CERN openlab

Generative Models for Simulation

Kristina Jaruskova (CERN, CTU), Kalliopi Tsolaki (CERN), Sofia Vallecorsa (CERN)
Duncan Kampert (SURF B.V.), Vikram Saletore (Intel Corp.)

CERN openlab Technical Workshop 2023

Content

In this talk – **projects in collaboration with Intel Corp. and SURF B.V.**

- Introduction and motivation
- 3DGAN quantization with Intel® Neural Compressor
- 3DGAN integration with Geant4

Following talk

- Foundation Models

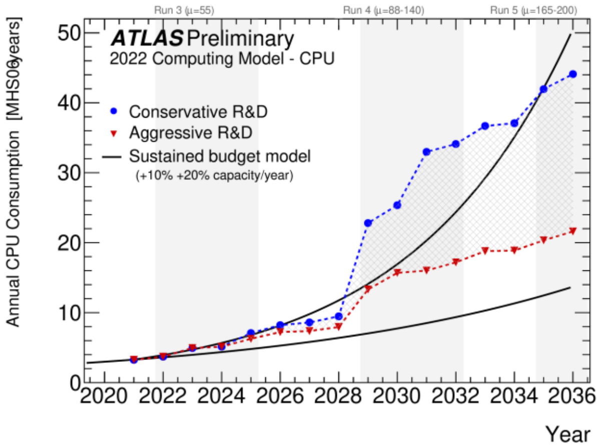
Introduction

Detector simulations

- Monte Carlo-based toolkits (Geant4) – particles interacting with matter
- HEP experiments – specific software frameworks using Geant4
- Computationally intensive
  - 50 % WLCG resources used for simulations[1]
  - E.g. ATLAS – aggressive R&D approach required for HL-LHC

Faster alternatives

- Deep learning models of different types
  - GANs, VAEs, NFs, GNNs, …
  - Development in experiment groups, Geant4, Openlab (IT)
- Focusing on electromagnetic calorimeters (ECAL)
  - High granularity -> most time demanding step in simulation (> 50 %[2])



ATLAS (2022)

[1] The HEP Software Foundation. A Roadmap for HEP Software and Computing R&D for the 2020s. Comput. Softw. Big. Sci 2019.

[2] M. Rama. Fast Calorimeter Simulation in the LHCb Gauss Framework.CHEP 2018.

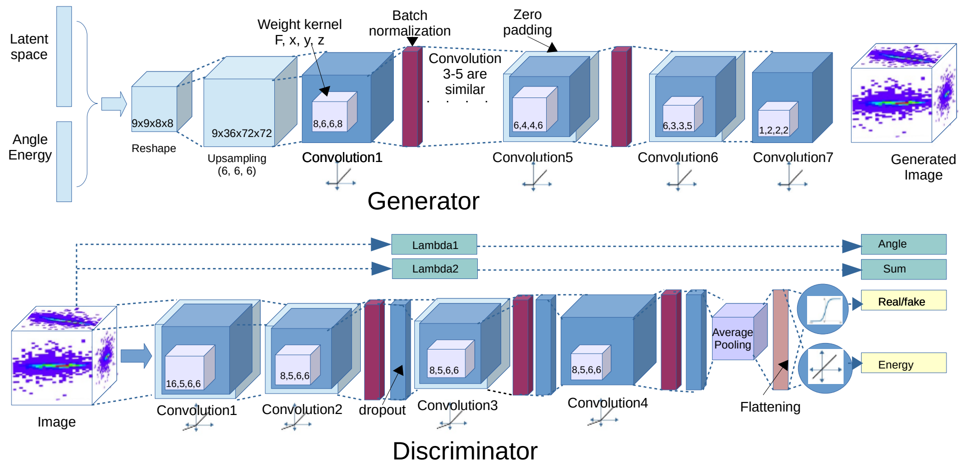
3DGAN and ECAL dataset

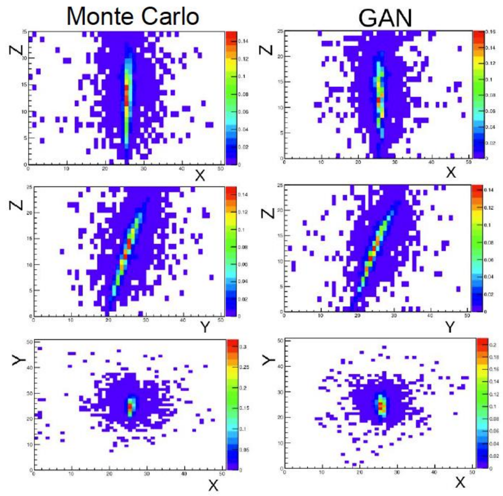
3DGAN model

- 7x 3DConv layers in generator
- Generator output conditioned by primary energy E_p and incident angle θ

