ParticleNet: Jet Tagging via Particle Clouds

Huilin Qu

UC Santa Barbara

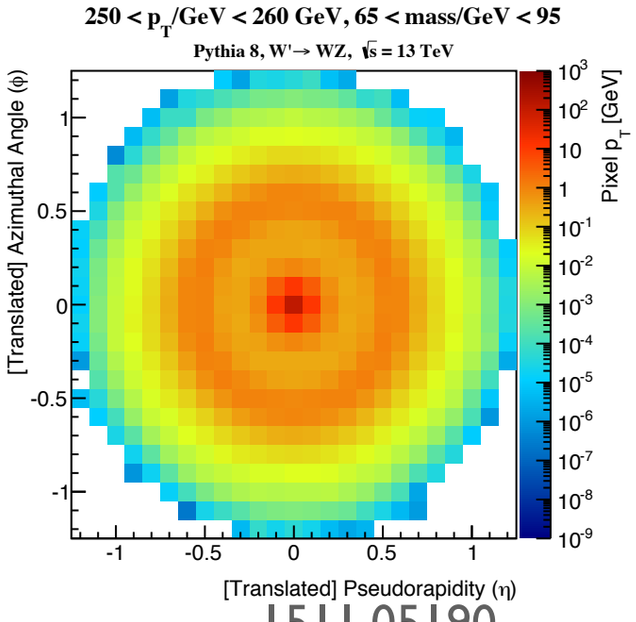
Based on work with Loukas Gouskos
arXiv:1902.08570

3rd IML Machine Learning Workshop
April 17, 2019

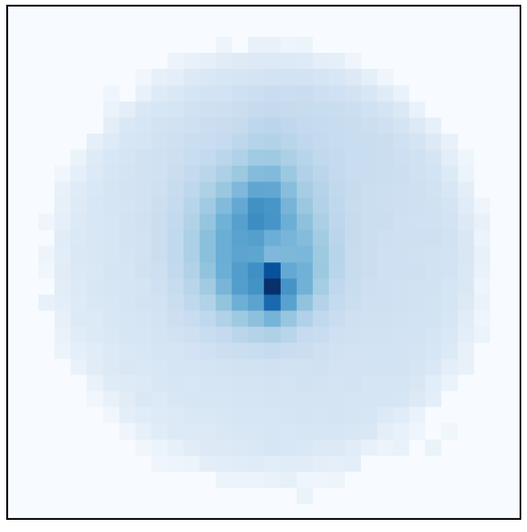
Introduction

- Jet is one of the most ubiquitous but also fascinating objects at the LHC
- Jet tagging: key to many new physics searches and SM measurements
  - heavy flavour tagging (bottom/charm)
  - heavy resonance tagging (top/W/Z/Higgs)
  - quark/gluon discrimination
  - …
- One of the most active areas for ML
  - lots of *deep* approaches have been proposed in the past few years
    - trending: low-level inputs + deep learning
- Question: *How to best represent a jet for ML?*

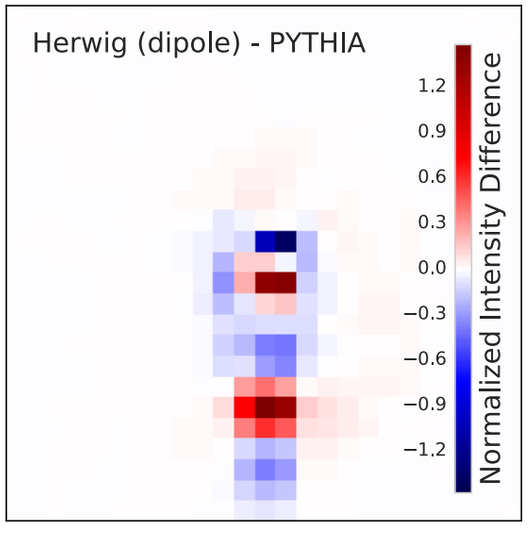
As Images…



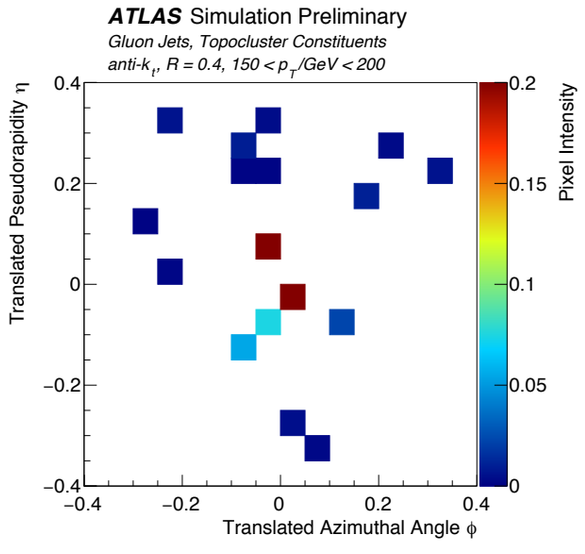
1511.05190



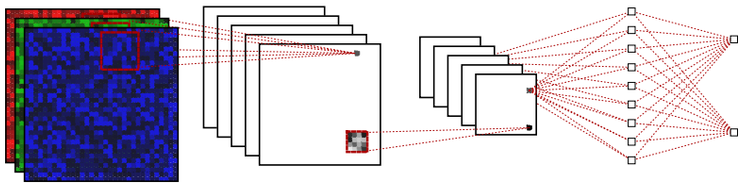
1603.09349



1609.00607



ATL-PHYS-PUB-2017-017



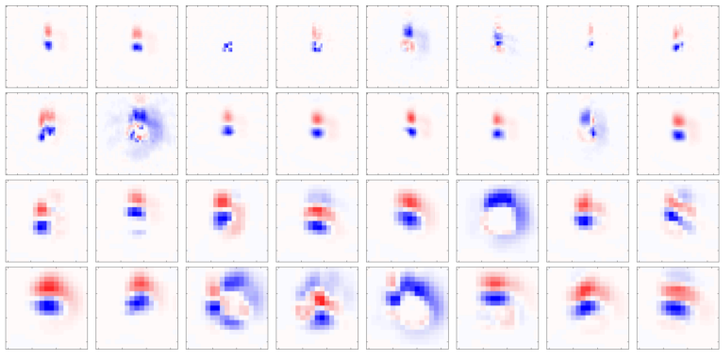
red = transverse momenta of charged particles

