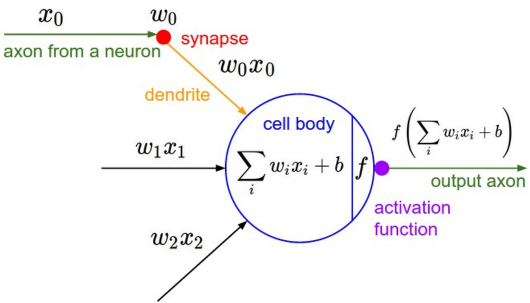
Primer on ML in HEP [3]

Overview

- Jets
  - b-tagging
  - t-tagging
  - grooming&PU
- Going adversarial
  - mass sculpting
  - DeepSF
  - domain adaptation
- Simulating with GAN&VAE
- Trigger
- Tracks

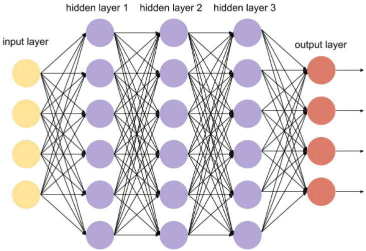
∗ Bonus
  - Kaggle
  - INFERNO

NN [recap]



x_0
w_0
synapse
axon from a neuron
$w_0 x_0$
dendrite
cell body
$f\left(\sum_i w_i x_i + b\right)$
$w_1 x_1$
$\sum_i w_i x_i + b$
f
output axon
activation
function
$w_2 x_2$

NN [recap]

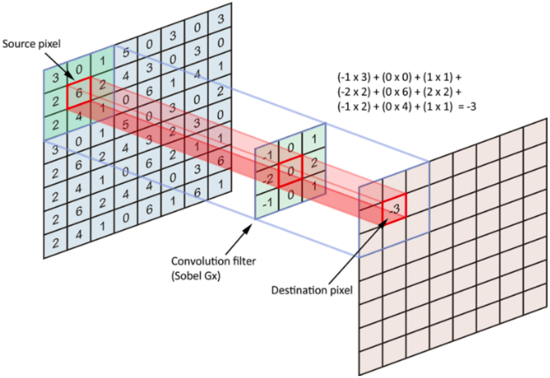


$$h_t = f(Wh_{t-1} + b)$$

$$a(x) = f_3(W_3 f_2(W_2 f_1(W_1 f_0(W_0 x + b_0) + b_1) + b_2) + b_3)$$

CNN [recap]

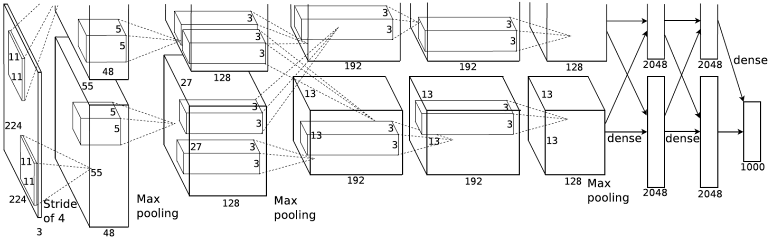
conv. layer



from [paper]

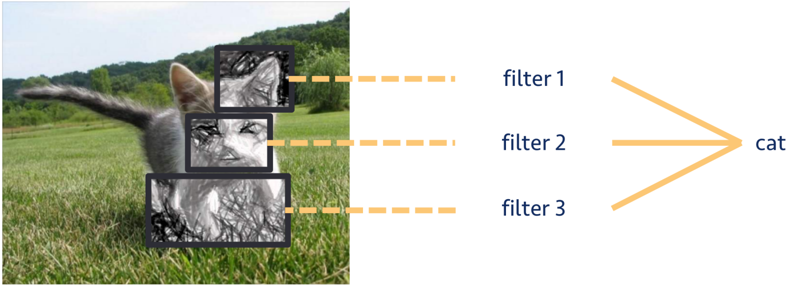
CNN [recap]

AlexNet



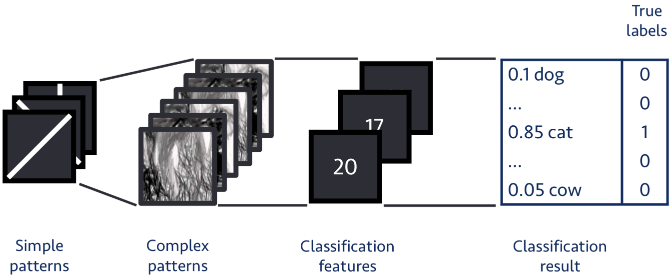
CNN [recap]

patterns

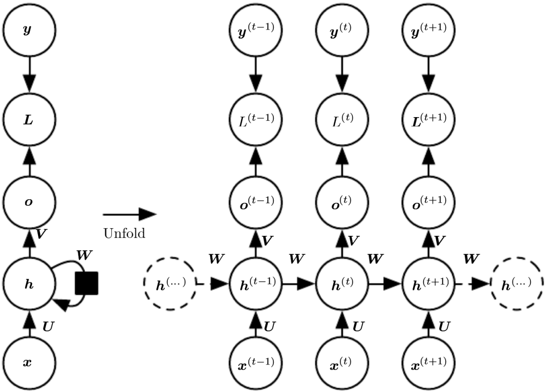


CNN [recap]

patterns

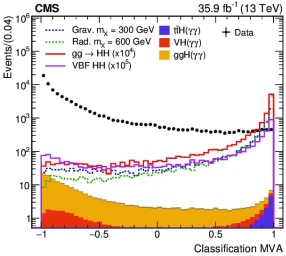


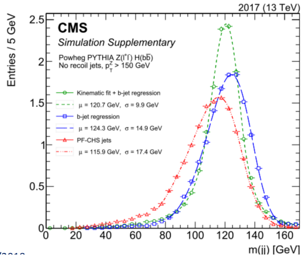
RNN [recap]
vanilla edition



Good old classics

- classification
  - μ, e^-, τ ID
  - sig vs. bkgr
- regression
  - mass resolution ($H \to bb$)



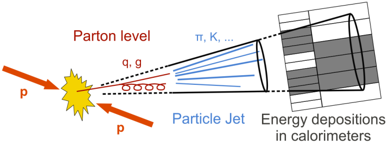


Jets

tagging

Tagging
why bother

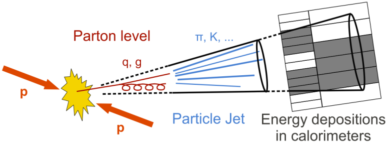
- distinguish btw. b/c/uds/g
- tag t/W/Z/H
- measure SM (e.g. VH(bb), ttH, H $\rightarrow c\bar{c}$)
- search for new (X $\rightarrow b\bar{b}$)



Tagging

why bother

- distinguish btw. b/c/uds/g
- tag t/W/Z/H
- measure SM (e.g. VH(bb), ttH, $H \to c\bar{c}$)
- search for new ($X \to b\bar{b}$)



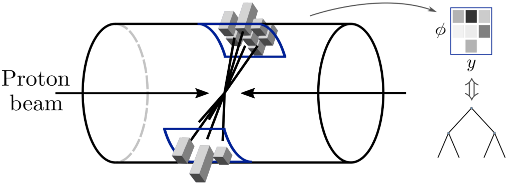
How would you represent a jet?

What's a jet, anyway?

A jet can be described in a variety of ways

- Sum of its particle constituents
- Image of energy deposits in rapidity-azimuth
- Complete basis of observables (e.g. energy flow polynomials)
- Image of primary Lund declustering sequence
- Binary tree associated with the jet clustering sequence
- ...

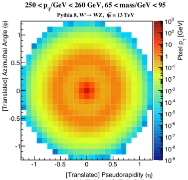
Most of them involve some trade-off between interpretability, representability, robustness and performance.



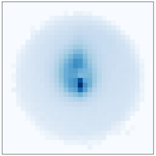
Frédéric Dreyer 4/14

[1]

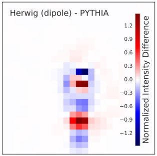
As Images...



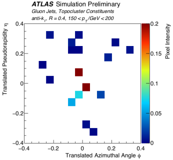
1511.05190



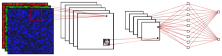
1603.09349



1609.00607



ATL-PHYS-PUB-2017-017



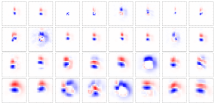
red = transverse momenta of charged particles

green = the transverse momenta of neutral particles

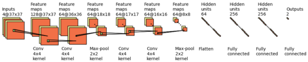
blue = charged particle multiplicity

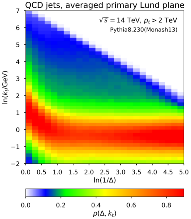
1612.01551

1701.08784



1803.00107



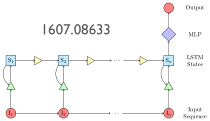


1807.04758

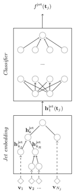
And more...

3 [2]

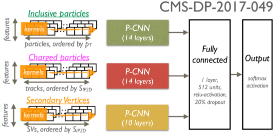
As Collections of Particles...



