Machine Learning @ CMS

Tommaso Boccali

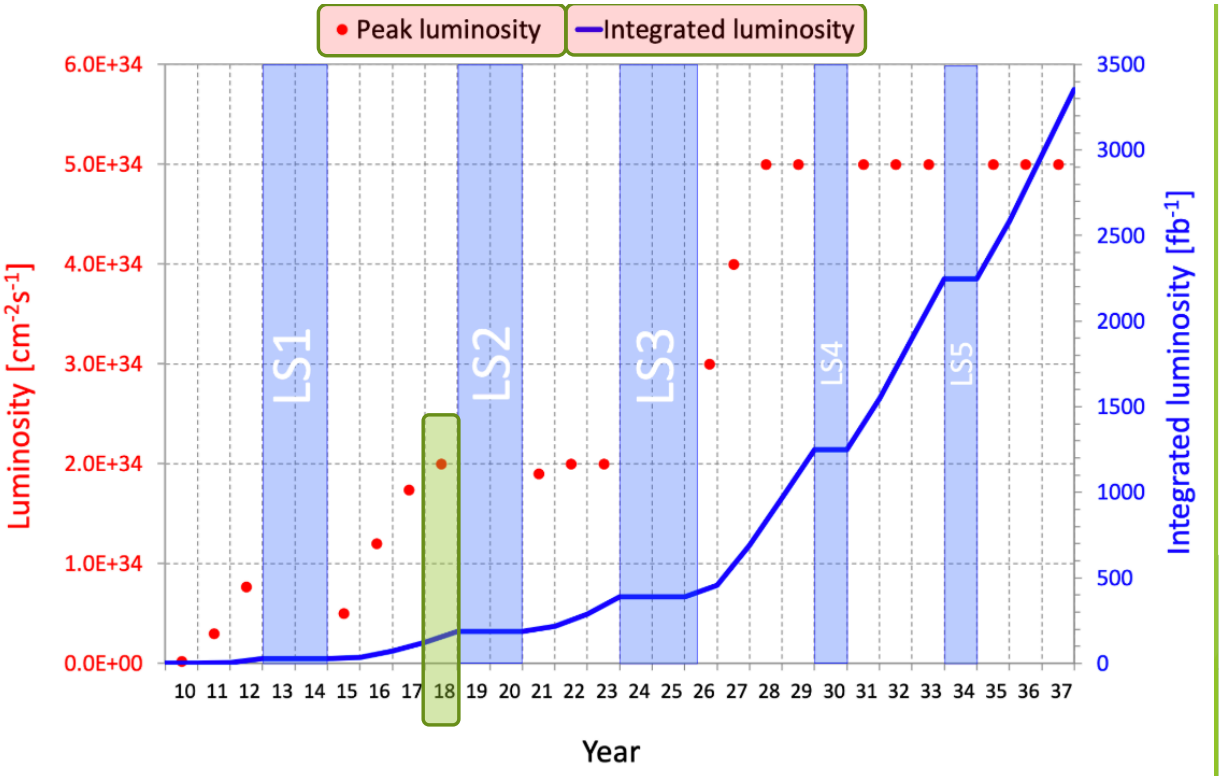
INFN-Pisa / CERN

Outline

- What is CMS@LHC?
- Which are the computing needs, today and in the next ~20 years?
- Which is the role of machine learning, and in general of artificial intelligence?
- Some example of current studies, and future oriented ideas

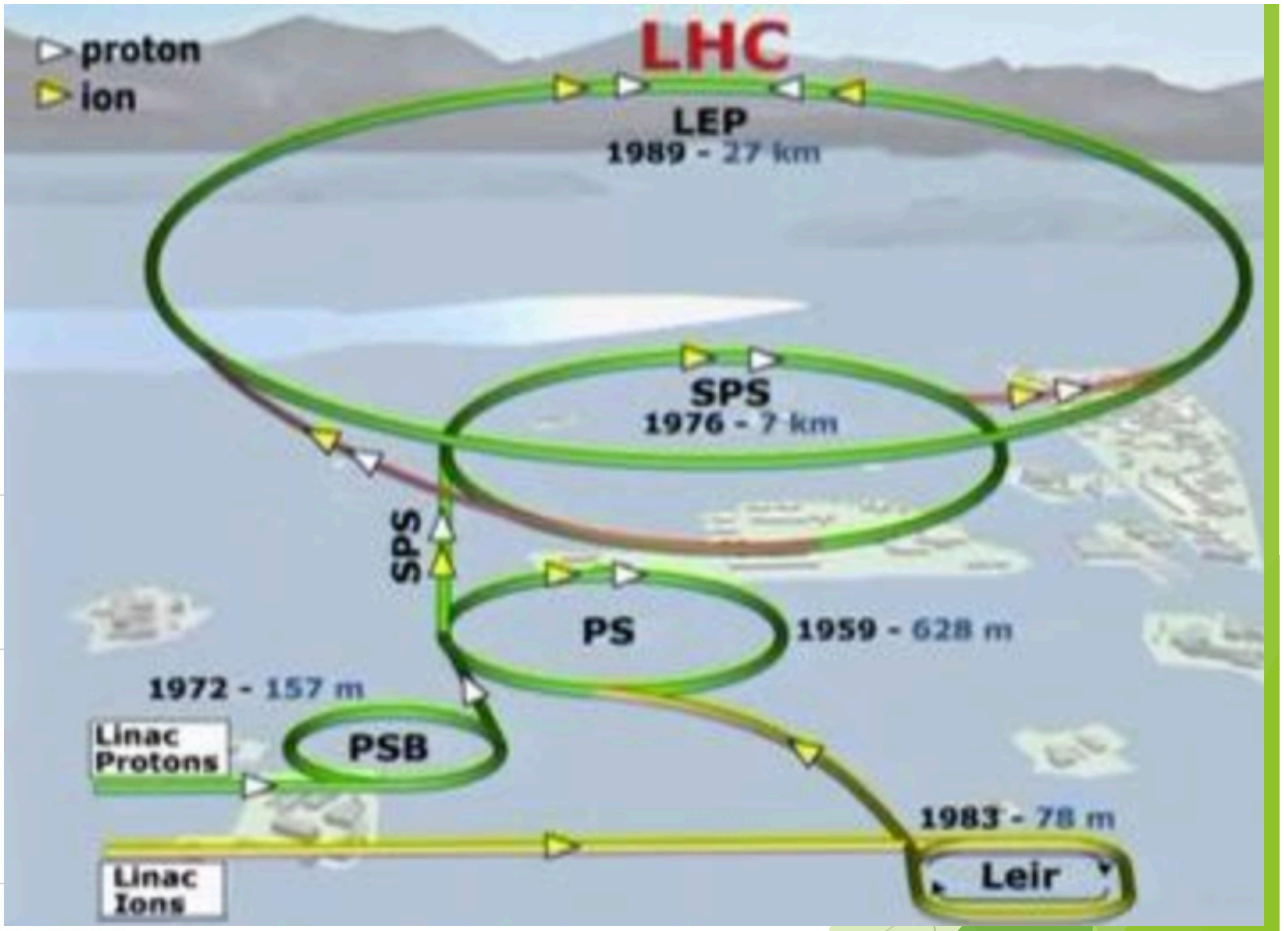
CMS @ LHC @ CERN

- LHC is today's **top energy pp collider**.
- It started operations in 2009, and is now at the end of the second run period
- At each "Run", the collider improves, colliding particles in greater number, of greater energy, or with greater efficiency
- The "**physics capability**" is measured in terms of "**luminosity**", which is proportional to the number of events you can collect of a given type



HL-LHC: High Luminosity LHC
LS: Long Shutdown

- Accelerating protons @ 6.5 TeV needs a chain of accelerators
- At full load, LHC contain 4800 bunches of ~1.5 10^{11} protons (1 ug)
- Each bunch is separated by 7.5 m (25 ns) from the previous



How much energy are we talking about?

7 TeV = $7 \cdot 10^{12}$ eV · $1{,}6 \cdot 10^{-19}$ J/eV = **$1{,}12 \cdot 10^{-6}$ J**

It doesn´t look like a lot of energy

For the **ALICE** experiment, **each ion of Pb-208 reaches** 1150/2 = **575 TeV.**

So, the energy per nucleon is: 575/208 = 2,76 TeV

Let´s calculate the kinetic energy of an insect of 60 mg flying at 20 cm/s:

$E_k = \frac{1}{2} m \cdot v^2 \Rightarrow E_k = \frac{1}{2} 6 \cdot 10^{-5} \cdot 0{,}2^{\,2} \sim 7$ TeV

That is, in LHC each proton will reach an energy similar to that of an annoying ... **MOSQUITO!**

But we have to keep in mind that this mosquito has 36 thousand trillion nucleons, whereas the **7 TeV in the LHC will be concentrate in one sole proton.**

Maybe this comparison is not very convincing so let´s look at it from another point of view.

Let´s calculate the energy present in each bunch:

7 TeV/protón·$1{,}15 \cdot 10^{11}$ protons/bunch ~**$1{,}29 \cdot 10^{5}$ J/bunch**

A powerful motorbike 150 kg travelling at 150 km/h_

$E_k = \frac{1}{2} \cdot 150 \cdot 41{,}7^2 \sim$ **$1{,}29 \cdot 10^{5}$ J**

So if a bunch of protons collides with you the impact is similar to that produced by a powerful motorbike travelling at 150 km/h.

If you are lucky to avoid that "**0,2 picogram motorbike**", don´t worry, there are 2807 following it. And if you decide to change lanes, the equivalent is coming in the opposite direction.

Another calculation which can show the enormous amount of energy reached is:
$1{,}29 \cdot 10^{5}$ J / bunch x 2808 bunches ~ **360 MJ**

-Stored beam energy-

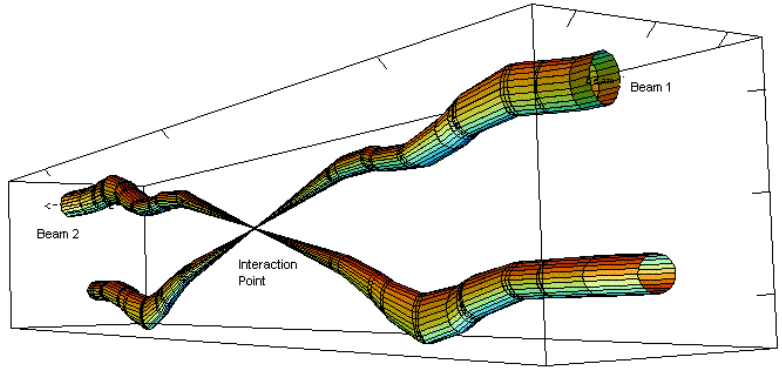
And that is equivalent to

77,4 kg of TNT

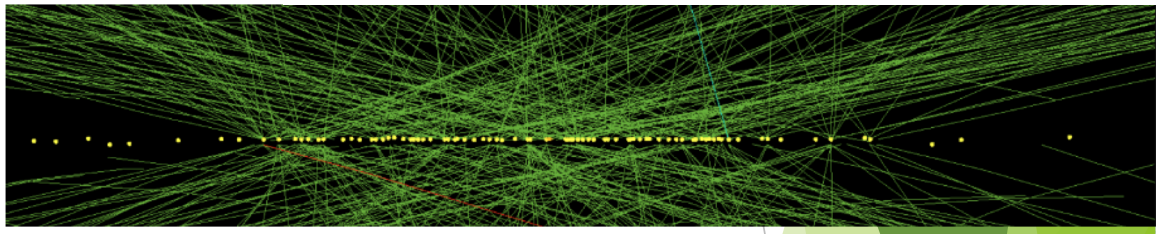
The energy content of TNT is 4.68MJ/kg (Beveridge 1998).

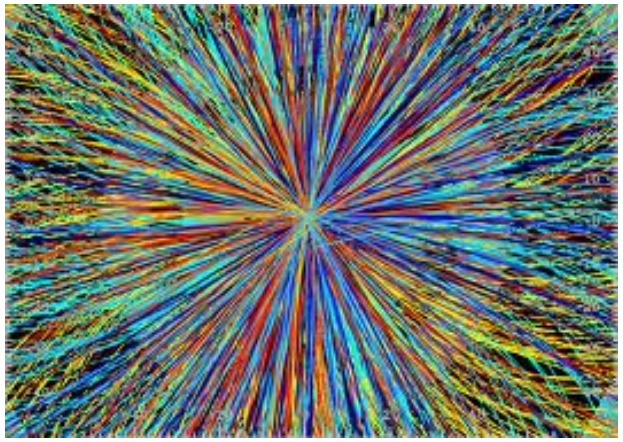
2018 LHC Parameters (to set some numbers)

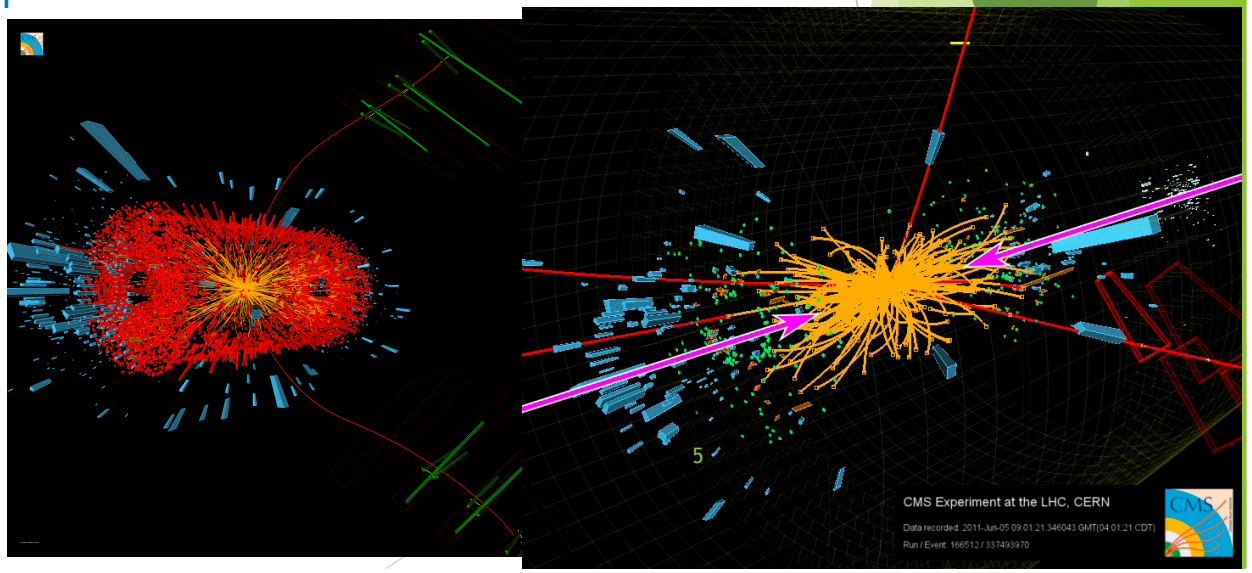
- At an average luminosity of 1.2 10^{34} $cm^{-2}s^{-1}$, every 25 ns:
  - 35 pp interactions, generating secondary particles (we could explain this in terms of cross sections)
  - Particles have a fraction of the venter of mass energy 6.5+6.5 TeV, and fly away from the collision point
  - They traverse the material surrounding the beam line, which we usually fill with active detectors
    - CMS is one of these!



