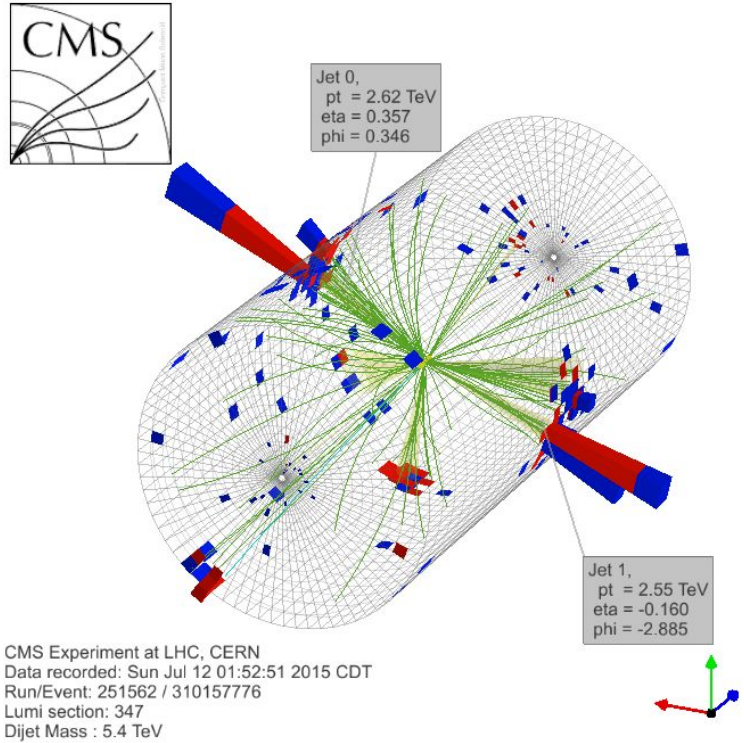
University of Zurich[UZH]

ABCNet: Jet-tagging and part segmentation

Vinicius M. Mikuni, Florencia Canelli

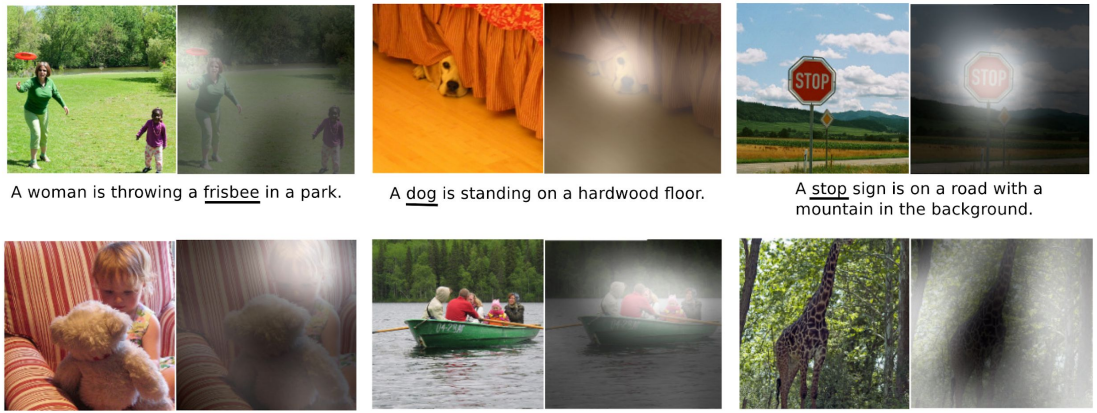
Introduction

- How do we **best extract information** out of a data set?
- Not a straightforward answer:
  - **What kind of information we want?**
  - **What is the data format?**
- Is there a more **natural approach** to treat **collider data?**
- **Point clouds:** set of **unordered** data points, in a well defined space, **representing a system**



ABCNet: Easy as 1-2-3

- Enters **A**ttention-**B**ased **C**loud **Net** (**ABCNet**) [arXiv:2001.05311]
- Uses the flexibility of **point clouds** with **attention mechanisms**
- Based on the GAPNet implementation
- How does it work?



