Fellini
Fellowship for Innovation at INFN

H2020 MSCA COFUND
G.A. 754496

INFN
Sezione di
Roma Tor Vergata

Lessons learned on FastCaloGAN

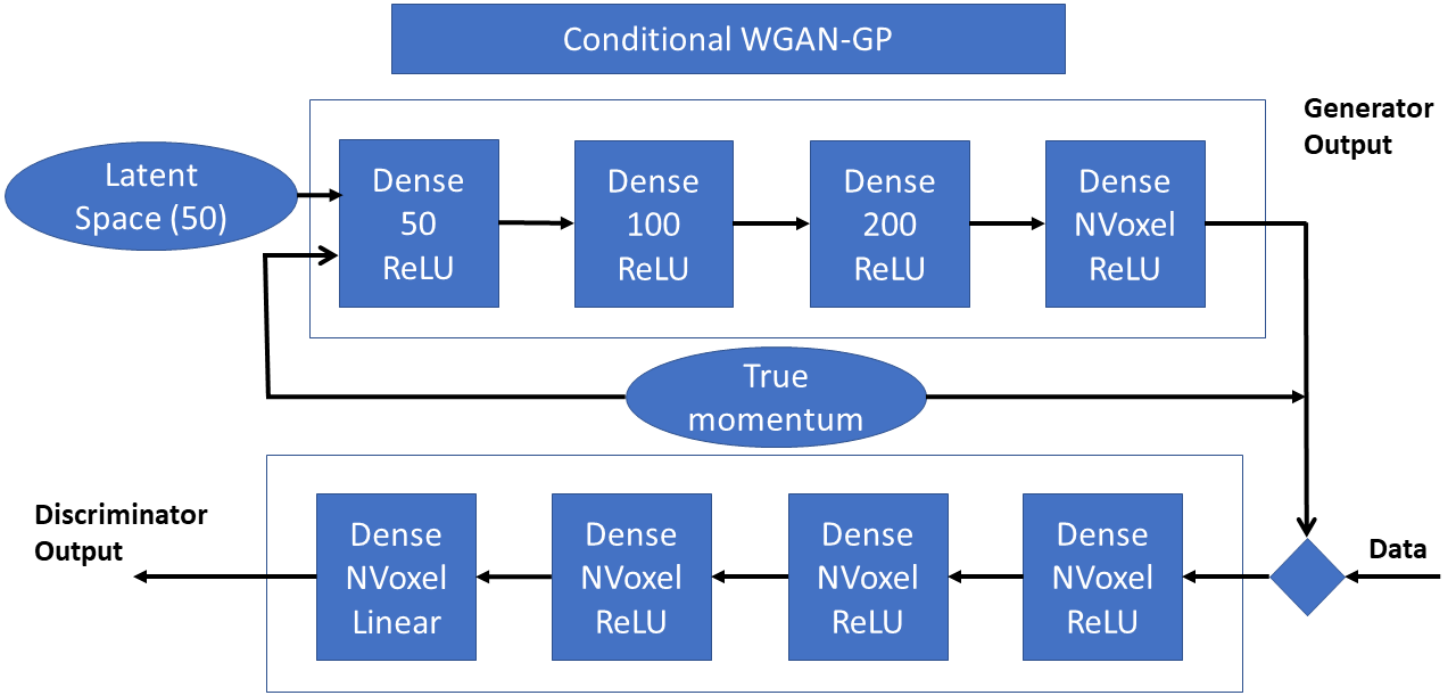
Michele Faucci Giannelli

22-04-2021

FastCaloGAN in one slide

- Use the 4500 single-particle samples produced by ATLAS as part of the effort to create a new Fast Calorimeter Simulation (FCSV2)
  - 3 particles, 100 η slices, 15 energy points per slice
  - Energies from 256 MeV to 4 TeV (in powers of two)
- Define a GAN for each particle in each η slice → 300 GANs
- Train the GANs on voxelised hits (see backup for hit definition)
  - Because cell structure is not homogeneous and would require different GAN architectures, voxelisation allow to reduce the differences
- Select best epoch based on total energy distribution of each sample
- Convert the selected generator into LWTNN
- Simulate hits in Athena inverting voxelisation
- Compare with Geant4 using high-level observables for single-particle and di-jet samples after reconstruction
- More information:
  - Pub note: ATL-SOFT-PUB-2020-006
  - Presentations at IML