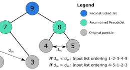
1607.08633



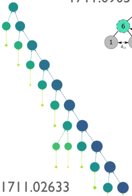
1702.00748



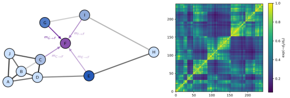
CMS-DP-2017-049



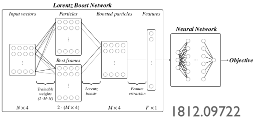
1711.09059



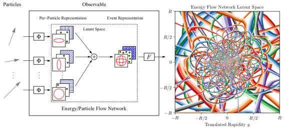
1711.02633



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



1812.09722



1810.05165

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

$$\text{with} \quad C = \begin{pmatrix} 1 & 1 & \cdots & 0 & \chi_1 & \cdots & 0 & C_{1,N+2} & \cdots & C_{1,M} \\ \vdots & & \ddots & & & \ddots & & \vdots & \ddots & \vdots \\ 1 & 0 & \cdots & 1 & 0 & \cdots & \chi_N & C_{N,N+2} & \cdots & C_{N,M} \end{pmatrix} . \qquad \tilde{k}_j \xrightarrow{\text{LoLa}} \hat{k}_j = \begin{pmatrix} m^2(\tilde{k}_j) \\ p_T(\tilde{k}_j) \\ p_T(\tilde{k}_j)\Delta R_{j,\text{jet}} \\ w_{jm}^{(E)} E(\tilde{k}_m) \\ w_{jm}^{(d)} d_{jm}^2 \\ E_T(\tilde{k}_j) E_T(\tilde{k}_m)(\Delta R_{jm})^{0.2} \end{pmatrix} ,$$

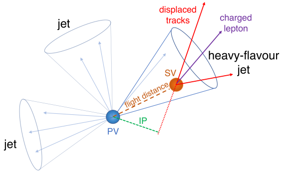
1707.08966, 1812.09223

And more...

4 [2]

ParticleNet: Jet Tagging via Particle Clouds - April 17, 2019 - Huilin Qu (UCSB)

Tagging
CMS taggers



circa 2017

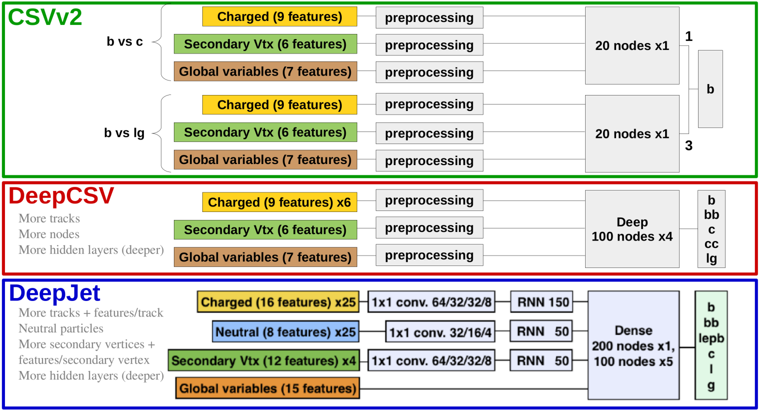
- JP/JBP
- CSVv1/2
- SE/SM
- cMVAv2

circa 2019

- DeepCSV
- DeepJet
- DeepDoubleX
- DeepAK8
- DeepYouNameIt

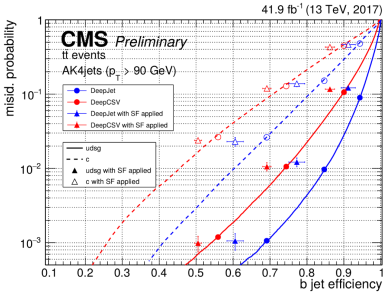
[link]

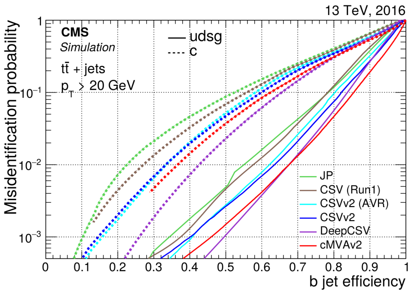
Tagging
CMS taggers



[1], [2]

Tagging
ROCs

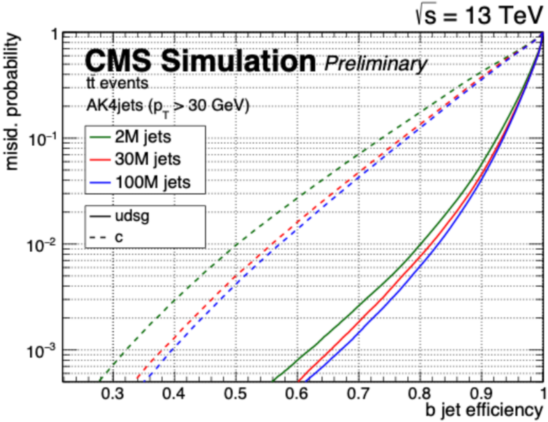




[1], [2]

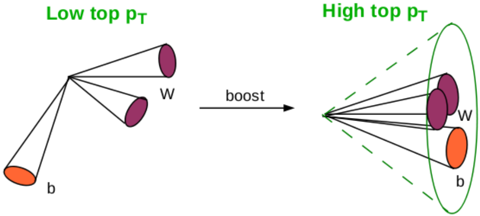
Tagging

ROCs



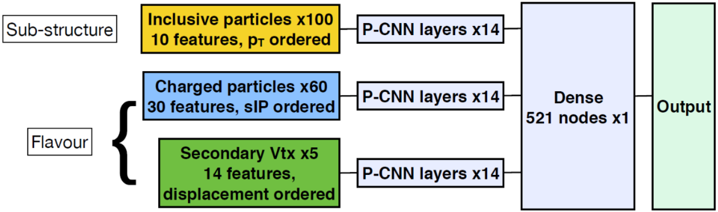
Tagging

boosted topology



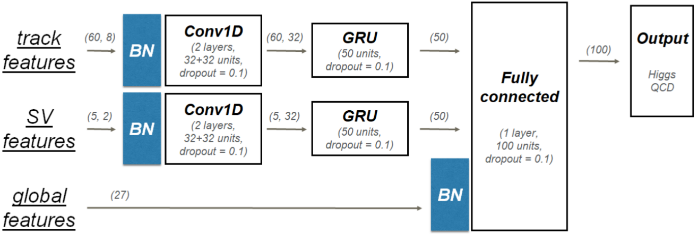
[link]

Tagging
DeepAK8



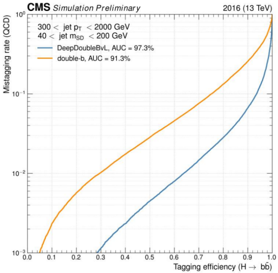
[link]

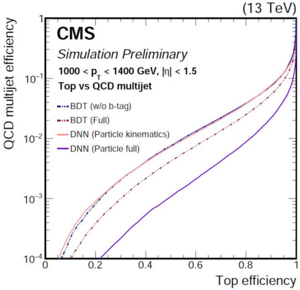
Tagging
DeepDoubleX



[link]

Tagging
boosted comparison

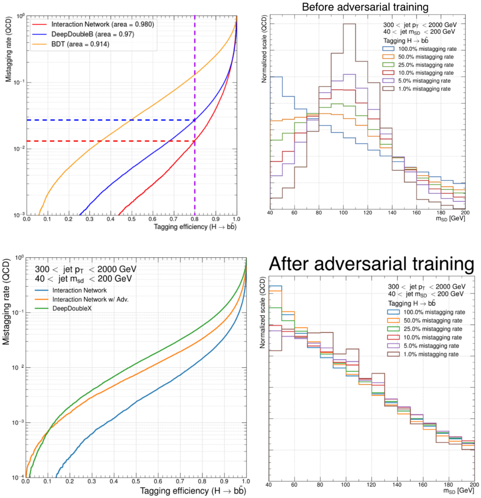




[link]

Tagging

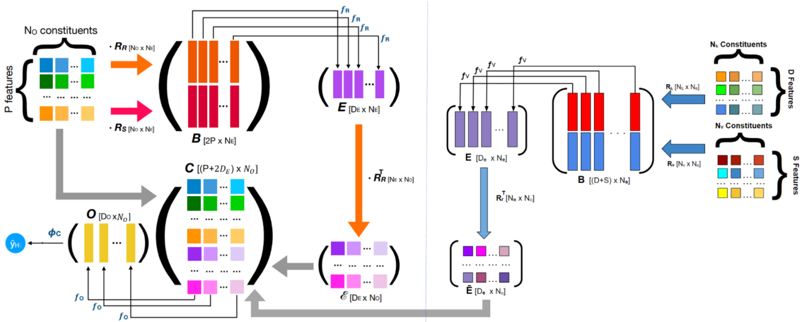
Interaction



[link]

Tagging

Interaction



Jets

state of the art

Jets

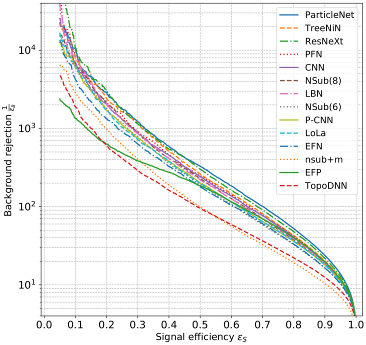
state of the art

(in top tagging)

you should definitely read [this]