A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

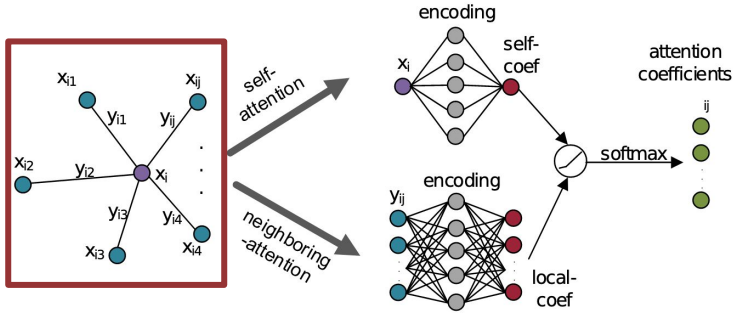
A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

Attention: Let the **method learn** the **relevant parts** for the task at hand (like the **bold text** I'm using in this presentation)

GAPLayers

- The core component of ABCNet are Graph attention pooling layers **GAPLayers**
- Start with a particle $\mathbf{x_i}$ (node)
- Take the **k-nearest neighbors** (in η-ϕ space or even in the latent space)
- Define the **edge features** y_{ij}
- **Encode the nodes and edges**
- Create **self-** and **local-attention coefficients**
- Merge
- **Align** the coefficients using **Softmax**
- Create **1 coefficient per node**

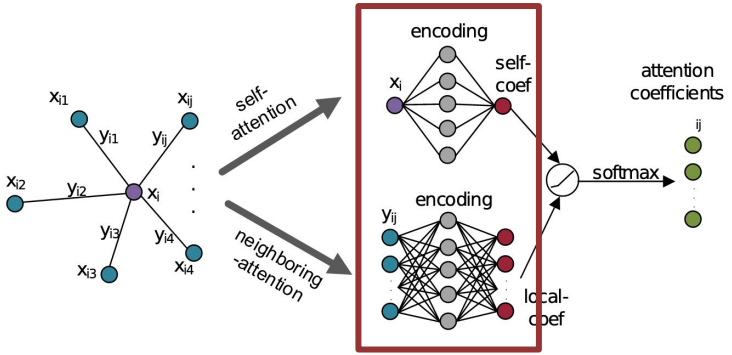


Nodes: $\mathbf{x_i}$

Edge features: $\mathbf{y_{ij}} = X_i - X_{ij}$

GAPLayers

- The core component of ABCNet are Graph attention pooling layers **GAPLayers**
- Start with a particle $\mathbf{x}_i$ (node)
- Take the **k-nearest neighbors** (in η-ϕ space or even in the latent space)
- Define the **edge features** y_{ij}
- **Encode the nodes and edges**
- Create **self-** and **local-attention coefficients**
- Merge
- **Align** the coefficients using **Softmax**
- Create **1 coefficient per node**



Encode the **nodes** (and **edges**) by passing it to a **2 layers NN** with **output size F** and **1**

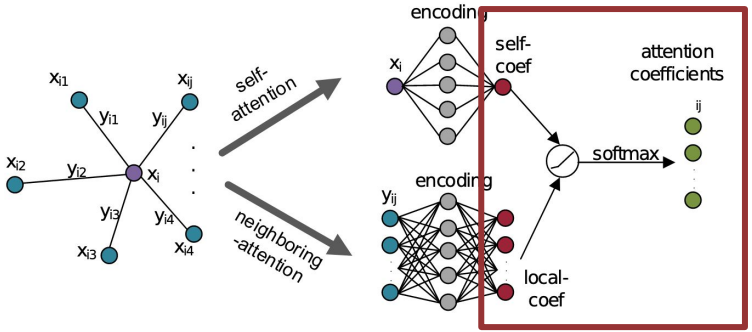
Self-attention: encoded nodes $\mathbf{x'}_i$

