Jet as a Particle Cloud

Loukas Gouskos, Huilin Qu

University of California, Santa Barbara

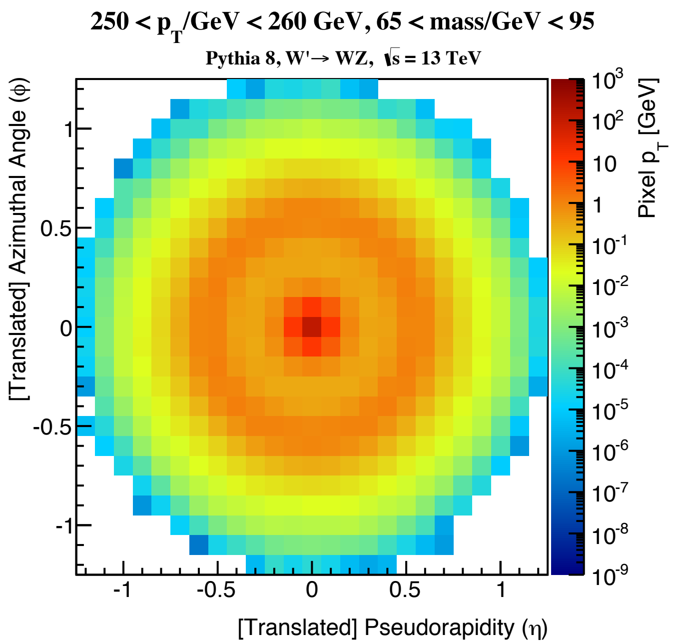
Machine Learning for Jet Physics

November 15, 2018

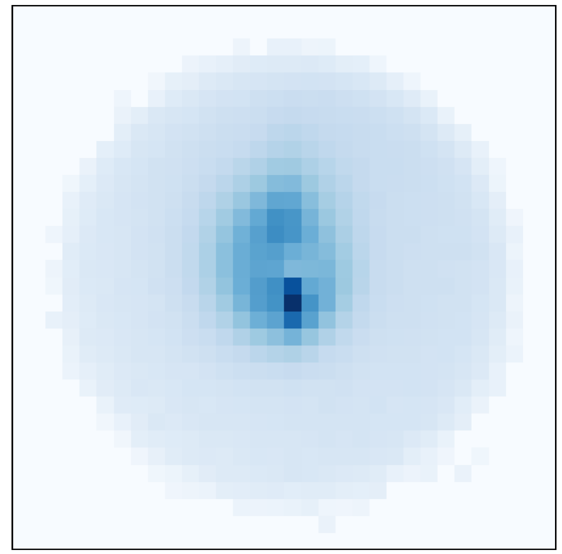
Introduction

- How to represent a jet is one of the key aspects of machine learning algorithms for jet physics
  - better representation can improve the performance/efficiency of the ML algorithms, therefore extend the reach of physics analyses
  - new representations (and ML algorithms) may lead to fresh insights into jets themselves, thus deepen our understanding of jet physics
- Lots of the approaches and techniques has been proposed in the past few years…

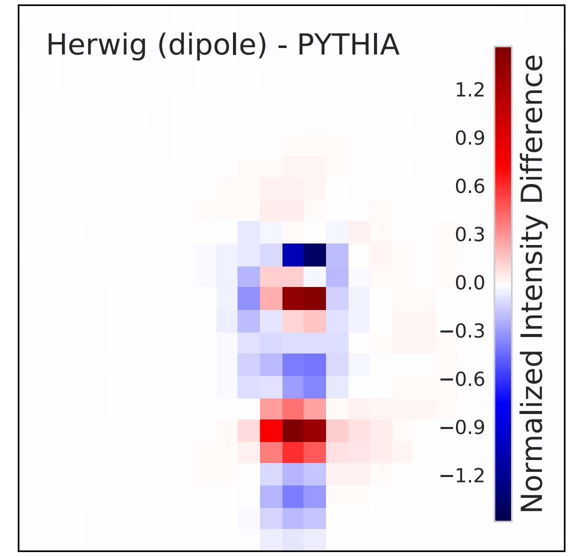
As Images…



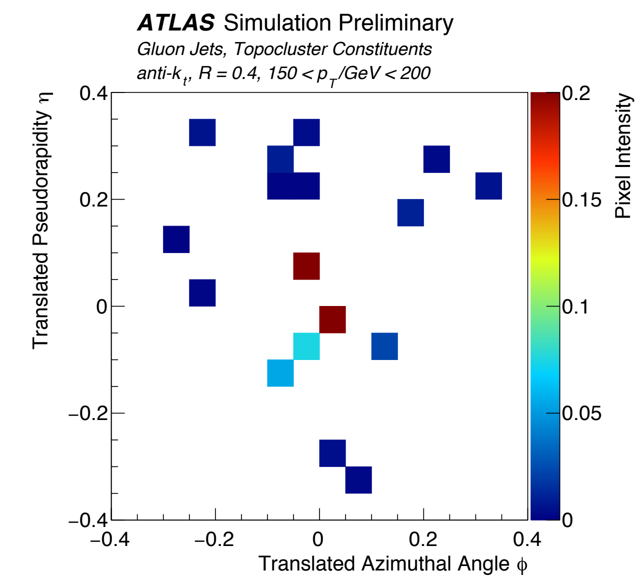
1511.05190



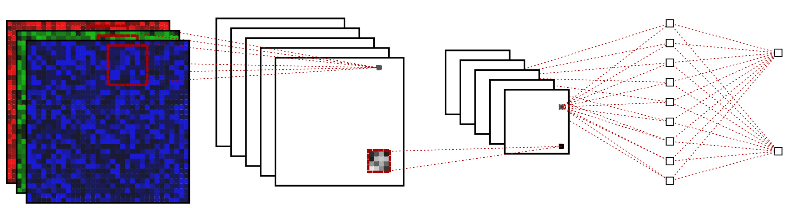
1603.09349



1609.00607



ATL-PHYS-PUB-2017-017



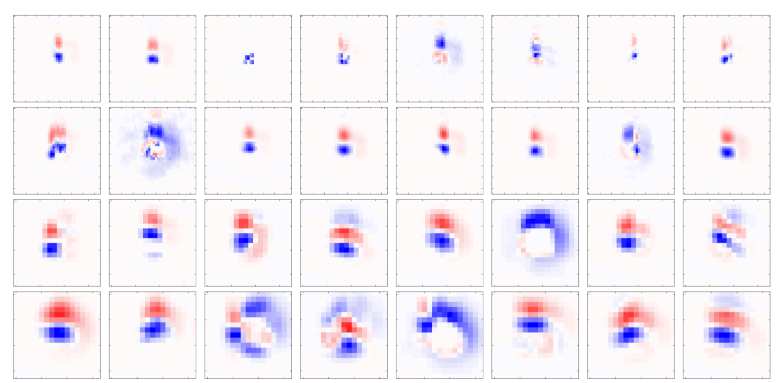
red = transverse momenta of charged particles