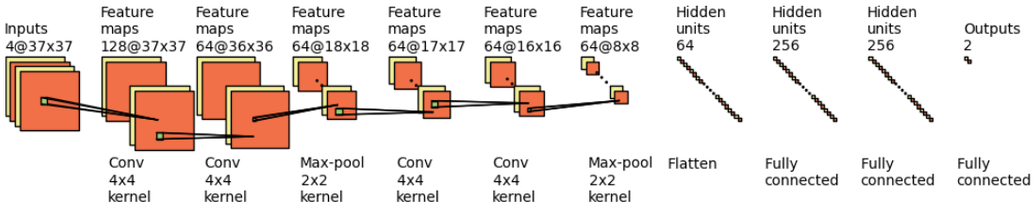
green = the transverse momenta of neutral particles

blue = charged particle multiplicity

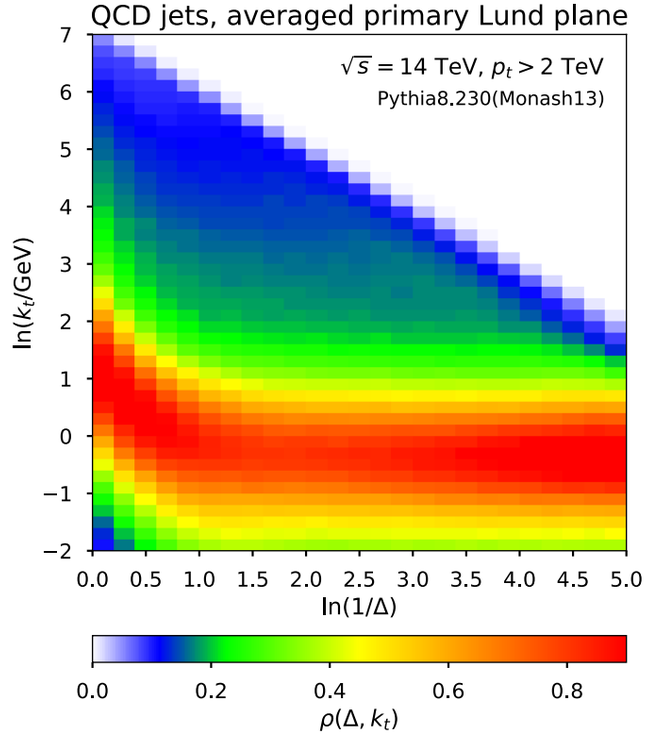
1612.01551

1701.08784





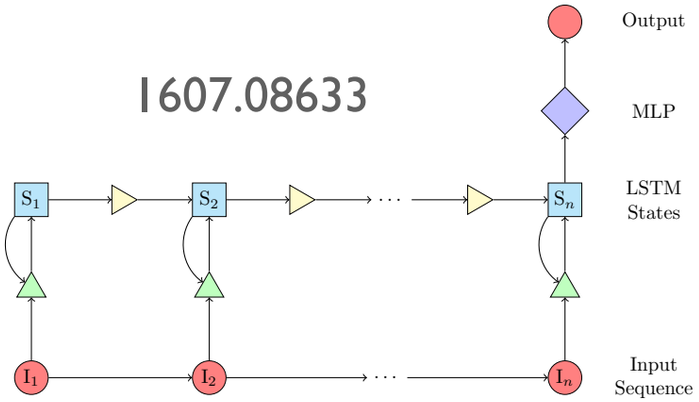
1803.00107

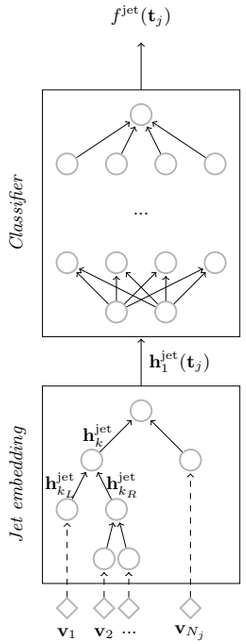


1807.04758

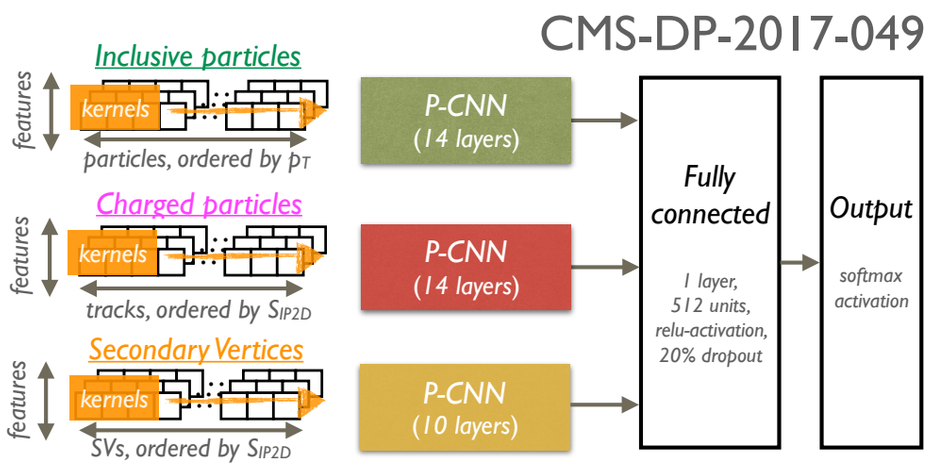
And more…

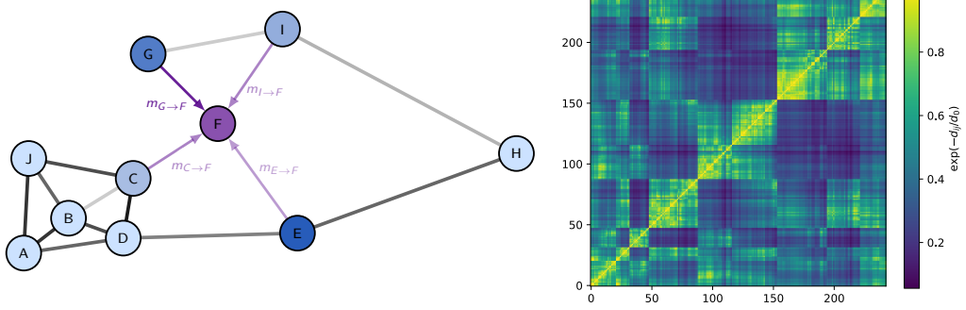
As Collections of Particles…



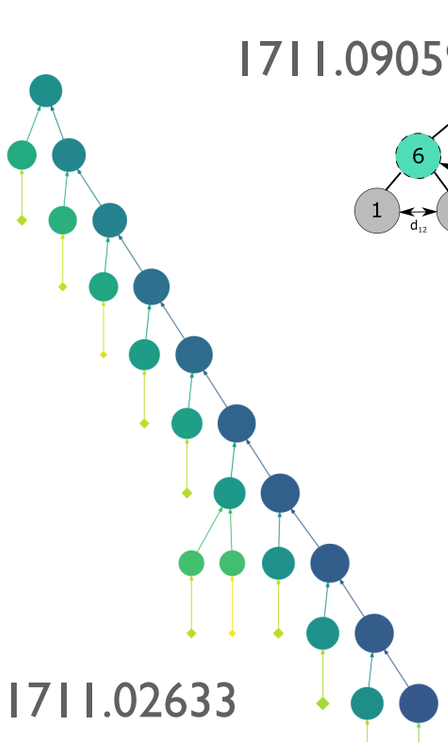


1702.00748

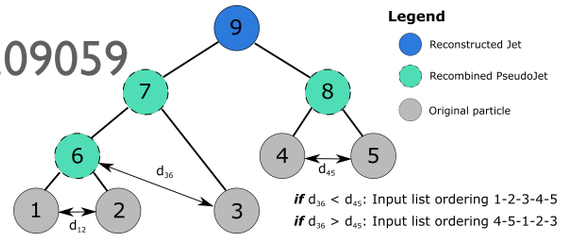


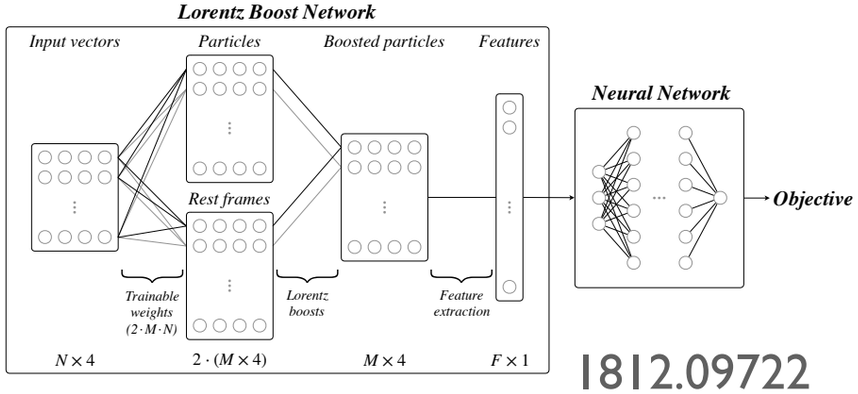


NIPS2017 workshop [http://bit.ly/2AkwYRG]

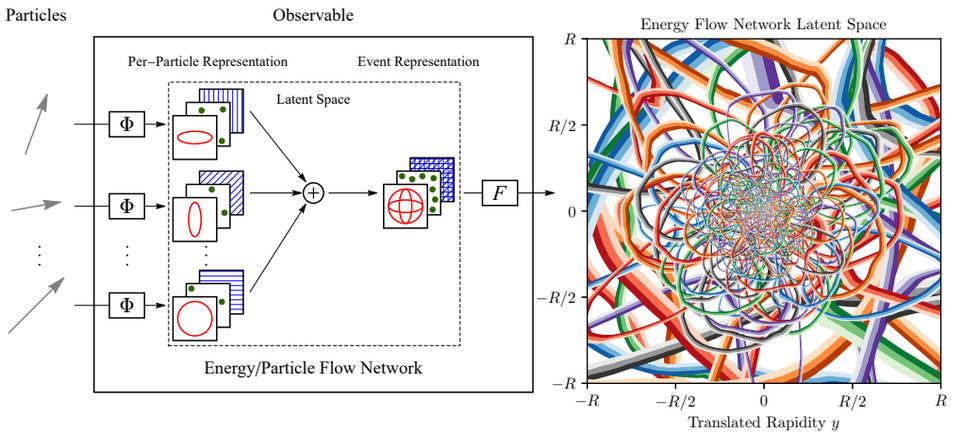


1711.02633





1812.09722