Training dataset (MC samples, Zenodo)

- 3D image of a shower from single e^- entering ECAL
  - 51x51x25 cells (~ 65k)
- Primary energy E_p = 2 - 500 GeV
- Incident angle θ = 60° - 120°





G. Khattak *et al.* Fast simulations of a high granularity calorimeter by generative adversarial networks. *Eur. Phys. J. C*. 2022

Quantization with Intel® Neural Compressor

- Intel® Neural Compressor
  - Tool for model compression - quantization, pruning, knowledge distillation
  - Automated strategies for model quantization – to meet given accuracy goals
- Quantizing 3DGAN generator: FP32 → INT8
  - Post-training static quantization
  - Use generator loss as accuracy metric for quantization
    - Based on mean and variance of the shower shapes

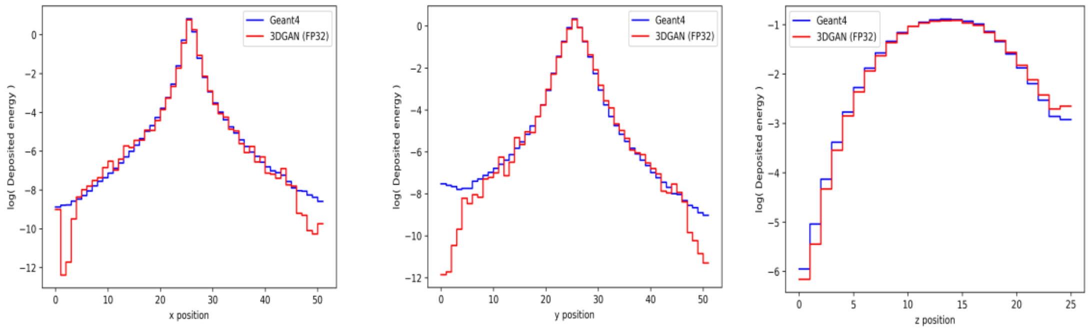


Image: Shower shapes in *x*, *y*, *z* axis (Geant4 vs. 3DGAN)

Generator loss for quantization

- Squared error on mean values and variance on the shower shapes

$$D = \text{dataset}$$
$$I = \text{network input}(D)$$
$$O = 3DGAN(I)$$

$$A_x = \text{mean}(D,\ \text{axis} = x), A_y, A_z$$
$$\widetilde{A}_x = \text{mean}(O,\ \text{axis} = x), \widetilde{A}_y, \widetilde{A}_z$$
$$V_x = \text{var}(D,\ \text{axis} = x), V_y, V_z$$
$$\widetilde{V}_x = \text{var}(O,\ \text{axis} = x), \widetilde{V}_y, \widetilde{V}_z$$

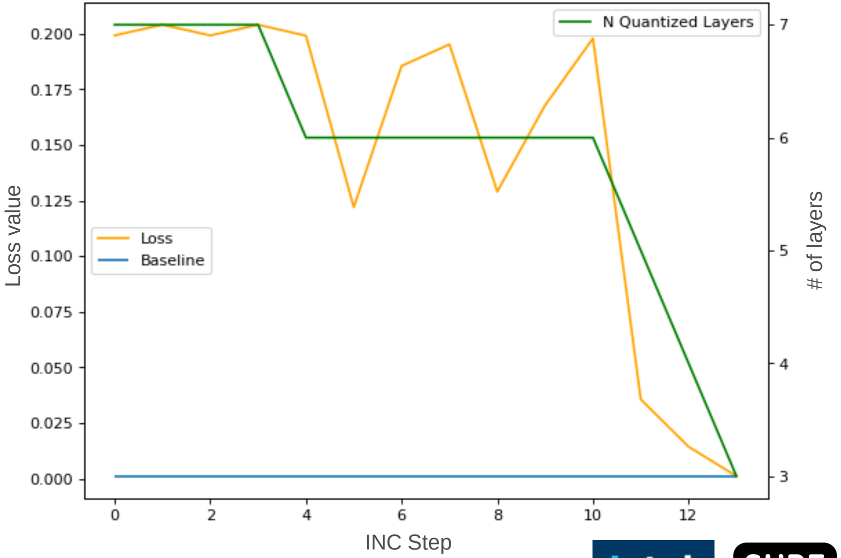
$$\text{loss}\,(D, O) = \sum_{x,y,z} \text{MSE}(A_., \widetilde{A}_.) + \sum_{x,y,z} \text{MSE}(V_., \widetilde{V}_.)$$

Quantization with Intel Neural Compressor

- Quantize all layers → step by step reverse the process until the accuracy target is met.