Local-attention: encoded edge $\mathbf{y'}_{ij}$

GAPLayers

- The core component of ABCNet are Graph attention pooling layers **GAPLayers**
- Start with a particle $\mathbf{x_i}$ (node)
- Take the **k-nearest neighbors** (in η-ϕ space or even in the latent space)
- Define the **edge features** y_{ij}
- **Encode the nodes and edges**
- Create **self-** and **local-attention coefficients**
- Merge
- **Align** the coefficients using **Softmax**
- Create **1 coefficient per node**

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_i}}$$



Merge all the **coefficients** and pass the result to a **nonlinear function**

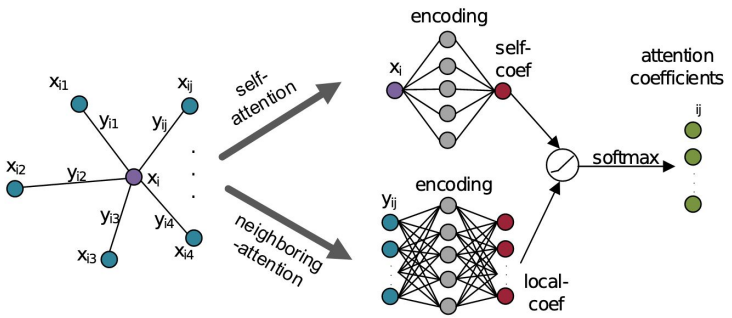
$\mathbf{c_{ij}}$ = LeakyRelu(x'_i + y'_{ij})

Align $\mathbf{c_{ij}}$ with softmax

$\mathbf{c'_{ij}}$ = S(c_{ij})

GAPLayers

- The core component of ABCNet are Graph attention pooling layers **GAPLayers**
- Start with a particle $\mathbf{x_i}$ (node)
- Take the **k-nearest neighbors** (in η-ϕ space or even in the latent space)
- Define the **edge features** y_{ij}
- **Encode the nodes and edges**
- Create **self-** and **local-attention coefficients**
- Merge
- **Align** the coefficients using **Softmax**
- Create **1 coefficient per node**



Each node $\mathbf{x_i}$ has receives **1 attention feature** written as:

$\mathbf{a_i} = \mathrm{Relu}(\Sigma_j\, c'_{ij}\, y'_{ij})$

GAPLayer **outputs**:

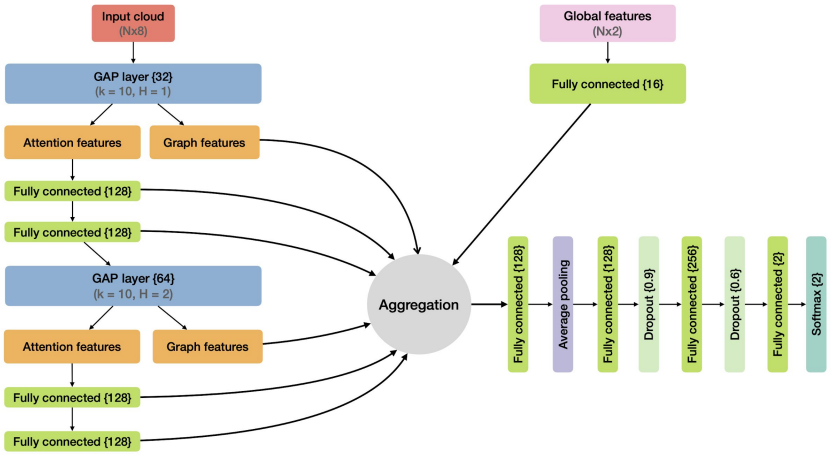
Graph features: y'_{ij}

Attention features: a_i

Applications: Classification

- **Quark-Gluon discrimination**
- Identify if a jet was quark or gluon initiated
- Use the same samples from [Energy Flow Networks]
- **Signal: Z(νν) + (u,d,s)**
- **Background: Z(νν) + g**
- Consider up to **100 particles**
- **Input variables** per particle:
  - $\Delta\eta$
  - $\Delta\phi$
  - Log p_T
  - Log E
  - Log $(p_T/p_T(\text{jet}))$
  - Log $(E/E(\text{jet}))$
  - ΔR
  - PID

