# Reduced Precision Strategies for Deep Learning: 3DGAN Use Case

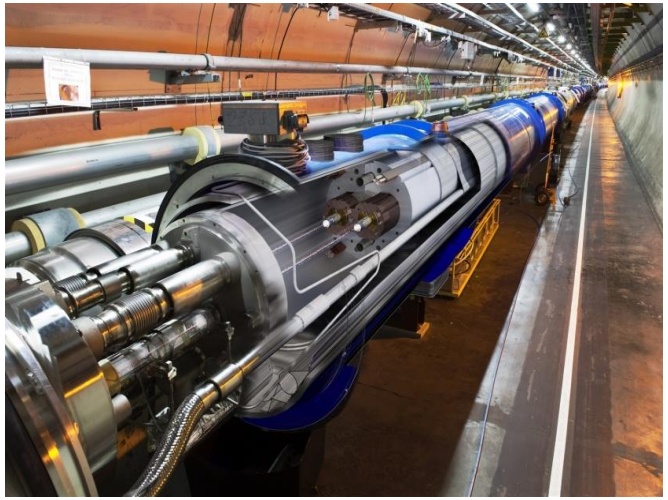
*10th ICPRAM Conference 2021*

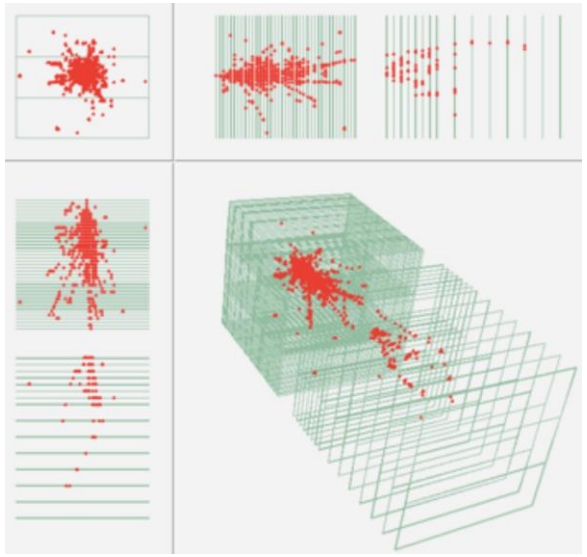
Florian Rehm [CERN openlab, RWTH Aachen University]

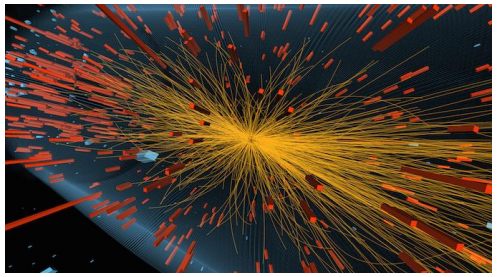
Sofia Vallecorsa [CERN openlab], Vikram Saletore [Intel], Hans Pabst [Intel], Adel Chaibi [Intel],
Kerstin Borras [DESY, RWTH Aachen University], Dirk Krücker [DESY]

# Calorimeter Detectors

- Particle physics studies the fundamental properties and interactions of (novel) elementary particles.
- Particles are colliding with highest energy to produce know and novel particles.
- Calorimeter detectors measure the energy of particles produced in the collisions.
- The entering particle produces a cascade of shower particles which are absorbed and detected.



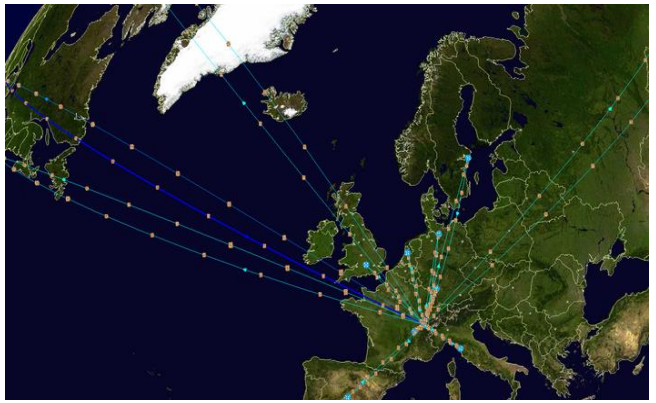




# Calorimeter Simulations

- Calorimeter Monte Carlo simulations are based on Geant4 which are computing resource intensive when used in detailed geometry and particle tracking
- They use about 50% of the computational resources of the worldwide LHC grid
- LHC high luminosity phase requires 100 times more simulated data*