1810.05165

$$k_{\mu,i} \xrightarrow{\text{CoLa}} \tilde{k}_{\mu,j} = k_{\mu,i}\, C_{ij}$$

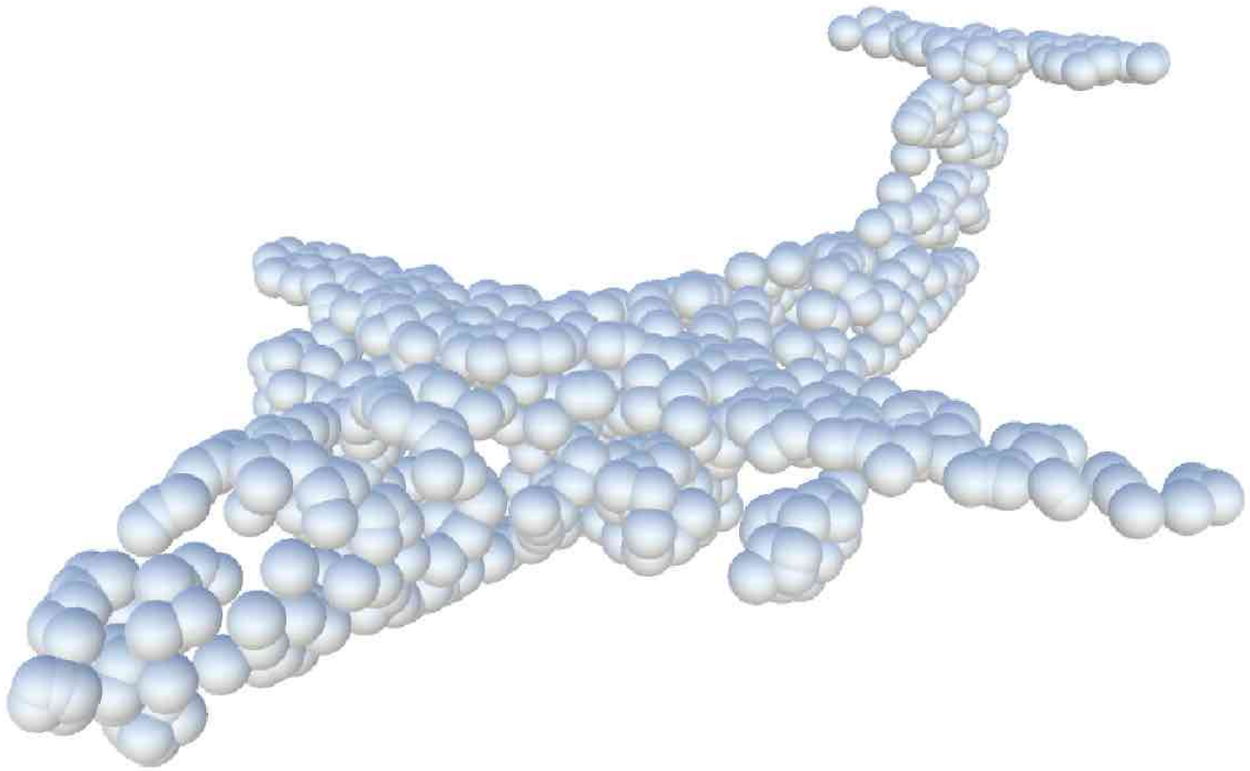
$$\text{with} \quad C = \begin{pmatrix} 1 & 1 & \cdots & 0 & \chi_1 & \cdots & 0 & C_{1,N+2} & \cdots & C_{1,M} \\ \vdots & & \ddots & & & \ddots & & \vdots & \ddots & \vdots \\ 1 & 0 & \cdots & 1 & 0 & \cdots & \chi_N & C_{N,N+2} & \cdots & C_{N,M} \end{pmatrix}$$

$$\tilde{k}_j \xrightarrow{\text{LoLa}} \hat{k}_j = \begin{pmatrix} m^2(\tilde{k}_j) \\ p_T(\tilde{k}_j) \\ p_T(\tilde{k}_j)\Delta R_{j,\text{jet}} \\ w_{jm}^{(E)}\, E(\tilde{k}_m) \\ w_{jm}^{(d)}\, d_{jm}^2 \\ E_T(\tilde{k}_j) E_T(\tilde{k}_m) (\Delta R_{jm})^{0.2} \end{pmatrix},$$