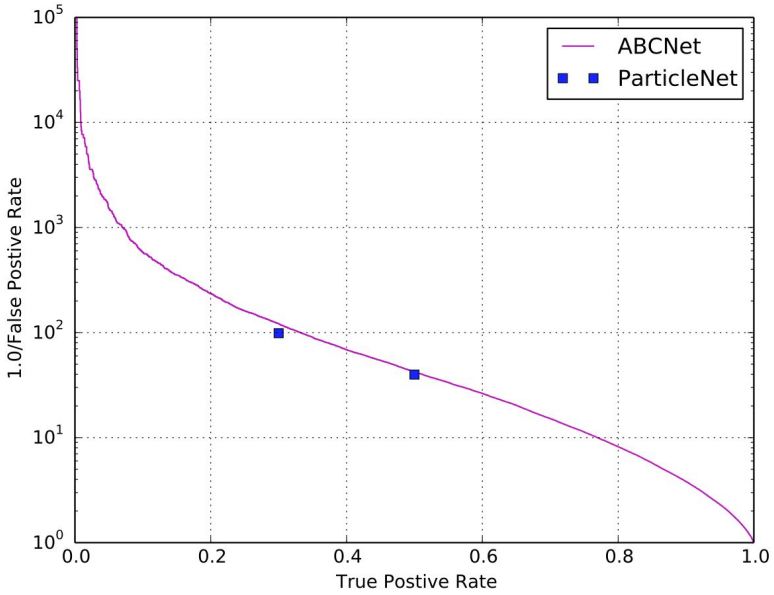
{} = Hidden layer size
k = Number of neighbors
H = number of heads



Global features: jet mass and p_T

Applications: Classification

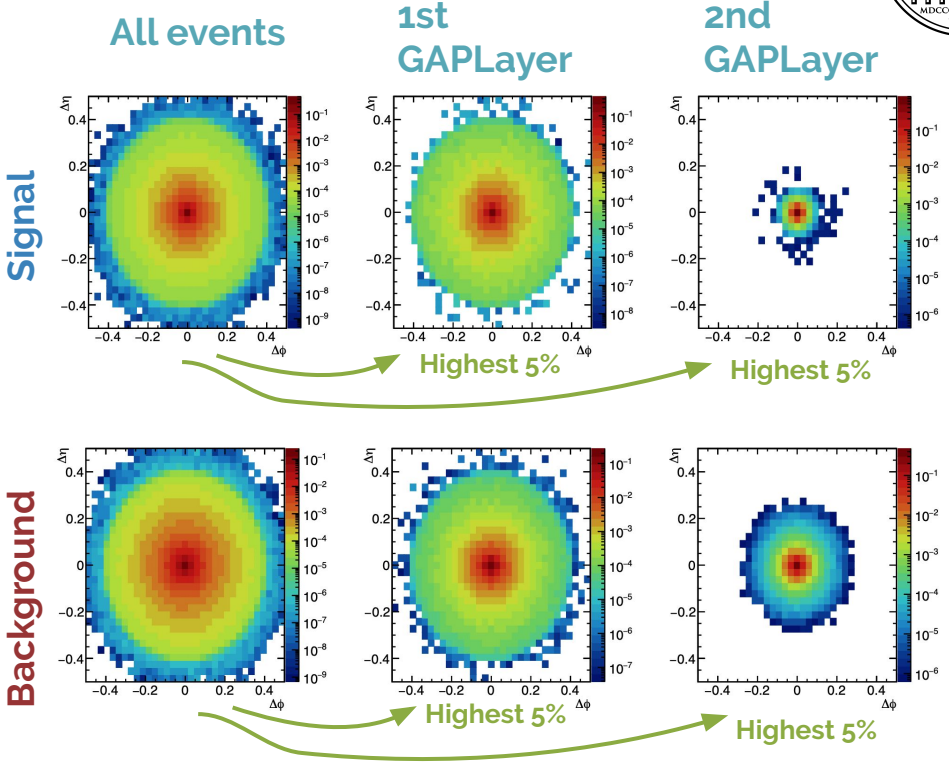
	Acc	AUC	$1/\epsilon_B$ ($\epsilon_S = 0.5$)	$1/\epsilon_B$ ($\epsilon_S = 0.3$)	Parameters
ResNeXt-50	0.821	0.960	30.9	80.8	1.46M
P-CNN	0.827	0.9002	34.7	91.0	348k
PFN	-	0.9005	34.7±0.4	-	82k
ParticleNet-Lite	0.835	0.9079	37.1	94.5	**26k**
ParticleNet	**0.840**	0.9116	39.8±0.2	98.6±1.3	366k
ABCNet	**0.840**	**0.9126**	**42.6±0.4**	**118.4±1.5**	230k