The GANs



G	50 (Input latent Space), 50, 100, 200, NVoxel (pid and η dependent)
D	NVoxel, NVoxel, NVoxel, NVoxel, 1
Activation function	ReLU (in all layers)
Optimiser	Adam [21]
Learning Rate	10^{-4}
β	0.5
Batchsize	128
Training ratio (D/G)	5
Gradient penalty λ	10

NVoxel depends on PID and η

Additional training information 1/1

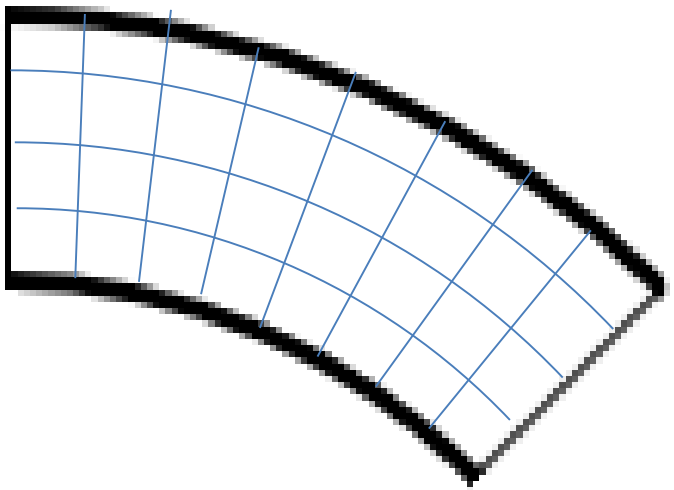
- Pre-processing of data:
  - Normalised the energy in the voxels by the true momentum of the event
  - Normalised all labels by the highest momentum
- Training strategy:
  - Training using all samples together was not producing good results
    - But we could train a single energy non-conditional GAN
  - Adopted an “incremental” training:
    - start with a single sample and train for 50k epochs
    - Add a new energy to the training mixture every 20k epochs
      - Training order: 32 GeV, 64 GeV, 16 GeV, 128 GeV,…

Additional training information 2/2

- Training is carried out for 1M epochs
- Checkpoints are saved every 1k epochs
  - Allows to select the best epoch in a second step improving on the selection criteria
- We used the GPU on HTCondor and the training time was ~8h per GAN → 100 day GPU time to train the full detector

Voxel to hit

- The voxelisation must be inverted in the Athena service by producing hits
- Hits are generated by sampling the area of each voxel
- The energy in the voxel is divided uniformly among the different hits



Lesson learned: scope

- While it seams a good idea to proceed by steps, there is indeed little benefit in doing so
  - We started with a GAN trained on a single energy point (no conditioning on energy)
  - Moving to a conditional GAN took a lot of time and we had to change many parts of the code (relatively easy) and the structure of the framework (much more time consuming)
- Better to define your goal and work directly toward it
  - The pre-processing code was rewritten at least 3 times as the goals changed
  - Optimise the parameters only once
    - Testing parameters can take a lot of time, because you want to be sure that it's not something on the ML side, but see later..