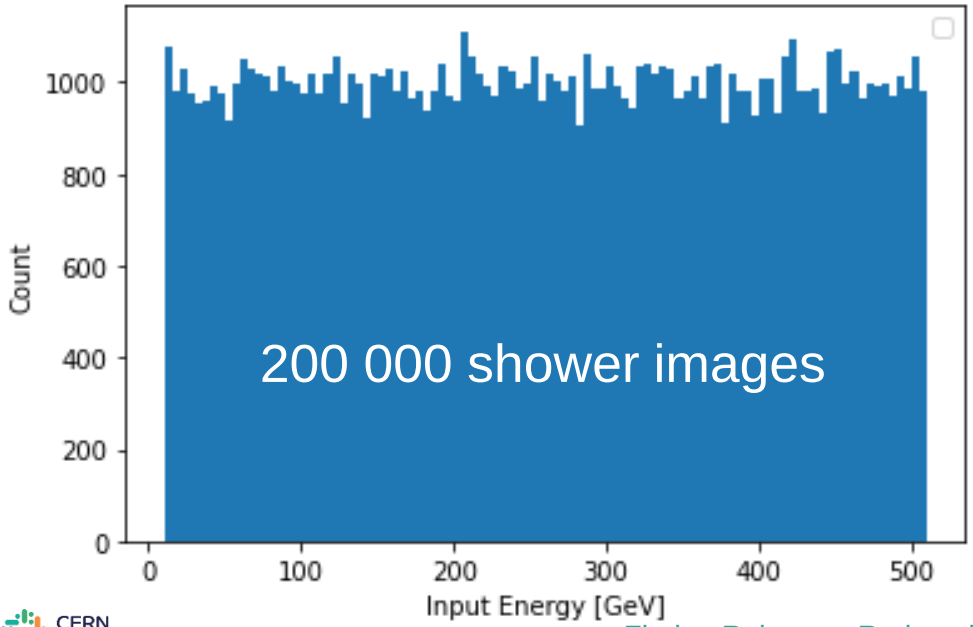
*A Roadmap for HEP Software and Computing R&D for the 2020s
https://doi.org/10.1007/s41781-018-0018-8

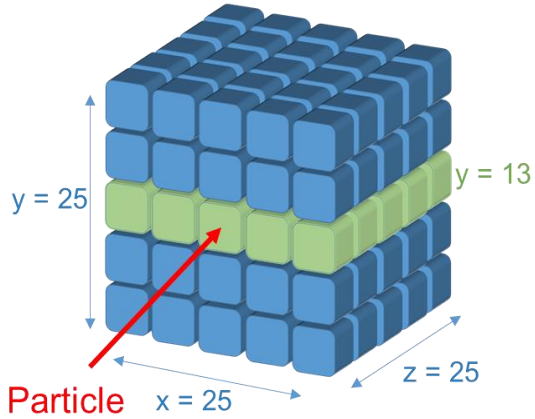


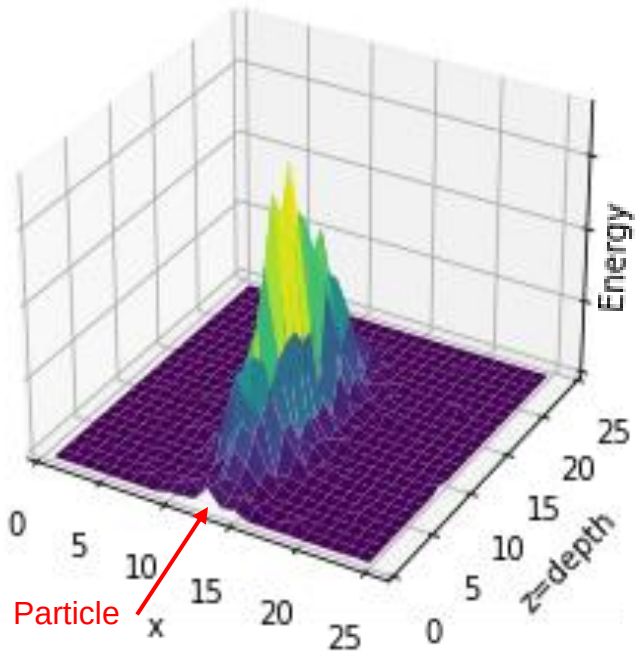
**→ Develop a new approach which occupies less resources**

