Deep Dive into Deep Learning for LHC and HL-LHC

Amir Farbin

ATLAS EXPERIMENT

UNIVERSITY OF TEXAS ARLINGTON

Introduction

- Utility of DL to HEP has been established in a variety of areas.
- This talk is not a comprehensive summary of all Deep Learning in HEP.
- Focus on next challenge: transition from feasibility to production
  - Attempt to speculate about the future…
- Disclaimer: I have a potential Conflict of Interest…

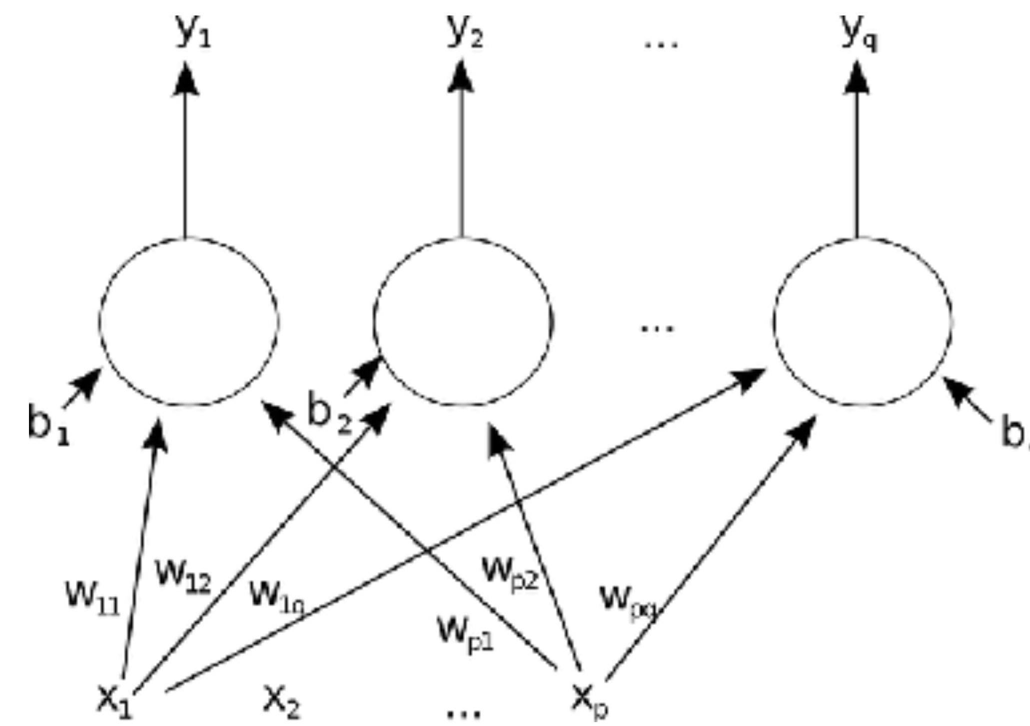
Menu

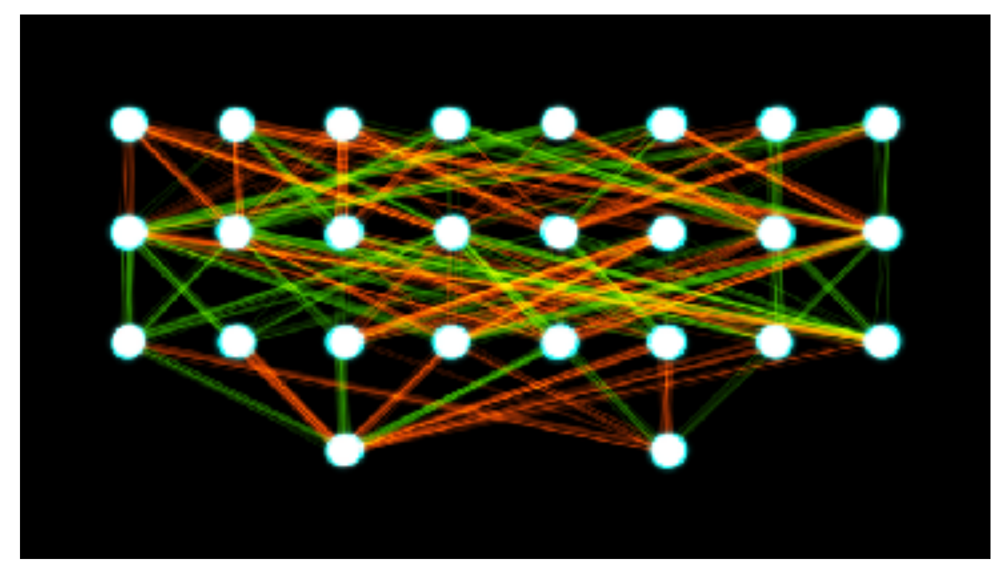
- Deep Learning
- DL in HEP
- 3 Areas:
  - Reconstruction / Trigger
  - Simulation
  - Analysis
    - Proposal
- Software and Technical Challenges
  - DL+HEP Software Needs
- Future
  - Science Fiction…

Deep Learning

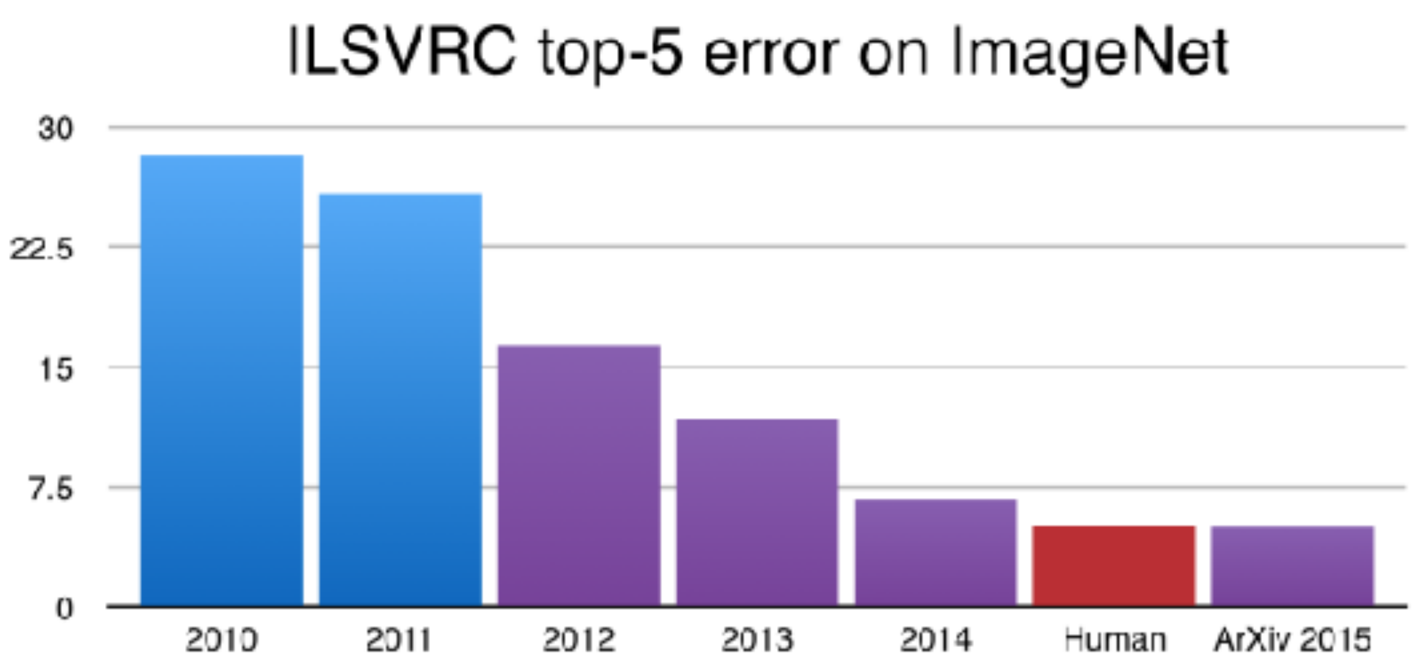
Artificial Neural Networks

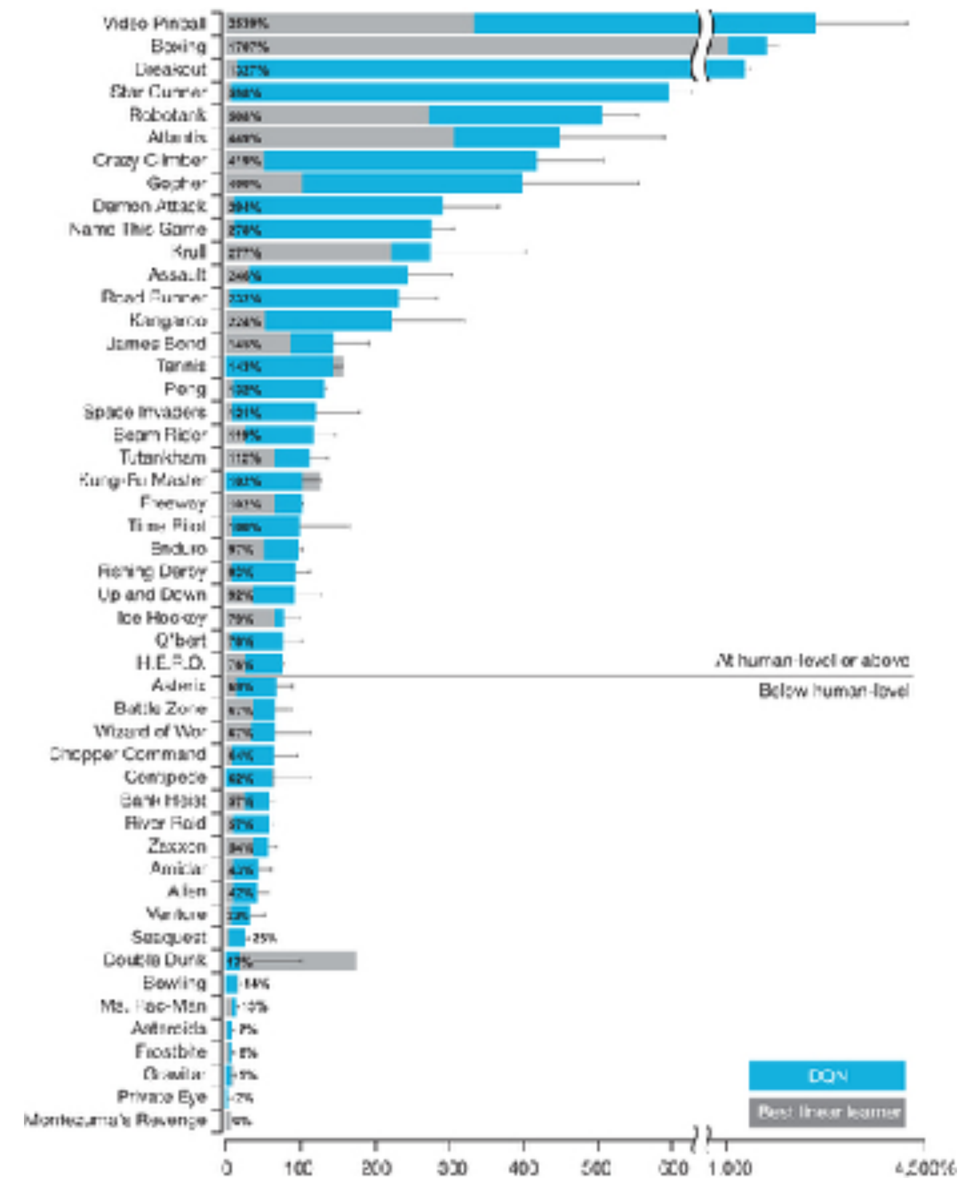
- ***Biologically inspired computation***, (first attempts in 1943)
    - ***Probabilistic Inference***: e.g. signal vs background
    - ***Universal Computation Theorem*** (1989)
- Multi-layer (***Deep***) Neutral Networks:
    - Not a new idea (1965), just impractical to train. ***Vanishing Gradient problem*** (1991)
    - Solutions:
        - New techniques: e.g. better activation or layer-wise training
        - ***More training***: big training datasets and lots of computation … ***big data and GPUs***
    - ***Deep Learning Renaissance***. First DNN in HEP (2014).
    - ***Amazing Feats***: Audio/Image/Video recognition, captioning, and generation. Text (sentiment) analysis. Language Translation. Video game playing agents.
    - ***Rich field***: Variety of architectures, techniques, and applications.



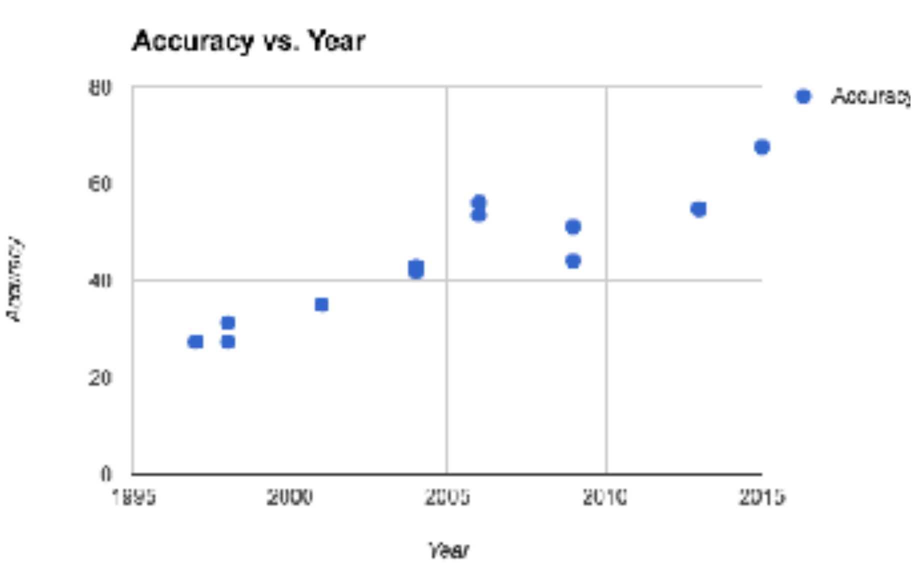


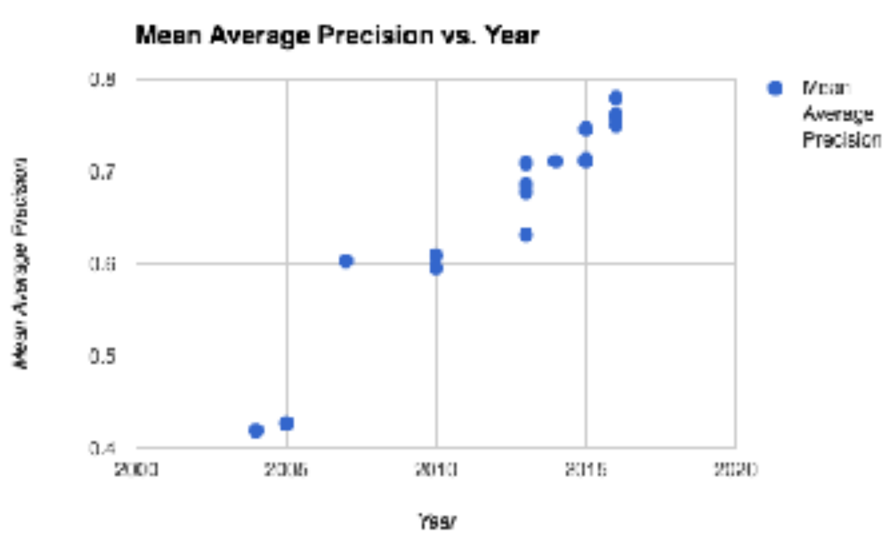
Images from Wikipedia

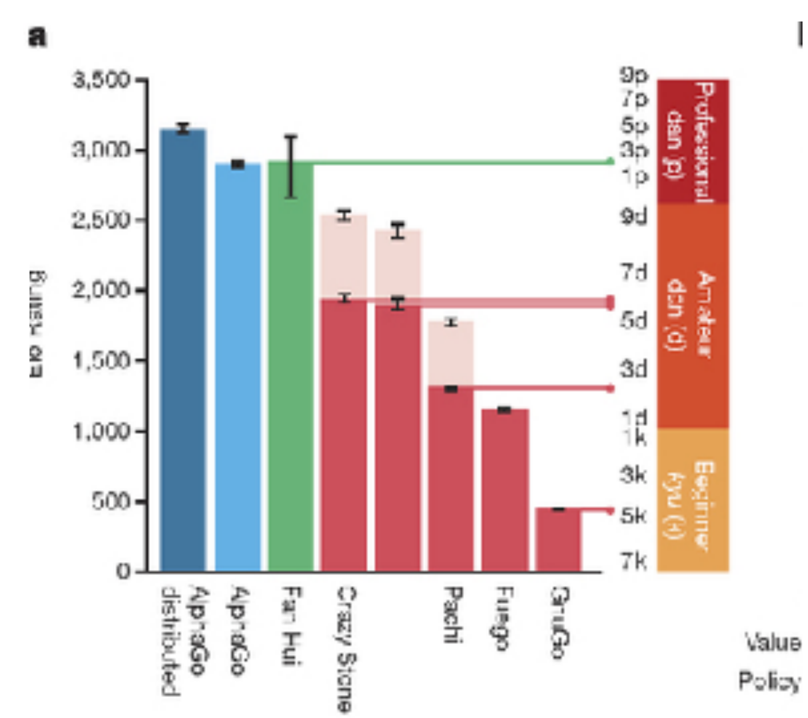




- Continuous server ASR word error rate (WER) reduction ~18% / year: combination of algorithms, data, and computing
- Deep learning (DNNs) is driving recent performance improvements in ASR and meaning extraction



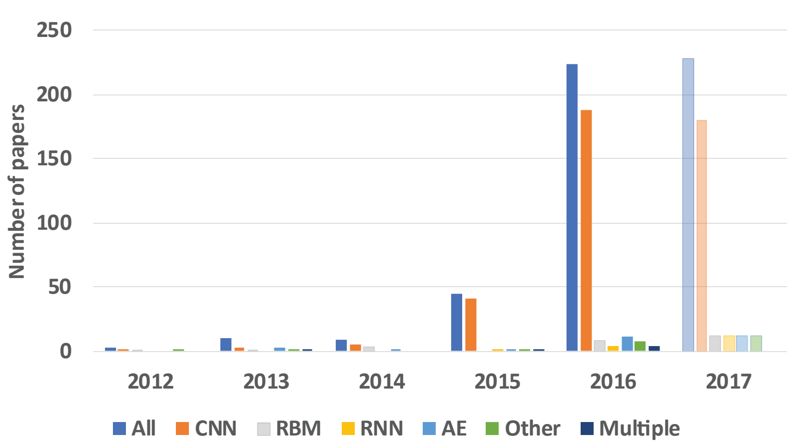


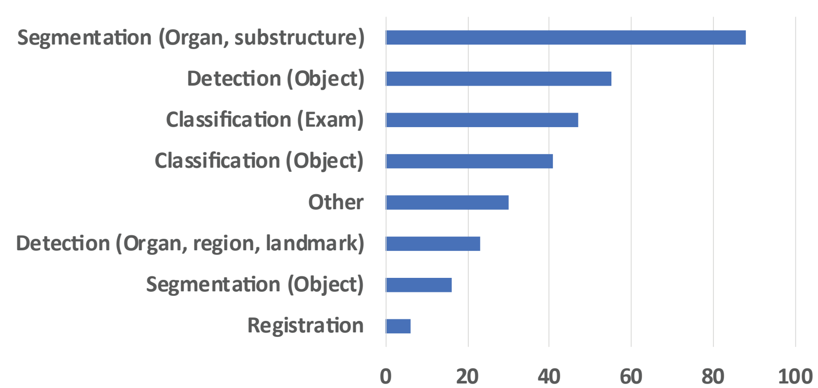


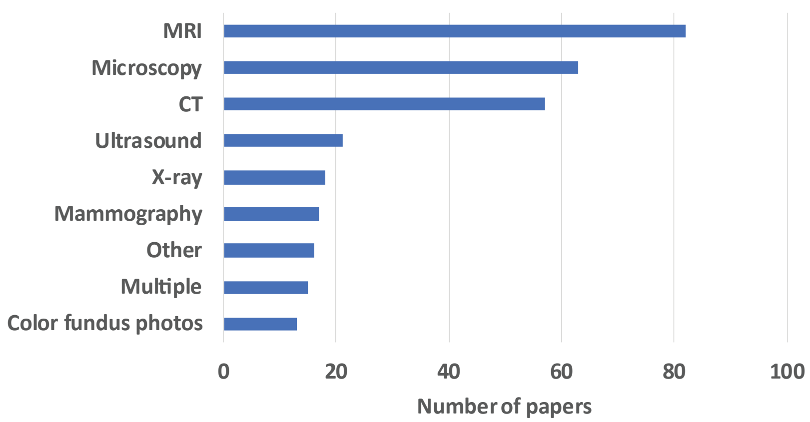
A Survey on Deep Learning in Medical Image Analysis

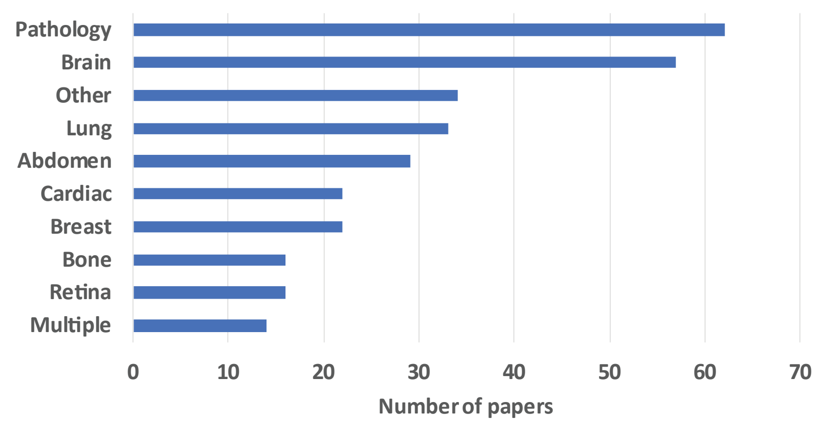
Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi,
Mohsen Ghafoorian, Jeroen A.W.M. van der Laak, Bram van Ginneken, Clara I. Sánchez

Diagnostic Image Analysis Group
Radboud University Medical Center
Nijmegen, The Netherlands









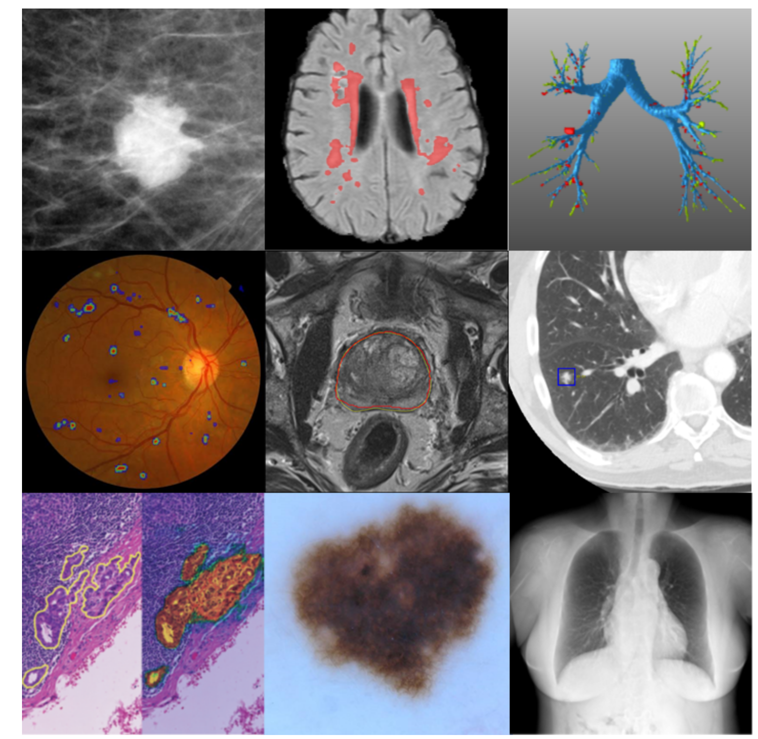


Figure 3: Collage of some medical imaging applications in which deep learning has achieved state-of-the-art results. From top-left to bottom-right: mammographic mass classification (Kooi et al., 2016), segmentation of lesions in the brain (top ranking in BRATS, ISLES and MRBrains challenges, image from Ghafoorian et al. (2016b), leak detection in airway tree segmentation (Charbonnier et al., 2017), diabetic retinopathy classification (Kaggle Diabetic Retinopathy challenge 2015, image from van Grinsven et al. (2016), prostate segmentation (top rank in PROMISE12 challenge), nodule classification (top ranking in LUNA16 challenge), breast cancer metastases detection in lymph nodes (top ranking and human expert performance in CAMELYON16), human expert performance in skin lesion classification (Esteva et al., 2017), and state-of-the-art bone suppression in x-rays, image from Yang et al. (2016c).

https://arxiv.org/pdf/1702.05747.pdf

Style Transfer

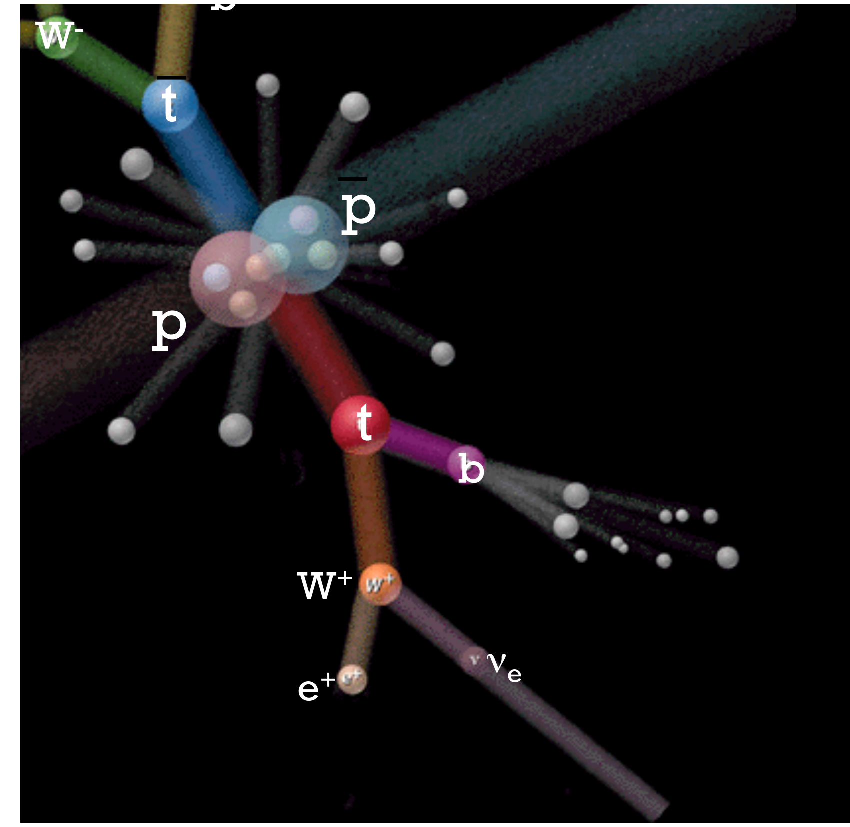
DeepFakes

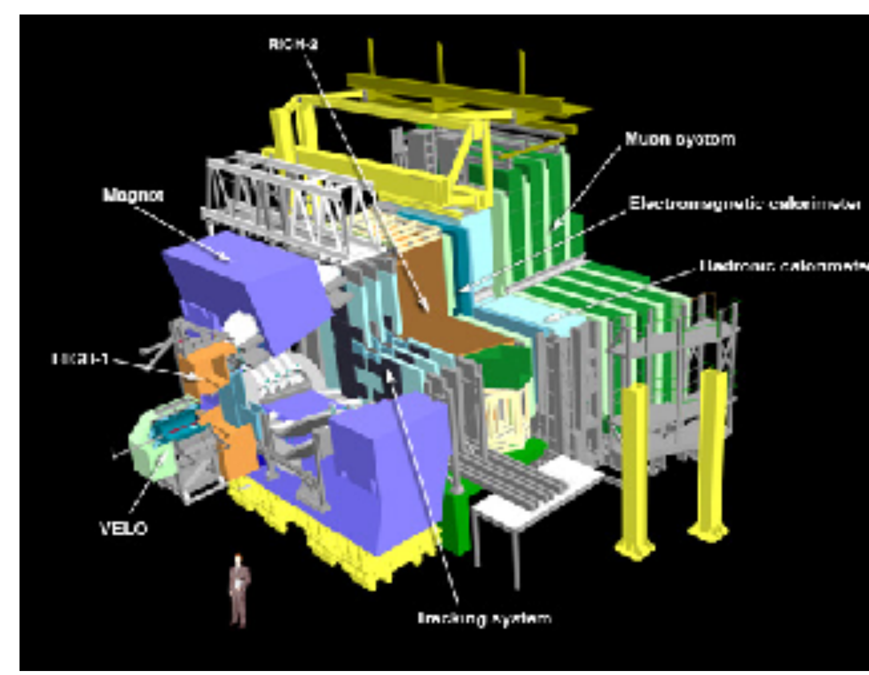
https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Gatys_Image_Style_Transfer_CVPR_2016_paper.pdf

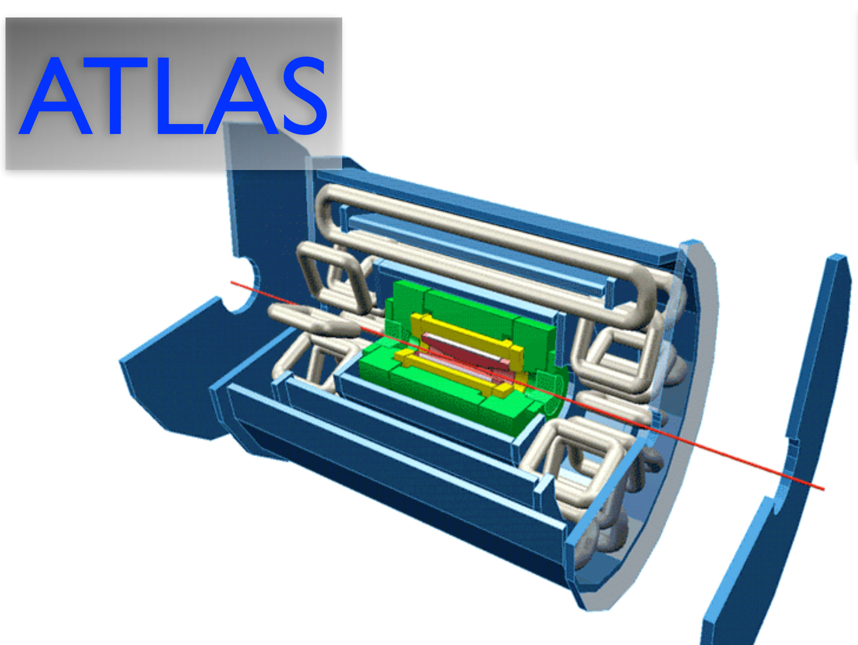
DL in HEP?

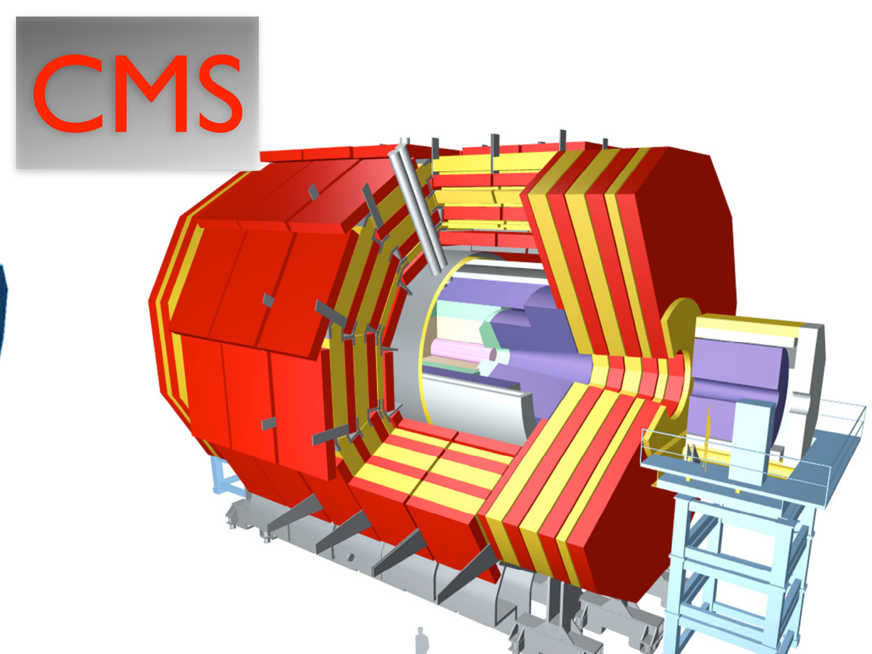
HEP Experiments

- 5 technical components to HEP experiment:
  - ***Accelerator***: e.g. LHC collisions creating quickly decaying heavy particles. Extremely high rate: 40 * O(50) Million collisions/sec.
  - ***Detector***: a big camera. ~ e.g. LHC 1.5 MB/event (60 TB/s)
    - Pictures of long-lived decay products of short lived heavy/ interesting particles.
    - Sub-detectors parts: Tracking, Calorimeters, Muon system, Particle ID (e.g. Cherenkov, Time of Flight)
  - ***DAQ/Trigger:*** Hardware/software
  - ***Simulation:*** Integral to design, SW development, analysis, …
  - ***Software:*** Reconstruction (Raw data -> particle “features”) / Analysis
  - ***Computing:*** GRID Monarch Model “Cloud” Computing/Data Management (software/hardware)









Why go Deep?

- ***Better Algorithms***
  - DNN-based classification/regression generally **out perform** hand crafted algorithms.
  - In some cases, it may provide a ***solution*** where ***algorithm approach doesn't exist or fails***.
  - ***Unsupervised learning***: make sense of complicated data that we don't understand or expect.
- ***Easier Algorithm Development***: ***Feature Learning*** instead of *Feature Engineering*
  - Reduce time physicists spend writing developing algorithms, ***saving time and cost***. (e.g. ATLAS > $250M spent software)
  - Quickly perform performance ***optimization*** or ***systematic studies***.
- ***Faster Algorithms***
  - After training, DNN inference is often *faster* than sophisticated algorithmic approach.
  - DNN can ***encapsulate expensive computations***, e.g. Matrix Element Method.
  - ***Generative Models*** enable fast simulations.
  - ***Already parallelized*** and optimized for GPUs/HPCs.
  - ***Neuromorphic*** processors.

Where is ML needed?

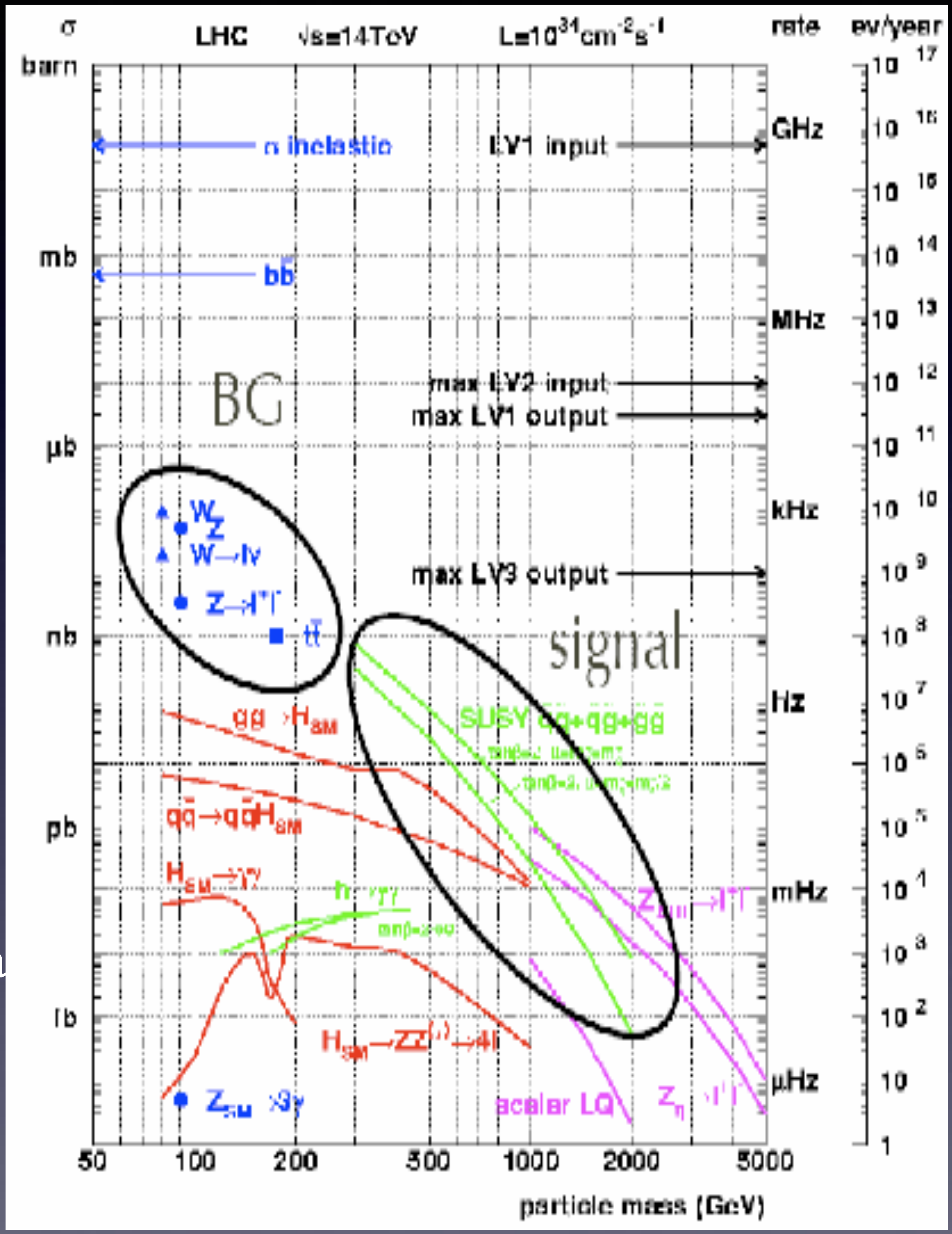
- Traditionally ML Techniques in HEP
    - Applied to Particle/Object Identification
    - Signal/Background separation
    - Here, ML maximizes reach of existing data/detector… equivalent to additional integral luminosity.
    - There is lots of interesting work here… and potential for big impact.
- Now we hope ML can help address looming computing problems of the next decade:
    - ***Reconstruction***
        1. Intensity Frontier- ***LArTPC*** Automatic Algorithmic ***Reconstruction*** still struggling
        2. Energy Frontier- ***HL-LHC Tracking***- Pattern Recognition blows up due to combinatorics
    - ***Simulation***
        3. LHC Calorimetry- Large Fraction of ATLAS CPU goes into ***shower simulation***.

Reconstruction

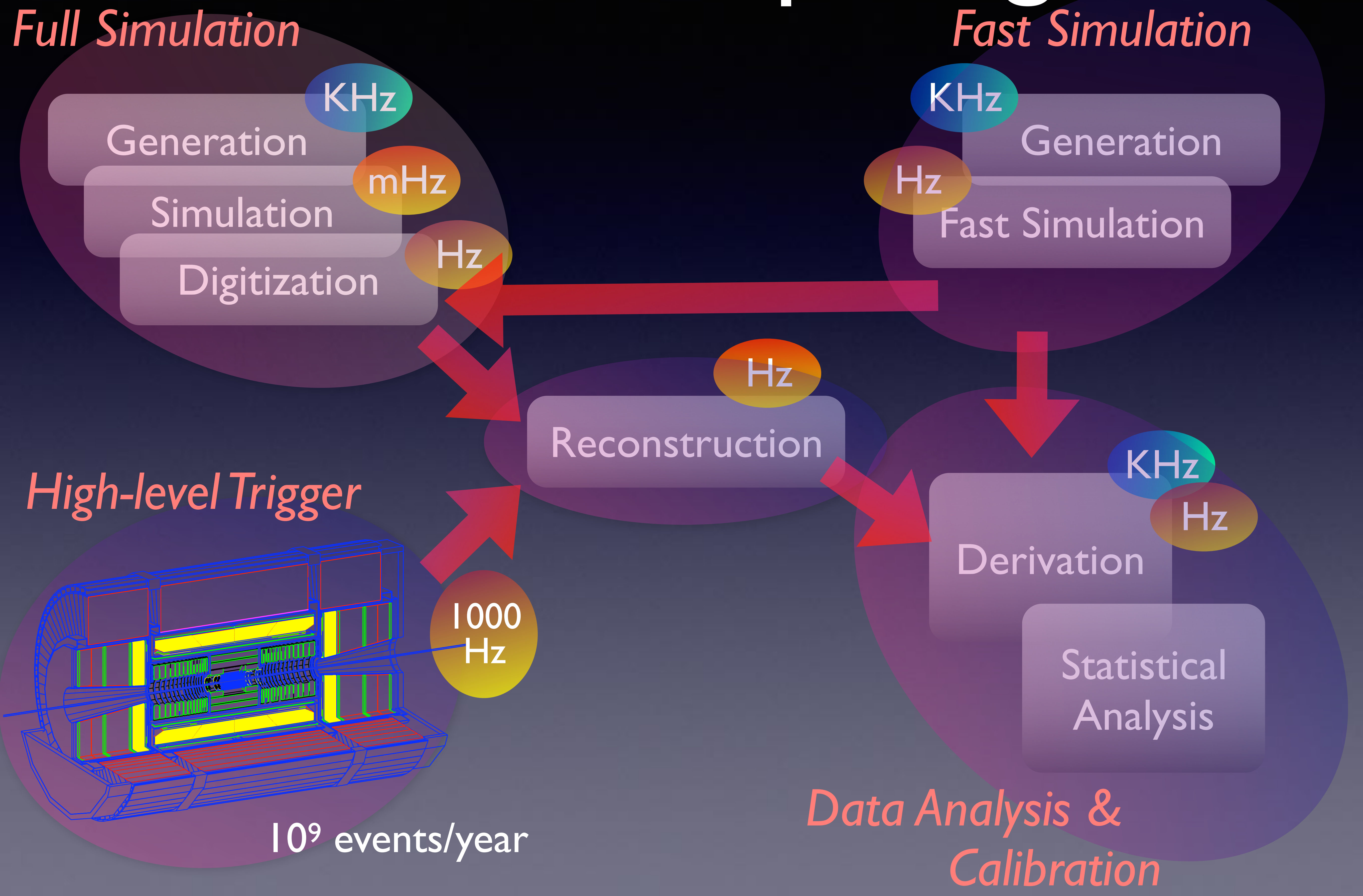
LHC Computing

Trigger

- Back of the envelope:
  - 100M Electronic Channels
  - 40 million collisions / sec at 1.5 MB/Event = 60 TB/sec.
  - Requires 2.5 m diameter bundle of Fibers to read off detector. (90's Tech, so 1 Gb/s)
- Fortunately interesting physics happens ~ 1 in 10^{11}
- Trigger system (input 40 MHz):
  - look for unique features of "interesting" events
  - analogue hardware determines if we should read data off of detector (@ 100 KHz)
  - Computing farm further reduces to 1 KHz (Run 2)
- ATLAS/CMS collect 10 PB/month, each. (?)
- High Luminosity LHC will have much busier events



HEP Computing



The Computing Model

- Resources Spread Around the GRID

- Interactive Analysis
- Plots, Fits, Toy MC, Studies, ...

Tier 3

- Reprocessing of full data with improved calibrations 2 months after data taking.
- Managed Tape Access: RAW, ESD
- Disk Access: AOD, fraction of ESD

DPD

Tier 2

AOD

- Derive 1st pass calibrations within 24 hours.
- Reconstruct rest of the data keeping up with data taking.