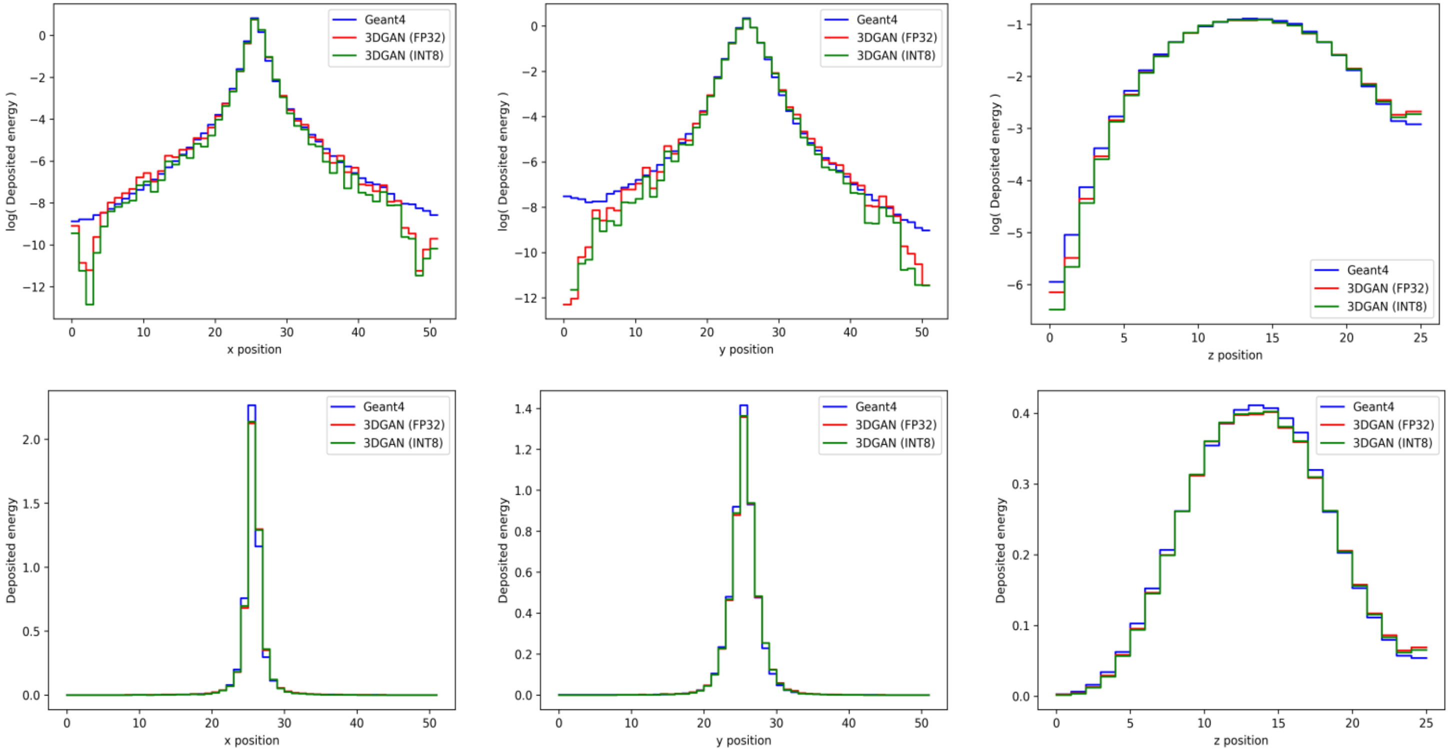
step_id	Num Quantized Layers	Loss
Baseline	**0**	**0.0011**
INC Steps		
0	7	0.1992
1	7	0.2040
2	7	0.1992
3	7	0.2040
4	6	0.1991
5	6	0.1220
6	6	0.1855
7	6	0.1952
8	6	0.1290
9	6	0.1678
10	6	0.1979
11	5	0.0356
12	4	0.0143
13	**3**	**0.0011**

- One data file - 5 000 samples
  - Training – 540k samples
- 10 minutes to quantize the model (13 iterations)



