

# **Distributed Training of Generative Adversarial Networks for Fast Detector Simulation on Intel® Xeon® HPC Cluster**

*Big Data Summit, 18 July 2018, NERSC, LBNL, Berkeley, CA*

**Sofia Vallecorsa<sup>β</sup>, Vikram Saletore<sup>α</sup>, Damian Podareanu<sup>η</sup>,**  
**Federico Carminati<sup>β</sup>, Valeriu Codreanu<sup>η</sup>, G. Khattak<sup>β</sup>, Hans Pabst<sup>α</sup>**

<sup>α</sup>Intel Corp., <sup>β</sup>CERN, <sup>η</sup>SURFsara

# Outline

Deep Learning for fast simulation

Generative Adversarial Networks

- Model architecture

- The training sample

- Physics Performance

Computing Performance

Outlook and plans

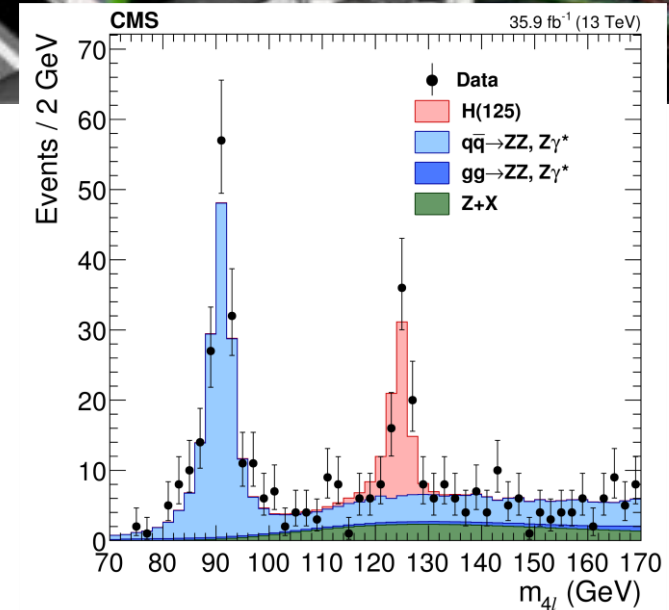
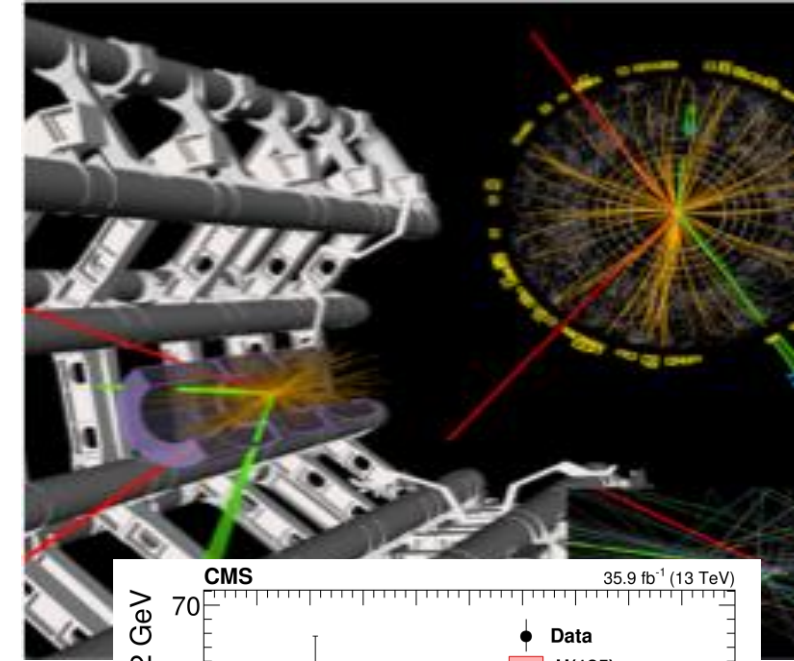
# Monte Carlo Simulation: Why

*Detailed simulation of subatomic particles is essential for data analysis, detector design*

Understand how detector design affect measurements and physics

Correct for inefficiencies, inaccuracies, unknowns.

Theory models to compare data against.



A good simulation demonstrates that we understand the detectors and the physics we are studying

# The problem

Complex physics and geometry modeling

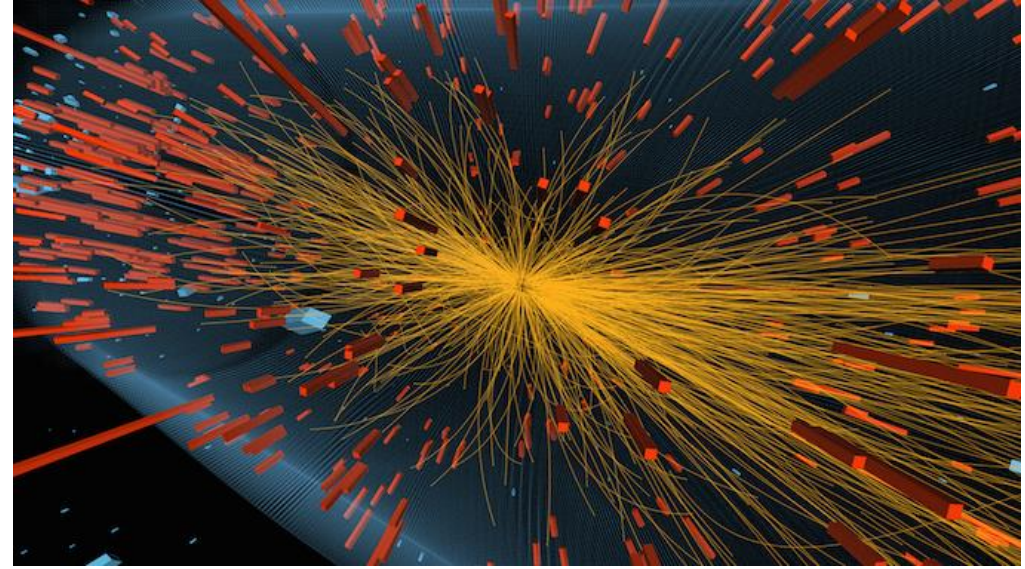
Heavy computation requirements


**>50% of WLCG power for simulations**

**Current code cannot cope (HL-LHC in 2025)**

Currently available solutions **detector dependent**

**Focus on EM Calorimeter**

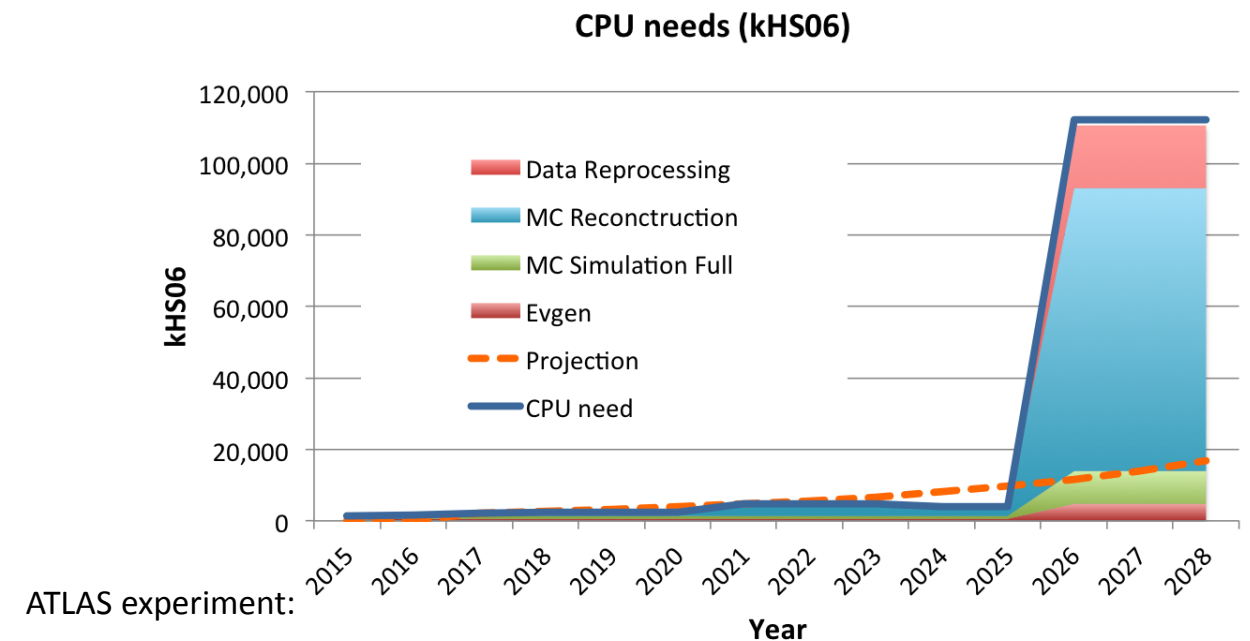




Worldwide LHC Computing Grid

200 Computing centers in 20 countries:  
> 600k cores

@CERN (20% WLCG): 65k cores;  
30PB disk + >35PB tape storage

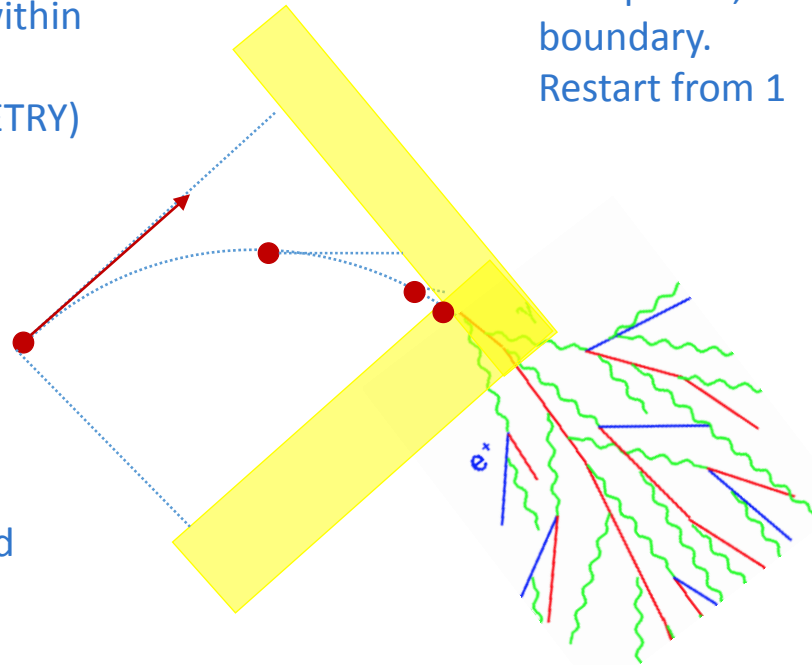


# Classical Monte Carlo simulation

2a. Check if step is within  
volume  
boundaries(GEOMETRY)