green = the transverse momenta of neutral particles

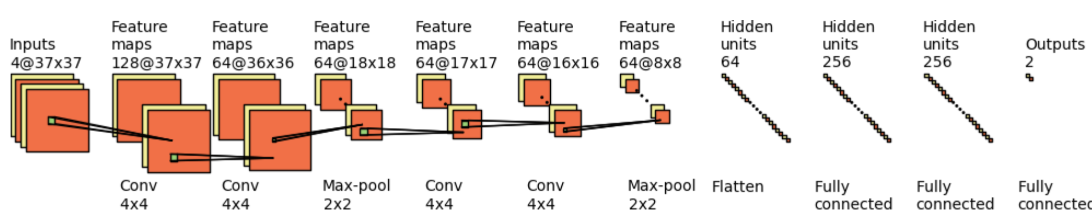
blue = charged particle multiplicity

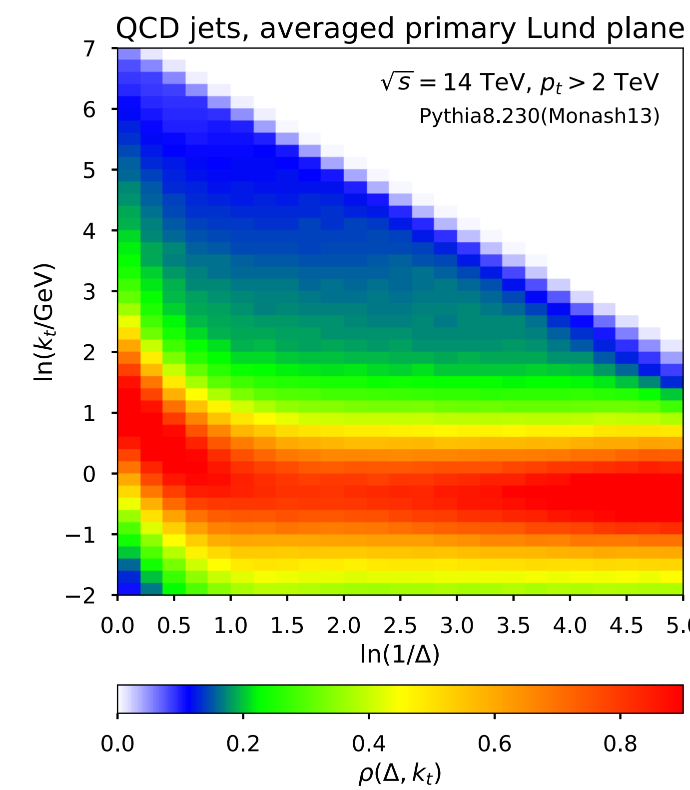
1612.01551

1701.08784



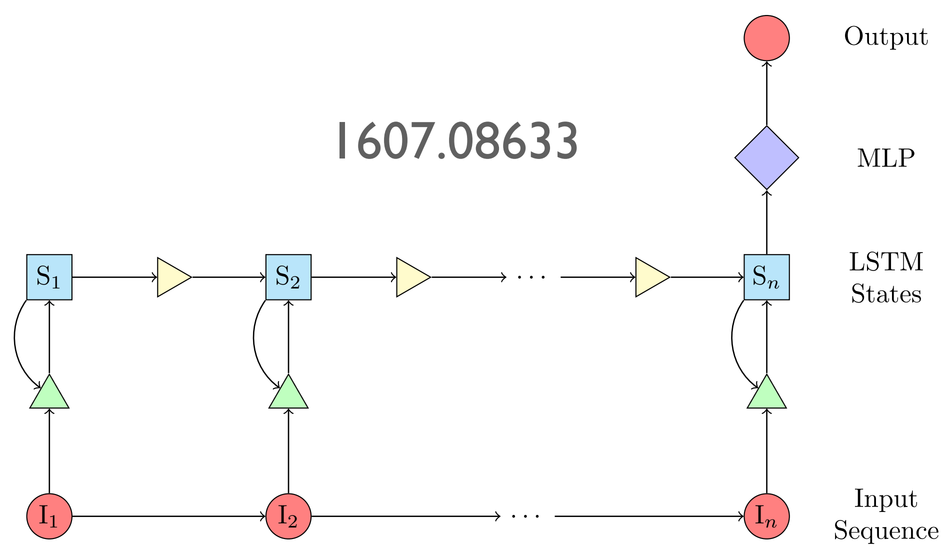
1803.00107



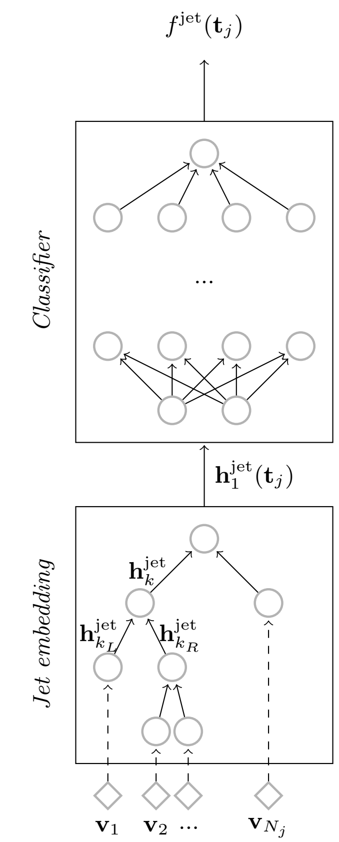


1807.04758

As Sequences…

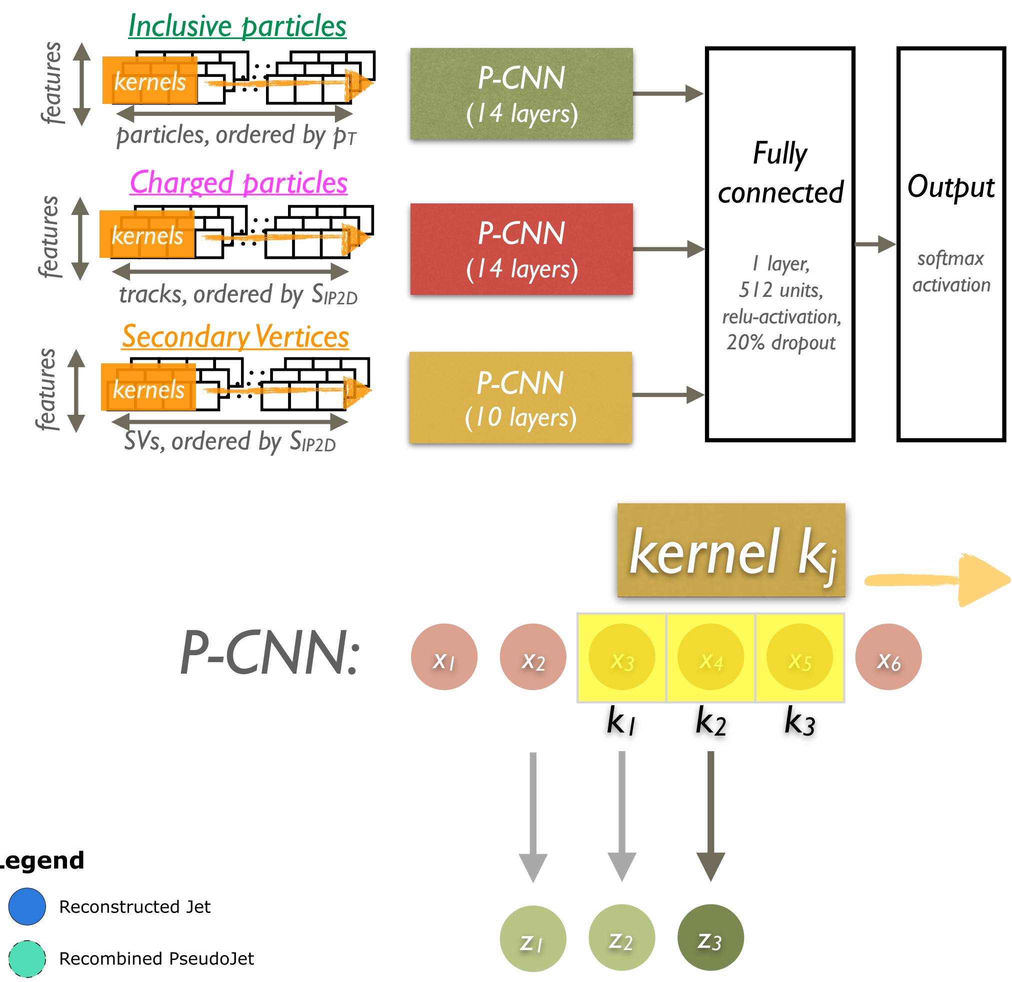


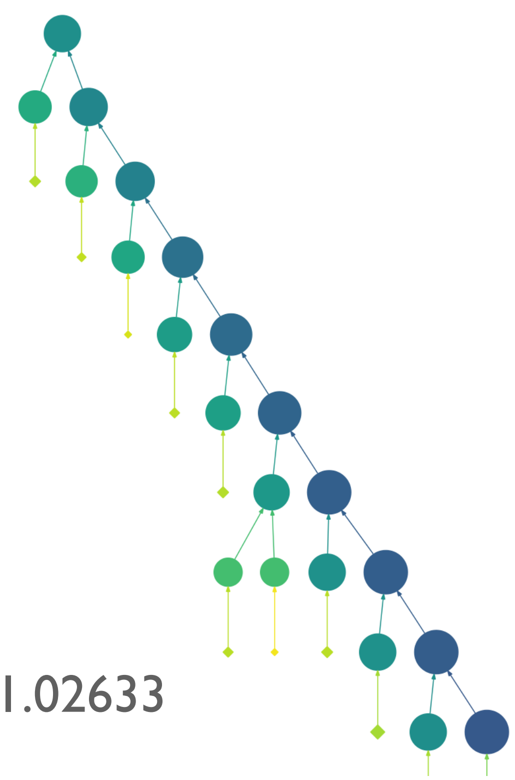