- **Same acc**. as **ParticleNet**
- **15-20% higher background rejection** for the available signal thresholds
- **40% less** trainable **parameters**

Applications: Classification

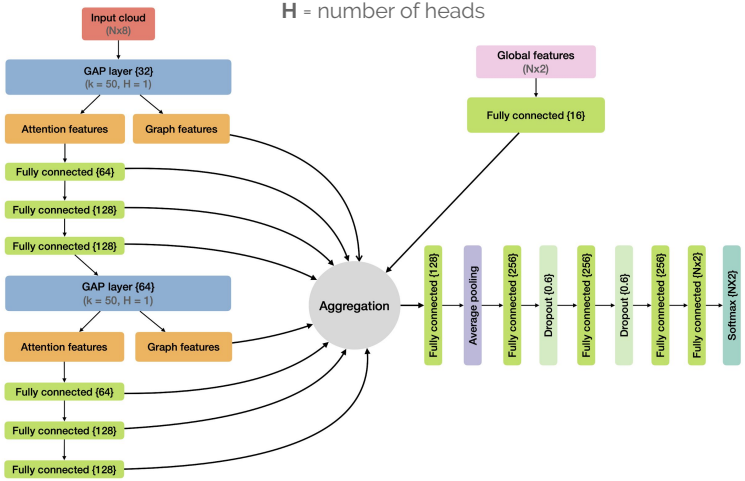
- Can we **look** at what **ABCNet** is **learning**?
  - Look at the **self-coefficients**
  - Only plot the **5% particles** inside a jet with the **highest self-coefficients**
- Particles with the **highest importance** for the classification are **closer to the axis**: **Colour factor**



Applications: Part segmentation

- **Pileup mitigation**
- **Identify if each particle** in the event **originates** from the **primary vertex**
- Use the same samples from PUMML
- **Signal:** Hypothetical particle **mϕ** with **mass 500 GeV decaying to qq**
- **Background: Soft QCD** overlaying each event
- Consider up to **500 particles**
- **Input variables** per particle:
  - η
  - ϕ
  - Log p_T
  - Charge
  - Log (p_T/p_T(jet))
  - Log (E/E(jet))
  - PUPPI weight
  - SoftKiller Flag

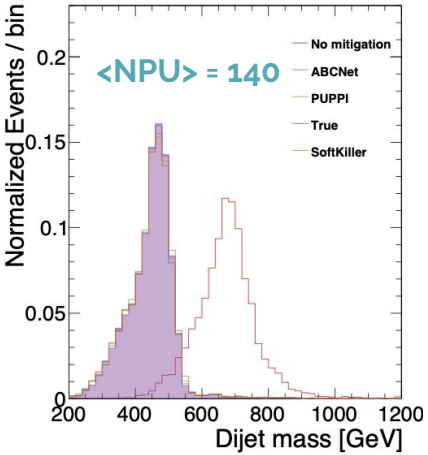
{} = Hidden layer size
k = Number of neighbors
H = number of heads