4. Repeat 2,3 until reaching volume  
boundary.  
Restart from 1

5. PHYSICS  
process

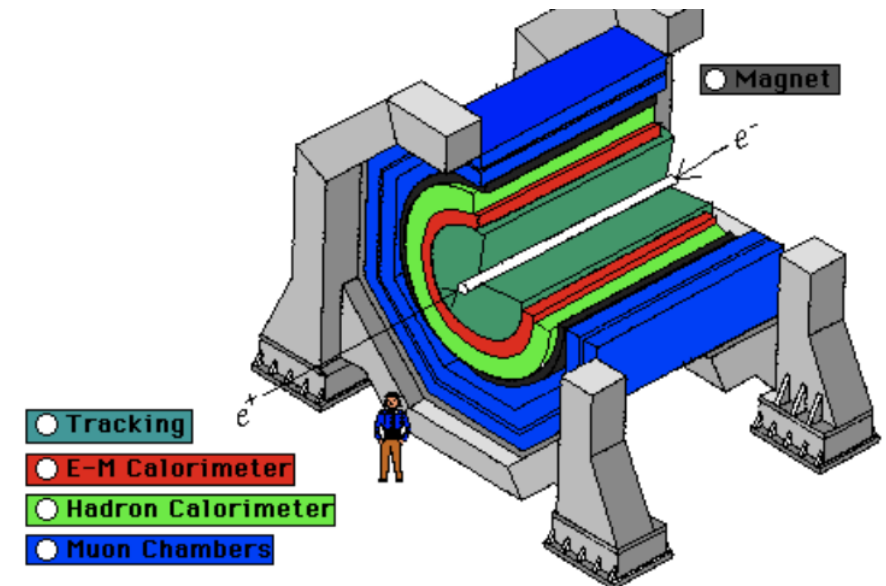


1. Calculate step particle  
could travel before doing a  
PHYSICS interaction

3. Propagate with selected  
step

Repeat **stages** 1 to 5 :

- For every particle trajectory step
- For every primary particle
- For every secondary particle



# Deep Learning for fast simulation

*Improved, efficient and accurate fast simulation*



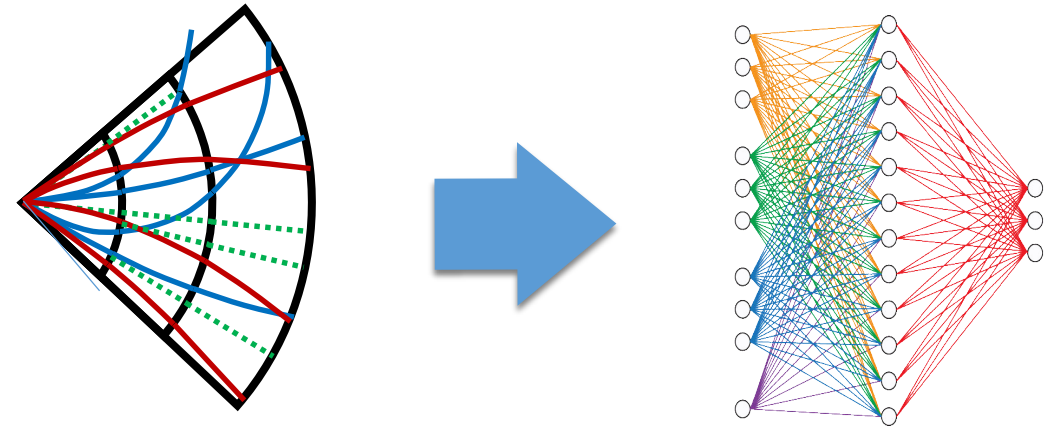
Generic approach

Can encapsulate expensive computations

Inference step is faster than algorithmic approach

Already parallelized and optimized for CPUs/HPCs.

Industry building highly optimized software, hardware, and cloud services.



Can we keep accuracy while doing things faster?



# Requirements

Precise simulation results:

- Detailed validation process

A fast inference step

Generic customizable tool

- Easy-to-use and easily extensible framework

Large hyper parameters scans and meta-optimisation:

- Training time under control

- Scalability

- Possibility to work across platforms

# A DL engine for fast simulation

Start with time consuming detectors

Reproduce particle showers in calorimeters

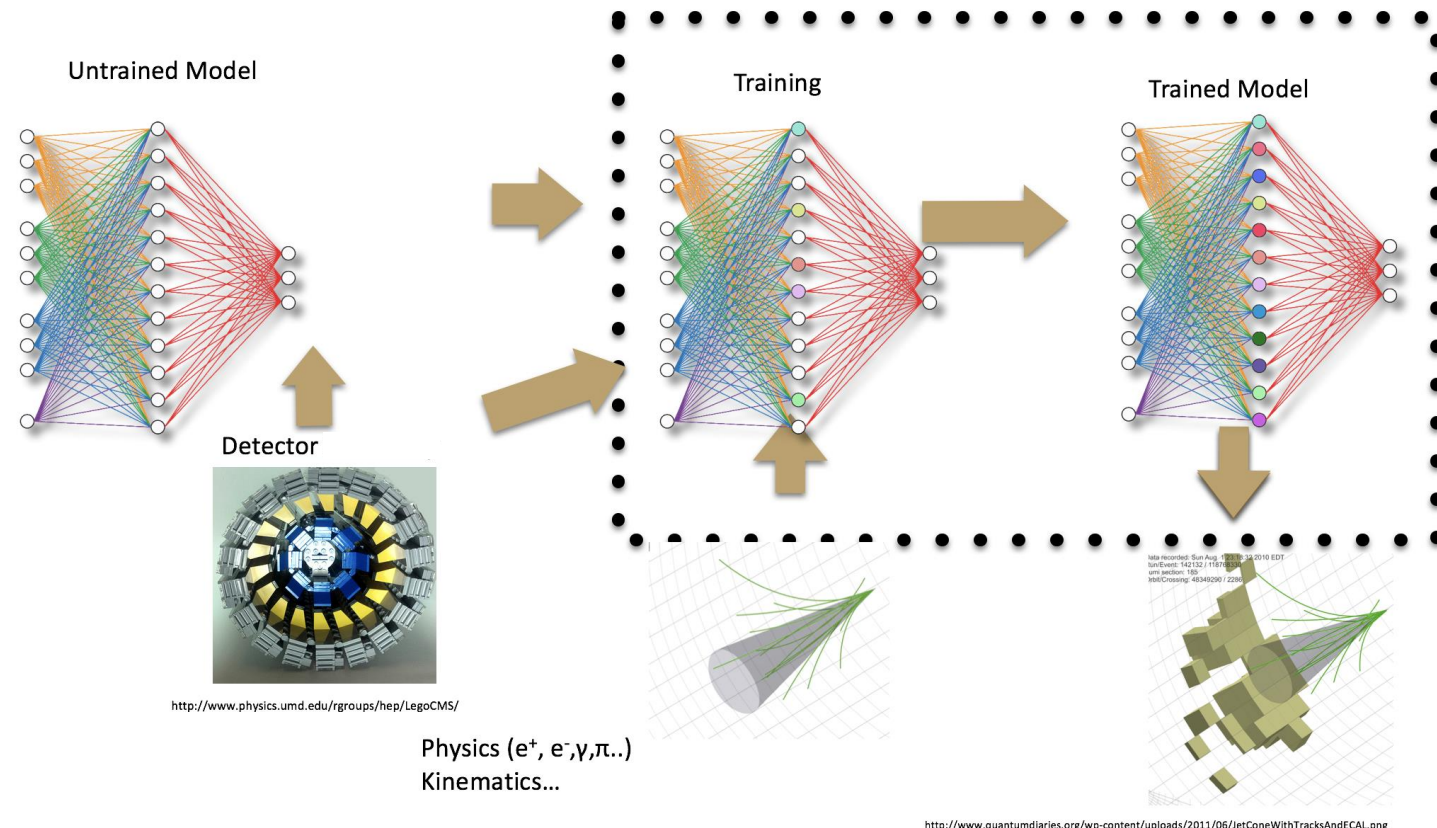
Train on detailed simulation

Test training on real data

Test different models

Generative Adversarial Networks

Embed training-inference cycle in simulation



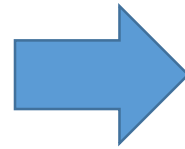


# A plan in two steps

Can image-processing approaches be useful?

Can we preserve accuracy while increasing speed?

Can we sustain the increase in detector complexity (future highly-granular calorimeters)?

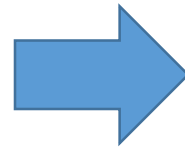


- A first proof of concept
- Understand performance and validate accuracy

How generic is this approach?

Can we “adjust” architecture to fit a large class of detectors?

What resources are needed?