# 3D Training Data

- Interpretation of the calorimeter outputs as images
- 3D shower image granularity: 25x25x25
- Energies between 2-500 GeV



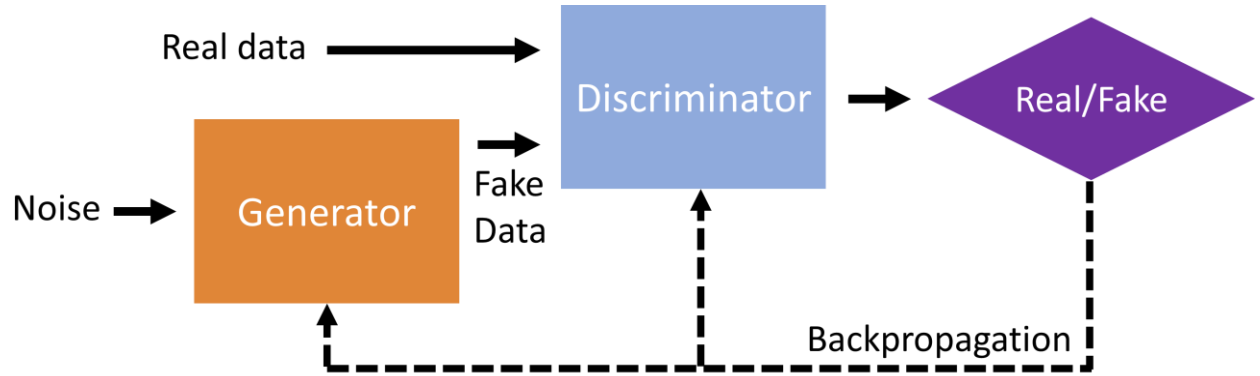



# Generative Adversarial Networks

*3DGAN*

- Train two networks (Generator & Discriminator) in a minmax game
- GANs reach a good level of accuracy*
- ➢ We want to further decrease the computational resources
- Only the generator network is used to generate shower images

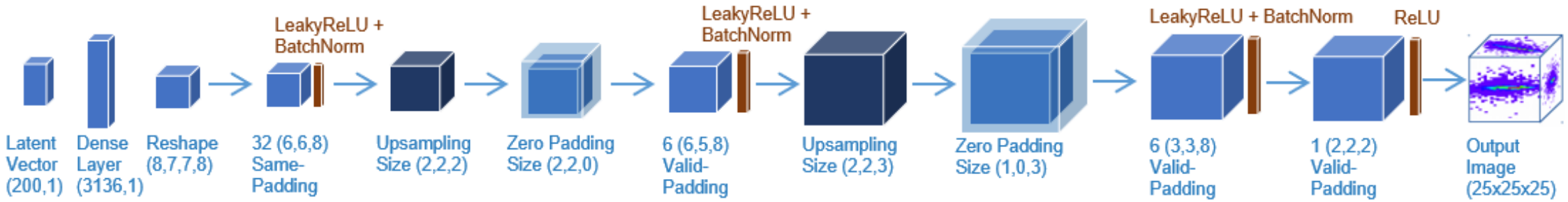



*G. R. Khattak, et al., ICMLA 2019 Particle Detector Simulation using Generative Adversarial Networks with Domain Related Constraints

# Baseline Conv3D Generator

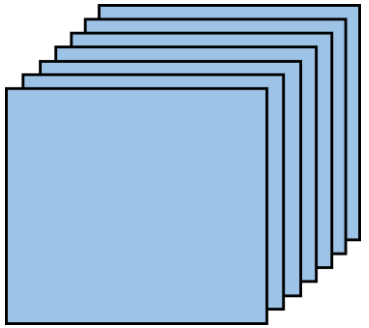
- Until now: Representing 3D images → Using 3D convolutional layers

Paper: Three dimensional Generative Adversarial Networks for fast simulation, Carminati et al., 2018