1707.08966, 1812.09223

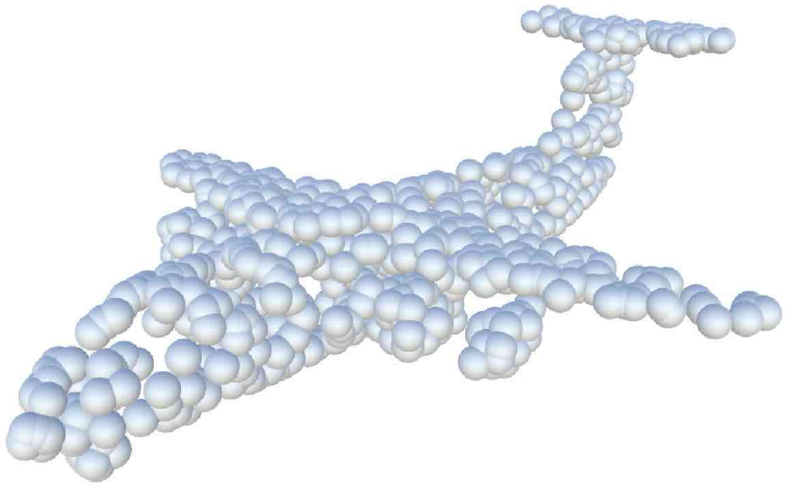
And more…

As… Point Clouds?



arXiv:1801.07829

As… Point Clouds?

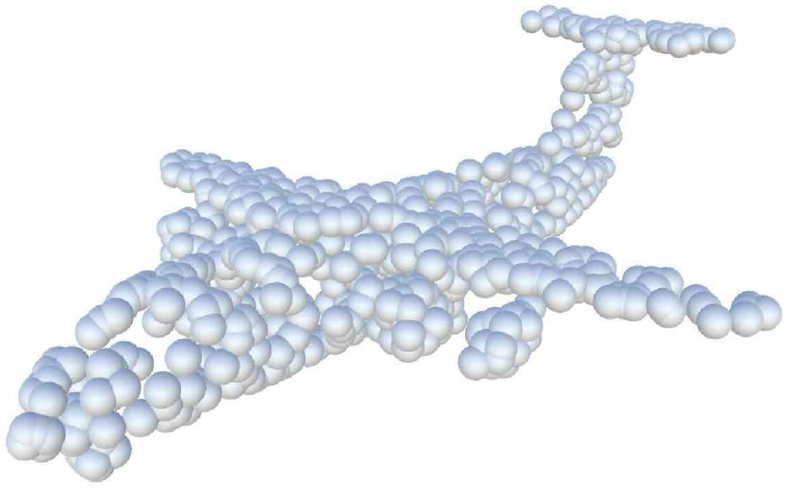


Point cloud

From Wikipedia, the free encyclopedia

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

As… Point Clouds?



Point cloud

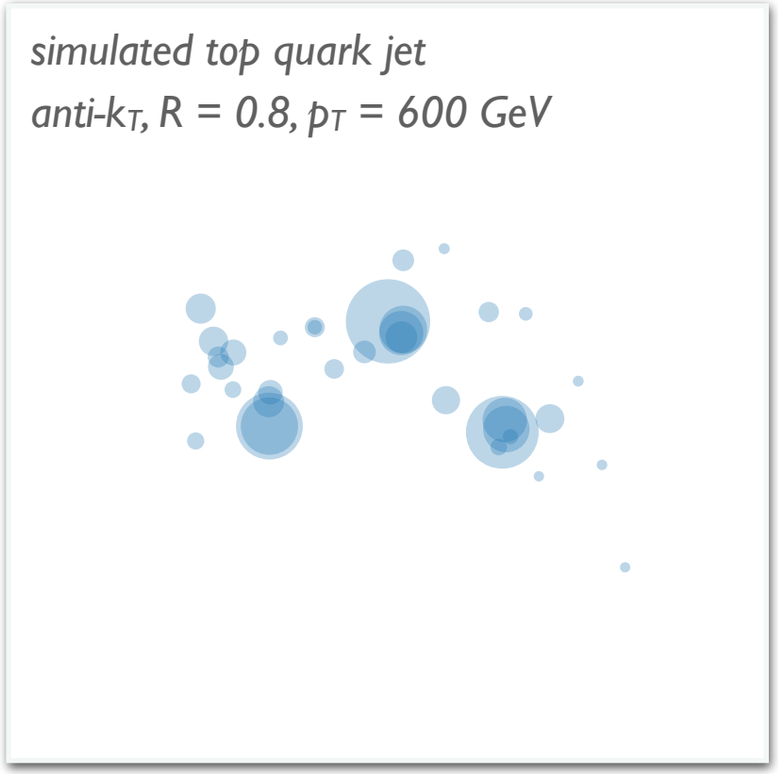
From Wikipedia, the free encyclopedia

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.

Point Cloud

Image from https://news.voyage.auto/an-introduction-to-lidar-the-key-self-driving-car-sensor-a7e405590cff

JET AS A ~~POINT~~ CLOUD
PARTICLE

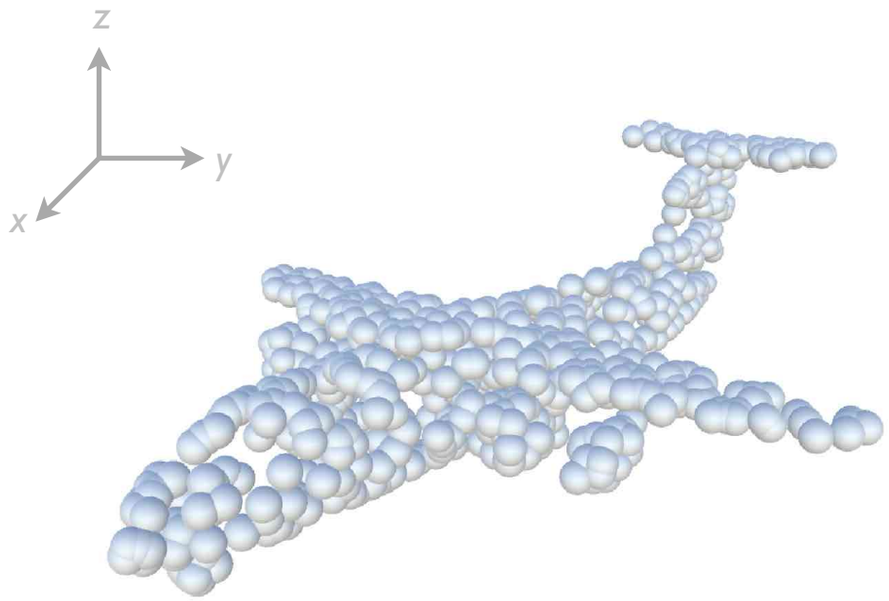


Jet (Particle cloud)