- Prove generalisation is possible
- Understand and optimise computing resources
- Reduce training time
- HPC friendly

# CLIC Calorimeter

*Array of absorber material and silicon sensors*

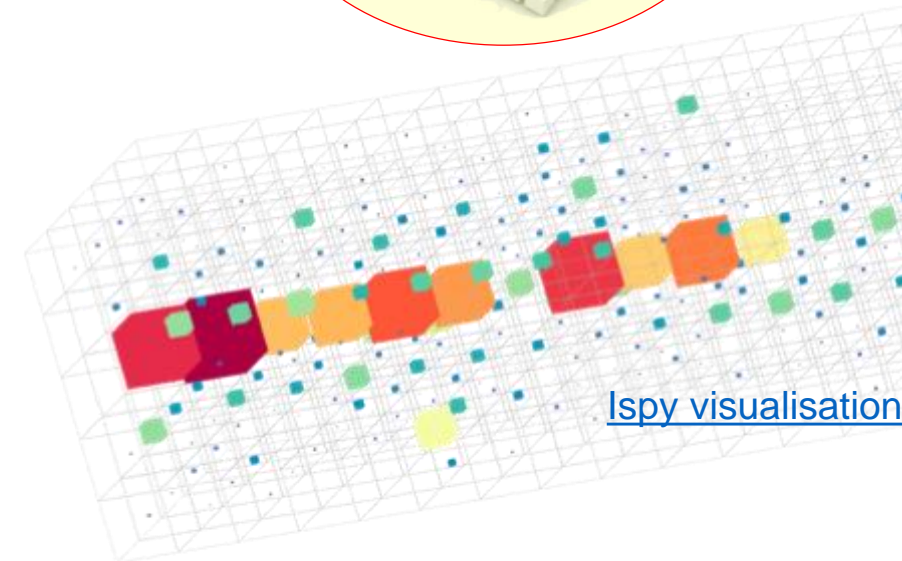
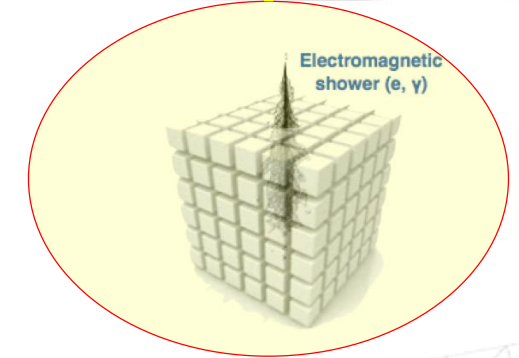
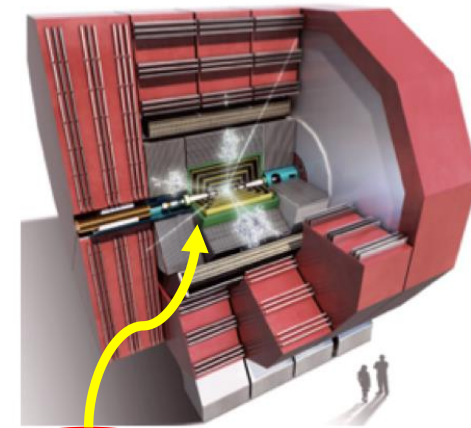
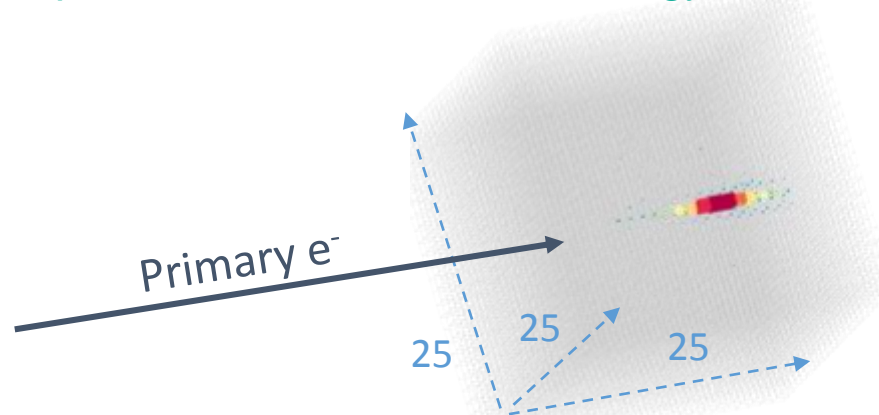
CLIC (Compact Linear Collider) is a CERN project for a linear accelerator of electrons and positrons to TeV energies

Associated electromagnetic calorimeter detector design(\*)

Highly segmented (pixelized)

Segmentation is critical for particle identification and energy calibration.

Detector output is essentially a 3D image

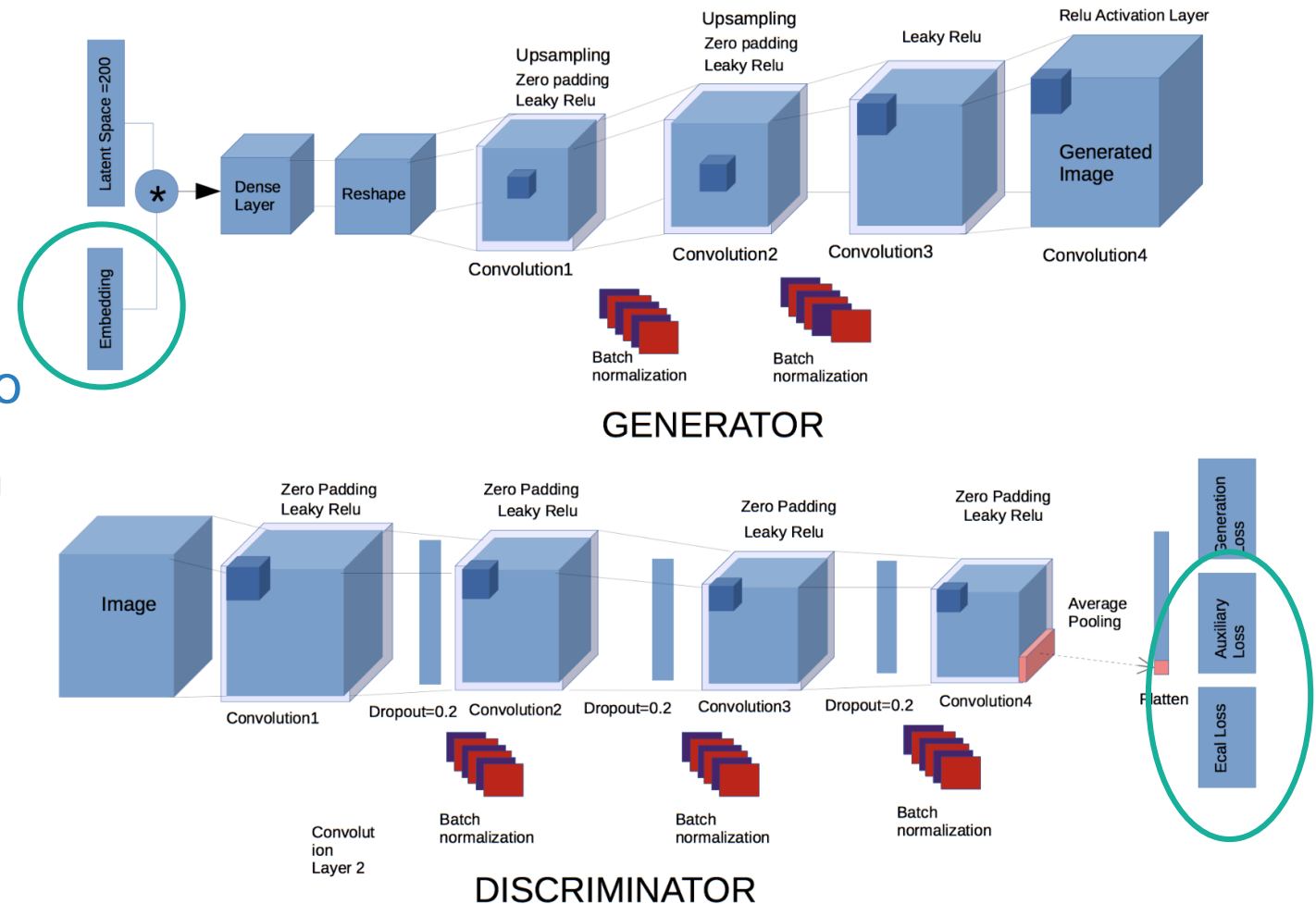


# Network architecture

## 3D conditional GAN

- reproduce full volumes of shower reconstruct in one go
- with two auxiliary regression tasks

Based on 3D convolution/deconvolutions to describe whole volume



# Conditioning and auxiliary tasks

Condition training on several input variables (particle type, energy, incidence angle)

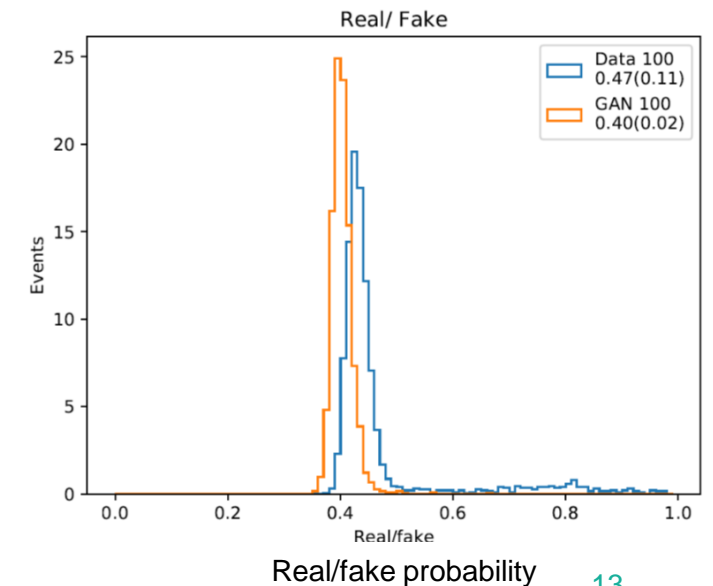
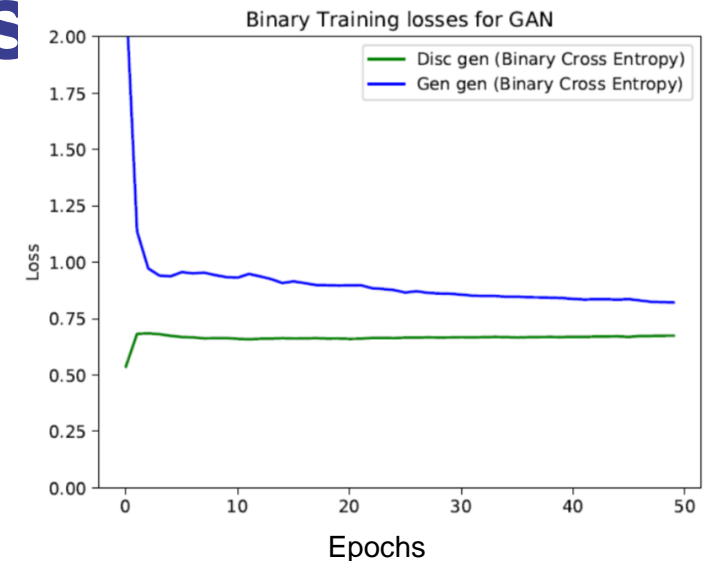
Auxiliary regression tasks assigned to the discriminator: primary particle energy, deposited energy, incidence angle