Lesson 2: R&D vs Full detector

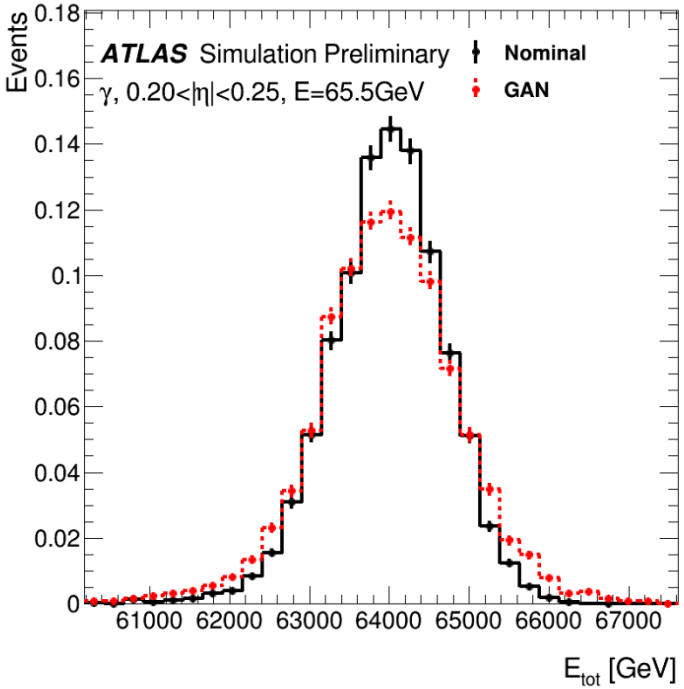
- With R&D (i.e. a single eta slice) you can do things that do not scale well to the whole detector
- Plan as much as possible considering the final scale of the project, this applies both to time to train and resources used
  - We were saving too many files (all the checkpoints, root files, images) and we broke the 2M files limit on EOS
  - Optimised HP would drive the training time from 100 to 300 days!

Lesson 3: ML vs domain knowledge

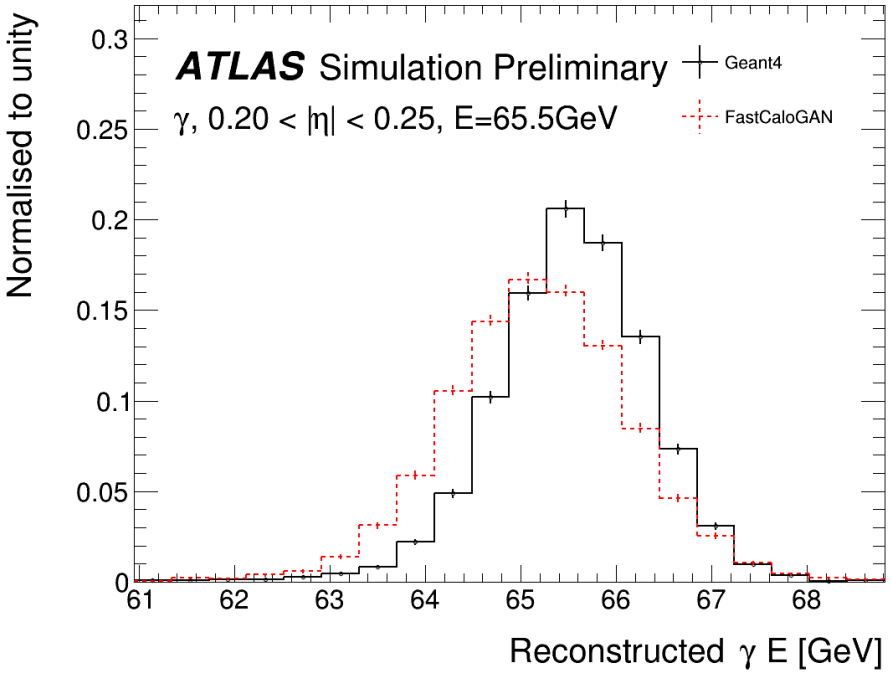
- If there is a problem or the performance are not as expected, the cause is almost never the ML
- You need to understand your data and make a solid analysis of it before starting the “fun” ML part
- In the case of simulation, there is also the need to understand better the “voxelisation inversion” problem
  - How to assign energy from voxels to cells is not trivial at all, again knowing the details of the detector, simulation and reconstruction is fundamental
- Be open to use non-ML solution:
  - parametric approach can be better than GAN
  - Very low energy primary done in G4 as they are fast and more accurate
- Bonus lesson from DiJetGAN: exploit symmetries and simplify the problem for your GANs, do not hope that they will learn symmetries (GANs are bad at it)

