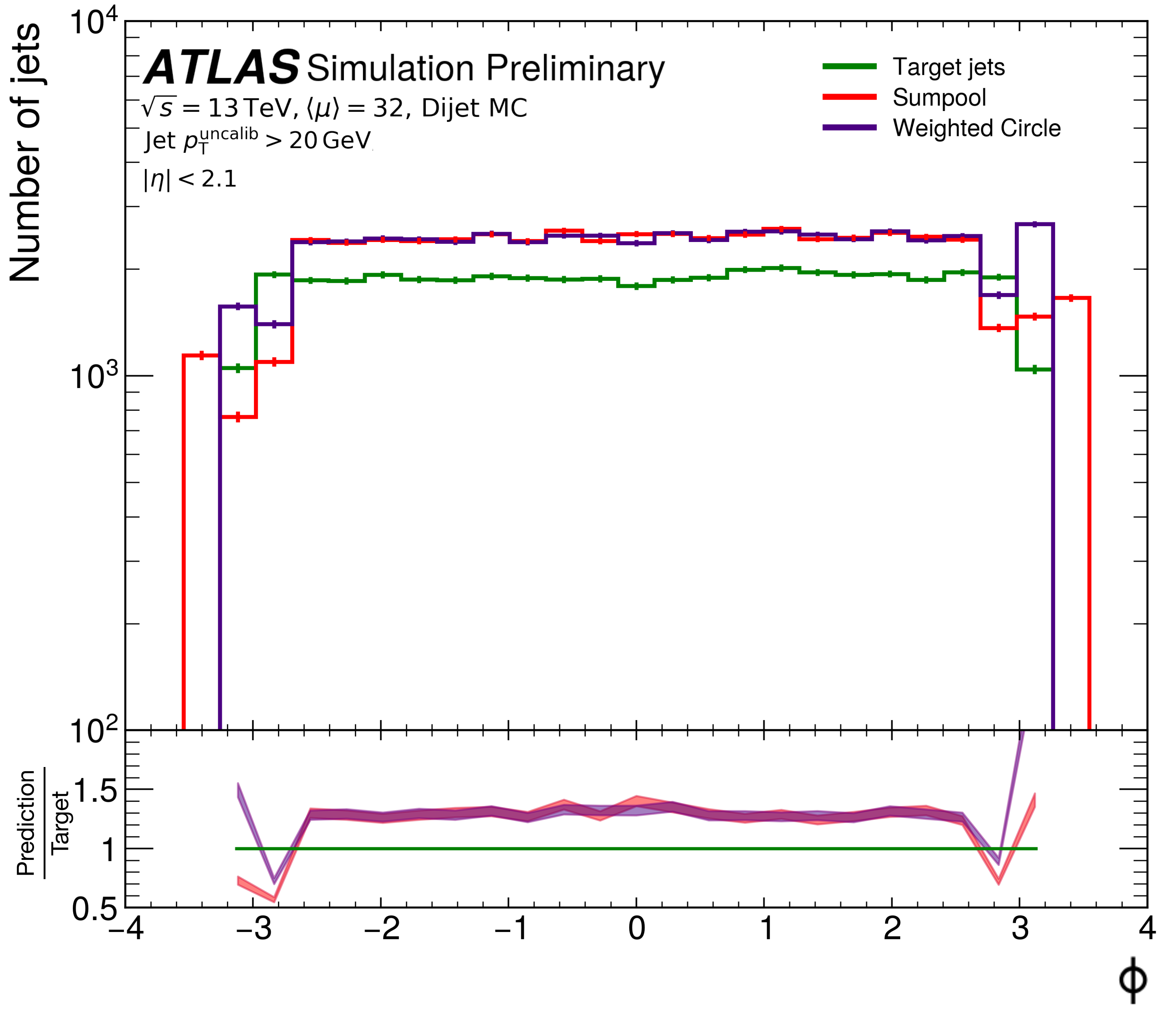
*HL-LHC **high pile-up** conditions*

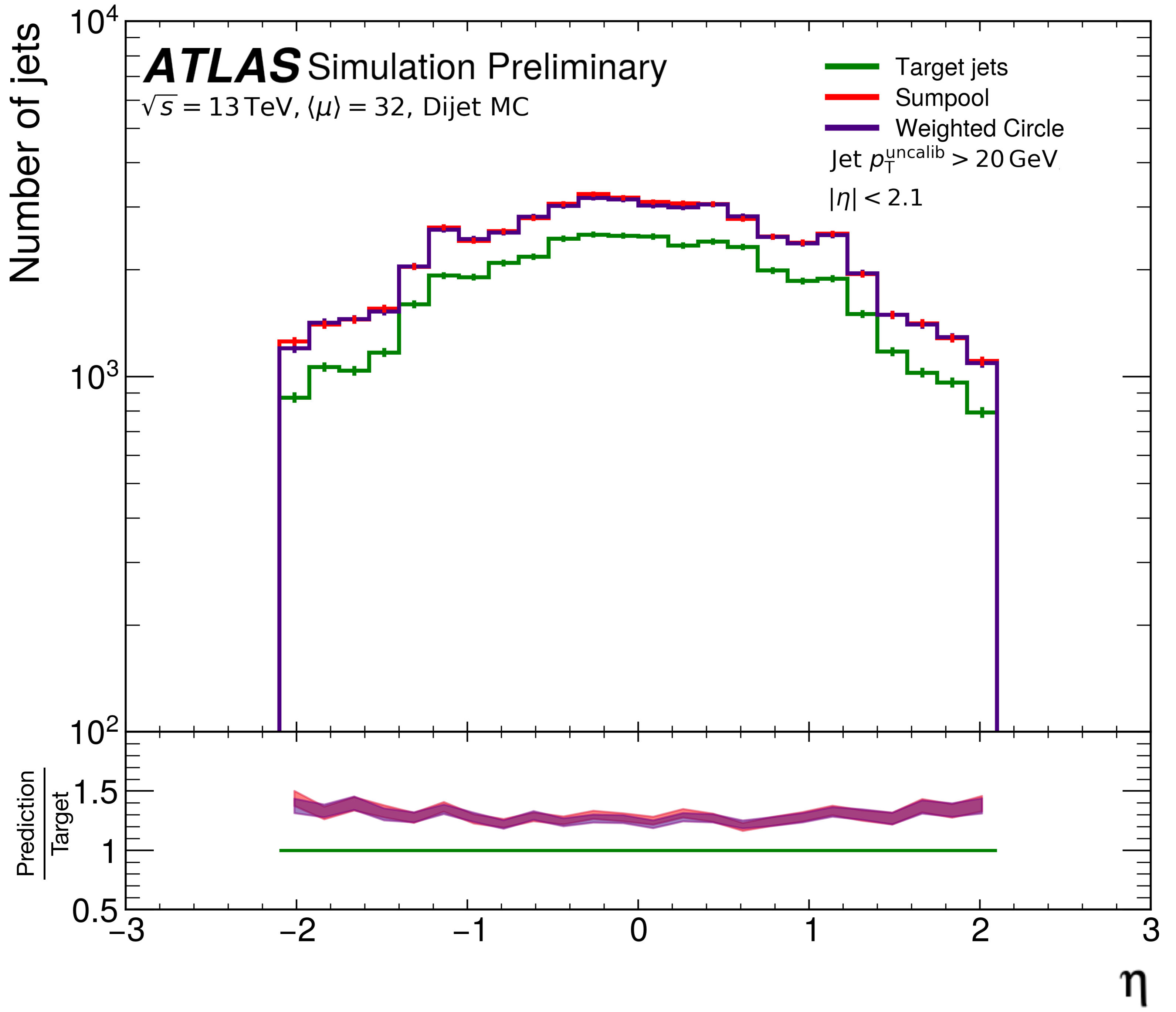


Jet Direction

Comparing to online anti-k_t algorithm

- The angular distributions for the entire test set of 10,000 events.
- Run 2-like conditions, 32 pile-up interactions on average.
- Compare sumpool and weighted circle method to anti-k_t algorithm.
- Sumpool: Geometric centre of the prediction.
- Weighted Circle: Energy weighted mean of cells.