Tier 1

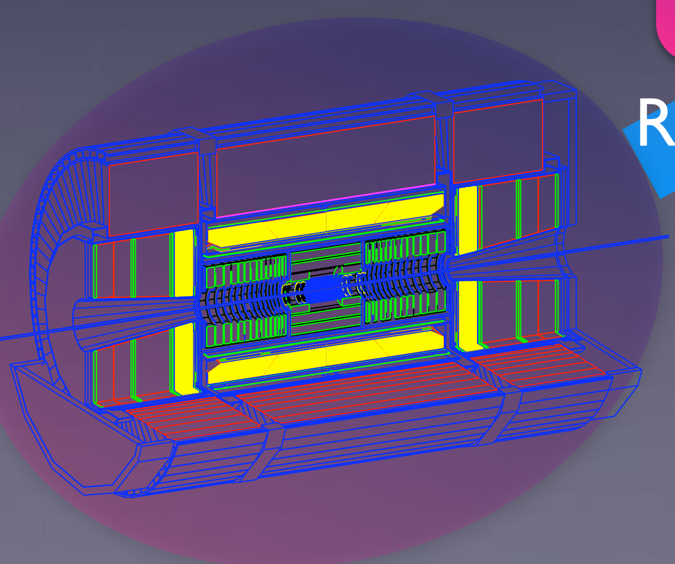
RAW/ AOD/ ESD

Tier 0

- Production of simulated events.
- User Analysis: 12 CPU/ Analyzer
- Disk Store: AOD

RAW

O(300) Sites Worldwide

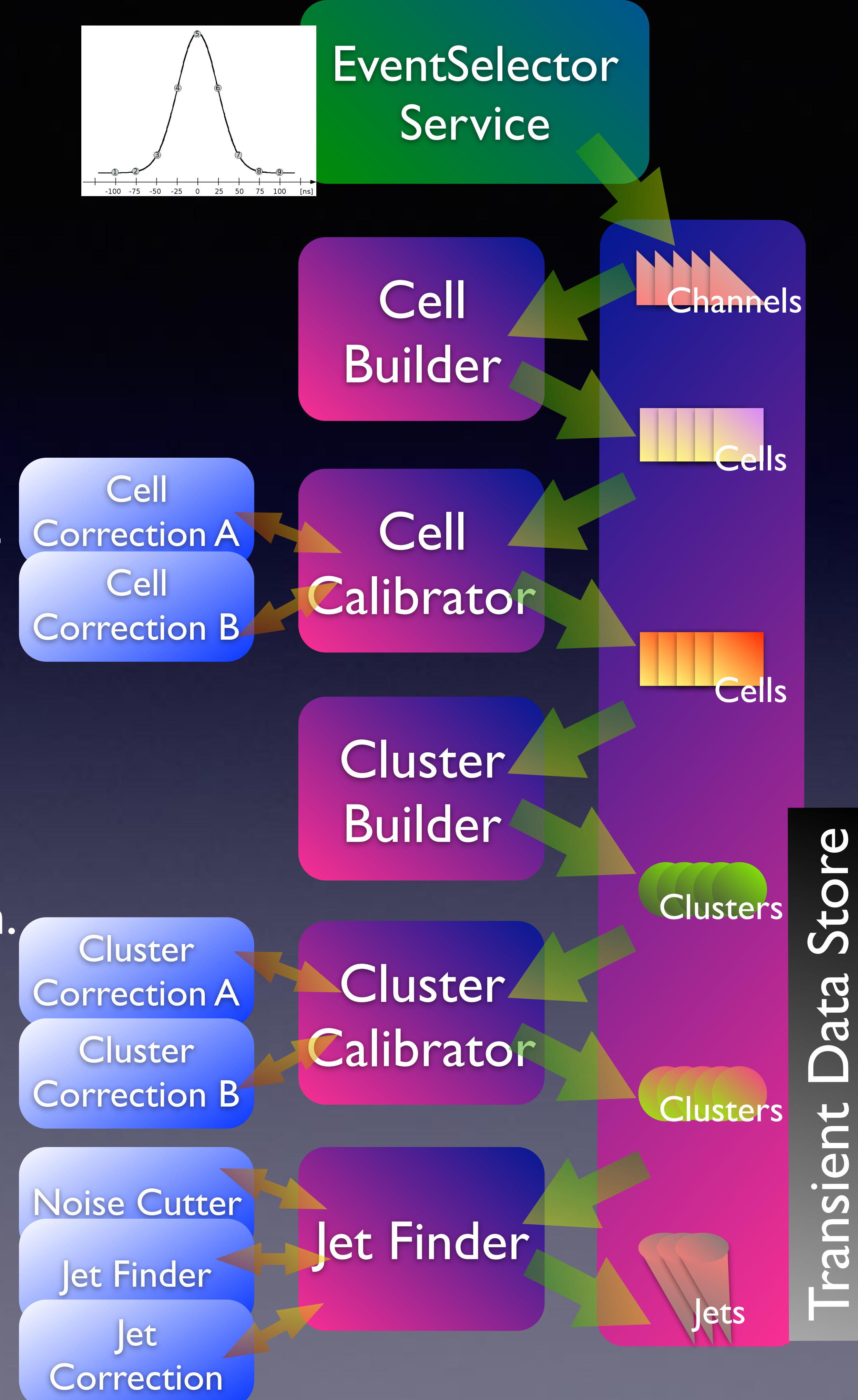


CERN Analysis Facility

- Primary purpose: calibrations
- Small subset of collaboration will have access to full ESD.
- Limited Access to RAW Data.

Reconstruction

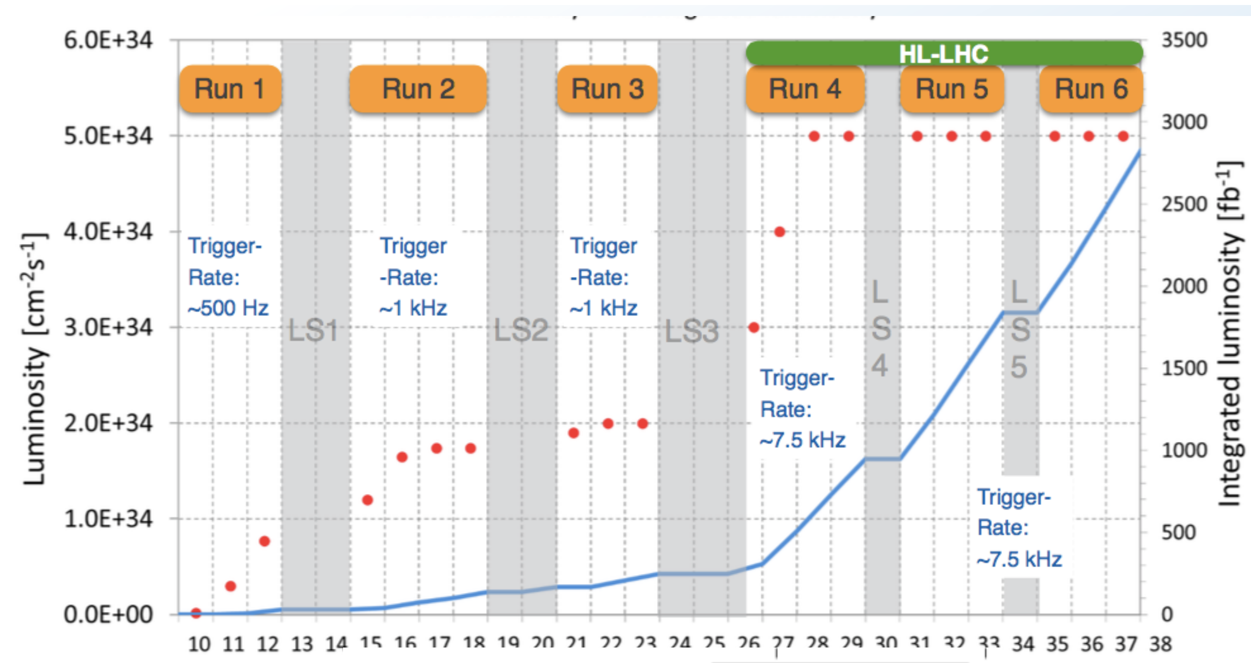
- Starts with ***raw inputs*** (e.g. Voltages)
- Low level ***Feature Extraction***: e,g, Energy/Time in each Calo Cell
- ***Pattern Recognition***: Cluster adjacent cells. Find hit pattern.
- ***Fitting***: Fit tracks to hits.
- ***Combined reco***: e.g.:
  - Matching Track+EM Cluster = Electron.
  - Matching Track in inter detector + muon system = Muon
- ***Output particle candidates*** and measurements of their properties (e.g. energy)

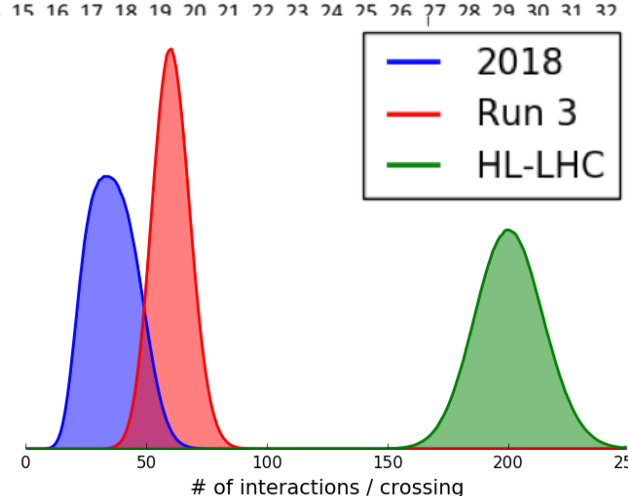


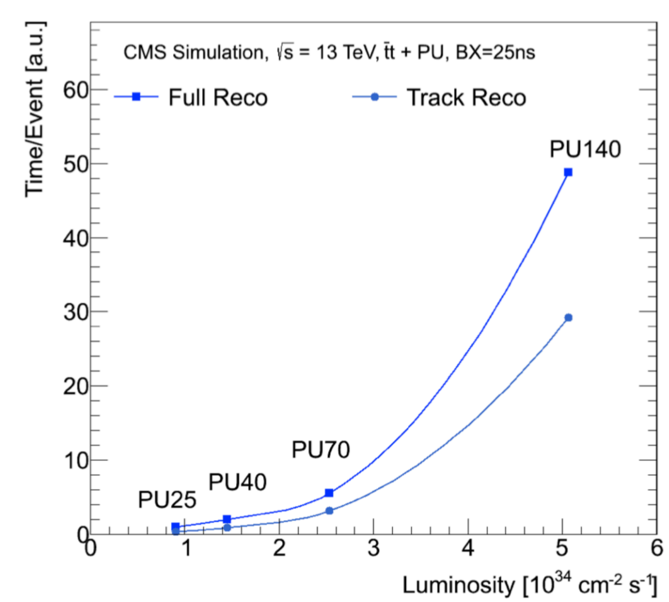
LHC → HL-HLC

HL-LHC

- ***Higher Granularity + High Trigger Rates***
  - ~10x higher input rates.
  - Trigger Needs:
    - Better Calorimetry
    - Tracking
- ***Low New Physics x-sections***, need:
  - Detail Physics: NLO / NNLO
  - Faithful Simulation: Geant
- ***High Pileup***: O(200) proton collision / crossing
  - Tracking Pattern Recognition



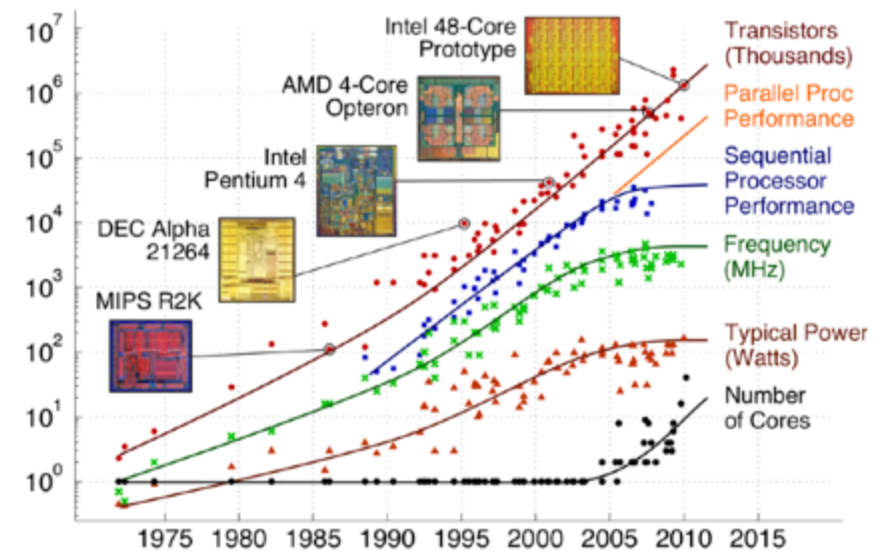


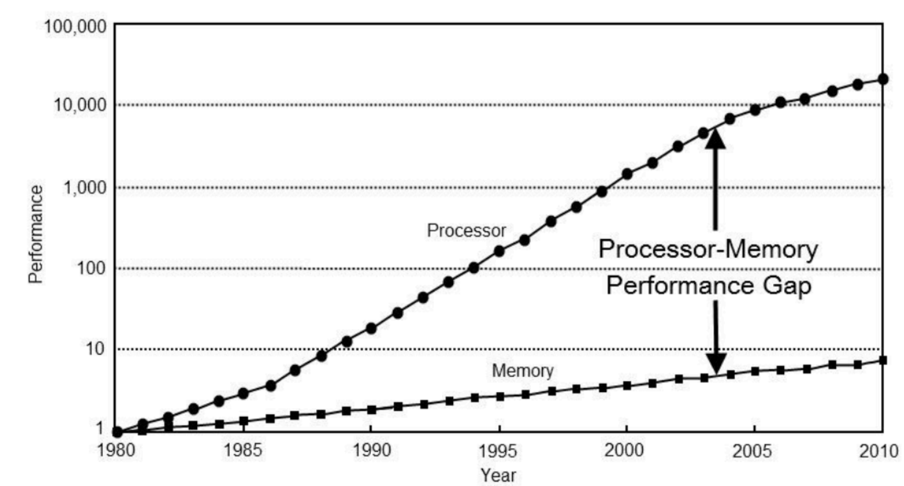


https://indico.cern.ch/event/681549/contributions/2930935/attachments/1664434/2667677/lhcp_lange_2018.pdf

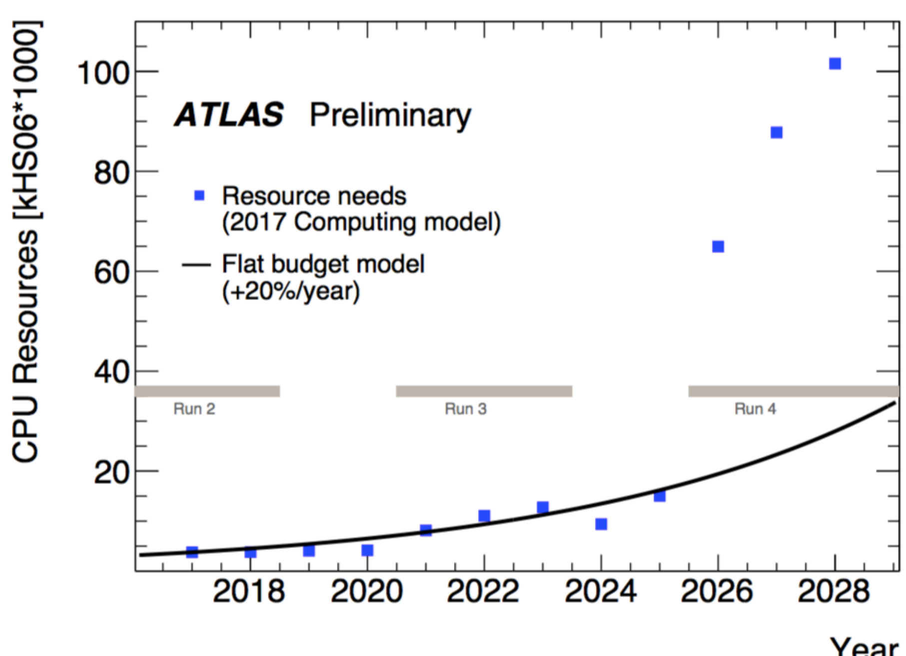
Computing

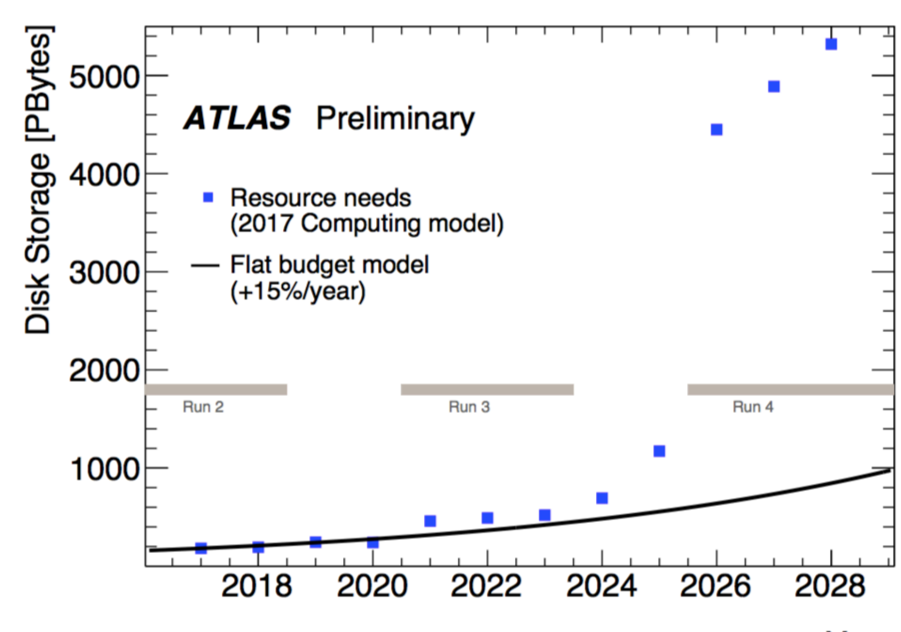
- ***HEP Reco*** is ***Embarrassingly parallel problem*** → Single threaded and memory-heavy software
    - Past few decades: scaling via ever faster / denser commodity linux boxes
- ***Moore's law has stalled***:
    - Cost of adding more transistors/silicon area no longer decreasing.
    - Trend towards more cores and slower memory access.
    - Co-processors: MiC, GPUs, FPGA, …
- Storage Scaling also a problem…
- ***HL-LHC computing budget many times larger than LHC***.





Processor-Memory speed evolution

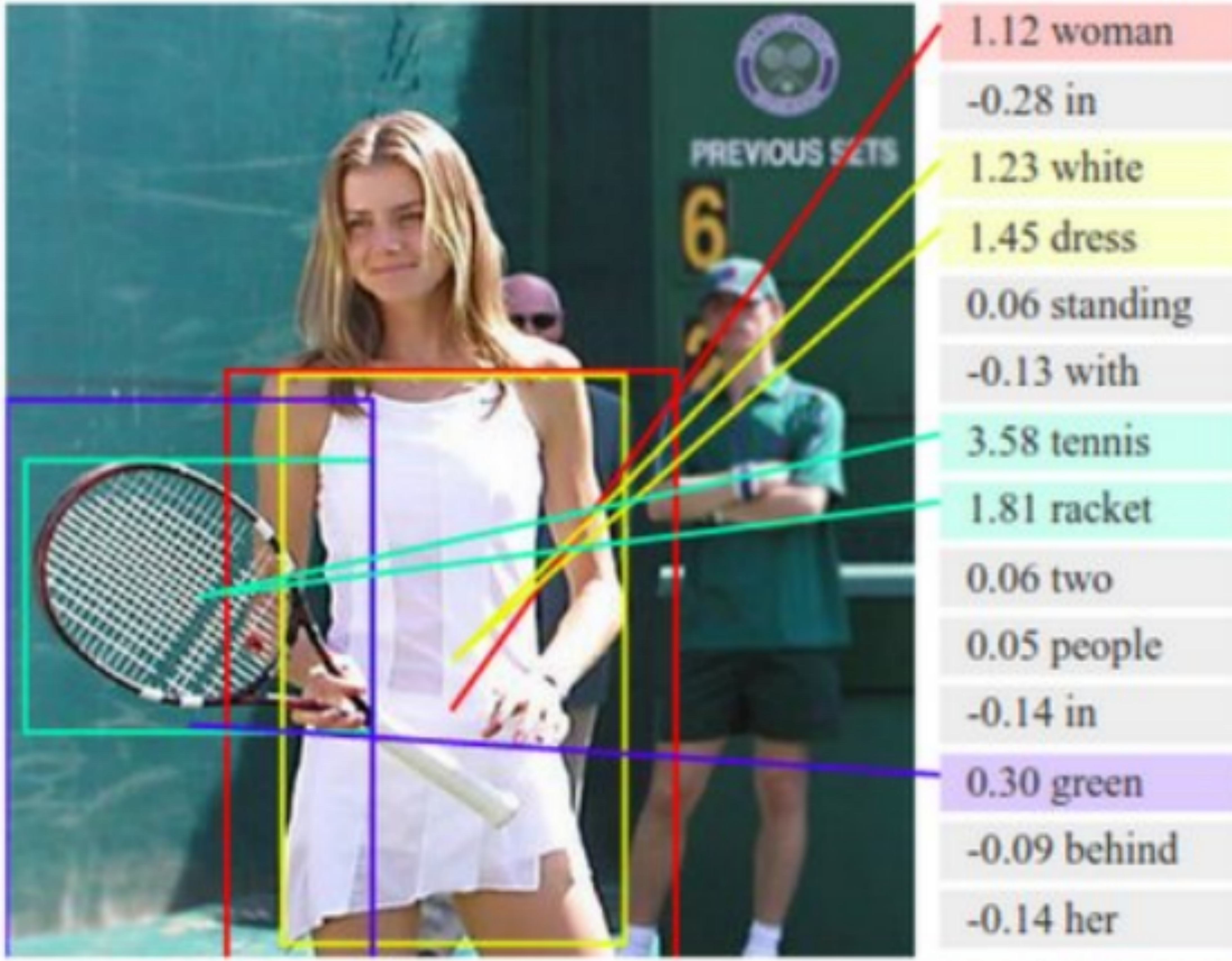




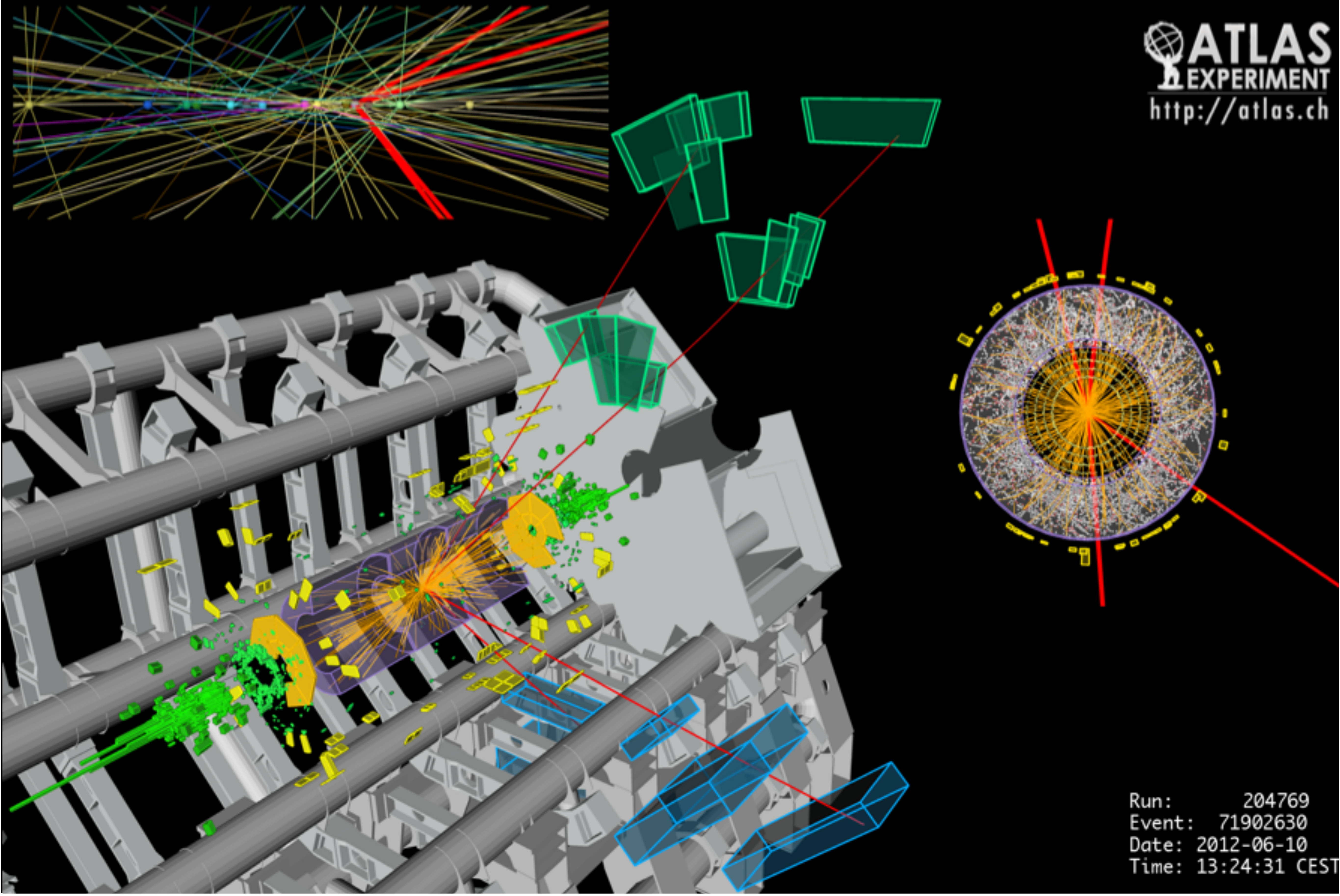
Solutions

- Framework Evolution: Multiple Events in Flight → Multiple Algs on same event → Parallelism within Algs → Offload to accelerators
- Optimizing Data and Algs for Parallelization:
  - Multi-thread: Really about memory
  - Vectorization
  - Co-processor: Significant rewriting of algorithms
- Great deal of work to evolve current frameworks to address these issues.
  - General feeling that restarting from scratch is not feasible.
- Real time analysis
- Machine Learning…

DL for LHC Reco


1.12 woman
-0.28 in
1.23 white
1.45 dress
0.06 standing
-0.13 with
3.58 tennis
1.81 racket
0.06 two
0.05 people
-0.14 in
0.30 green
-0.09 behind
-0.14 her

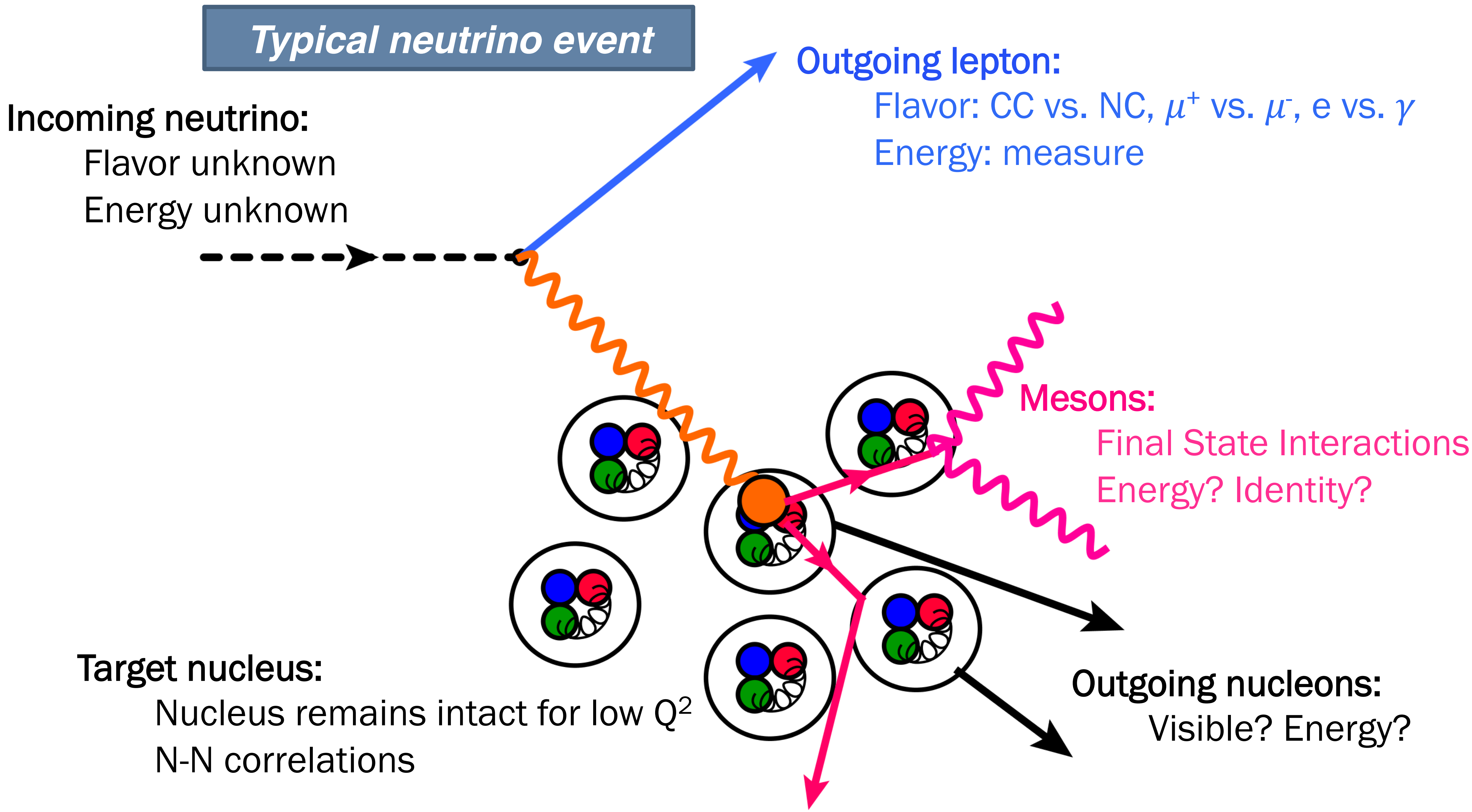
$$H \to ZZ \to 4l$$


ATLAS
EXPERIMENT
http://atlas.ch
Run: 204769
Event: 71902630
Date: 2012-06-10
Time: 13:24:31 CEST

Neutrinos…

Neutrino Detection

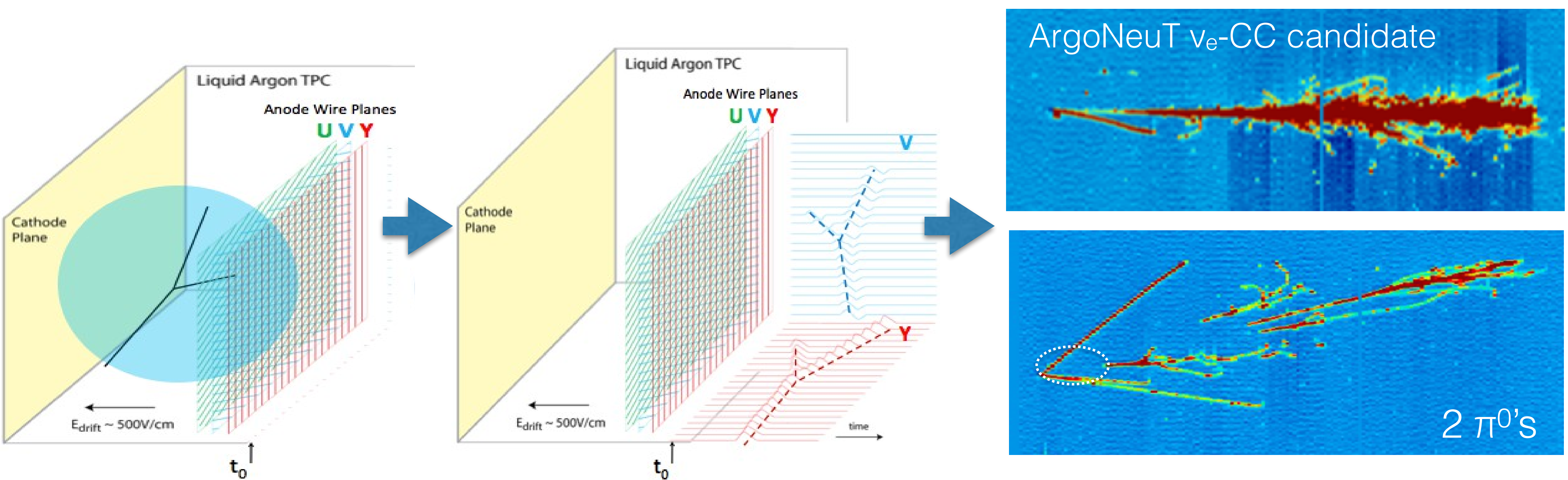
In neutrino experiments, try to determine flavor and estimate energy of incoming neutrino by looking at outgoing products of the interaction.



Jen Raaf

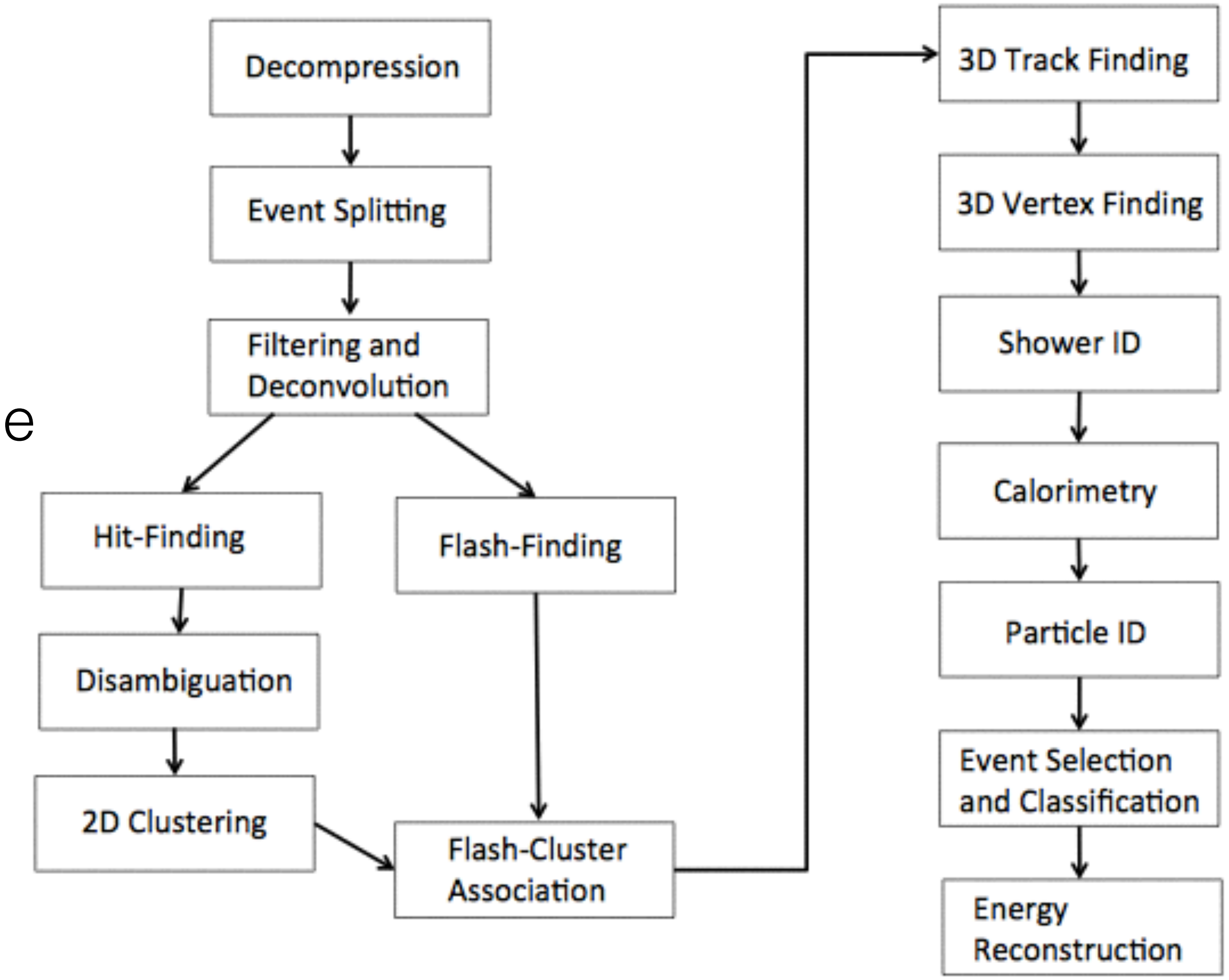
Neutrino Detectors

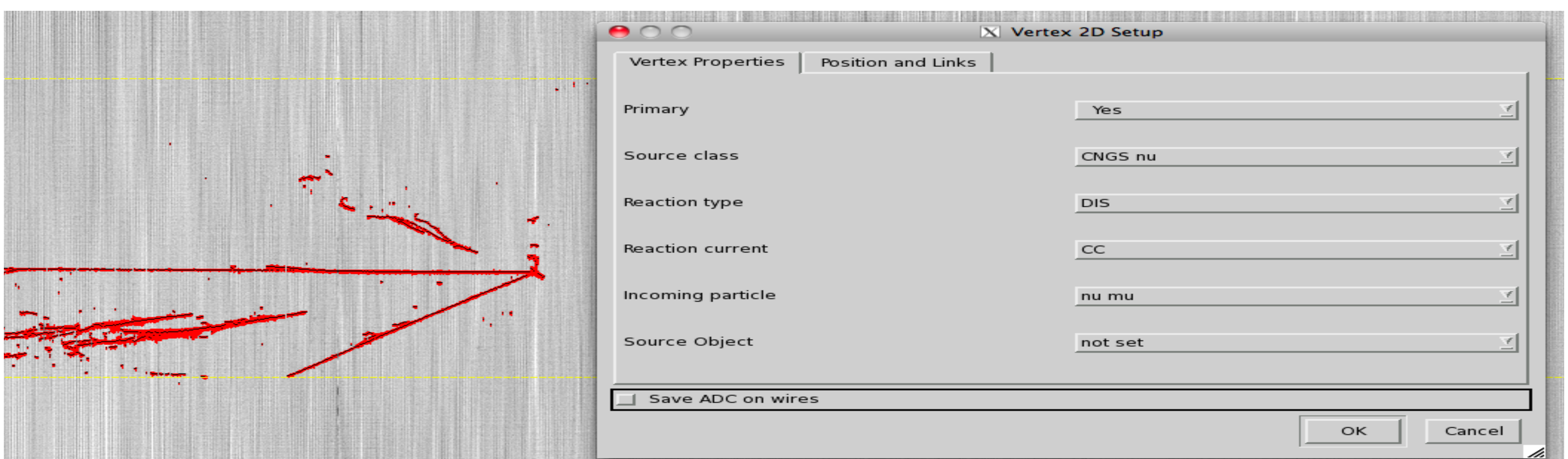
- ***Need large mass/volume*** to maximize chance of neutrino interaction.
- Technologies:
  - Water/Oil Cherenkov
  - Segmented Scintillators
  - ***Liquid Argon Time Projection Chamber: promises ~ 2x detection efficiency.***
    - ***Provides tracking, calorimetry, and ID all in same detector***.
    - Chosen technology for US's flagship LBNF/DUNE program.
    - Usually 2D read-out… 3D inferred.
- After many years of trying, ***good automatic reconstruction still not demonstrated.***



LArTPC Reco

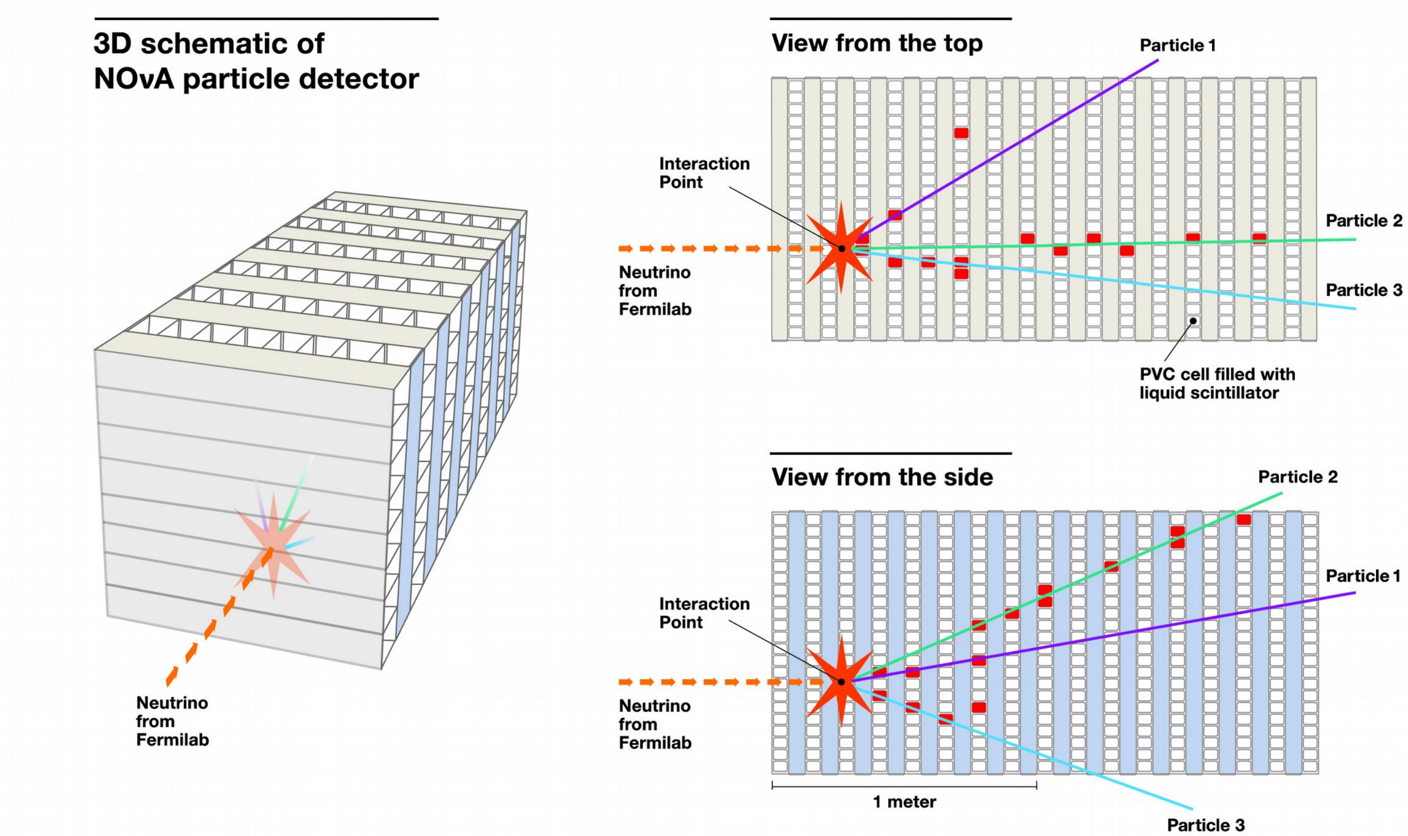
- Neutrino Physics has a long history of ***hand scans***.
    - *QScan*: ICARUS user assisted reconstruction.
    - ***Full automatic reconstruction*** has yet to be demonstrated.
    - LArSoft project: art framework + LArTPC reconstruction algorithms developed by LArIAT, MicroBooNE, DUNE, …
    - Still… full neutrino reconstruction is still far from expected performance.





Neutrino Physics

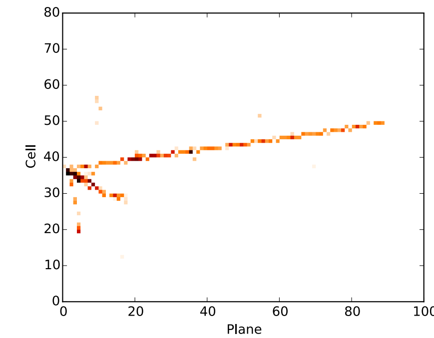
- Core Physics requires just measuring ***neutrino flavor and energy***.
- Generally clean (low multiplicity) and high granularity.
- ***First HEP CNN application: <u>Nova</u>*** using Siamese Inception CNN.