Duncan Kampert, SURF B.V. Code: https://github.com/sara-nl/QuantizedGAN

Quantization: Shower shapes



Quantization: Inference Throughput Performance

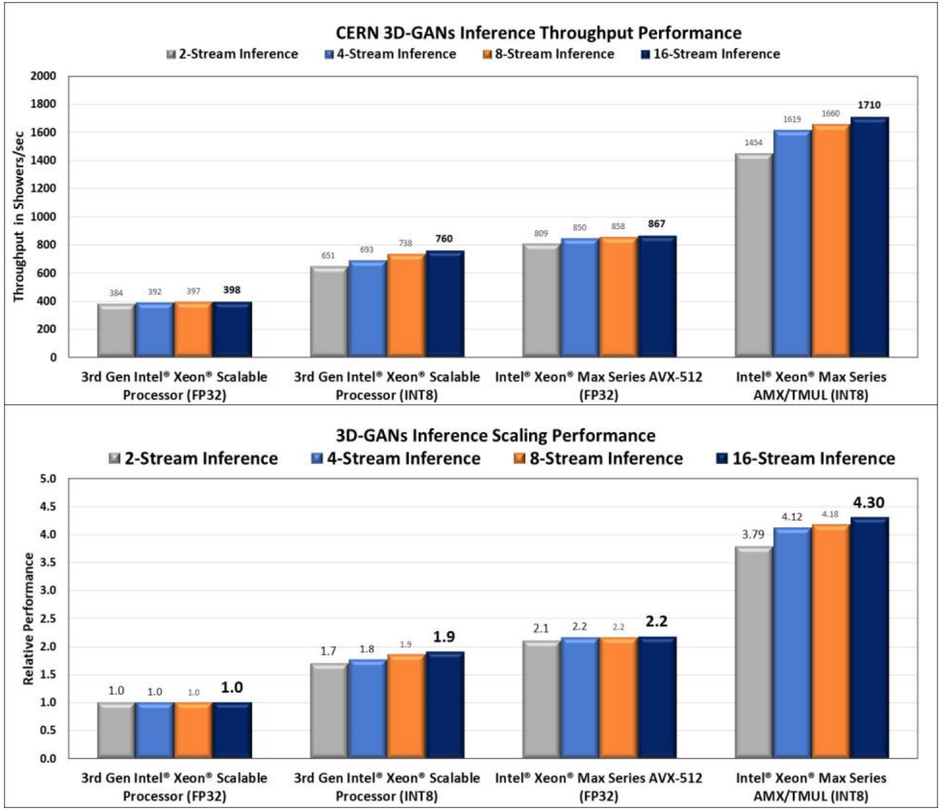
- On all platforms:
  - Applier Intel® Neural Compressor Graph Optimizations
  - Intel® optimized Tensorflow 2.10

- **Baseline:** 3rd Gen Intel® Xeon® Scalable Processor 8360 – FP32
- 3rd Gen Intel® Xeon® Scalable Processor 8360 – INT8
- Intel® Xeon® Max Series – FP32
- Intel® Xeon® Max Series – AMX/TMUL INT8

Up to **4.3x speed up** compared to baseline.

Credits:

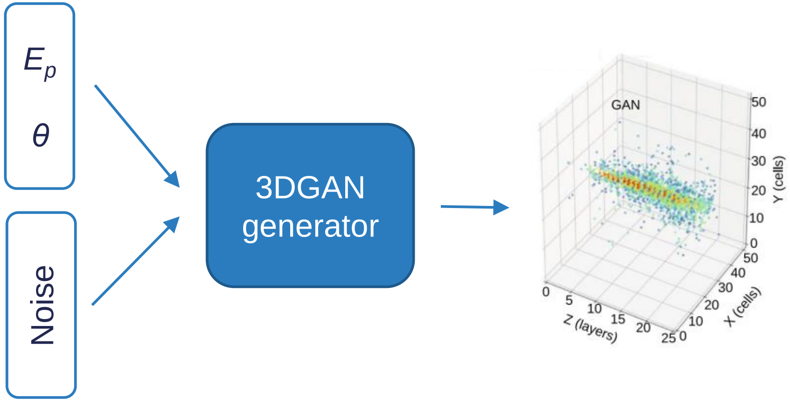
Duncan Kampert, SURF B.V
duncan.kamper@surf.nl