Example: photons

Energy of sum of voxels



Energy of cluster



The reconstructed photon energy is wrong because there are two energy corrections that needed to be applied but we did not know. Fixing it improves the agreement significantly

We need to get an even better agreement than the one shown for voxels, hence it is important to work on HPO and all other ML stuff too

Lesson 4: ML can improve the results

- HPO was performed using a grid search.
  - The χ^2/NDF 16→11 but tripled the training time
  - NB: there was no global minima in a multi-dimensional plane but we observed some trends common to all particles and detector regions
    - Good because we want to keep the GANs similar to have a simple system
  - More parameters could be tuned (not done as they worked so far)
- Optimising the binning can improve the learning
  - This also affects voxelisation inversion, not simple
- Adding more information degrades the learning of the energy
  - A possible solution is to have multiple loss
- Version of backend improve results
  - The code was written in TF1 but running with TF2 is faster and better
- Being smart on the training strategy can save time and improve performance
  - The incremental training worked well
  - Having a two-step training is also being investigated and is very promising solution

Backup

The training sample

- The single-particle samples have:
  - detailed hits (i.e. with a step << cell size) to better map ATLAS cells, the position is stored
  - without z-vertex spread
  - without noise
  - without parts of the electronic cross talk
- Events generated by momentum
  - But used the Ekin for the simulation
- The statistics in the high energy region is lower than at low energies
  - 10k events up to 256 GeV
  - Drop to 1k events for 4 TeV samples

Voxelisation

- Hits are transformed from ATLAS (x,y,z) coordinates to cylindrical (r, α, R) coordinates
  - R is not use, as hits are grouped in layers
  - Same procedure used in FCSV2 with two differences:
    - Alpha is not symmetrised (we want to keep the correlations between voxels)
    - # of bins in alpha is 10 instead of 8
    - We do apply both symmetrisation by charge and in η
- GAN cannot be trained on hits, so they are grouped in areas in the (r, α) plane in each layer
  - each volume in the (r, α, layer) space is called a voxel
- Voxels are optimised to:
  - Have enough energy to avoid large event-by-event fluctuations
    - Binning in α only for layers with a large energy deposits (i.e. EMB1 and EMB2)
  - Contain all the energy in a shower
  - Reproduce the Energy Centroids in η and φ