Loss is linear combination of 3 terms:

- Combined cross entropy (real/fake)

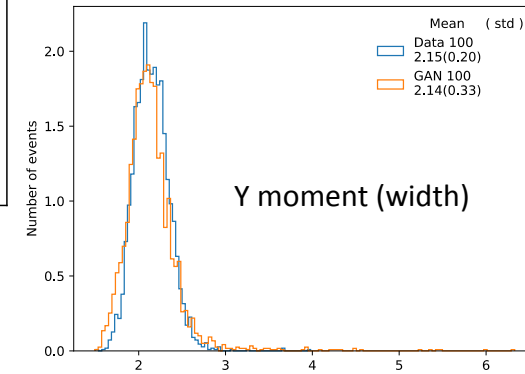
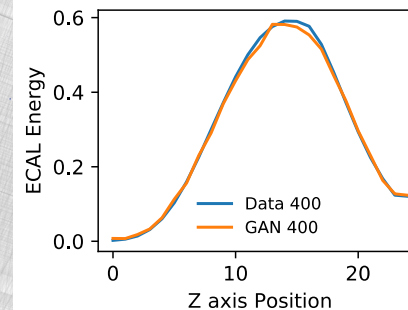
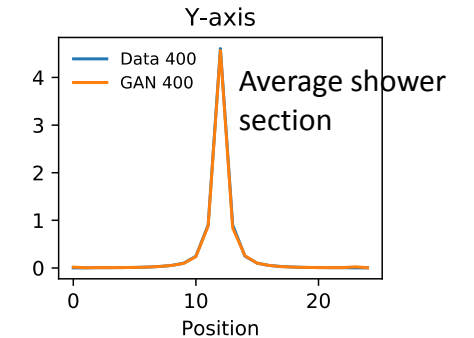
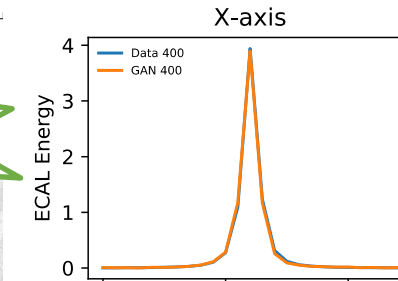
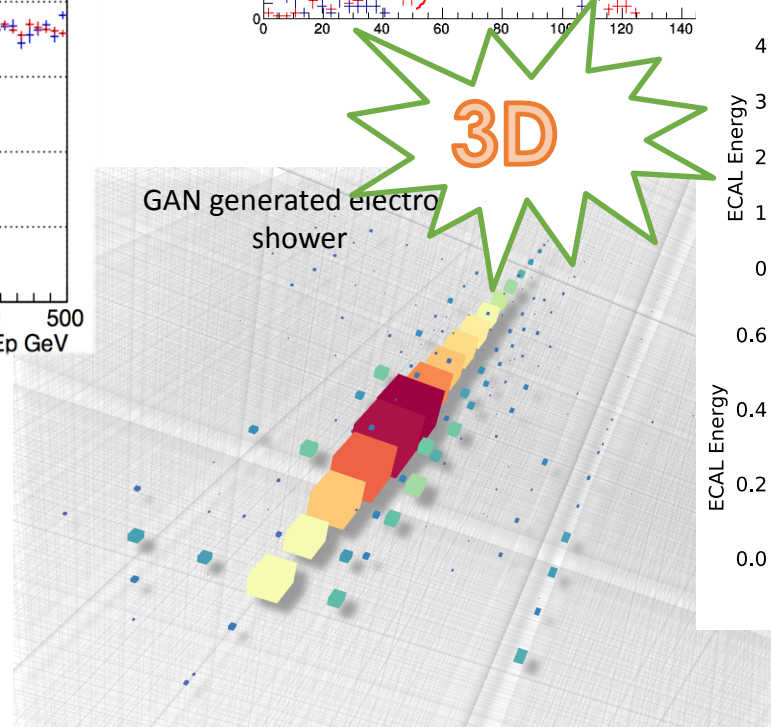
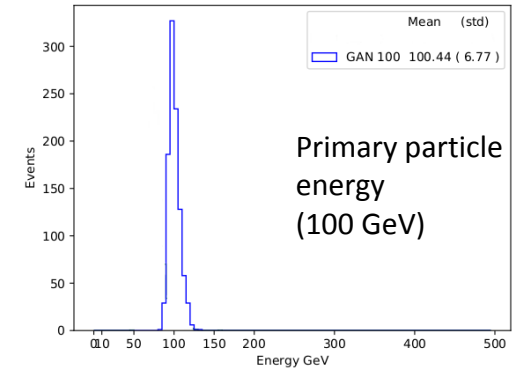
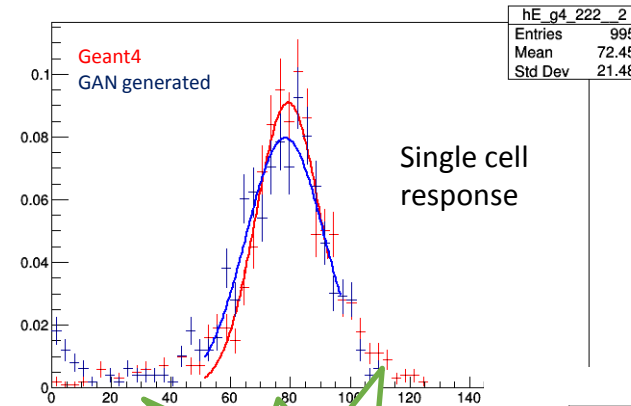
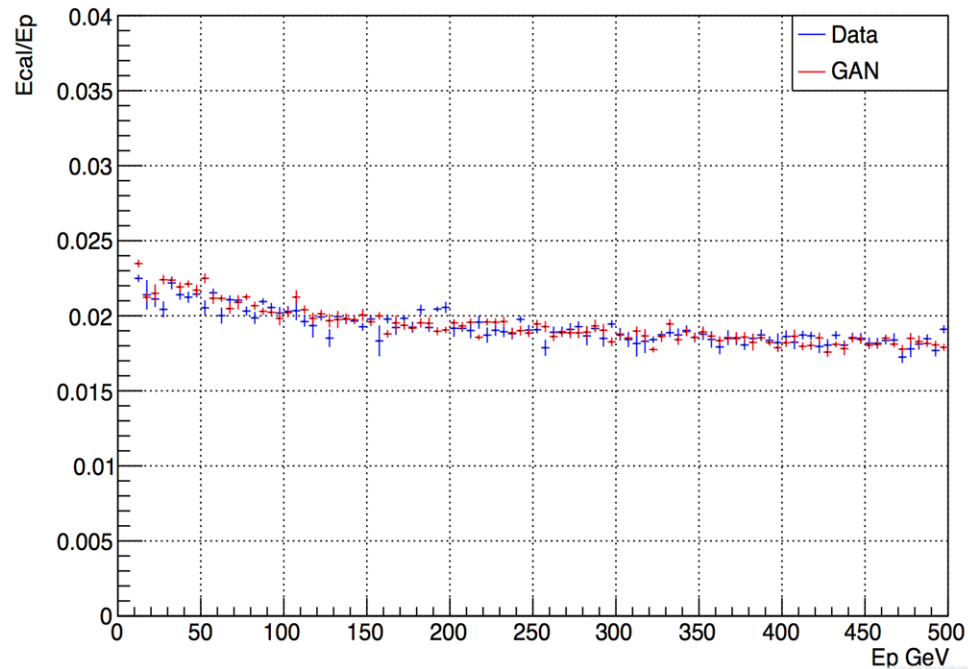
- Mean absolute percentage error for regression tasks

Easily generalisable to multi-class approach (or multi-discriminator approach): angle..



# RESULTS validation

*Comparison to Monte Carlo data*



# Outline

Deep Learning for fast simulation

Generative Adversarial Networks

- Model architecture

- The training sample

- Physics Performance

**Computing Performance**

**Outlook and plans**



# Generation speedup

*Using a trained model is very fast*

## Inference:

Classical Monte Carlo requires 17 secs/shower using Geant4

3DGAN takes **7 msec/shower**

→ speedup factor > 2500!!

| Time to create an electron shower |                          |                    |
|-----------------------------------|--------------------------|--------------------|
| Method                            | Machine                  | Time/Shower (msec) |
| Full Simulation (geant4)          | Intel Xeon Platinum 8180 | 17000              |
| 3D GAN (batch size 128)           | Intel Xeon Platinum 8180 | 7                  |

# Distributed training

Use keras 2.13 /Tensorflow 1.9  
(Intel optimised)

- AVX512 –FMA support
- Intel® MKL-DNN (with 3D convolution support)

Optimised multicore utilisation

- inter\_op\_parallelism\_threads
- intra\_op\_parallelism\_threads
- OMP\_NUM\_THREADS

Horovod 0.13.4

- MPI\_AllReduce



Run on TACC Stampede2 cluster:

- Dual socket Intel Xeon 8160
- 2x 24 cores per node, 192 GB RAM
- Intel® Omni-Path Architecture

Test several MPI scheduling configurations

- 2,4, 8 processes per nodes.
- Best machine efficiency with 4 processes/node

# Optimizing & Parallelizing Training



Karas:

**Simplicity and high productivity**

TensorFlow + MKL-DNN

**Special MKL-DNN build with 3D Conv Support**

Horovod

**Init & Wrapping TensorFlow/Karas optimizer inside HorovodDistributedOptimizer class**

**Broadcast shared variables to the Horovod World**

**Data Loading: Ensure data is loaded in parallel:** Adapting existing code to take into account of Horovod World Size

Optimizer: RMSprop

# GANs, Dataset, TF, & Runtime Options

## Original GANs

**Conv Filters:** non-multiple of 16

**Parameters:** Generator: 872736 & Discriminator: 73646; Model Size: 3.8MB

## Modified Filter for Optimized Performance

**Modified Conv Filters:** Multiple of 16 for MKL-DNN optimizations

**Parameters:** Generator: 1042276 & Discriminator: 174966; Model-Size: ~5MB

**Dataset:** 200000

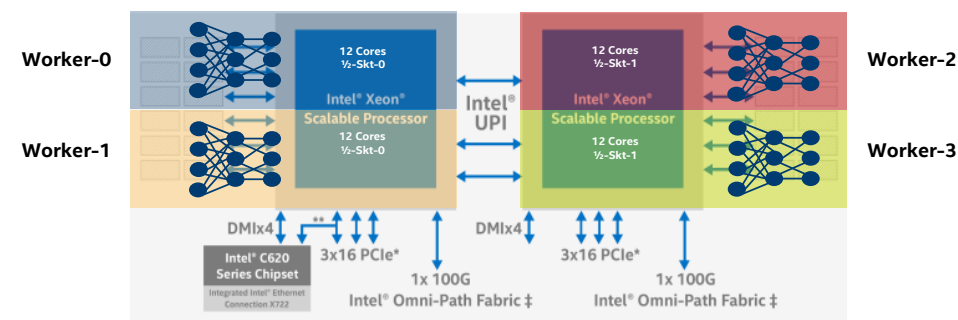
**Training Samples:** 180000 & **Validation:** 20000