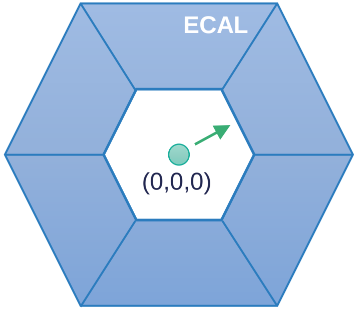
*AMX = advanced matrix extensions,TMUL = tiled matrix multiplication

3DGAN and Geant4 integration

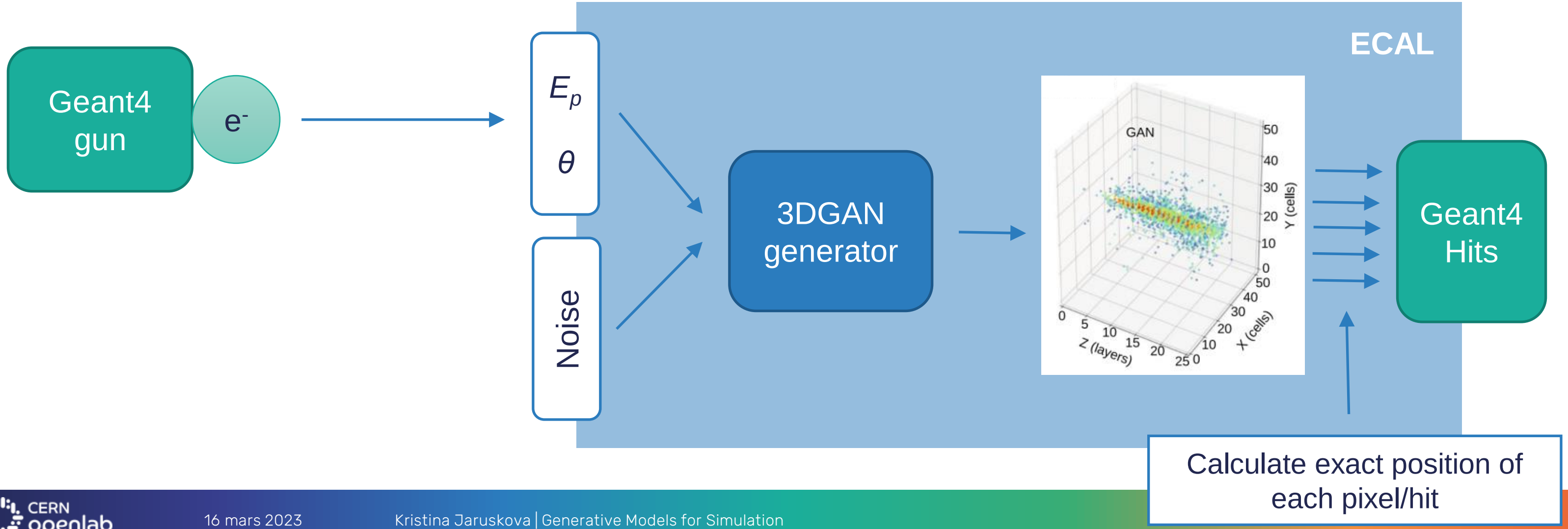
- 3DGAN model
  - Simulating ECAL region of a detector that was designed for CLIC – single electron
  - Input: initial particle energy E_p and incident angle θ + random noise.



3DGAN and Geant4 integration



- 3DGAN model
  - Simulating ECAL region of a detector that was designed for CLIC – single electron
  - Input: initial particle energy E_p and incident angle θ + random noise.
- Starting from Par04 example (Geant4) – Geant4-based framework with VAE for inference (only ECAL)
  - Manual detector definition → using GDML file with LCD ECAL description
  - Replace VAE with 3DGAN
  - Prepare input to 3DGAN and adjust post-processing of the model output