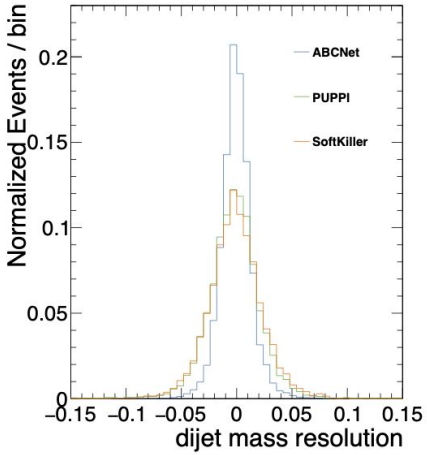


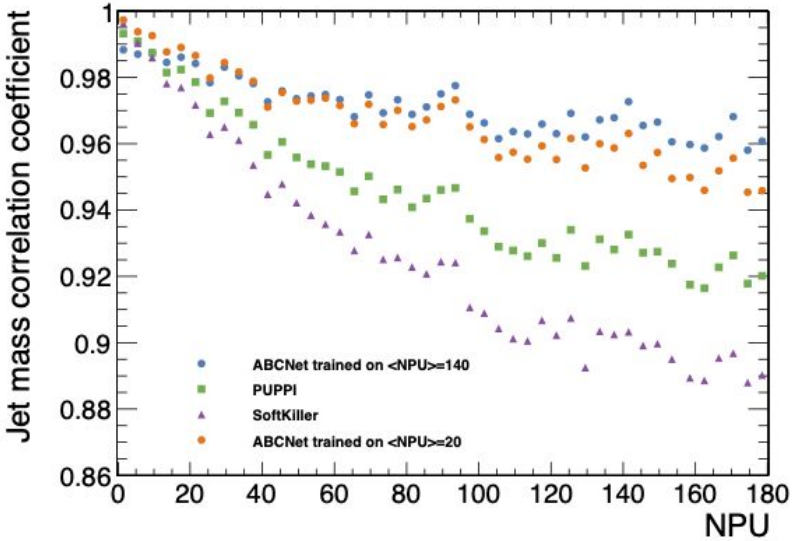
Global features: number of pileup interactions and particles in the event

Applications: Part segmentation

- ABCNet returns a **probability for each particle**
- Use the probability to **weight the 4-momentum** for each particle (Like PUPPI does)







Conclusion

- **ABCNet** uses a **graph-based approach** to collider data **enhanced by attention mechanisms**
- **State-of-the-art performance** for the tasks investigated
- The attention allows the **learning** to be **more effective**: Compact architecture
- Easy to **find** the **relevant objects** for a certain task: **Visualization**
- Applicable to other tasks: **flavour tagging, particle tracking, boosted jet identification**

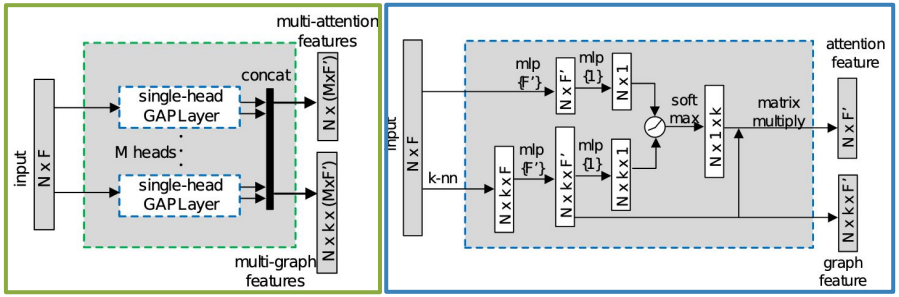
Thanks!

Any questions?

Backup

Multi-head mechanism

- To increase the stability of the network, a GAPLayer can be **repeated multiple times.**
- The results are concatenated in the end (**multi-head**)
- Apply an an **average** pooling on **all heads** to combine the results



Training details

- Pure **Tensorflow** implementation
- **Adam** optimizer
- **Learning rate**: linear **decrease** from **1e-3 up to 1e-7**
- **Batch size**: 64
- **Loss**: Cross entropy
- **Early stop**: **5 epochs** without improvement
- **Quark-Gluon:**
  - Bets epoch: **Largest accuracy**
  - Training-testing events: **1.6M/200k**
- **PU:**
  - Bets epoch: **Minimum loss**
  - Training-testing events: **44k/7k**

How PUPPI works

- **PileUp Per Particle Identification (PUPPI)**
- It takes as input particle flow objects (charged/neutral hadrons, photon and charged leptons)
  - and it gives weight for each of them.
- Defines ***α*** of each particle (*i*) using other particles (*j*) around it.