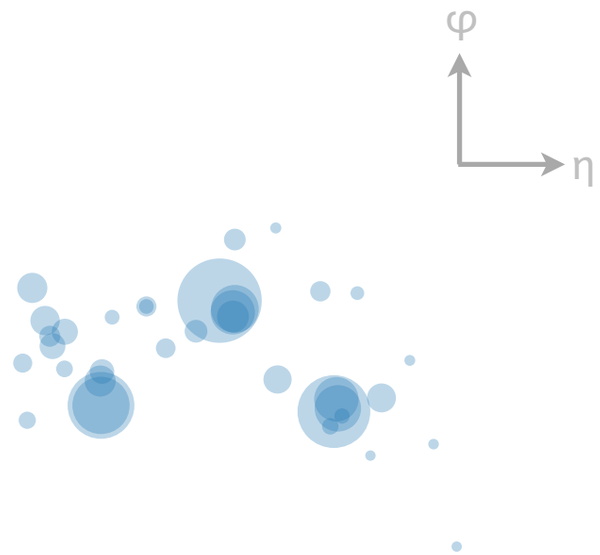
From Wikipedia, the free encyclopedia

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 many additional features!*
    - energy/momenta
    - charge/particle ID
    - track quality/impact parameters/etc.

Why Particle Cloud?

- Image-based approaches
  - natural idea for calorimeters
  - powerful performance using convolutional neural networks (CNNs)
  - projecting particles into pixels leads to high sparsity and loss of information
  - difficult to include non-additive features (e.g., particle ID, track impact parameters)
- Particle-based approaches
  - preserves full granularity - no loss of information from pixelation
  - straightforward to include any kind of features for each particle
  - constituent particles of a jet are intrinsically unordered — permutation symmetry!
  - however, most particle-based approaches (RNN/TreeNN/1D CNN) failed to exploit this symmetry
- The particle cloud approach
  - also particle-based
  - builds on permutation symmetry from the beginning

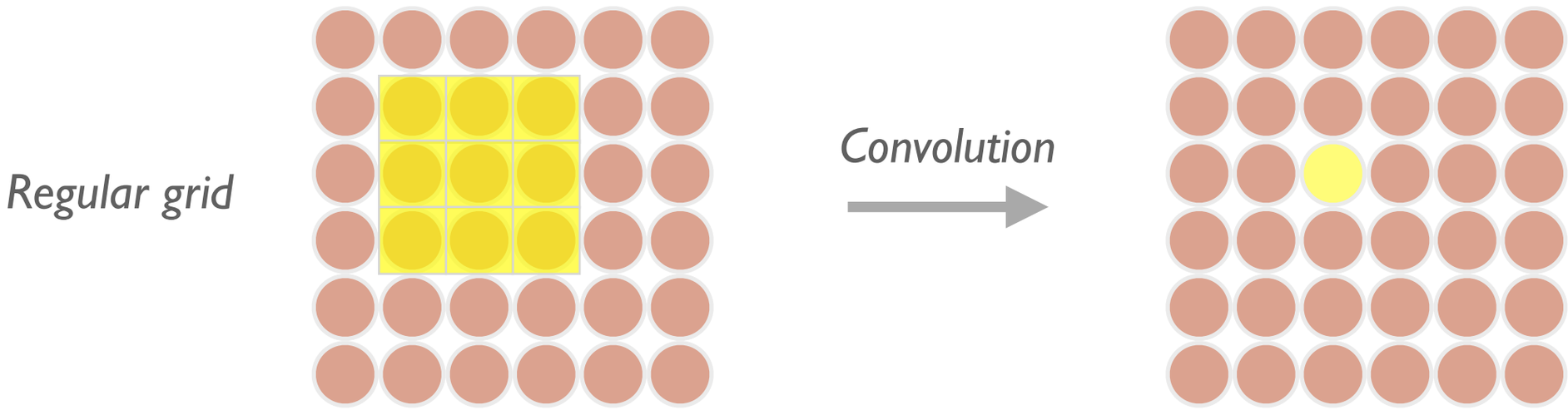
LEARNING FROM PARTICLE CLOUDS

- Learning from point clouds
  - active research area in the ML community, mainly due to the rise in autonomous driving researches
  - methods for point clouds can be imported and improved for particle clouds
- The key is to exploit the permutation symmetry
  - one approach: Energy Flow Network [arXiv:1810.05165]
    - per-particle transformations + symmetric global function

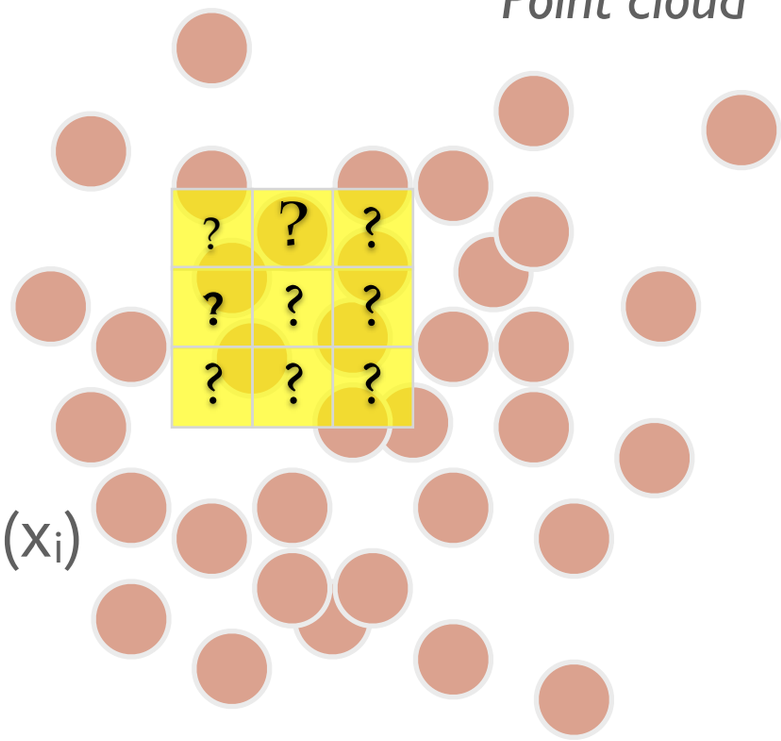
$$\mathcal{O}(\{p_1, \ldots, p_M\}) = F\left(\sum_{i=1}^{M} \Phi(p_i)\right),$$

    - generally, any permutation-invariant functions can be approximated with this form [Deep Sets, arXiv:1703.06114]
- our approach:
  - use permutation-invariant convolution designed for point clouds
  - exploit the spatial structure of particles explicitly