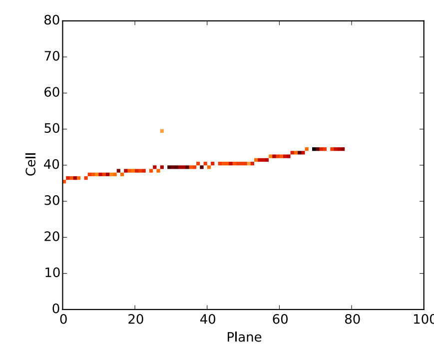
	CVN Selection Value	ν_e sig	Tot bkg	NC	ν_μ CC	Beam ν_e	Signal Efficiency	Purity
Contained Events	–	88.4	509.0	344.8	132.1	32.1	–	14.8%
$s/\sqrt{b}$ opt	0.94	43.4	6.7	2.1	0.4	4.3	49.1%	86.6%
$s/\sqrt{s+b}$ opt	0.72	58.8	18.6	10.3	2.1	6.1	66.4%	76.0%

	CVN Selection Value	ν_μ sig	Tot bkg	NC	Appeared ν_e	Beam ν_e	Signal Efficiency	Purity
Contained Events	–	355.5	1269.8	1099.7	135.7	34.4	–	21.9%
$s/\sqrt{b}$ opt	0.99	61.8	0.1	0.1	0.0	0.0	17.4%	99.9%
$s/\sqrt{s+b}$ opt	0.45	206.8	7.6	6.8	0.7	0.1	58.2%	96.4%

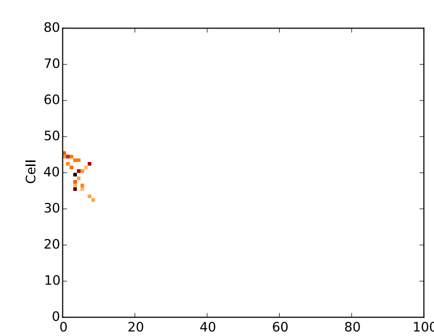
- ***40% Better Electron Efficiency for same background.***



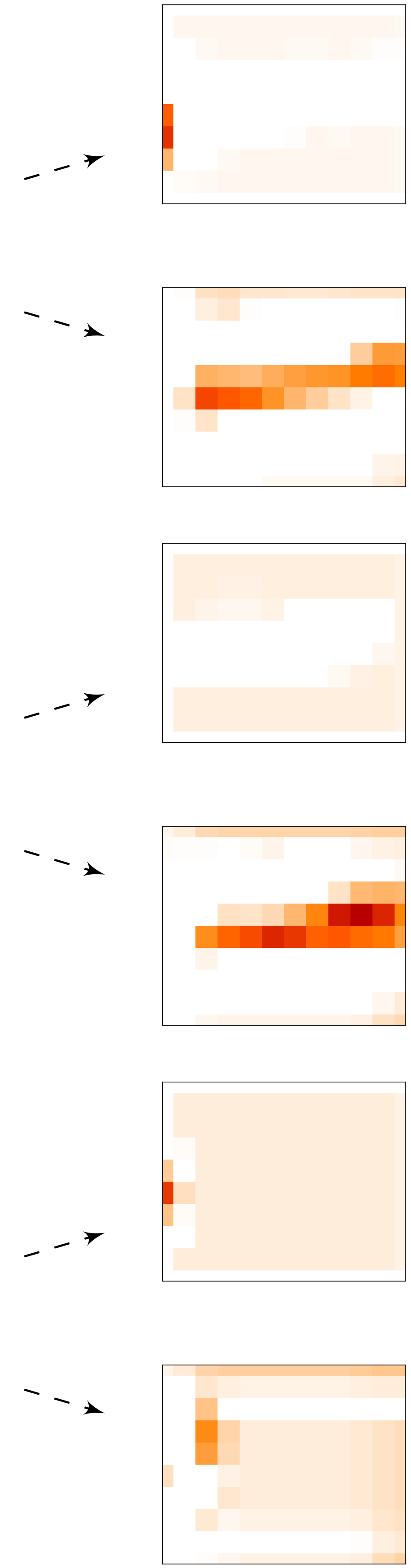
Muon Neutrino
DIS CC



Muon Neutrino
QE CC



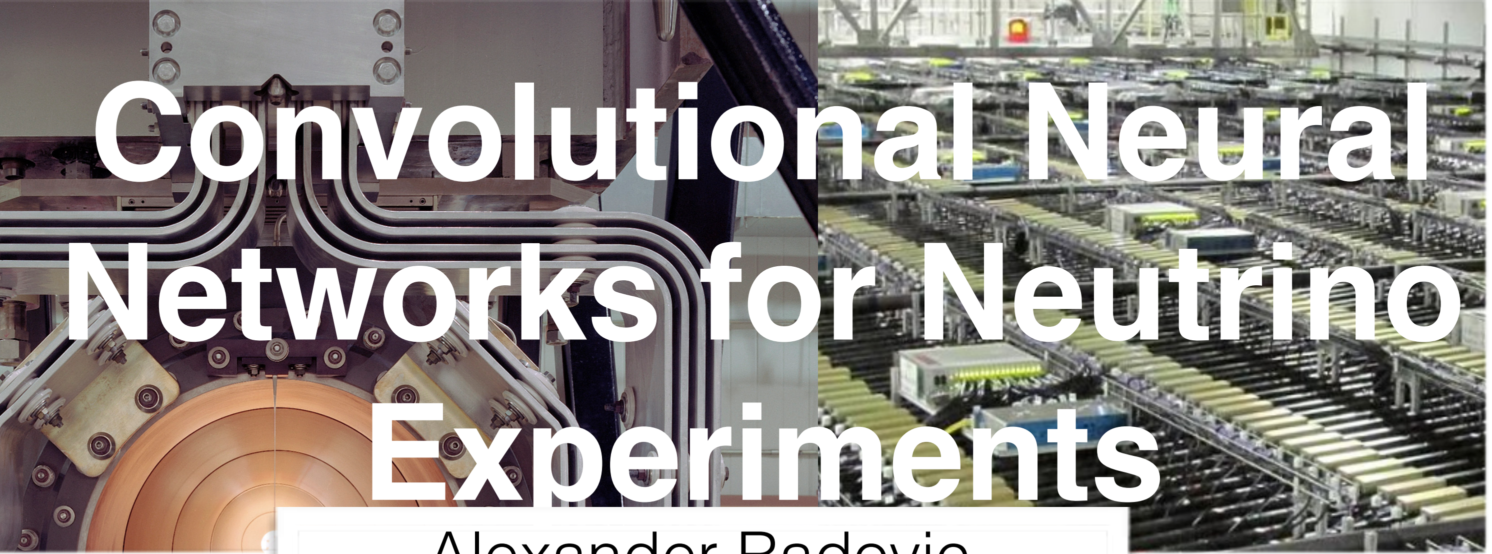
Muon Neutrino
NC



Hadronic Feature Map

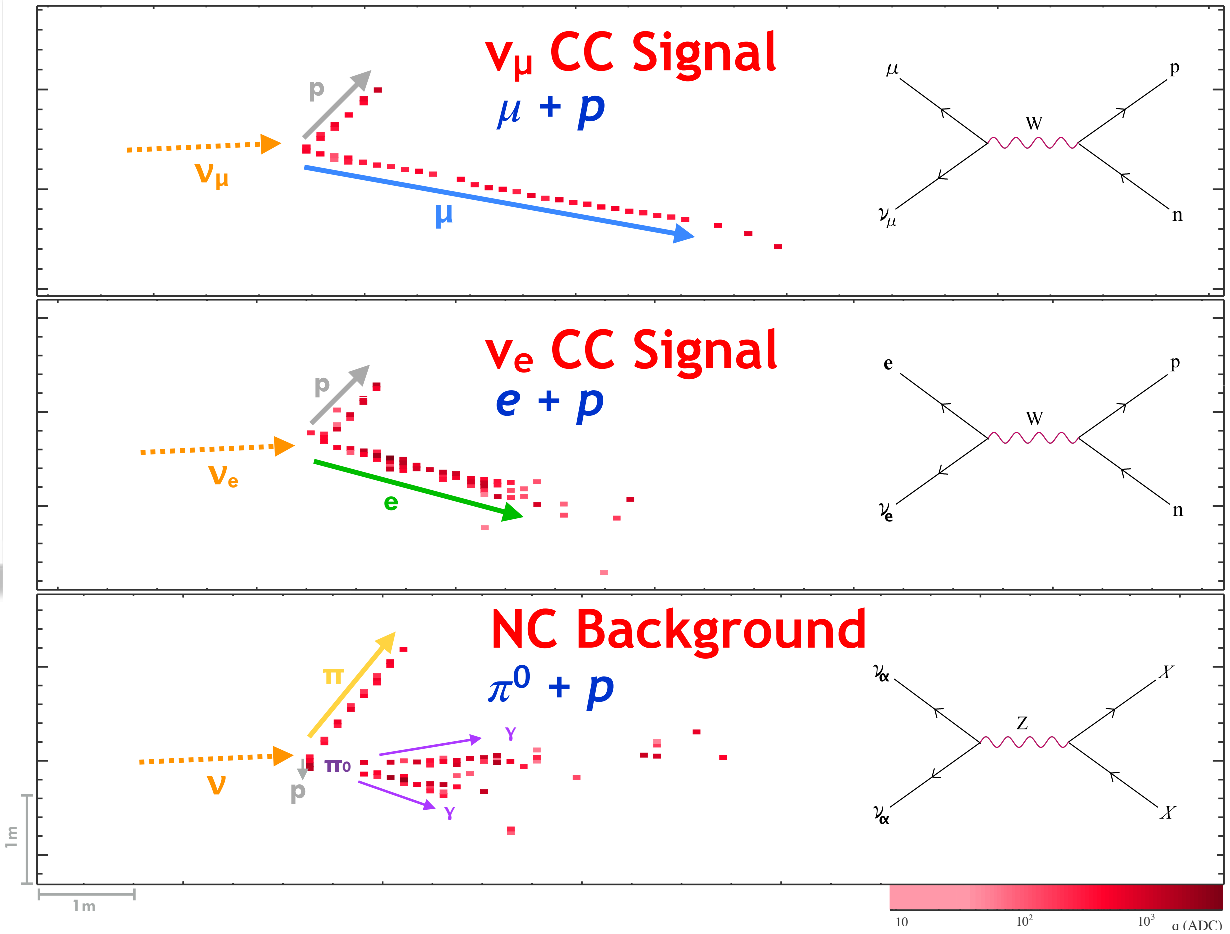
Muon Feature Map

http://arxiv.org/pdf/1604.01444.pdf



Convolutional Neural Networks for Neutrino Experiments

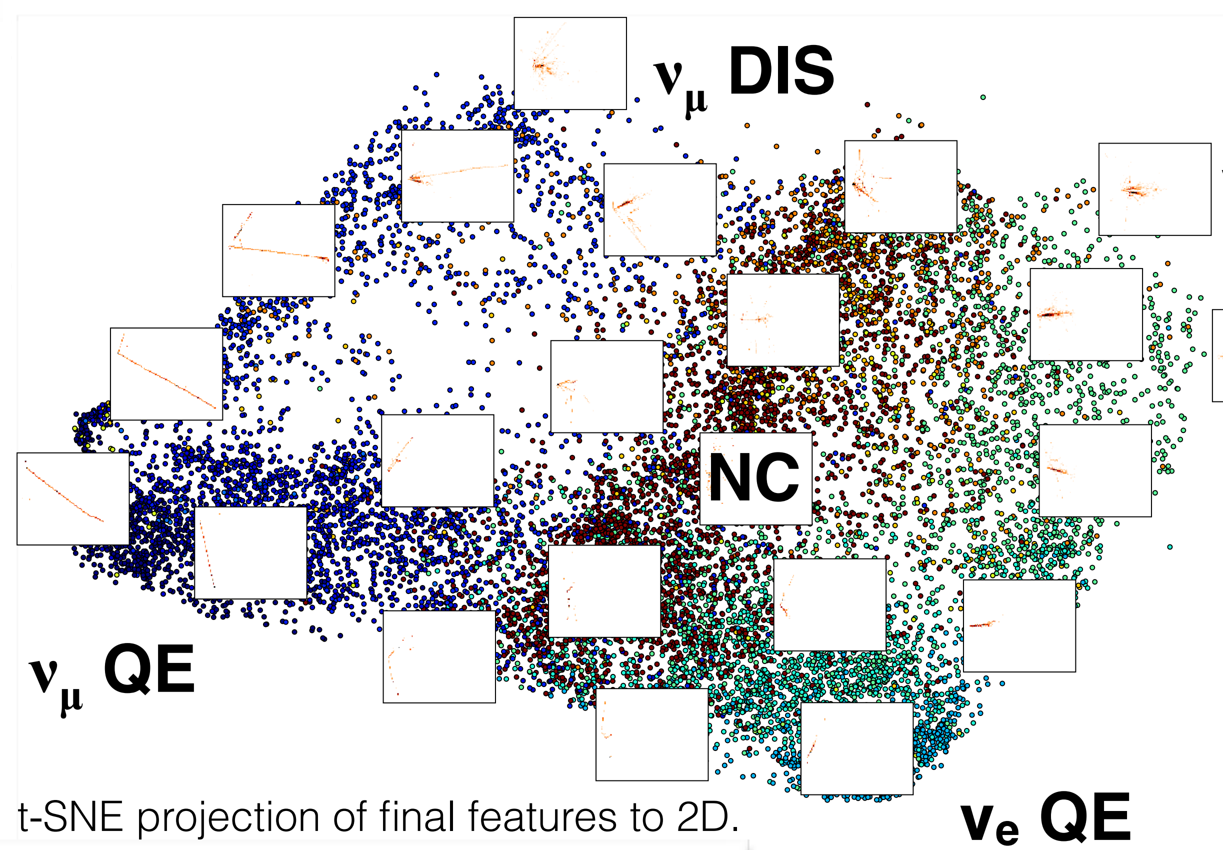
Alexander Radovic
College of William and Mary



Why Convolutional Neural Networks?

- That means that any oscillation analysis can benefit fror precise identification of the interaction in two ways:
  - Estimating the lepton flavor of the incoming neutrino.
  - Correctly identifying the type of neutrino interaction, tc better estimate the neutrino energy, aka is it a quasi elastic event or a resonance event?

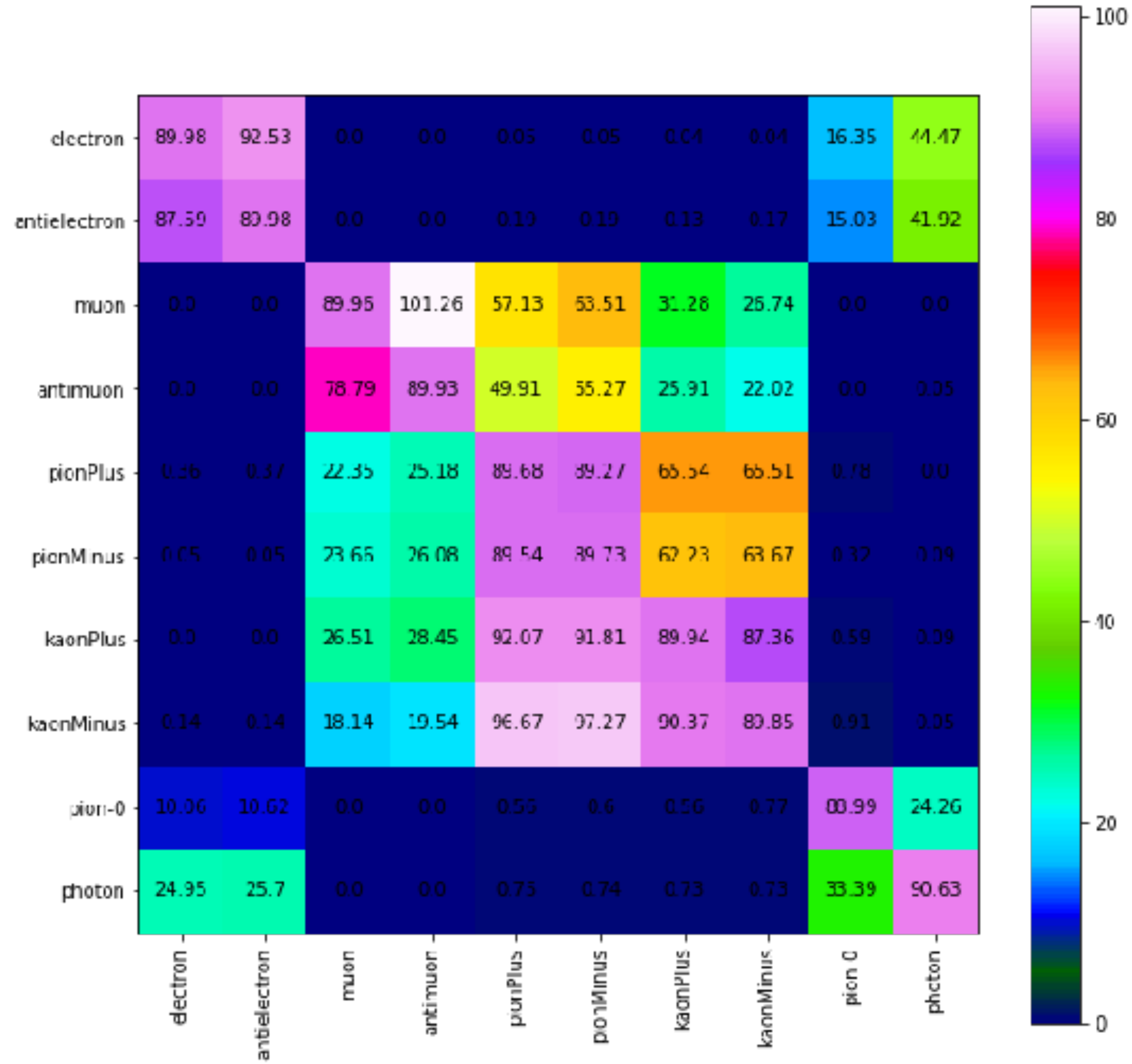
- Our detectors are also often the perfect domain:
  - Large ~uniform volumes where spatially invariant response is a benefit.
  - Usually only one or two detector systems.



t-SNE projection of final features to 2D.

However our CNN achieves **73%** efficiency and **76%** purity on ν_e selection at the $s/\sqrt{s+b}$ optimized cut.
Equivalent to **30%** more exposure with the old PIDs.

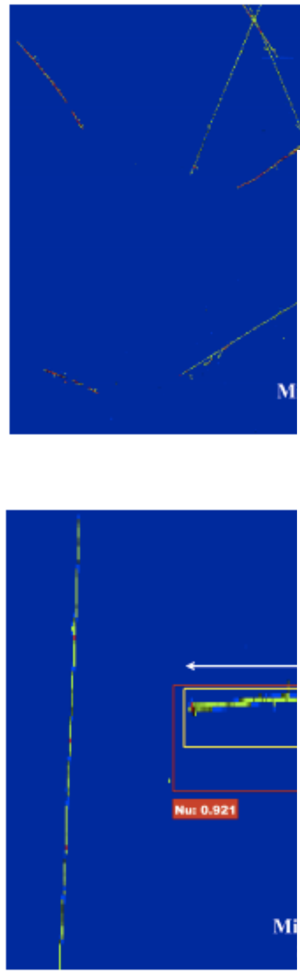
LArIAT: DNN vs Alg



	π+	κ+	μ+	e+	γ
DNN	74.42%	40.67%	6.37%	0.12%	0%
LArIAT	74.5%	68.8%	88.4%	6.8%	2.4%

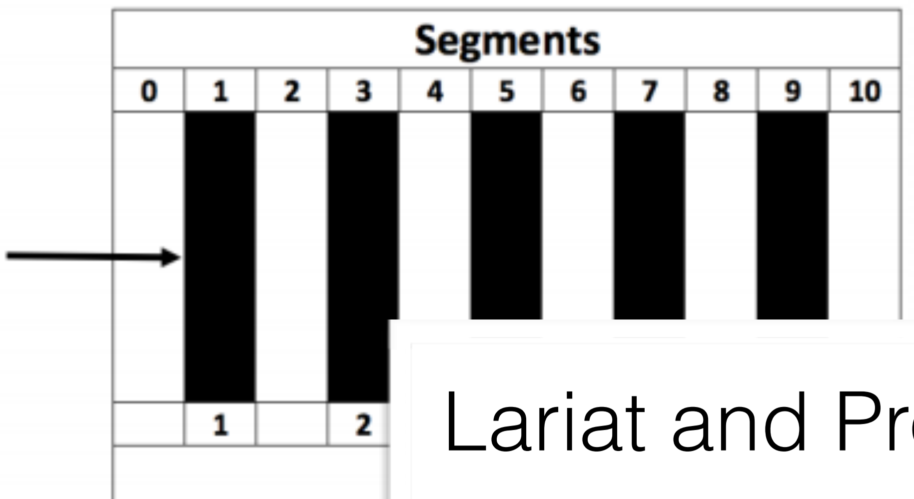
	π−	κ−	μ−	e−	γ
DNN	78.68%	54.47%	13.54%	0.11%	0.25%
LArIAT	78.7%	73.4%	91.0%	7.5%	2.4%

MicroBooNE



JINST 12 (2017) r

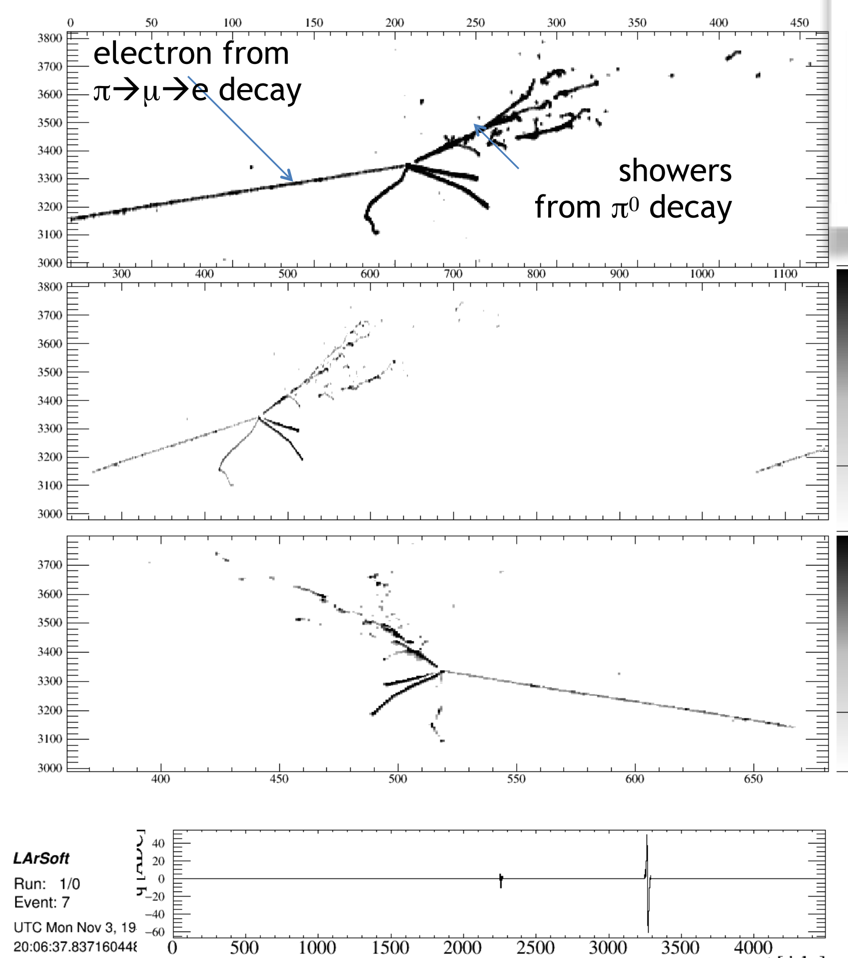
MINERvA and MENNDL



PhyStat-nu Fermila

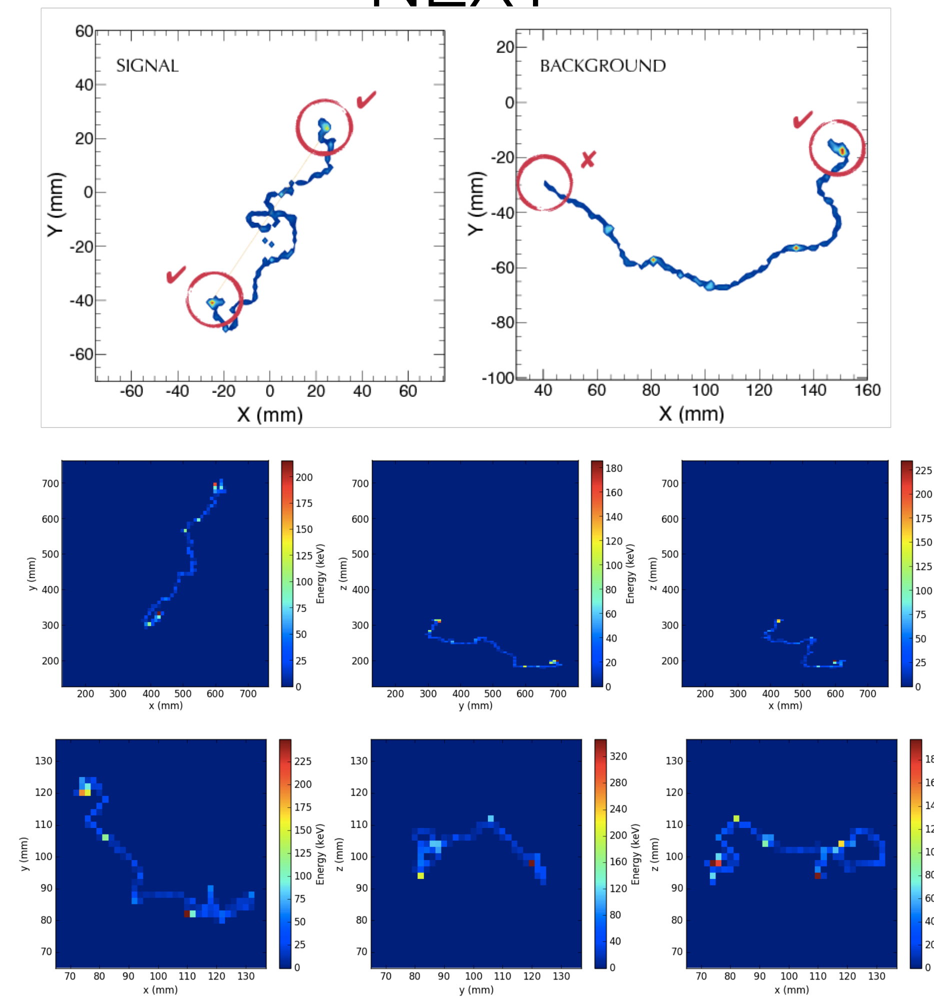
Lariat and ProtoDUNE

EM / hadronic component discrimination



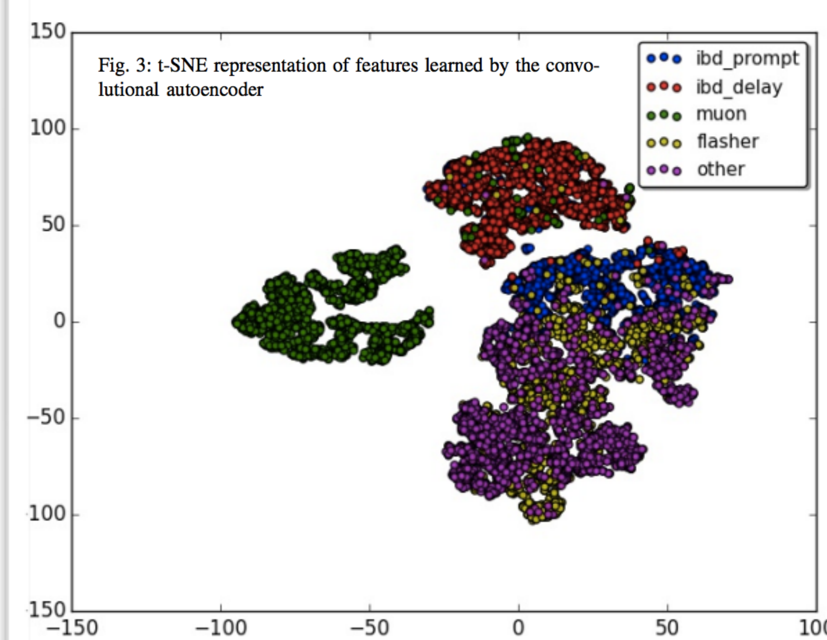
Private communication, Robert Sutlej

NEXT



JINST 12 (2017) no.01, T01004

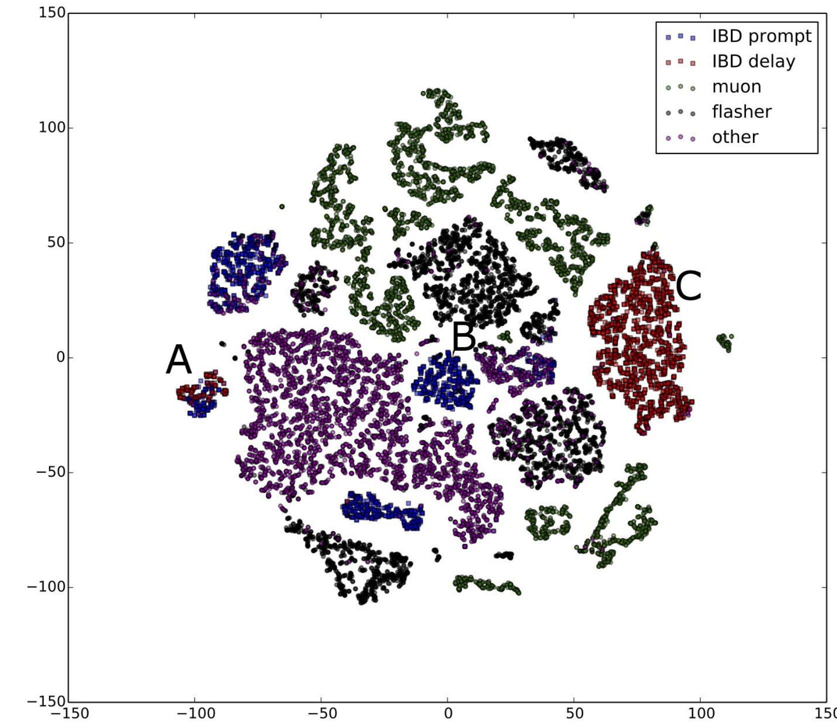
Daya Bay



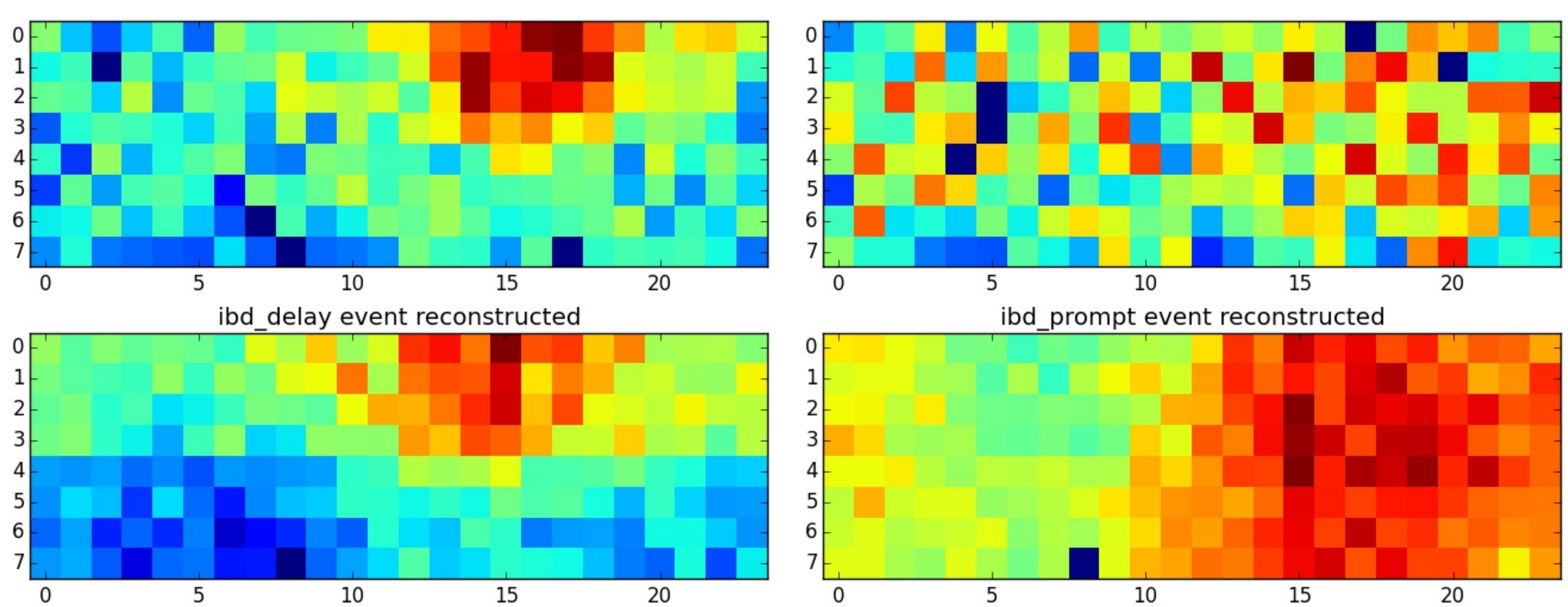
arXiv:1601.07621

Learning Representations

- Example: ***Daya Bay Experiment*** (*Evan Racah, et al*)
- Input: 8 x 24 PMT unrolled cylinder. ***Real Data (no simulation)***
- 2 Studies:
    - ***Supervised CNN Classifier***
        - Labels from standard analysis: Prompt/Delayed Inverse Beta Decay, Muon, Flasher, Other.
    - ***Convolutional Auto-encoder*** (semi-supervised)
        - Clearly separates muon and IBD delay ***without any physics knowledge***.
        - Potentially could have ID'ed problematic data (e.g. flashers) much earlier.

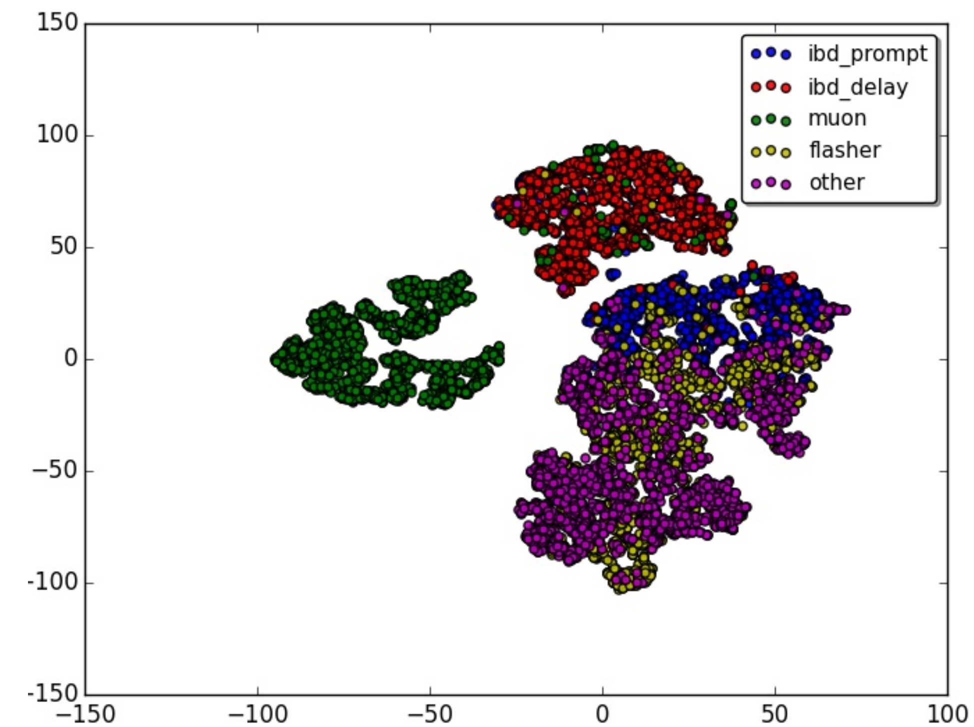


t-SNE reduction of 26-dim representation of the last fully connected layer.



(a) Example of an "IBD delay" event

(b) Example of an "IBD prompt" event



t-SNE reduction of 10 parameter latent representation.

Calorimetry with Deep Learning

Calorimetry with Deep Learning: Particle Classification, Energy Regression, and Simulation for High-Energy Physics

Federico Carminati, Gulrukh Khattak, Maurizio Pierini
CERN

Amir Farbin
Univ. of Texas Arlington

Benjamin Hooberman, Wei Wei, and Matt Zhang
Univ. of Illinois at Urbana-Champaign

Vitória Barin Pacela
Univ. of Helsinki
California Institute of Technology

Sofia Vallecorsafac
Gangneung-Wonju National Univ.

Maria Spiropulu and Jean-Roch Vlimant
California Institute of Technology

Abstract

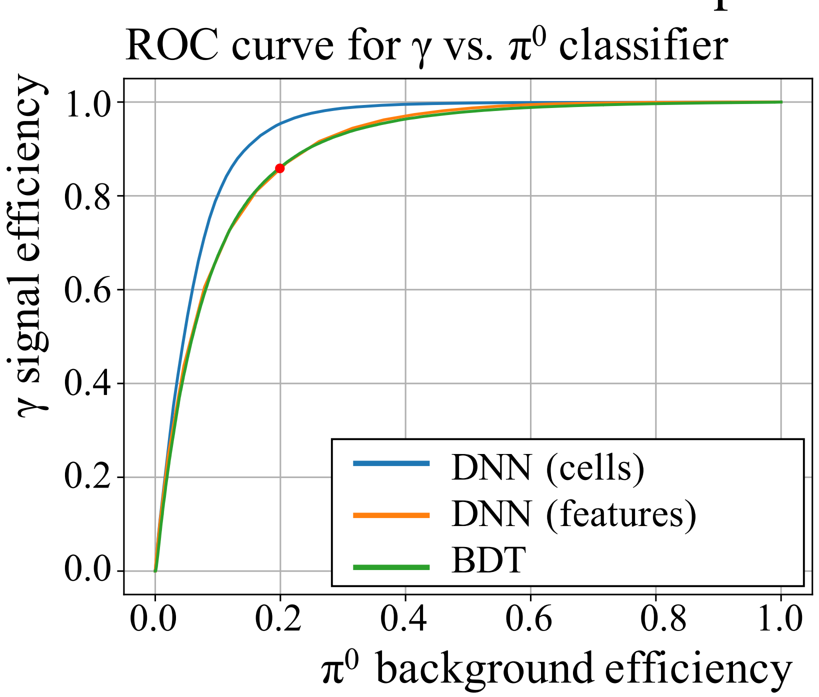
We present studies of the application of Deep Neural Networks and Convolutional Neural Networks for the classification, energy regression, and simulation of particles produced in high-energy particle collisions.We train cell-based Neural Nets that provide significant improvement in performance for particle classification and energy regression compared to feature-based Neural Nets and Boosted Decision Trees, and Generative Adversarial Networks that provide reasonable modeling of several but not all shower features.

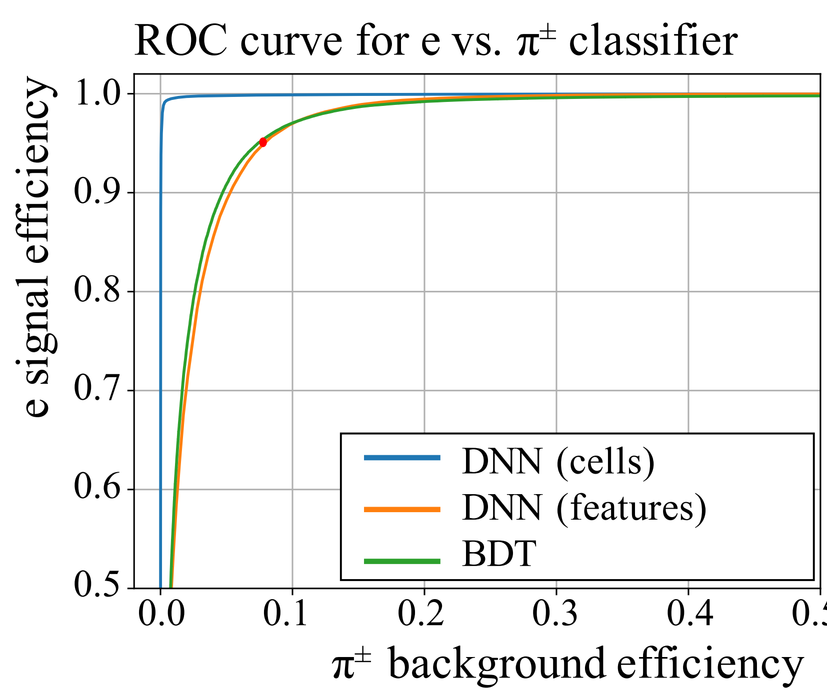
1. e/γ Particle Identification (Classification)

- Photon/lepton ID requires factor ~10000 jet rejection
- Jet like photon/lepton classification tasks:
  - *Task 1:* Electrons vs Electromagnetic $\pi^{+/-}$ (HCAL/ECAL Energy < 0.025)
  - *Task 2:* Photons vs Merging π^0 (2γ opening angel < 0.01 rad)
- *Comparison*:
  - ***Feature based*** BDT and DNN
  - ***Cell-based*** DNN (fully connected).
- Significant Improvement with cell-based DNNs.

	γ **vs.** π^0				e **vs.** π			
Model	**acc.**	**AUC**	$\Delta\epsilon_{\text{sig}}$	ΔR_{bkg}	**acc.**	**AUC**	$\Delta\epsilon_{\text{sig}}$	ΔR_{bkg}
BDT	83.1%	89.8%	-	-	93.8%	98.0%	-	-
DNN (features)	82.8%	90.2%	0.9%	0.95	93.6%	98.0%	-0.1%	0.95
DNN (cells)	87.2%	93.5%	9.4%	1.63	99.4%	99.9%	4.9%	151

Table 1: Performance parameters for BDT and DNN classifiers.





2. *Energy Calibration (Regression)*

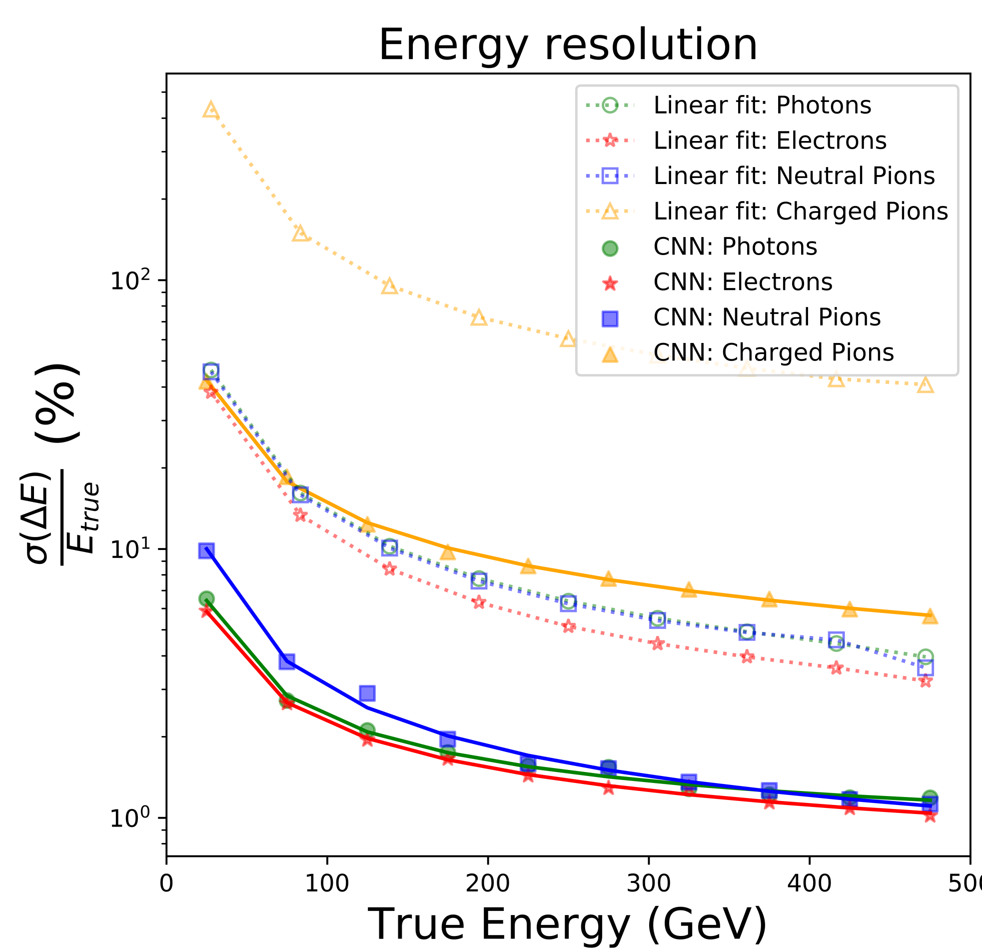
- Energy ***resolution improves with energy***:
  - $\sigma(E)\,/\,E = a/\sqrt{E} \oplus b/E \oplus c.$
  - a = sampling, b = noise, c = leakage.
- *Comparison*:
  - ***Simple calibration***: Sum energies (no noise) and scale.
  - ***CNN calibration***: Cells → Particle energy
- Significant Improvement with CNN

Simple Linear Model

Particle Type	a	b	c
Photons	55.5	1.85	1245
Electrons	42.3	1.51	1037
Neutral pions	55.3	1.71	1222
Charged pions	442	25	11706

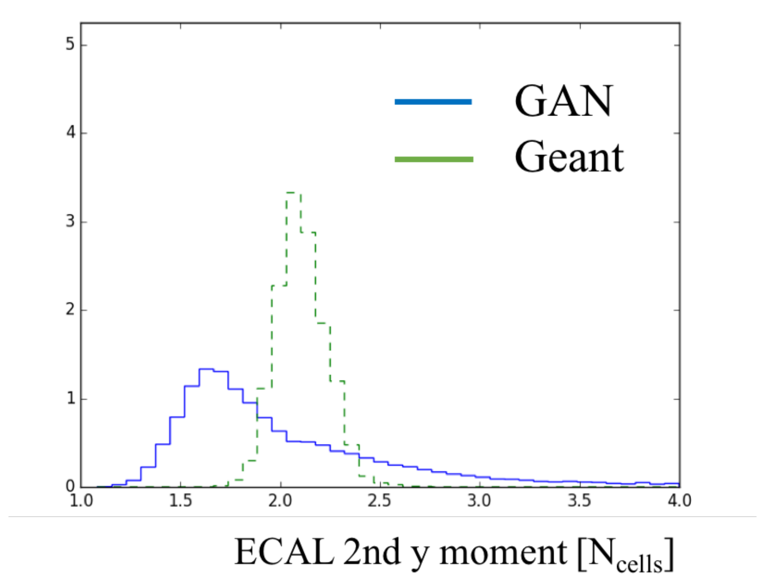
CNN Model

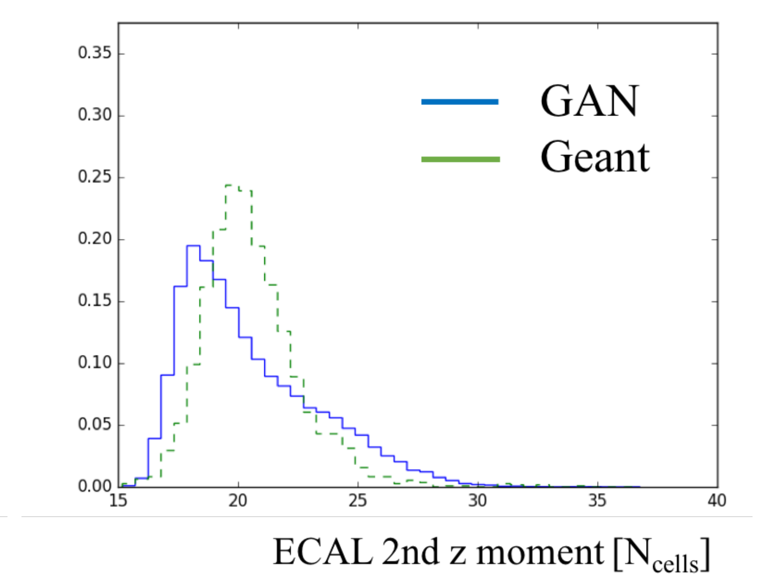
Particle Type	a	b	c
Photons	18.3	0.75	131
Electrons	18.7	0.574	111
Neutral pions	19.3	0.45	231
Charged pions	114	1.02	893



3. Simulation (Generative Model)

- Physics measurements typically require extremely detailed and precise simulation,
  - Software packages (e.g. Geant4) simulated the well understood ***micro-physics*** governing the interaction of particles with matter.
  - Generally very CPU intensive
    - *Example*: ATLAS experiment uses half of the experiment's computing resources for simulation.
- *Task:* ***CNN GAN conditioned on particle energy***
  - Accelerate simulation by many orders of magnitude.
- Promising start… but not yet faithfully reproducing all commonly used features extracted from generated images.