1607.08633



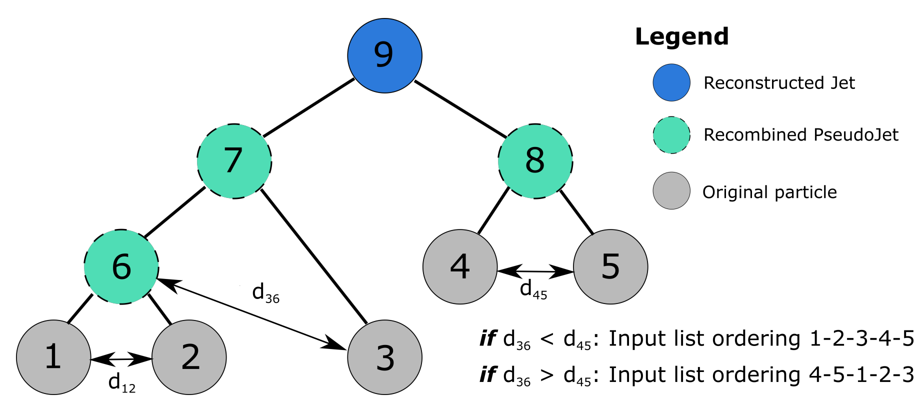
1702.00748

"DeepAK8"
CMS-DP-2017-049



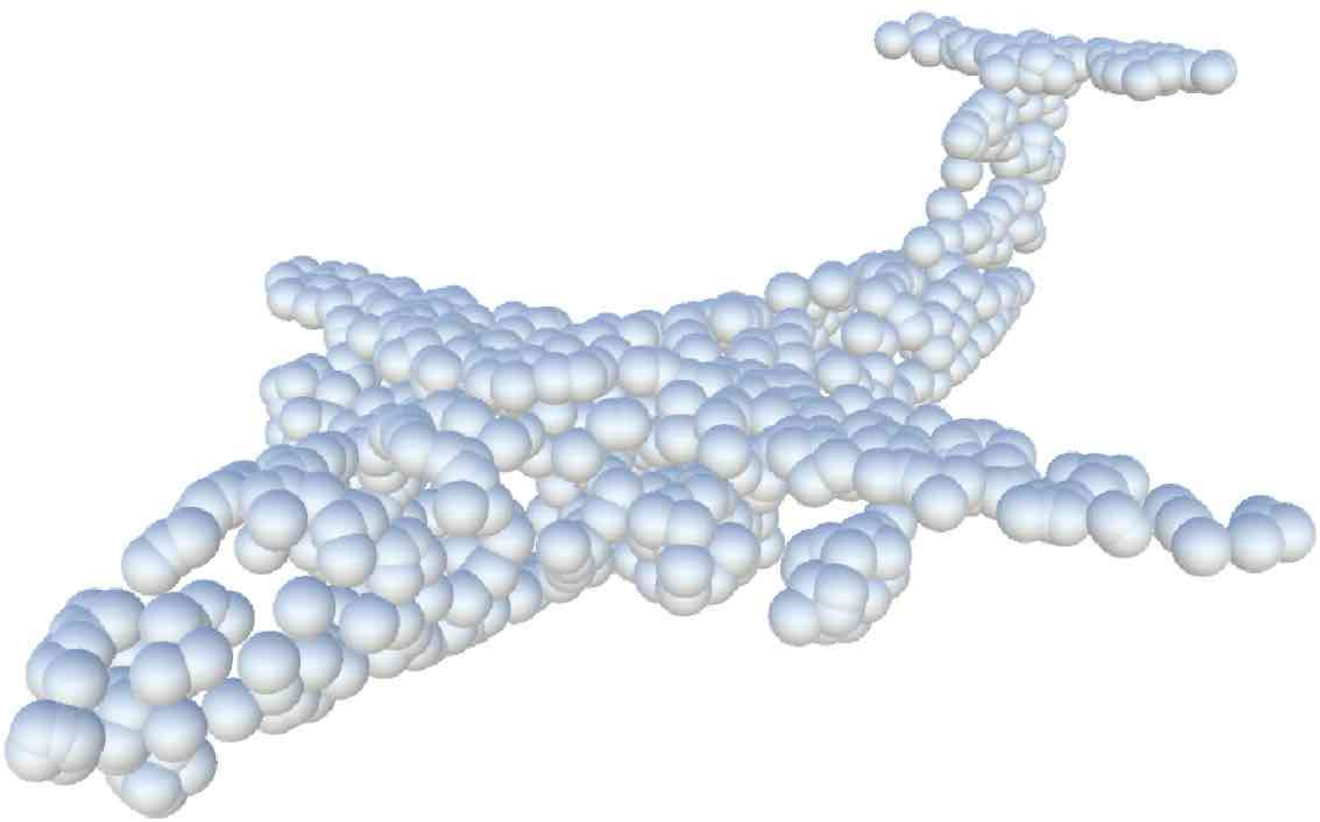


1711.02633



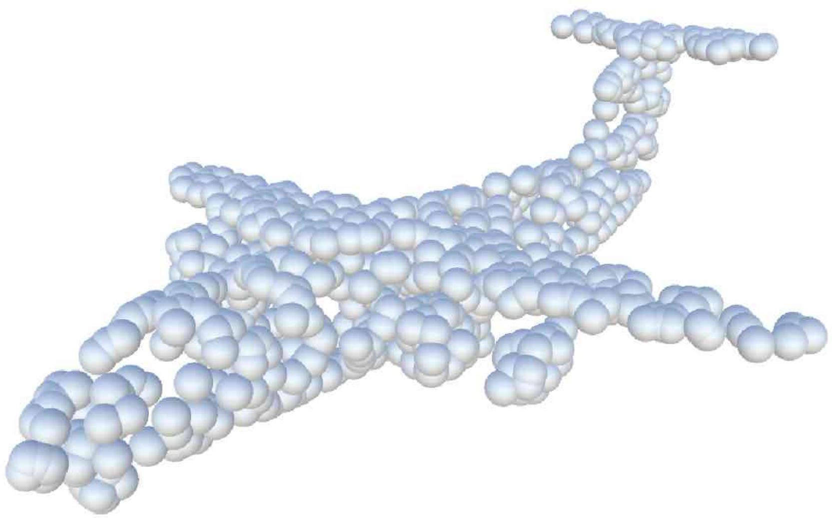
1711.09059

As… Point Clouds?



arXiv:1801.07829

As… Point Clouds?

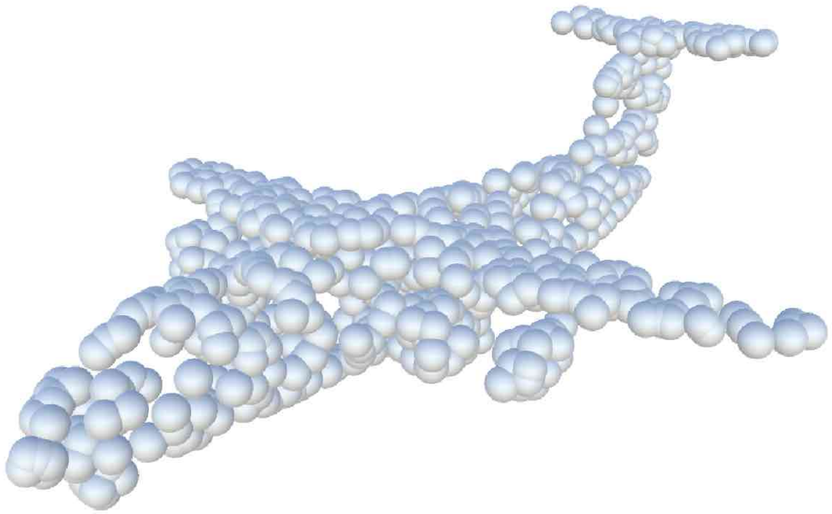


Point cloud

From Wikipedia, the free encyclopedia

A **point cloud** is a set of data points in space.