TensorFlow 1.9 (private branch) + MKL-DNN (w/ 3D Conv Support)

Batch Size: 8/Worker, # Workers/Node=4/Node; 2P Xeon Nodes: 1 to 128

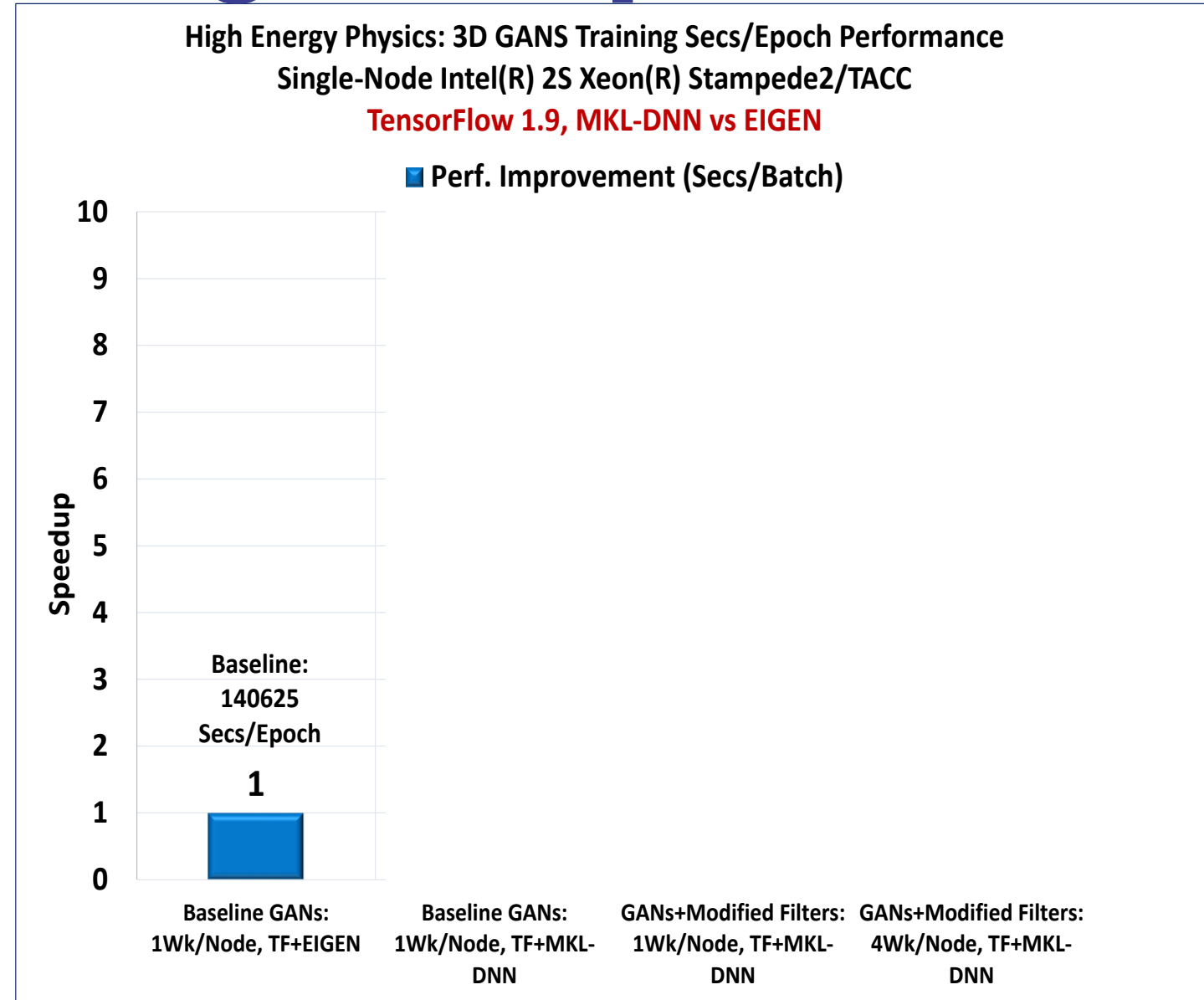
Tuning: inter\_op: 2 & Intra\_op: 11 (Xeon® 8160 is 24C/CPU); LR: 0.001, Optimizer: RMSprop

Warmup Epochs: 5 (Facebook Methodology), Training Epochs: 20, horovod fusion buf: 64MB



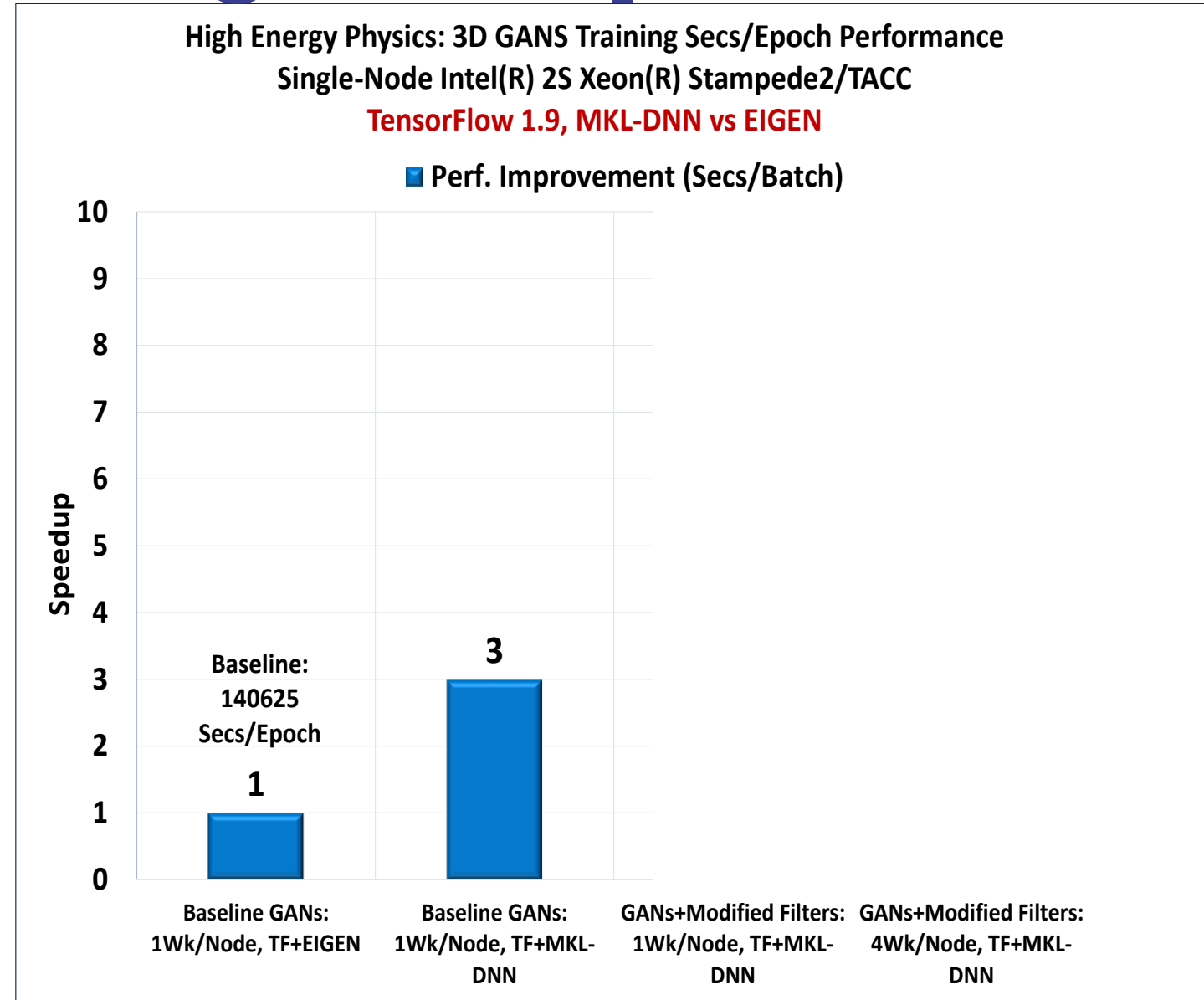
# Single-Node Training Perf. optimisation

- 1 worker/node TF + Eigen (baseline)
- 1 worker/node TF + MKL-DNN
- 1 worker/node, TF+ MKL-DNN, optimised number of convolution filters
- 4 workers/node, TF+ MKL-DNN, optimised number of convolution filters