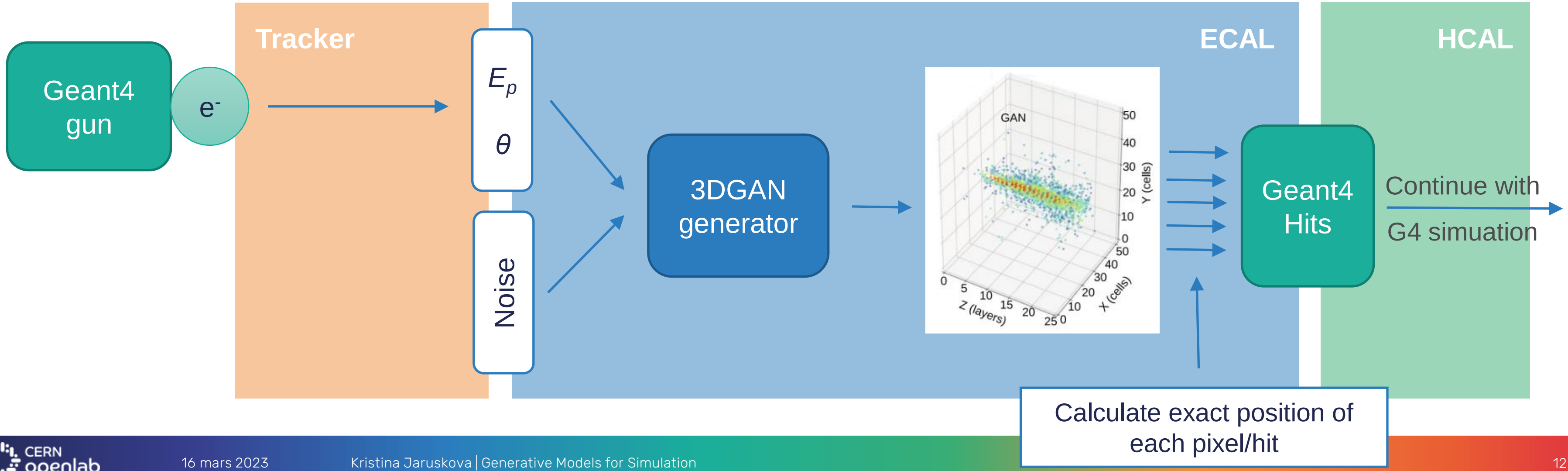


3DGAN and Geant4 integration

- 3DGAN model
  - Simulating ECAL region of a detector that was designed for CLIC – single electron
  - Input: initial particle energy E_p and incident angle θ + random noise.
- Starting from Par04 example (Geant4) – Geant4-based framework with VAE for inference (only ECAL)
  - Manual detector definition → using GDML file with full LCD detector description
  - Replace VAE with 3DGAN
  - Prepare input to 3DGAN and adjust post-processing of the model output

Work in Progress

- Include the full detector description and simulate the whole passage.
- Compare full and fast simulation performance (speed up).



Next talk…

- Drawback of 3DGAN and other fastsim models - one model per specific MC dataset.
- Change in detector material, size, thickness of layers → generate new training data and train a new DL model

- Idea – one large model that can adapt to some changes in the detector design

Next talk…

- Drawback of 3DGAN and other fastsim models - one model per specific MC dataset.
- Change in detector material, size, thickness of layers → generate new training data and train a new DL model