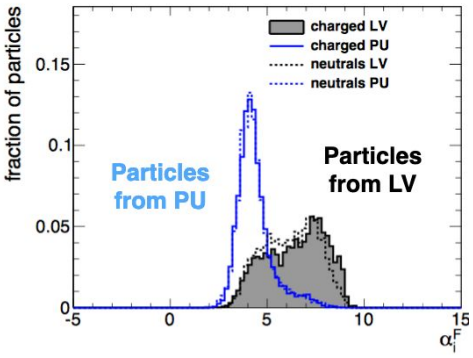
  For example

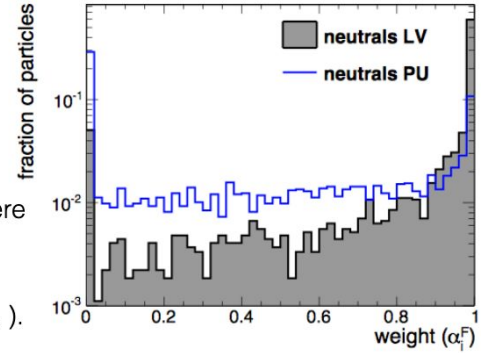
$$\alpha_i = \log \sum_{j \in \text{event}} \frac{P_T^j}{\Delta R_{ij}} \Theta(R_{\min} < \Delta R_{ij} < R_0)$$

PT sum weighted with distance

Step function to take into account only particles around it.

- Transforms the distribution of ***α*** in a **weight** (1 for particle from LV, 0 for particles from PU)
- This weight can be also defined at large rapidities where there is no coverage from the tracker.
- Then, jet reconstruction algorithm can run on the particles with the **weight** (but PUPPI is not only for jet.).

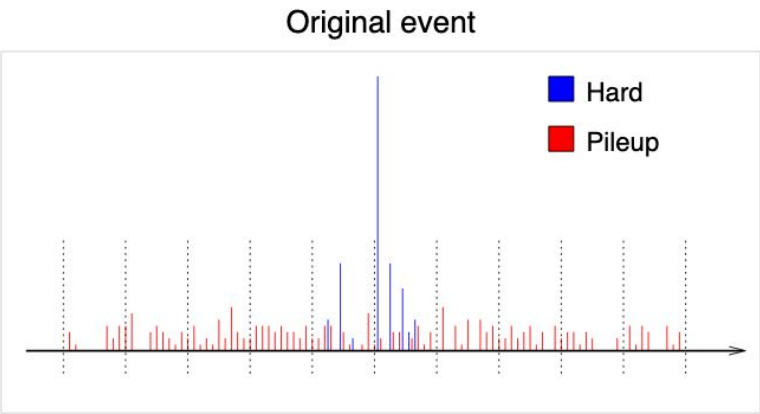


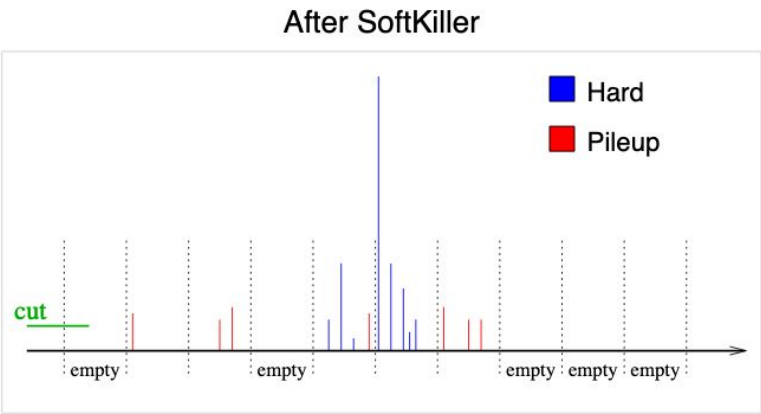


7

[Slide from BOOST 2015]

SoftKiller





Eliminates particles with pT below the minimum threshold that ensures the median of the distribution to be 0