Relative beam sizes around IP1 (Atlas) in collision

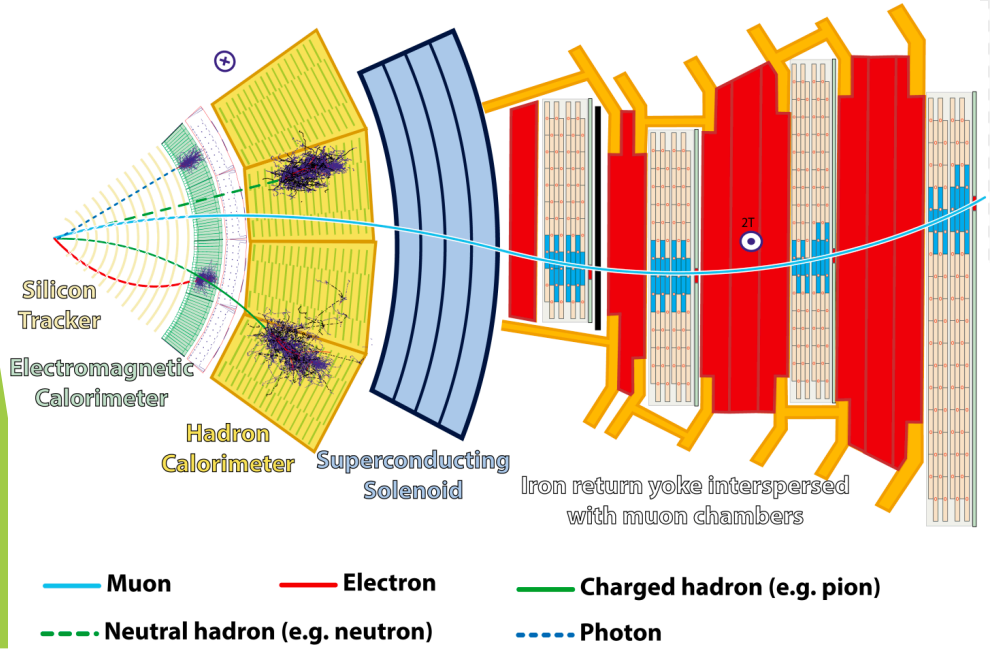


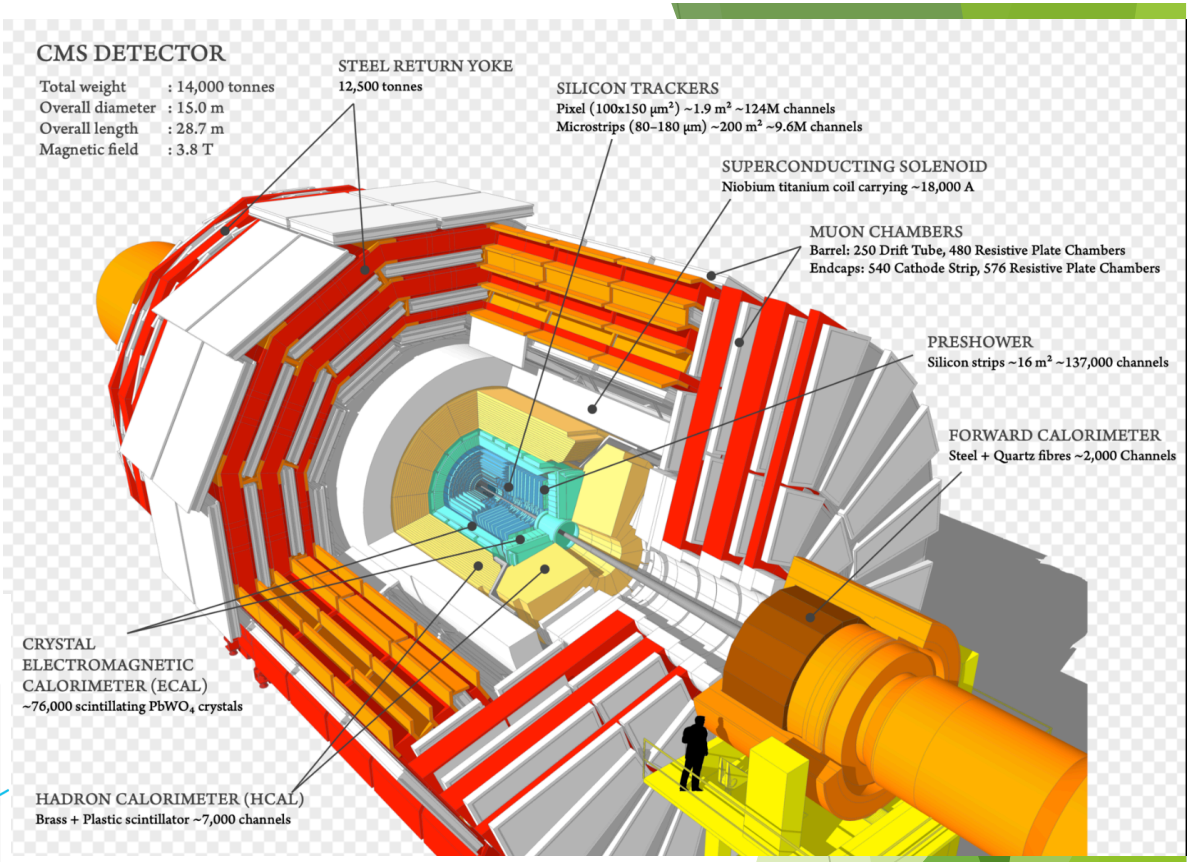




CMS

- The Compart Muon Solenoid (CMS) is one the 4 major experiments at LHC
- It is general purpose: can do precision physics and discovery physics
- It uses subdetectors with different technologies, targeted to measuring / stopping different particle types

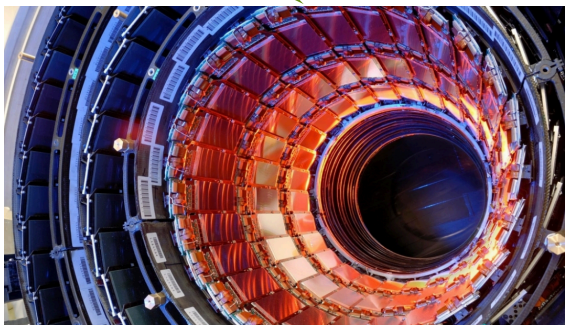


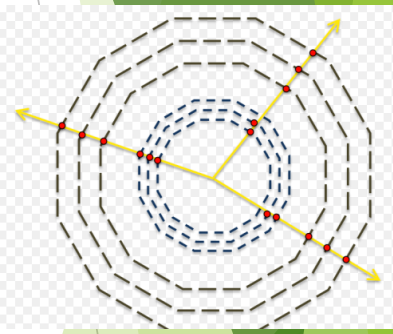


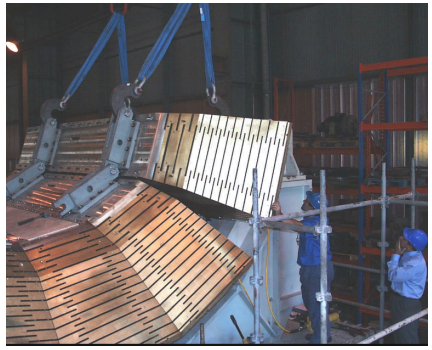
All in all, > 100M acquisition channels need to be read in principle every 25 ns (40 MHz)

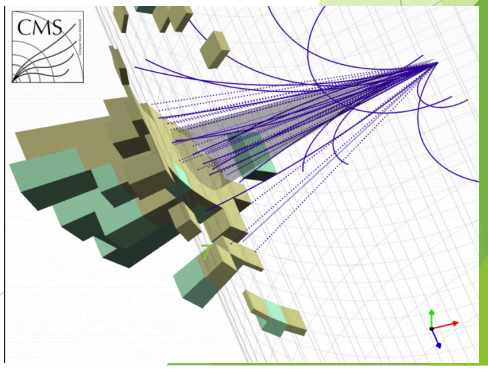
Measuring or Stopping particles?

- When you want to measure a moving object, you generally have 2 ways:
  1. "speed cameras": you measure the momentum of something slightly perturbing it (for example launching some photons toward it)
     - We call them "Tracking devices"; typically, made of thin silicon layers which signal the passage of a charged particle via ionization loss
  2. "crash test": you stop it and measure the amount of energy it releases to you
     - We call them "calorimeters": bulky detectors, with heavy material, stop incoming particles and measure their energy







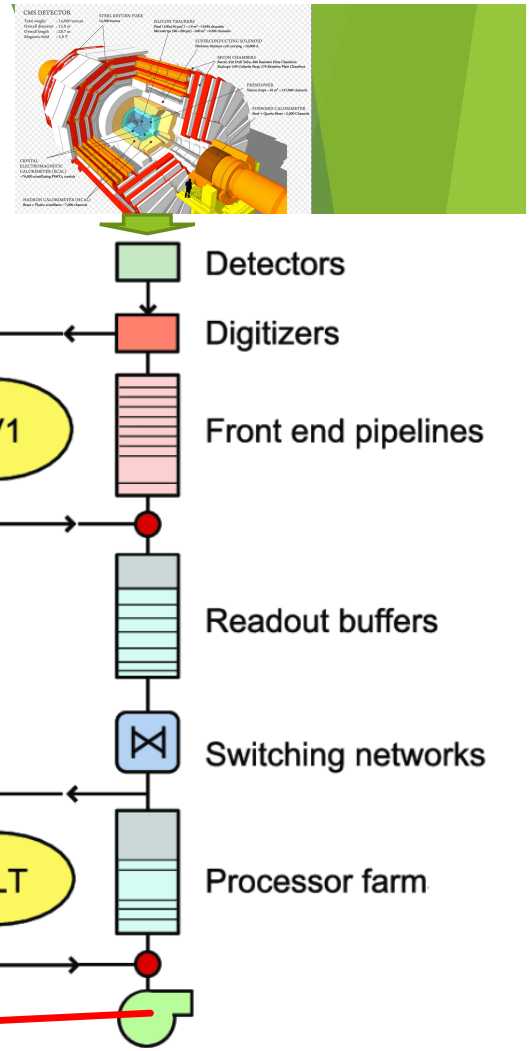


How to do physics analyses at CMS?

- Collision rate is 40 MHz ((25 ns)$^{-1}$), with ~100M acquisition channels
  - Assume naively 1 byte per channel → 100 MB * 40 MHz = **4 PetaByte/s**
- **Zero suppression**: read only interesting channels → 1/100
- **L1 Trigger**: look into the events in hardware, within 6 usec → 1/400
- **HighLevel Trigger**: use 25000 PCs to look further, within 300 msec → 1/100

- **All in all, the physics output of CMS is ~1 GB/s, sustained for 30% of the year**

- This gets saved to tape, then analyzed in GRID farms, and eventually shipped to analyzers



So overall, how much resources do we need to analyze CMS data?

- These include resources to :
  - **CPU**: process data (reconstructing particles, jets, leptons), process Monte Carlo, analyze data/MC, perform fits, …
  - **Disk**: provide inputs to processing and analysis
  - **Tape**: save 2 custodial copies of all the irreproducible data

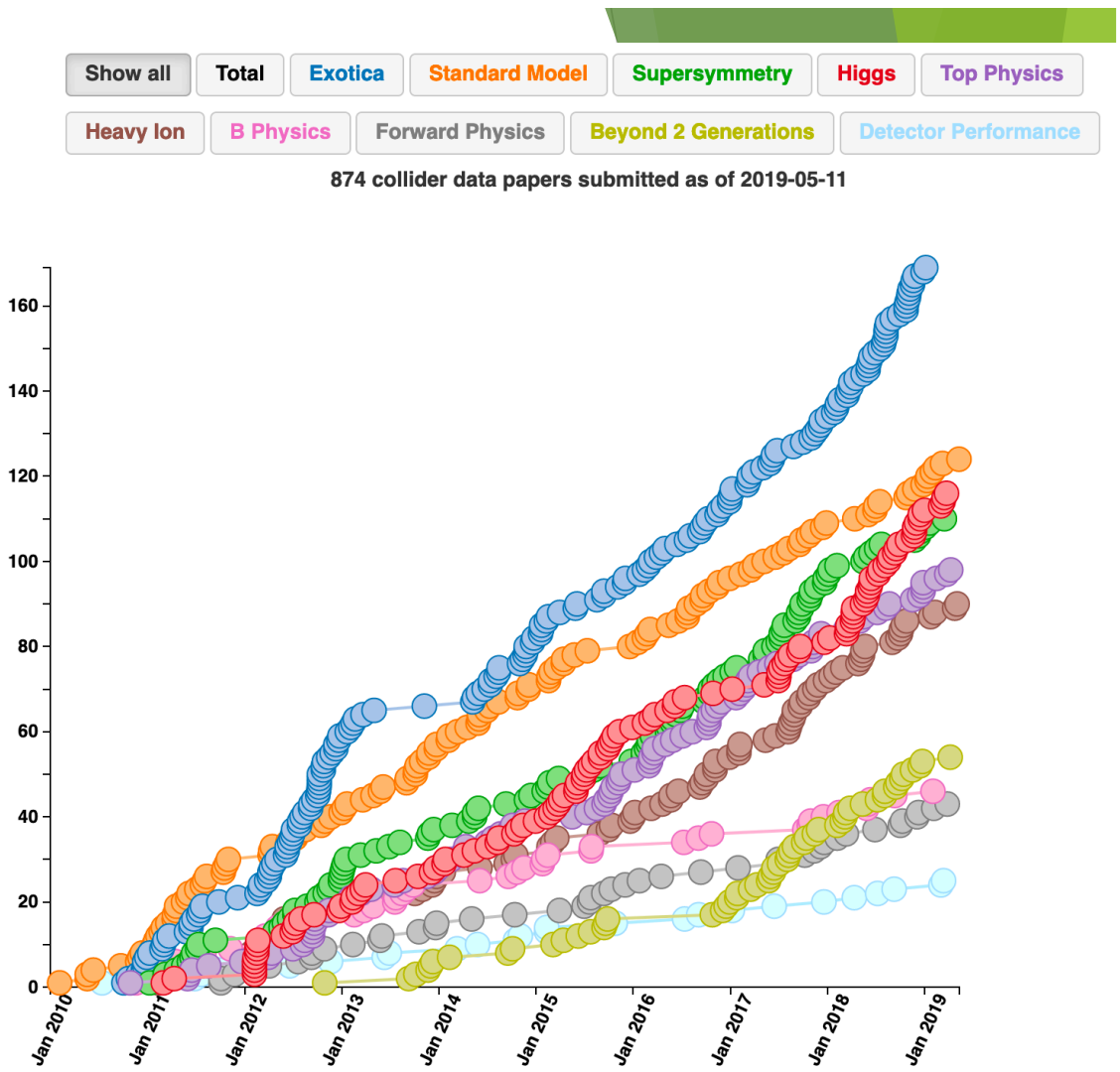
Table 1. WLCG Rebus CMSresource deployments for 2019.