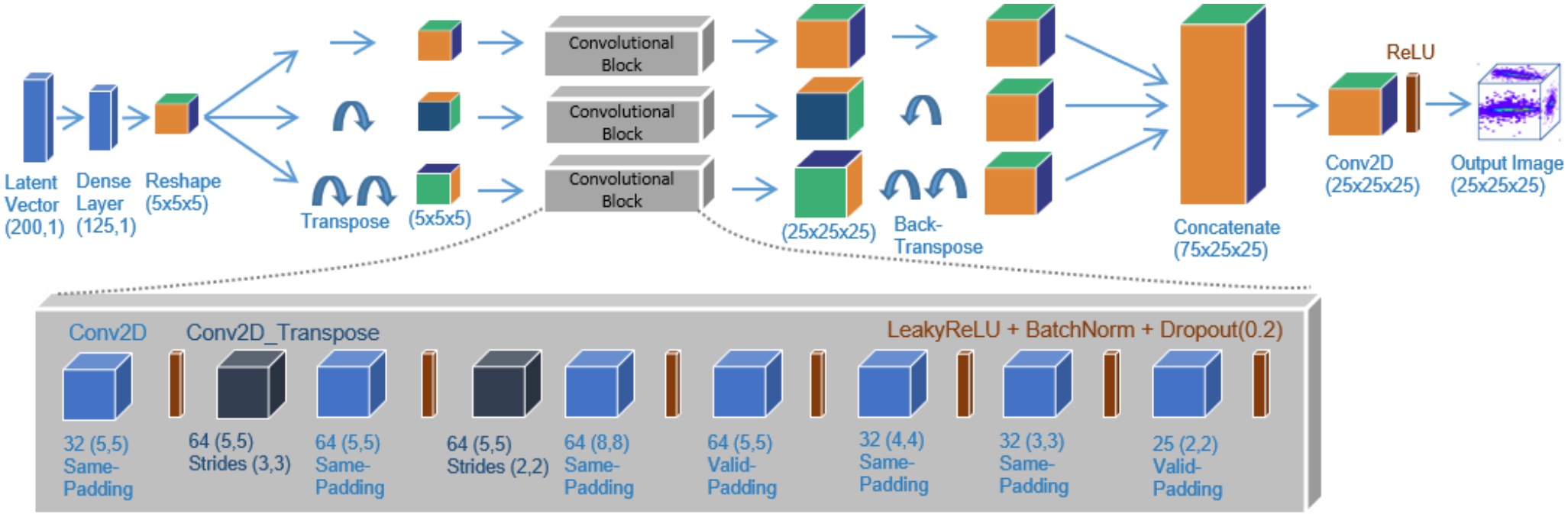



- Conv3D layers are not supported in lower precision

  → Creating neural network consisting only of Conv2D layers
  - First approach: Channel dimension as 3rd dimension
    - Bad accuracy



# New Conv2D Generator




- Solving a 3D problem with 2D layers
- Increasing the number of parameters → more powerful network → higher accuracy
- **2.1x speedup** on CPU

# Why Reduced Precision?

- Goal: Develop and optimize a new simulation approach to make sure to use the hardware as efficiently as possible

- Deep Learning training and inference are computationally intensive
  - Models need a large amount of memory
  - Moving data to and from the processor units strains the bandwidth

→ Reduced precision computation reduces memory and bandwidth occupation

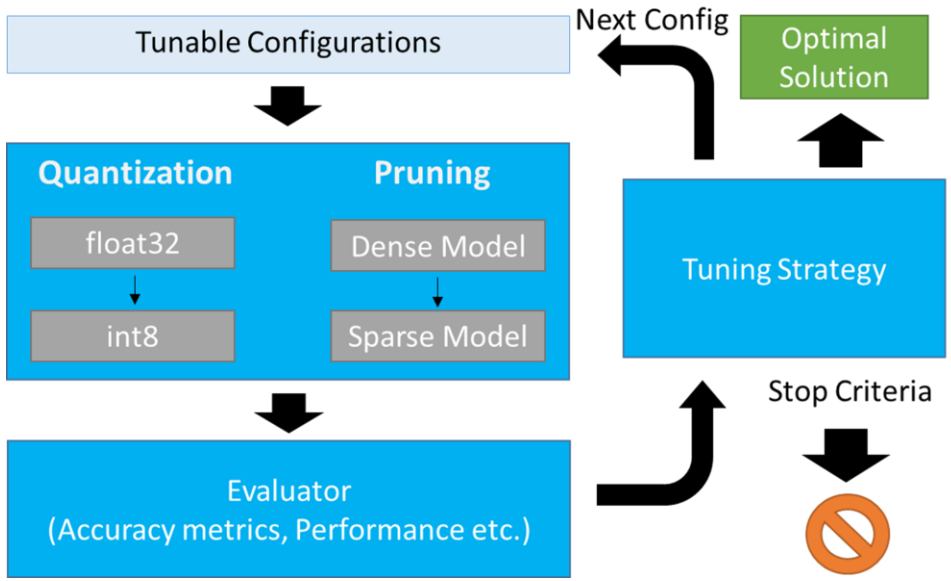
→ Speed-up and lower memory requirements