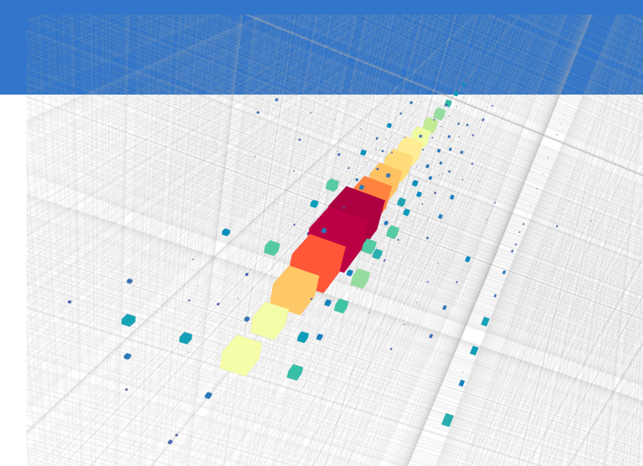


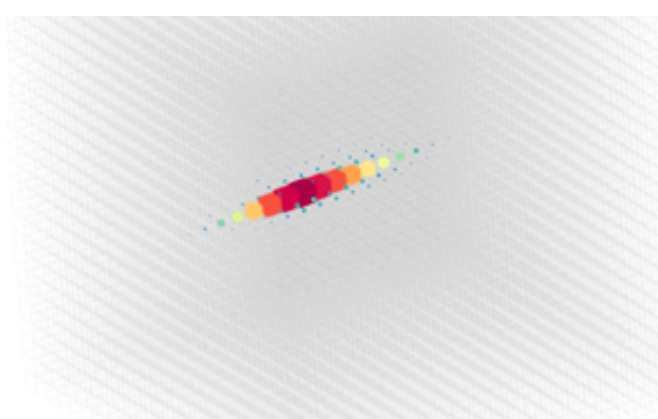


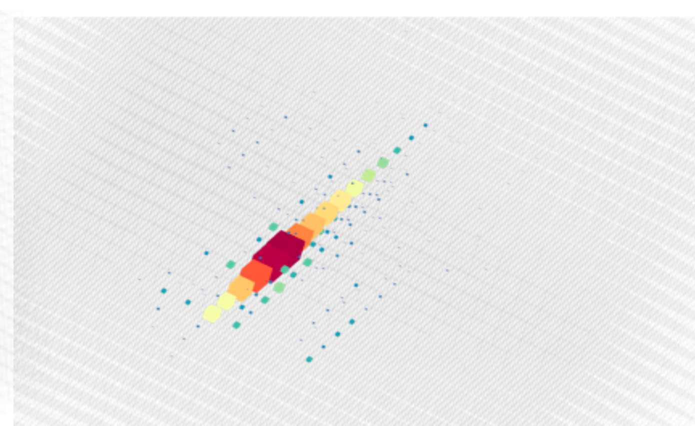
Preliminary

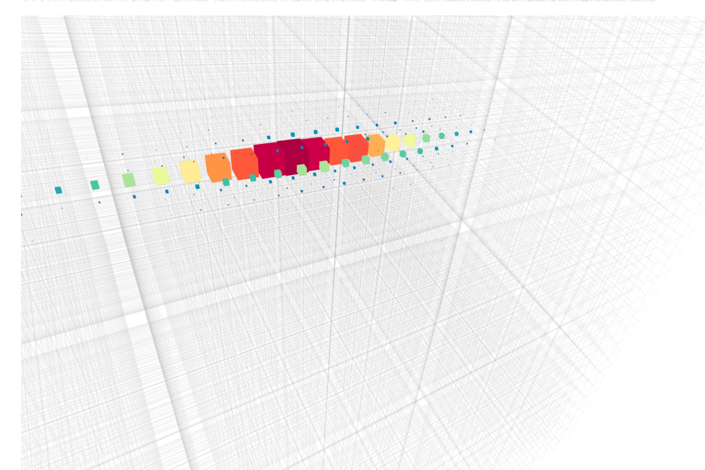
Some images

- Slice energy spectrum
- Start with photons & electrons





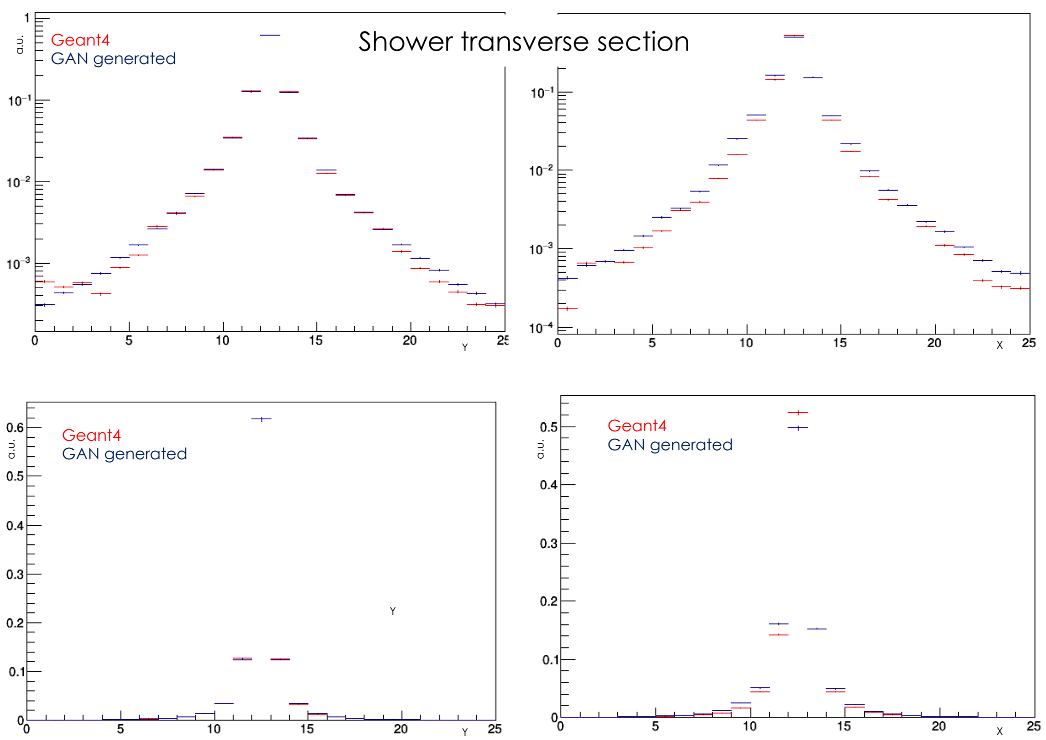




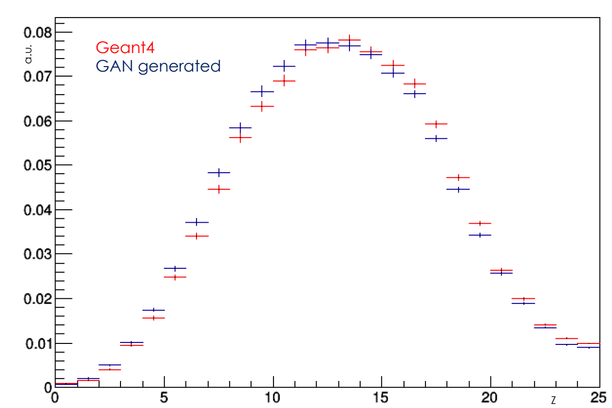
Preliminar

GAN generated electrons

Shower transverse section

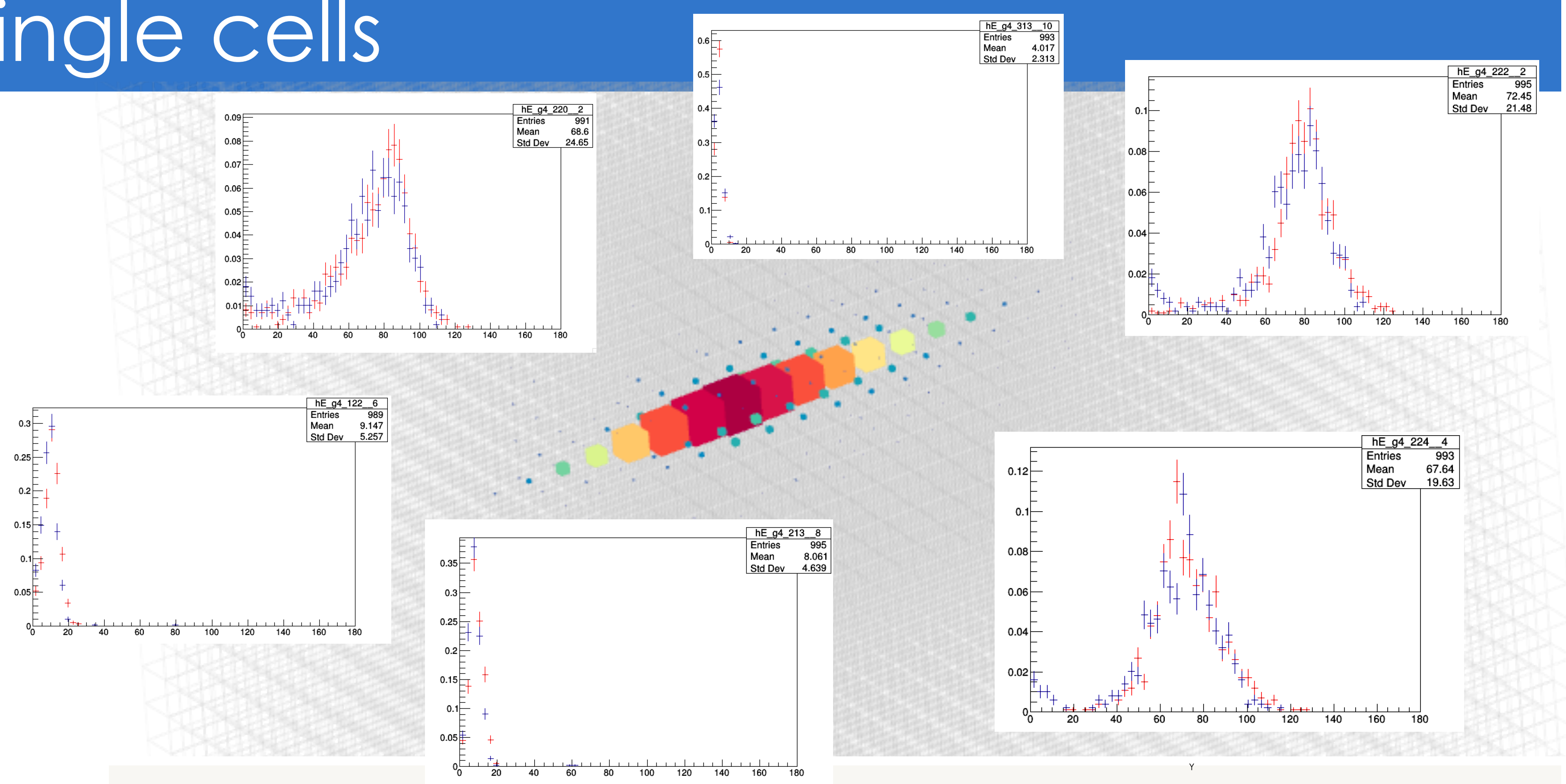


Shower longitudinal section



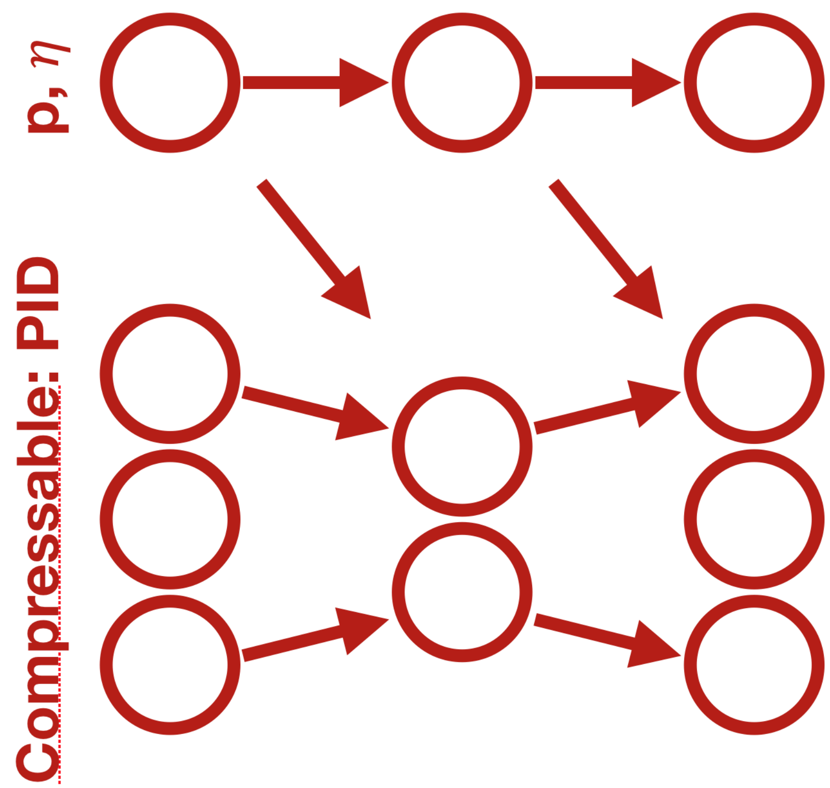
Preliminary

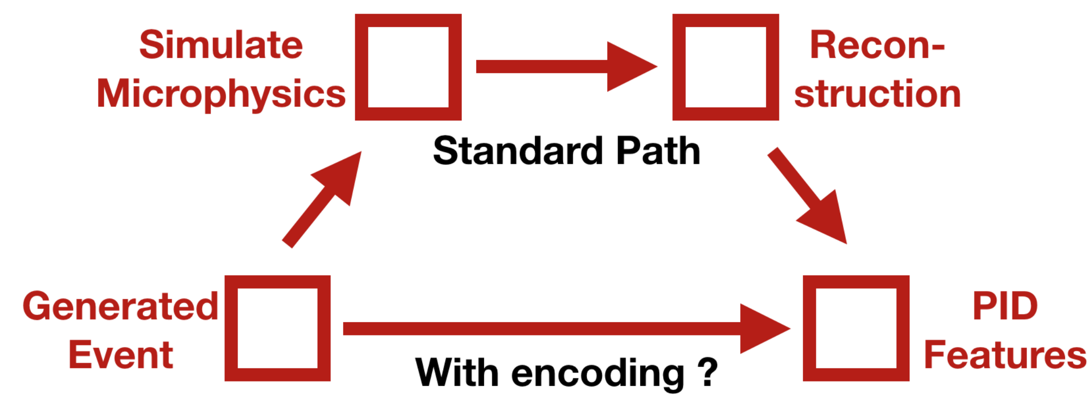
Single cells

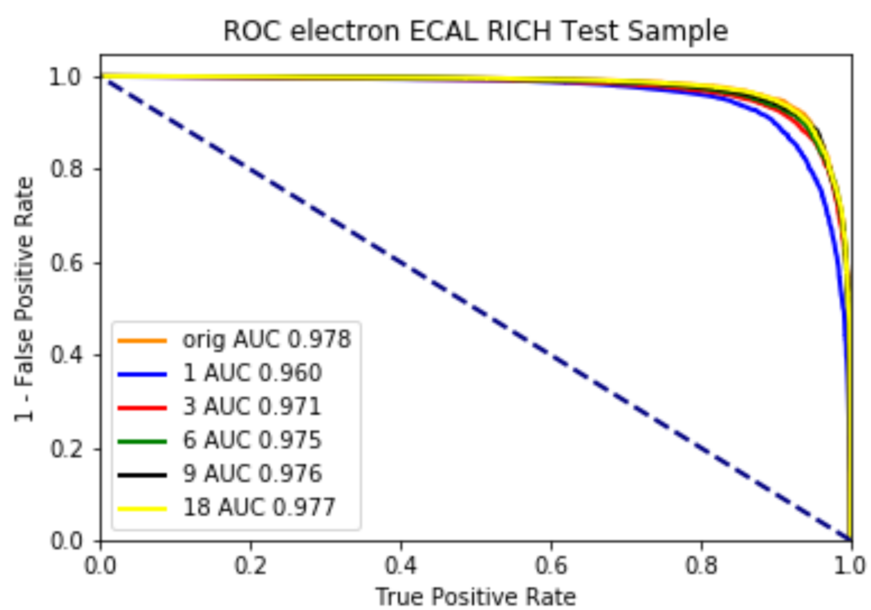


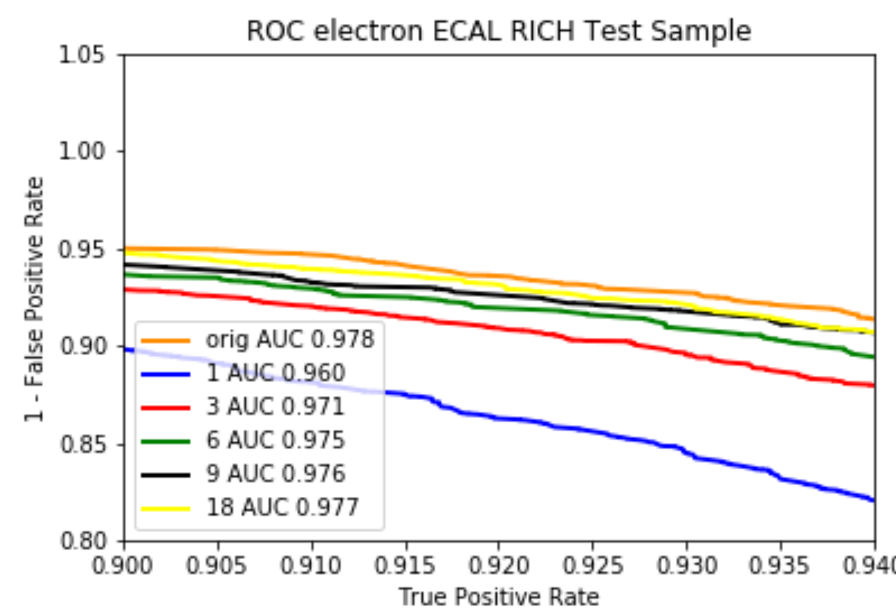
LHCb PID Compression

Constantin Weisser, Mike Williams
LHCb / MIT









https://github.com/weissercn/LHCb_PID_Compression/blob/master/Presentation/Autoencoder_MIT_Weisser.pdf

Simulation

Approximating the Likelihood

- Physics is all about establishing a very precise "model" of the underlying phenomena… so ***we can model our data very well.***

- Enables ***multi-step ab-initio simulations***:

  1. ***Generation***: Standard Model and New Physics are expressed in language of Quantum Field Theory.

     ➡Feynman Diagrams simplify perturbative prediction of HEP interactions among the most fundamental particles (leptons, quarks)

  2. ***Hadronization***: Quarks turn to jets of particles via Quantum Chromodynamics (QCD) at energies where theory is too strong to compute perturbatively.

     ➡Use semi-empirical models tuned to Data.

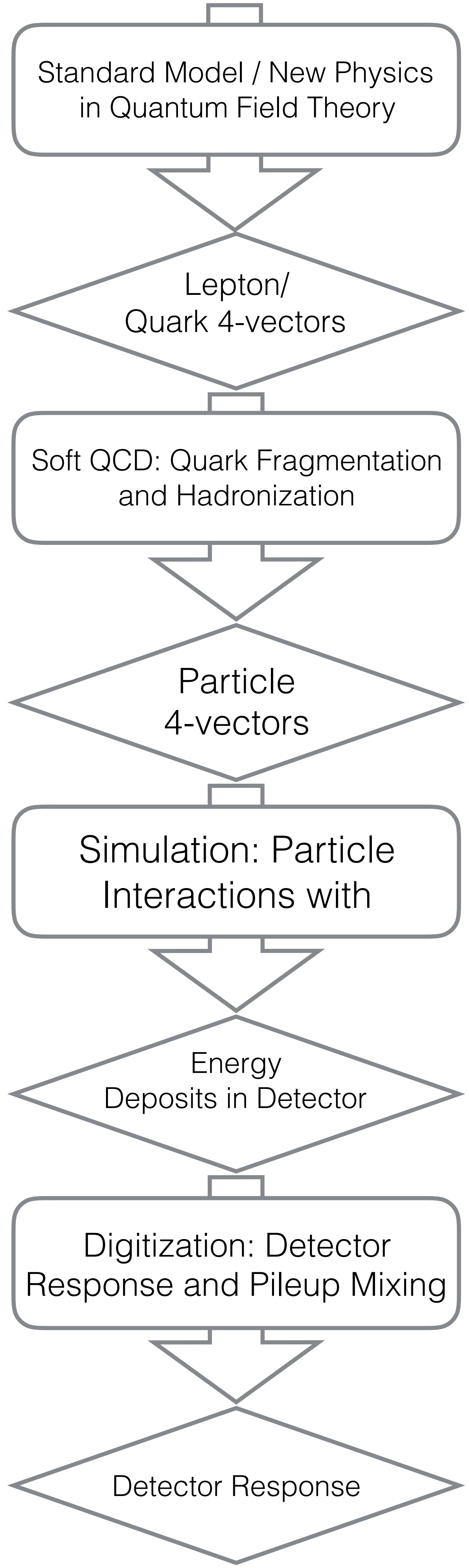
  3. ***Simulation***: Particles interact with the Detector via stochastic processes

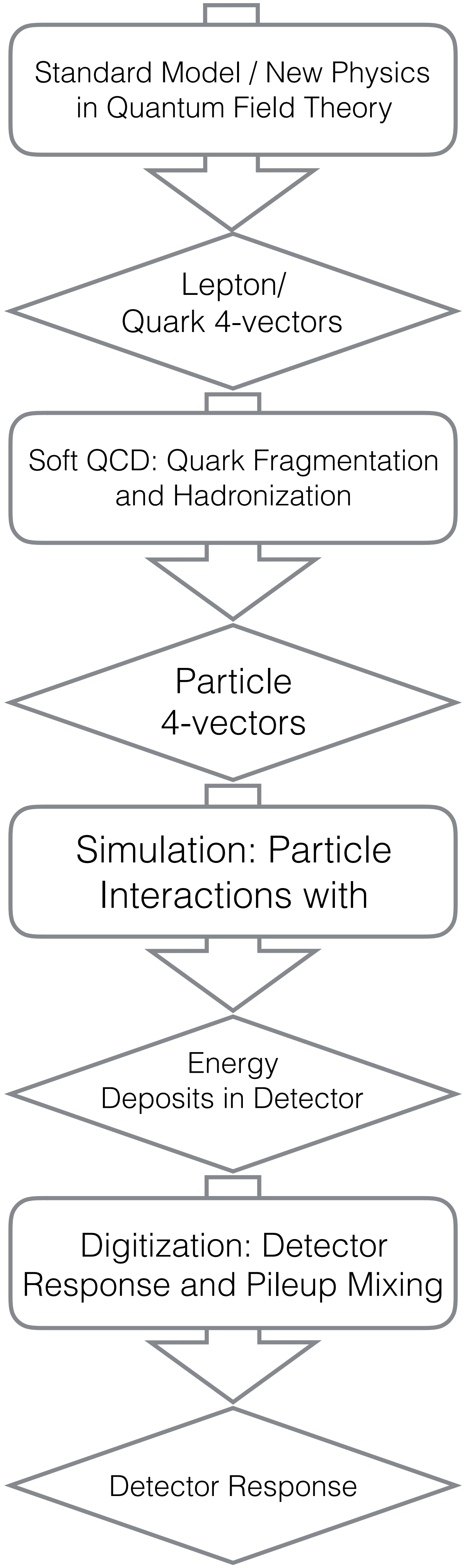
     ➡Use detailed Monte Carlo integration over the "micro-physics"

  4. ***Digitization***: Ultimately the energy deposits lead to electronic signals in the O(100 Million) channels of the detector.

     ➡Model using test beam data and calibrations.

- Output is fed through ***same reconstruction as real data***.





Simulation

- Simulation in HEP is a multi-step process…
- Hadronization and Simulation steps are irreversible.
- Therefore we cannot formally evaluate the likelihoods.
- Rely on Monte Carlo Method to perform Probability Density Estimation
- The simulation step is extremely time consuming… O(1 hr) / collision… LHC produces 40 million/sec
  - ATLAS simulation takes O(50%) of ATLAS resource
    - Lager fraction than CMS because of calorimeter
- For HL-LHC, NLO and NNLO generation will become even more relevant… these can be time consuming too.

Generative Models @ LHC

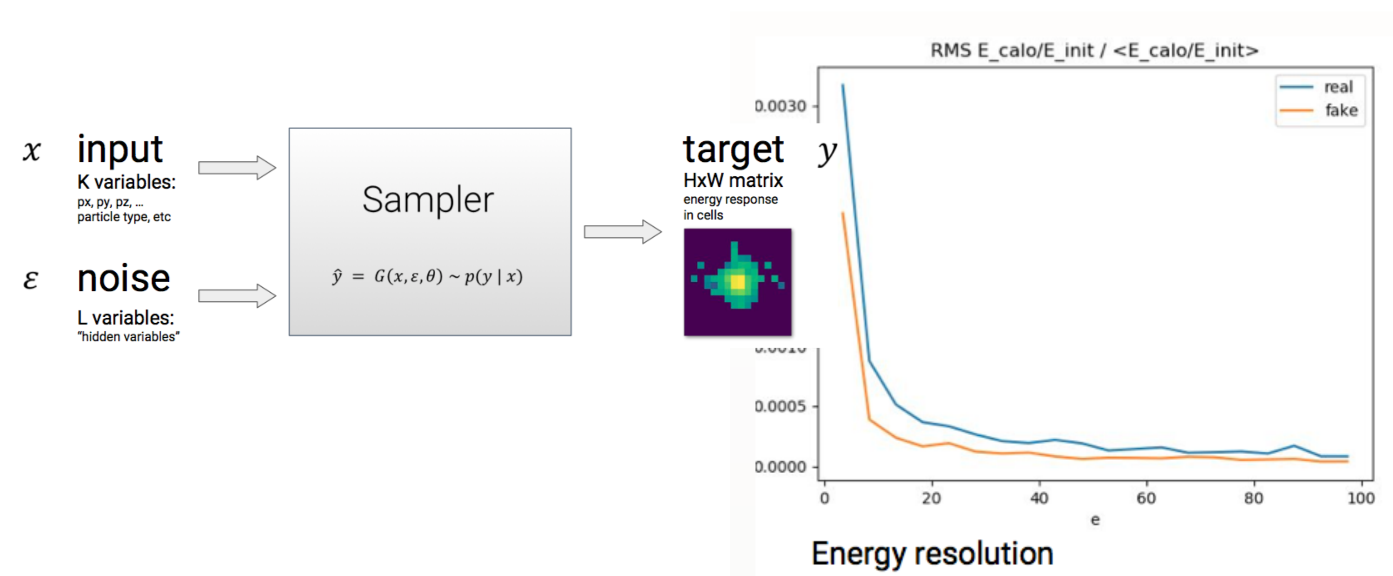
- Every Experiment is Exploring: ATLAS, CMS, LHCb, ALICE

Generative models for fast cluster simulation @ALICE

Most computational expensive step in simulation is the particle propagation
$\Rightarrow$ avoiding the step using generative models

Method	MSE(mm)	speedup
GEANT3	0.085	1
Random (estimated)	166.155	N/A
GAN-MLP	55.385	10^4
GAN-LSTM	54.395	10^4
VAE	37.415	10^4
DCGAN	26.18	10^2
cVAE	13.33	10
proGAN	0.88	30

Fast calorimeter simulation @ LHCb



https://indico.cern.ch/event/681549/contributions/2930939/attachments/1664416/2667649/MachineLearning_LHC.pdf

Learning Particle Physics by Example: Location-Aware Generative Adversarial Networks for Physics Synthesis

Luke de Oliveira[a], **Michela Pag**

[a] *Lawrence Berkeley National Lab*

[b] *Department of Physics, Yale Un*

E-mail: lukedeoliveira@lbl.

Abstract: We provide a bridg
and simulated physical processe
Adversarial Network (GAN) ar
energy depositions from particle
the Location-Aware Generative A
from simulated high energy partic
span over many orders of magnit
jet mass, n-subjettiness, etc.). W
of image quality and validity of
a base for further explorations of

CaloGAN: Simulating 3D High Energy Particle Showers in Multi-Layer Electromagnetic Calorimeters with Generative Adversarial Networks

Michela Paganini[a,b], **Luke de Oliveira**[a], **and Benjamin Nachman**[a]

[a] *Lawrence Berkeley National Laboratory, 1 Cyclotron Rd, Berkeley, CA, 94720, USA*

[b] *Department of Physics, Yale University, New Haven, CT 06520, USA*

E-mail: michela.paganini@yale.edu, lukedeoliveira@lbl.gov, bnachman@cern.ch

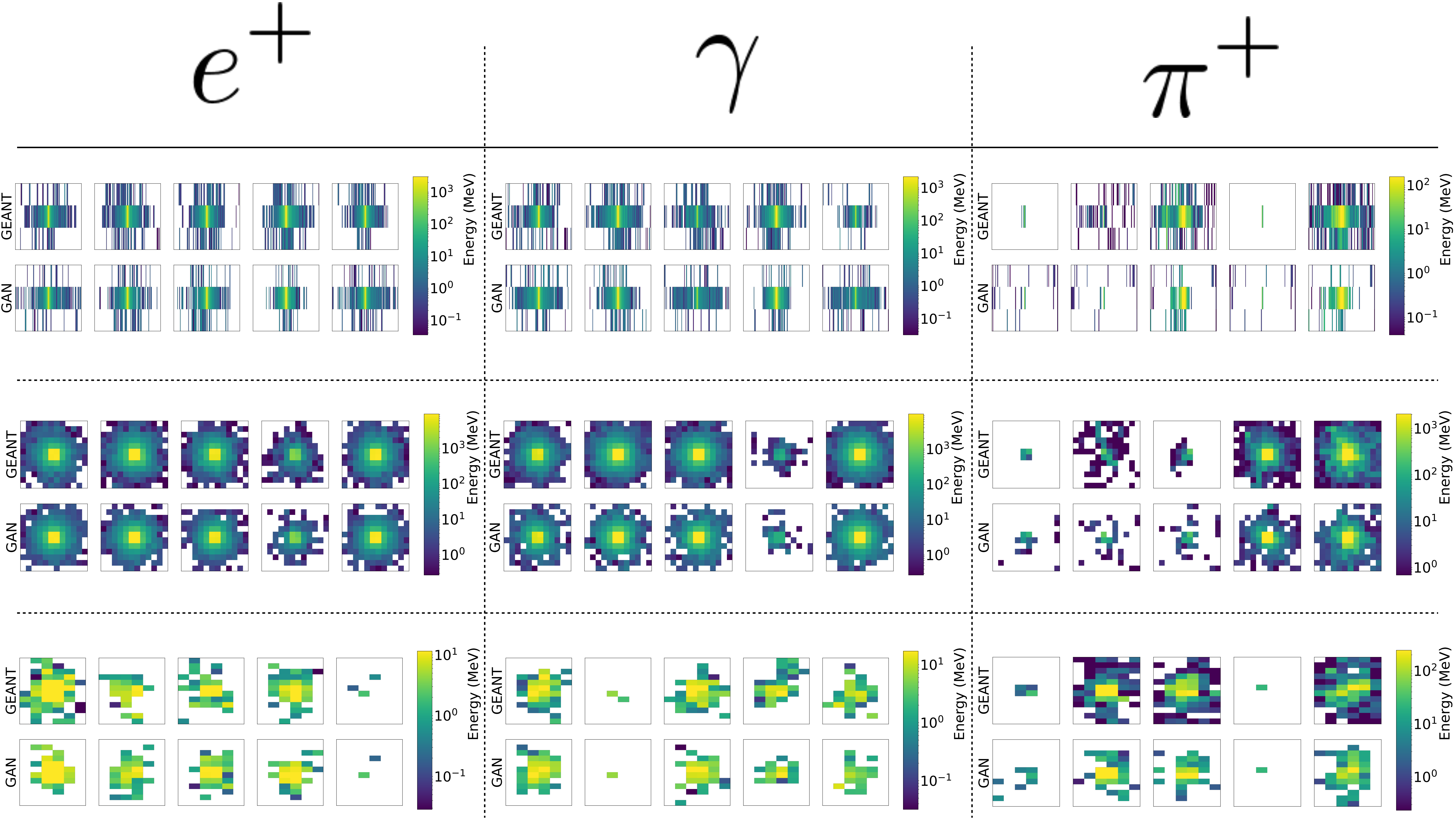
Abstract: Simulation is a key component of physics analysis in particle physics and nuclear physics. The most computationally expensive simulation step is the detailed modeling of particle showers inside calorimeters. Full detector simulations are too slow to meet the growing demands resulting from large quantities of data; current fast simulations are not precise enough to serve the entire physics program. Therefore, we introduce CaloGAN, a new fast simulation based on generative adversarial neural networks (GANs). We apply the CaloGAN to model electromagnetic showers in a longitudinally segmented calorimeter. This represents a significant stepping stone toward a full neural network-based detector simulation that could save significant computing time and enable many analyses now and in the future. In particular, the CaloGAN achieves speedup factors comparable to or better than existing fast simulation techniques on CPU ($100\times$-$1000\times$) and even faster on GPU (up to $\sim 10^5\times$)) and has the capability of faithfully reproducing many aspects of key shower shape variables for a variety of particle types.

https://arxiv.org/pdf/1701.05927.pdf

https://arxiv.org/pdf/1705.02355.pdf

Qualitative Performance (2)

Yale



Timing

M. Paganini et al., 1705.02355

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 ←

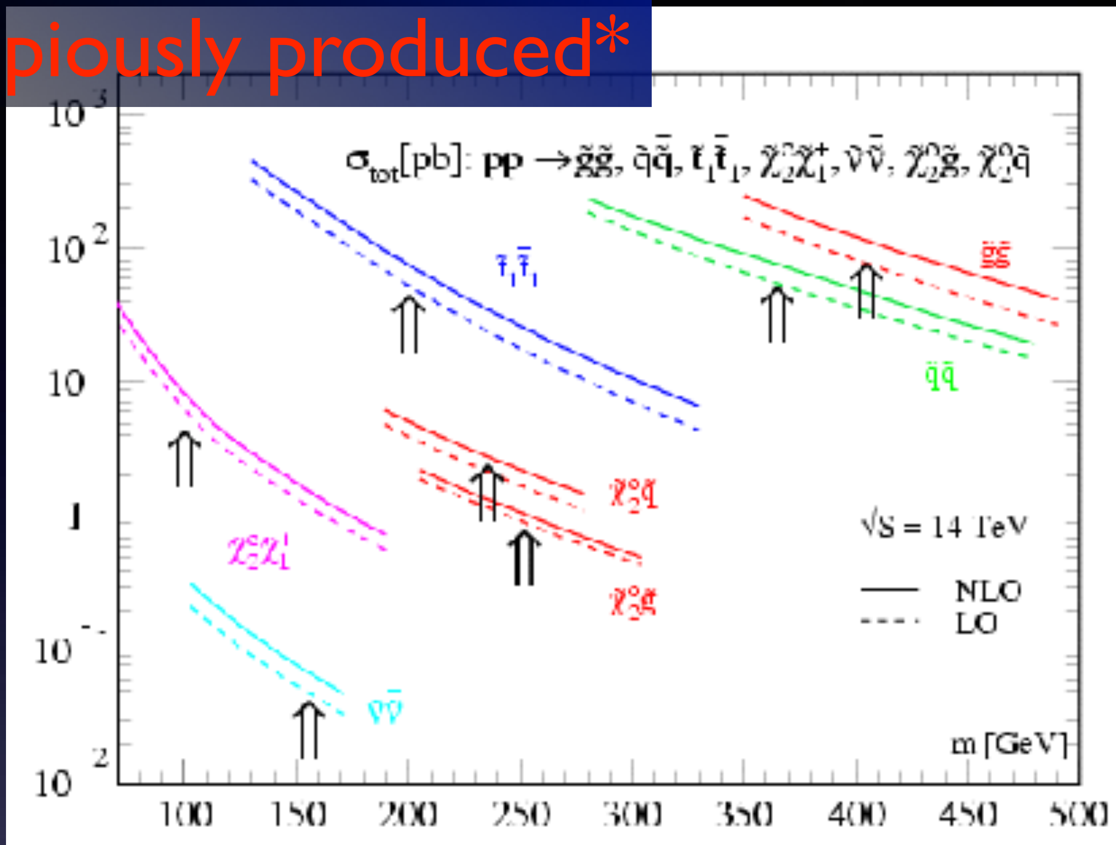
See also S. Vallecorsa et al. (GeantV), C. Guthrie et al. (NYU), W. Wei et al. (LCD dataset group), D. Salamani et al. (Geneva), D. Rousseau et al. (Orsay), L. de Oliveira et al. (Berkeley)

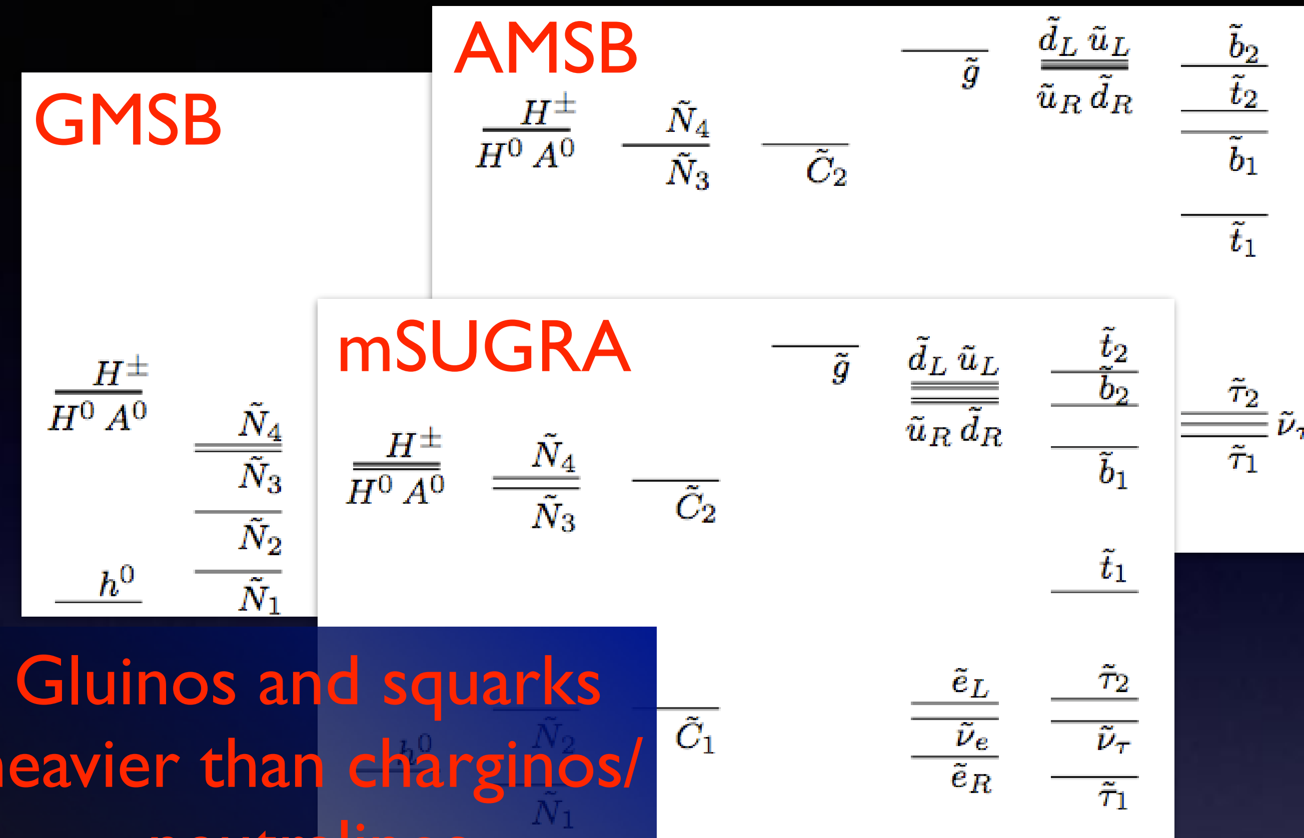
Analysis

HEP Searches (SUSY Example)

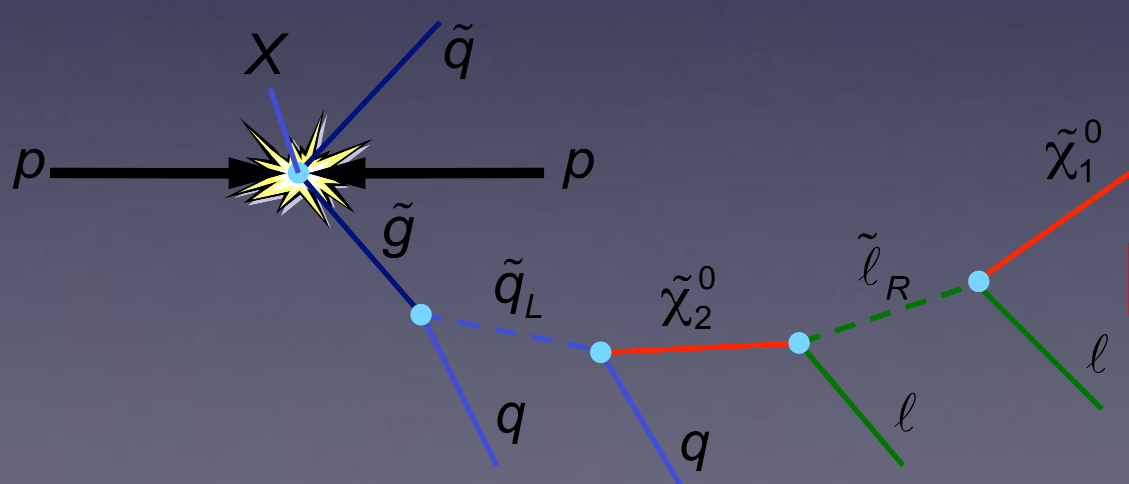
SUSY at LHC

Gluinos and squarks copiously produced*

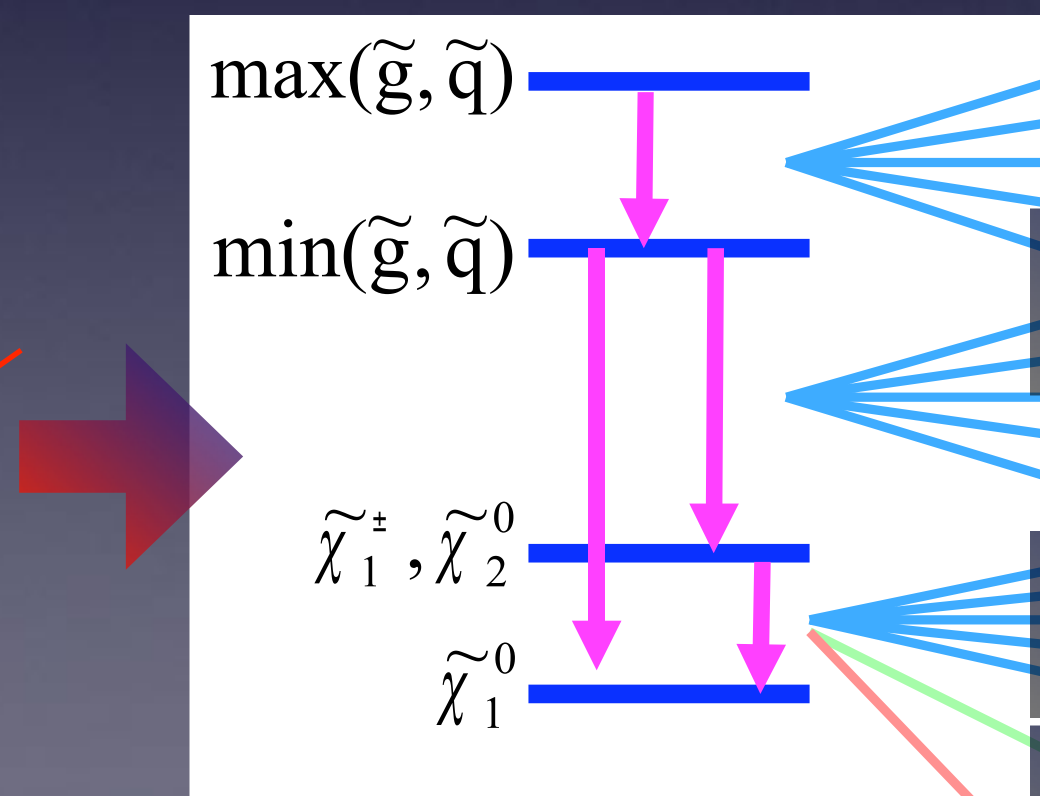




Gluinos and squarks heavier than charginos/ neutralinos



Long Cascades



Many Jets (100's of GeV)

Leptons (10's of GeV)

Missing Energy (100's of GeV)

Inclusive Signatures

Signature	Motivating Model(s)	Comments
1 Jet + 0 Lepton + MET	• Large Extra Dim (ExoGraviton) • strong qG production, G propagate in extra Dim • Planck Scale is MD in 4+δ dim • Normal Gravity >> R • SUSY • qg→ISR + 2 Neutralino or squark + Neutralino	• Not primary discovery channel for SUGRA, GMSB, AMSB... but helps in characterization • Possible leading discovery for neutralino NLSP with nearly degenerate gluino
2,3,4 [b]-Jet + 0 Lepton + MET	• Squark/gluino production • squark→q+LSP, gluino→q+squark+LSP	• Possible leading squark/gluino discovery channel • Must manage QCD bkg
2,3,4 [b]-Jet + 1 Lepton + MET	• squark/gluino production with cascades which include electroweak (or partner) decays • high tan β leads to more b/t/τ's	• Lepton requirement suppresses QCD • τ's partially covered by e/μ
2 lepton + MET	• Same sign: gluino cascade can have either sign lepton... squark/gluino prod can produce same sign. • Opposite sign: squark/gluino decay mediated by Z (or partner) • Same flavor: 2 leptons from same sparticle cascade must be same flavor	• Reduced SM backgrounds for same sign • Opposite Sign-Flavor Subtraction
3 lepton + MET	• SUSY events ending in Chargino/neutralino pair decays • Weak Chargino/Neutralino production • Exotic sources	• Low SM bkgs
2 photon + MET	• GMSB models with gravitino LSP and neutralino or stau NLSP • UED- each KK partons cascade to LKP which decays to graviton + γ	• No SUSY limit (not sensitive at the time)

0 Lepton Event Selections

- No leptons (medium electrons and muons) >10 GeV
- 4 signal regions defined to maximize m_{squark}-m_{gluino} coverage :
  - At least 2 Jets
    - Low mass squark anti-squark (A)
    - High mass squark anti-squark (B)
  - At least 3 Jets
    - Direct gluino pairs (C)
    - Associated gluino-squark (D)
      - Higher x-section → Tighter cuts!