# Single-Node Training Perf. optimisation

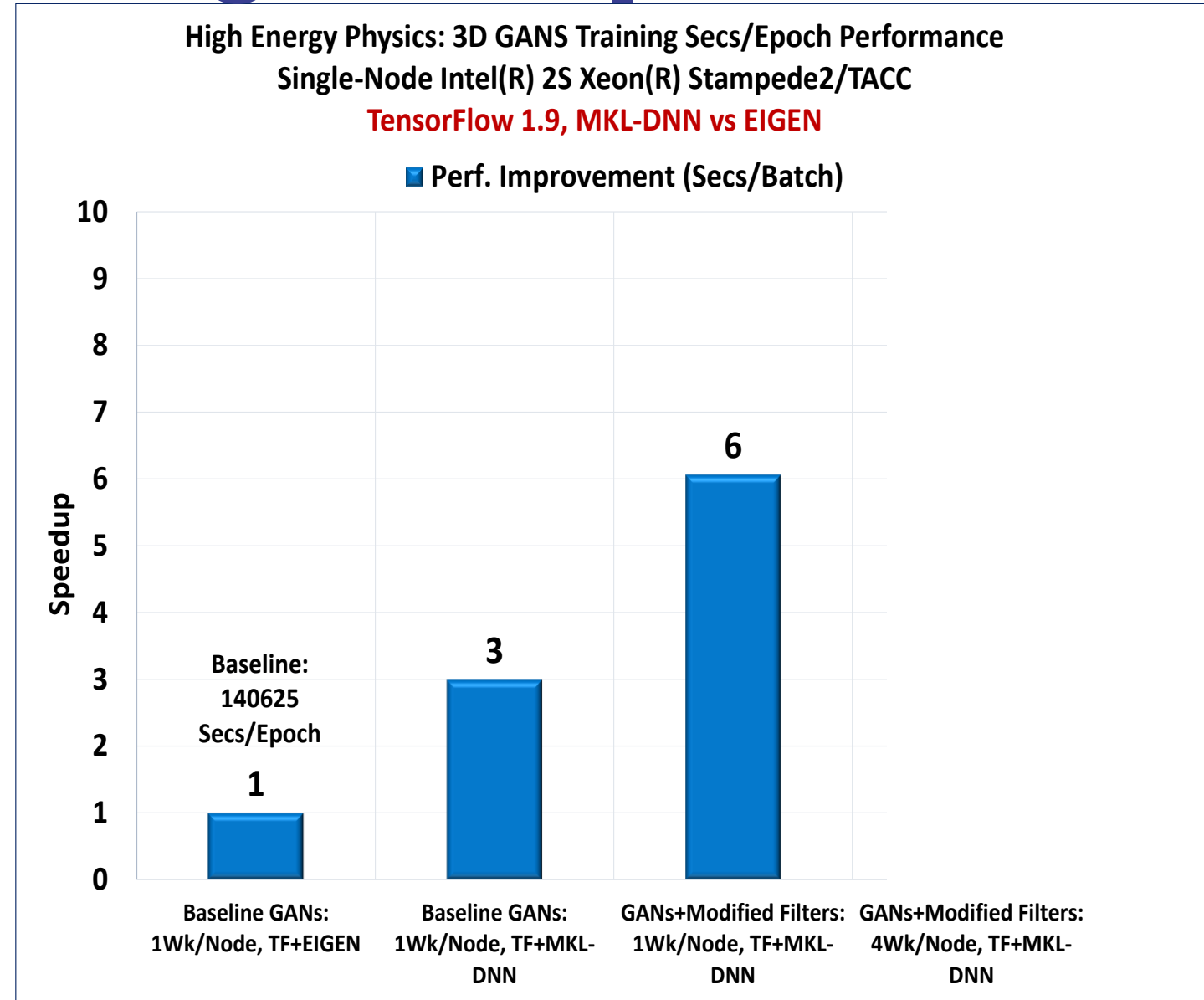
- 1 worker/node TF + Eigen (baseline)
- 1 worker/node TF + MKL-DNN
- 1 worker/node, TF+ MKL-DNN, optimised number of convolution filters
- 4 workers/node, TF+ MKL-DNN, optimised number of convolution filters





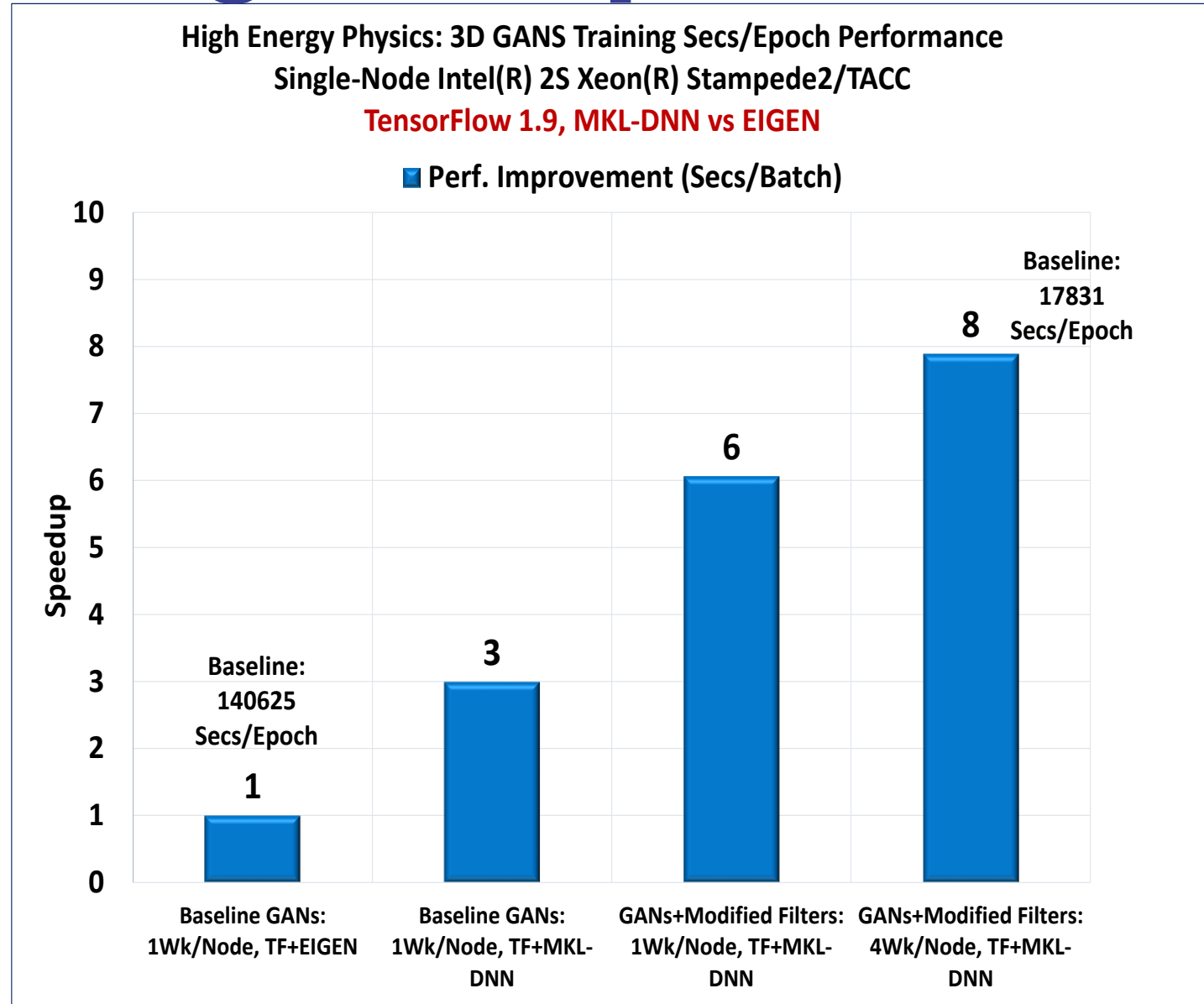
# Single-Node Training Perf. optimisation

- 1 worker/node TF + Eigen (baseline)
- 1 worker/node TF + MKL-DNN
- 1 worker/node, TF+ MKL-DNN, optimised number of convolution filters
- 4 workers/node, TF+ MKL-DNN, optimised number of convolution filters