As… Point Clouds?

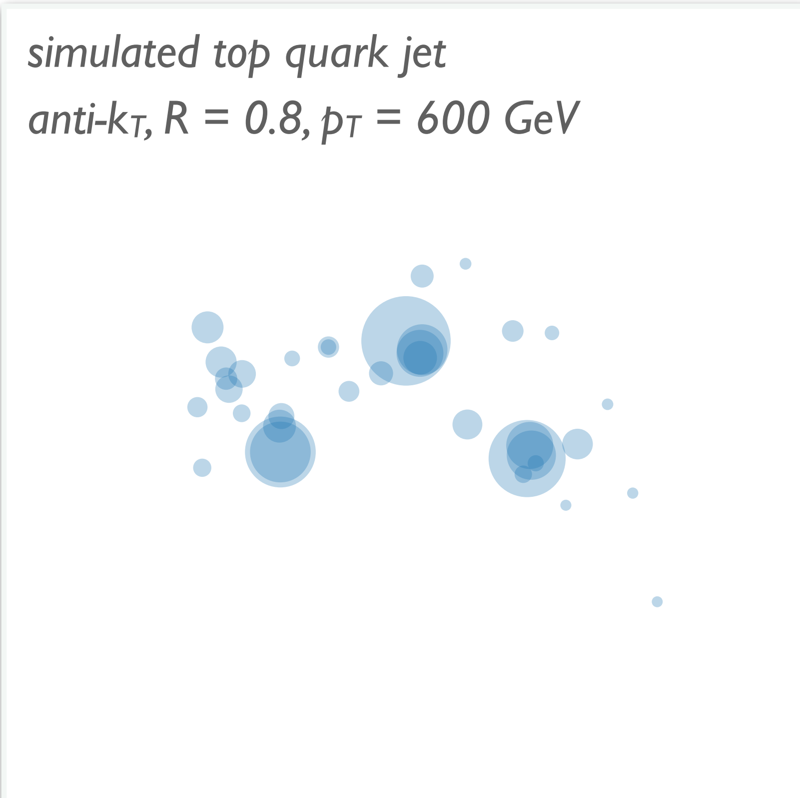


Point cloud

From Wikipedia, the free encyclopedia

A **point cloud** is a set of data points in space. Point clouds are generally produced by 3D scanners, which measure a large number of points on the external surfaces of objects around them.

Jet as a Particle Cloud

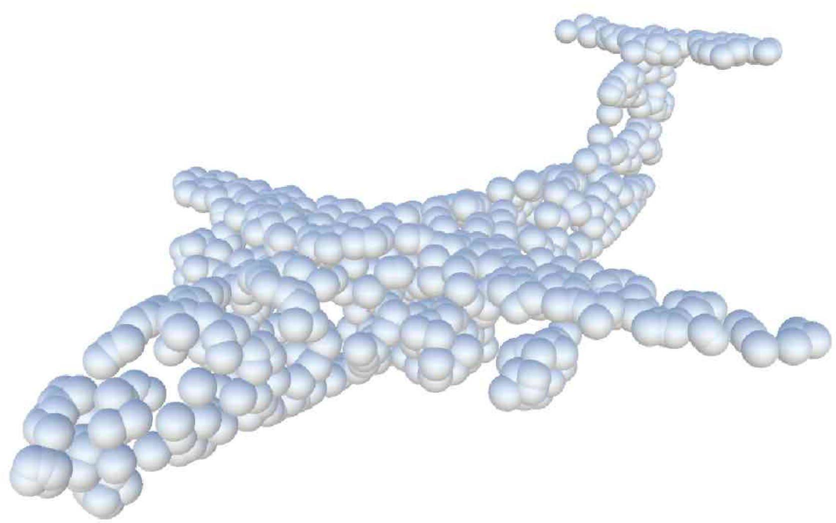


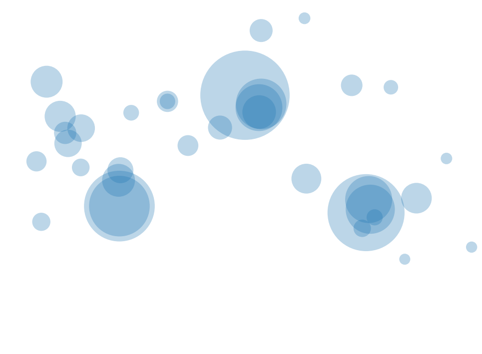
Jet (Particle cloud)

From Wikipedia, the free encyclopedia

A **jet (particle cloud)** is a set of particles in space. Particle clouds are generally created by clustering a large number of particles measured by particle detectors, e.g., ATLAS EXPERIMENT and CMS.

POINT CLOUDS VS PARTICLE CLOUDS





- Point cloud
  - points are intrinsically unordered
  - primary information:
    - 3D coordinates in the xyz space

- Particle cloud
  - particles are intrinsically unordered
  - primary information:
    - 2D coordinates in the η-φ space
  - but also additional "features":
    - energy/momenta
    - charge/particle type
    - track quality/impact parameters/etc.

Much richer content than typical point clouds!

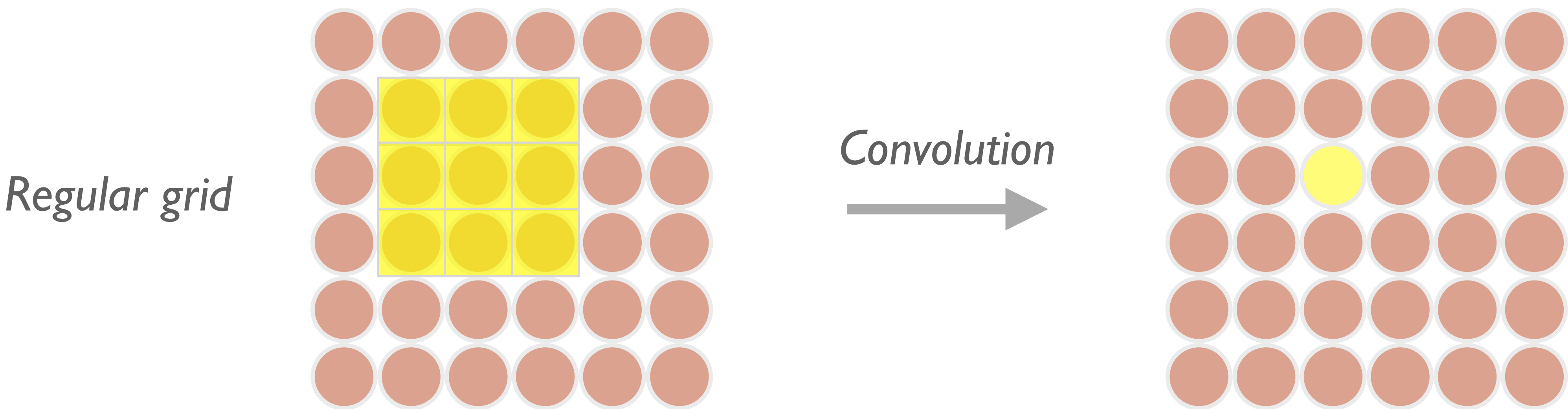
Why Particle Cloud?