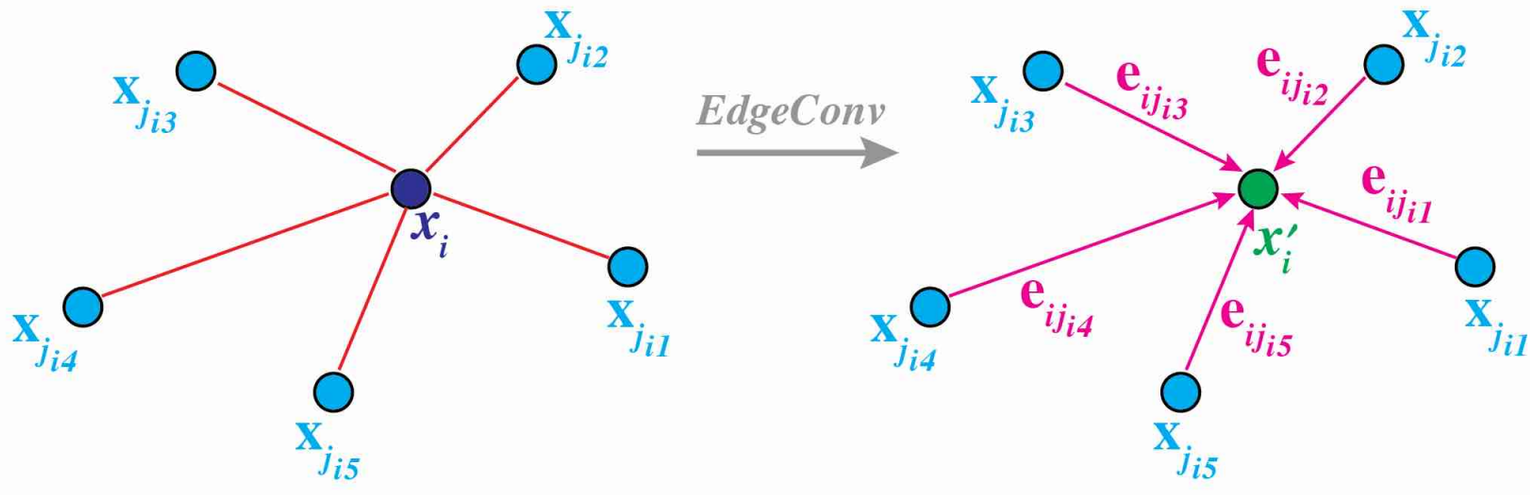
Convolution on Regular Grids



- 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



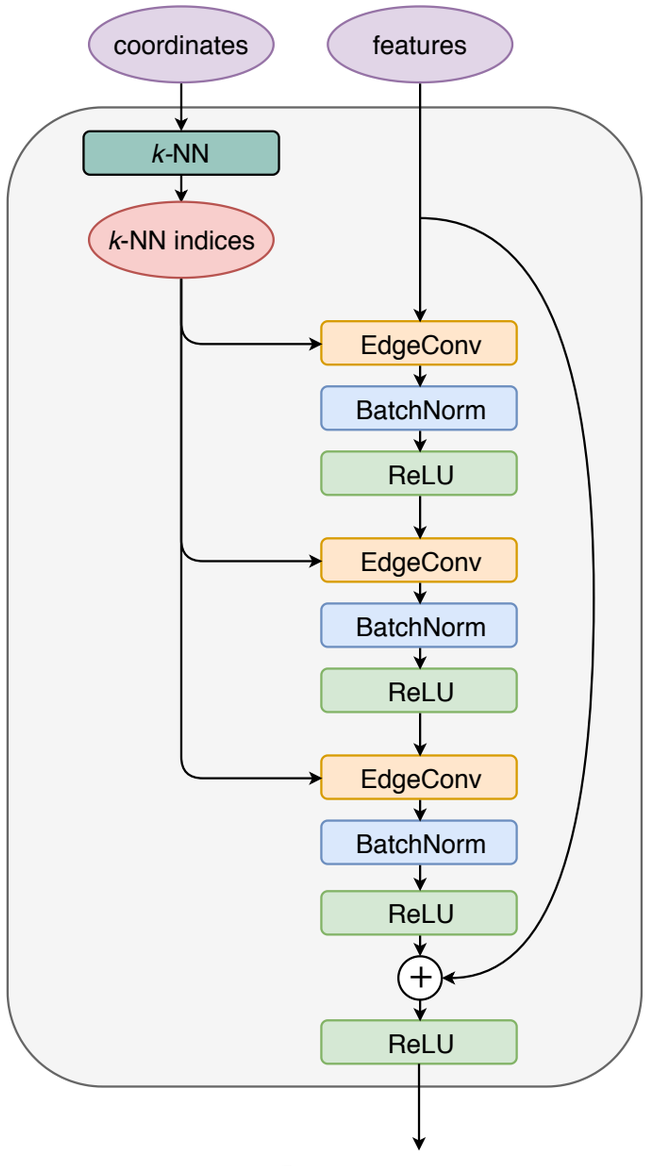
- Convolution on point clouds: *EdgeConv* [arXiv:1801.07829]
  - defining the local patch
    - for each point, a local patch is defined by finding its k-nearest neighbors
    - since points have coordinates, distance can be naturally defined
  - designing a symmetric "convolution" function
    - define "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: $x_i' = \square_j\ e_{ij}$
      - $\square$ is a permutation-symmetric function, e.g., sum, mean, max

Dynamic Graph CNN

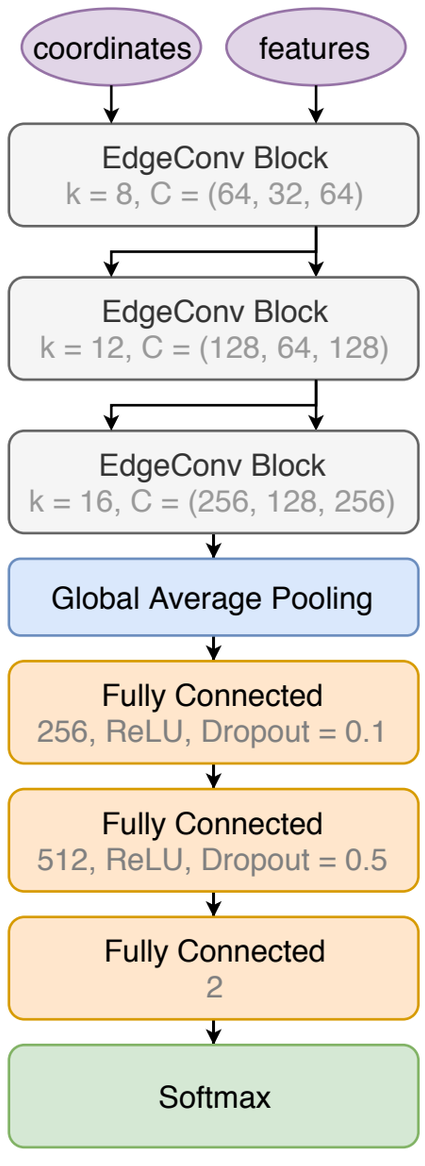
- *EdgeConv* shares many nice properties of regular CNNs
  - incorporates local neighborhood information
  - EdgeConv layers can be stacked to allow the network to learn in a hierarchical manner
- Dynamic Graph CNN (DGCNN) [arXiv:1801.07829]
  - an EdgeConv operation outputs a set of learned features for each point
  - these features can also be interpreted as learned "coordinates" for each point (in a high-dim latent space)
  - such an interpretation enables us to recompute the "distances" between points and therefore dynamically update the k-nearest neighbors as going from one EdgeConv layer to another
  - found to be beneficial in the original paper