# Single-Node Training Perf. optimisation

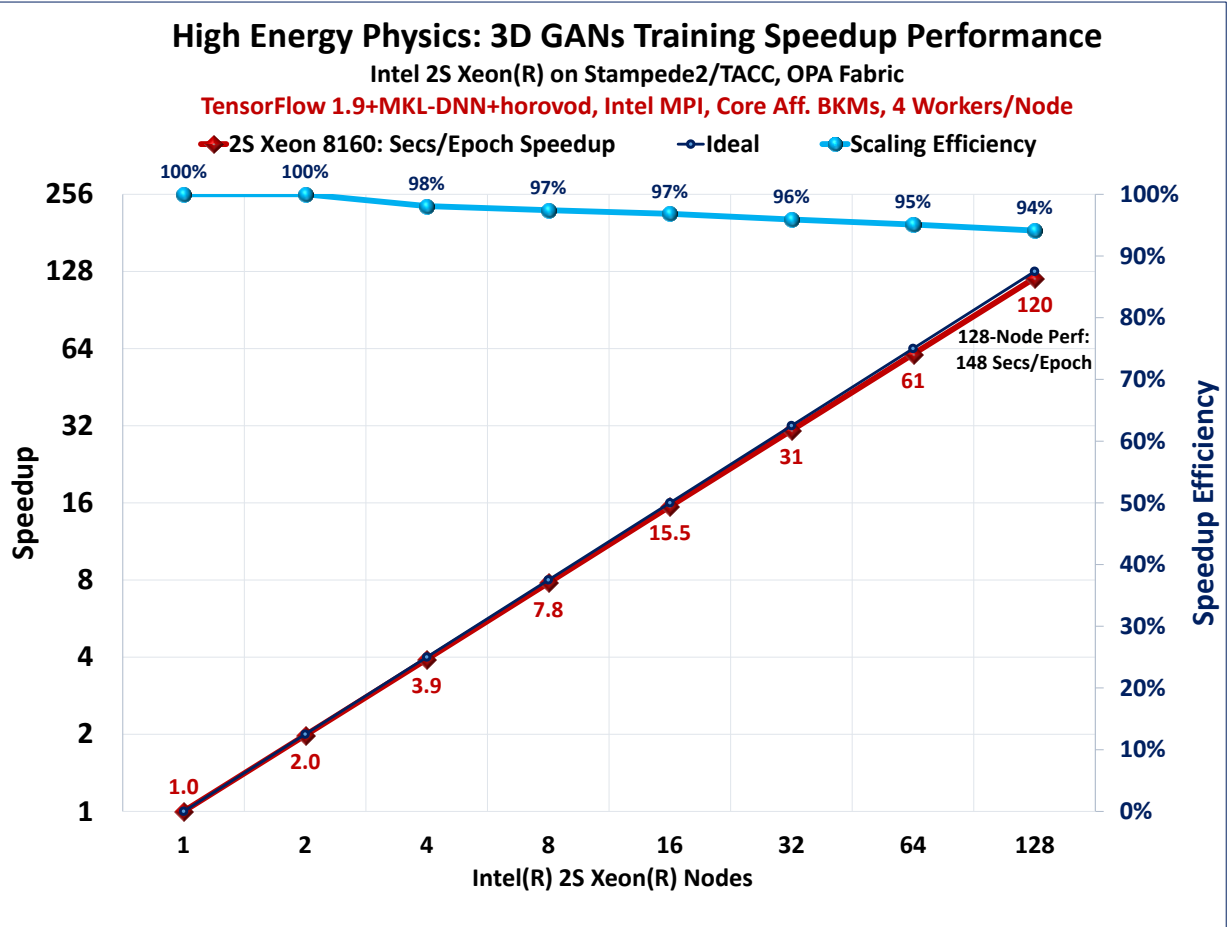
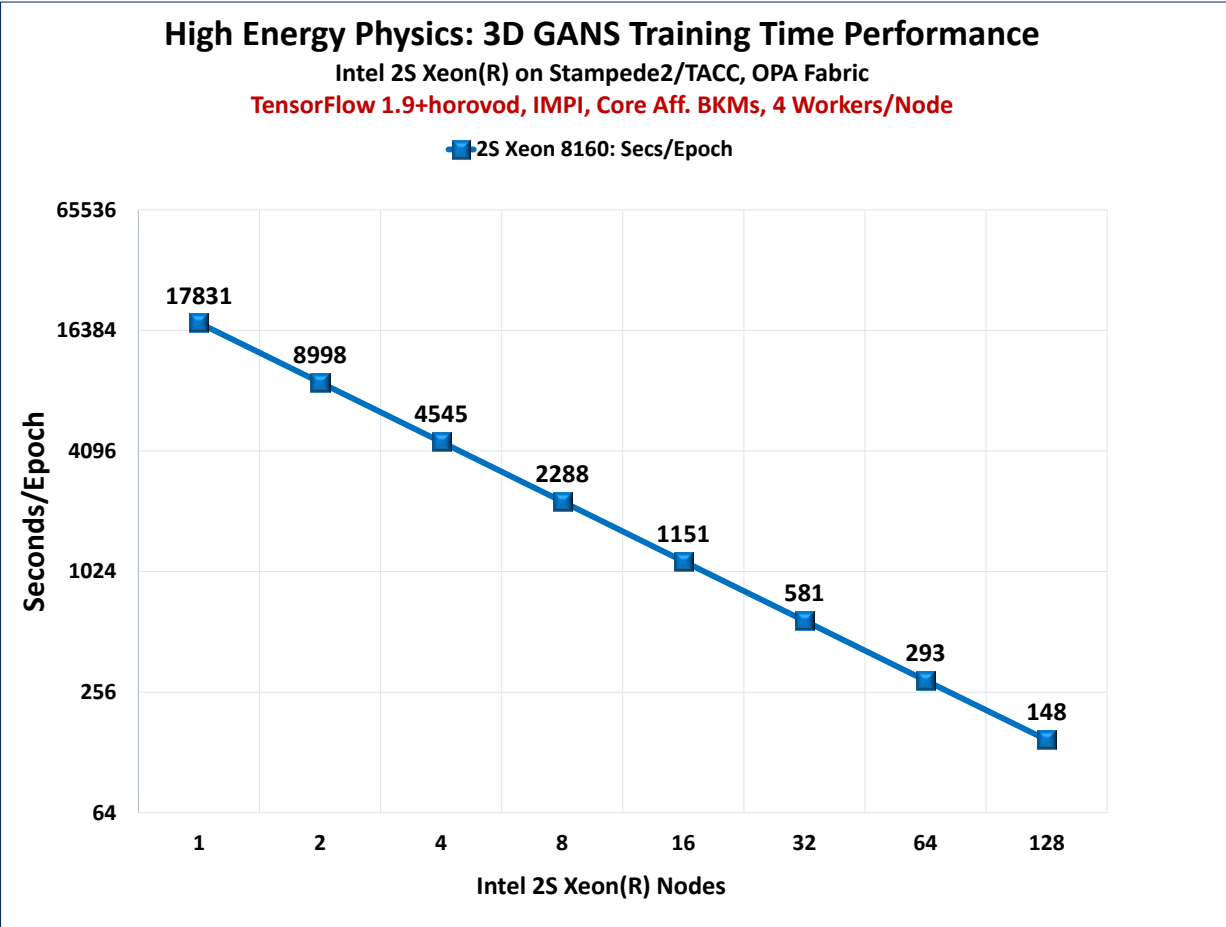
- 1 worker/node TF + Eigen (baseline)
- 1 worker/node TF + MKL-DNN
- 1 worker/node, TF+ MKL-DNN, optimised number of convolution filters
- 4 workers/node, TF+ MKL-DNN, optimised number of convolution filters



# Multi-Node Time/Epoch Scaling Performance

*Distributed training using data parallelism*

94% scaling efficiency up to 128 nodes



# Physics performance at scale

Ratio of Ecal and Ep

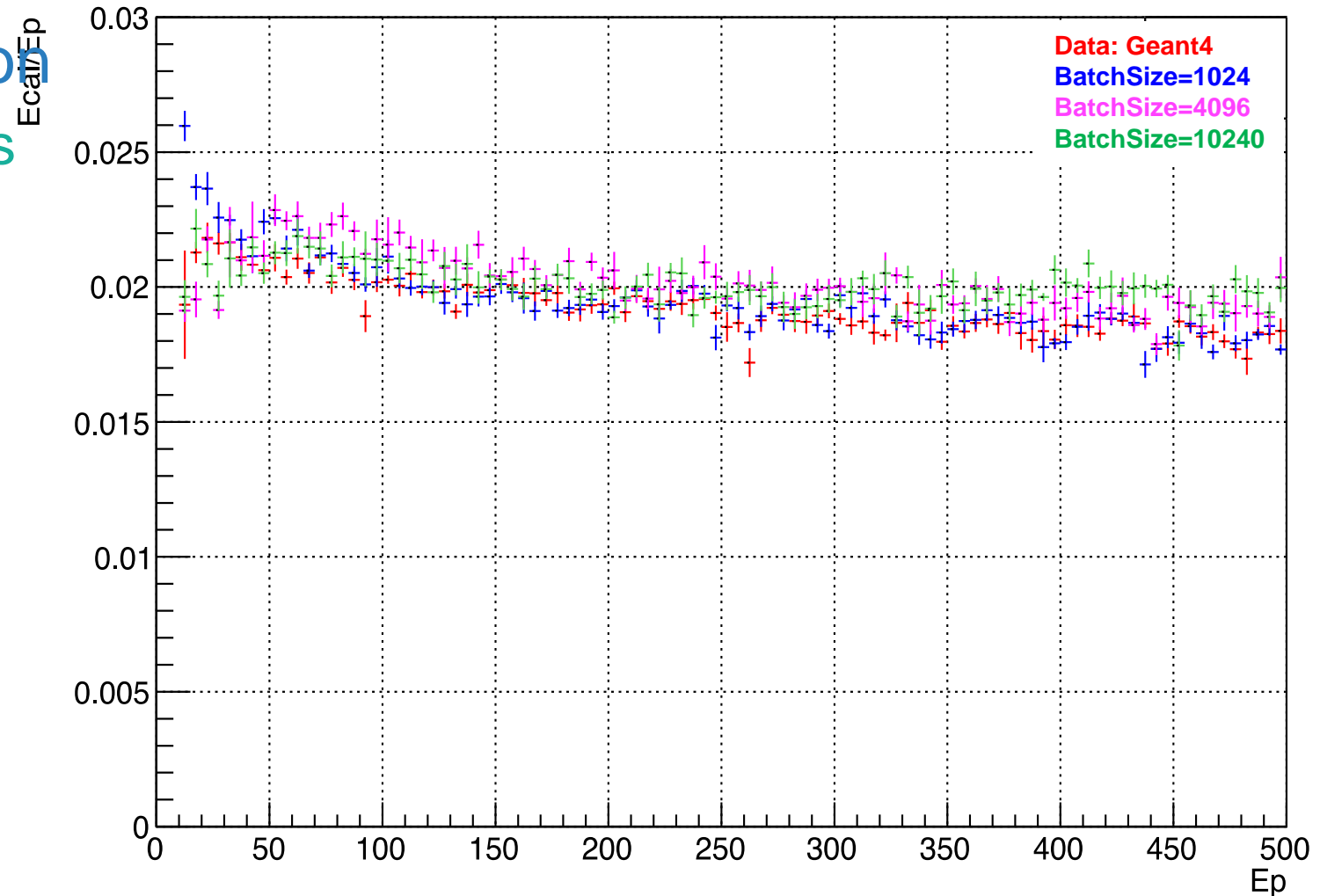
Some performance degradation

Mostly in the low energy regions  
for batchsize

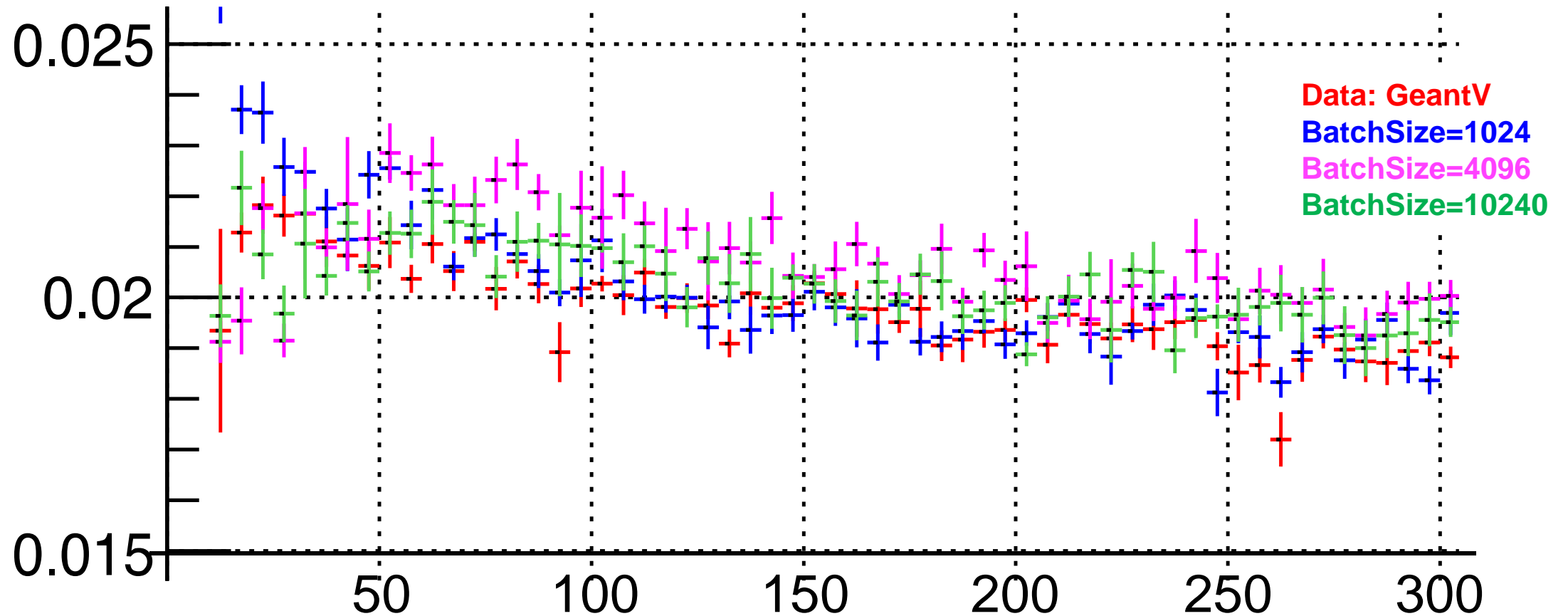
Network optimised for the 100-  
200 GeV central region