# Reduced Precision Computing

- Quantization: Converting a number from a higher to a lower format
  - E.g. from float32 to int8

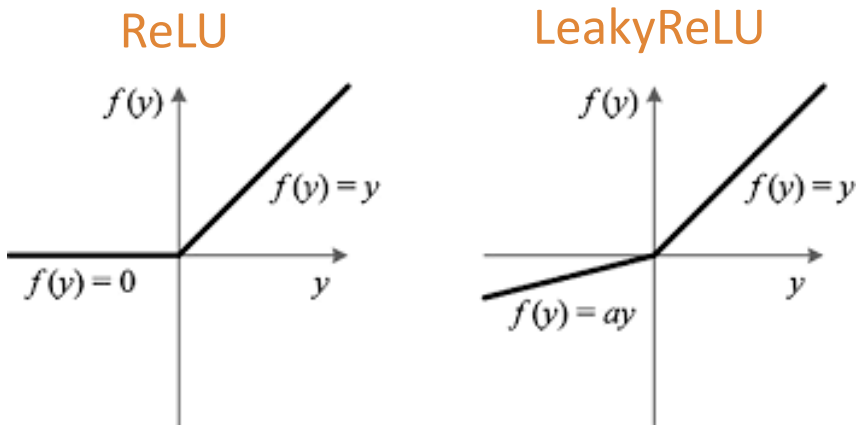
| Float32 | | Int8 |
|---|---|---|
| 4 byte | → | 1 byte |
| Max Number: $3.4 * 10^{38}$ | | Max Number: 255 |

- Quantization Tool:
  Intel Low Precision Optimization Tool (iLoT)
  https://github.com/intel/lp-opt-tool



# Quantization Problems

- TensorFlow supports no negative quantized values (signed int8)
- → All quantization tools do not support LeakyReLU function
- Needed to be implemented