State of the Art

top taggers

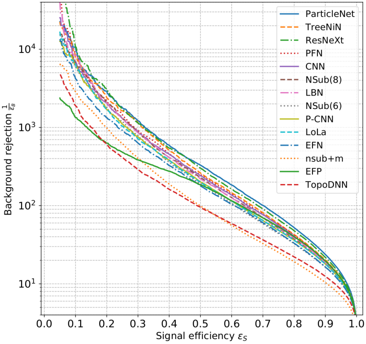


	AUC	Acc	$1/\epsilon_B$ ($\epsilon_S = 0.3$) single	mean	median	#Param
CNN [16]	0.981	0.930	914±14	995±15	975±18	610k
ResNeXt [30]	0.984	0.936	1122±47	1270±28	1286±31	1.46M
TopoDNN [18]	0.972	0.916	295±5	382± 5	378 ± 8	59k
Multi-body N-subjettiness 6 [24]	0.979	0.922	792±18	798±12	808±13	57k
Multi-body 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

you should definitely read [this]

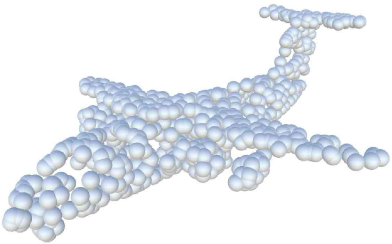
State of the Art

top taggers



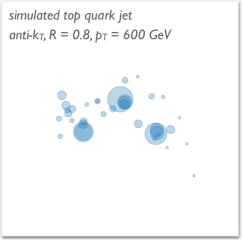
	AUC	Acc	$1/\epsilon_B$ ($\epsilon_S = 0.3$) single	mean	median	#Param
CNN [16]	0.981	0.930	914±14	995±15	975±18	610k
ResNeXt [30]	0.984	0.936	1122±47	1270±28	1286±31	1.46M
TopoDNN [18]	0.972	0.916	295±5	382± 5	378 ± 8	59k
Multi-body N-subjettiness 6 [24]	0.979	0.922	792±18	798±12	808±13	57k
Multi-body 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

As... Point Clouds?



arXiv:1801.07829

Jet as a ~~Point~~ Cloud
Particle

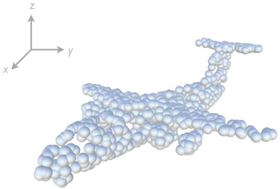


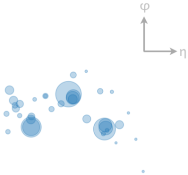
Jet (Particle cloud)

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



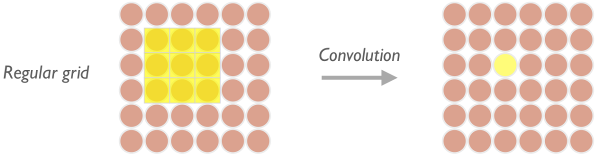


- Point cloud
  - points are intrinsically *unordered*
  - primary information:
    - 3D coordinates in the xyz space
- Particle cloud
  - particles are intrinsically *unordered*
  - primary information:
    - 2D coordinates in the η-φ space
  - *but also many additional features!*
    - energy/momenta
    - charge/particle ID
    - track quality/impact parameters/etc.

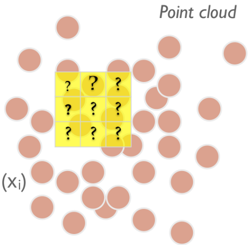
Why Particle Cloud?

- 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

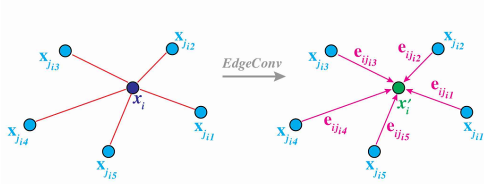
Convolution on Regular Grids



- 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

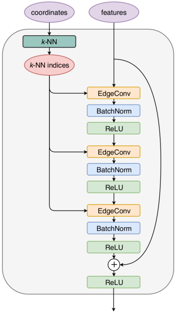


- Convolution on point clouds: *EdgeConv* [arXiv:1801.07829]
  - defining the local patch
    - for each point, a local patch is defined by finding its k-nearest neighbors
    - since points have coordinates, distance can be naturally defined
  - designing a symmetric "convolution" function
    - define "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

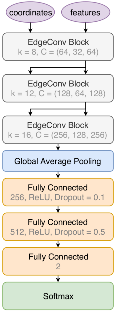
from [link]

State of the Art

ParticleNet



EdgeConv block (inspired by ResNet)

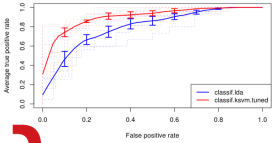


ParticleNet architecture

	AUC	Acc	$1/\epsilon_B$ ($\epsilon_S = 0.3$)			#Param
			single	mean	median	
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

Current ML uncertainties

Figure: https://mlr.mlr-org.com



1) Multiple training runs
   - Weight initialisation
2) Different training data
   - Training statistical uncertainty
3) Different testing data
   - Testing statistical uncertainty

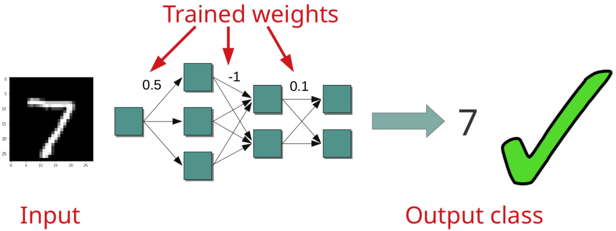
e.g. cross validation

No out-of-distribution or systematic uncertainty

3 / 25

Example: Bayesian motivation

Example: 1. Train a neural network on MNIST
2. Test on MNIST test data
3. Success

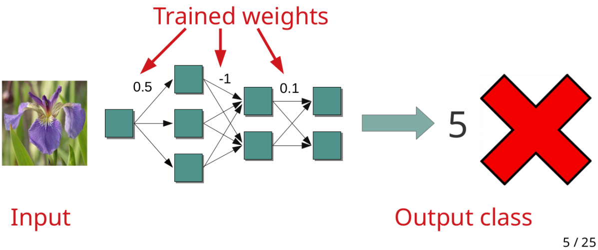


4 / 25

MNIST: LeCun et al., 1998a

Example: Bayesian motivation

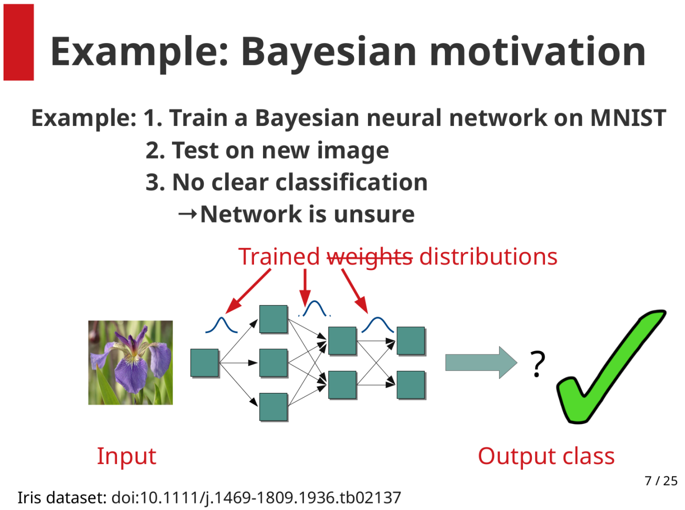
Example: 1. Train a neural network on MNIST
2. Test on ~~MNIST test data~~ new image
3. ~~Success~~ Confident, incorrect answer

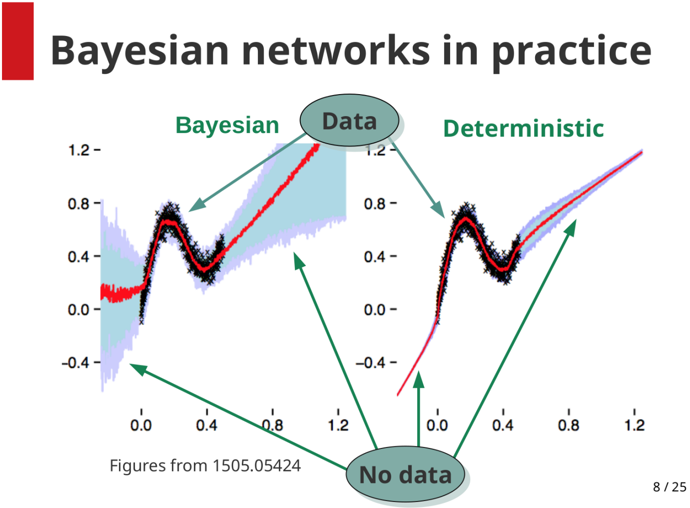


Iris dataset: doi:10.1111/j.1469-1809.1936.tb02137

from [link]

Going bayesian

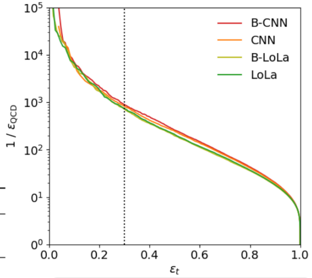



Bayesian networks in practice
Bayesian
Data
Deterministic
1.2
0.8
0.4
0.0
-0.4
0.0
0.4
0.8
1.2
No data
Figures from 1505.05424
8 / 25