The ParticleNet Architecture

- Based on EdgeConv and DGCNN, we developed ParticleNet, a customized architecture for jet tagging on particle clouds



EdgeConv block (inspired by ResNet)



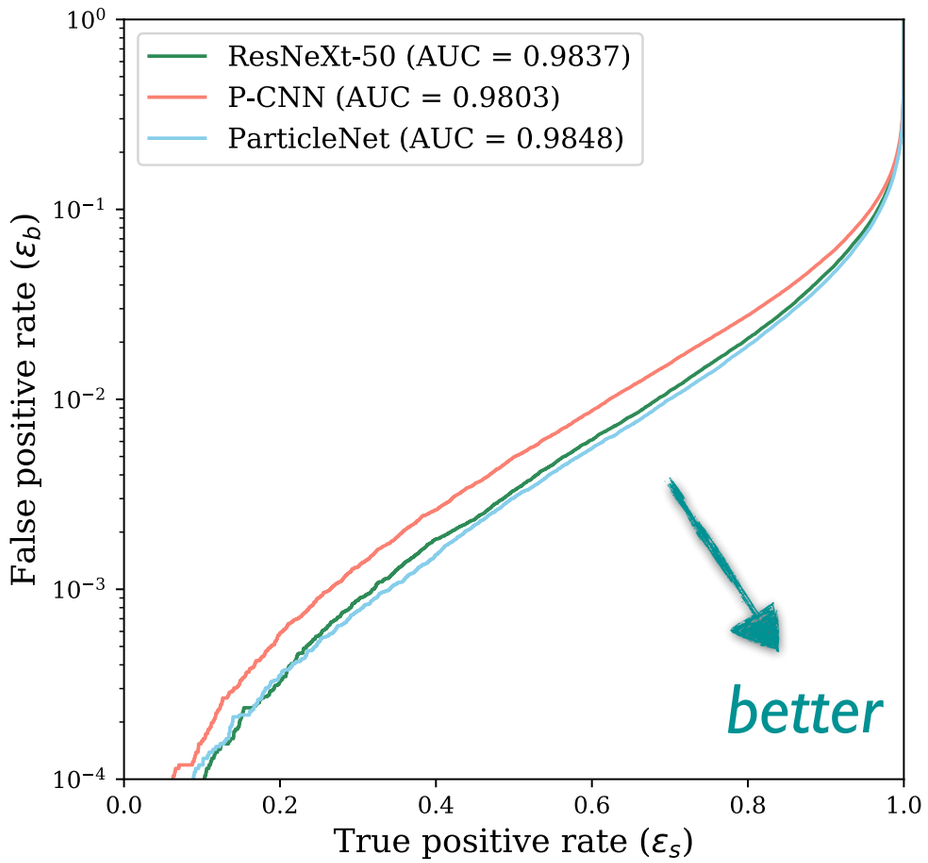
ParticleNet architecture

Performance Comparison

- The performance of ParticleNet 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]
- Results compared with a few alternative algorithms
  - 1D CNN over particle sequence (P-CNN)
    - i.e., the 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 for image recognition: ResNeXt-50 [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]
    - not retrained — results are the ones reported in the original paper

Performance: Top Tagging

- Top tagging:
  - only four-momenta of the particles available in this dataset
  - train/val/test 1.2M/400k/400k
- Metrics:
  - accuracy
  - area under the ROC curve (AUC)
  - background rejection ($1/\varepsilon_b$) at signal efficiency of 30% and 50%
- Results:
  - substantial improvement over P-CNN/PFN
  - >10% better in background rejection compared to ResNeXt-50



	Accuracy	AUC	$1/\varepsilon_b$ at $\varepsilon_s = 50\%$	$1/\varepsilon_b$ at $\varepsilon_s = 30\%$
ResNeXt-50	0.936	0.9837	302	1147
P-CNN	0.930	0.9803	201	759
PFN	-	0.9819 ± 0.0001	247 ± 3	888 ± 17
ParticleNet	**0.938**	**0.9848**	**329**	**1294**

Top Tagging Comparison

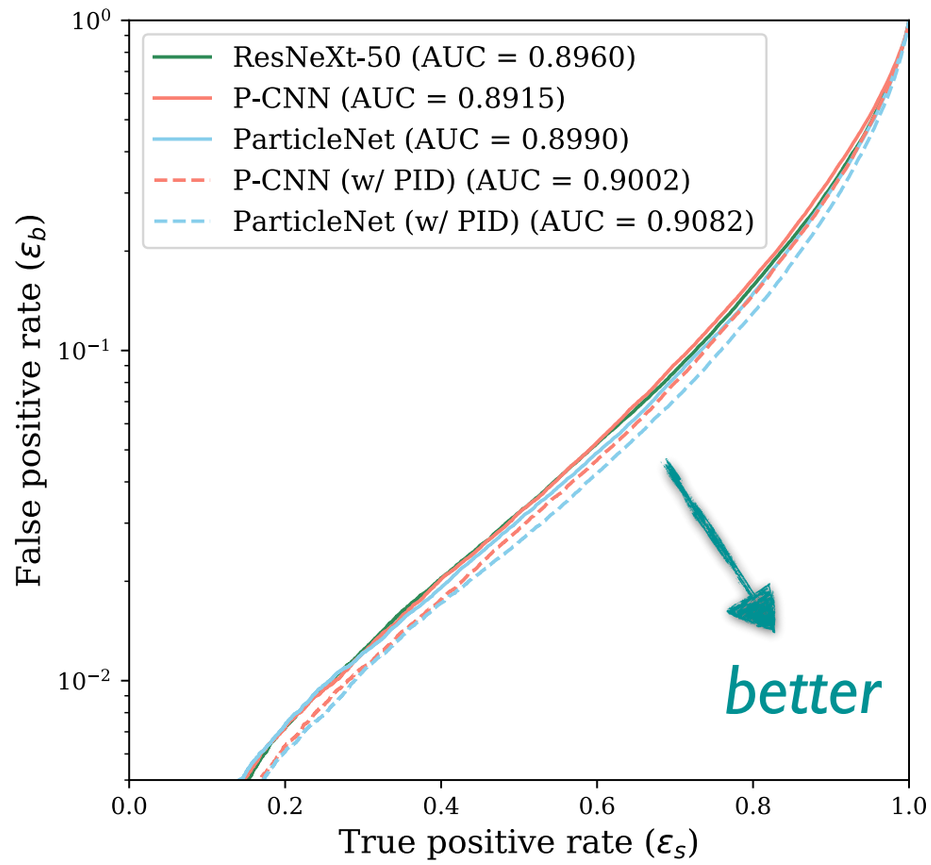
- First direct comparison of a large number of ML algorithms on jet tagging, thanks to the public dataset prepared by G. Kasieczka et al.
  - ParticleNet leads the performance up to now
  - an ensemble of all taggers (GoaT) can push the performance even further
    - complementarity between different algorithms