Resource Type	Unit of Measurement	Year	Pledged Amount
CPU	HS06	2019	2M (∼ corresponding to 200k computing cores)
Disk	TB	2019	160k
Tape	TB	2019	290k

- **In general, these numbers are driven by**
  - **CPU: reconstruction time (20 sec/ev)**
  - **CPU: simulation time (50 sec/ev)**
  - **Disk: size of reconstructed events (500 kB/ev)**
  - **Tape: size of the raw event from DAQ (1 MB/ev)**

Did it “work”?

- CMS to-date:
  - 874 physics papers
  - Observation of Higgs boson: 9000 citations



So far so good, then why do we need ML @ CMS?

- 2 very different needs

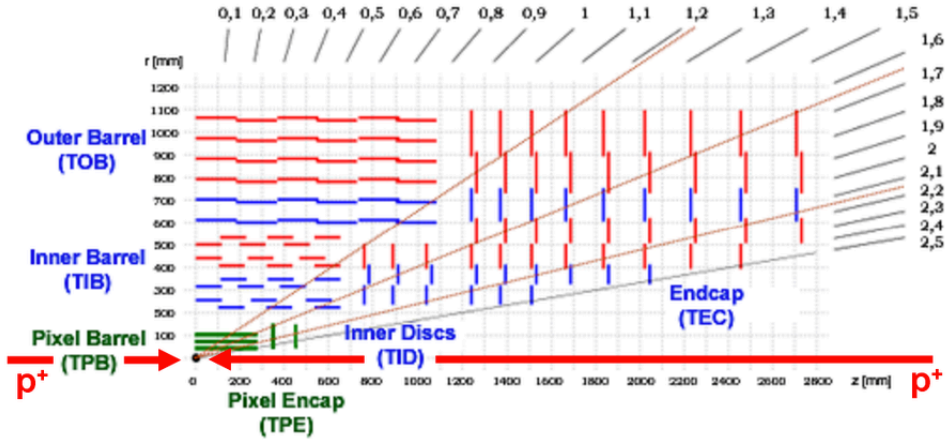
1. **Do better (more precise algorithms, more more physics performance) → more physics output!**
2. **Save resources (less computers, less manpower for operations) → less money**

- #1 is important, but there is general belief that without #2 we could simply NOT be able to work in the next 20 years … let's this this before

CMS Needs for 2026+

- In 2026, LHC will start its “High Luminosity” Phase, with parameters largely improved
  - Average number of pp events per bunch collision 35 → 200 (6x)
- At the same time, CMS will have a much improved detector, with > 2x the number of readout channels
  - Bigger events, more complex reconstruction algorithms
- Also, CMS will need to focus on Higgs physics. To do so, there is an increase of trigger rate from 1 kHz to ~10 kHz (10x)

- Hence, back-of-the-envelope estimate for 2026+ LHC run is a factor 120x of computing resources
  - Scaling from today means, by 2026:
    - 24M CPU cores
    - 19 Exabytes disk (1 Exabyte = 1.000.000 TeraBytes)
    - 35 Exabyte tape
  - (multiply by 4 for the 4 major experiments)
- **Clearly impossible, would be Billions Eur/y**
  - **How can ML help??**



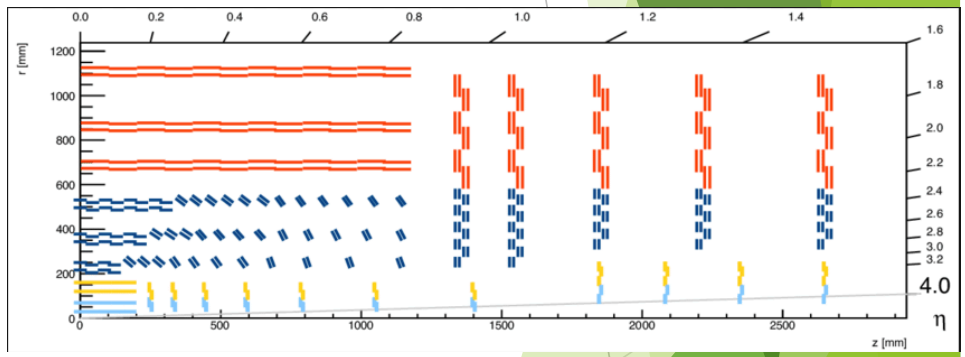
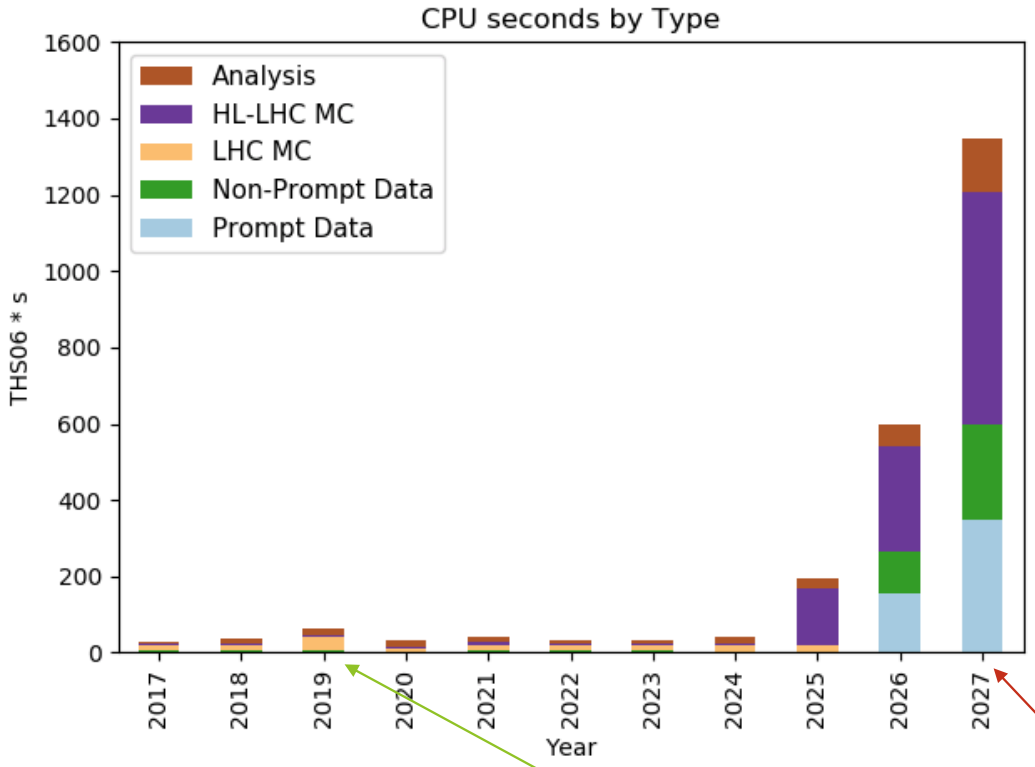
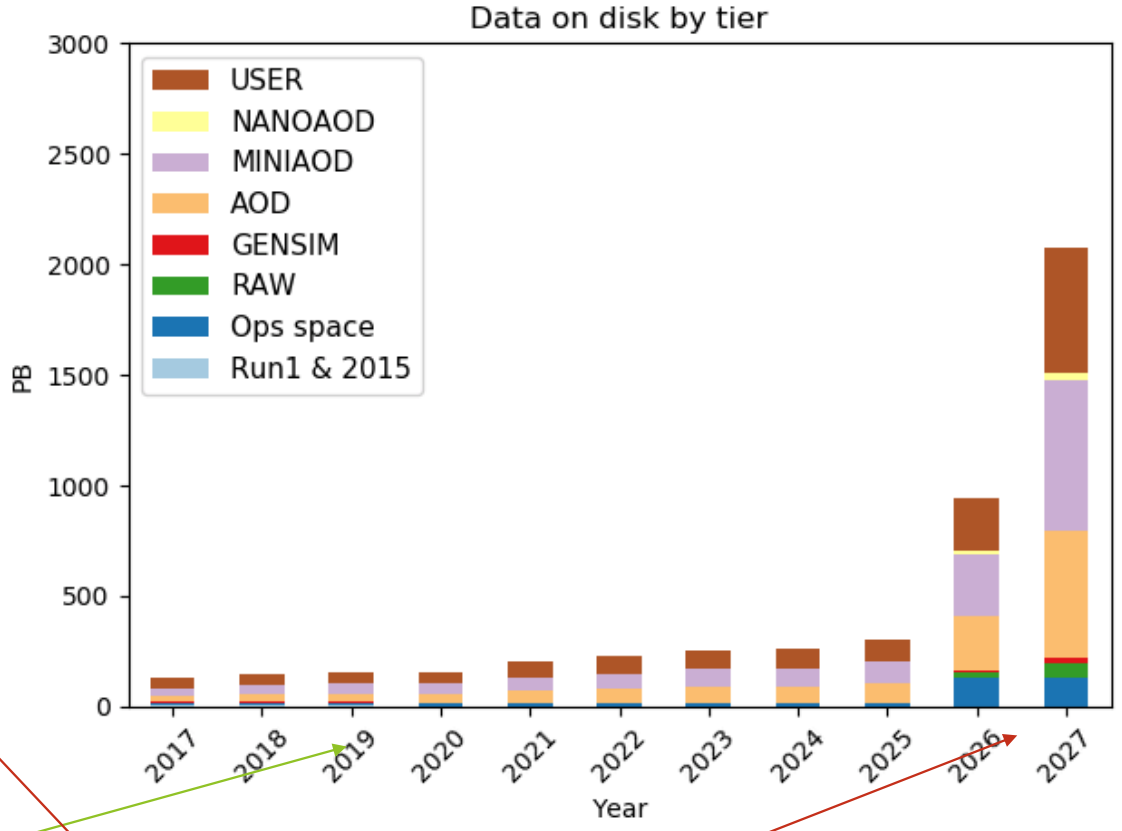


Table 7. basic parameters fro next generation of colliders, as known to-date.

Collider	Operations start date	Length (km)	Particles colliding	Type	Cms collision energy (GeV)	Instantaneous luminosity (10^{34} cm^{-2} s^{-1})	Superimposed pp interactions
LHC (for reference)	2009	27	pp	Circular	7000, 8000, 13000, 14000	Up to 2	60 (peak), 35(average)
HL-LHC	2026	27	pp	Circular	14000	5, 7.5	200

2018 Estimates for 2019 CMS Computing needs



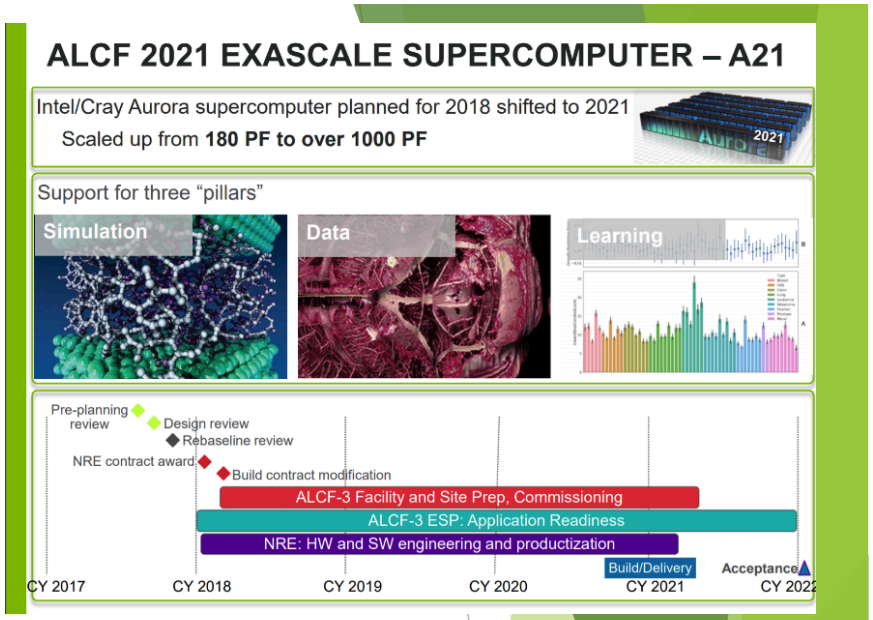


You are here

You need to go here

ML @ CMS as a resource saver

- Machine learning **training is from slow to very slow**, but it is not a big issue since it has to be repeated only a few times a year; instead it is generally **fast at inference** time (when using the training)
  - At least ML algorithm using CNN or (D)FF are “simply algebra operations”: no loops, no recursion - fit well also common processors, using vector registers
- If we could substitute **standard** reconstruction / simulation **algorithms** with **trained ML algorithms**, there is the hope to scale better with event complexity
  - Examples later: tracking, particle-matter interaction
- There are other interesting aspects with minor expected impact:
  - Save smaller data via ML driven data compression (auto encoders)
  - Operation supervision (anomaly detections, data certification) – potentially save manpower
- Training also at large scale is not really an issue also because “we” are offered access to Super Computers (HPC) with hardware specifically tuned for that