- Image
  - can benefit directly from powerful and sophisticated algorithms (CNNs) from the computer vision community
  - images are uniform grids of pixels:
    - projecting particles into pixels leads to high sparsity and loss of granularity
    - also nontrivial to include features other than particle energy/momenta (e.g., track impact parameters)
- Sequence
  - preserves full granularity, no loss of information from pixelation
  - straightforward to include any features for each particle
  - all sequence-based algorithm (RNN/RecNN/1D CNN) assumes an explicit ordering
    - but jet constituents are intrinsically unordered
- Point cloud
  - shares all the benefits of sequence
  - and points are unordered

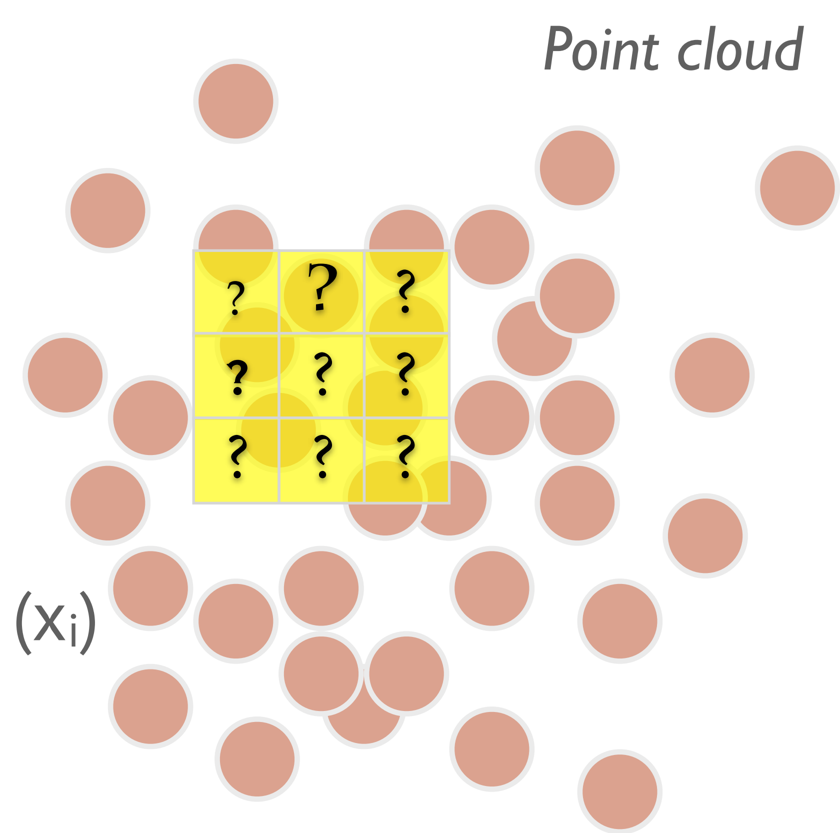
LEARNING FROM POINT CLOUDS

- Efficient representation already half of success
  - but only half…
  - the other half: a powerful network that fully exploits all the information in the representation
- Learning from point clouds
  - active research area in the ML community mainly due to the prosperity of autonomous driving technology
  - many custom algorithms proposed recently
- One of the key aspects is to respect/exploit the permutation invariance of the inputs
  - one approach: use a "global" symmetric function over inputs (e.g., Deep Sets)
    - adapted to particle clouds -> Energy Flow Network *see Patrick's talk!*
  - another approach: hierarchical learning from "local" to "global"
    - an example: convolution operation
      - key contributor to the overwhelming success of CNNs in image recognition
    - can we adapt convolution to work on point clouds?

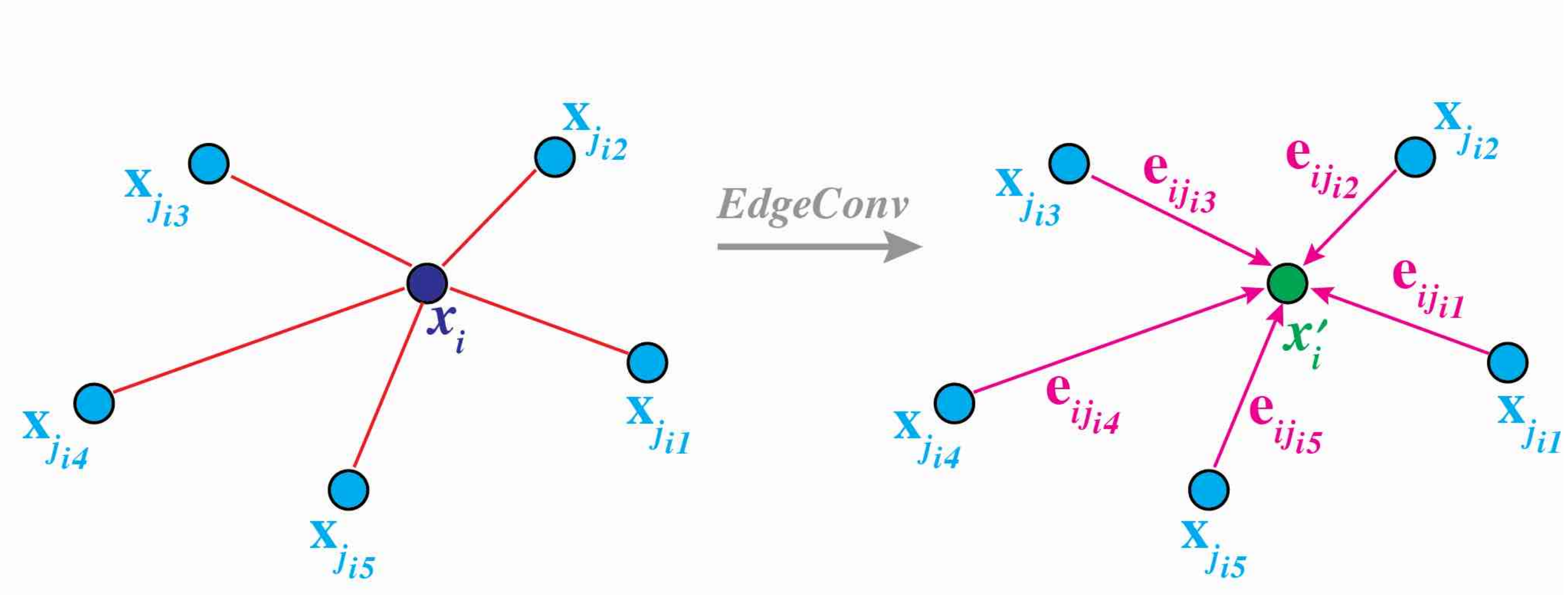
CONVOLUTION ON REGULAR GRIDS



- Conventional convolution only operates on regular grids and cannot be applied on point clouds
  - point clouds are irregular
    - how to define a "local" patch to convolve?
  - point clouds are unordered
    - conventional convolution operation ($\Sigma_i K_i x_i$) is not invariant under permutation of the points (x_i)