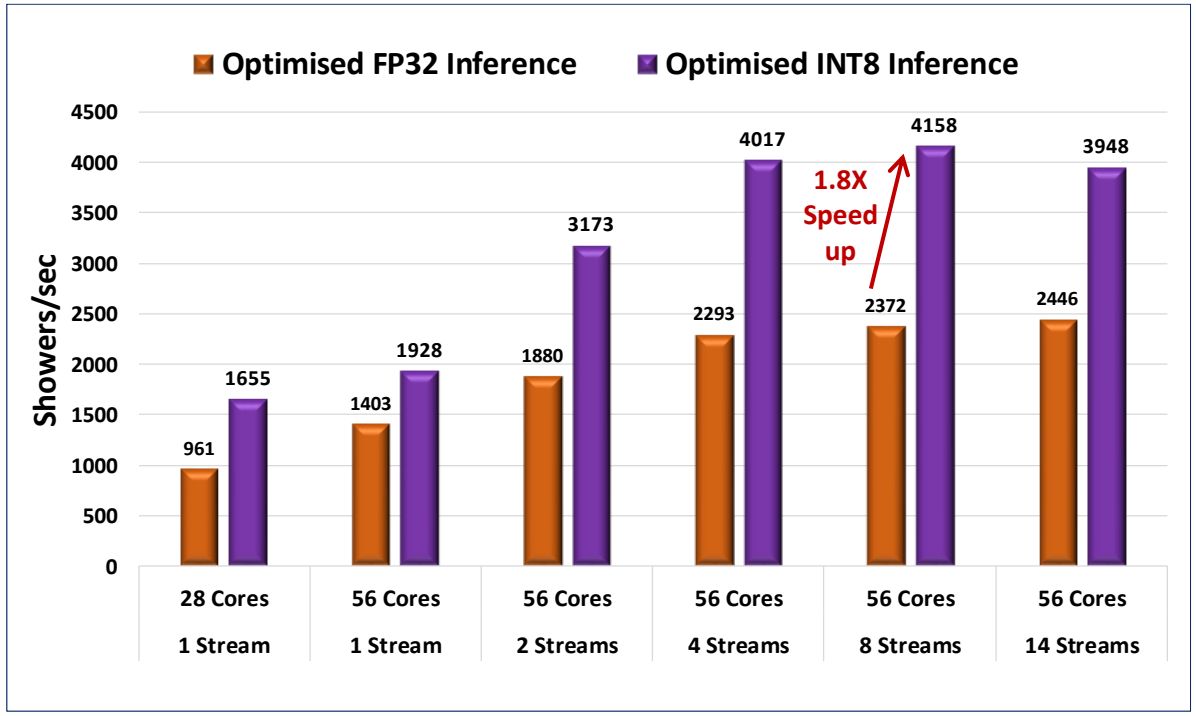

- Reference Tool: TensorFlow Lite
  - Does not support signed int8 → no LeakyReLU
  - Does not support transpose convolutional layers for up-sampling

  → Use quantized TensorFlow Lite model only for accuracy comparison

https://www.tensorflow.org/lite

# Computational Evaluation

*(of iLoT model)*




| Model | Speedup vs Monte Carlo |
|---|---|
| float32 | 38 000x |
| int8 | 68 000x |

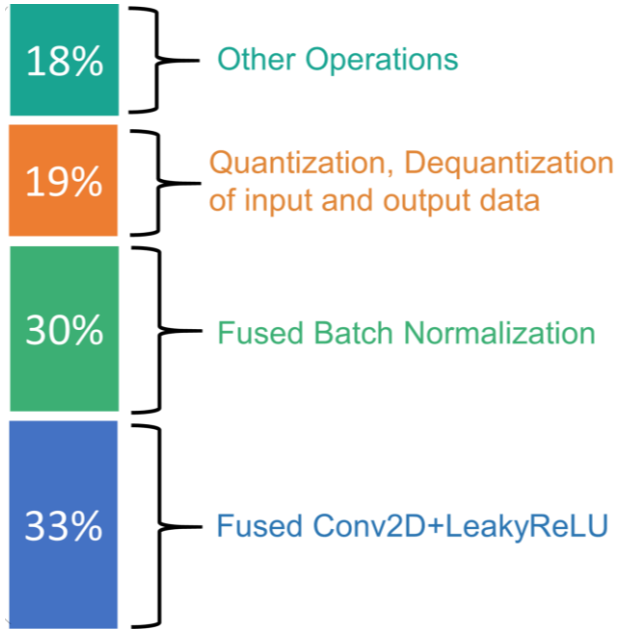
Test Setup:
TensorFlow v2.3; Platform: Intel(R) Xeon(R) Platinum 8280 CPU; Cascade Lake architecture; #Nodes: 1; #Sockets: 2; Cores/socket: 28; Threads/socket: 56;
DDR Mem Config: 12 slots / 16GB / 2933;

- Speedup of **1.8x**

→ Total speedup of 68 000x versus Monte Carlo

# Computational Evaluation

*(of iLoT model)*




- Reduction in model memory size of **2.26x**

| Model | Memory [MB] |
|---|---|
| float32 | 8.08 |
| int8 | 3.57 |

- Future: Fusion of [Conv2D + LeakyReLU + Batch Normalization]