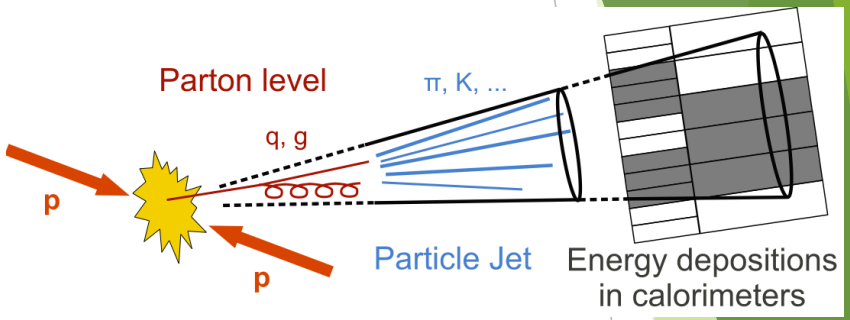
US Department of Energy Supercomputers			
	Frontier	**Aurora**	**Summit**
CPU Architecture	AMD EPYC (Future Zen)	Intel Xeon Scalable	IBM POWER9
GPU Architecture	Radeon Instinct	Intel Xe	NVIDIA Volta
Performance (RPEAK)	1.5 EFLOPS	1 EFLOPS	200 PFLOPS
Power Consumption	~30MW	N/A	13MW
Nodes	100 Cabinets	N/A	3,400
Laboratory	Oak Ridge	Argonne	Oak Ridge
Vendor	Cray	Intel	IBM
Year	2021	2021	2018

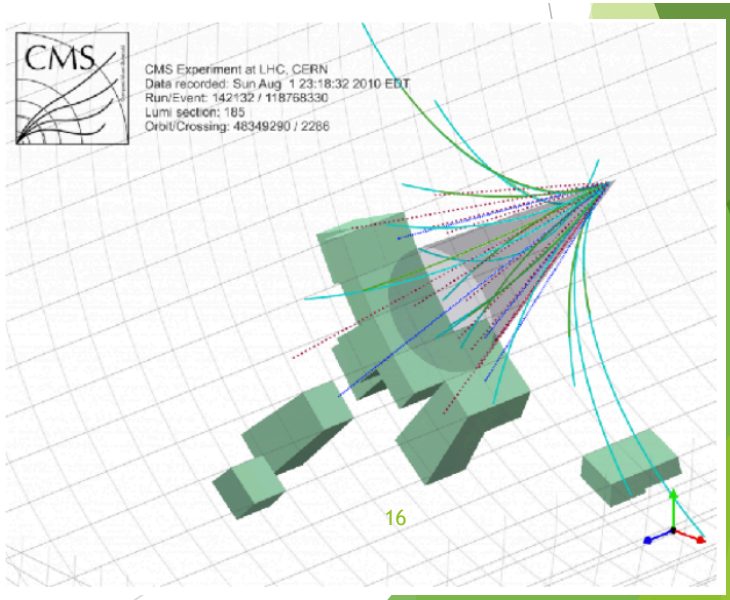
Frontier: Powered by Cray & AMD

ML @ CMS

Faster / Better Jet reconstruction

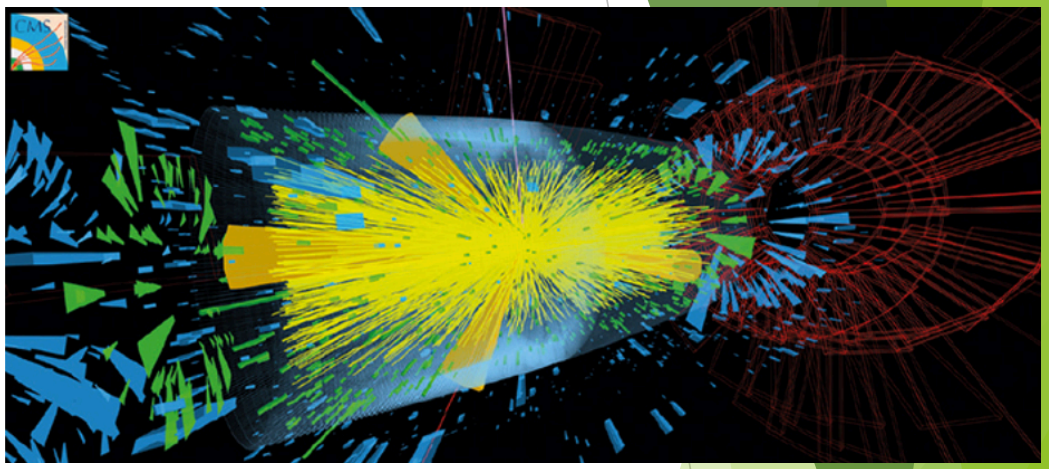
- A jet @ LHC is the result of the shower coming from an unstable quark/gluon decay; all the decay particles are “close” to each other due to the momentum of the decaying particle
- @ CMS jets are reconstructed (mostly) as signals in the Calorimeters, where these particles are stopped and give position / energy information
- By combining the signals, one can in principle reconstruct the energy / direction of the unstable particle, and thus extract physics information on its generating process

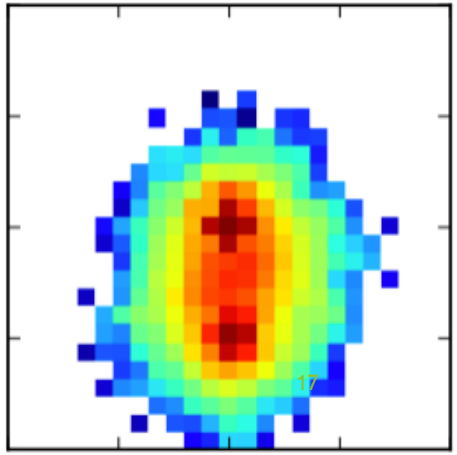




Faster / Better Jet reconstruction #2

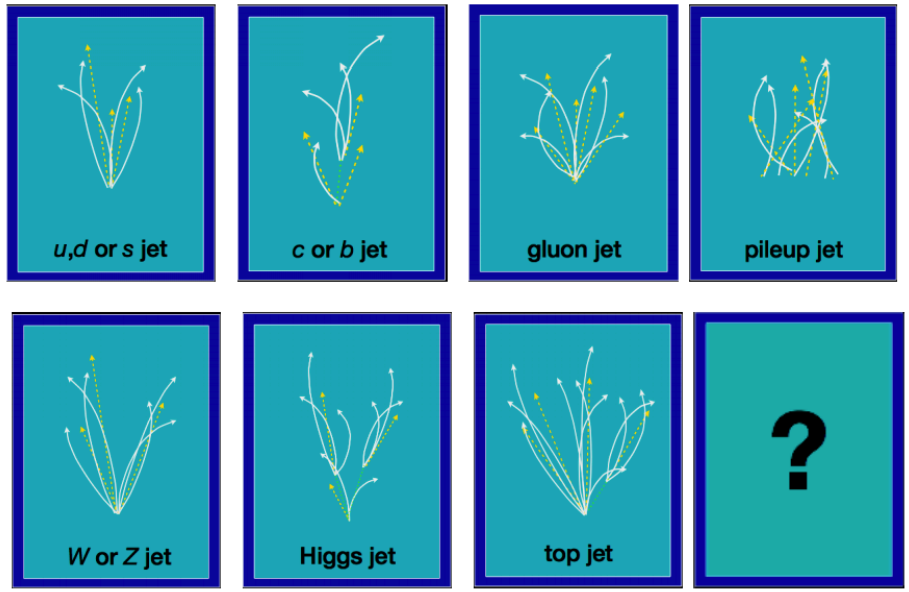
- Classical algorithms for jet reconstruction are iterative and have to face with a ”noisy” environment, with signal jets polluted by particles from the other 35 (200) pp interactions
- **Iterative**: search for a calorimeter cell with very high deposit, and start **joining** nearby active cells up to a certain threshold. Then apply corrections due to the energy response
- Calorimeter response **looks like an image**: why not use the large experience in ML image processing to understand features like
  - Which particle generated the image (a quark, a gluon, a W/Z boson)? – categorization
  - Which is the energy on the originating particle? - regression

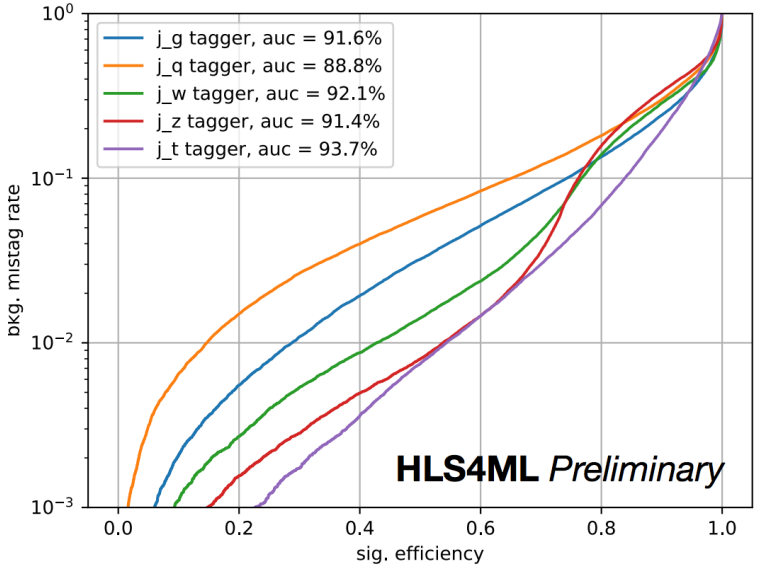




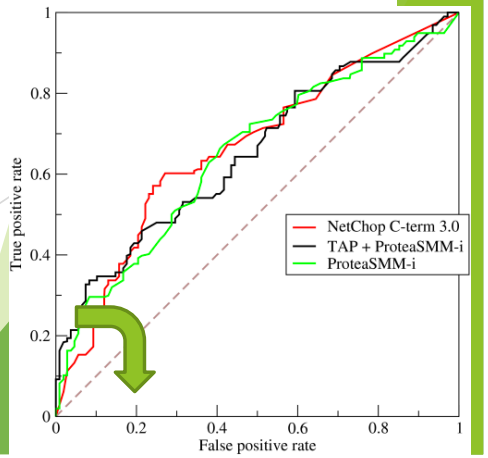
Categorization: an example

- Train a simple 5-layer DNN on Jet calorimetry + other basic event features, and optimize on discrimination



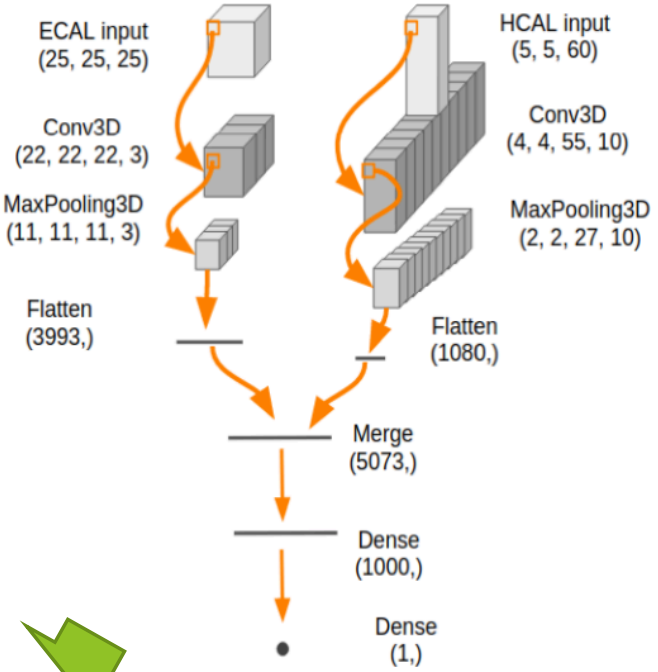


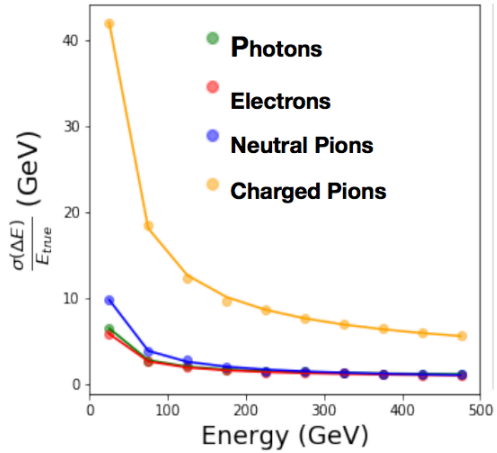
Note: wrt to the ROCs you are more used in other sciences, these curves have Axes swapped, so the lowest right corner is the best performance; then y axis is log…

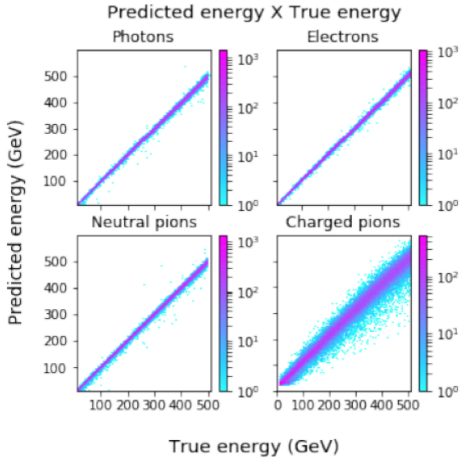


Jet energy Regression …

- … or **how to measure the energy of the initial particle from the "image on calorimeters"** (2 for CMS: ECAL and HCAL)
- Start with images (3rd dimension is time): so more a **movie** than an **image**
- **Convolution layers do sampling and feature discovery**
- **MaxPooling** reduces the parameter space
- The rest is mostly a **dense layer** with 1000 neurons
- Being a classical DNN (no recursion, no loops) its **timing is deterministic, and 1000x faster than standard approaches**