Convolution on Point Clouds



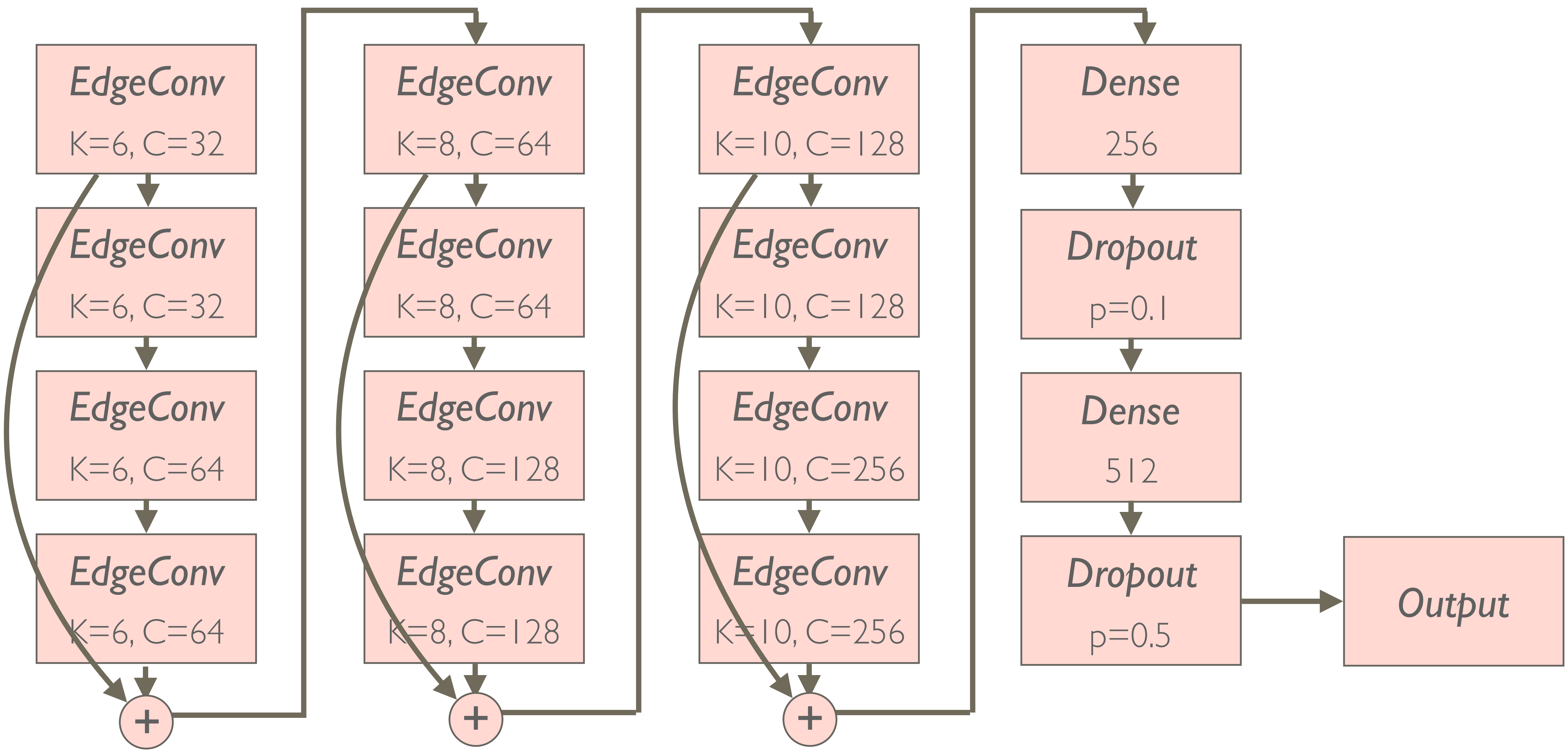
- Convolution on point clouds: *EdgeConv* [arXiv:1801.07829]
  - treating a point cloud as a graph:
    - each point is a vertex
  - for each point, a local patch is defined by getting the K-nearest neighbors to it
    - distance defined based on the point "coordinates"
  - designing a symmetric "convolution" function
    - define "edge feature" for each center-neighbor pair: $e_{ij} = h_\Theta(x_i, x_j)$
      - same h_Θ for all neighbor points, and all center points, for symmetry
    - aggregate the edge features in a symmetric way: $x_i' = \Sigma_j e_{ij}$

Dynamic Graph CNN

- *EdgeConv* shares many nice properties of regular CNNs
  - incorporates local neighborhood information (correlations)
  - can be stacked to perform a hierarchical learning from local to global features
- Dynamic Graph CNN (DGCNN)
  - when stacking the EdgeConv layers, it is possible to recompute the graph using nearest neighbors in the features space produces by each layer
    - i.e., the output features of each EdgeConv layer can be treated as a new "coordinate" for each point
    - point distances can be updated using these learned coordinates (in a latent space)
  - found to be beneficial in the ML paper
- Customization for particle clouds
  - the original DGCNN is actually not directly applicable to jets, as the particle inputs have not only "coordinates" (i.e., η, φ), but also additional features (p_T, charge, particle ID, etc.)
  - to apply DGCNN on particle clouds, some customizations are made to the first EdgeConv layer:
    - the nearest neighbor finding is purely based on the (η, φ) coordinates
    - then, the other features are added to the (η, φ) coordinates of each particle for producing the edge features

Network Architecture

- Implemented with mxnet



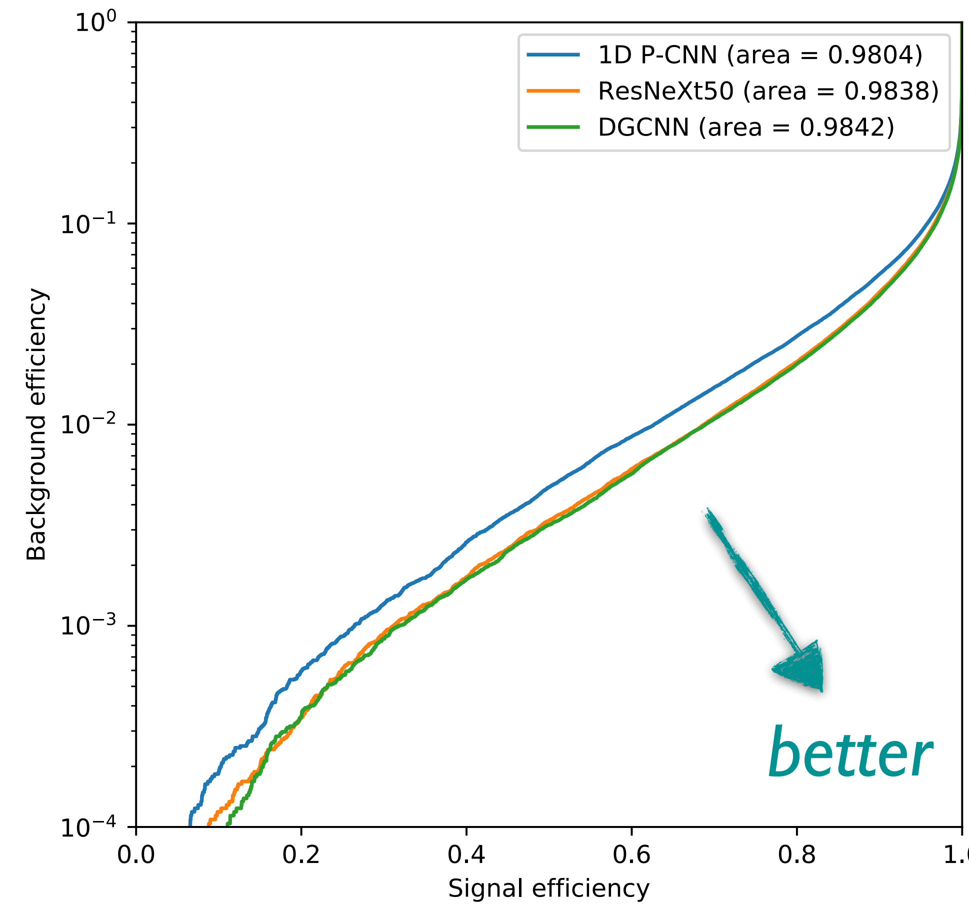
- 3 stages: k-nearest neighbors updated at the beginning of each stage
- batch normalization (BN) used after each EdgeConv operation
- residual connection (RC) [1512.03385, 1603.05027] added between EdgeConv layers
- BN and RC helped greatly for stabilizing the training and also improving the performance

Performance Comparison

- The performance of DGCNN is benchmarked on two jet tagging tasks using public datasets:
  - top tagging dataset: A. Butter, G. Kasieczka, T. Plehn and M. Russell [arXiv:1707.08966, link]
  - quark/gluon tagging dataset: P. T. Komiske, E. M. Metodiev and J. Thaler [arXiv:1810.05165, link]
  - very nice public datasets
    - ML-friendly format, convenient for developing/testing new algorithms
    - allow for consistent comparison between algorithms
    - the community needs more public datasets
      - especially ones closer to real experiments (pileup, tracking, detector resolution, etc.)
      - open data/simulation in ML-friendly format from CMS and ATLAS would be of great help!
- Results compared with a few alternative algorithms
  - 1D CNN over particle sequence (P-CNN)
    - CMS "DeepAK8" architecture [CMS-DP-2017-049]
    - but using only information available in the datasets (e.g., no tracking)
  - 2D CNN over 64x64 jet image
    - state-of-the-art model from image recognition: ResNeXt50 [arXiv:1611.05431]
    - #filters reduced by a factor or 4 to avoid overfitting and also to speed up training
  - Particle Flow Network (PFN) [arXiv:1810.05165]

PERFORMANCE: TOP TAGGING

- Top tagging:
  - only particle 4-momentum is available
  - train/val/test 1.2M/400k/400k
  - results of more algorithms available at link
- We managed to push the boundary a bit further
  - >20% lower background at signal efficiency of 30%
  - *Is the gain real? Or is it just learning more details of the parton shower model?*
  - but personally I would not really consider the jet tagging problem as "solved"
    - especially facing realistic experimental challenges like pileup, detector effects, and additional information (e.g., tracking, timing, etc.)