Bayesian networks as taggers

Bayesian ~ deterministic
→ State-of-the-art taggers

		AUC	$1/\epsilon_{QCD}$ for $\epsilon_t = 30\%$
CNN		0.982	820
B-CNN		0.982	900
LoLa	$N_{const} = 40$	0.979	640
B-Lola		0.979	600
LoLa	$N_{const} = 100$	0.981	740
B-Lola		0.980	680
LoLa	$N_{const} = 200$	0.979	590
B-Lola		0.979	630

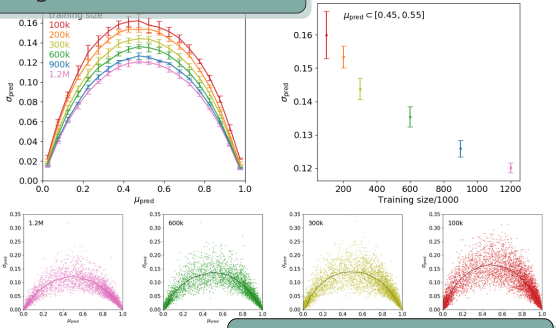


Works with:
- Images (CNN)
- Constituents (LoLa)

12 / 25

Statistical uncertainties

Strong correlation with mean



Training size ↑ uncertainty ↓

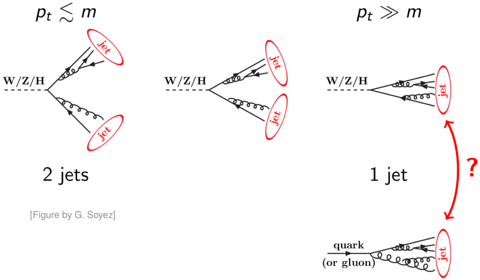
13 / 25

Jets
misc.

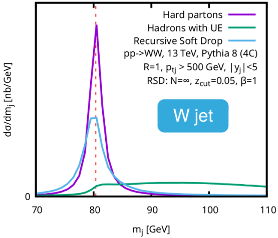
Grooming

with RL

from [link]



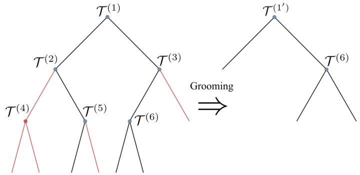
[Figure by G. Soyez]

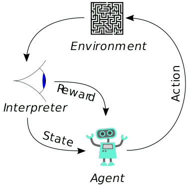


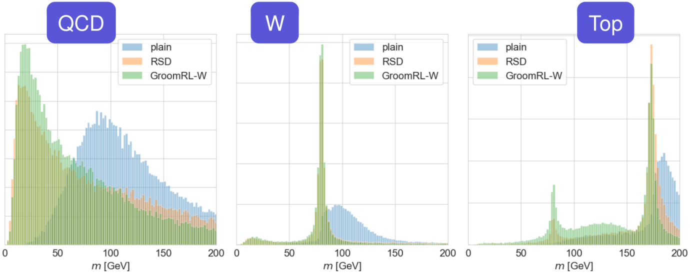
from [link]

Grooming

with RL

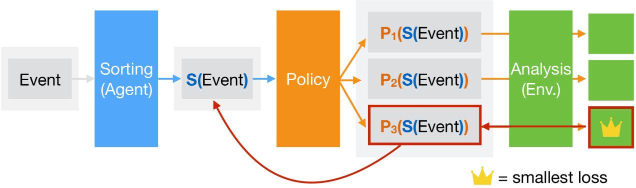


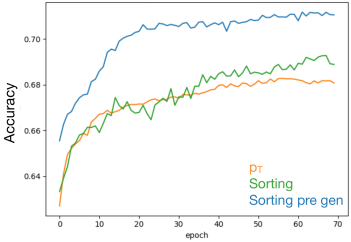




from [link]

Particle sorting
with RL

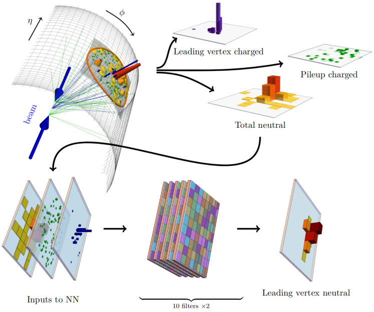


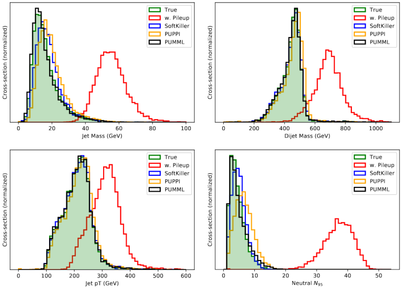


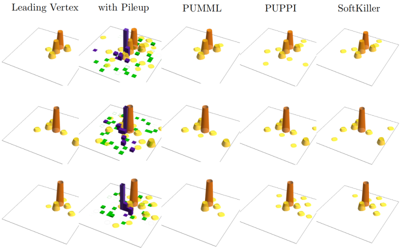
from [paper]

PU mitigation

PUMML



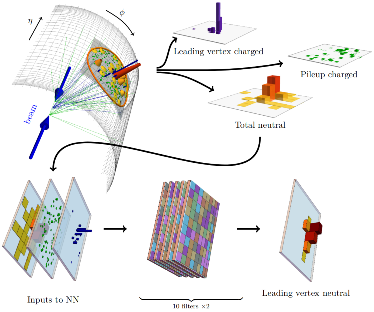


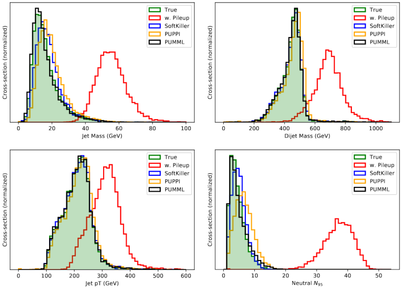


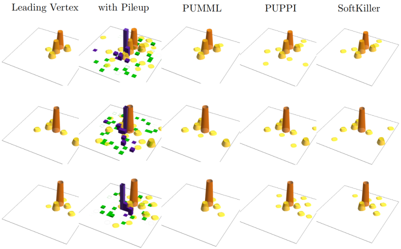
from [paper]

PU mitigation

PUMML







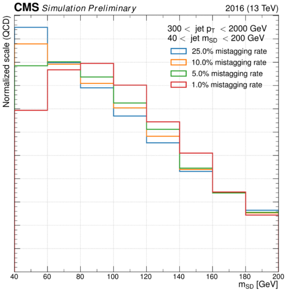
which branch of ML?

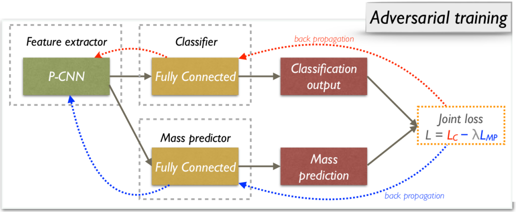
Going adversarial

[link]

Going adversarial

Mass sculpting

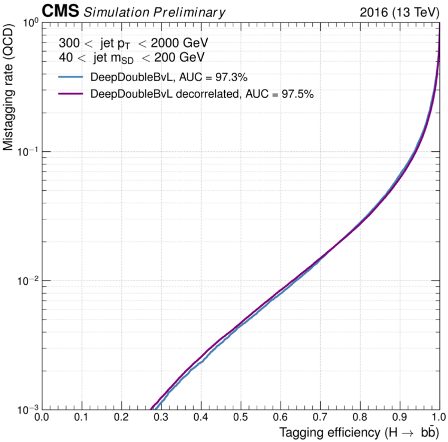




pic from [1]
sculpting in B physics [2]
uBoost [3], and more on mass decorr. [4]

Going adversarial

Mass sculpting

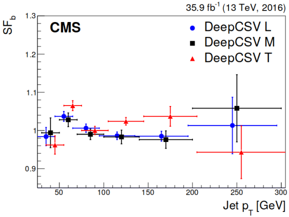


[ordinary SF link]

Going adversarial

DeepSF

- MC used for training and data differ
  - $\rightarrow$ reweight events to match MC and data
- use Tag&Probe method in bins of p_T, η
  - $\rightarrow$ SF(p_T, η)

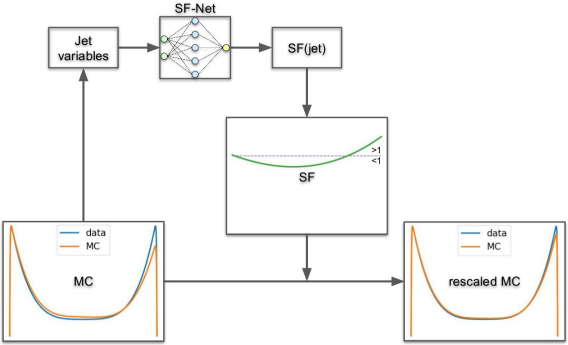


- discrete values
- should have enough statistics in bins
- can't go many variables
- babysitting with bins choice and fitting

[DeepSF link]