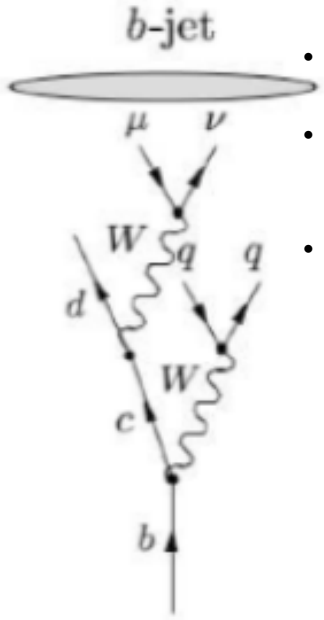






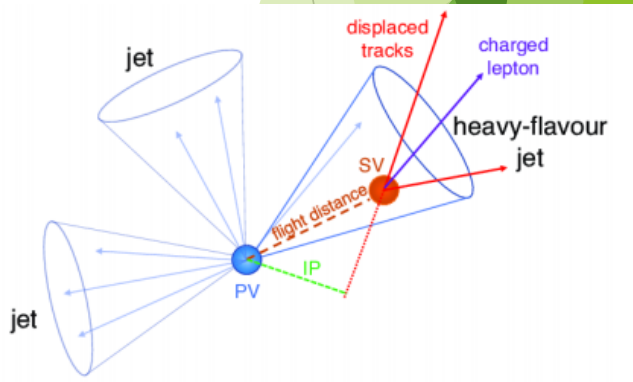
Another categorization example: tagging of b quark Jets

- The identification of **b quarks** is essential for many frontier studies
  - Top quark physics: the top quark decays virtually only to b quarks
  - Many beyond the standard model scenarios: where the coupling is higher for higher mass fermions
  - The Higgs boson has 2 bs as most probably decay mode
- Classically, the discrimination between light (u, d, s + gluon) and b quarks is done looking into the decay topologies
- It is a discrimination problem: answer is binary "looks like b" / "does not look like b"



- A b hadron has non negligible lifetime (10^{-12} sec) and can fly mm before decaying
- A b hadron generates ~5 particles when it decays
- None of these characteristics is clean and easy to spot, and they are valid on a statistical basis
- In general, try to use many inputs with small discrimination and combine them to get more power

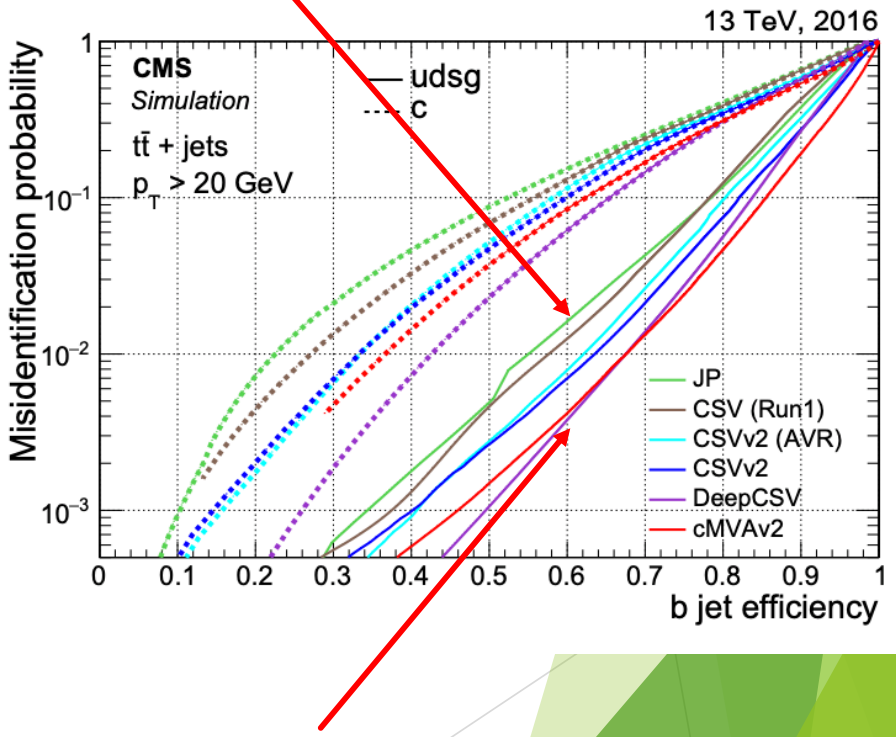
…



b tagging standard algorithms …

- Use 1-50 of these “features” (with most of the discriminating power coming from a few)
- Use statistical methods like likelihoods to combine them
- Different possible approach: build a big DL, including even the inputs wild only mild correlation with the selection
  - **DeepCSV** is an algorithm that uses a deep neural network for identification of b-jets
  - The inputs are secondary vertices parameters, and the parameters of up to 6 per jet
  - DeepCSV is based on a deep neural network training, with 4 hidden dense layers with 100 nodes each
- Why is it better? The DNN is able to «see» correlations between multiple variables, difficutl via statistical methods or analytical representations

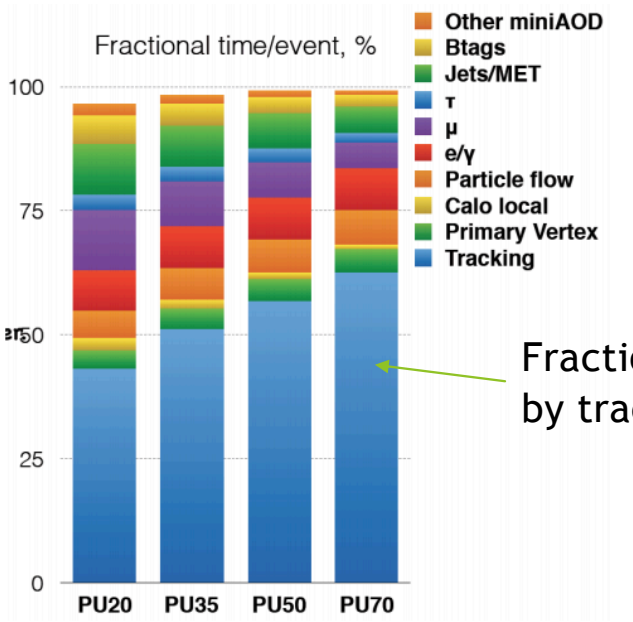
Typical classical algorithm:
60% efficiency for 50x rejection



DeepCSV and other AI based algorithms: 60% efficiency for 300x rejection

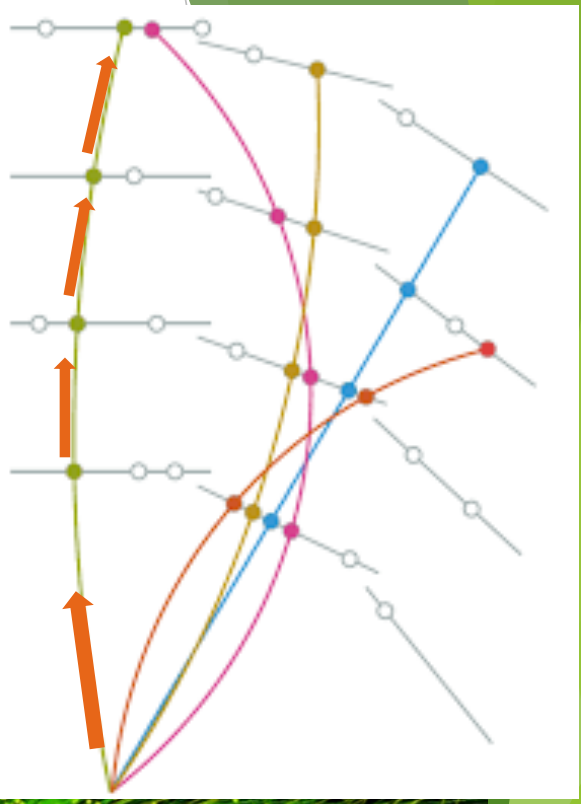
Tracking with ML

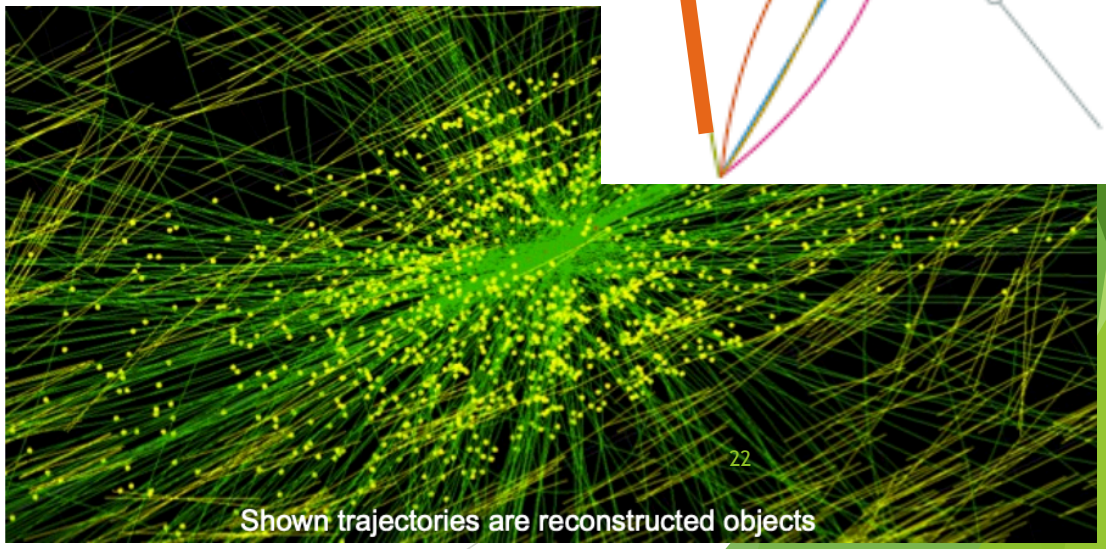
- Tracking would be the **holy grail** of ML
- Currently, tracking mostly via Iterative Kalman Filter algorithms
- Timing-wise, it takes the biggest part of the reconstruction time, and it is exploding



Fraction taken by tracking

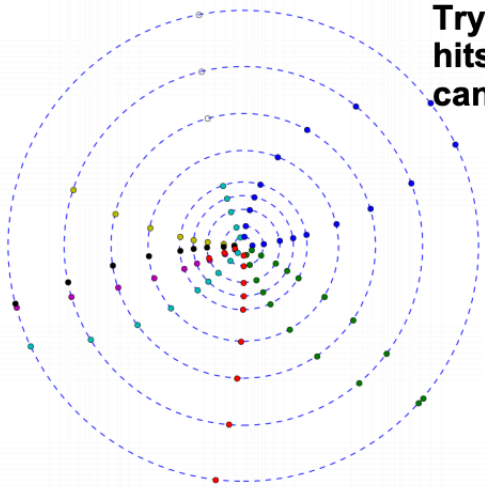
Event complexity





Tracking

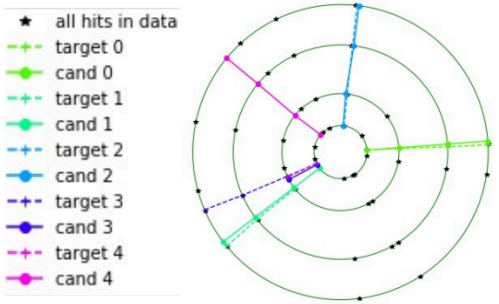
- Extraction of track parameters from a ML system is difficult to think of
  - The final fit uses analytical models, knowledge of the materials, precise knowledge of the magnetic field
- What make sense is to try and have ML defining which hits belong to the same track («**pattern recognition**»), and leaving the computation of the final parameters to a classic system («**track fitting**»)
- HEP.TrkX project: several approaches for a partial (seeding, pixel only, …) or global track finding approach; tested with:
  - Convolutional neural nets (no LSTM)
  - Convolutional auto-encoder
  - Bi-directional LSTM
  - Prediction on next layer with LSTM
  - Definitely NOT ready for prime time



Try to assemble hits into track candidates.