$$EC_\eta = \frac{\sum_i E_i \eta_i}{E}$$

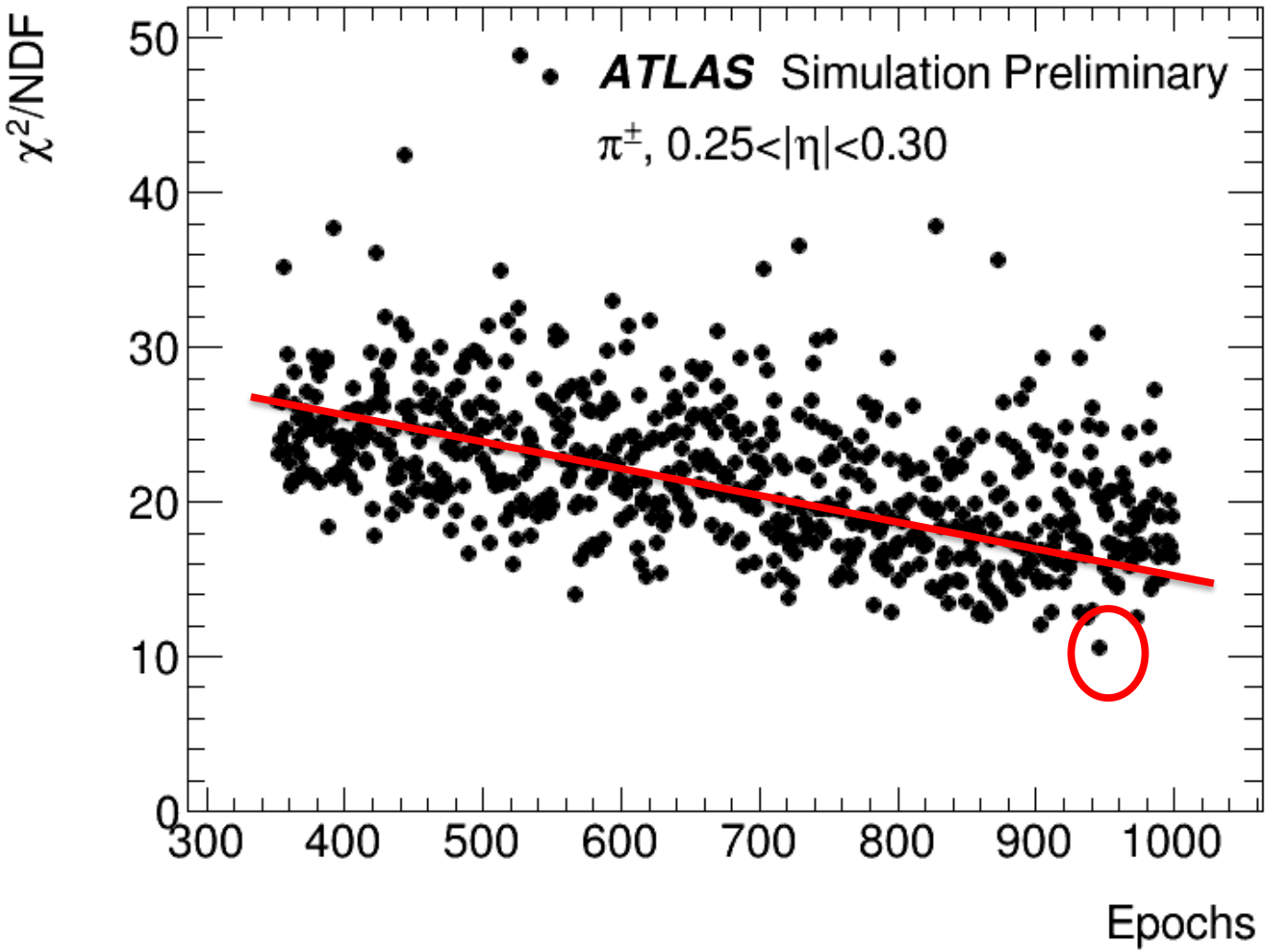
The training sample and voxelisation

- The single-particle samples are “special”
  - detailed hits to better map ATLAS cells
  - No noise
- The statistics of the samples:
  - 10k events up to 256 GeV
  - Drop to 1k events for 4 TeV samples
- Hits are transformed from ATLAS (x,y,z) coordinates to cylindrical (r, α, R) coordinates
  - R is not use, as hits are grouped in layers
- GAN cannot be trained on hits, so they are grouped in areas (voxels) in the (r, α) plane in each layer
  - The size of the voxels was optimised

Choice of best epoch

- The training of a GAN is not a minimisation process, so there is no assurance that a later epoch will give a better result than an earlier one
  - You could see “fluctuations” during the training
- Choosing an epoch is non-trivial, we have to compromise on many distributions
- For this version of FastCaloGAN, we used the total energy of the 15 samples. The χ^2 between G4 and the generated events is calculated for all sample. The sum of the χ^2 is used as figure of merit.
- The epoch with the smallest χ^2 is used

A summary of the training



Chosen epoch

Integration in Athena

- NB: All figures until now are done at voxel level
- To assess the real performance of FastCaloGAN, integration in Athena is required
- This is done through a conversion of the generator from TF to Keras and then LWTNN
- The LWTNN inputs are open by the FastCaloGAN simulation service
- The simulation configuration used for the physics sample pass back to Geant4 e/gamma and pions below 200 MeV and k/p belowe 700 MeV in momentum
- Simulation time for a pion at the calo surface is 2 orders of magnitude smaller than Geant4
  - 70*ms* vs 6*s* for 65 GeV or 162*s* for 2 TeV
- Memory consumption is small, only 2.5 Gb is used by a job
  - We could easily add more GANs, i.e. other hadrons