- Idea – one large model that can adapt to some changes in the detector design

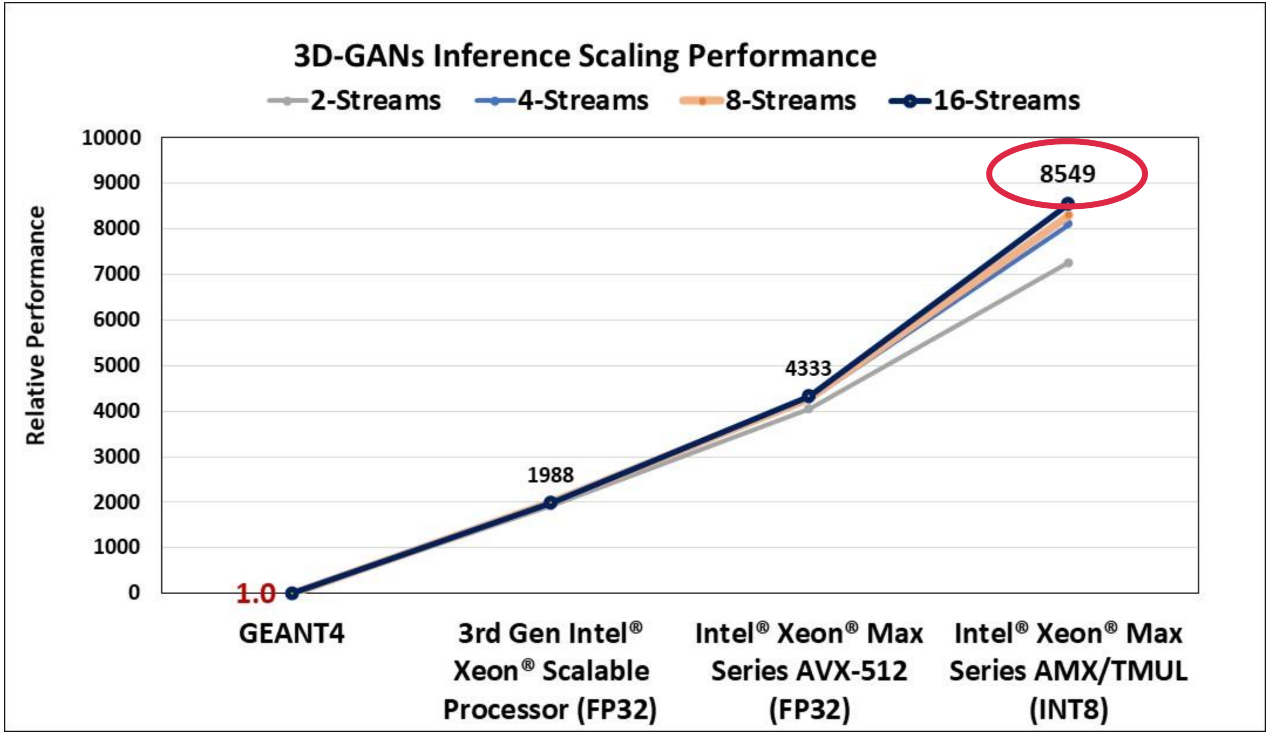
→ Foundation Models

CERN openlab

Configuration Details

- BASELINE: 3rd Gen Intel® Xeon® Scalable Processor 8360Y (AVX512, FP32 and INT8)**:** Test by Intel as of 08/21/2022. 2x Intel® Xeon® Scalable Processor 8360Y, HT On, Turbo On, Total Memory 512 GB (16 slots/ 32 GB/ 3200 MHz, DDR4), BIOS WLYDCRB1.SYS.0021.P21.2106280839, ucode 0x8d9522d4, CentOS Linux 8, kernel 4.18.0-240.22.1.el8_3.x86_64, https://github.com/svalleco/3Dgan/, Intel TensorFlow 2.10.0, Keras 2.10.0, GCC 11.20, Intel Neural Compressor: https://www.intel.com/content/www/us/en/developer/tools/oneapi/neural-compressor.html, TF Wheel intel_tensorflow_avx512-2.10.0.202230-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl, Python 3.8.13
- Intel® Xeon® Max Series (AVX-512 FP32): Test by Intel as of 10/18/2022. 1 node, 2x Intel® Xeon® Processors codenamed Sapphire Rapids with HBM, HT On, Turbo On, Total Memory 128 HBM, BIOS SE5C7411.86B.8424.D03.2208100444, ucode 0x2c000020, CentOS Stream 8, kernel 5.19.0-rc6.0712.intel_next.1.x86_64+server https://github.com/svalleco/3Dgan/, Intel TensorFlow 2.10.0, Keras 2.10.0, GCC 11.20, Intel Neural Compressor: https://www.intel.com/content/www/us/en/developer/tools/oneapi/neural-compressor.html, TF Wheel intel_tensorflow_avx512-2.10.0.202230-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl, Python 3.8.13
- Intel® Xeon® Max Series (AMX/INT8): Test by Intel as of 10/18/2022. 1 node, 2x Intel® Xeon® Processors codenamed Sapphire Rapids with HBM, HT On, Turbo On, Total Memory 128 HBM, BIOS SE5C7411.86B.8424.D03.2208100444, ucode 0x2c000020, CentOS Stream 8, kernel 5.19.0-rc6.0712.intel_next.1.x86_64+server, https://github.com/svalleco/3Dgan/, Intel TensorFlow 2.10.0, Keras 2.10.0, GCC 11.20, Intel Neural Compressor: https://www.intel.com/content/www/us/en/developer/tools/oneapi/neural-compressor.html, TF Wheel intel_tensorflow_avx512-2.10.0.202230-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl, Python 3.8.13