Going adversarial

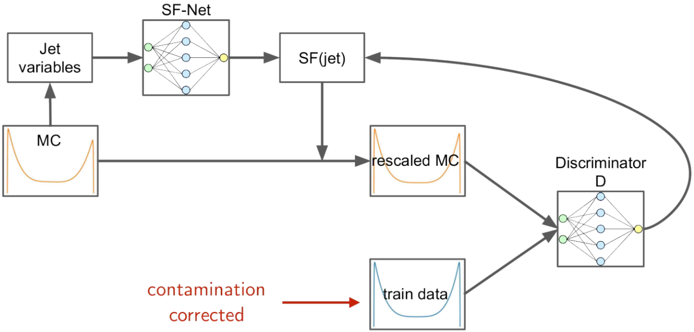
DeepSF



[DeepSF link]

Going adversarial

DeepSF

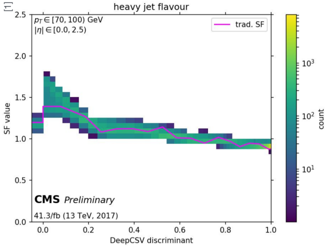


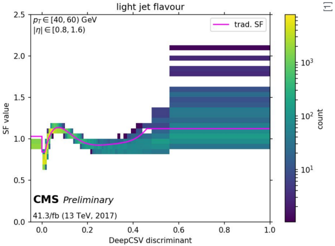
contamination corrected

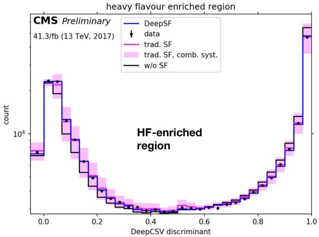
fancy BDT reweighting [link]

Going adversarial

DeepSF



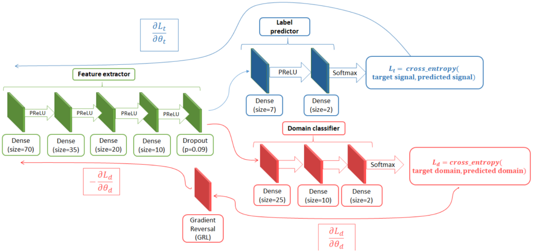


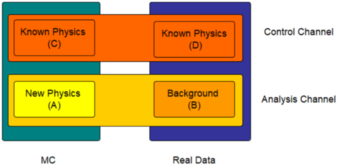


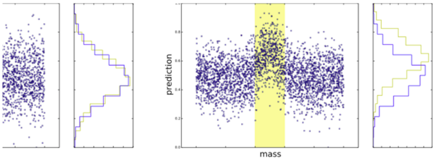
more on adversarial methods [1], [2]
Learning to Pivot [3]

Going adversarial

Domain adaptation







GAN & VAE

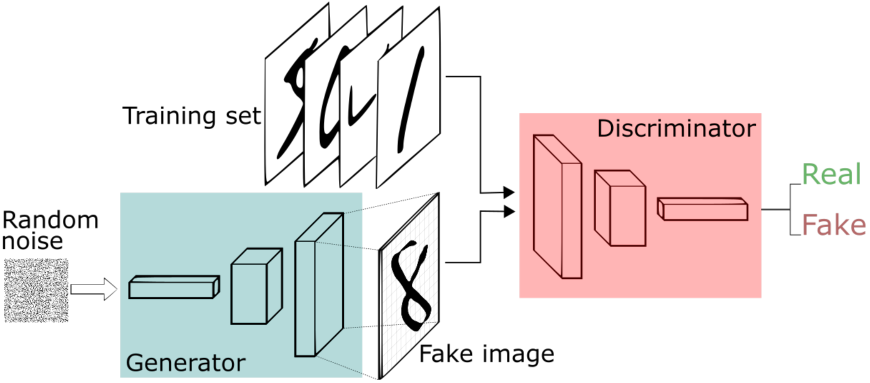
GAN & VAE

Generative Adversarial Network
Variational AutoEncoder

[more here]

GAN

vanilla edition



- GAN - Generative Adversarial Networks
- 3D-GAN - Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling
- acGAN - Face Aging With Conditional Generative Adversarial Networks
- AC-GAN - Conditional Image Synthesis With Auxiliary Classifier GANs
- AdaGAN - AdaGAN: Boosting Generative Models
- AEGAN - Learning Inverse Mapping by Autoencoder based Generative Adversarial Nets
- AffGAN - Amortised MAP Inference for Image Super-resolution
- AL-CGAN - Learning to Generate Images of Outdoor Scenes from Attributes and Semantic Layouts
- ALI - Adversarially Learned Inference
- AM-GAN - Generative Adversarial Nets with Labeled Data by Activation Maximization
- AnoGAN - Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery
- ArtGAN - ArtGAN: Artwork Synthesis with Conditional Categorial GANs
- b-GAN - b-GAN: Unified Framework of Generative Adversarial Networks
- Bayesian GAN - Deep and Hierarchical Implicit Models
- BEGAN - BEGAN: Boundary Equilibrium Generative Adversarial Networks
- BiGAN - Adversarial Feature Learning
- BS-GAN - Boundary-Seeking Generative Adversarial Networks
- CGAN - Conditional Generative Adversarial Nets
- CaloGAN - CaloGAN: Simulating 3D High Energy Particle Showers in Multi-Layer Electromagnetic Calorimeters with Generative Adversarial Networks
- CCGAN - Semi-Supervised Learning with Context-Conditional Generative Adversarial Networks
- CatGAN - Unsupervised and Semi-supervised Learning with Categorical Generative Adversarial Networks
- CoGAN - Coupled Generative Adversarial Networks
- Context-RNN-GAN - Contextual RNN-GANs for Abstract Reasoning Diagram Generation
- C-RNN-GAN - C-RNN-GAN: Continuous recurrent neural networks with adversarial training
- CS-GAN - Improving Neural Machine Translation with Conditional Sequence Generative Adversarial Nets
- CVAE-GAN - CVAE-GAN: Fine-Grained Image Generation through Asymmetric Training
- CycleGAN - Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks
- DTN - Unsupervised Cross-Domain Image Generation
- DCGAN - Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks
- DiscoGAN - Learning to Discover Cross-Domain Relations with Generative Adversarial Networks
- DR-GAN - Disentangled Representation Learning GAN for Pose-Invariant Face Recognition
- DualGAN - DualGAN: Unsupervised Dual Learning for Image-to-Image Translation
- EBGAN - Energy-based Generative Adversarial Network
- f-GAN - f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization
- FF-GAN - Towards Large-Pose Face Frontalization in the Wild
- GAWWN - Learning What and Where to Draw
- GeneGAN - GeneGAN: Learning Object Transfiguration and Attribute Subspace from Unpaired Data
- Geometric GAN - Geometric GAN
- GoGAN - Gang of GANs: Generative Adversarial Networks with Maximum Margin Ranking
- GP-GAN - GP-GAN: Towards Realistic High-Resolution Image Blending
- IAN - Neural Photo Editing with Introspective Adversarial Networks
- iGAN - Generative Visual Manipulation on the Natural Image Manifold
- IcGAN - Invertible Conditional GANs for image editing
- ID-CGAN - Image De-raining Using a Conditional Generative Adversarial Network
- Improved GAN - Improved Techniques for Training GANs
- InfoGAN - InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets
- LAGAN - Learning Particle Physics by Example: Location-Aware Generative Adversarial Networks for Physics Synthesis
- LAPGAN - Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks

pic from [1]
more fancy [2]

GAN

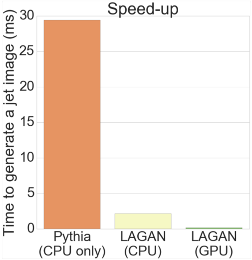
- Really fast simulation

GAN

motivation

- Really fast simulation

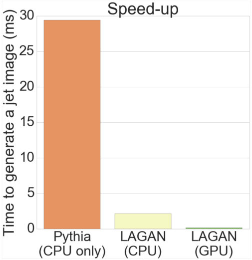
  in fact $10^2 \div 10^5$ faster than Geant



Generation Method	Hardware	Batch Size	milliseconds/shower
GEANT4	CPU	N/A	1772
CALOGAN	CPU	1	13.1
		10	5.11
		128	2.19
		1024	2.03
	GPU	1	14.5
		4	3.68
		128	0.021
		512	0.014
		1024	0.012

GAN

motivation



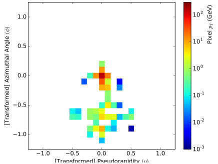
- Really fast simulation

  in fact $10^2 \div 10^5$ faster than Geant
- Save space
- Rare events

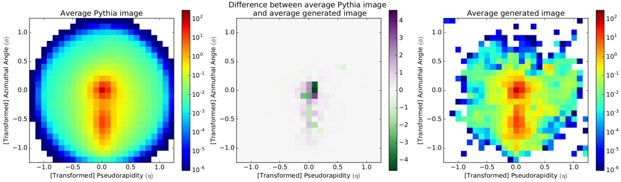
Generation Method	Hardware	Batch Size	milliseconds/shower
GEANT4	CPU	N/A	1772
CALOGAN	CPU	1	13.1
		10	5.11
		128	2.19
		1024	2.03
	GPU	1	14.5
		4	3.68
		128	0.021
		512	0.014
		1024	0.012