		A	B	C	D
Pre-selection	Number of required jets	≥ 2	≥ 2	≥ 3	≥ 3
	Leading jet p_T [GeV]	> 120	> 120	> 120	> 120
	Other jet(s) p_T [GeV]	> 40	> 40	> 40	> 40
	E_T^{miss} [GeV]	> 100	> 100	> 100	> 100
Final selection	$\Delta\phi(\text{jet}, \vec{P}_T^{miss})_{min}$	> 0.4	> 0.4	> 0.4	> 0.4
	E_T^{miss}/m_{eff}	> 0.3	–	> 0.25	> 0.25
	m_{eff} [GeV]	> 500	–	> 500	> 1000
	m_{T2} [GeV]	–	> 300	–	–

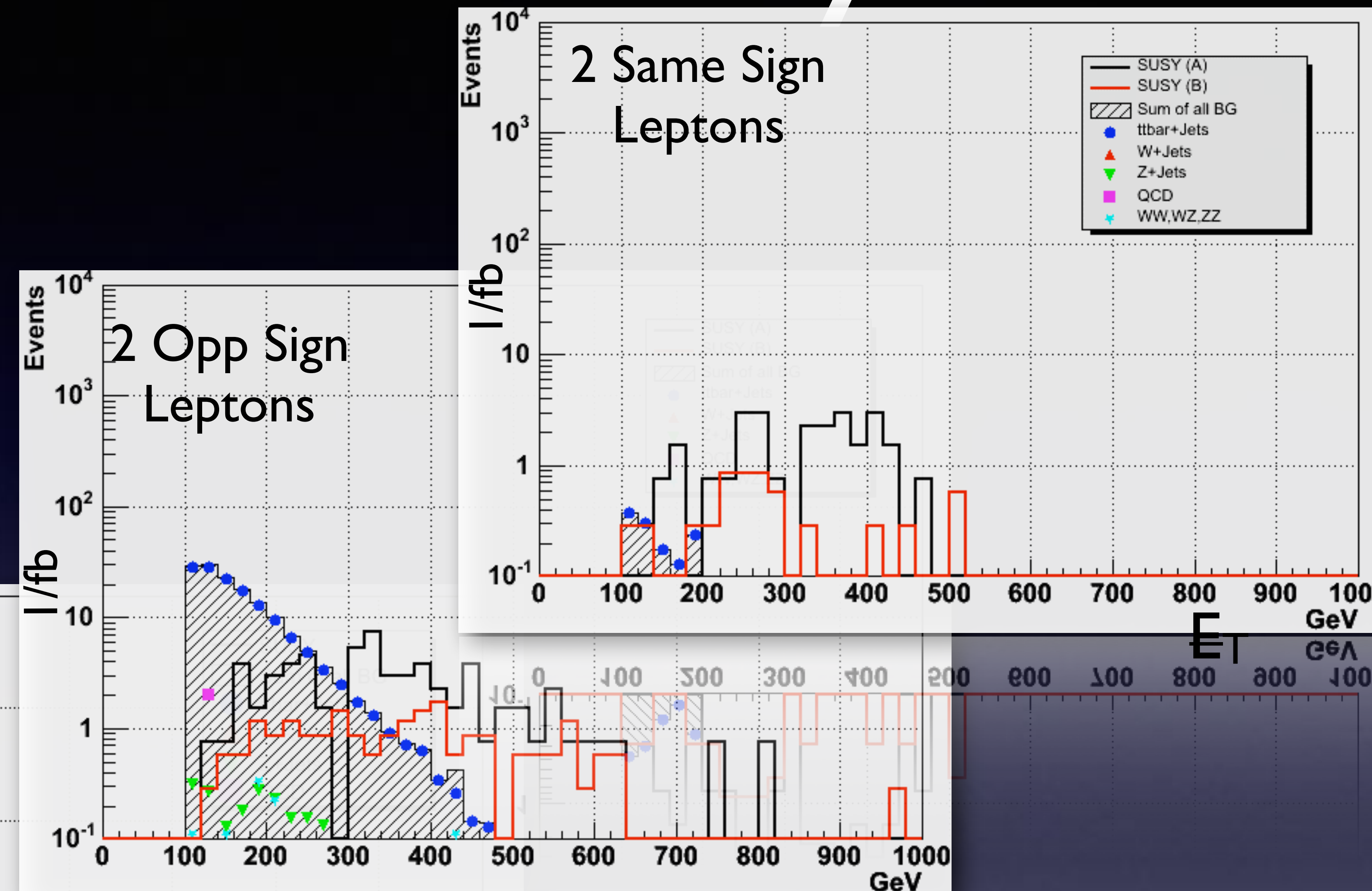
$$m_{\text{eff}} \equiv \sum_{i=1}^{n} |\mathbf{p}_T^{(i)}| + E_T^{\text{miss}}$$

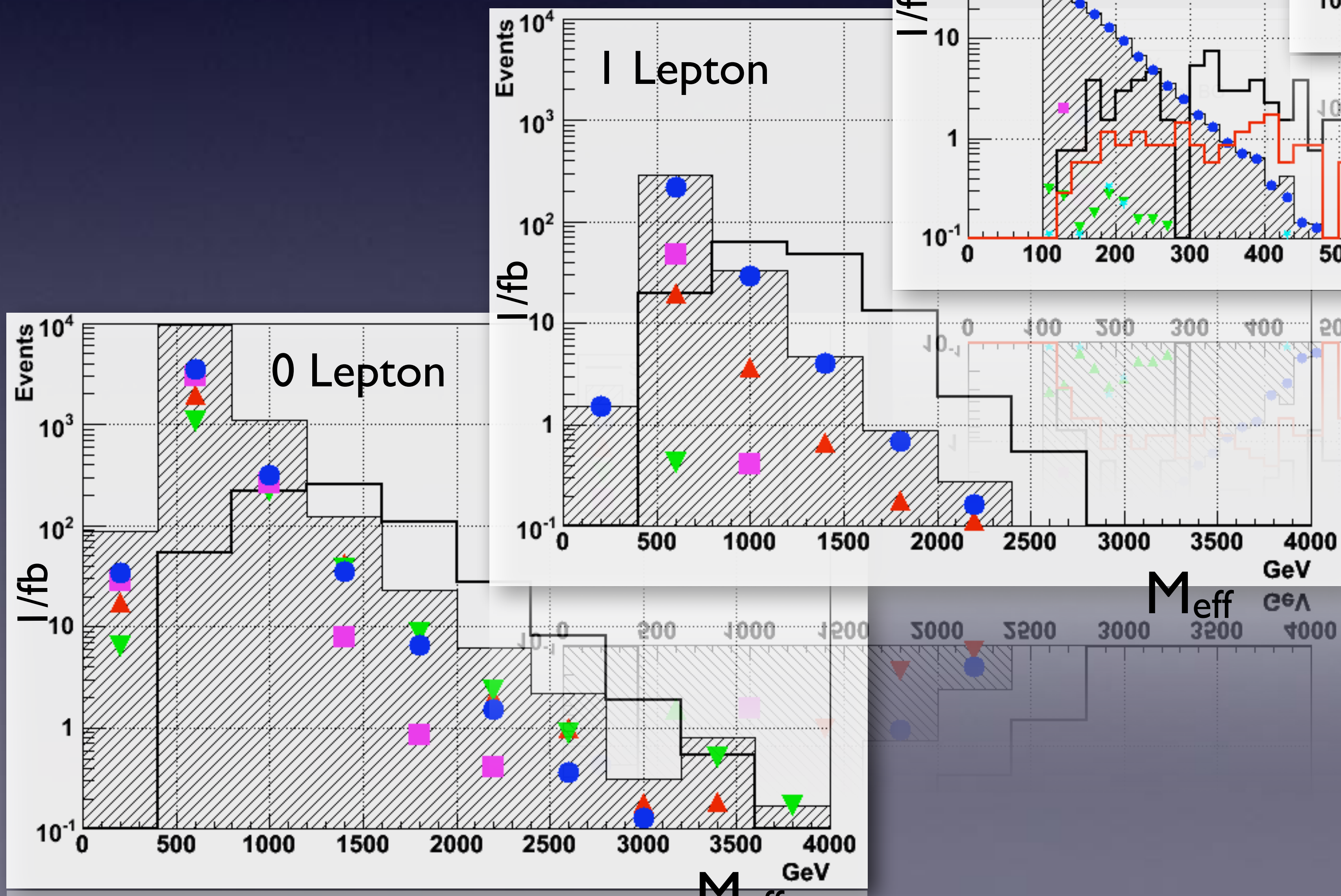
$$m_{T2}\left(\mathbf{p}_T^{(1)}, \mathbf{p}_T^{(2)}, \not{\mathbf{p}}_T\right) \equiv \min_{\not{\mathbf{q}}_T^{(1)}+\not{\mathbf{q}}_T^{(2)}=\tilde{\mathbf{E}}_T^{\text{miss}}} \left\{\max\left(m_T\left(\mathbf{p}_T^{(1)}, \not{\mathbf{q}}_T^{(1)}\right), m_T\left(\mathbf{p}_T^{(2)}, \not{\mathbf{q}}_T^{(2)}\right)\right)\right\}$$

$$m_T^2\left(\mathbf{p}_T^{(i)}, \not{\mathbf{q}}_T^{(\mathbf{i})}\right) \equiv 2|\mathbf{p}_T^{(i)}||\not{\mathbf{q}}_T^{(\mathbf{i})}| - \mathbf{2p}_T^{(\mathbf{i})} \cdot \not{\mathbf{q}}_T^{(\mathbf{i})}$$

Standard SUSY Analyses

- Require:
  - Large $\not{E}_T$ (> 100 GeV)
  - 4 Hard Jets
  - Sphericity?
- Look at: $M_{eff} = \sum^{4}_{jets} p_T + \not{E}_T$ for N=0,1,2 (SS/OS) leptons

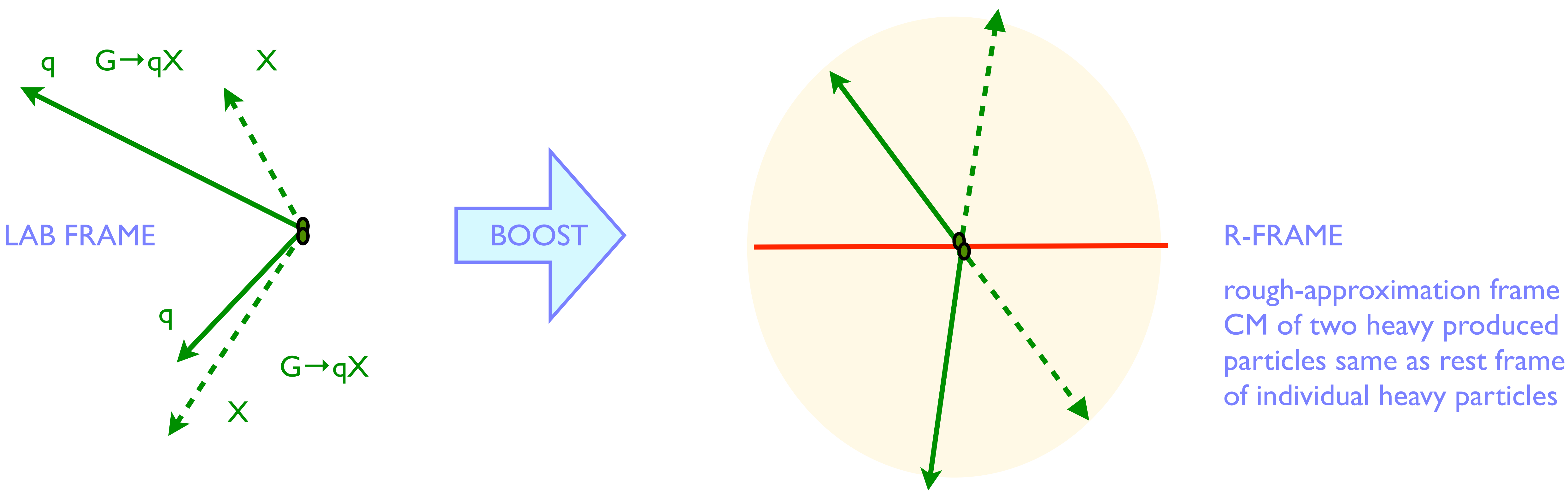




- Adding Leptons reduces QCD Background
- Better S/B in same sign than opposite sign

Razor variables

- Razor variables (C. Rogan arXiv:1006.2727) are kinematical variables to identify SUSY-like events
- Variables take advantage of symmetric decay of SUSY events by forming two hemispheres (aka mega-jets) using all final state visible objects



- define variables that take advantage of the symmetry of the SUSY event:

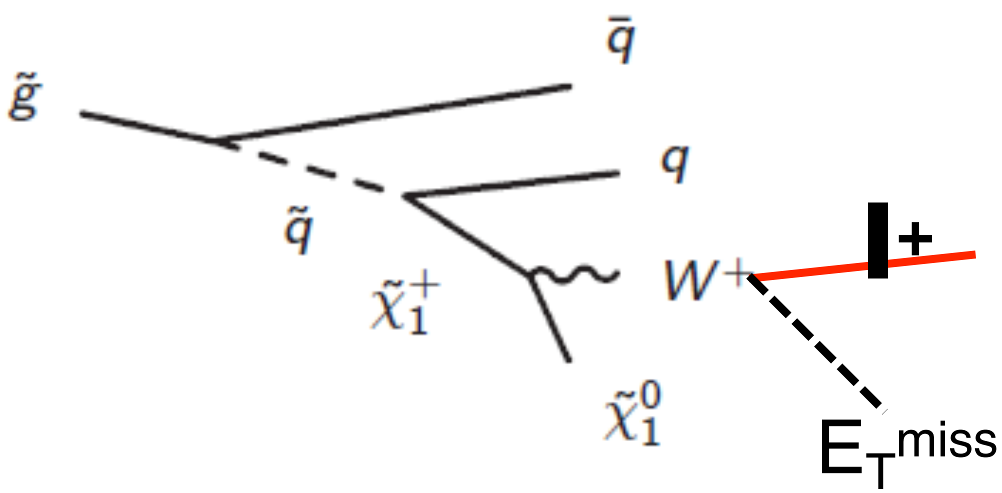
  $\boldsymbol{\gamma_R M_R}$ contains longitudinal event information, related to the SUSY mass scale

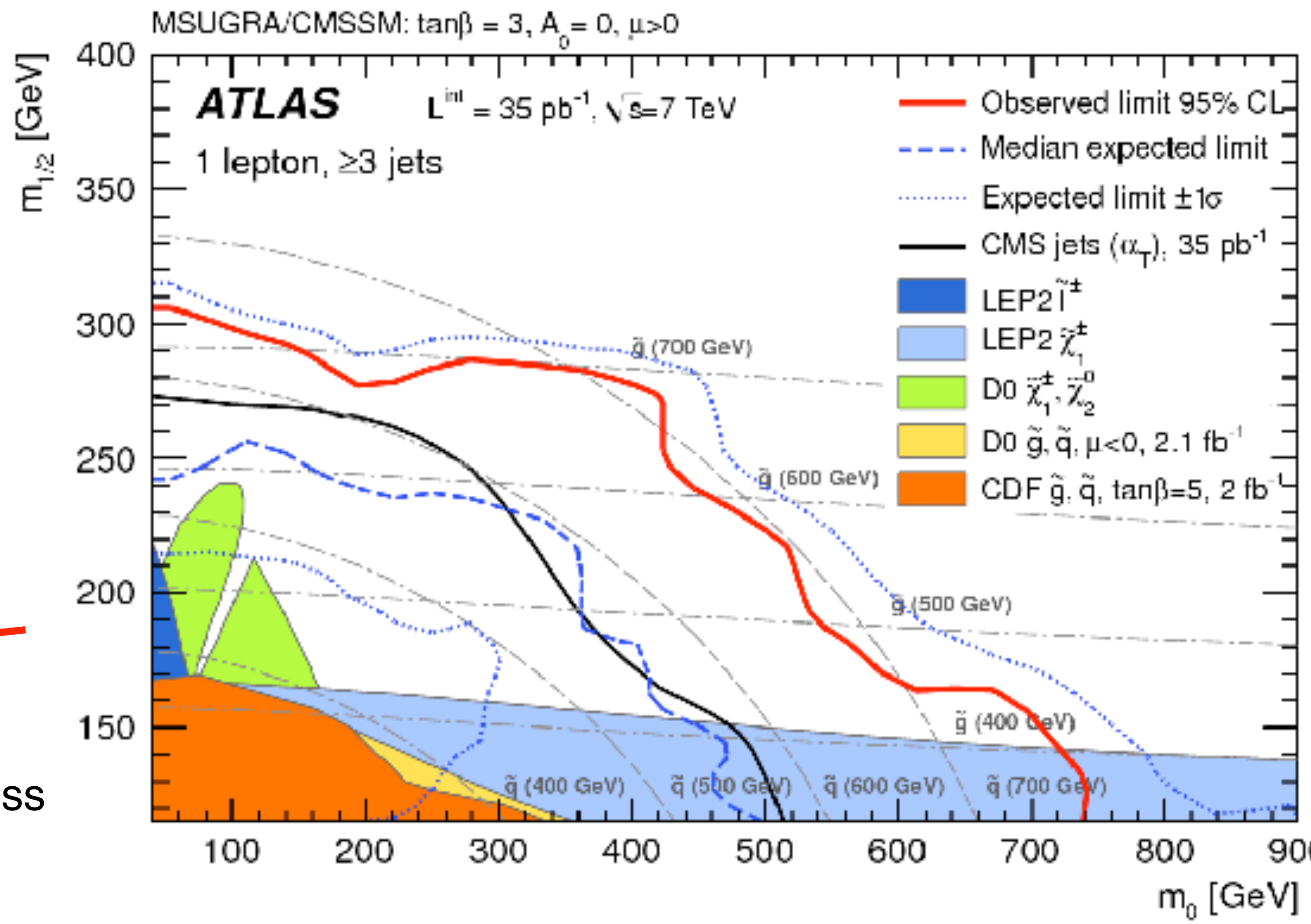
  $\boldsymbol{M_T^R}$ contains transverse event information

  $\boldsymbol{R} = M_T^R/\gamma_R M_R$ a signal-to-background discriminant

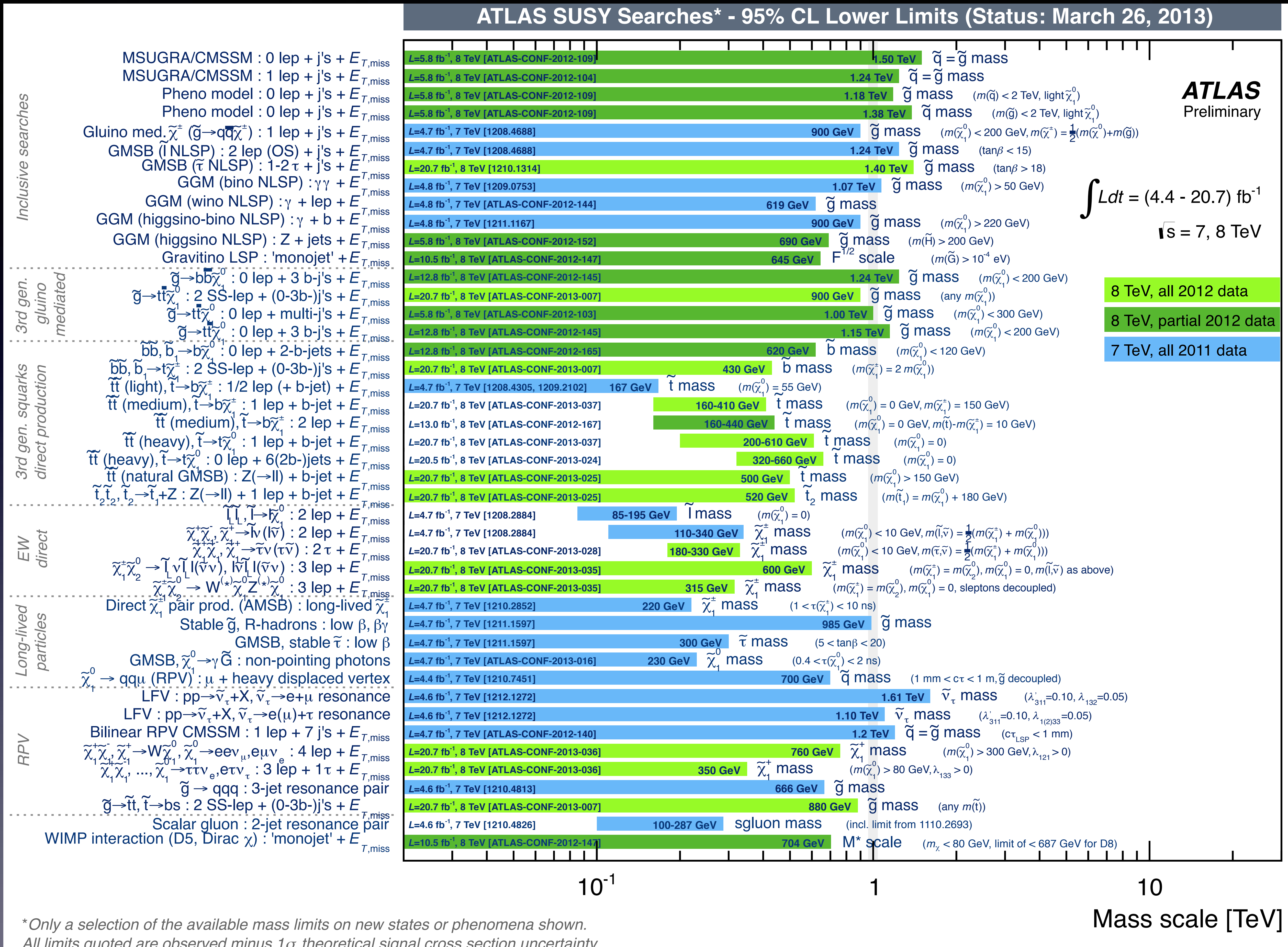
The 1st ATLAS SUSY paper

Phys. Rev. Lett. **106**, 131802 (2011)





- heavy colored particles production fully benefits from the LHC energy
    - if SUSY: gluinos & squarks
- surpassed Tevatron with only 0.035 fb^{-1}
    - one of the top-cited LHC papers



Higgs is at 125 GeV and no sign of new physics at LHC → Nature is not "natural"?

Data Analysis

- Objectives:

    - ***Searches*** (hypothesis testing): Likelihood Ratio Test (Neyman-Pearson lemma)

    - ***Limits*** (confidence intervals): Also based on Likelihood

    - ***Measurements***: Maximum Likelihood Estimate

$$\frac{P(x|H_1)}{P(x|H_0)} > k_\alpha$$

- ***Likelihood***

$$p(\{x\}|\theta) = \mathrm{Pois}(n|\nu(\theta)) \prod_{e=1}^{n} p(x_e|\theta)$$

    - n Independent Events (e) with Identically Distributed Observables ($\{x\}$)

    - Significant part of Data Analysis is ***approximating the likelihood*** as best as we can.

- Likelihood is estimated via Monte Carlo sampling using highly faithful Simulation

Likelihood Approximations

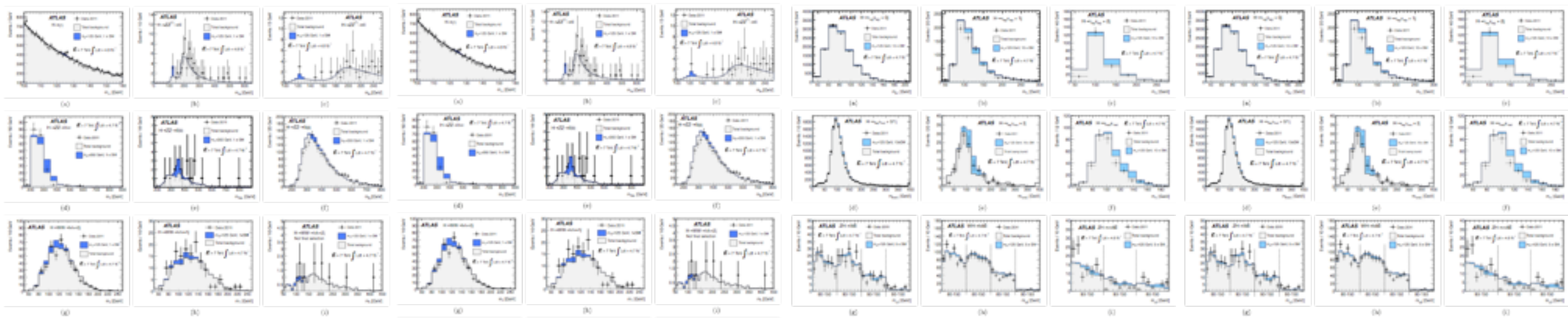
- Need $P(\{x_e\}|\theta)$ of an observed event (e). The better we do, the more sensitive our measurements.
- Monte Carlo can only be done in the ***forward mode*** because of Hadronization and Simulation
  - ➡ ***cannot evaluate the likelihood.***
- So we simulate a lot of events and use a ***Probability Density Estimator (PDE)***, e.g. a histogram.
  - $\{x_e\}$ = {100M Detector Channels} or even { particle 4-vectors } are too high dimension.
  - Instead we derive $\{x_e\}$ = { small set of physics motivated observables } → ***Lose information.***
    - ***Isolate signal*** dominating regions of $\{x_e\}$ → ***Lose Efficiency.***
    - Sometimes use ***classifiers*** to further reduce dimensionality and improve significance
    - ***Profile the likelihood*** in 1 or 2 (ideally uncorrelated) observables.
- Alternative, try to brute force calculate via ***Matrix Element Method***:

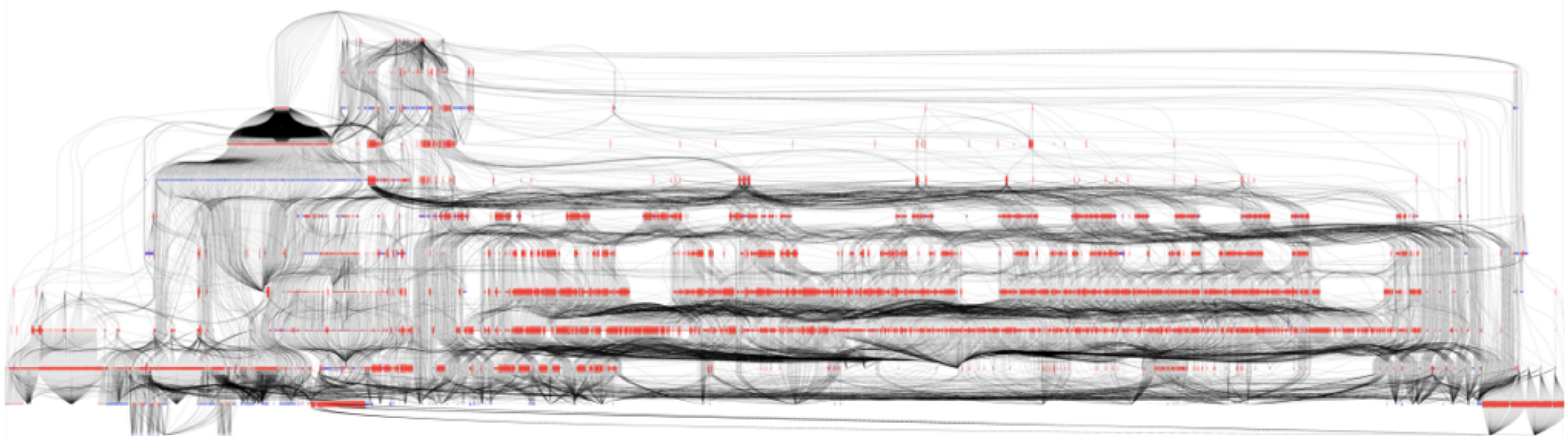
$$\mathcal{P}(\boldsymbol{p}^{vis}|\alpha) = \frac{1}{\sigma_\alpha}\int d\Phi dx_1 dx_2 |M_\alpha(\boldsymbol{p})|^2 W(\boldsymbol{p}, \boldsymbol{p}^{vis})$$

- But it's technically difficult, computationally expensive, mistreats hadronization, and avoids simulation by highly simplifying the detector response.

Slide from Kyle Cranmer:

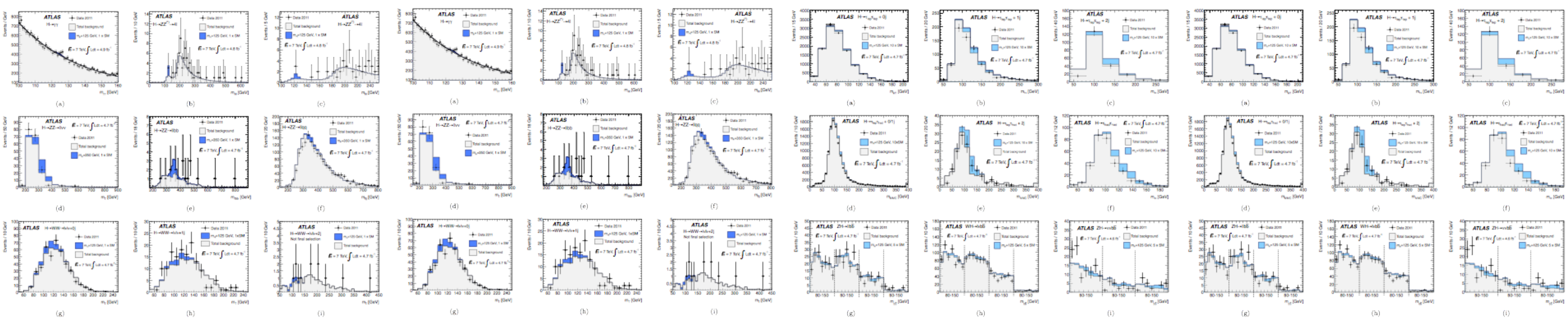
LIKELIHOOD-BASED COMBINATIONS

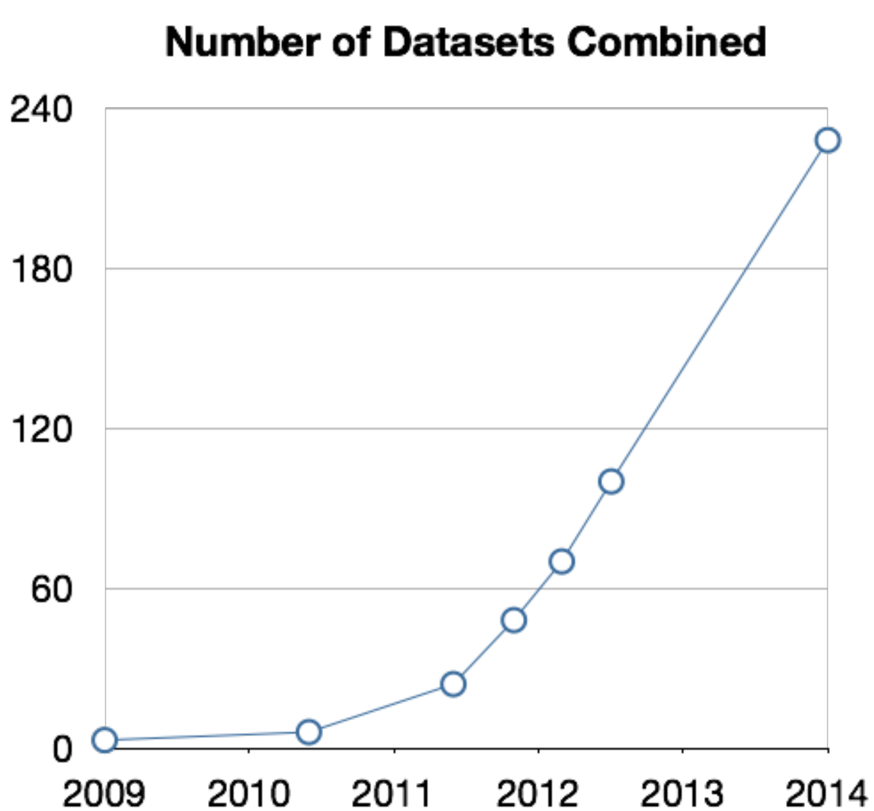


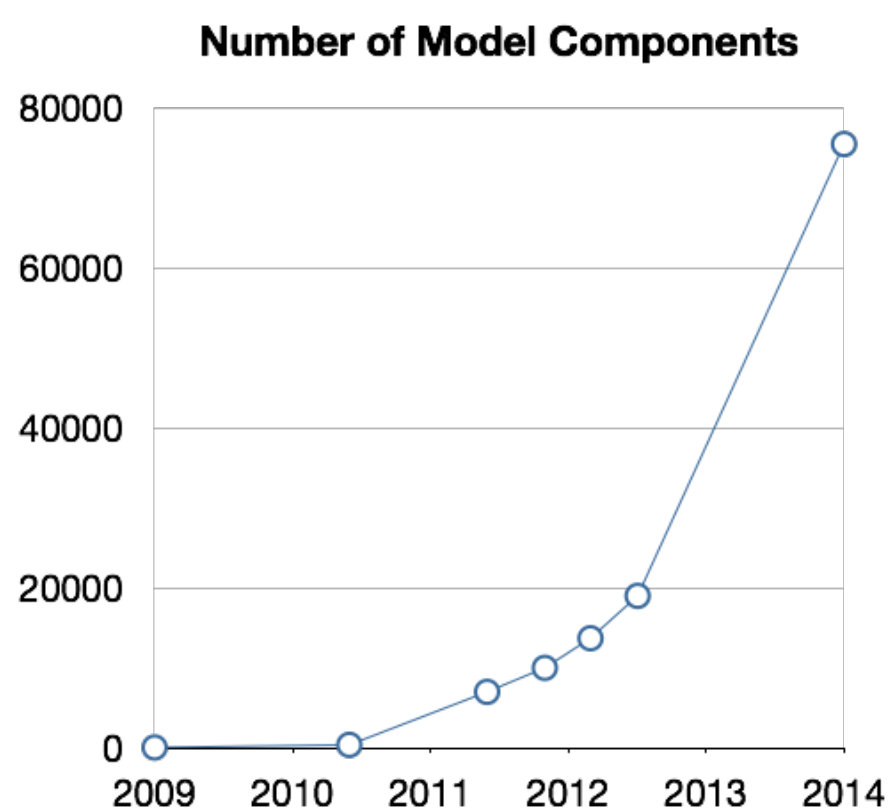


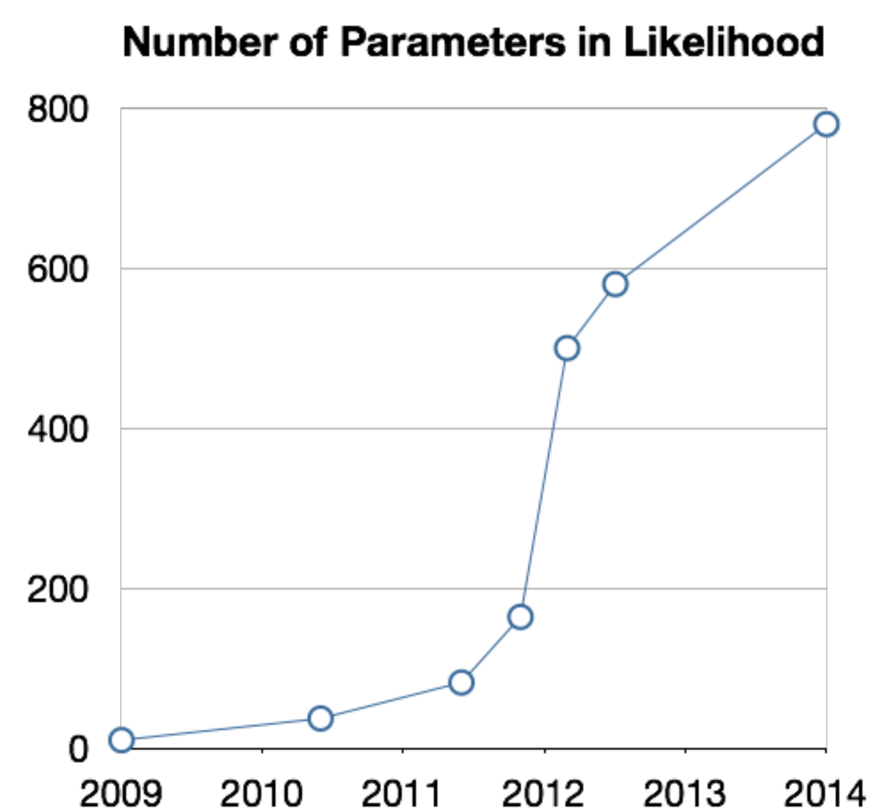
$$\mathbf{f}_{\text{tot}}(\mathcal{D}_{\text{sim}}, \mathcal{G}|\boldsymbol{\alpha}) = \prod_{c\in\text{channels}} \left[\text{Pois}(n_c|\nu_c(\boldsymbol{\alpha})) \prod_{e=1}^{n_c} f_c(x_{ce}|\boldsymbol{\alpha})\right] \cdot \prod_{p\in\mathbb{S}} f_p(a_p|\alpha_p)$$

Collaborative Statistical Modeling



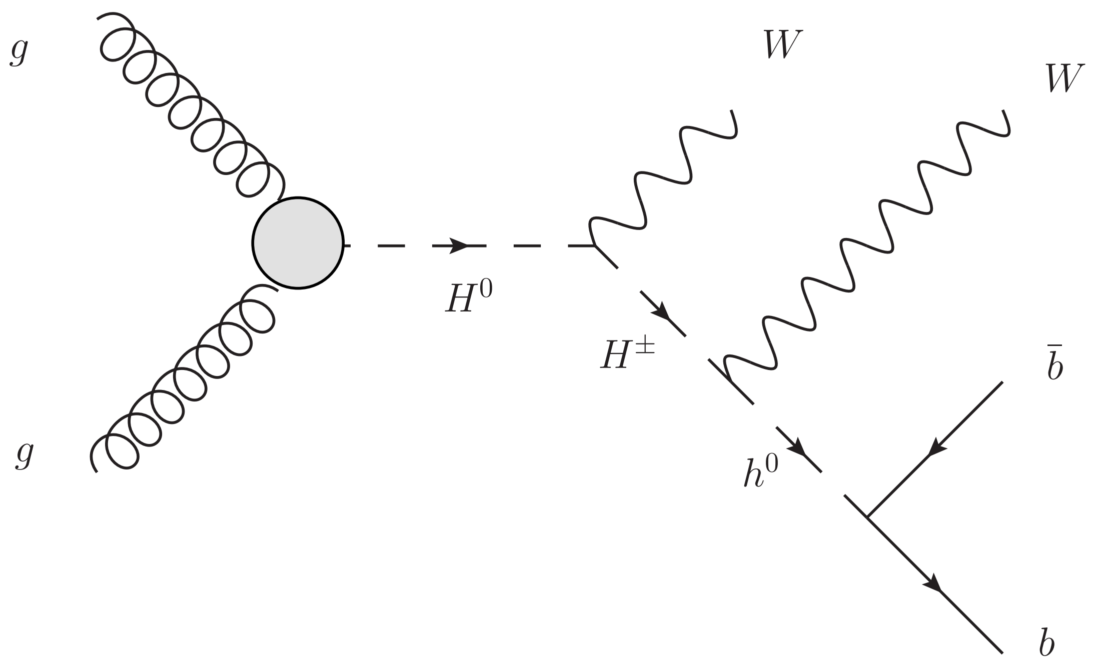




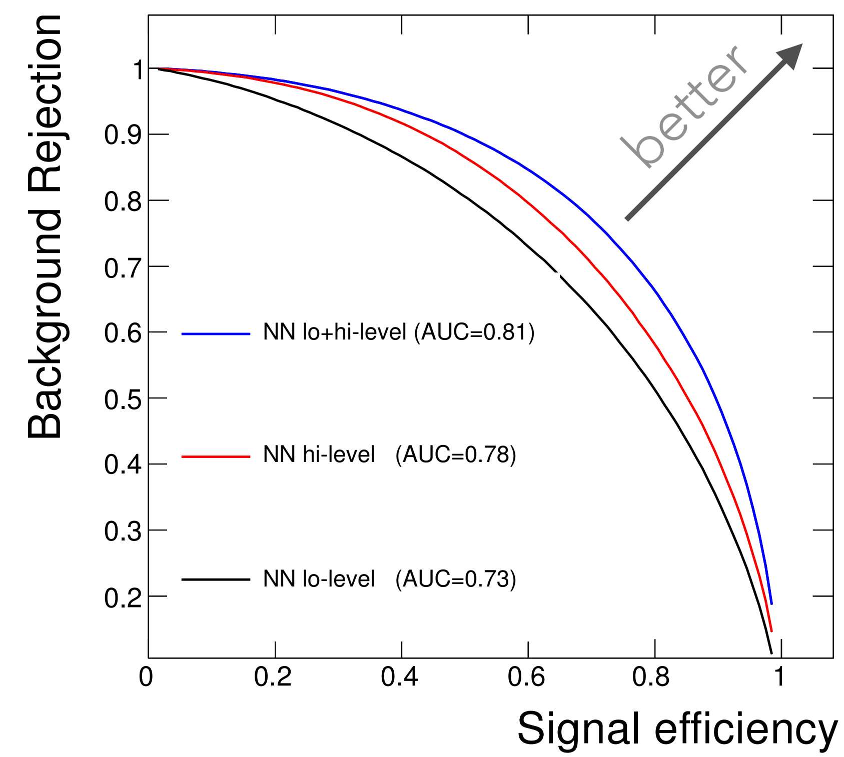


DEEP LEARNING IN HEP

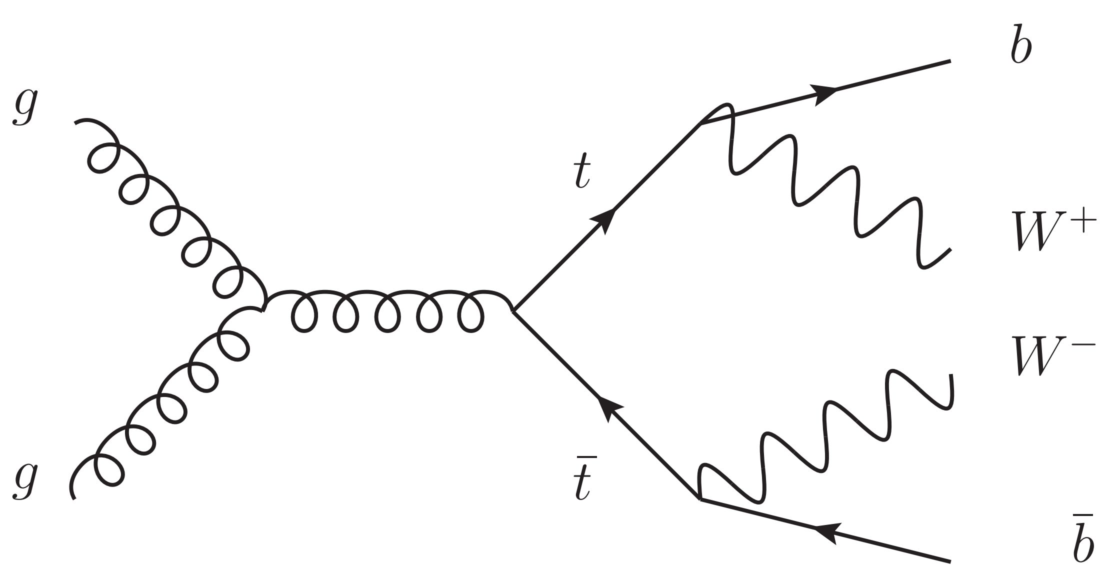
Baldi, Sadowski,Whiteson
arxiv:1402.4735