Quantization: Inference Throughput Performance

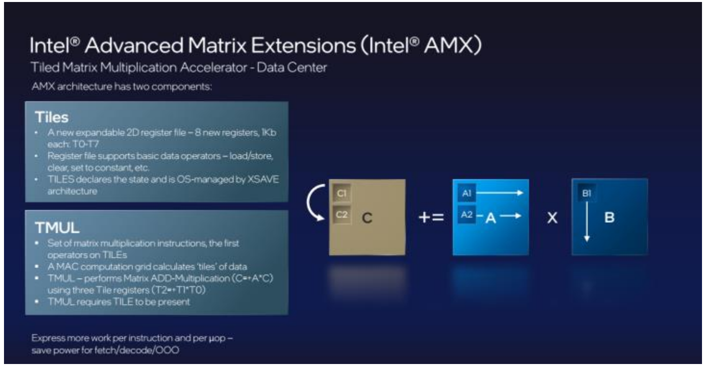


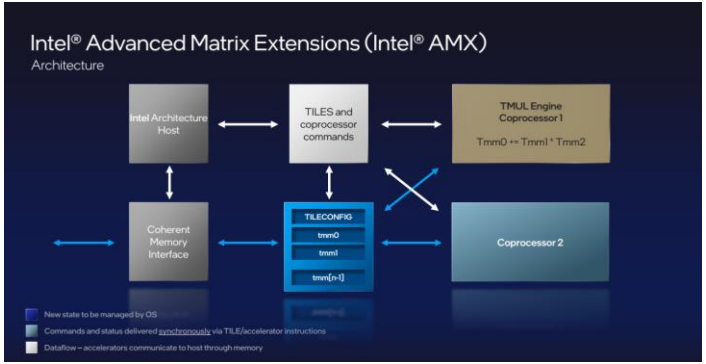
Credits:

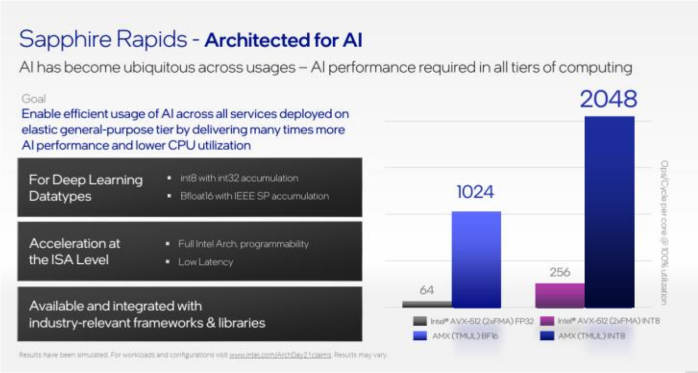
Duncan Kampert, SURF B.V

duncan.kamper@surf.nl

Intel® Xeon® Max Series: AMX and TMUL



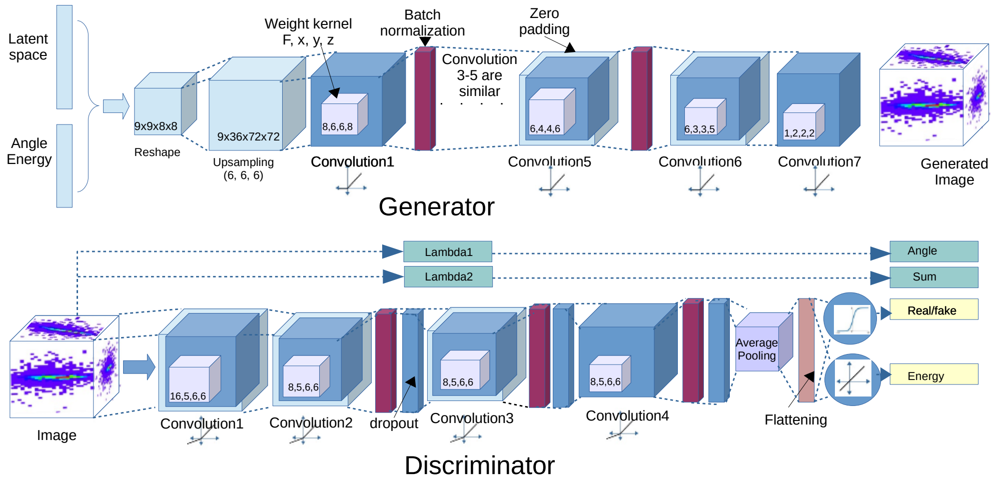




https://download.intel.com/newsroom/2021/client-computing/intel-architecture-day-2021-presentation.pdf

More about 3DGAN

- Training in 2 steps (use of transfer learning)
  1) Dataset with E_p = 100 - 200 GeV – 137k samples, 130 epochs
  2) Dataset with E_p = 2 - 500 GeV – 400k samples, 30 epochs



- Discriminator
  - Real/fake probability
  - E_p estimation
  - Angle calculation
  - Total energy calculation

  Auxiliary tasks

- Loss:

$$L_{3DGAN} = W_G L_G + W_P L_P + W_\theta L_\theta + W_E L_E$$