Pattern Recognition with LSTM

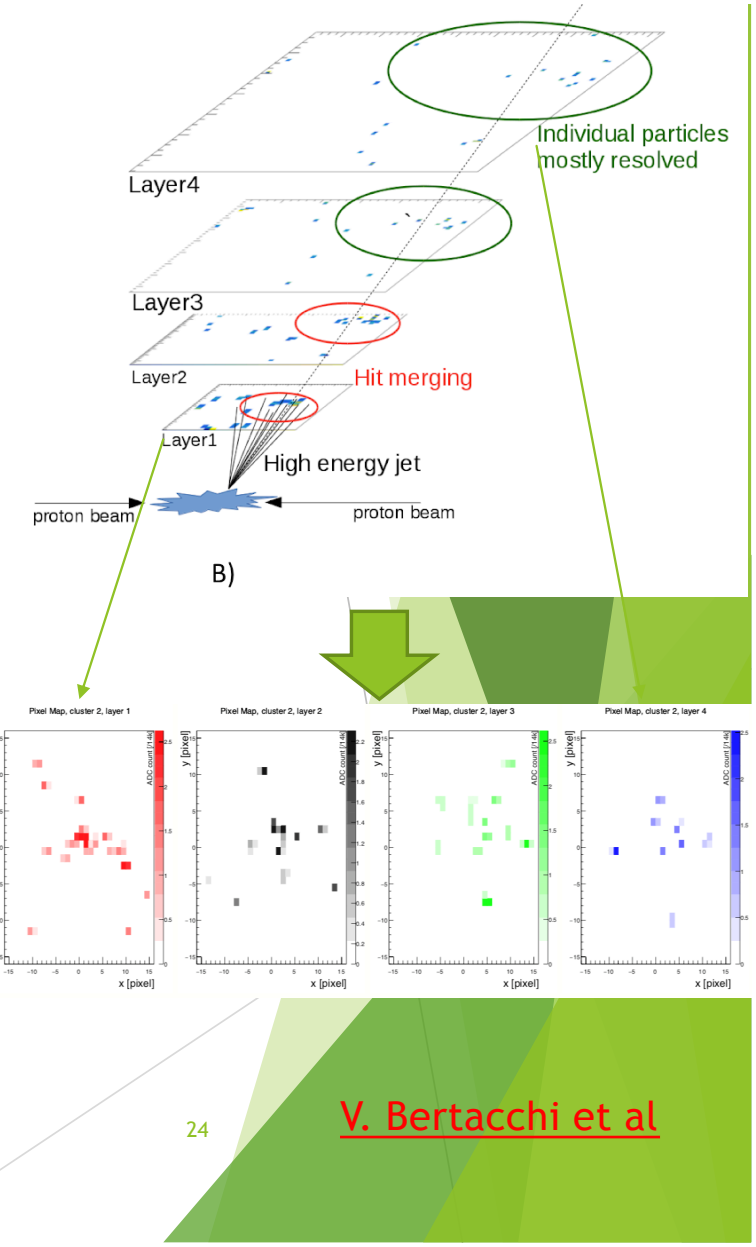
- Input sequence of hits per layers (one sequence per layer)
  - One LSTM cell per layer
- Output sequence of hits per candidates
  - Final LSTM runs for as many candidates the model can predict



- Still work in progress
- Restricted to 4 layers (with seeding in mind)
- Work to some extend

A simple / easier approach: pixel clustering with High Pt jets

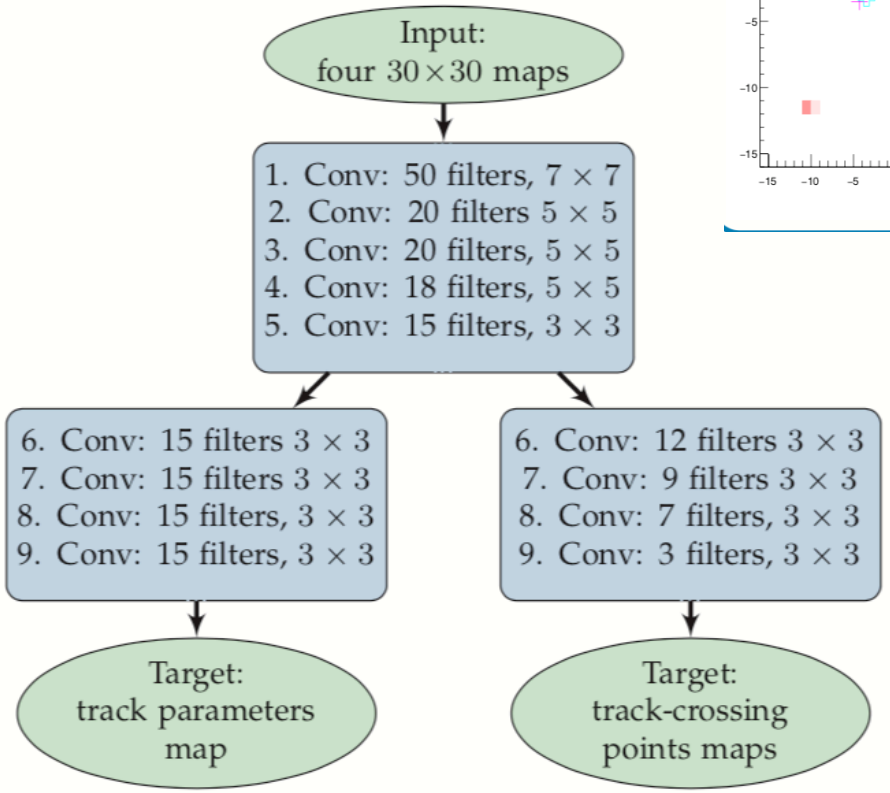
- The problem: when a very collimated jet of tracks hits the CMS pixel detectors, more than 1 track can contribute signal to the same pixel cell
- How to identify this? Currently it is not possible, since local pixel reconstruction is not aware of the global event
  - A global view in outer layers would clarify the situation; but algorithmically that icomes «later»
- Use a CNN using as input the «4 images on the pixel detectors» in order to improve the understanding on the first layers
- Train on Full simulation samples, with MC truth
- Even more → you can use it not only to predict the **correct point of impact**, but directly **track parameters** (**direction, momentum**) and if a given pixel cell shares signal from more than 1 track



V. Bertacchi et al

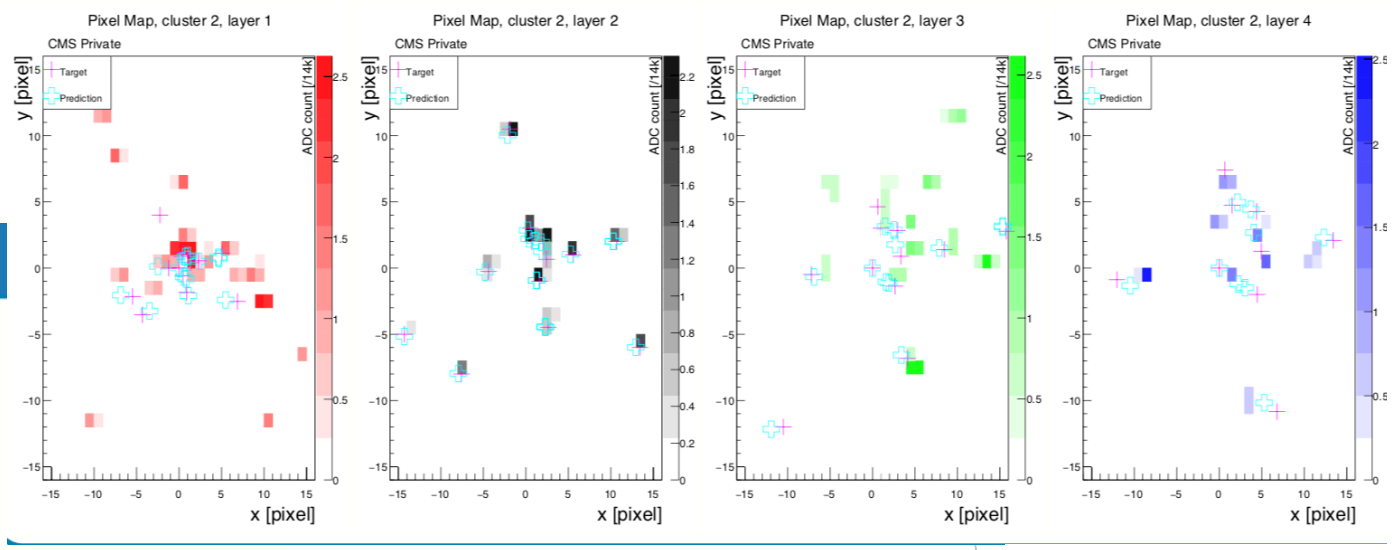
Results

Network Architecture

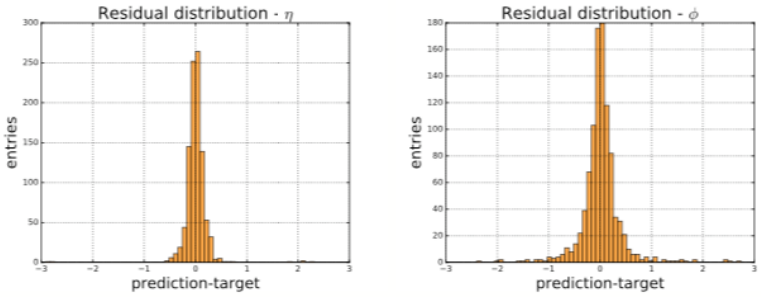


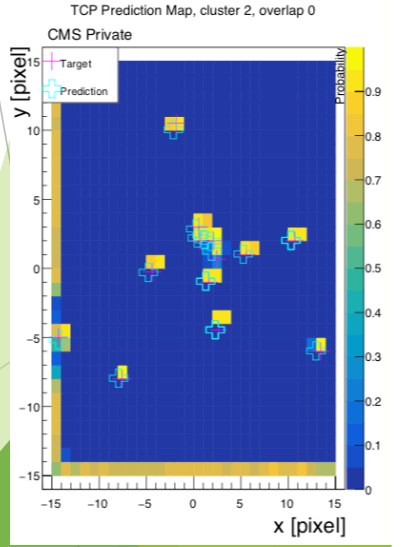
Track parameters
(direction, momentum, …)

Impact point on
pixel dets



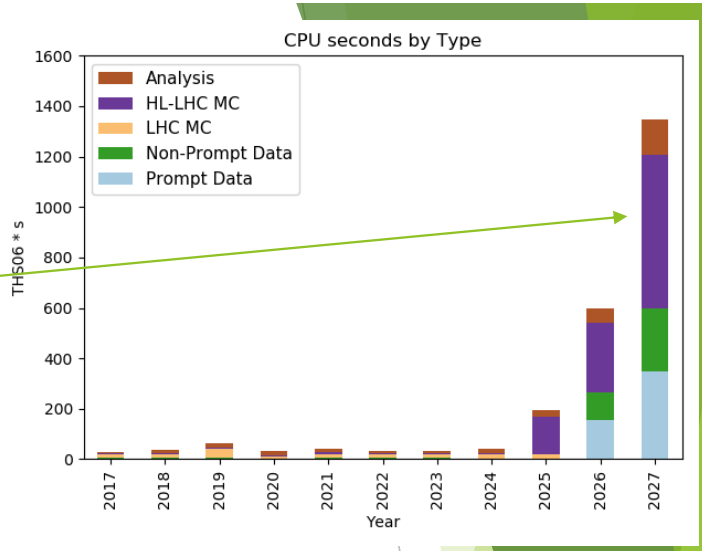
Residual of Angular Parameters:

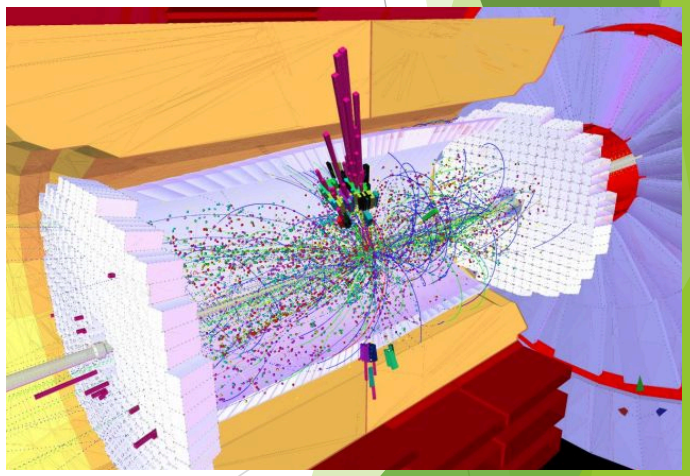




GANs for simulation

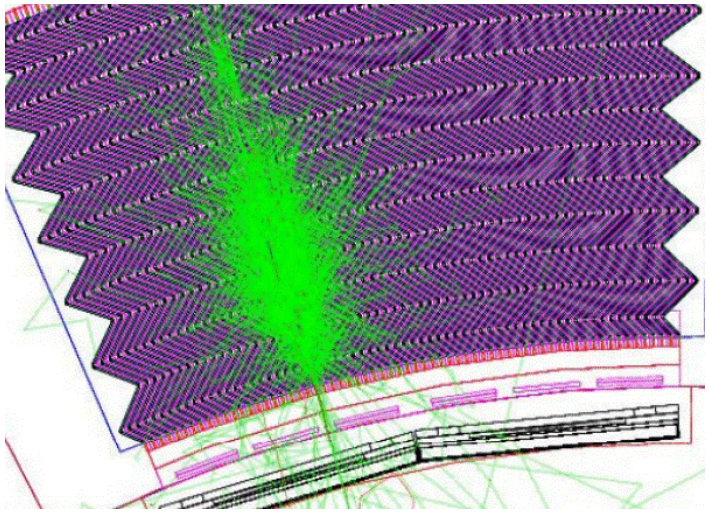
- Going back to the projection plot for CPU, it is clear the **major offender is Monte Carlo simulation**
- It includes
  - The generation of the high energy collision event from theoretical models («generation»)
  - The simulation of the interactions of the primary and secondary particles with the detector (Geant4!)
    - Needs very detailed description of the geometry, + a very detailed simulation of the physics processes down to few MeVs
    - It can be very slow, depending on the size and precision of the detectors you need to simulate
    - CMS: simulation time > reconstruction time (50 sec vs 20 sec today)
- Since you need 1-2 simulate events per collected data events, the large impact of simulation in the overall budget is clear
- How to try and reduce that component?
  - → Fast Simulation!!