GAN

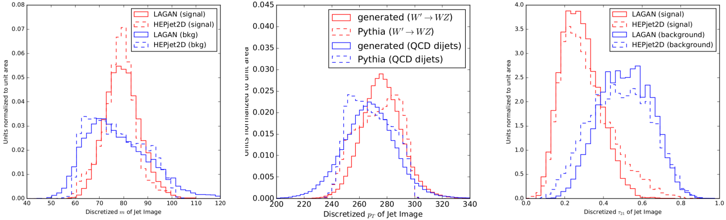
LAGAN



[Transformed] Azimuthal Angle (ϕ)
[Transformed] Pseudorapidity (η)
Pixel p_T (GeV)



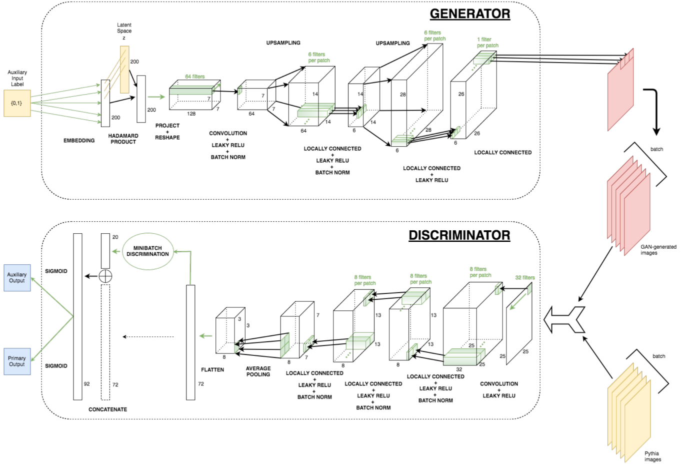
Average Pythia image
Difference between average Pythia image and average generated image
Average generated image
[Transformed] Azimuthal Angle (ϕ)
[Transformed] Pseudorapidity (η)



LAGAN (signal)
HEPjet2D (signal)
LAGAN (bkg)
HEPjet2D (bkg)
Units normalized to unit area
Discretized m of Jet Image
generated ($W' \to WZ$)
Pythia ($W' \to WZ$)
generated (QCD dijets)
Pythia (QCD dijets)
Discretized p_T of Jet Image
LAGAN (signal)
HEPjet2D (signal)
LAGAN (background)
HEPjet2D (background)
Units normalized to unit area
Discretized τ_{21} of Jet Image

[paper]

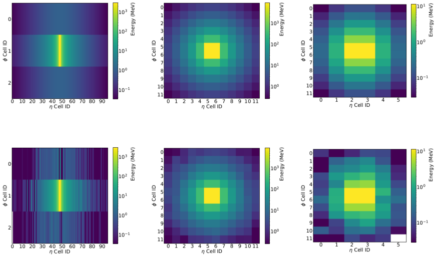
GAN
LAGAN

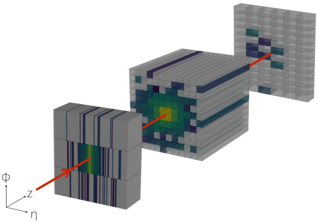


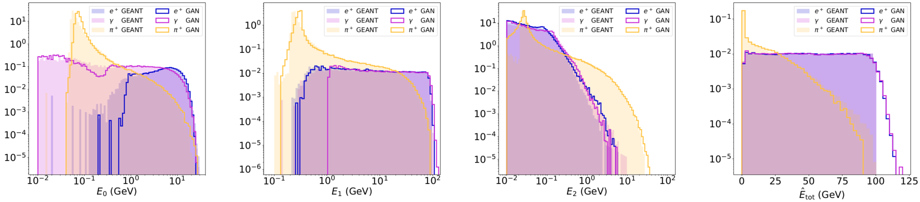
[paper]

GAN

CaloGAN



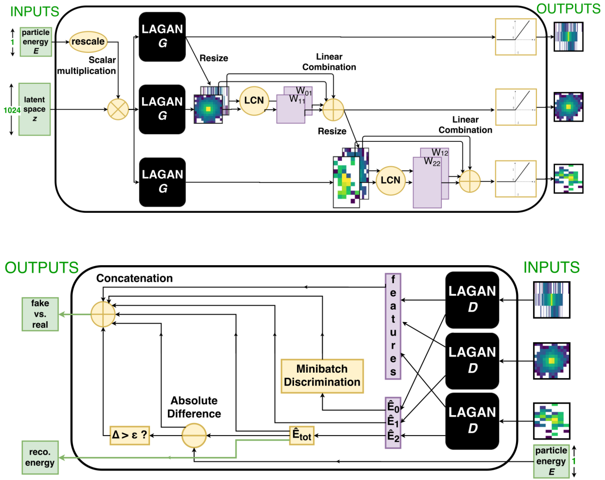




[paper]

GAN

CaloGAN

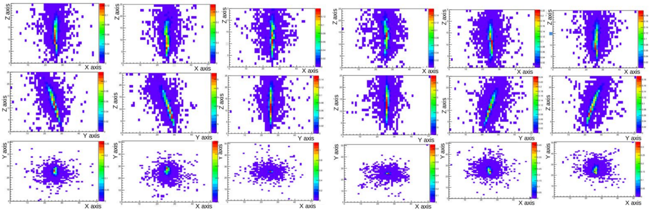


[link]

GAN

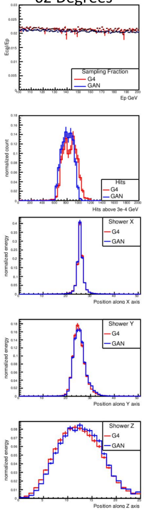
3DGAN

111.07 GeV and 115.54 Degrees	147.49 GeV and 87.83 Degrees	188.95 GeV and 62.96 Degrees



G4 GAN G4 GAN G4 GAN

62 Degrees

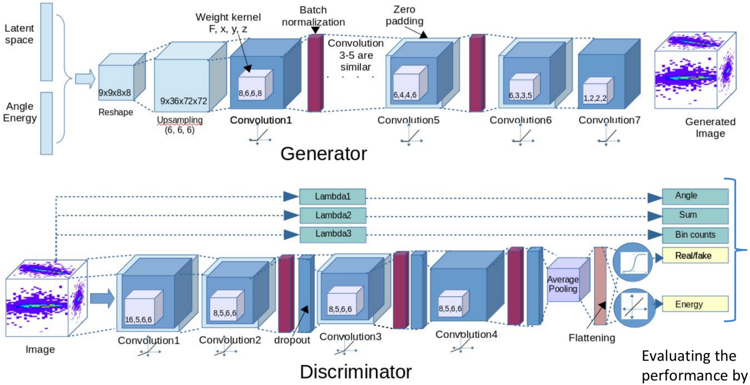


Time to create an electron shower for Fixed angle		
Method	**Machine**	**Time/Shower (msec)**
Full Simulation Fixed Energy (Geant 4)	Intel Xeon Platinum 8180	17000
3DGAN (Fixed Angle) (batch size 128)	Intel Xeon Platinum 8180(TF 1.12)	1
	GTX 1080	0.1
3DGAN (Variable Angle) (batch size 64)	GTX 1080	4.6

[link]

GAN

3DGAN



Generator Params 1066 k
Discriminator Params = 87 k

Evaluating the performance by agreement to labels and Physics related constraints

more GANs

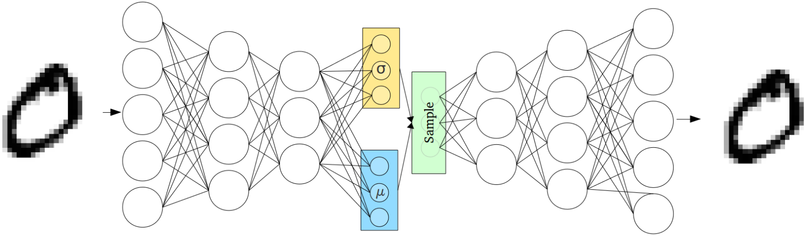
- Generate kinematics
  - DijetGAN (conv) - gen dijet QCD/$t\bar{t}$ [1]
  - Particle-GAN (dense+GRU) - gen min-bias PU? [2]
  - LHCb RICH DLL (dense) [3]
- Generate showers pictures
  - ATLAS ECal GAN (conv) - full ECal geometry accounted for [4]
  - LHCb Ecal GAN (conv) - 2D pics, as usual [5]
  - Belle Pixel bkgr (conv) [6]
  - * GraphRNN + GraphCNN + Mixture Density Network
    3D EM showers in SHiP [7]

more GANs

- Generate kinematics
  - DijetGAN (conv) - gen dijet QCD/$t\bar{t}$ [1]
  - Particle-GAN (dense+GRU) - gen min-bias PU? [2]
  - LHCb RICH DLL (dense) [3]
- Generate showers pictures
  - ATLAS ECal GAN (conv) - full ECal geometry accounted for [4]
  - LHCb Ecal GAN (conv) - 2D pics, as usual [5]
  - Belle Pixel bkgr (conv) [6]
  * GraphRNN + GraphCNN + Mixture Density Network
    3D EM showers in SHiP [7]

what about VAE?