	AUC	Acc	$1/\epsilon_B$ ($\epsilon_S = 0.3$) single	mean	median	#Param
CNN [16]	0.981	0.930	914±14	995±15	975±18	610k
ResNeXt [30]	0.984	0.936	1122±47	1270±28	1286±31	1.46M
TopoDNN [18]	0.972	0.916	295±5	382± 5	378 ± 8	59k
Multi-body N-subjettiness 6 [24]	0.979	0.922	792±18	798±12	808±13	57k
Multi-body N-subjettiness 8 [24]	0.981	0.929	867±15	918±20	926±18	58k
TreeNiN [43]	0.982	0.933	1025±11	1202±23	1188±24	34k
P-CNN	0.980	0.930	732±24	845±13	834±14	348k
ParticleNet [47]	0.985	0.938	1298±46	1412±45	1393±41	498k
LBN [19]	0.981	0.931	836±17	859±67	966±20	705k
LoLa [22]	0.980	0.929	722±17	768±11	765±11	127k
Energy Flow Polynomials [21]	0.980	0.932	384			1k
Energy Flow Network [23]	0.979	0.927	633±31	729±13	726±11	82k
Particle Flow Network [23]	0.982	0.932	891±18	1063±21	1052±29	82k
GoaT	0.985	0.939	1368±140		1549±208	35k

arXiv:1902.09914

PERFORMANCE: QUARK/GLUON TAGGING

- Quark/gluon tagging
  - two versions with different information:
    - 4-momentum only
    - 4-momentum + particle ID
  - train/val/test 1.6M/200k/200k
- Results
  - best performance obtained with ParticleNet in both cases (w/ and w/o PID)
  - addition of PID inputs has a large impact on the performance

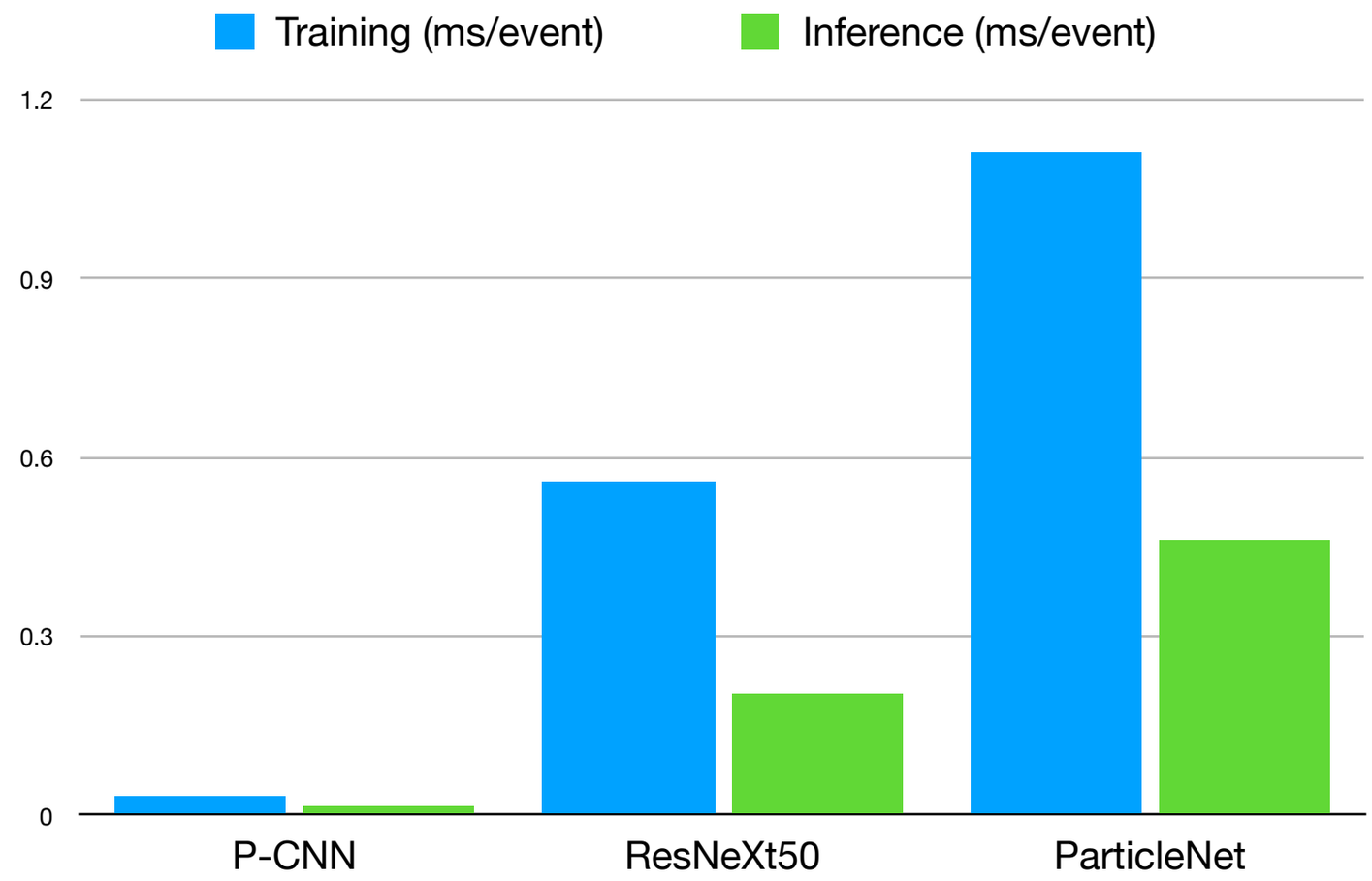


	Accuracy	AUC	$1/\varepsilon_b$ at $\varepsilon_s = 50\%$	$1/\varepsilon_b$ at $\varepsilon_s = 30\%$
ResNeXt-50	0.821	0.8960	30.9	80.8
P-CNN	0.818	0.8915	31.0	82.3
PFN	-	0.8911 ± 0.0008	30.8 ± 0.4	-
ParticleNet	0.826	0.8990	32.6	82.4
P-CNN (w/ PID)	0.827	0.9002	34.7	91.0
PFN-Ex (w/ PID)	-	0.9005 ± 0.0003	34.7 ± 0.4	-
ParticleNet (w/ PID)	**0.836**	**0.9082**	**38.0**	**93.7**

SUMMARY

- Jet as a particle cloud
  - inspired by point clouds, a jet can be viewed as a cloud of particles
  - new approach of representing jets for ML, with manifest permutation symmetry
- ParticleNet
  - a custom neural network architecture for jet tagging using particle clouds as inputs
  - significantly improved performance compared to existing ML approaches
- Particle cloud: a natural and flexible representation for jets as well as collision events, with a lot of potential yet to be fully explored
  - particle cloud classification (labelling the whole cloud) -> jet tagging
  - particle cloud segmentation (labelling each point) -> jet grooming / pileup identification?
  - particle cloud regression -> jet energy / mass regression?
  - more… ?

Backups

Training and Inference Speed

- Training and inference speed
  - benchmarked on a GTX 1080Ti
  - networks implemented with MXNet
  - caveats
    - implementation of EdgeConv/Graph CNN far less optimized compared to e.g., Conv2D