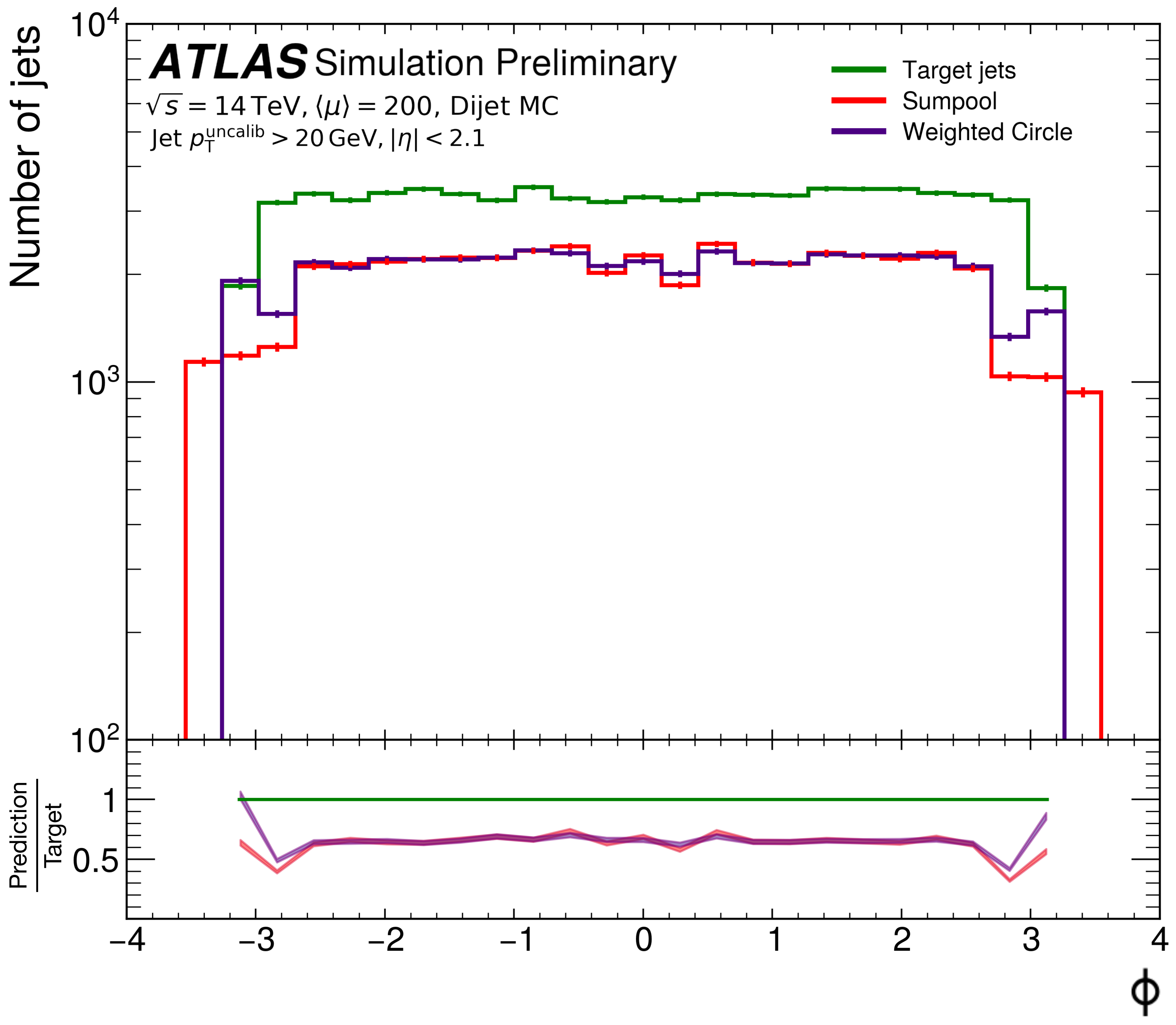


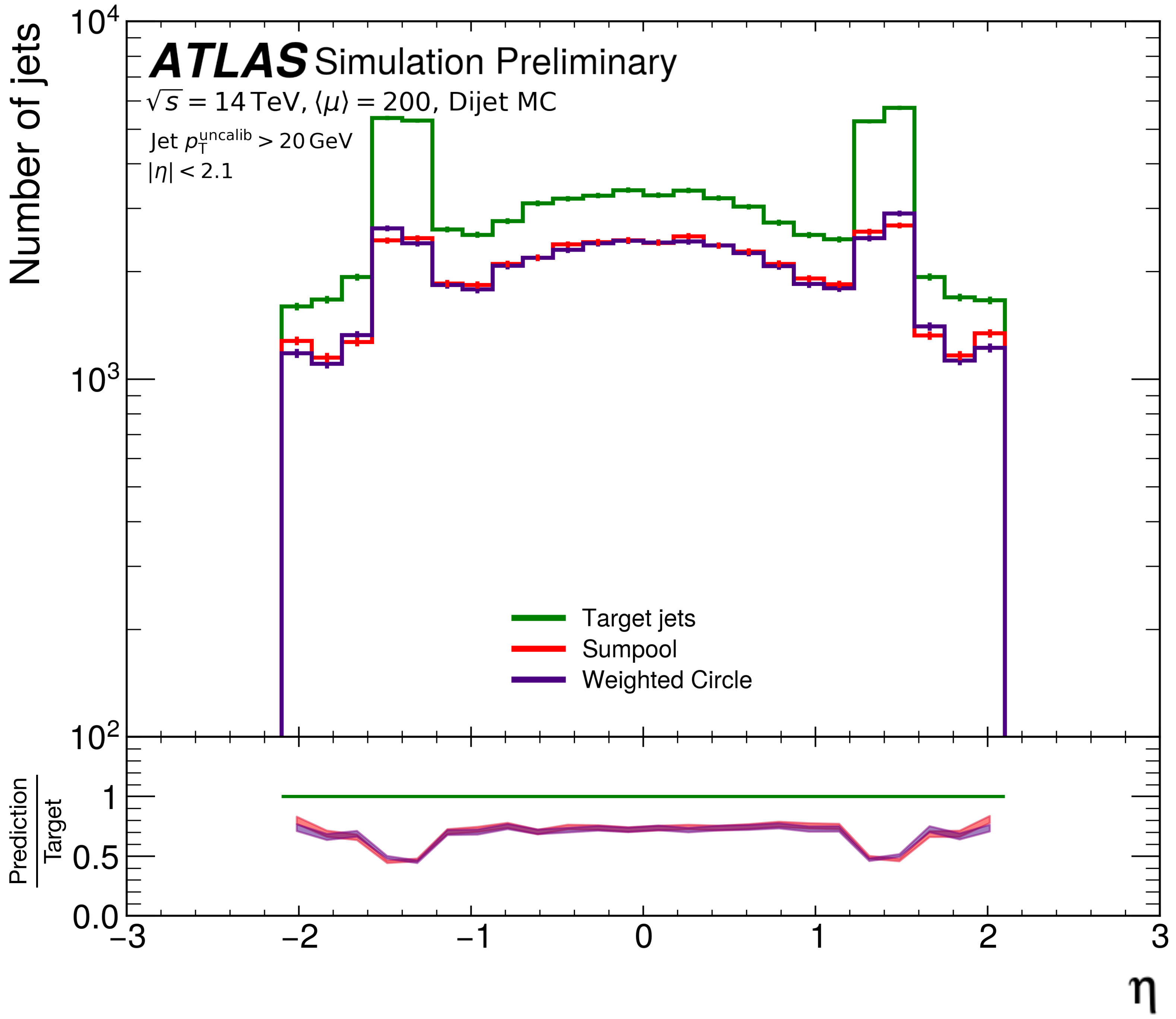


Jet Direction

Comparing to online anti-k_t algorithm

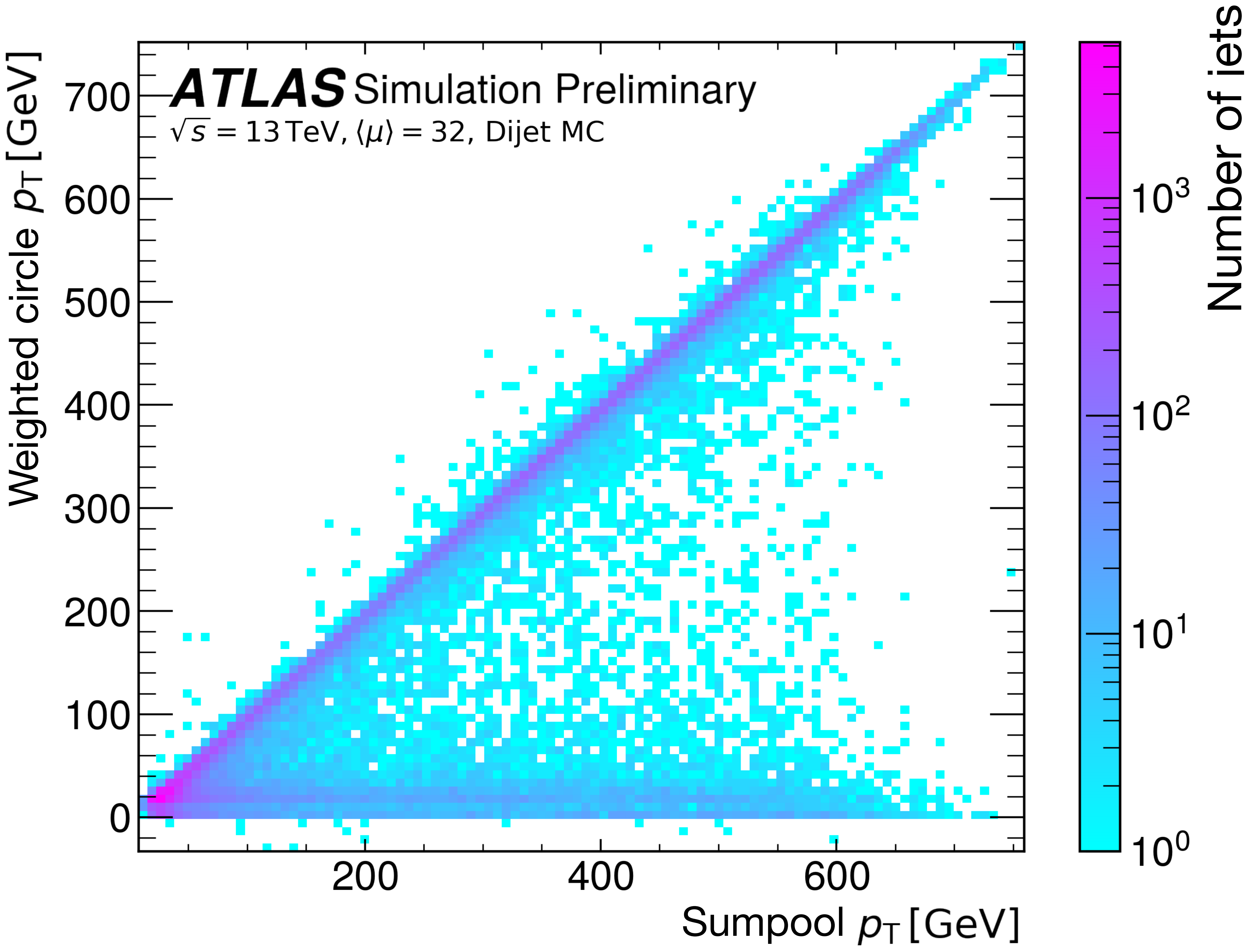
- The angular distributions for the entire test set of 10,000 events.
- Run 4-like conditions, 200 pile-up interactions on average.
- Compare sumpool and weighted circle method to anti-k_t algorithm.
- Sumpool: Geometric centre of the prediction.
- Weighted Circle: Energy weighted mean of cells.





Comparing p_T methods

- **Sumpool method**: Sumpool output of the network.
  - 9x9 window centred on jet.
  - Vulnerable to overlapping jets
- **Weighted circle method**: Weighted circle.
  - Retrieve cells in $R = 0.4$ circle centred on each prediction.
  - Share p_{T} among overlapping predictions.



Timing Evaluation

Comparing to online anti-k_t algorithm?

- Timing estimates for current model implementation.
- Pre- and post-processing executed on single CPU (AMD EPYC 7742 CPU).
- Model inference and data transfer with one NVidia RTX 2080 Ti GPU.
    - Includes transfer calorimeter image to GPU, a single forward pass, output transfer to CPU and a decoding of the output.
    - Model size no longer limiting latency, rather the size of the input image.

Preprocessing

8.1 ± 4.3ms

Model inference*

4.4 ± 1.5ms

(Optional post-processing)
Weighted circle method

11.3 ± 4.9ms

*data transfer + decoding