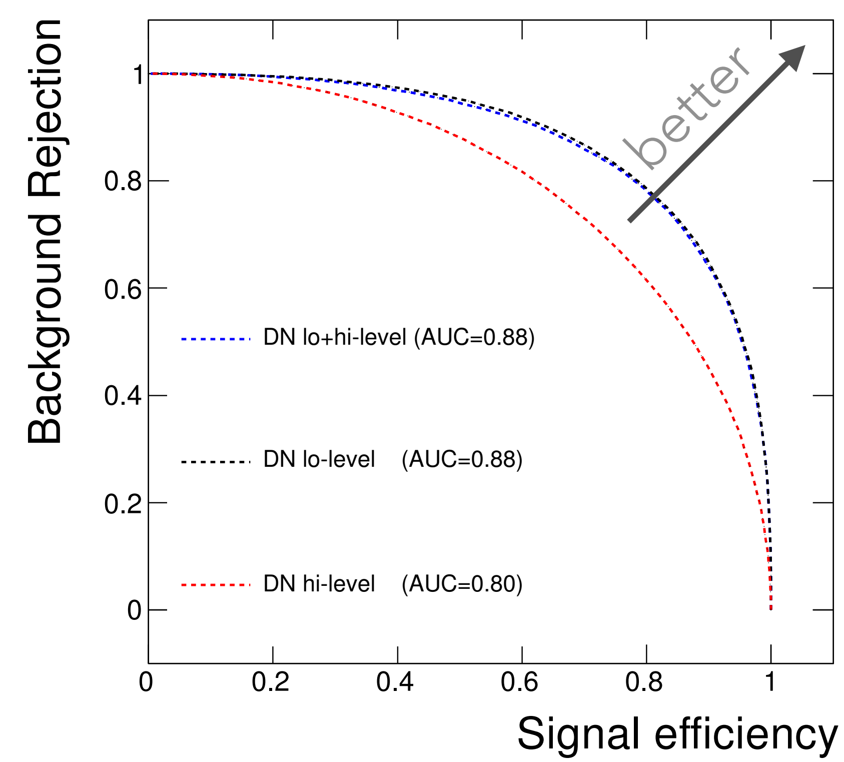
(a)



(a)



(b)



Slide from Kyle Cranmer

Opening the black box of neural nets: case studies in stop/top discrimination

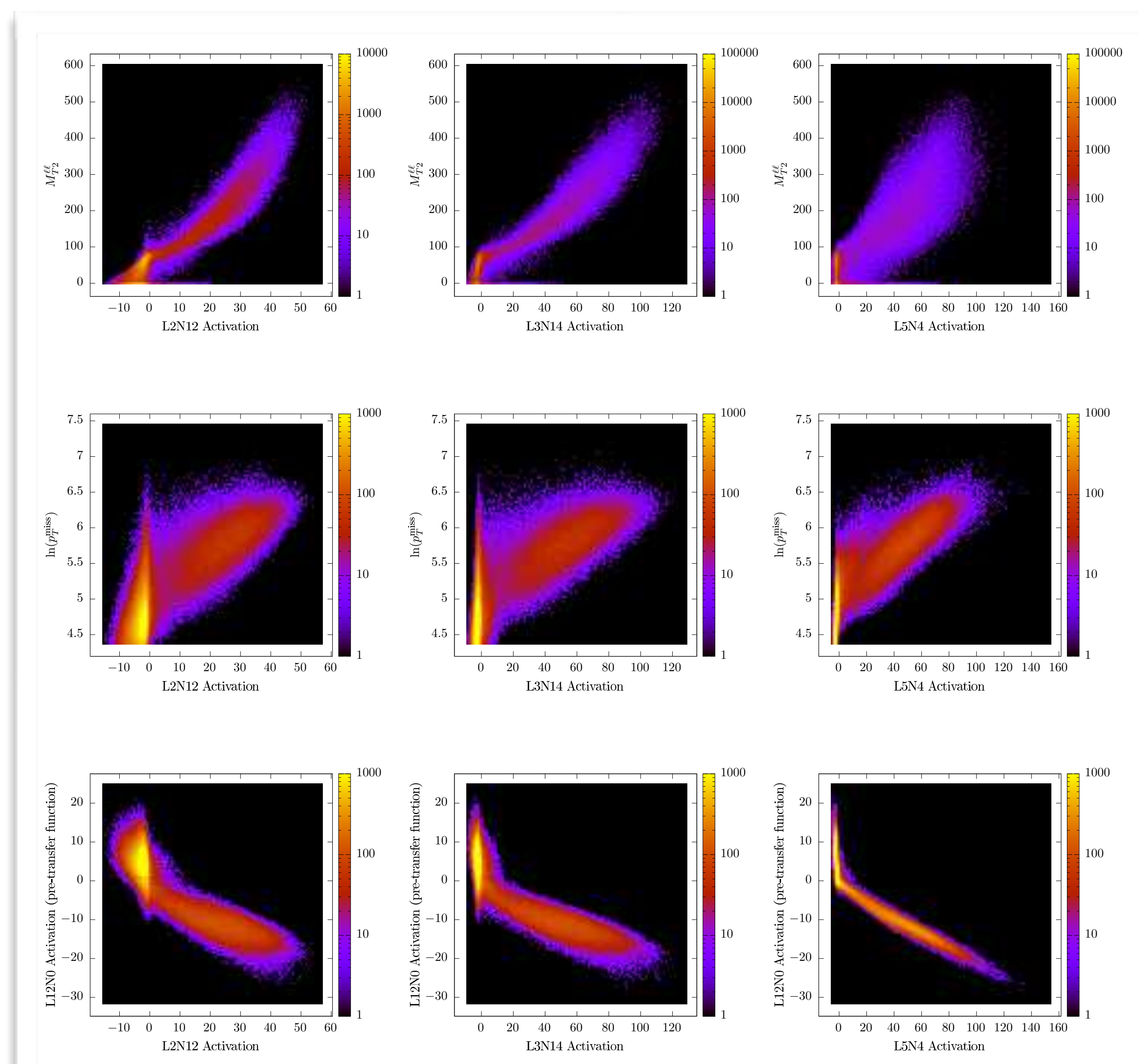
Thomas Roxlo and Matthew Reece
Department of Physics, Harvard University, Cambridge, MA, 02138

April 26, 2018

Abstract

We introduce techniques for exploring the functionality of a neural network and extracting simple, human-readable approximations to its performance. By performing gradient ascent on the input space of the network, we are able to produce large populations of artificial events which strongly excite a given classifier. By studying the populations of these events, we then directly produce what are essentially contour maps of the network's classification function. Combined with a suite of tools for identifying the input dimensions deemed most important by the network, we can utilize these maps to efficiently interpret the dominant criteria by which the network makes its classification.

As a test case, we study networks trained to discriminate supersymmetric stop production in the dilepton channel from Standard Model backgrounds. In the case of a heavy stop decaying to a light neutralino, we find individual neurons with large mutual information with $m_{T2}^{\ell\ell}$, a human-designed variable for optimizing the analysis. The network selects events with significant missing p_T oriented azimuthally away from both leptons, efficiently rejecting $t\bar{t}$ background. In the case of a light stop with three-body decays to $Wb\tilde{\chi}$ and little phase space, we find neurons that smoothly interpolate between a similar top-rejection strategy and an ISR-tagging strategy allowing for more missing momentum. We also find that a neural network trained on a stealth stop parameter point learns novel angular correlations.



https://arxiv.org/abs/1804.09278

Observations

- Given sufficient training data
    - DNNs learn features
    - Provide maximal signal vs background separation.
- In principle, no need for extended feature studies and optimization.
- In practice, gains wrt existing analyses (e.g. using BDTs) often not observed or negligible.
    - My guess: signal train sample size.
- Small data sets
    - Transfer learning
    - Better architectures

Proposal

Goal

- Develop techniques to find New Physics (a.k.a. Beyond Standard Model) without specifying the New Physics at LHC, HL-LHC, HE-LHC, …
- Address problems of
  - insufficient data for training.
  - consistency between analyses / experiments.
- Setting up the problems enables us to also tackle auxiliary problems
  - Fast Physics Generator Model
- Proposal: factorize problem into
  - Physics: kinematics
  - Detector: systematics (beyond scope here?)

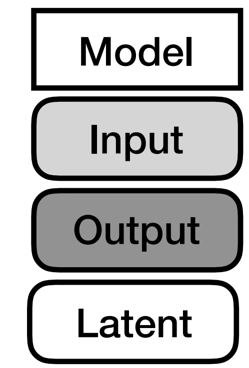
Problem Formulation: Data

- Dataset (Monte Carlo Simulation):
  - Large samples of “background” processes.
    - Signal to background in real data is 1 in 10^{11}.
    - Easily reduced a few orders of magnitude, but generally background >> signal.
  - Signal processes
    - Potentially N “free” physics model parameters, e.g. mass of new particle
    - 2 strategies:
      - One/Few processes: techniques that only look for deviation from Standard Model.
      - Many processes: techniques that attempt to learn features.
  - Levels of realism:
    - Generator: 4-vectors, perfect resolution, quarks not jets, all particles “observed”.
    - Hadronized: turn quarks to jets + apply jet alg. Perfect resolution…
    - Simulated: “smear” 4-vector quantities to sim
- Input:
  - “Raw”: 3-vectors separated by particle type (electron, photon, muon, jet, b-jet, tau) + Missing Energy (2-vector)
  - “Features”: physics motivated functions of 3-vectors. e.g. M_T, M_eff, M_T2, Razor, …
  - Note, these are variable length.

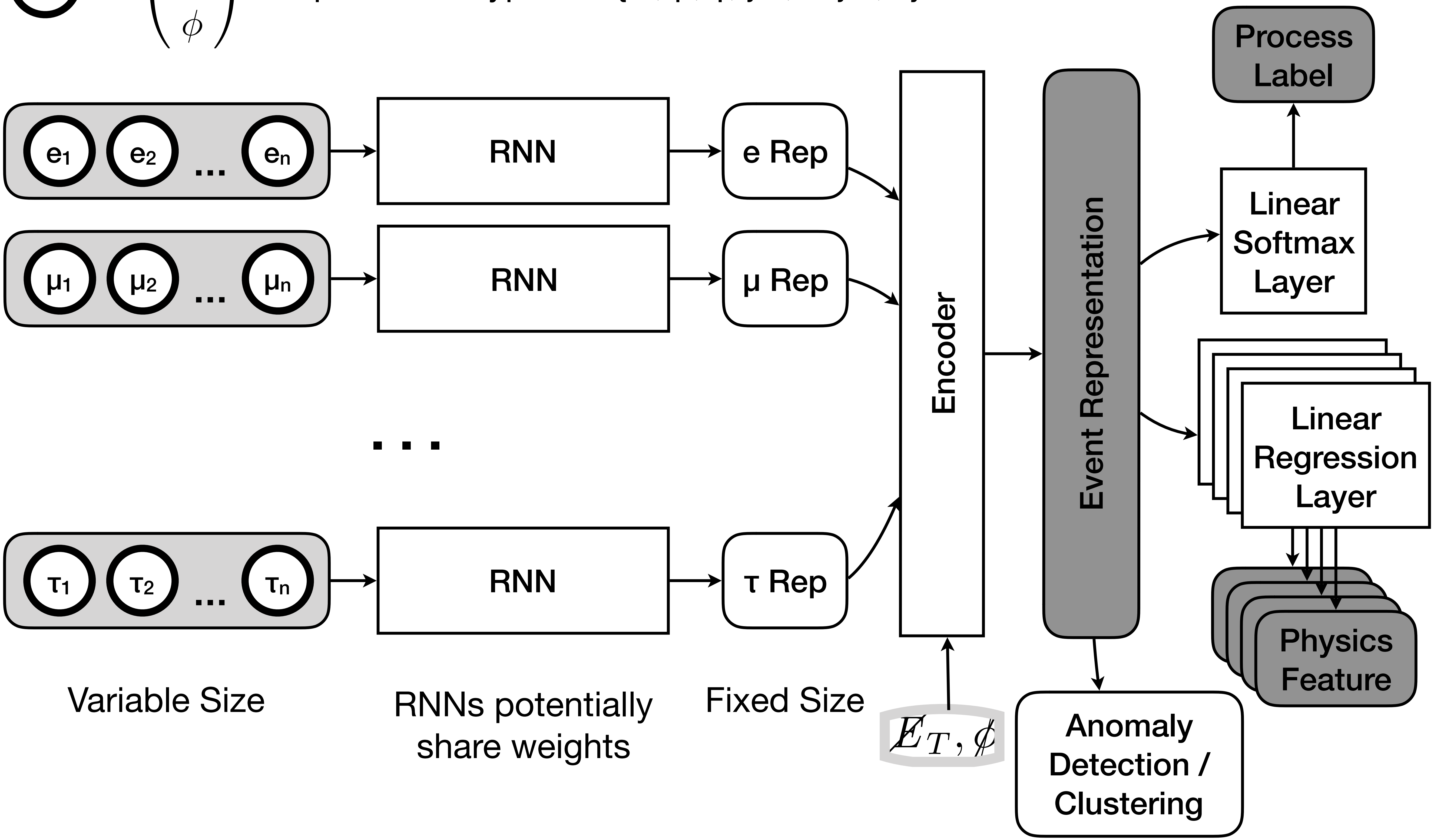
Problem Formulation

- Supervised Classification: We simulate the data, so we have perfect labels (physics process)
  - Baseline: the best an anomaly detection can do …
  - Helps in learning features …
- Unsupervised Classification (clustering):
  - Test if clusters ~ physics processes … “e.g. recognizes SUSY though never told about SUSY”
  - Out of cluster → Anomaly
- Anomaly Detection:
  - never-before-seen process can be detected.
  - 2 proposed paths to compare:
    - Raw → Learned Features → Clustering/Anomaly Detection
    - Raw → Clustering/Anomaly Detection

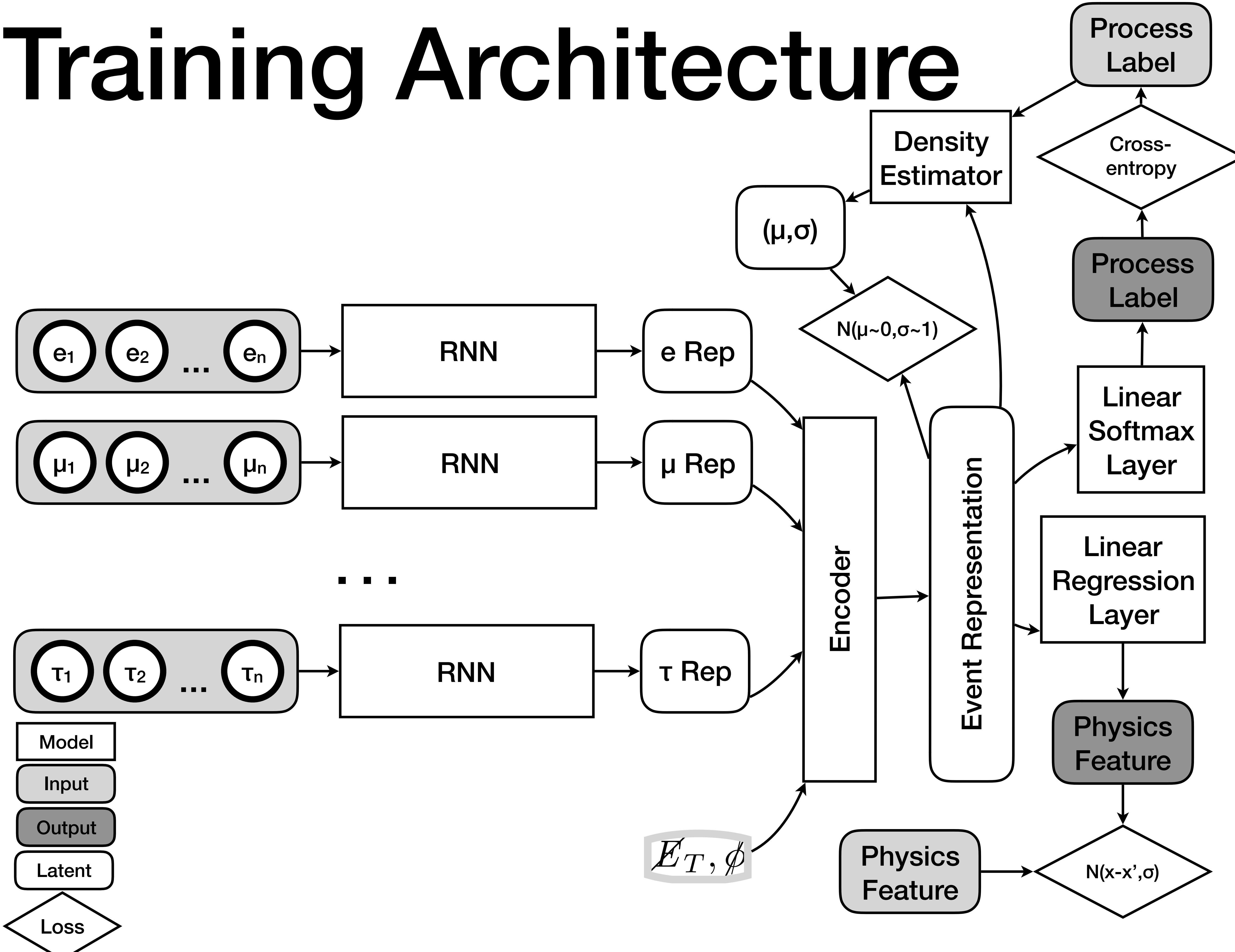
Inference Architecture



$X = \begin{pmatrix} p_T \\ \eta \\ \phi \end{pmatrix}$ for particle of type X = { e, μ, γ, jet, b-jet, τ}

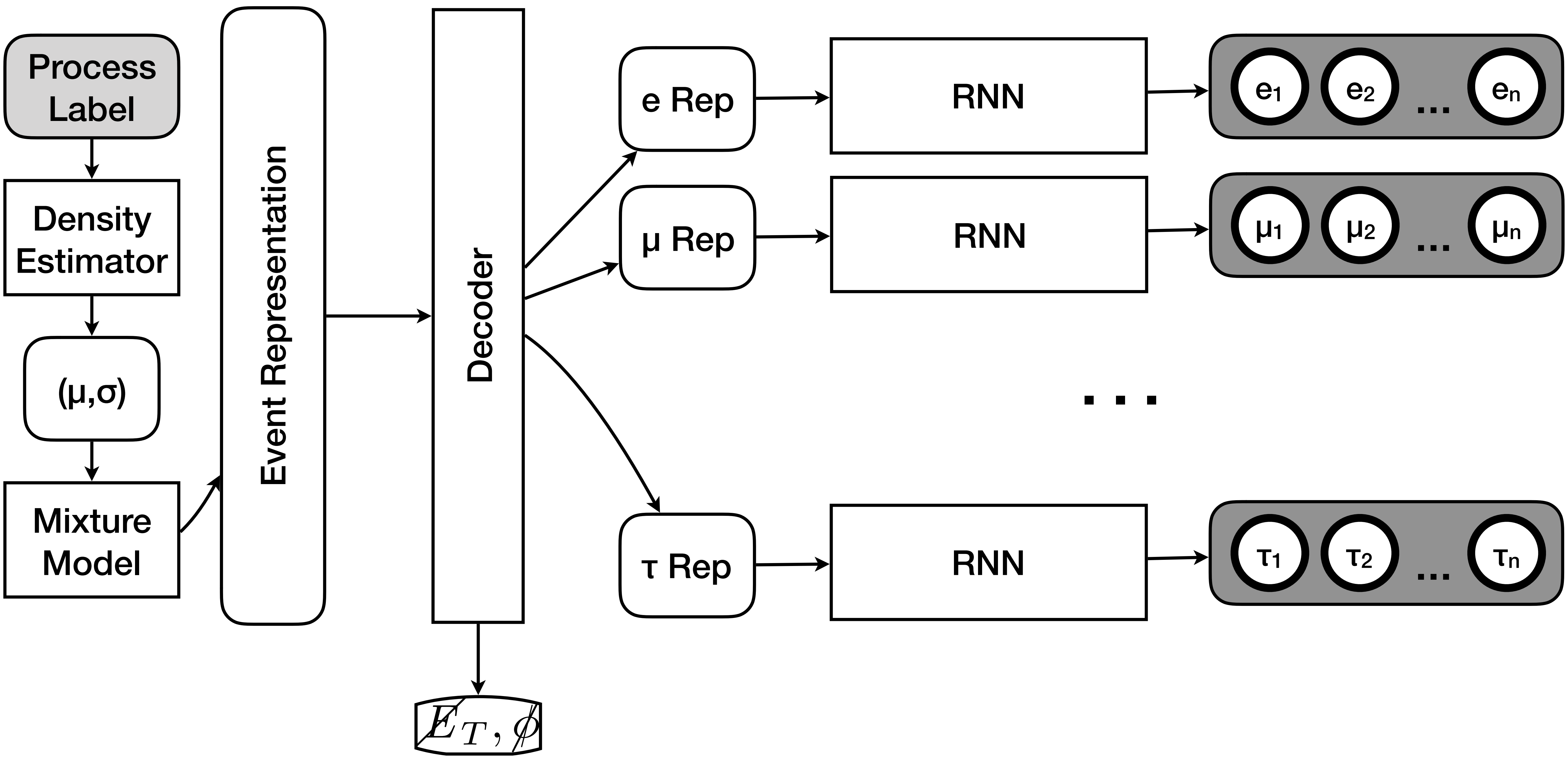


Training Architecture



Generative Architecture

- Basically a Variational Auto-encoder
- Also enables unsupervised feature learning



Clustering & Anomaly Detection

- In both raw (4-vector) and Event Representation space.
- A variety of possible techniques:
    - k-means, …
    - self-similarity test
    - self organizing maps
- Challenge here is computational
    - Most anomaly detect / clustering algs scale poorly.
        - N = Billions

DL Software and Technical Challenges

Basic DL Workflow

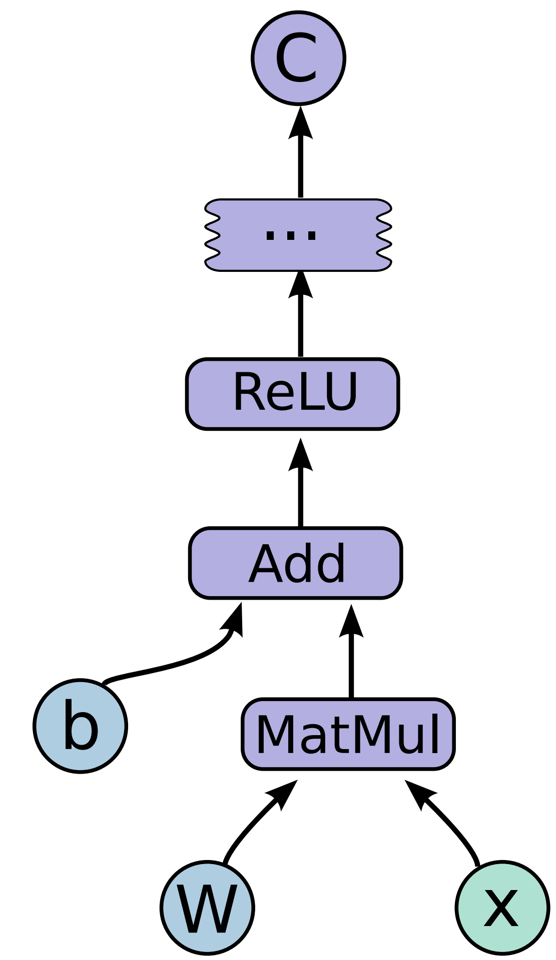
- Prepare data- 80% of the work…
- Build Model
- Define Cost/Loss Function
- Run training (most commonly Gradient Decent)
- Assess performance.
- Run lots of experiments…

numpy, Theano, Keras

- Numpy
    - Provides a tensor representation.
    - It's interface has been adopted by everyone.
        - e.g. HDF5, Then, TensorFlow, … all have their own tensors.
        - You can use other tensors, for the most part interchangeably with numpy.
    - Provides extensive library of tensor operations.
        - D = A * B + C, immediately computes the product of A and B matrices, and then computes the sum with C.
- Theano (TensorFlow)
    - Allows you write tensor expressions symbolically.
        - A * B + C is an expression.
    - Compiles the expression into fast executing code on CPU/GPU: F(A,B,C)
    - You apply the Compiled function to data get at a result.
        - D=F(A,B,C)
- Keras
    - Neutral Networks can be written as a Tensor mathematical expression.
    - Keras writes the expression for you.

DNN Software

- Common features of modern DL Frameworks:
    - Everything build by building mathematical expression for Model, Loss, Training from primitive ops on Tensors
    - Auto-differentiation: Symbolic derivatives for the Gradient Decent
- 2 Classes of DNN Software:
    - Hep-Framework-Like: e.g. PyTorch, Torch, Caffe, …
        - C++ Layers (i.e. Algorithms) steered/configured via interpreted script.
        - Allows dynamic network construction…
        - Faster Research Workflow iteration.
    - General Computation Frameworks: Theano and TensorFlow
        - Builds Directed Acyclic Graph of the computation, performs optimizations
        - High-level tools make this look like HEP Frameworks (e.g. pylearn2, Lasagna, Keras, …)
        - Optimizes for Production Workflow
    - In practice performance is almost identical because majority of time spent in GPU computation which use same libraries.
- Convergence:
    - PyTorch: _____ Mode
    - TensorFlow: Eager Mode

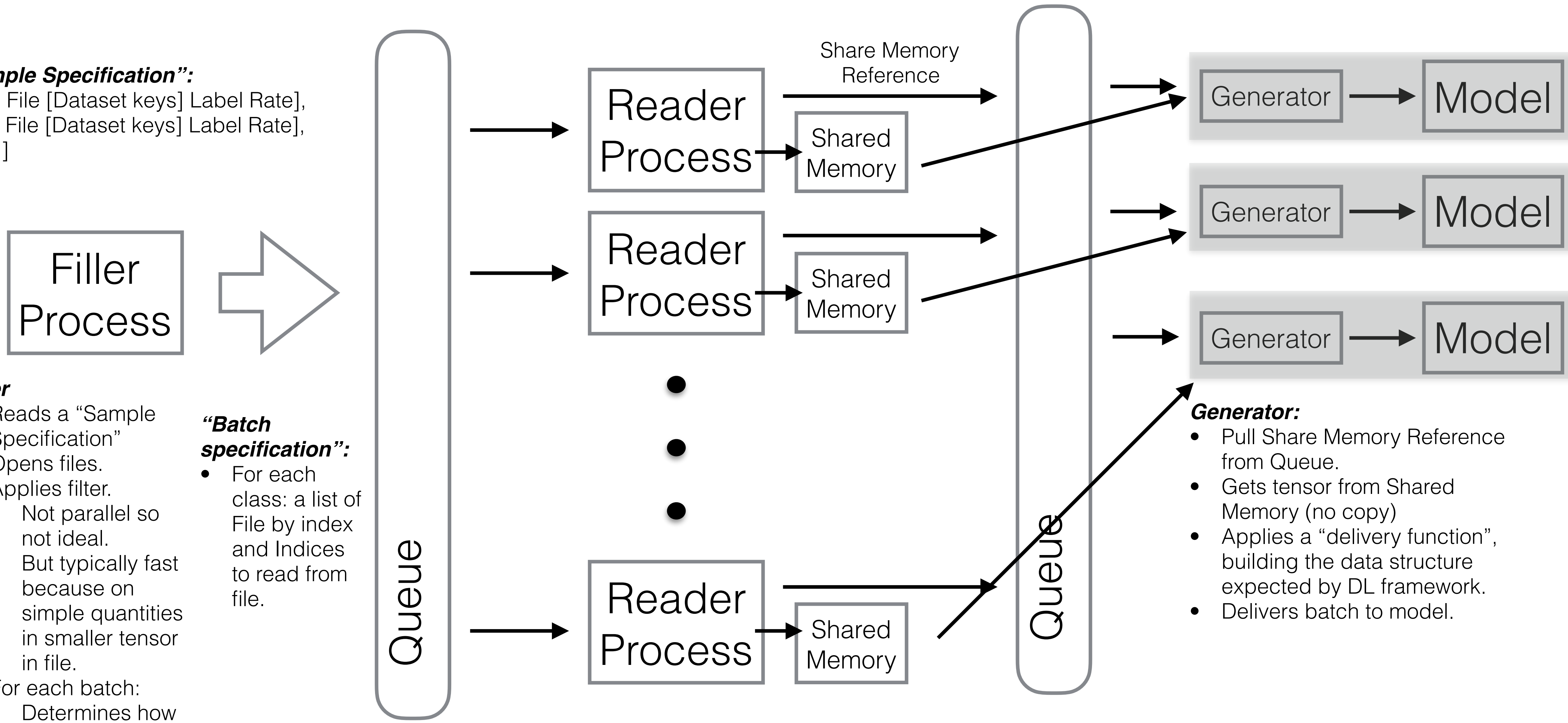


Technical Challenges

- Typically in HEP:
  - Datasets are too large to fit in memory.
  - Data comes as many files, each containing O(1000) events, organized into directories by particle type.
  - Potentially O(10000) processes ~ classes… hard to book-keep.
  - For training, data needs to be read, mixed, “labeled”, possibly augmented, and normalized…. can be time consuming.
- Very difficult to keep the GPU fed with data. GPU utilization often < 10%, rarely > 50%.
- Solutions:
  - Keras python multi-process generator mechanism has limitations…
  - PyTorch and TensorFlow very recently added parallel ETL (Extract, Transform, Load) pipelines
    - While significantly simplifiled, still can require custom code and hand tuning.
    - Performance sub-par.

Data Providers

"Sample Specification":
[[File [Dataset keys] Label Rate],
[File [Dataset keys] Label Rate],
…]



Filler
- Reads a "Sample Specification"
- Opens files.
- Applies filter.
  - Not parallel so not ideal.
  - But typically fast because on simple quantities in smaller tensor in file.
- For each batch:
  - Determines how many events to read from each file.
- Generates a "Batch Specification"

"Batch specification":
- For each class: a list of File by index and Indices to read from file.

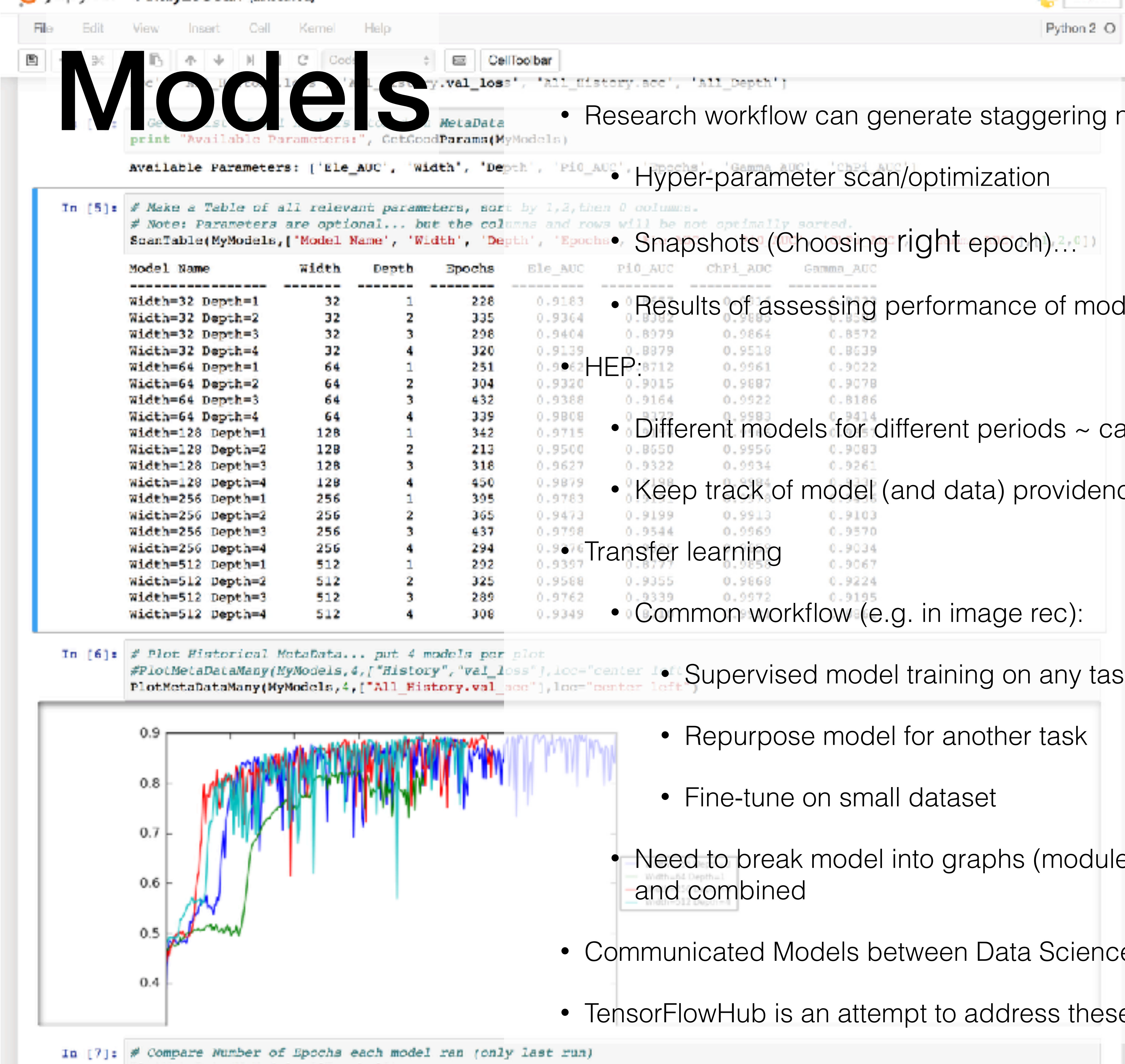
Readers:
- Fetch Data From files.
- Label and shuffle.
- Apply a process function
- May produce completely different tensors.
- Caches File handles

Shared Memory:
- Store Tensors
- Shape known until first batch comes through

Generator:
- Pull Share Memory Reference from Queue.
- Gets tensor from Shared Memory (no copy)
- Applies a "delivery function", building the data structure expected by DL framework.
- Delivers batch to model.

- Processing happens on First Epoch Only!
- Data Cached into file during first epoch.
  - All of the processing only needs to be done once.
- Automatically use another instance of data providers to read cache file for all other epochs.
- Note that for now, we have both train and test data providers… better use of resources if we merge.

Models



- Research workflow can generate staggering number of models
  - Hyper-parameter scan/optimization
  - Snapshots (Choosing right epoch)…
  - Results of assessing performance of models
- HEP:
  - Different models for different periods ~ calibrations in conditions
  - Keep track of model (and data) providence
- Transfer learning
  - Common workflow (e.g. in image rec):
    - Supervised model training on any task on large dataset.
    - Repurpose model for another task
    - Fine-tune on small dataset
  - Need to break model into graphs (modules) that can be repurposed and combined
- Communicated Models between Data Science and Production Teams
- TensorFlowHub is an attempt to address these issues

ML2.0?

Slide from Kyle Cranmer

How do these fit together?

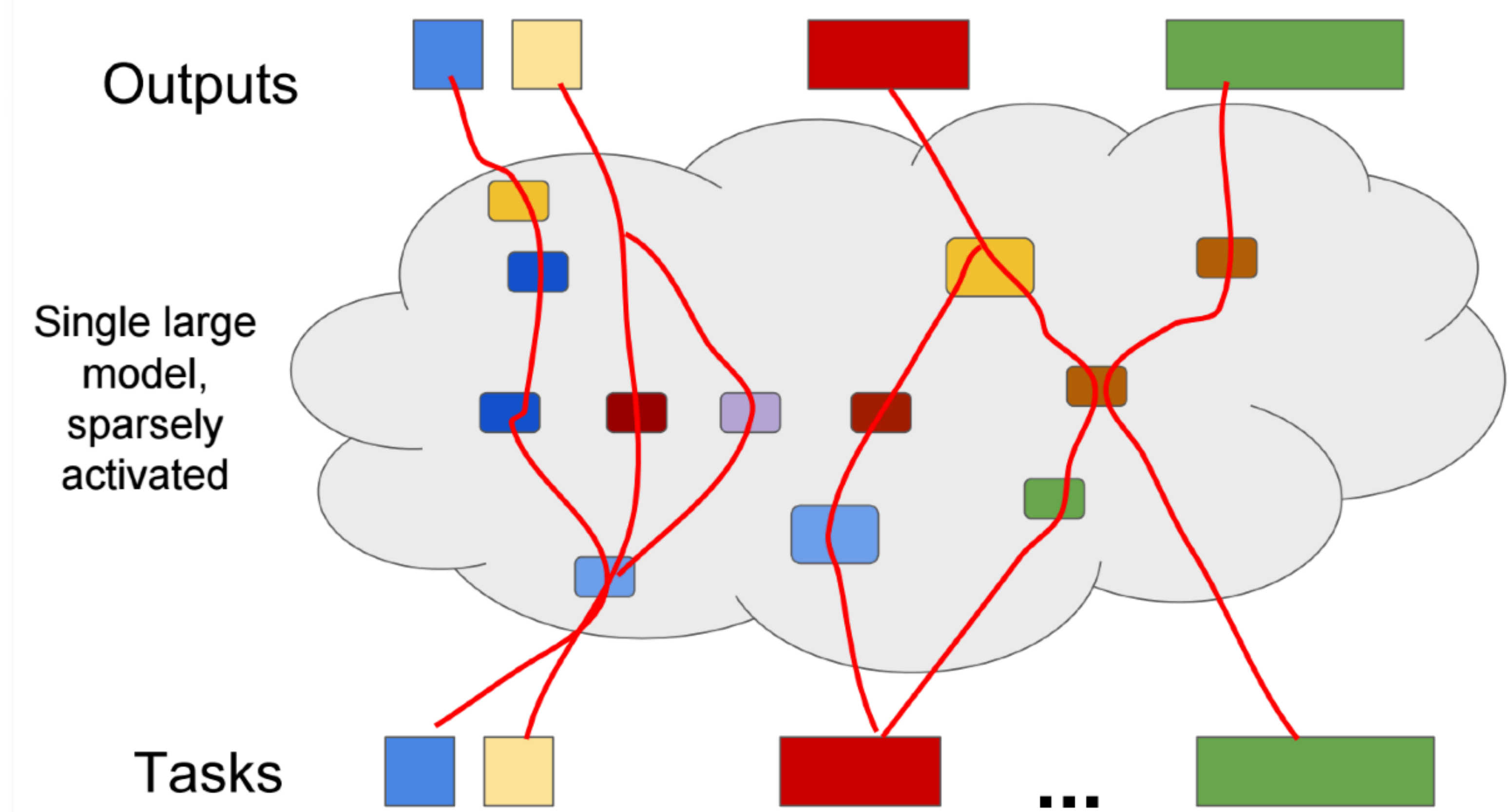
Combine many of these ideas:

Large model, but **sparsely activated**
Single model to **solve many tasks** (100s to 1Ms)
Dynamically learn and **grow pathways** through large model
Hardware **specialized for ML supercomputing**
ML for efficient mapping onto this hardware

Google

Kyunghyun Cho
July 10 ·

ML 2.0 at Google



Slides from Jeff Dean of Google Brain @ Jeju last week

https://drive.google.com/file/d/0B8z5oUpB2DysZWF1RTFuX1NEZUk/view

Future

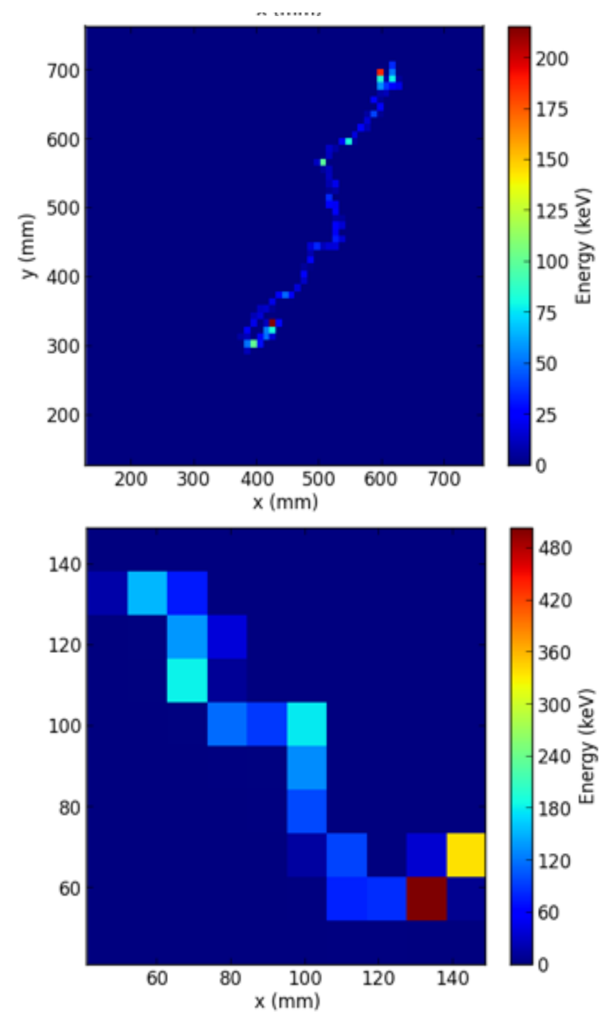
DL Based Reco

- Immediate uses:
  - “Imaging” detectors likely path:
    1. Improved classification/regression with Convolutional NNs.
    2. Fast Showers with Generative models.
    3. Feature (particle) extraction with Regional NN and semantic segmentation.
    4. Full event classification
       - Recurrent networks
       - Reinforcement training… turn reconstruction into a board game.
  - Help with detector optimization:
    - DL provides easily obtainable, consistent, and probably optimal metrics.
    - Just simulate… no need to build reco tuned to every possibility.
    - Understand the fundamental limits but turning on physics / detector effects one by one in simulation.