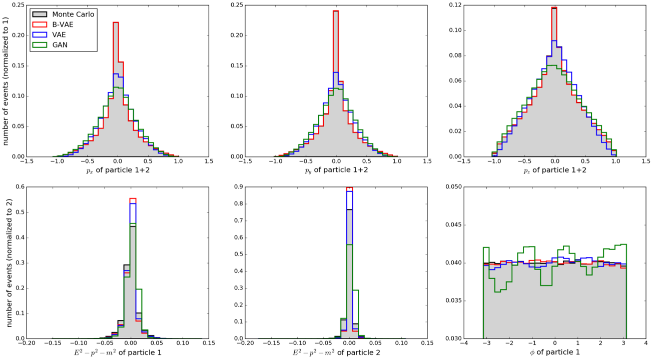
VAE



from [link]

VAE

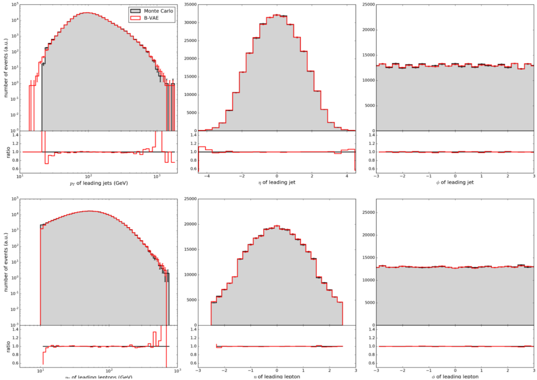
2-body decay



from [link]

VAE

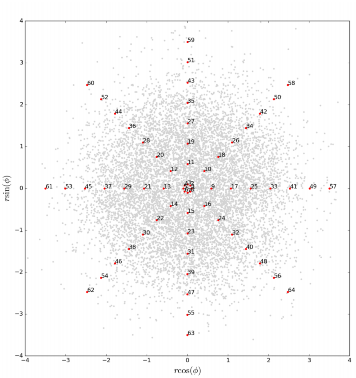
$t\bar{t}$

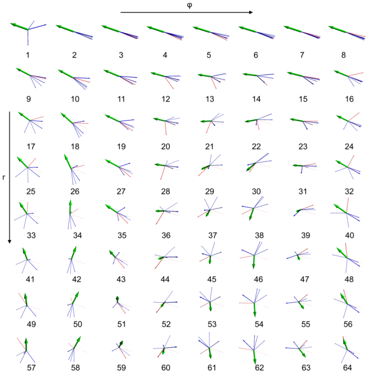


from [link]

VAE

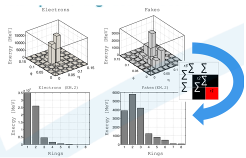
latent space

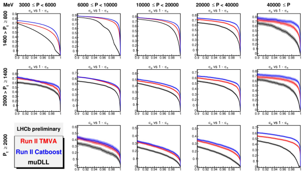




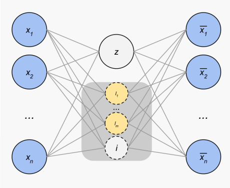
Trigger

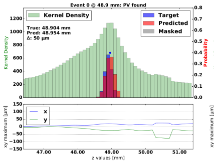
Trigger



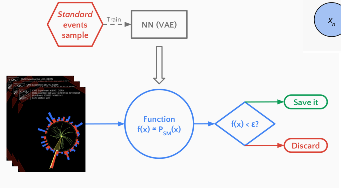


- e$^-$ HLT in ATLAS - dense on rings of energy [1]
- PV finder in LHCb - CNN on 1D z-KDE + regression [2]
- hls4ml - how to deploy DL models on firmware [3]
- Muon ID in LHCb - CatBoost [4]
- Model-independent anomaly detection - VAE [5]
- HLT rate monitoring in CMS - VAE [6]





PV found example



Tracks

from [link]

Clustering showers

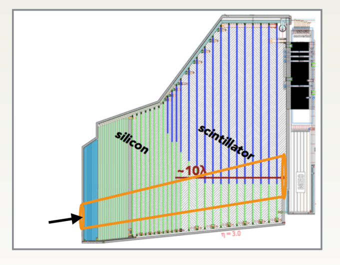
? sparse showers

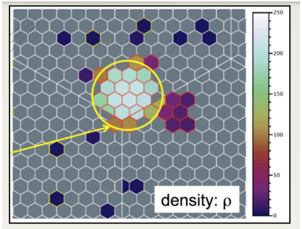
? irregular geometry

? CNN

→ (good old) point clouds

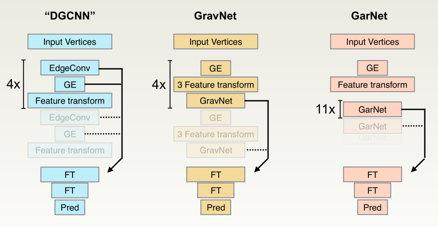
→ modified graph nets - GravNet and GarNet

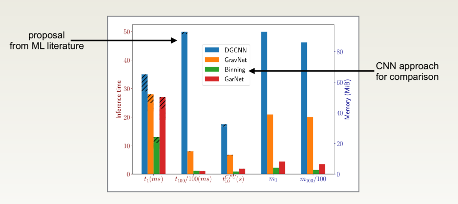


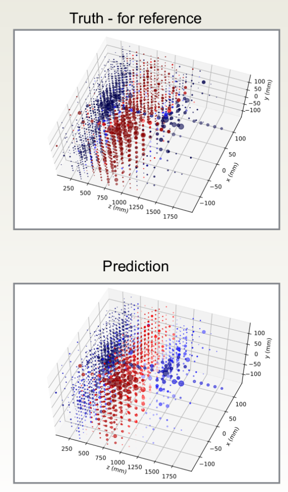


from [link]

Clustering showers

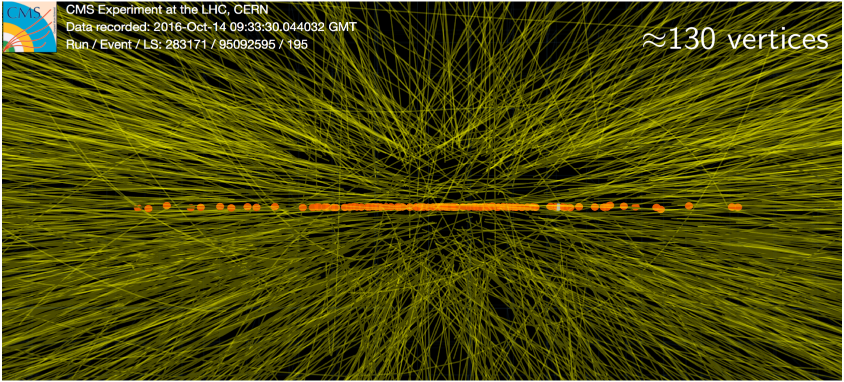






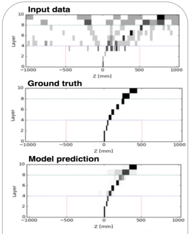
Tracks

HEP.TrkX

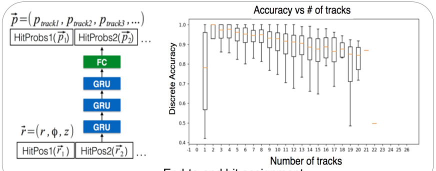


Tracks

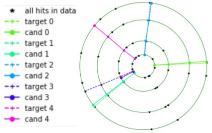
HEP.TrkX



Track following with RNN



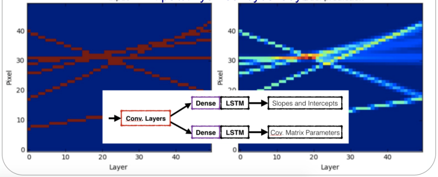
End-to-end hit assignment

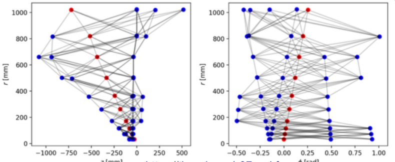


Seq-to-seq track finding

https://heptrkx.github.io/

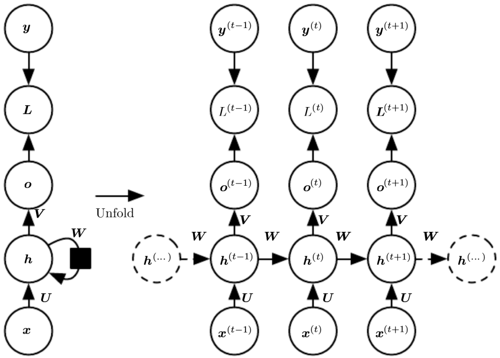
https://tinyurl.com/yb3v93y9





https://tinyurl.com/y87saehf

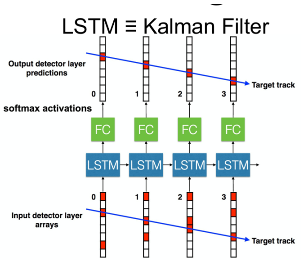
Tracks
HEP.TrkX



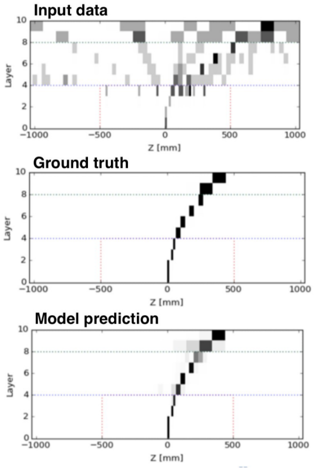
from [link]