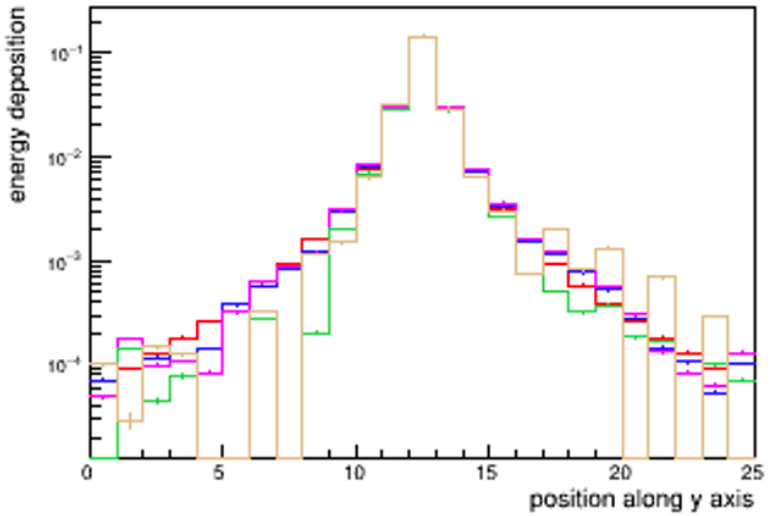
→Possible additional speedup

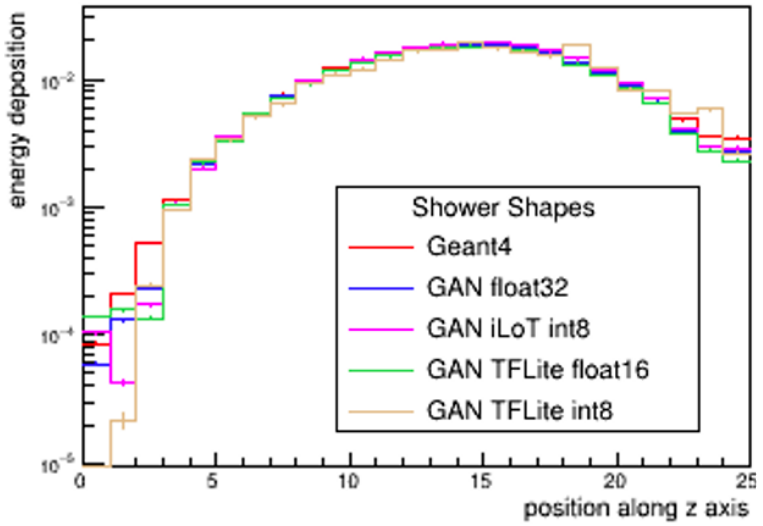
# Physics Evaluation

*Shower Shapes*

- Mean squared error (MSE) between GAN and validation data

| Model | MSE (Lower is better) |
|---|---|
| float32 | 0.061 |
| **iLoT int8** | **0.053** |
| TFLite float16 | 0.253 |
| TFLite int8 | 0.340 |