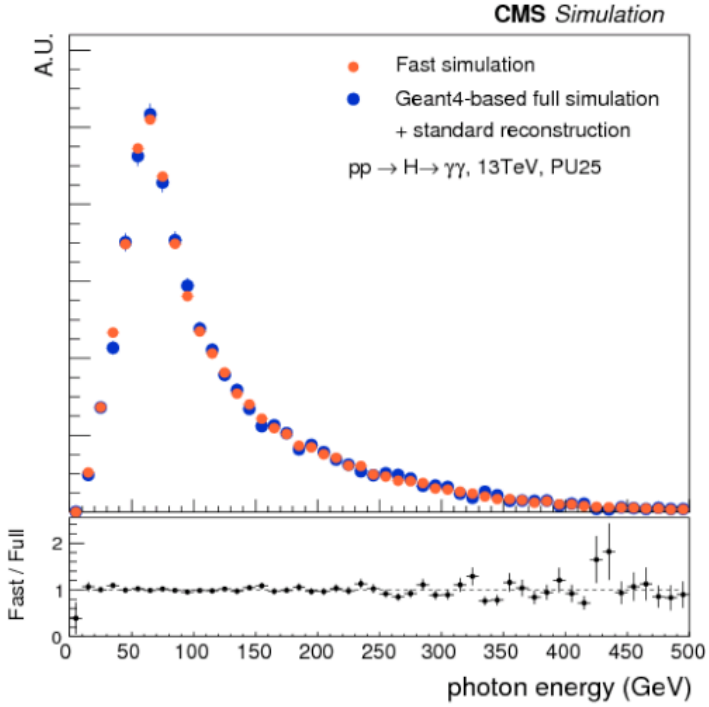




Principles of Fast Simulation

- «Full Simulation» uses low level matter – particle processes to simulate the effect of the interaction
  - Generation of secondary particles, their travel through matter, additional showers, in principle down to few keV (see G4 lessons on cuts)…
  - Taking into account ionization, elastic processes, decays, excitation, hadronic effects , …
  - After all of this, the tota energy deposited is summed and gives the raw energy response of a given detector
- Fast simulation tries to compute the final quantity directly from
  1. **A parametrization**
  2. **A fit to full simulation or test beam data**
  - It is fast since there is no particle explosion or propagation, a single formula suffices
  - It is as good as the fit /parametrization makes sense
    - Usually very good for the bulk of events, but unable to reproduce particular parts of the phase space
  - (Some Fast Simulation approaches are even more inclusive, and try and generate directly the **reconstructed quantities, not only to mimick G4**)

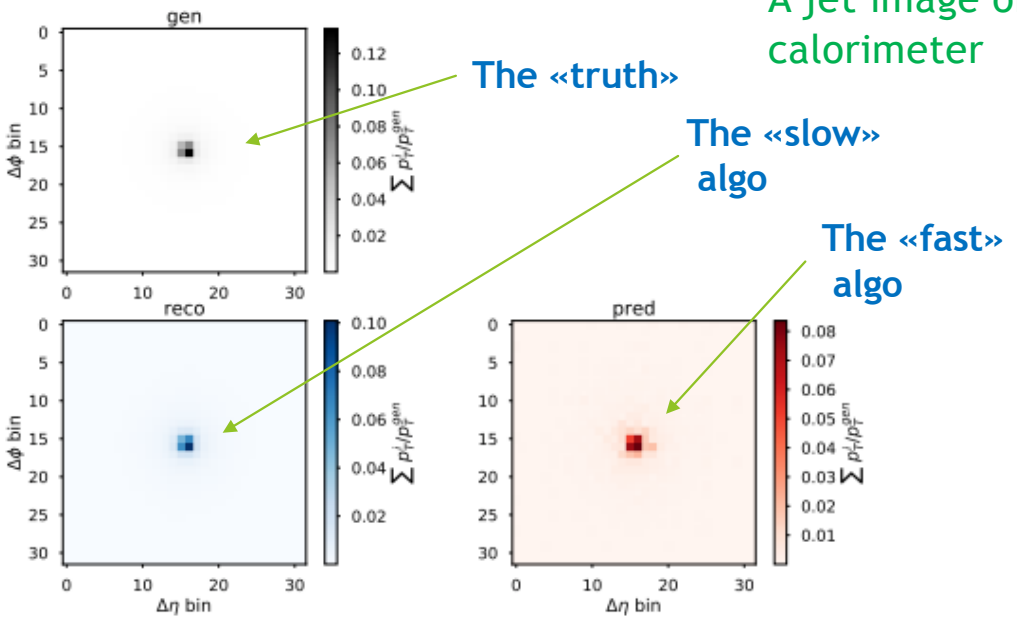


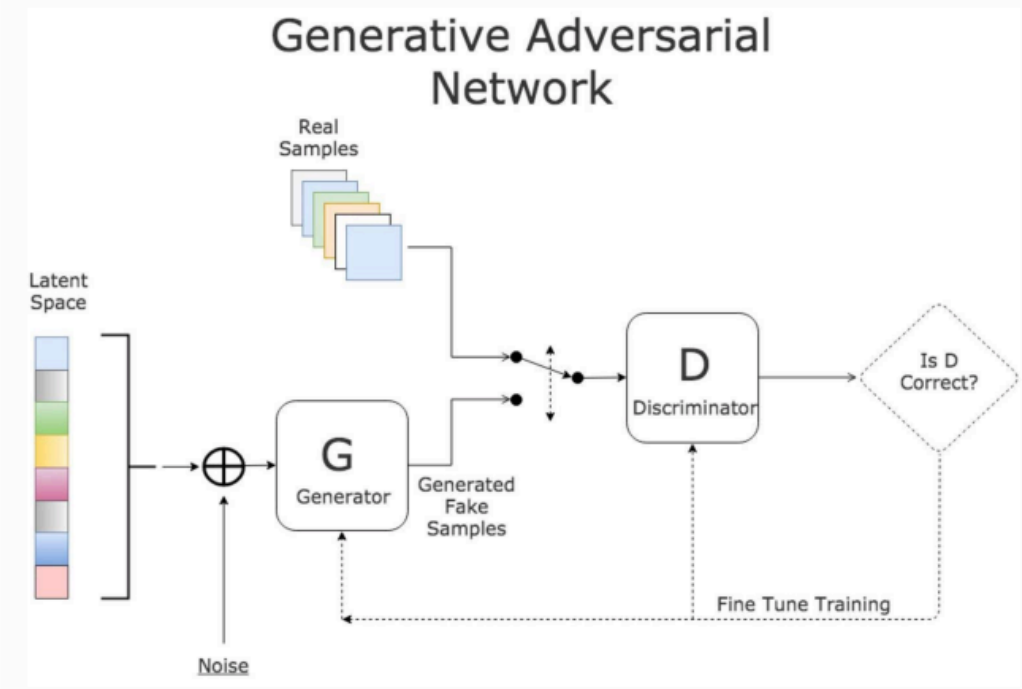


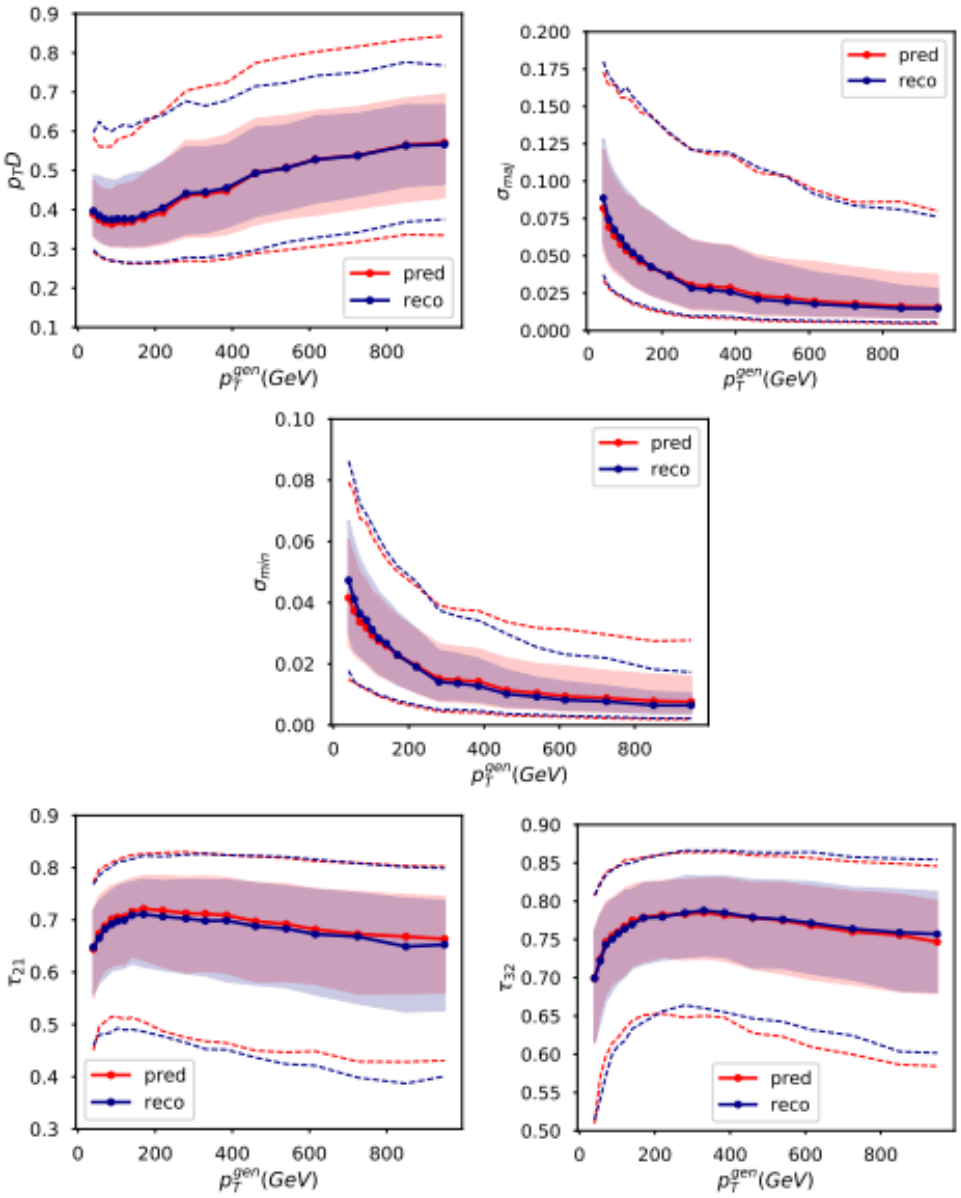
Fast Simulation with Generative Adversarial Networks

P. Musella et al

- Jet reconstruction







Full vs Fast

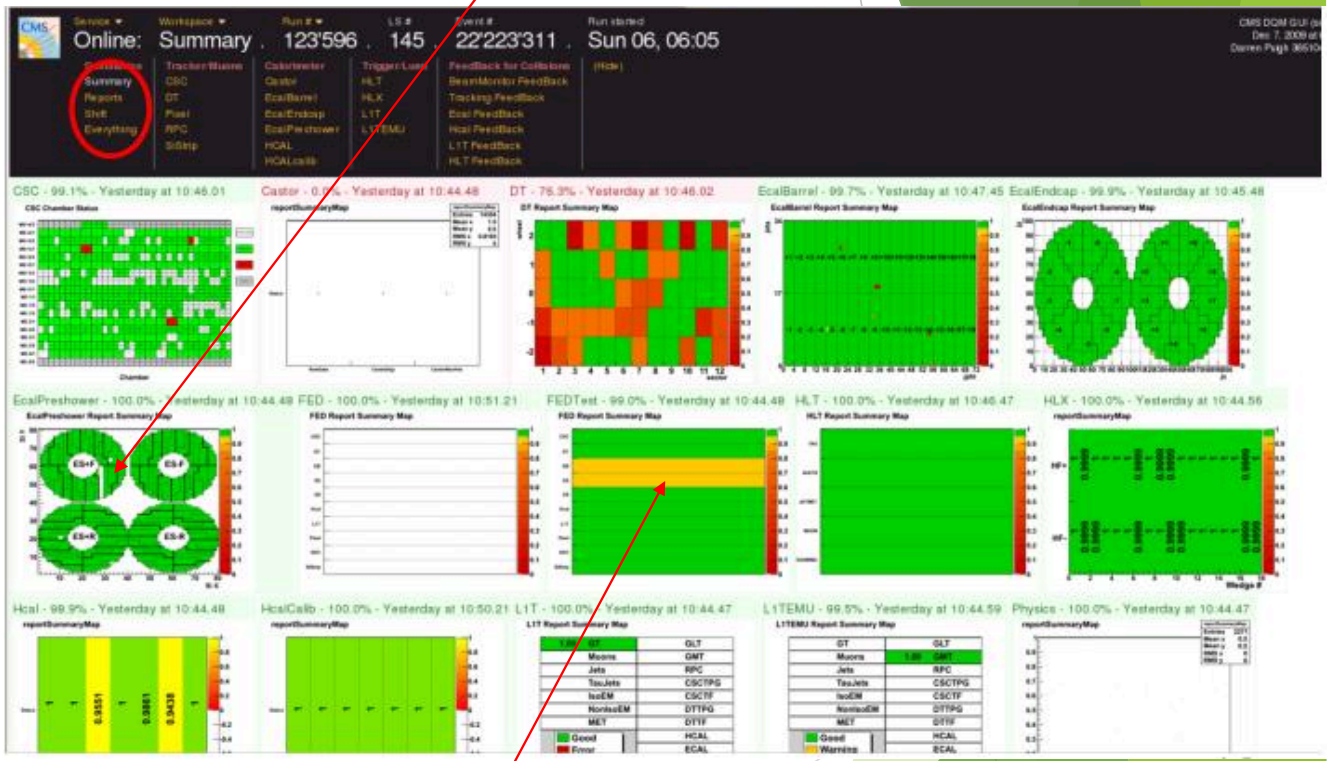
If one could use these jets for physics, timing would be stellar:

- **Full algo**: ~ 1 Hz per jet (simulate particles, their response in calorimenter, jet finding, reconstructed energy)
- **Fast algo**: ~ 10 kHz on a Nvidia P100

- Hidden facts:
  - Training is slow (tens of hours)
    - But it is way faster than writing an algorithm by hand
  - Very good results; but still more a proof of concept than anything else.
  - No commitment to use GANs as substitution of Full Simulation

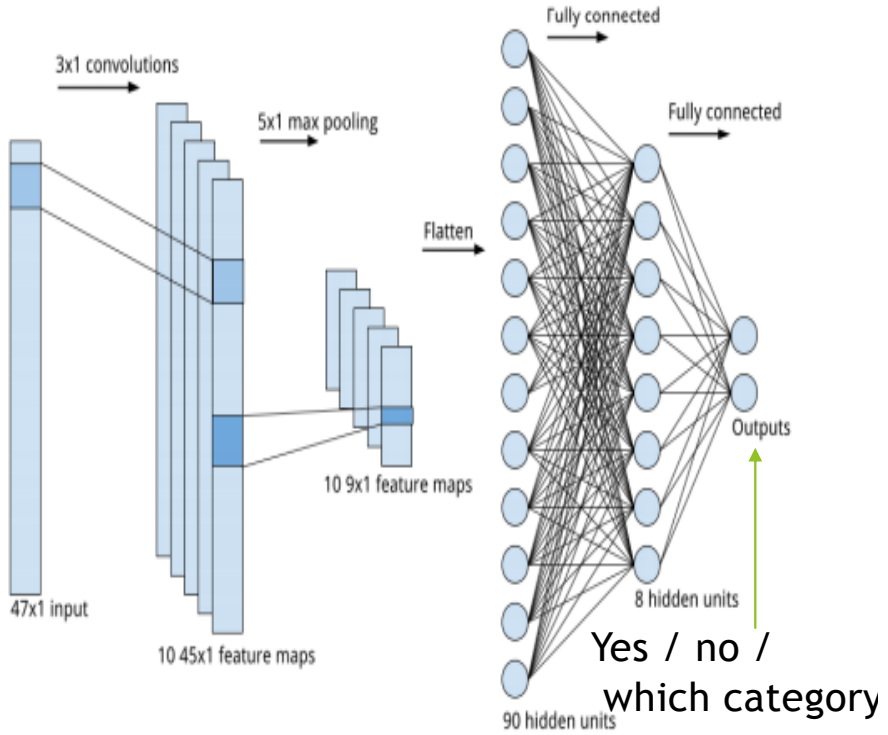
ML for Data Quality

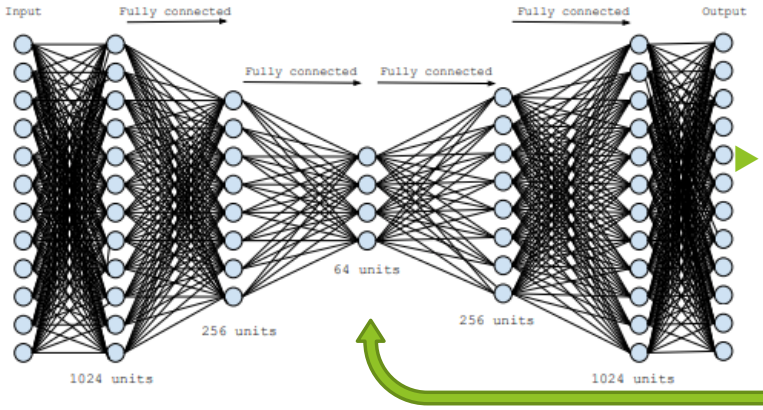
- During beam operations, at least one person needs full time to look into some typical detector plots in order to see that (for example) a part of the calorimeter did not go off, or that there is abnormal noise in another part
- Can this be substituted with a person? What a trainer «data quality certifier» does is:
  1. Search for known problems (he/she has been taught typical failure modes, in order to recognize them)
  2. Search for unknown problems (spotting something unexpected, which is «strange enough» to pass a «personal» threshold)
- A person is trained to the task by working as shadow for some time with an experienced shifter → a classical example of training!