Tracks

HEP.TrkX



- Search seeded from a known tracklet
- Hit location is discretized to fixed length
- Model predicts the binned position of the hit on the next layer

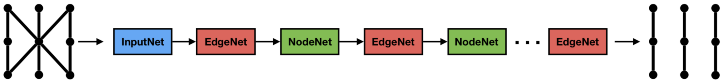


from [link]

Tracks

trying GNN

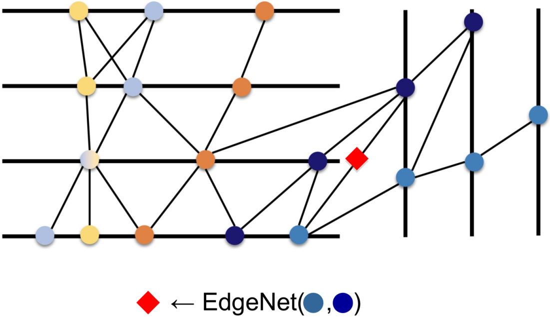
- Input Network
  - hit features $(r, \phi, z) \rightarrow$ node latent features
  - Dense: $3 \rightarrow N$
- Edge Network
  - node representations at both ends $\rightarrow$ edge weight
  - Dense: $N + N \rightarrow 1$
- Node Network
  - current node representation, w. sum of incoming edges and w.sum of outgoing edges $\rightarrow$ node latent representation
  - Dense: $N + N + N \rightarrow N$



from [link]

Tracks

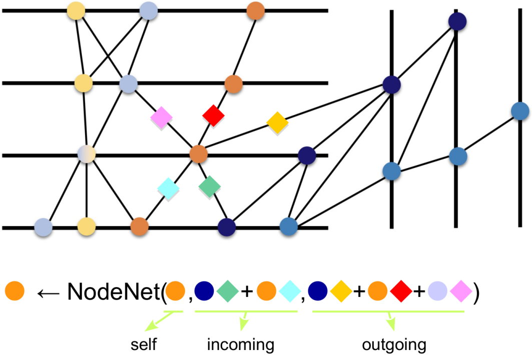
trying GNN



from [link]

Tracks

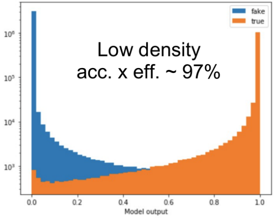
trying GNN

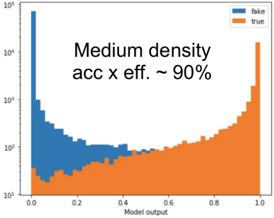


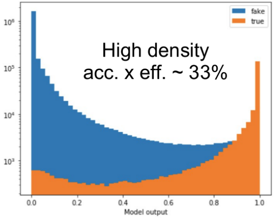
from [link]

Tracks

trying GNN







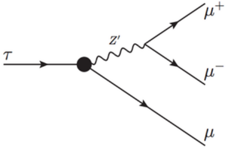
Kaggle

how to compete [1]
nice interview [2]

kaggle

- one of the largest data communities in the world
- ML competitions
- public datasets and kernels (code snippets)
- short-form AI education

2014	Higgs Boson challenge by ATLAS [1]
2015	$\tau \to \mu\mu\mu$ by LHCb and Yandex [2]
2018	TrackML by CERN [3], [4]

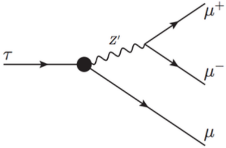


2014 Higgs Boson challenge by ATLAS [1]

2015 $\tau \to \mu\mu\mu$ by LHCb and Yandex [2]

2018 TrackML by CERN [3], [4]

huge impact on ML community in HEP



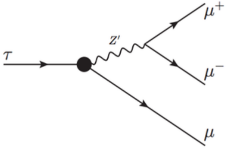
open data and baselines for practice!

2014 Higgs Boson challenge by ATLAS [1]

2015 $\tau \to \mu\mu\mu$ by LHCb and Yandex [2]

2018 TrackML by CERN [3], [4]

huge impact on ML community in HEP



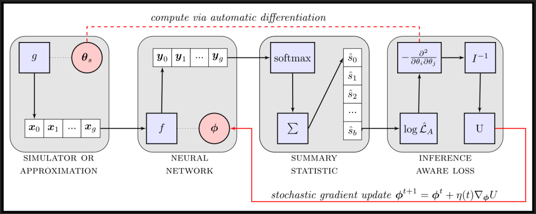
INFERNO

INFERNO

teaser

INFERENCE-AWARE NEURAL OPTIMISATION

An approach to learn non-linear summary statistics by directly minimizing an approximation the expected profiled (or marginalised) interval width accounting for the effect of nuisance parameters



check arxiv.org/abs/1806.04743 for a more detailed description.

INFERNO

if you dare to know

- why train NN in the first place
- and why it is not enough

INFERNO

if you dare to know

- why train NN in the first place
- and why it is not enough
- why you really need MC

INFERNO

if you dare to know

- why train NN in the first place
- and why it is not enough
- why you really need MC
- what is nuisance parameters
- and why bother with them
- and with test statistic choice

INFERNO

if you dare to know

- why train NN in the first place
- and why it is not enough
- why you really need MC
- what is nuisance parameters
- and why bother with them
- and with test statistic choice
- what is bigger inference perspective
- and why should you care

INFERNO

if you dare to know

- why train NN in the first place
- and why it is not enough [1]
- why you really need MC
- what is nuisance parameters [2]
- and why bother with them
- and with test statistic choice
- what is bigger inference perspective
- and why should you care

Closing remarks

(subjective here)

Closing remarks

future directions

- interpretability
  - what is NN learning? physics, perhaps?
  - do we train it properly?
  - can we fully understand it?

(subjective here)

Closing remarks

future directions

- interpretability
  - what is NN learning? physics, perhaps?
  - do we train it properly?
  - can we fully understand it?
- incorporating physics domain knowledge
  - systematics
  - physics laws and symmetries
  - model's place in HEP pipeline

(subjective here)

Closing remarks

future directions

- interpretability
  - what is NN learning? physics, perhaps?
  - do we train it properly?
  - can we fully understand it?
- incorporating physics domain knowledge
  - systematics
  - physics laws and symmetries
  - model's place in HEP pipeline
- close collaboration with ML community
  - Kaggle
  - industry experience
  - new architectures and concepts

(subjective here)

Closing remarks

about to come

- now
  - simple s/b classification in analyses
  - particle ID/tagging into production (e.g. CMSSW)
  - why don't you use ML in your analysis?

(subjective here)

Closing remarks

about to come

- now
  - simple s/b classification in analyses
  - particle ID/tagging into production (e.g. CMSSW)
  - why don't you use ML in your analysis?
- soon
  - ML at trigger level
  - complex taggers (top, boosted) into production
  - centralized simulation with GANs
  - track/object reconstruction
  - architecture competition and evolution (in fact, now)

Closing remarks

about to come

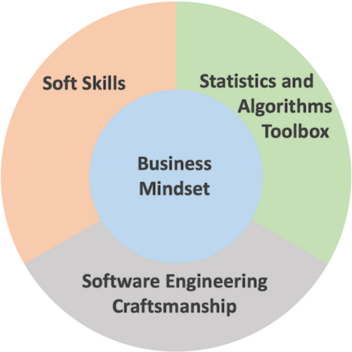
- now
  - simple s/b classification in analyses
  - particle ID/tagging into production (e.g. CMSSW)
  - why don't you use ML in your analysis?
- soon
  - ML at trigger level
  - complex taggers (top, boosted) into production
  - centralized simulation with GANs
  - track/object reconstruction
  - architecture competition and evolution (in fact, now)
- future
  - fully inference/systematics-aware learning
  - HEP specific DL libraries
  - end-to-end training
  - continuous assimilation with ML community
  - ...?

amazing talk on HEP&ML future [link]

from [link]

Closing remarks

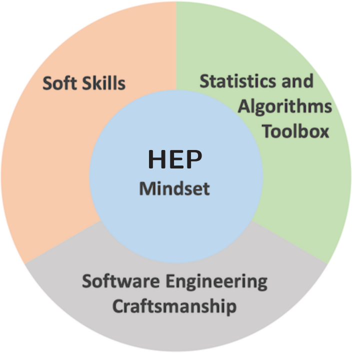
third wave data scientist



from [link]

Closing remarks

third wave data scientist in HEP



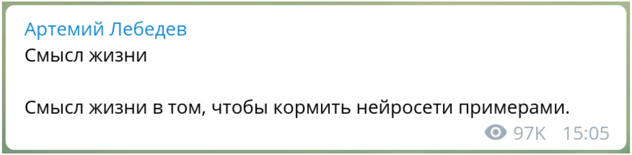
Summary

- Jets
  - b-tagging
  - t-tagging
  - grooming&PU
- Going adversarial
  - mass sculpting
  - DeepSF
  - domain adaptation
- Simulating with GAN&VAE
- Trigger
- Tracks

∗ Bonus
  - Kaggle
  - INFERNO

@yaourtpourtoi
oleg.filatov@cern.ch

Backup


Артемий Лебедев
Смысл жизни
Смысл жизни в том, чтобы кормить нейросети примерами.
97K
15:05