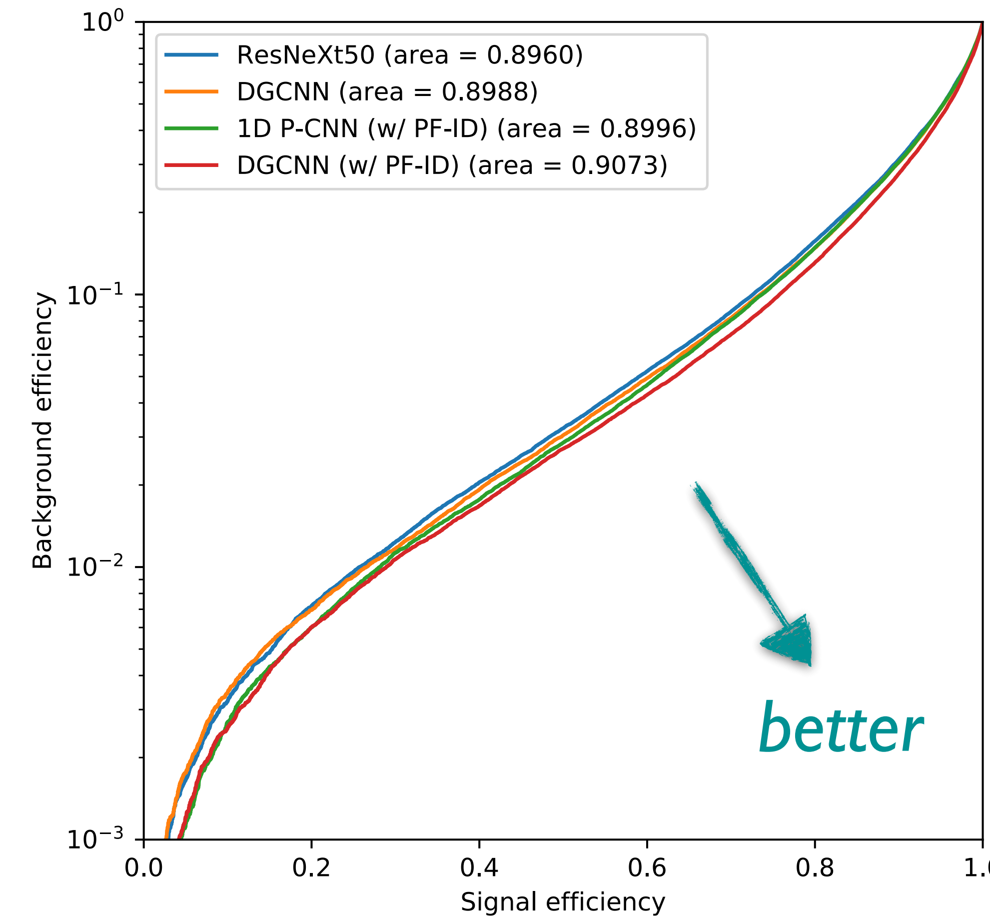


Performance on top-tagging dataset

Algorithm	Accuracy	ROC AUC	$1/\varepsilon_{bkg}$ @ ε_{sig}=30%
1D P-CNN	0.930	0.9804	780
2D CNN [ResNeXt50]	0.936	0.9838	1086
DGCNN	**0.937**	**0.9842**	**1160**
PFN-r.r. [arXiv:1810.05165]	*0.932*	*0.9819 ± 0.0001*	*888±17*

Performance: Quark/Gluon Tagging

- Quark/gluon tagging
  - dataset from P. T. Komiske, E. M. Metodiev and J. Thaler [arXiv:1810.05165, link]
  - train/val/test 1.6M/200k/200k
  - two versions with different information:
    - momentum-only
    - momentum + realistic particle ID (e/μ/γ/charged hadron/neutral hadron) + charge

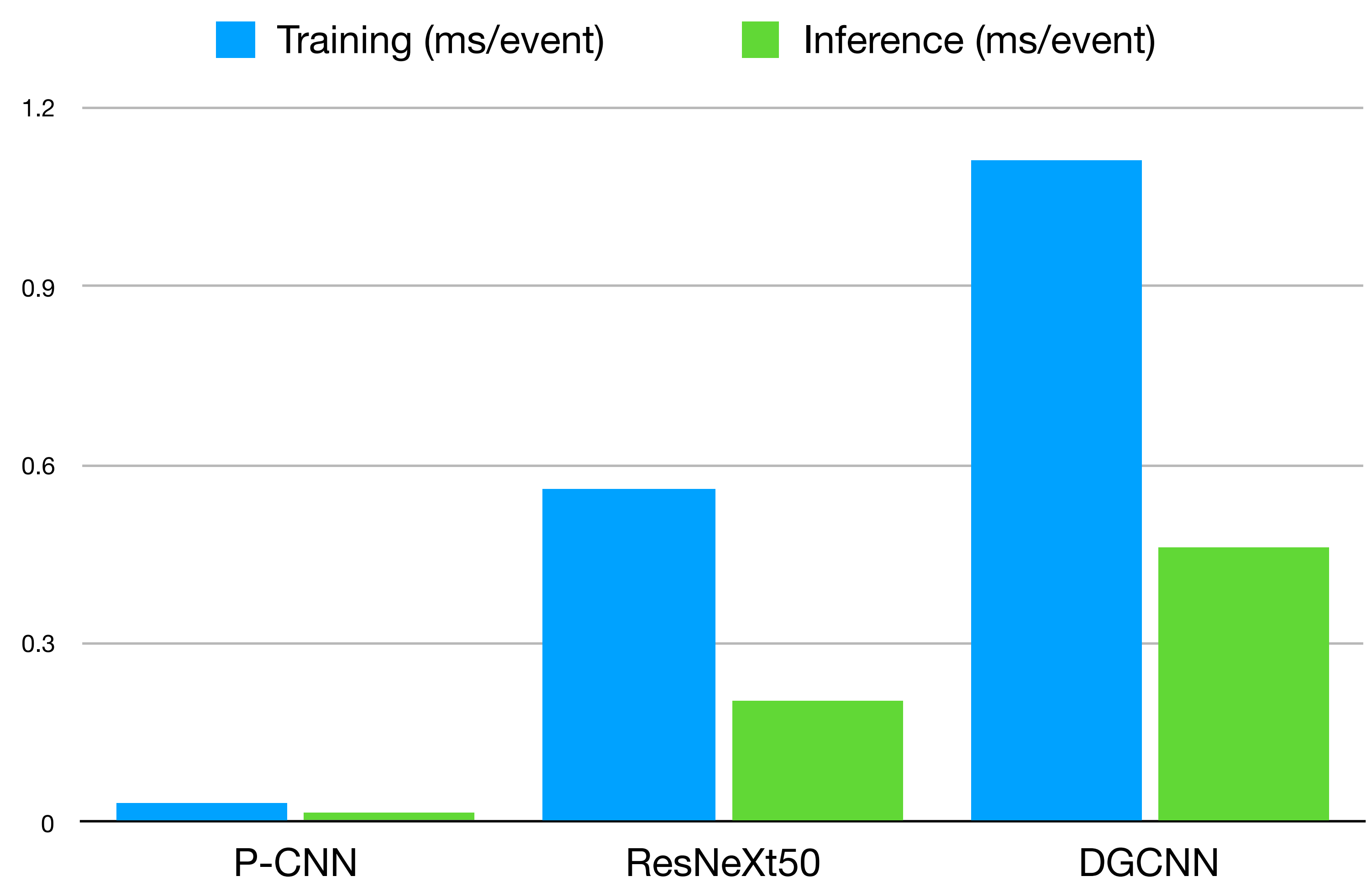


Performance on quark/gluon-tagging dataset

	Algorithm	Accuracy	ROC AUC	$1/\varepsilon_g$ @ ε_q=50%
Momentum-only	*PFN [arXiv:1810.05165]*	-	*0.8911 ± 0.0008*	*30.8 ± 0.4*
	2D CNN [ResNeXt50]	0.821	0.8960	30.9
	DGCNN	0.826	0.8988	32.8
Momentum + realistic particle ID	*PFN-Ex [arXiv:1810.05165]*	-	*0.9005 ± 0.0003*	*34.7 ± 0.4*
	1D P-CNN	0.826	0.8996	34.9
	DGCNN	**0.835**	**0.9073**	**36.8**