- iLoT shows a good accuracy
- TensorFlow Lite performs worse

# Physics Evaluation

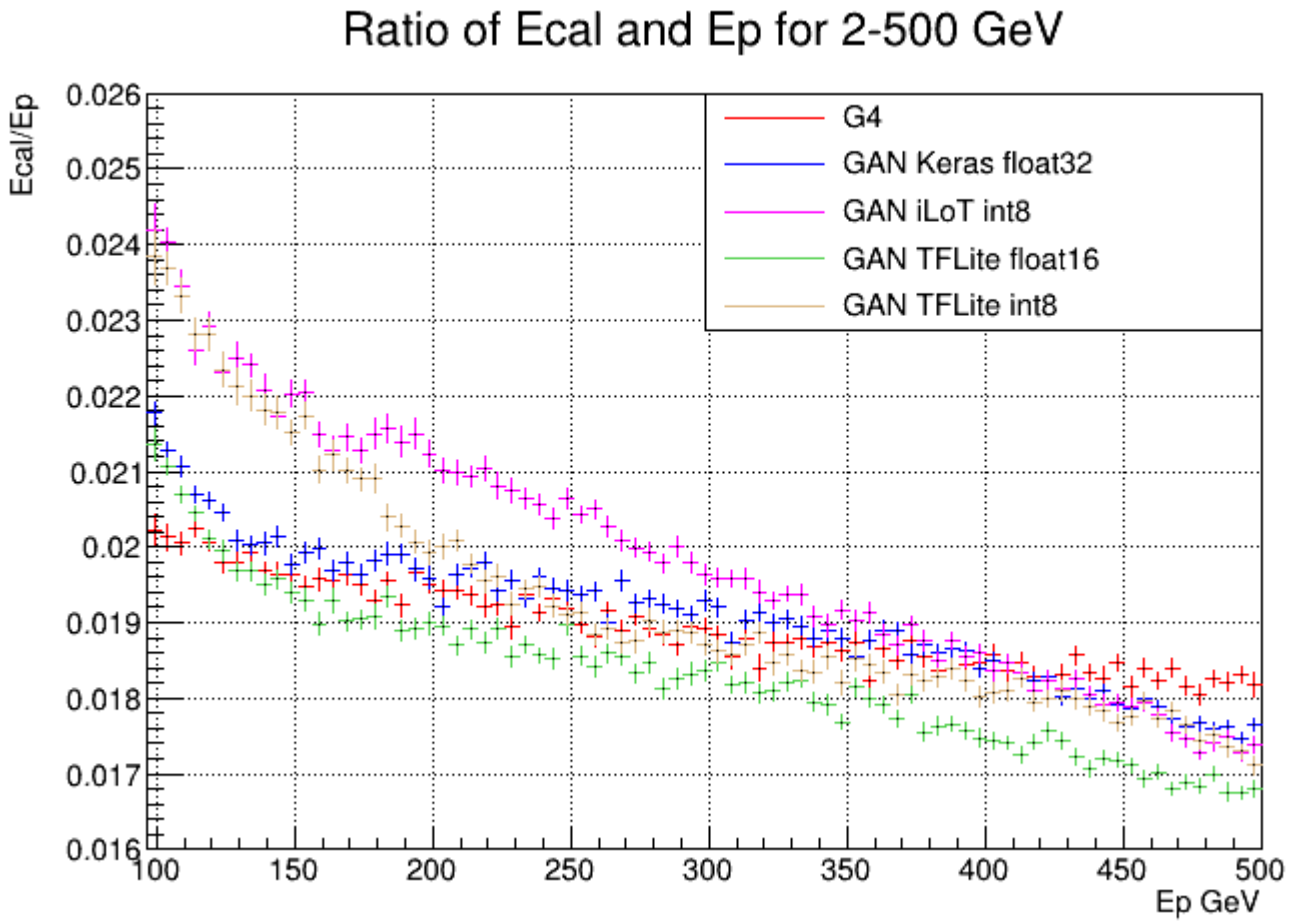
*Sampling Fraction*

- Ecal: Sum of all image cells ≙ total measured energy
- Ep: Energy of the injected particle into the calorimeter or generator network

**The quantization does not take this metrics into account**

→ More detailed studies needed

→ Define new physics metrics



# Summary

- **2.1x** speedup due to conversion from Conv3D to Conv2D
- **38000x** speedup of GAN to Geant4

- **1.8x** speedup due to quantization from float32 to int8
- **68000x** total speedup of quantized GAN versus Geant4 simulation

- Good physics accuracy for optimization metrics

# QUESTIONS?

Reduced Precision Strategies for Deep Learning: 3DGAN Use Case

Florian Rehm [CERN openlab, RWTH Aachen]

Sofia Vallecorsa [CERN openlab], Vikram Saletore [Intel], Hans Pabst [Intel], Adel Chaibi [Intel],
Kerstin Borras [DESY, RWTH Aachen], Dirk Krücker [DESY]

# Backup: Computational Evaluation

*Tests on Intel CPUs*

Test Setup:
TensorFlow v2.3; Platform: Intel(R) Xeon(R) Platinum 8280 CPU;
Cascade Lake architecture; #Nodes: 1; #Sockets: 2;
Cores/socket: 28; Threads/socket: 56;
DDR Mem Config: 12 slots / 16GB / 2933;

| Method | SW Framework Year | Precision | HW Platform | Time/Shower | Speedup |
|---|---|---|---|---|---|
| Monte-Carlo (Geant4) Sim SW | Simulation SW | FP32 | 2S Intel Xeon® Processor **8180** | 17000 | **1.0 (Baseline)** |
| 3D-GANs (3D conv) 4-Streams | TF 1.14 2018 | FP32 | 2S Intel Xeon® Processor **8160** | 0.85 | 20,000x |
| | | | | | |
| 3D-GANs (using 2D conv) 4-Streams | TF 2.4 2020 | **FP32** | **2nd Gen 2S Intel Xeon® Processor 8280** | 0.43 | 38,000x |
| **3D-GANs (using 2D conv) 4-Streams** | **TF 2.4 2020** | **INT8** | **2nd Gen 2S Intel Xeon® Processor 8280** | **0.25** | **68,000x** |

# Backup: Pixelwise Comparison

- Measures how different the output images of the models are
- Same Input vector to all models
- Sum of the absolute elementwise difference of the outputs

| Model | Mean | STD |
|---|---|---|
| iLoT int8 | 0.133 | 0.291 |
| TFLite float16 | 4.054 | 0.721 |
| TFLite int8 | 1.550 | 0.191 |

# Backup: Shower Shapes