NEXT Detector Optimization

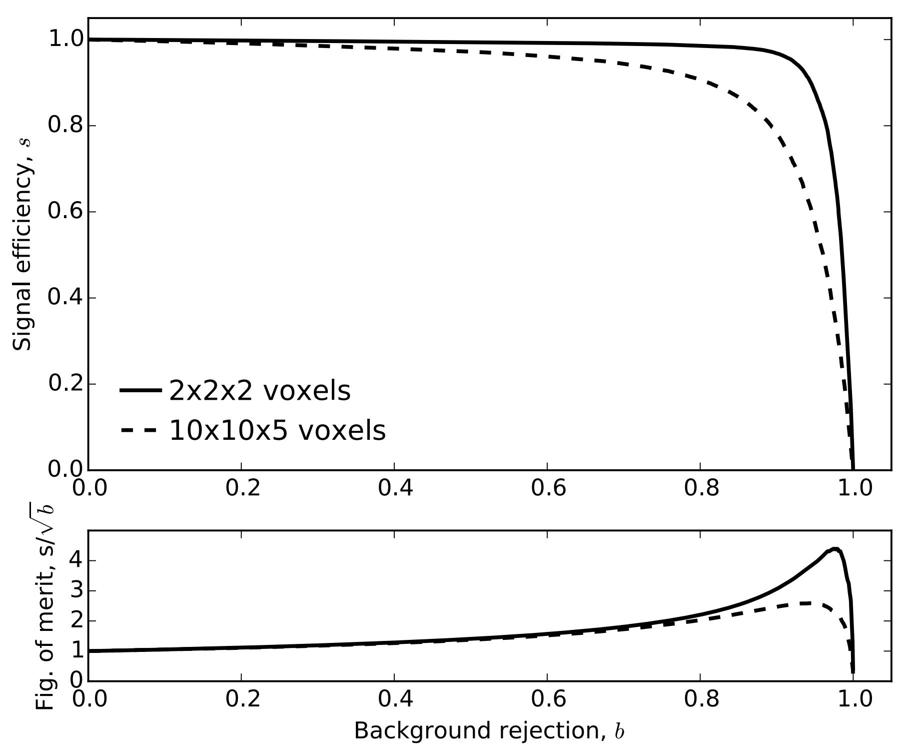
- Idea 1: use DNNs to ***optimize detector***.
  - Simulate data at different resolutions
  - Use DNN to quickly/easily assess best performance for given resolution.

Analysis	Signal eff. (%)	B.G. accepted (%)
DNN analysis (2 x 2 x 2 voxels)	86.2	4.7
Conventional analysis (2 x 2 x 2 voxels)	86.2	7.6
DNN analysis (10 x 10 x 5 voxels)	76.6	9.4
Conventional analysis (10 x 10 x 5 voxels)	76.6	11.0



- Idea 2: ***systematically study*** the relative importance of various physics/detector effects.
  - Start with simplified simulation. Use DNN to assess performance.
  - Turn on effects one-by-one.

2x2x2 voxels	Run description	Avg. accuracy (%)
	Toy MC, ideal	99.8
	Toy MC, realistic $0\nu\beta\beta$ distribution	98.9
	Xe box GEANT4, no secondaries, no E-fluctuations	98.3
	Xe box GEANT4, no secondaries, no E-fluctuations, no brem.	98.3
	Toy MC, realistic $0\nu\beta\beta$ distribution, double multiple scattering	97.8
	Xe box GEANT4, no secondaries	94.6
	Xe box GEANT4, no E-fluctuations	93.0
	Xe box, no brem.	92.4
	Xe box, all physics	92.1
	NEXT-100 GEANT4	91.6
10x10x5 voxels		
	NEXT-100 GEANT4	84.5



DNN+HEP Software Needs (1/4)

1. *Inference in HEP Frameworks:*
    - Need optimized and validated inference implementation.
        - This problem is mostly addressed…
            - ATLAS: Lightweight DNN Inference Framework
            - CMS: TensorFlow integration into CMS-SW
    - DNN weights can be Gigabytes, likely need
        - Condition DB-like systems storage.
        - Memory sharing between processes/threads.
    - I can imagine a DL service similar to ATLAS APE GPU service:
        - Processes are client of server(s) that talk to backends/accelerators.
        - No reason for every experiment to reinvent the wheel here…

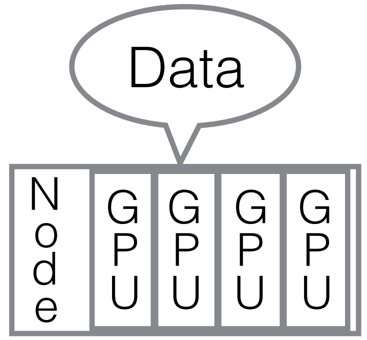
DNN+HEP Software Needs (2/4)

2. Training systems:

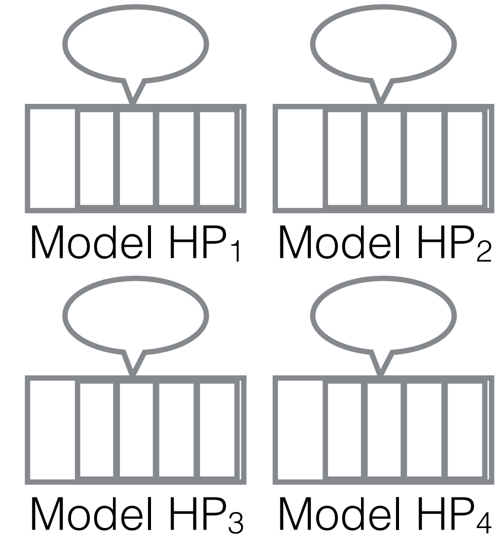
- Training DNNs efficiently generally requires GPUs (or other future accelerators).
- Hyper-parameter scans / optimization critical part of DNN development workflow.
    - Great use of GPUs on HPCs.
    - Google and other clouds specifically target DL.
- Today's training samples can already be 10s of Terabytes, requiring massive parallelism.
    - *Data Parallelism*: Bottlenecked by gradient syncing between GPUs or systems. Lots of Engineering in Industry already. And some HEP solutions…
    - *Model Parallelism*: Less sync'ing but only makes sense for large enough model.
    - No more embarrassingly parallel. Must provision large number of machines.
- As DNNs become essential, training them becomes part of software releases, simulation, reco,… cycle.
    - New simulation/reco can require regenerating large training sets (various conditions) and running long training before using reco.
    - Somewhat analogous to calibration on express streams.
- I can imagine Workflow and Data Management systems designed for DL training workflows on any available resource.

Parallelism

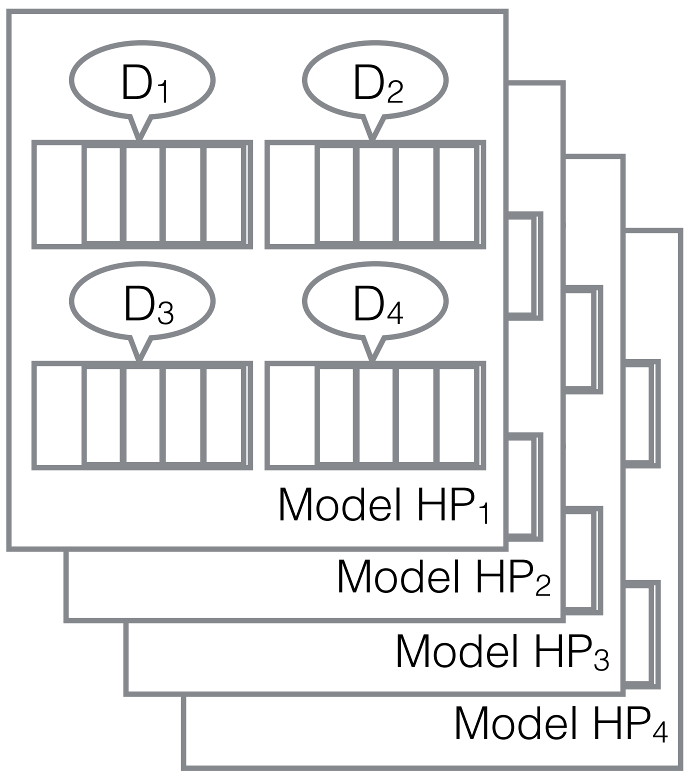
1. *Tensor operation parallelism*: GPUs, FPGA, and ASICs (Google's Tensor Processing Unit).
    - Note additional HN, Data, Model parallelism with multi-GPU

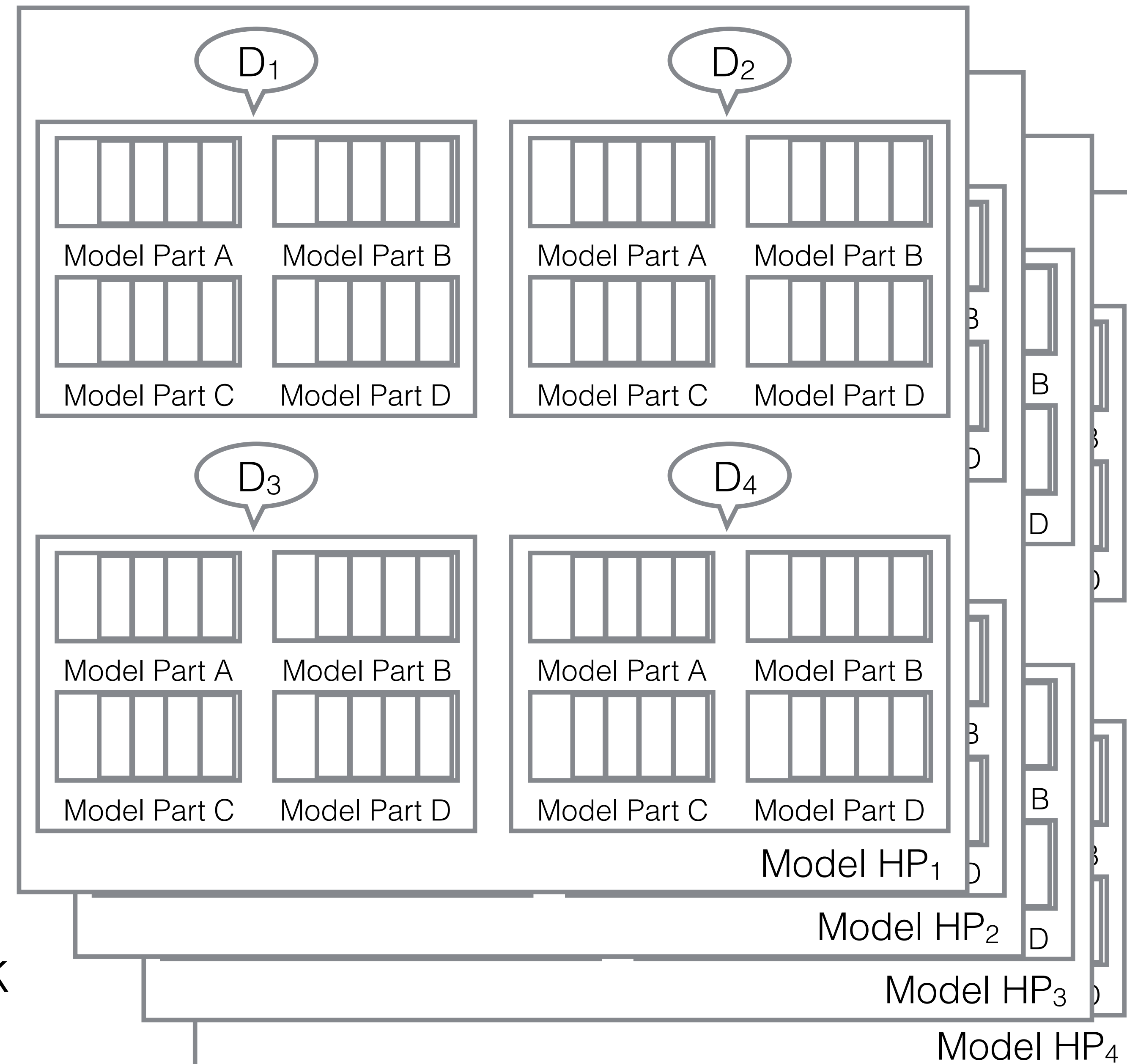


2. *Hyper-parameter scan*: simultaneously train multiple models. *e.g.* 1 model per GPU or node.



3. *Data Parallelism*: Each GPU or Node computes gradient on sub-set of data. Sync'ing gradients bottlenecked by bus or network.





4. *Model Parallelism*: Large model spread over many GPUs or nodes. Less network traffic but only efficient for large models.

DNN+HEP Software Needs (3/4)

3. Training Datasets…

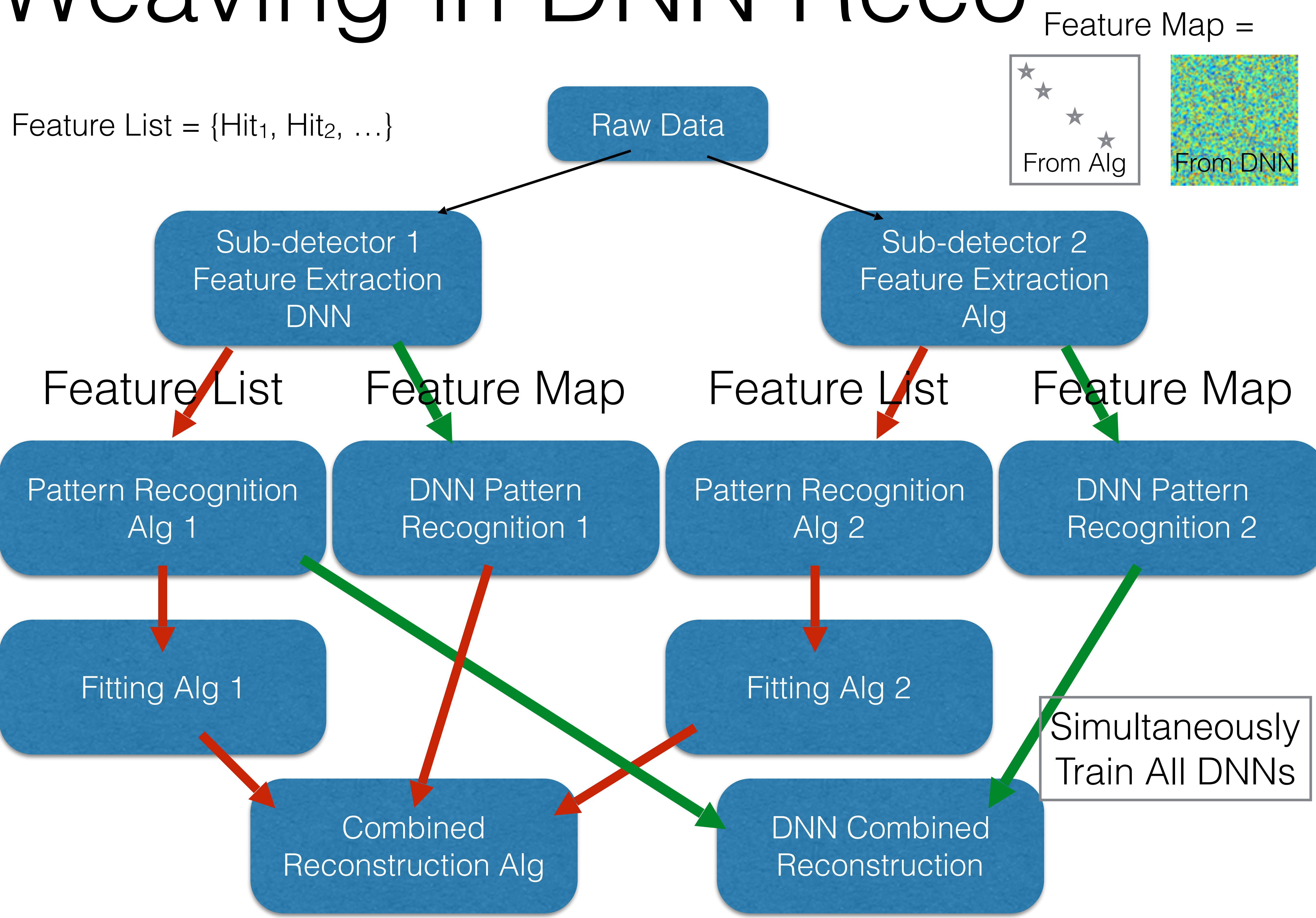
- DL generally requires huge independent simulated training samples.
  - But HEP Experimental data is private, making collaboration and rapid publication difficult.
  - Reconstruction DNNs will likely require Geant4. (i.e. CPU intensive)
- Collaboration with Machine Learning experts and among experiments require public data sets.
  - Publicly available simulation and reconstruction (for base-line).
    - Some are now available…
  - Need to store and distribute large data-sets to public.
    - CERN Open Data?

DNN+HEP Software Needs (4/4)

4. *Event Processing within Deep Learning Frameworks*

- DL will potentially become integral to our software and trigger
    - We may replace code with weights.
    - DL integrated into HEP frameworks. Not just an external. (example next slide)
- Many-core/FPGA/neuro-morphic accelerators may prolong Moore's law
    - Experiments like DUNE will run for 30 years and must keep up with emerging tech.
    - Frameworks must [automatically] optimize and place computations on a variety of rapidly evolving hardware and software.
    - May need to distribute processing of individual events across cluster (like HEP trigger)
        - Use network hardware for primitive operations during transfers.
        - Partially process on specialized machines (specific accelerators, HPC, massive memory, …)
- Industry will highly optimize DL systems and provide services around them.

Weaving-in DNN Reco



R&D Proposal

- *Premise*: We need new frameworks to take advantage of DL and emerging architectures.
    - ➡ Build HEP Framework on top of a DL Framework.
- If we envision new frameworks need to do R&D now, ver 1.0 by 2020, deployed by 2025.
- R&D Proposal (can we do traditional HEP Reco in DL Framework?):
    - Build HEP Reco on top of Google's OpenSource TensorFlow
        - General computation system, based on Directed Acyclic Graphs.
            - Framework for Automatic optimizations (like Theano), though currently primitive.
        - Supports all architectures and distributes computation across GPUs and clusters.
        - Build a HEP Framework in python (like Keras) with C++ wrapped in TF ops.
    - 3 project ideas:
        - First steps of LArTPC reco: deconvolution, hit finding, …
        - Online Sparsification and compression of LArTPC data for protoDUNEs.
        - ATLAS GPU Trigger Demonstrator: Wrap the existing GPU/CPU kernels in TensorFlow Ops.

Science Fiction?

- Imagine in next 10 years DNN lives up to the hype…
  - We've proven DNNs gets us better, faster, easier software… and hardware.
  - Industry investment in DNNs has yielded significant gain over Moore's Law
    - Custom DL/neuromorphic chips and HPCs
    - Software Frameworks
    - Cloud Services
    - Consultants:
      - Data Scientists: DL reduces need for domain-specific expertise (e.g. in biology now).
      - Data Engineers: low level optimization, deployment, operation…
  - Actually, all of these already exist!

- Large portions of HEP code replaced by deep neutral network architecture and weights.
  - HEP Software Frameworks built on top of DL Frameworks.
  - To DL systems, our computing looks like everyone else's… e.g. other sciences.
- Optimization, deployment, operations handled by professional Data Engineers.
- Trigger implemented in custom inference systems built from heterogeneous commodity hardware.
- Computation performed on DL Clouds and scientific HPCs.
- DNNs designed and trained in collaboration with professional Data Scientists.
- HEP PhDs trained/funded by industry to apply DL to HEP and then transition to industry.

Final Thoughts

- Deep Learning can change how science is done.
  - Improve performance. Save time and money.
  - Mitigate stalling of Moore's law.
  - Use most recent hardware.
  - Allow scientists to focus on concepts rather than implementation.
- Over past few years the utility and feasibility of applying DL to solve HEP problems has be established in many areas…
  - Adding realism and moving into production is the next challenge.
  - We can't forget that DL can complicated things:
    - Systematics. Data/MC agreement.
    - Generate large independent training and calibration samples.
    - New complicated "release", production, and analysis cycles/work-flows.
- If we want to be ready for the DL revolution in 10 years, we need to do R&D now.

Jet Physics with Deep Learning

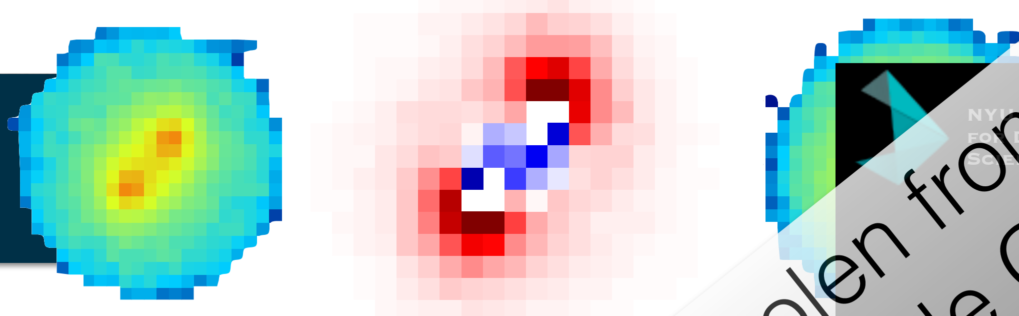
Modern Machine Learning

for Classification, Regression, and Generation in Jet Physics

BERKELEY LAB

Benjamin Nachman

Lawrence Berkeley National Laboratory



CERN Data Science Seminar, Novem

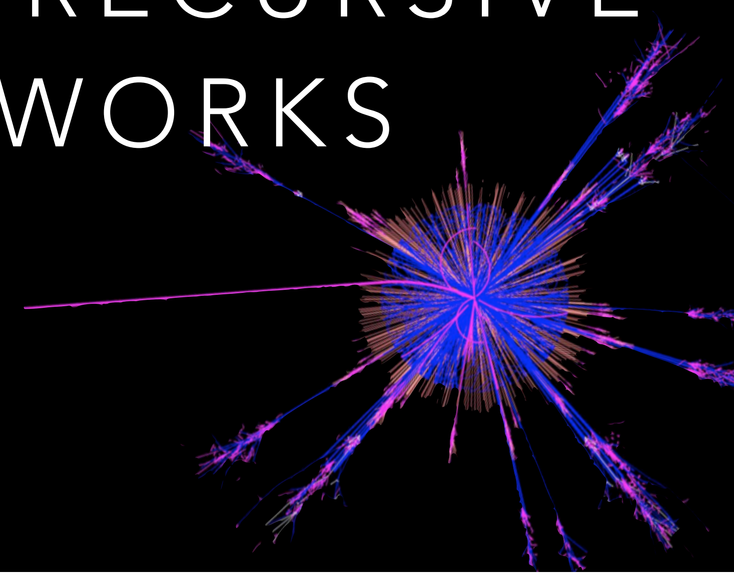
Next slides stolen from Ben Nachmans and Kyle Cranmer's Excellent CERN Seminars

NYU Center for Data Science

Center for Cosmology and Particle Physics

ARXIV:1702.00748

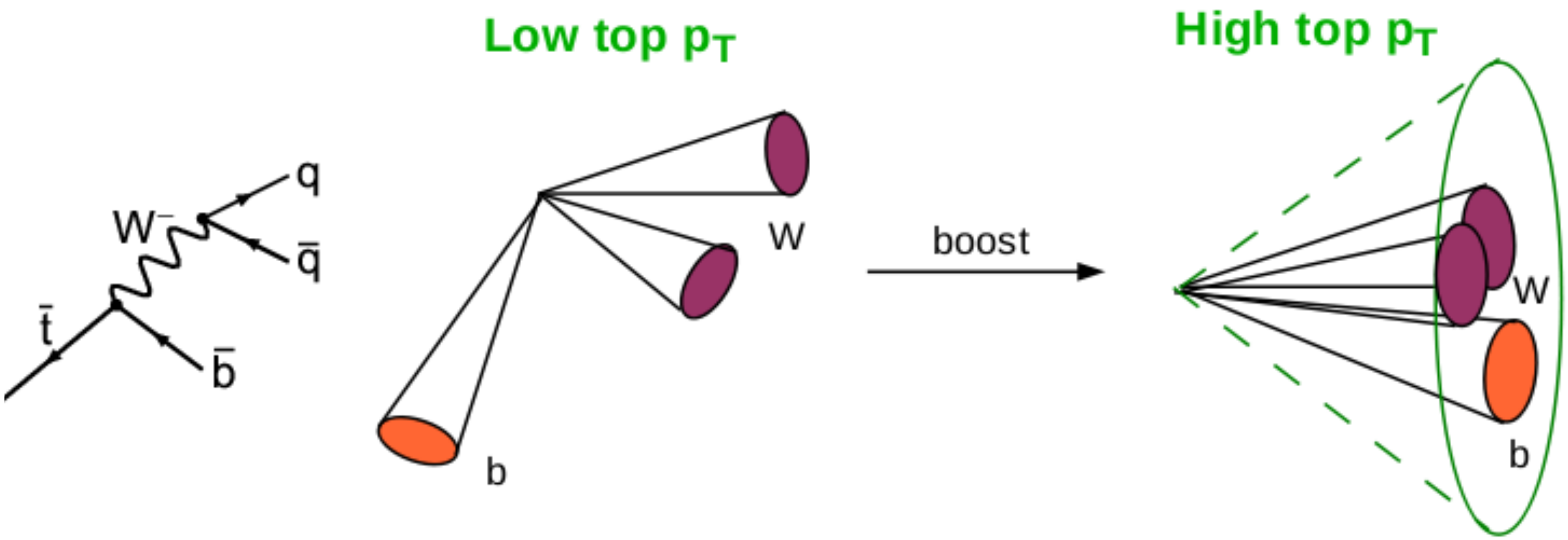
QCD-AWARE RECURSIVE NEURAL NETWORKS



@KyleCranmer
New York University
Department of Physics
Center for Data Science

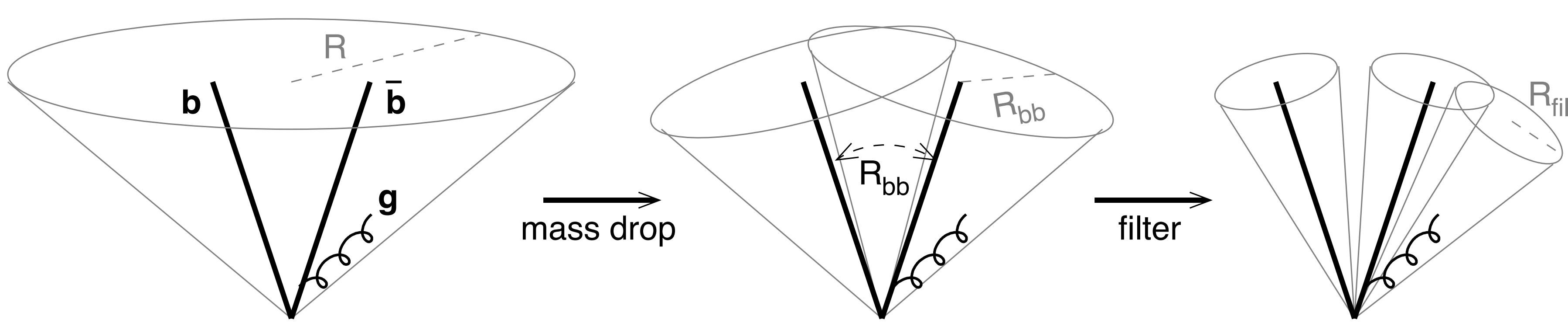
with:
Gilles Louppe
Kyunghyun Cho
Joan Bruna
Cyril Becot

Many scenarios for physics Beyond the Standard Model include highly boosted W, Z, H bosons or top quarks



Identifying these rests on subtle substructure inside jets

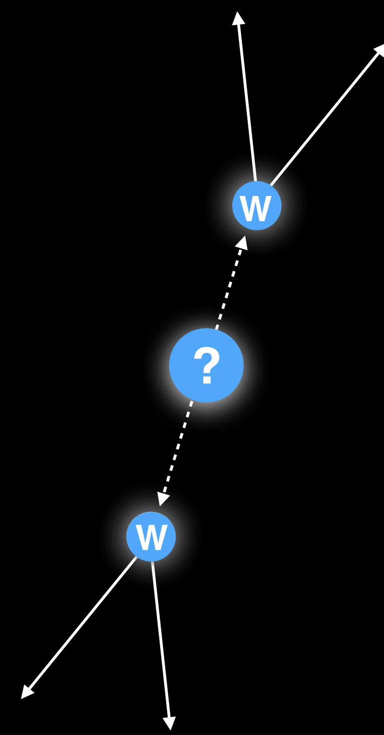
- an enormous number of theoretical effort in developing observables and techniques to tag jets like this



Goal: Find W jets in an enormous sea of generic q/g jets

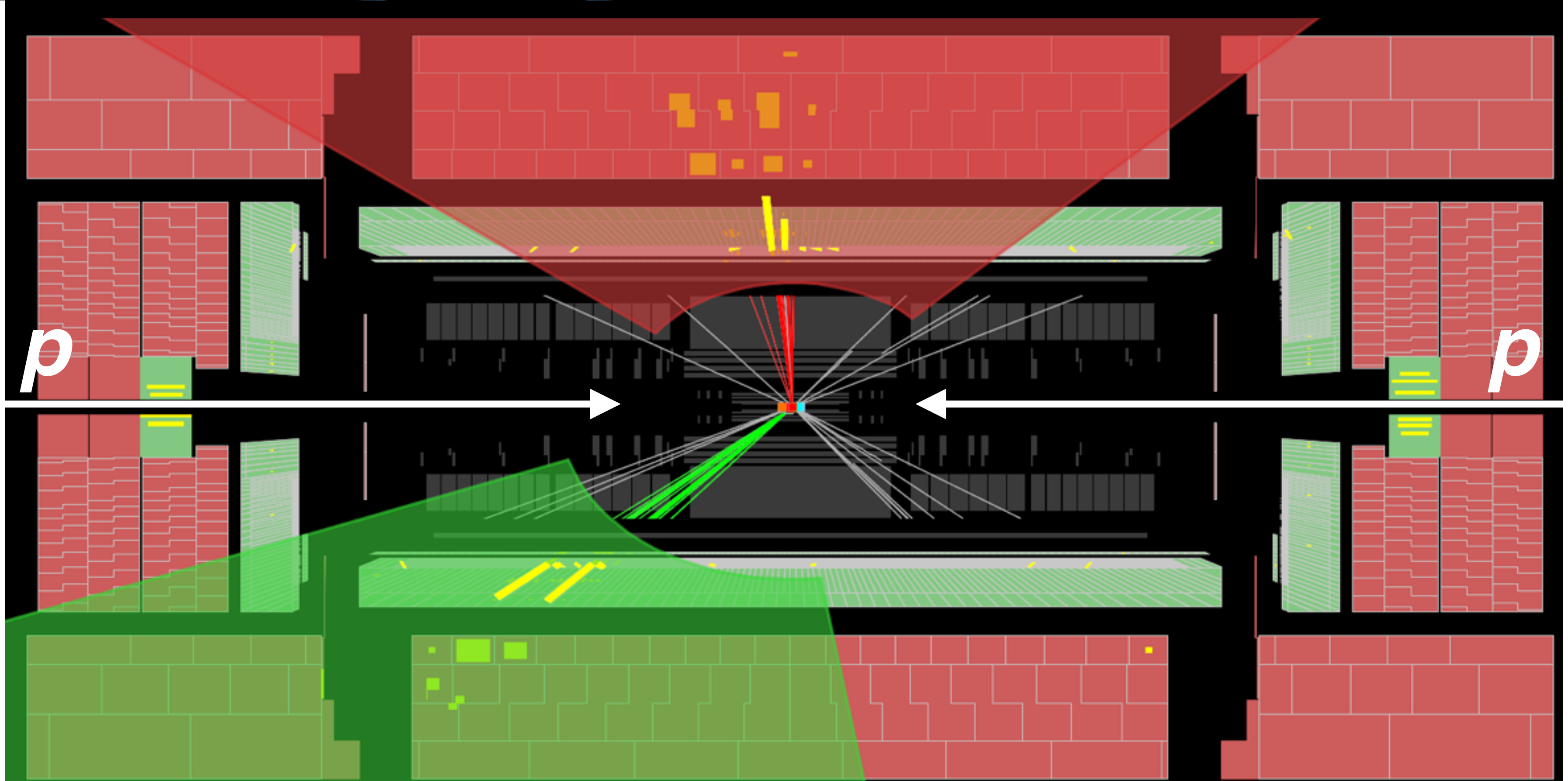
These jets have a non-trivial structure!

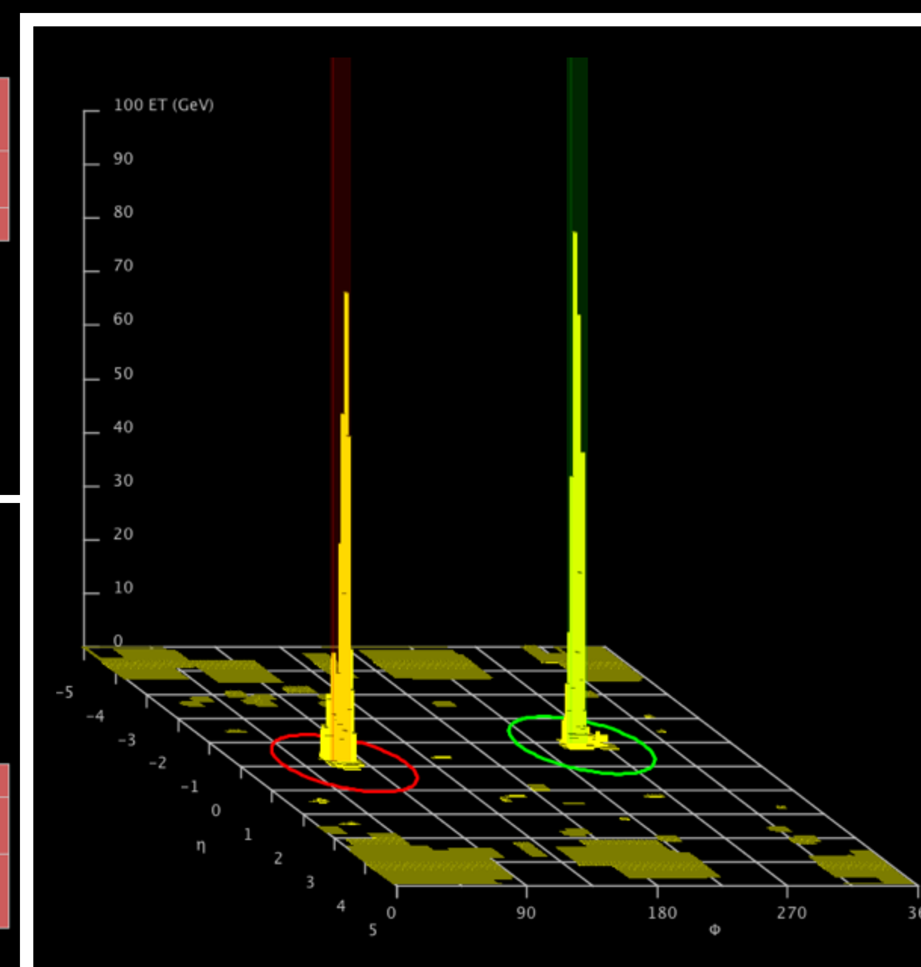
W bosons are naturally boosted if they result from the decay of something even heavier



Searching for new particles decaying into boosted W bosons requires **looking at the radiation pattern inside jets**

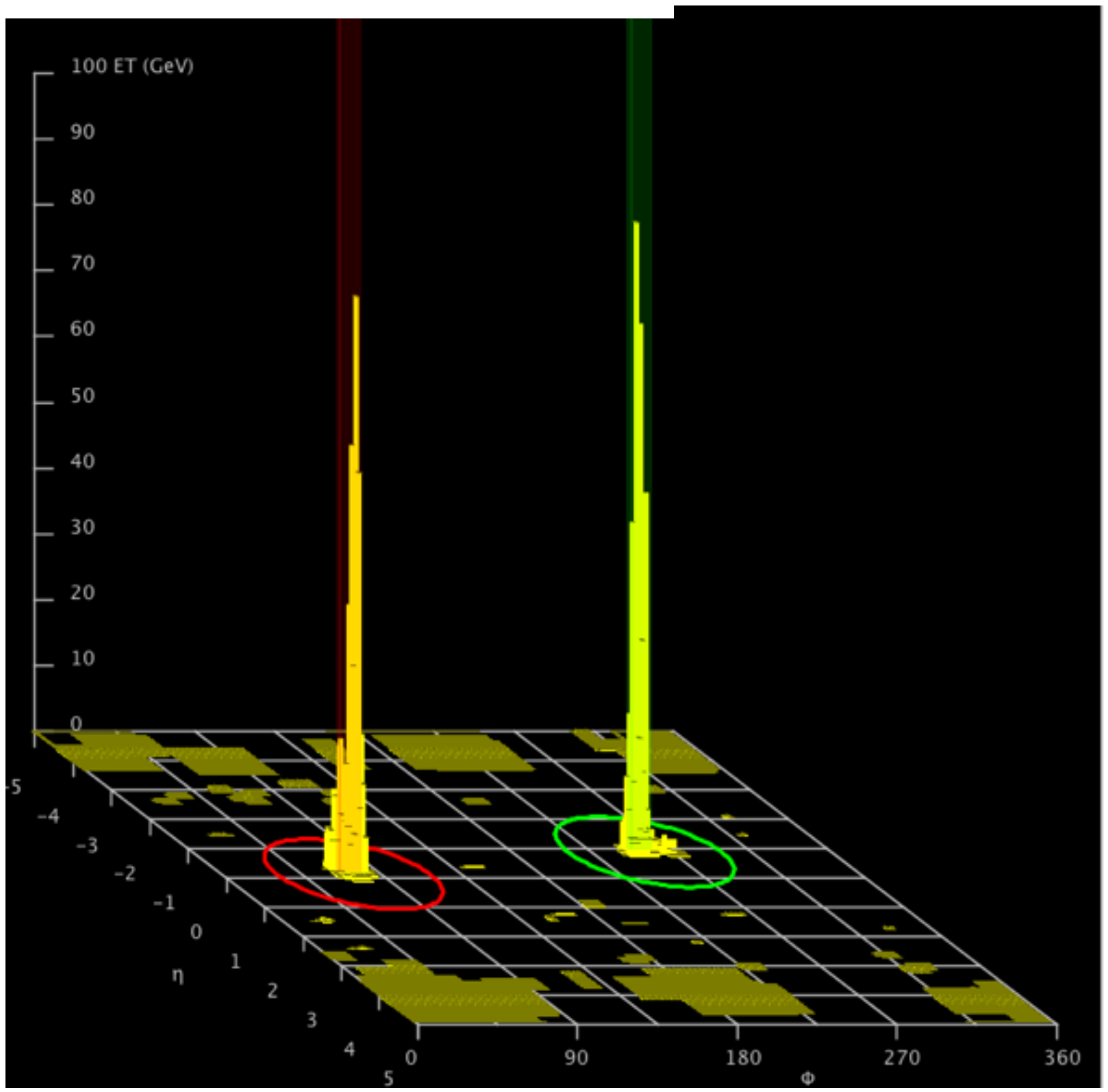
like a digital image!

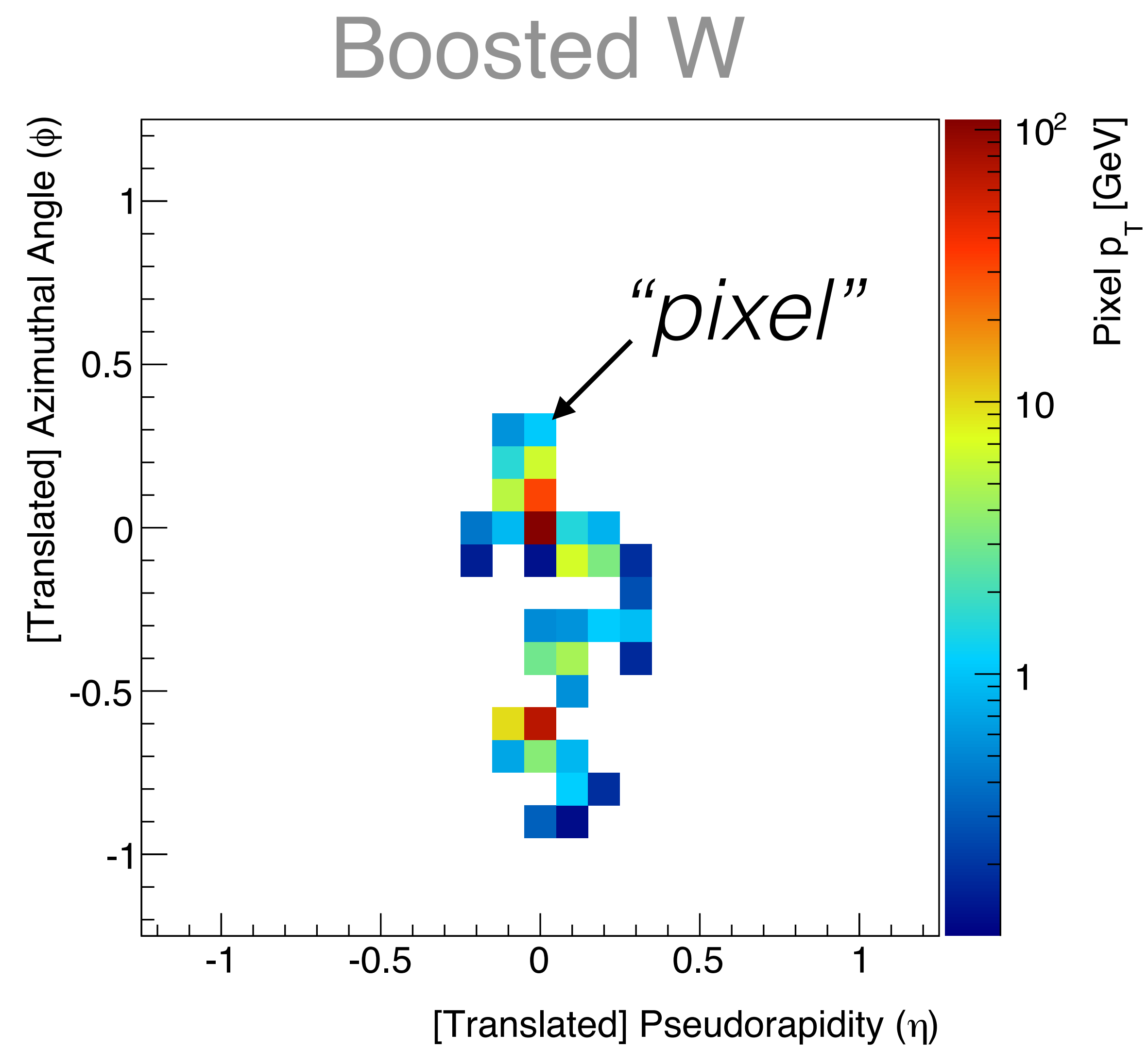




the Jet Image

J. Cogan et al. JHEP 02 (2015) 118



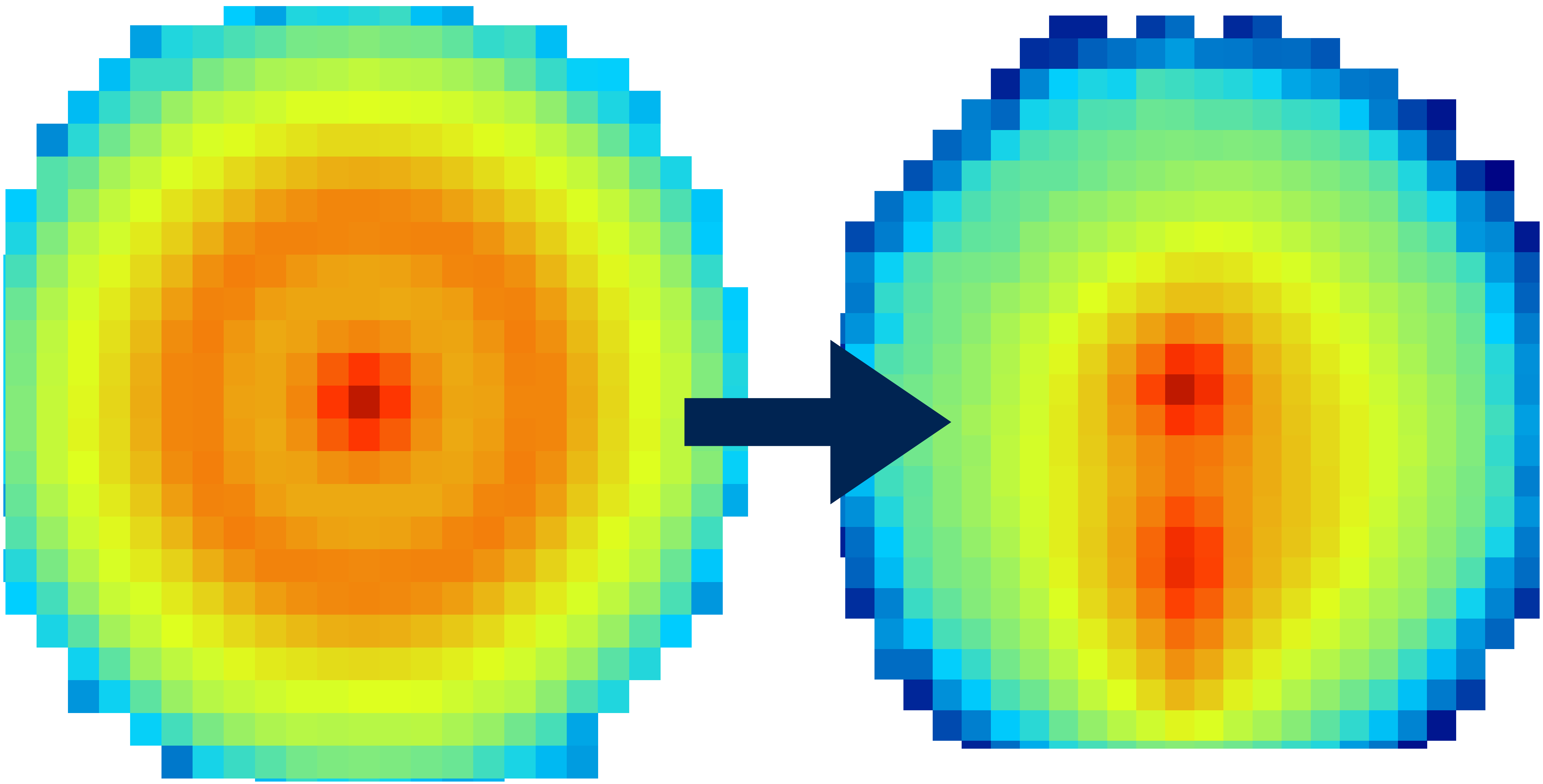


Credit: Peter G Trimming (Wikipedia)

no smooth edges, clear features, low occupancy (number of hit pixels)

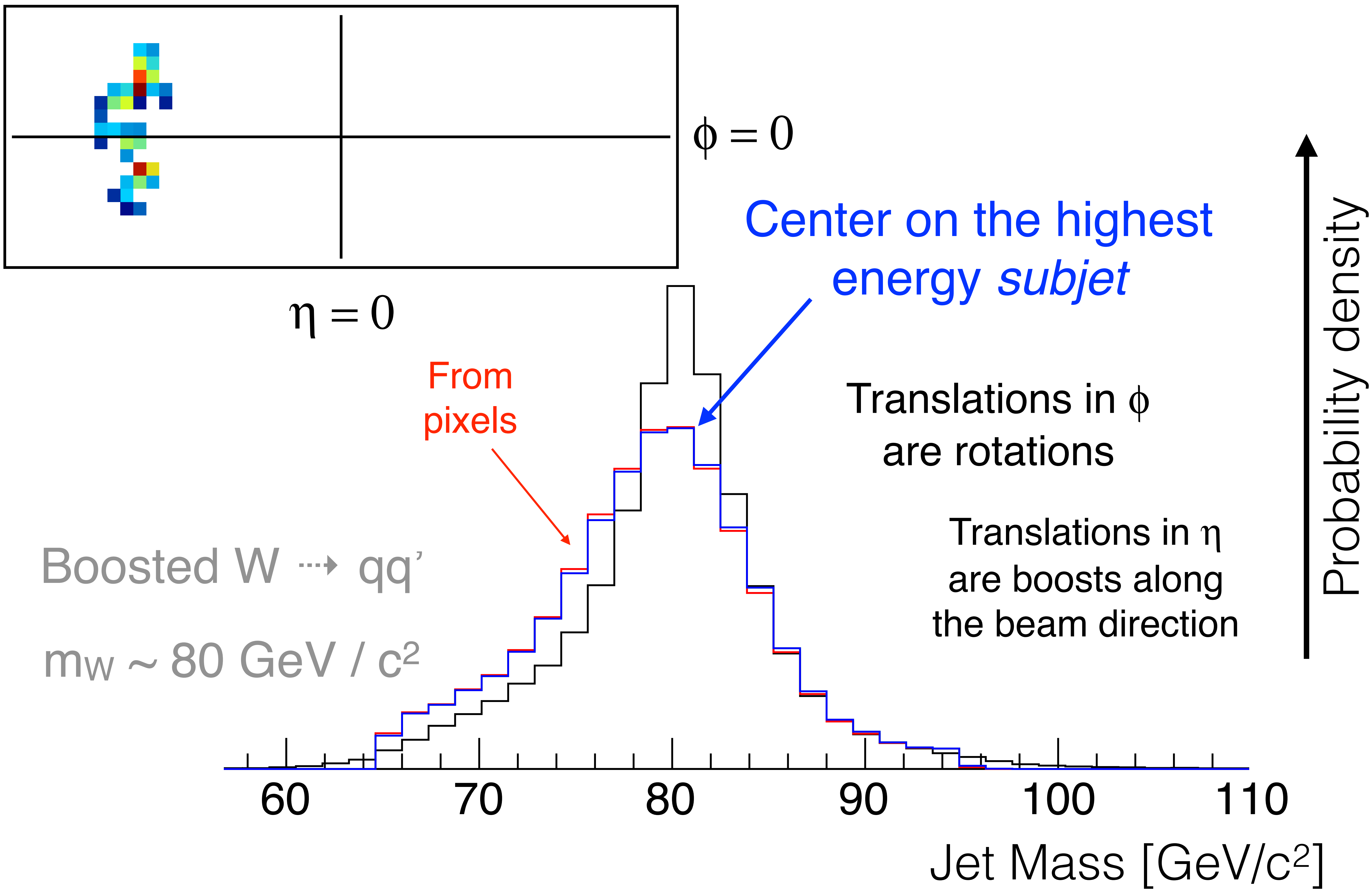
Pre-processing & spacetime symmetries

One of the first typical steps is pre-processing



Can help to learn faster & smarter; but must be careful!