Supervised vs unsupervised ML approaches

- Search for «known problems»: classical pattern recognition problem, trained with the «correct answer» (problem yes/no) for both normal and problematic images
- Search for «unknown problems»: Train only on «normal» behavior, and let the ML encode it internally (autoencoder net)



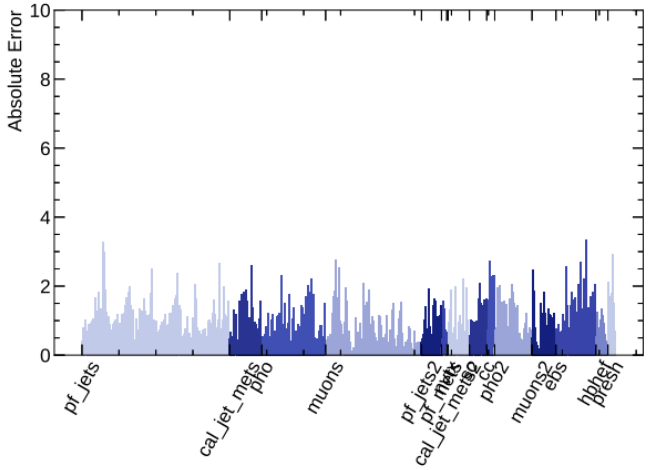


At that point, the net will have difficulties encoding something different → the output will be largely different from the input

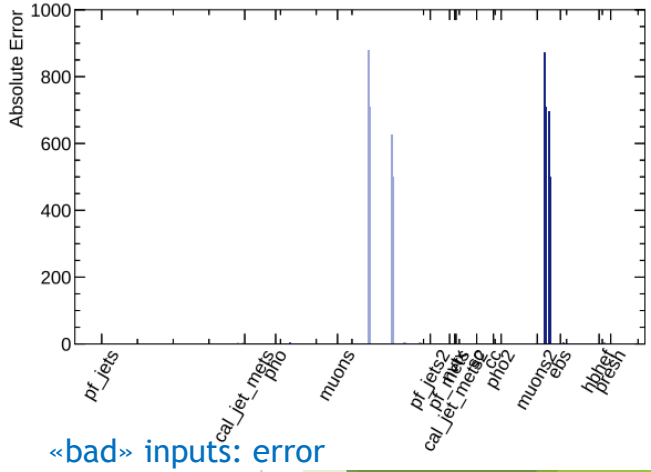
Internal representation of the event, with fewer dimensions

A.Pol et al

Data quality monitoring via autoencoders

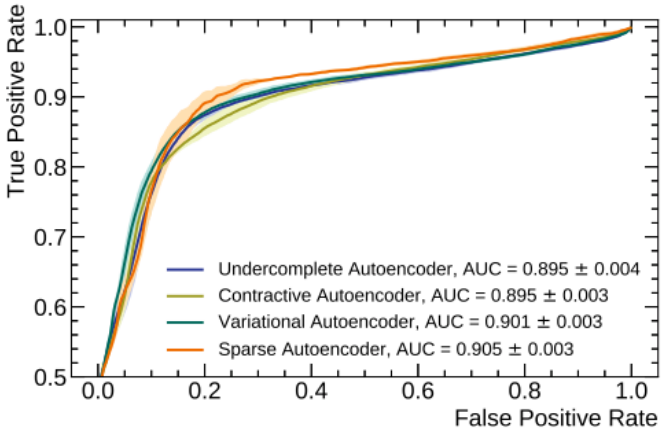


«good» inputs: error is low



«bad» inputs: error is high
(and which variables are specifically high tells you where the problem stays

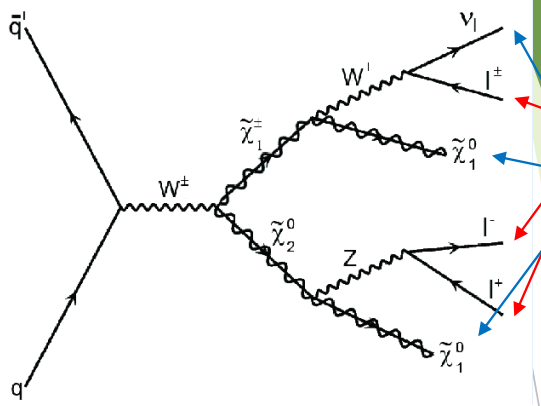
- Give as input the main features used in monitoring a CMS event
  - 401 variables (# jets, #tracks, # hits, # muons, …), 7 numbers x variable (rms, men, 5 quantiles)
  - Train the system on 5 months of monitoring (a value each 23 sec) which has been already declared good by humans
  - Test on all the data from these 5 months, including the bad ones
  - Test the match between what humans declared bad and the autoencoder response + some additional good data; the autoencoder tells you the «error» in the encoding
    - How far a the output was from the input
- Not production ready, but aimed to Run-III as a way to reduce monitoring manpower



Anomaly detections for physics?

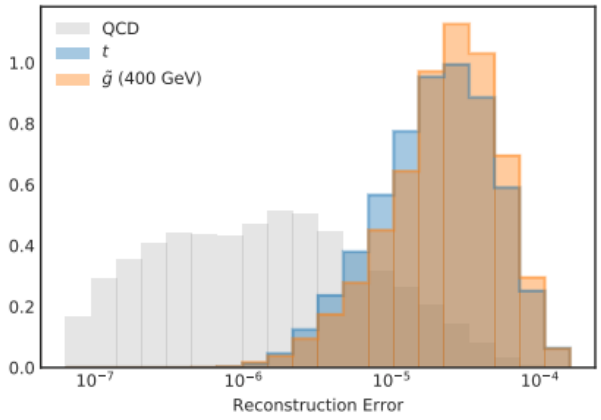
- Only a wild idea at the moment – supported by some theoretical papers
- In standard new physics searches, you search for a physics signatures for a specific model
- What is there is physics you did not think of? (well, do not worry, we are somehow already covered)
- **Train an autoencoder on «standard physics» (hard to define, but assuming that new physics is VERY rare, it is not an issue)**

Farina et al.



Search events with
- 3 leptons
- Missing energy

Proof of principle: if trained on QCD, error is high on top and gluoino events (all from simulation here)



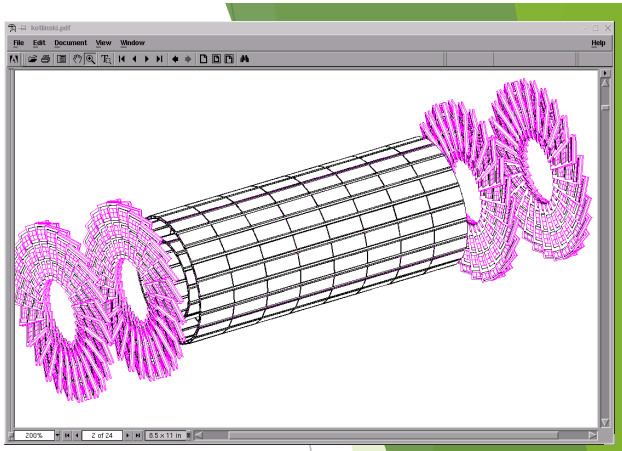
DeepLearning as a complete replacement of «every algorithm»

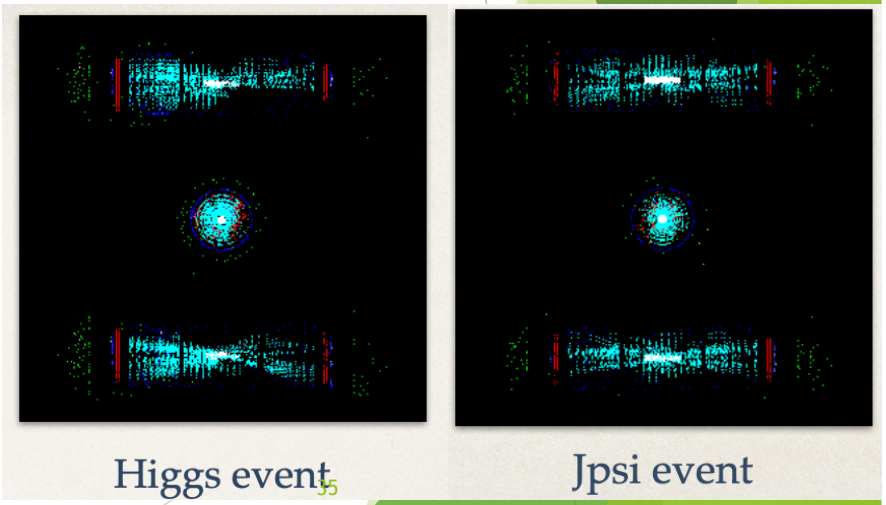
- The algorithmic steps we need to go from RAW detector data («a bunch of 01001010», 1MB sized) to the selection of different types of events («does this event contain the decay products of an Higgs boson?») are many
  - Take the RAW data for every subdetector, and interpret them as local signals (hits on a silicon detector, energy in a calorimeter cell)
  - Use local signal from different subdetectors, to form a global object (a track, a jet)
  - Consider all the objects in an event, and try and understand the topology; evenrtually, find an Higgs boson decaying to something
    - Involves statistical methods, understanding of the different topologies from physics processes, …
- Up to here: understand the workflow in terms of smaller algorithms, try and replace some of them with ML. What if …

- … **we try the longest possible step**: train a very large ML system to start from RAW detector data, and give a final event characterization (Higgs/notHiggs)?
- It is defintely too early; for example how to feed 100M inputs to a network? Hot to have enough events for training?

- Still we can try with a simplified model:
  - clean events (no pile up)
  - only tracking detectors
  - Reduce granularity of input

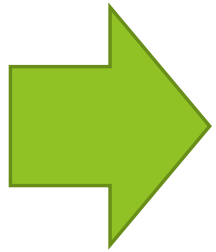
- Idea: **take pictures of the hits in the CMS tracker**, from different views (xy, xz, zy), as lowish resolution Jpegs (to reduce the # of inputs)
  - 300x300 pixel images = 90k (sparse) inputs

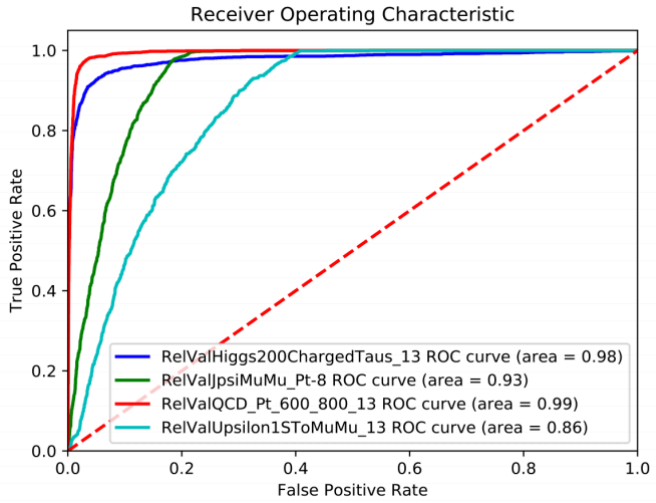




Event categories and results

- Use 4 event categories
  - Higgs decays tu tau leptons — Medium event complexity
  - QCD (strong interactions) — High event complexity
  - Jpsi decays (to 2 muons) — Low event complexity
  - Upsilon decays (to 2 muons) — Low event complexity
- And train with ~10k simulated events per category





(each category against all others)

- It works! Are we done? No…
  - Model very simplified
  - Never tried on complex events (add 35 pp interactions and see..)
  - Eventually, who would trust the result now?

Conclusions

- ML today is for CMS, apart from a few tools deployed in production, an R&D field; we expect its utilization to become more important by RunIII and eventually on the critical path on RunIV
- It is a field which evolves fast, GANs are now ubiquotous and it is difficult to believe they were proposed in 2014. Somehow complex to stay up-to-date
  - Not described here, but GraphNets are now studied as a general tool for HEP, and they are from 2018.
- In general, we foresee the need for different expertise in the coming year: less «analysists», more «data scientists»
  - But there we are in competition with the private sector
- We still have an advantage over many other sectors trying ML approaches: we literally have Petabytes of data on which to train; we just need to understand HOW to evolve our systems
- **Will you join us?**