Training and Inference Speed

- Another important factor of a ML algorithm is the training and inference speed
  - benchmarked on a GTX 1080Ti
  - current ML package not well optimized for graph network
    - future implementation/hardware(e.g., FPGA) may greatly improve the speed

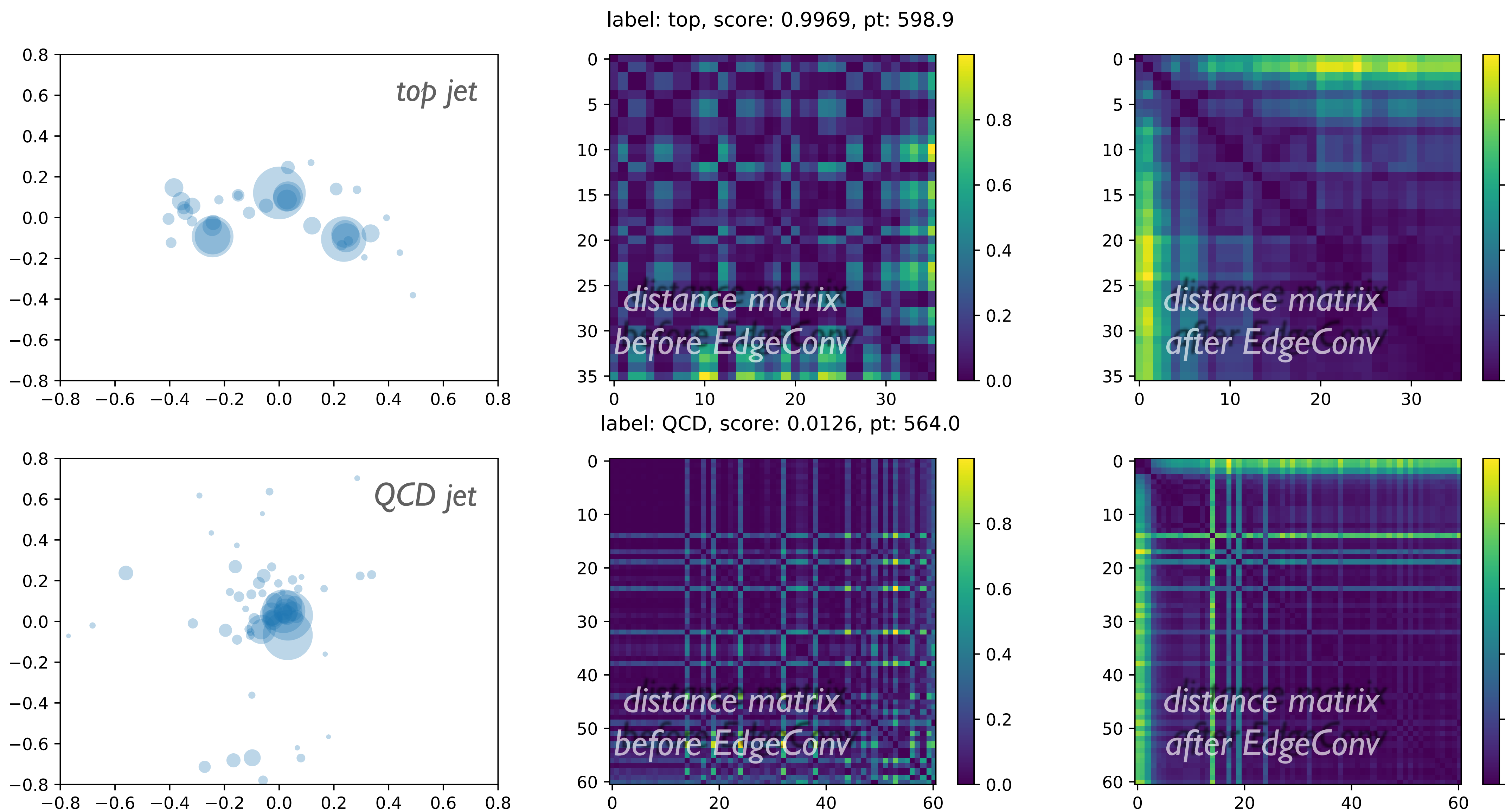


Outlook

- We present another point cloud inspired approach for jet tagging
  - a jet can be viewed as a cloud of unordered particles
  - dynamic graph CNN, based on permutation-invariant EdgeConv operation, is applied on particle clouds for jet tagging
  - better performance for jet tagging compared to existing approaches based on jet images and sequences
- Point cloud: a natural and flexible representation for jets as well as collision events, with a rich connection yet to be fully explored
  - point cloud classification (labelling the whole cloud) <-> jet tagging
  - point cloud segmentation (labelling each point)
    - pileup identification?
    - particle tracking?
  - so far we are importing expertise from ML community
    - can we propose better ML algorithms based on physics?
    - can we export better ideas to the ML community?

Outlook (cont.)

- Point cloud: a natural and flexible representation for jets as well as collision events, with a rich connection yet to be fully explored
  - can we peek in the ML model to see what/how it learned?
  - can we learn jet physics from ML?



Backups

TOP TAGGING COMPARISON

Approach	AUC	Acc.	1/eB (@ eS=0.3)	Contact	Comments
LoLa	0.979	0.928		GK / SImon Leiss	Preliminary number, based on LoLa
LBN	0.981	0.931	863	Marcel Rieger	Preliminary number
CNN	0.981	0.93	780	David Shih	Model from *Pulling Out All the Tops with Computer Vision and Deep Learning (1803.00107)*
P-CNN (1D CNN)	0.980	0.930	782	Huilin Qu, Loukas Gouskos	Preliminary, use kinematic info only (https://indico.physics.lbl.gov/indico/event/546/contributions/1270/)
6-body N-subjettiness (+mass and pT) NN	0.979	0.922	856	Karl Nordstrom	Based on 1807.04769 (*Reports of My Demise Are Greatly Exaggerated: N-subjettiness Taggers Take On Jet Images*)
8-body N-subjettiness (+mass and pT) NN	0.980	0.928	795	Karl Nordstrom	Based on 1807.04769 (*Reports of My Demise Are Greatly Exaggerated: N-subjettiness Taggers Take On Jet Images*)
Linear EFPs	0.980	0.932	380	Patrick Komiske, Eric Metodiev	d<= 7, chi <= 3 EFPs with FLD. Based on 1712.07124: *Energy Flow Polynomials: A complete linear basis for jet substructure.*
Particle Flow Network (PFN)	0.982	0.932	888	Patrick Komiske, Eric Metodiev	Median over ten trainings. Based on Table 5 in 1810.05165: *Energy Flow Networks: Deep Sets for Particle Jets.*
Energy Flow Network (EFN)	0.979	0.927	619	Patrick Komiske, Eric Metodiev	Median over ten trainings. Based on Table 5 in 1810.05165: *Energy Flow Networks: Deep Sets for Particle Jets.*