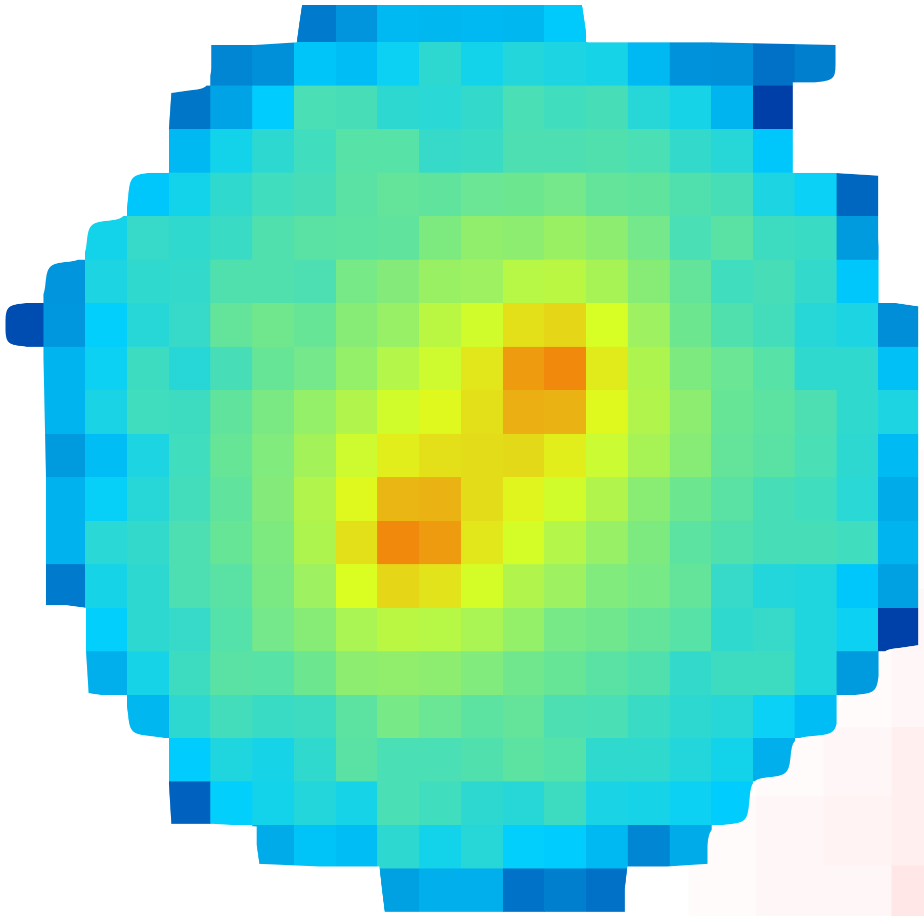
One of the most useful physics-inspired features is the ***jet mass***



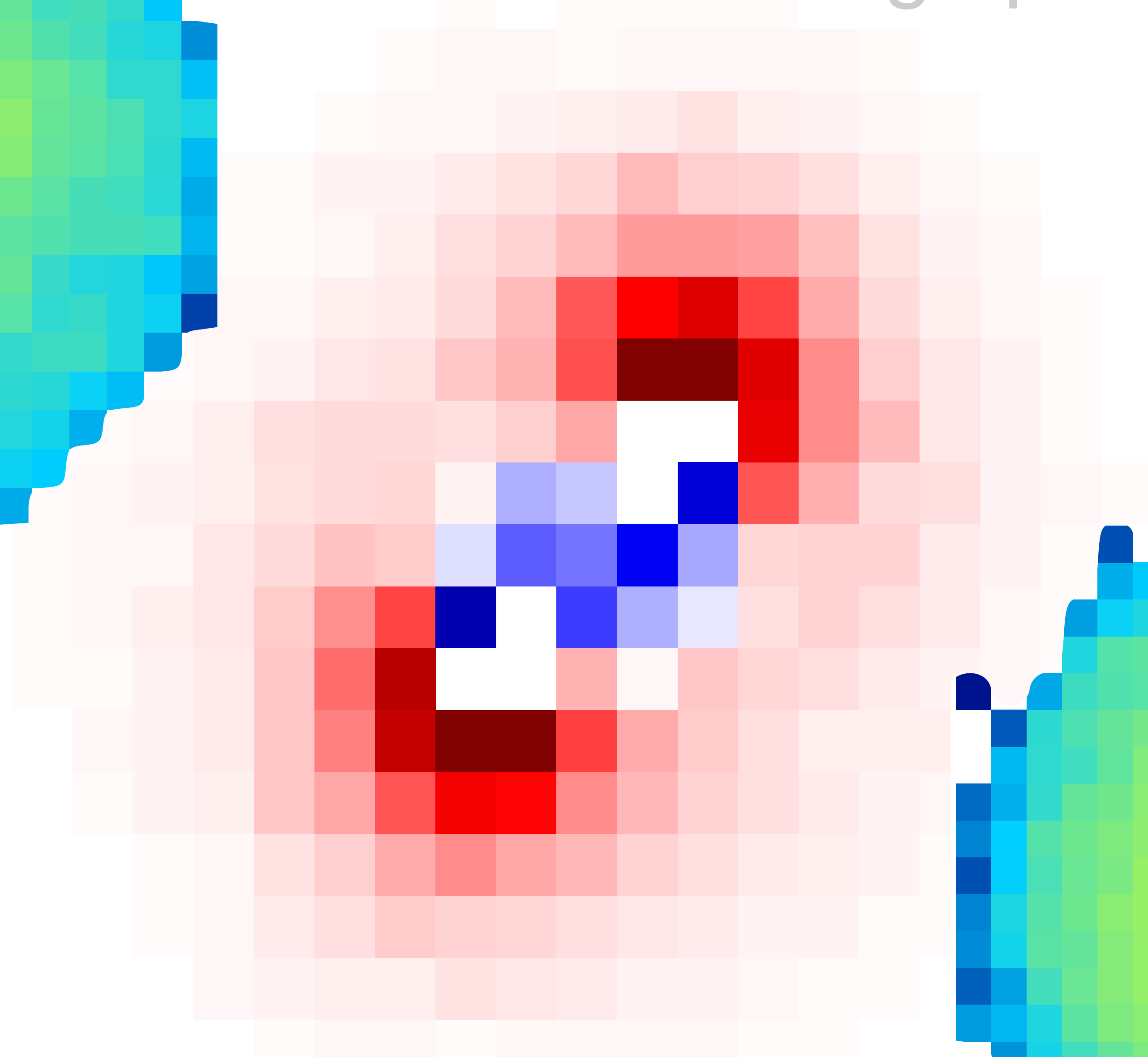
Why images?

Can directly visualize physics

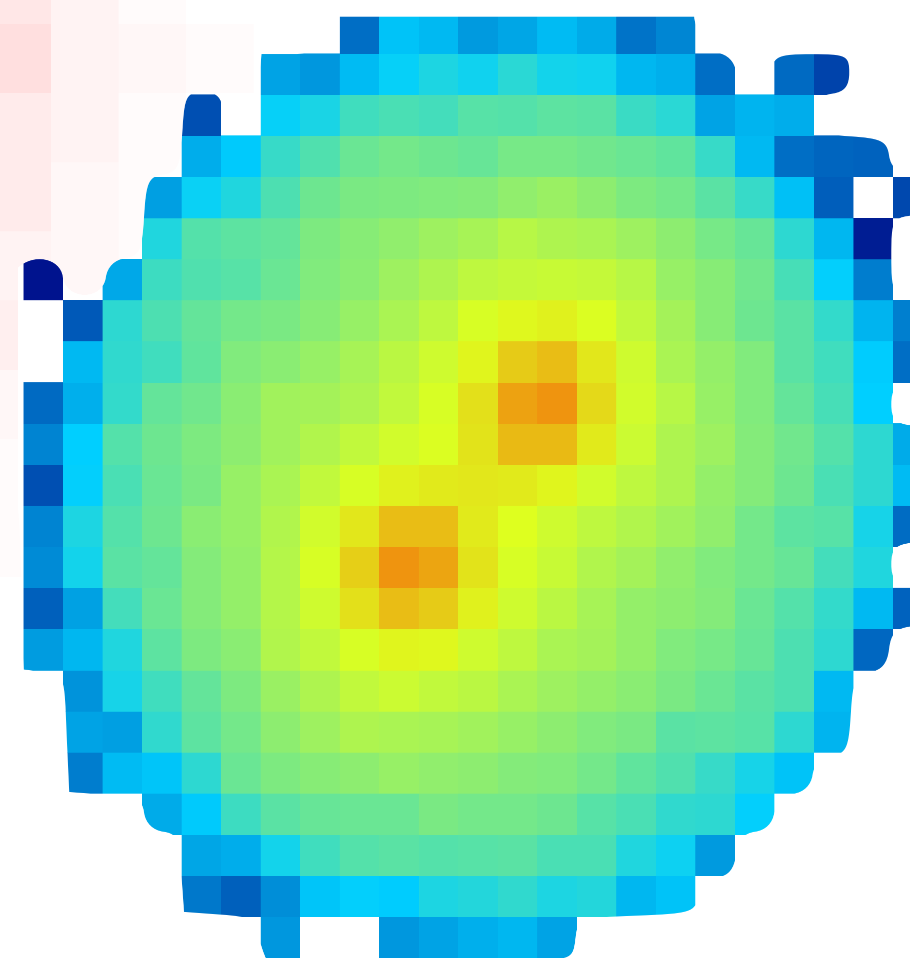
and we can benefit from the extensive image processing literature



W $\dashrightarrow$ $q\bar{q}$

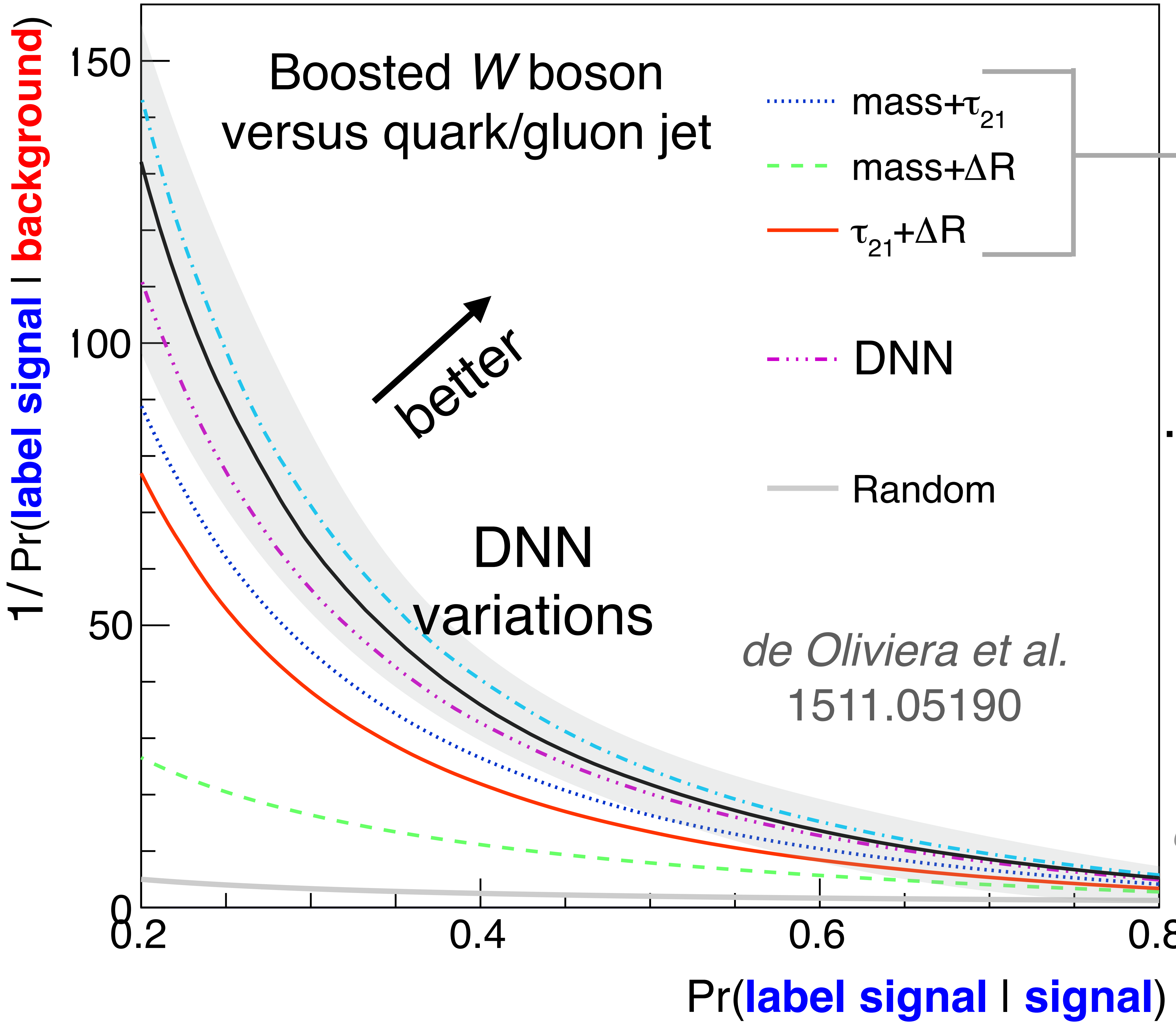


g $\dashrightarrow$ $q\bar{q}$



there is information encoded in the physical distance between pixels

Modern Deep NN's for Classification



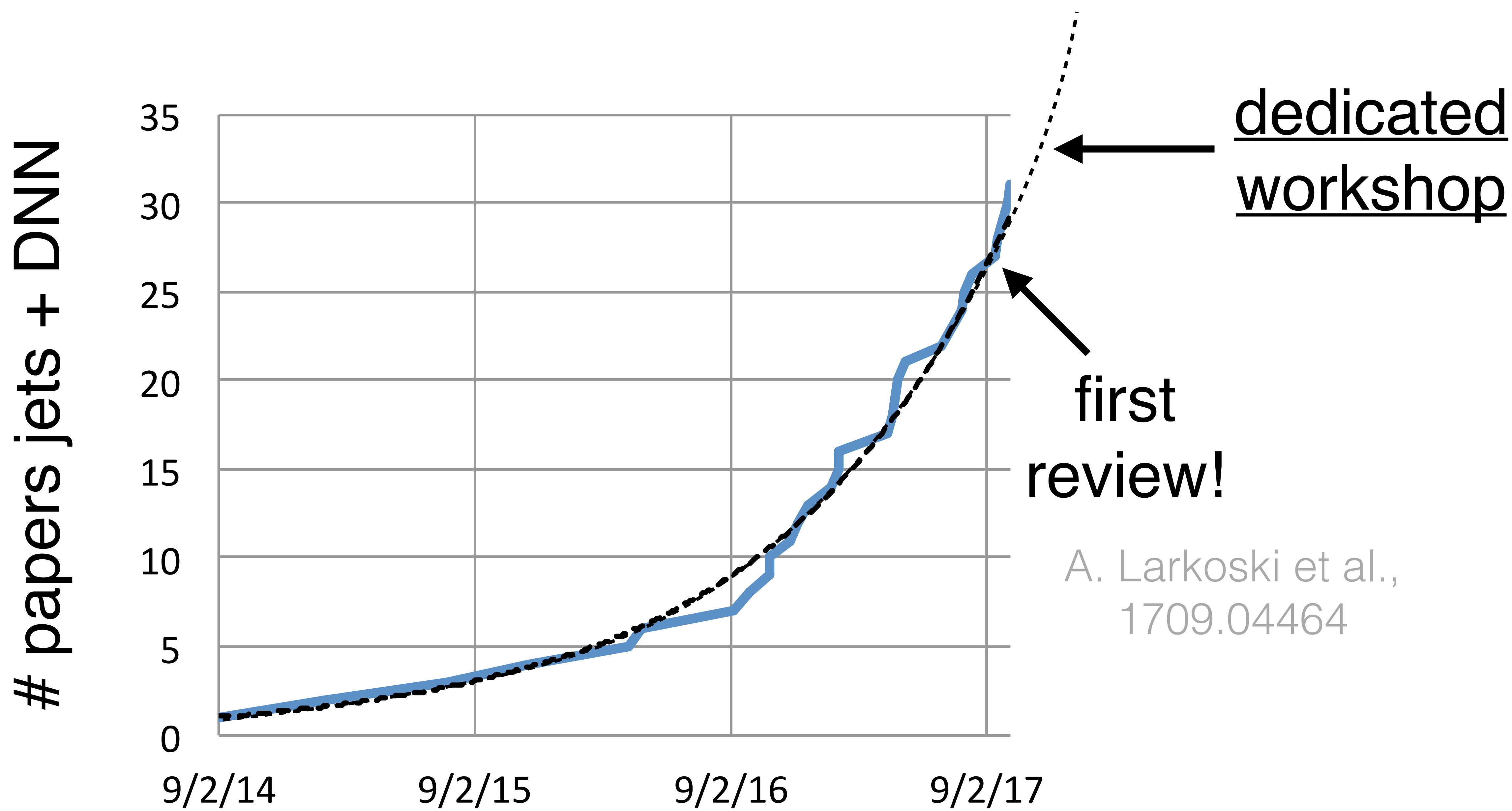
mass, τ_{21}, ΔR are all simple functions of the image

…what the DNN is learning is active R&D!

See also
L. Almeida et al. 1501.05968
Baldi et al. 1603.09349
J. Barnard et al. 1609.00607
P. Komiske et al. 1612.01551
G. Kasieczka et al. 1701.08784
W. Bhimji et al. 1711.03573

Exciting New Directions

So far only scratches the surface
…this is a very active field of research!



A. Larkoski et al., 1709.04464

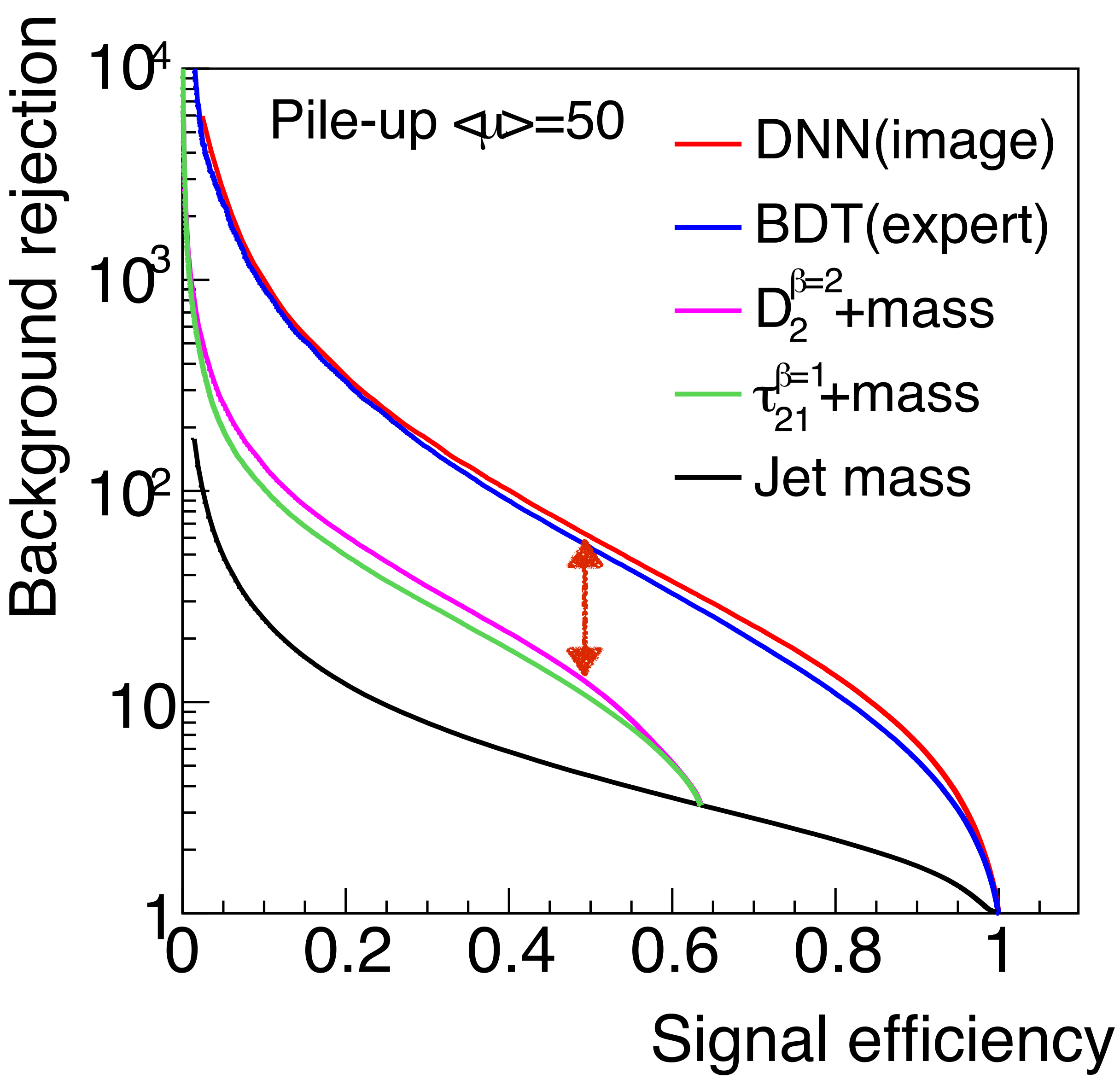
DEEP LEARNING VS. THEORY

Oliveira, et. al arXiv:1511.05190
Whiteson, et al arXiv:1603.09349
Barnard, et al arXiv:1609.00607

While the DNN shows a significant improvement with respect to the jet mass combined with single theory inspired variable (eg. τ_{21}, D_2), only a small improvement with respect to a BDT using several theory-inspired variables

Other Problems:

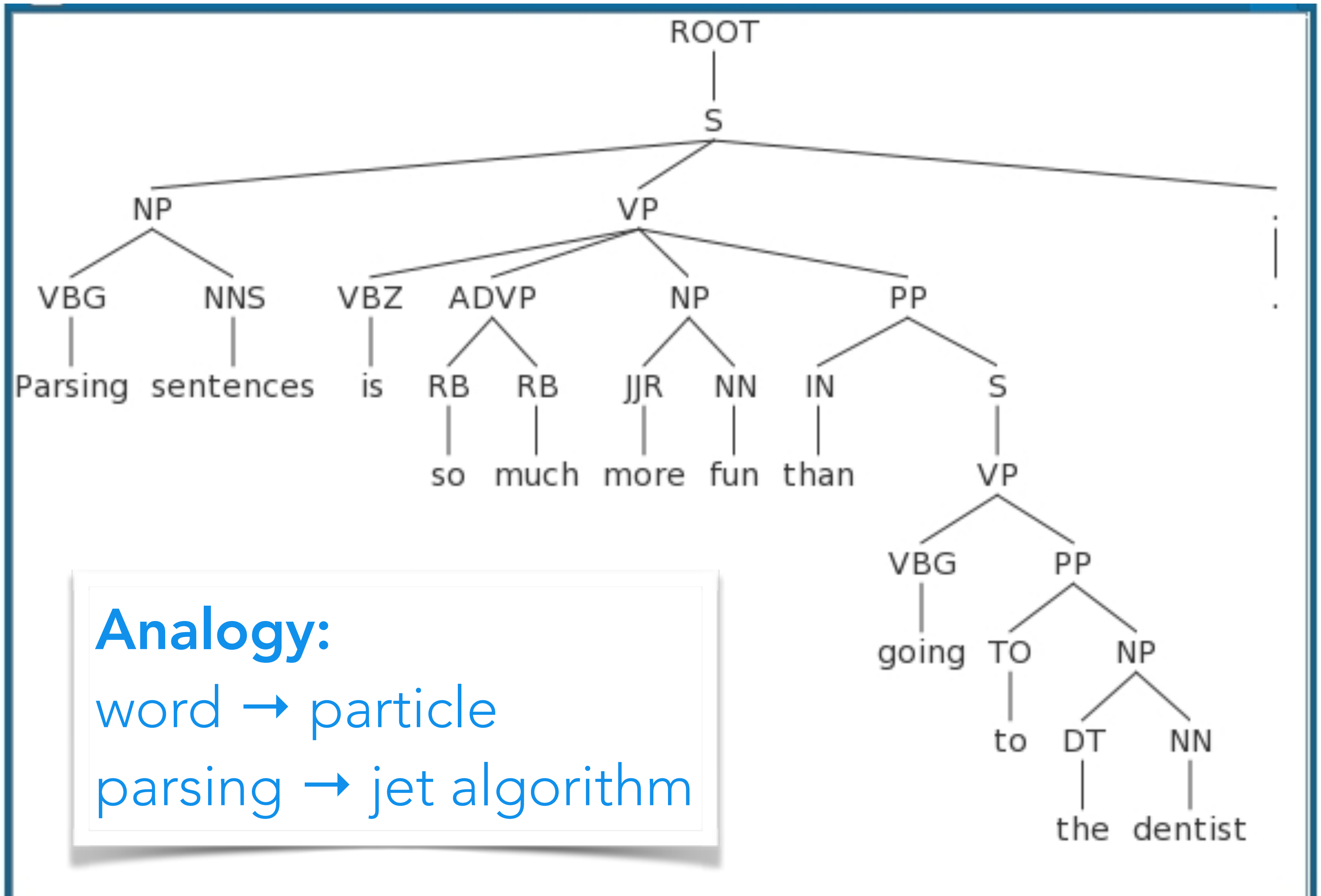
- image-based approach not easily generalized to non-uniform calorimeters
- not easy to extend to tracks, projecting into towers looses information
- theory inspired variables work on set of 4-vectors & have important theoretical properties



FROM IMAGES TO SENTENCES

Recursive Neural Networks showing great performance for Natural Language Processing tasks

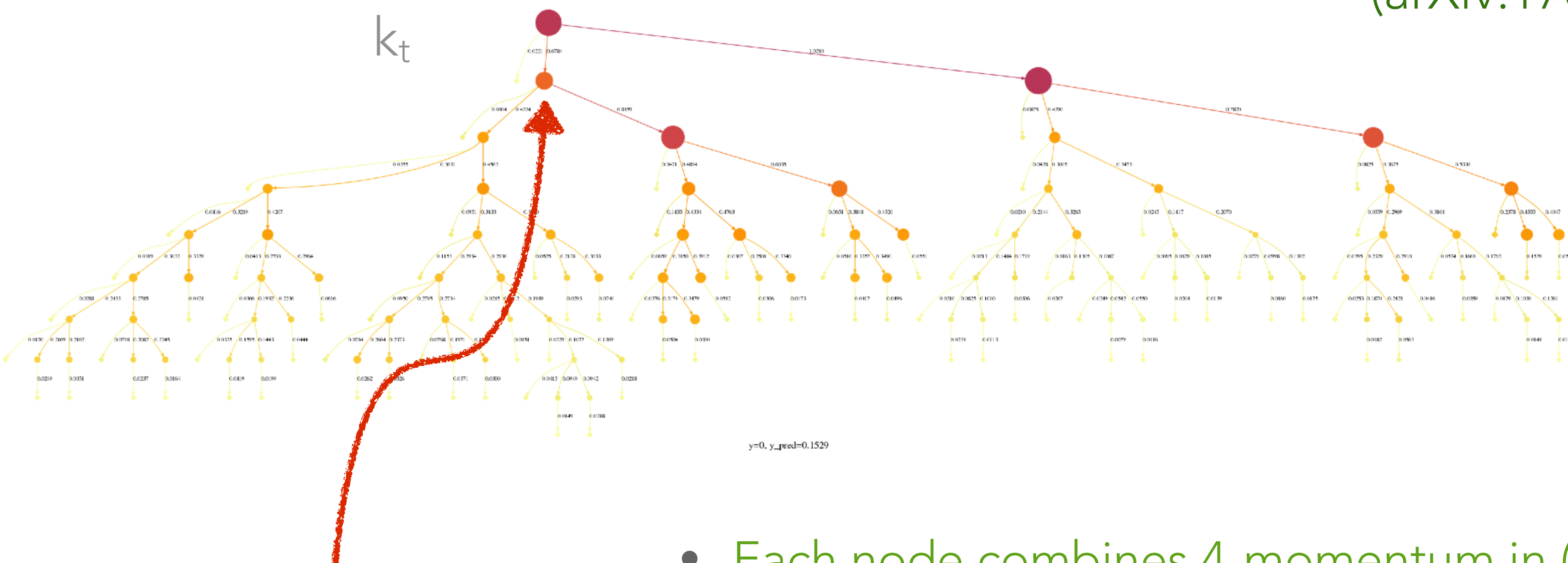
- neural network's topology given by parsing of sentence!



Analogy:
word → particle
parsing → jet algorithm

QCD-INSPIRED RECURSIVE NEURAL NETWORKS

(arXiv:1702.00748)



$$\mathbf{h}_k^{\text{jet}} = \begin{cases} \mathbf{u}_k & \text{if } k \text{ is a leaf} \\ \mathbf{z}_H \odot \tilde{\mathbf{h}}_k^{\text{jet}} + \mathbf{z}_L \odot \mathbf{h}_{k_L}^{\text{jet}} + & \text{otherwise} \\ \hookrightarrow \mathbf{z}_R \odot \mathbf{h}_{k_R}^{\text{jet}} + \mathbf{z}_N \odot \mathbf{u}_k & \end{cases}$$

$$\mathbf{u}_k = \sigma\left(W_u g(\mathbf{o}_k) + b_u\right)$$

$$\mathbf{o}_k = \begin{cases} \mathbf{v}_{i(k)} & \text{if } k \text{ is a leaf} \\ \mathbf{o}_{k_L} + \mathbf{o}_{k_R} & \text{otherwise} \end{cases}$$

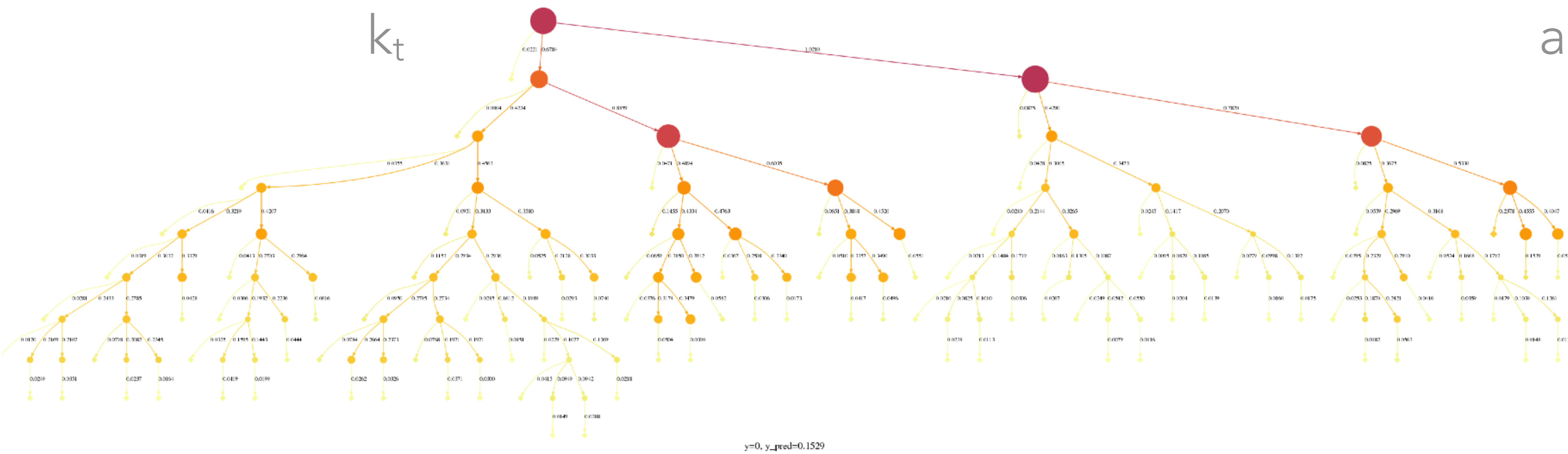
$$\tilde{\mathbf{h}}_k^{\text{jet}} = \sigma\left(W_{\tilde{h}} \begin{bmatrix} \mathbf{r}_L \odot \mathbf{h}_{k_L}^{\text{jet}} \\ \mathbf{r}_R \odot \mathbf{h}_{k_R}^{\text{jet}} \\ \mathbf{r}_N \odot \mathbf{u}_k \end{bmatrix} + b_{\tilde{h}} \right)$$

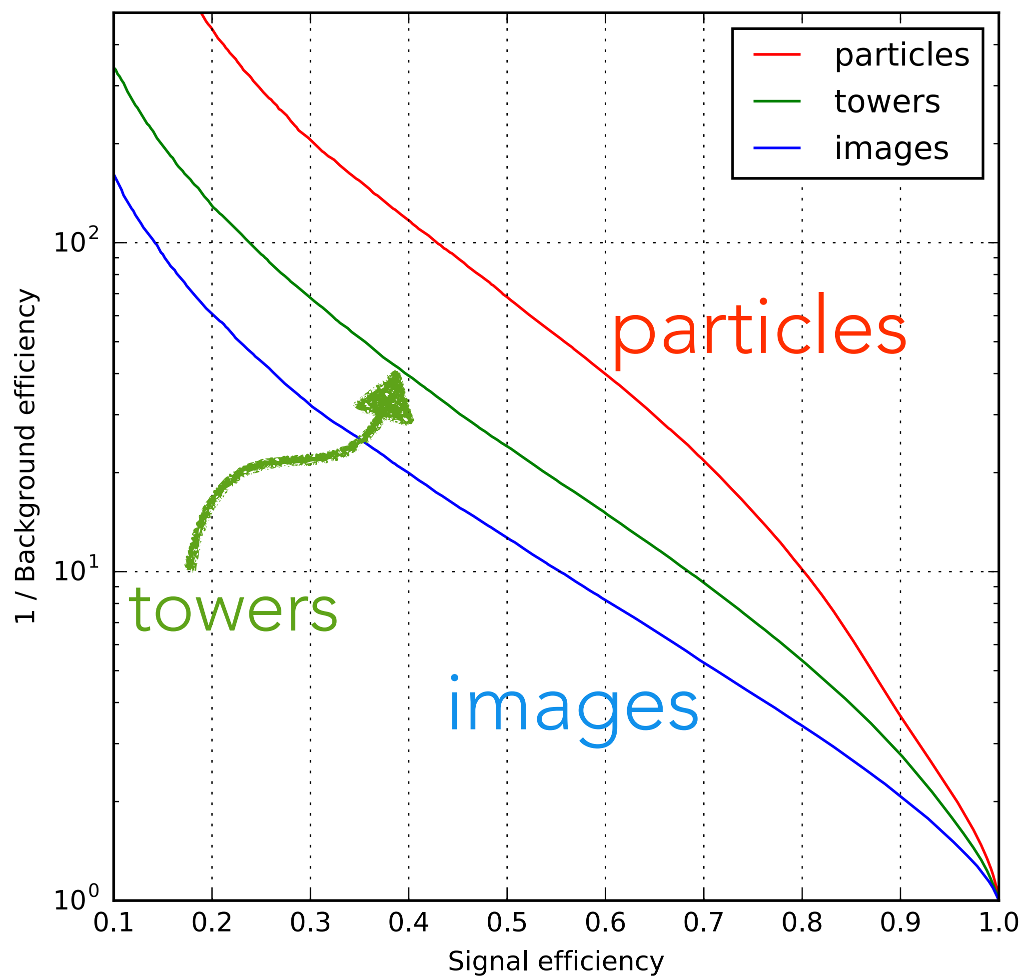
$$\begin{bmatrix} \mathbf{z}_H \\ \mathbf{z}_L \\ \mathbf{z}_R \\ \mathbf{z}_N \end{bmatrix} = \text{softmax}\left(W_z \begin{bmatrix} \tilde{\mathbf{h}}_k^{\text{jet}} \\ \mathbf{h}_{k_L}^{\text{jet}} \\ \mathbf{h}_{k_R}^{\text{jet}} \\ \mathbf{u}_k \end{bmatrix} + b_z \right)$$

$$\begin{bmatrix} \mathbf{r}_L \\ \mathbf{r}_R \\ \mathbf{r}_N \end{bmatrix} = \text{sigmoid}\left(W_r \begin{bmatrix} \mathbf{h}_{k_L}^{\text{jet}} \\ \mathbf{h}_{k_R}^{\text{jet}} \\ \mathbf{u}_k \end{bmatrix} + b_r \right)$$

- Each node combines 4-momentum in (E-scheme recombination of o_k) and a non-linear transformation of hidden state of children h_{kL}, $h_{kR} \in \mathbb{R}^{40}$
- Recursively applied (shared weights, Markov)
- "gating" allows for weighting of information of L/R children and for to flow directly along one branch

QCD-INSPIRED RECURSIVE NEURAL NETWORKS





- W-jet tagging example using data from Dawe, et al arXiv:1609.00607
- down-sampling by projecting into images looses information
- RNN needs much less data to train!

Neural Message Passing for Jet Physics

Isaac Henrion, Johann Brehmer, Joan Bruna, Kyunghun Cho, Kyle Cranmer
Center for Data Science
New York University
New York, NY 10012
{henrion*, johann.brehmer, bruna, kyunghyun, kyle.cranmer*}@nyu.edu

Gilles Louppe
Department of Computer Science
University of Liège
Belgium
g.louppe@ulg.ac.be

Gaspar Rochette
Department of Computer Science
École Normale Supérieure
Paris, France
gaspar.rochette@ens.fr

Abstract

Supervised learning has incredible potential for particle physics, and one application that has received a great deal of attention involves collimated sprays of particles called jets. Recent progress for jet physics has leveraged machine learning techniques based on computer vision and natural language processing. In this work, we consider message passing on a graph where the nodes are the particles in a jet. We design variants of a message-passing neural network (MPNN); (1) with a learnable adjacency matrix, (2) with a learnable symmetric adjacency matrix, and (3) with a set2set aggregated hidden state and MPNN with an identity adjacency matrix. We compare these against the previously proposed recursive neural network with a fixed tree structure and show that the MPNN with a learnable adjacency matrix and two message-passing iterations outperforms all the others.

Table 1: Summary of classification performance for several approaches.

Network	Iterations	ROC AUC	$R_{\epsilon=50\%}$
RecNN-k_t (without gating) [10]	1	0.9185 ± 0.0006	68.3 ± 1.8
RecNN-k_t (with gating) [10]	1	0.9195 ± 0.0009	74.3 ± 2.4
RecNN-desc-p_T (without gating) [10]	1	0.9189 ± 0.0009	70.4 ± 3.6
RecNN-desc-p_T (with gating) [10]	1	0.9212 ± 0.0005	$\mathbf{83.3} \pm \mathbf{3.1}$
RelNet	1	0.9161 ± 0.0029	67.69 ± 6.80
MPNN (directed)	1	0.9196 ± 0.0015	89.35 ± 3.54
MPNN (directed)	2	0.9223 ± 0.0008	$\mathbf{98.26} \pm \mathbf{4.28}$
MPNN (directed)	3	0.9188 ± 0.0031	85.93 ± 8.50
MPNN (undirected)	1	0.9193 ± 0.0015	86.41 ± 3.80
MPNN (undirected)	2	0.8949 ± 0.1004	$\mathbf{97.27} \pm \mathbf{5.02}$
MPNN (undirected)	3	0.9185 ± 0.0036	84.53 ± 8.64
MPNN (set, directed)	1	0.9189 ± 0.0017	88.23 ± 4.53
MPNN (set, directed)	2	0.9191 ± 0.0046	87.46 ± 14.14
MPNN (set, directed)	3	0.9176 ± 0.0049	$\mathbf{88.33} \pm \mathbf{9.84}$
MPNN (set, undirected)	1	0.9196 ± 0.0014	85.65 ± 4.48
MPNN (set, undirected)	2	0.9220 ± 0.0007	$\mathbf{94.70} \pm \mathbf{2.95}$
MPNN (set, undirected)	3	0.9158 ± 0.0054	75.94 ± 12.54
MPNN (id)	1	0.9169 ± 0.0013	74.75 ± 2.65
MPNN (id)	2	0.9162 ± 0.0020	74.41 ± 3.50
MPNN (id)	3	0.9158 ± 0.0029	74.51 ± 5.20