Applied warmup and scaling of  
initial learning rate

Further investigation ongoing



# Physics Performance at scale



# Conclusion & Plans

*First results are very promising from physics perspective*

Distributed training process and optimisation to scale on clusters is critical

- Allows meta-optimisation and hyperparameter scans in order to generalize to different detectors

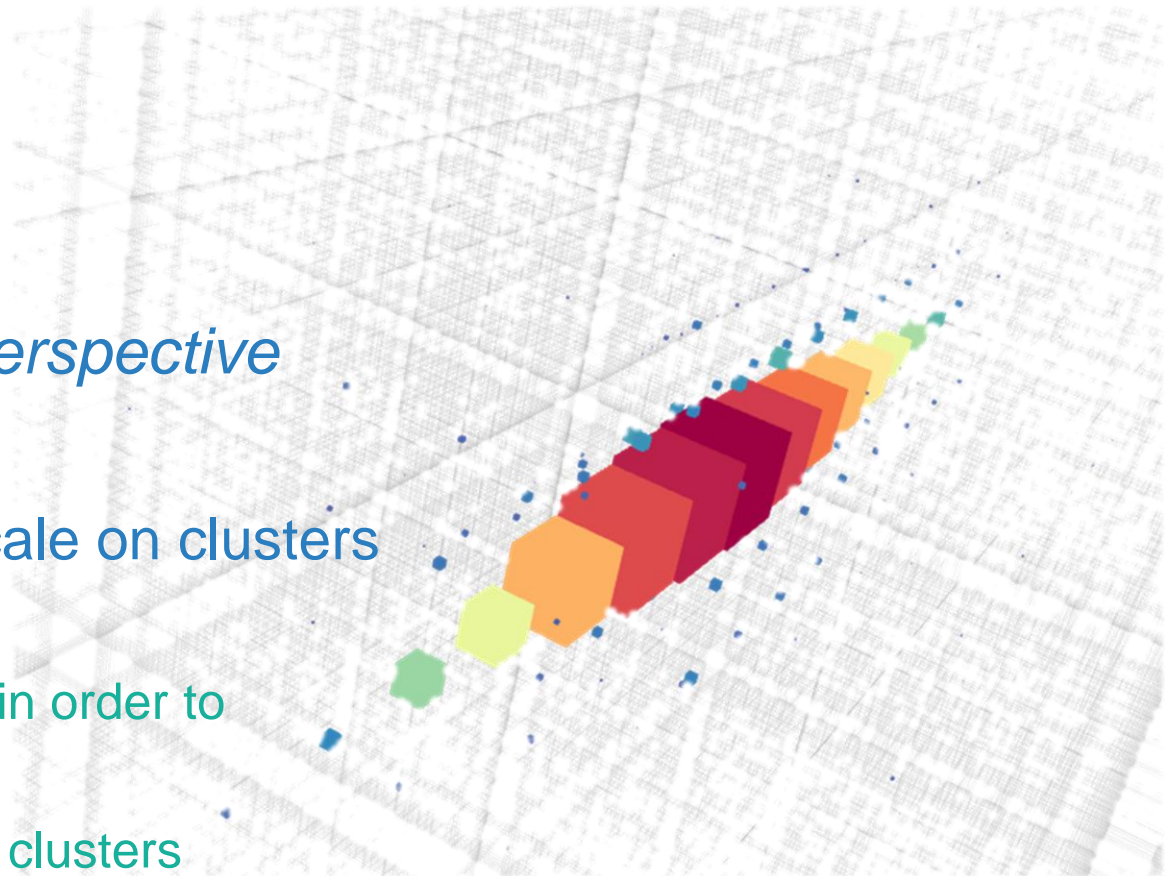
- Parallelizing training process and optimize scaling on clusters

Initial results are very promising

- Reduced training time by x8 on single node

- Linear scaling brings down training time to ~2min/Epoch on 128 nodes

Keep working on the understanding / optimisation of physics performance at scale







# Questions?



# LEGAL NOTICES & DISCLAIMERS

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at [intel.com](http://intel.com), or from the OEM or retailer. No computer system can be absolutely secure.

Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit <http://www.intel.com/performance>.

Cost reduction scenarios described are intended as examples of how a given Intel-based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.

Statements in this document that refer to Intel's plans and expectations for the quarter, the year, and the future, are forward-looking statements that involve a number of risks and uncertainties. A detailed discussion of the factors that could affect Intel's results and plans is included in Intel's SEC filings, including the annual report on Form 10-K.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Performance estimates were obtained prior to implementation of recent software patches and firmware updates intended to address exploits referred to as "Spectre" and "Meltdown." Implementation of these updates may make these results inapplicable to your device or system.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.

Results have been estimated or simulated using internal Intel analysis or architecture simulation or modeling, and provided to you for informational purposes. Any differences in your system hardware, software or configuration may affect your actual performance.

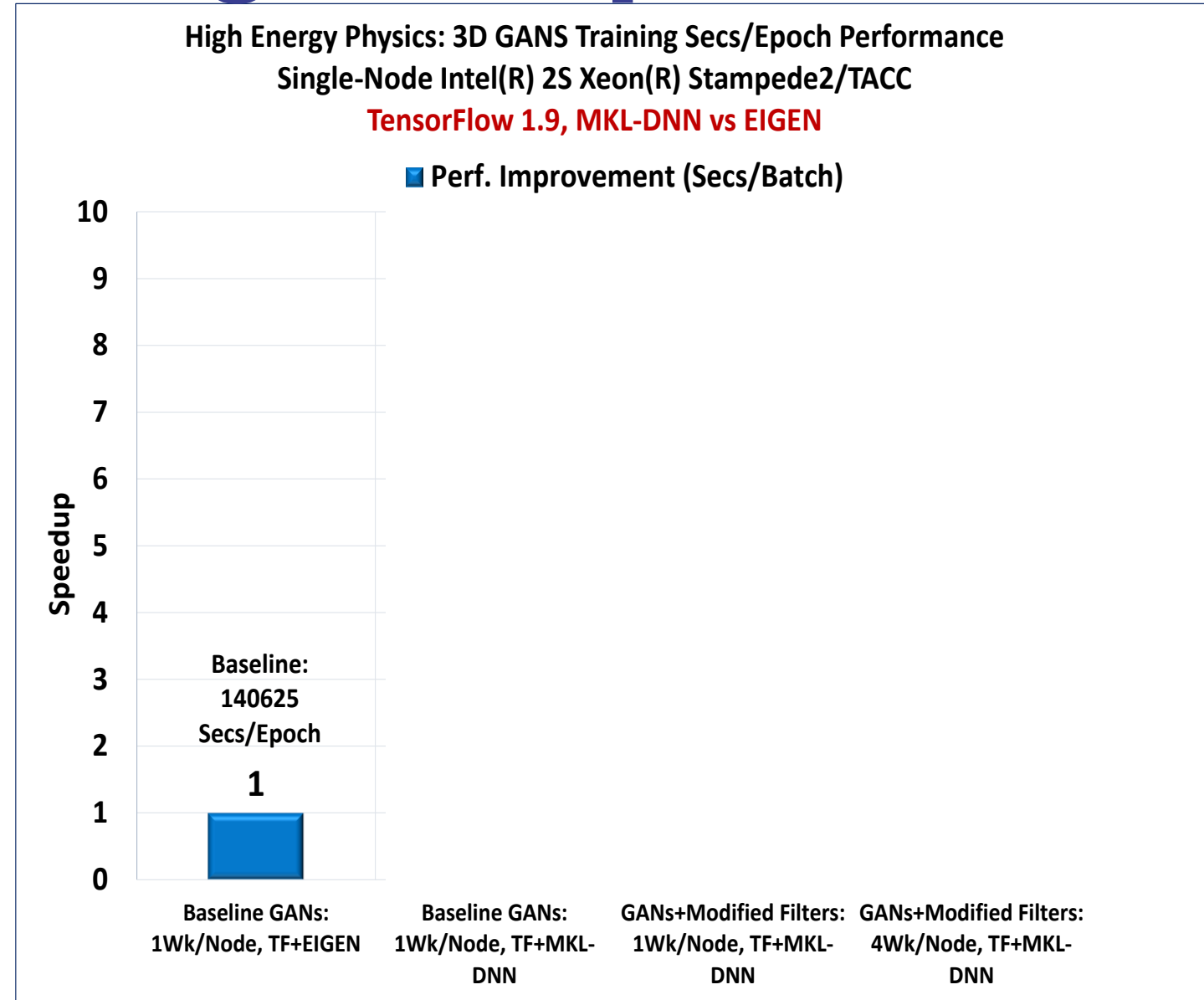
Intel, the Intel logo, Pentium, Celeron, Atom, Core, Xeon, Movidius, Saffron and others are trademarks of Intel Corporation in the U.S. and/or other countries.

\*Other names and brands may be claimed as the property of others.

© 2018 Intel Corporation.

# Single-Node Training Perf. optimisation

- 1 worker/node TF + Eigen (baseline)
- 1 worker/node TF + MKL-DNN
- 1 worker/node, TF+ MKL-DNN, optimised number of convolution filters
- 4 workers/node, TF+ MKL-DNN, optimised number of convolution filters



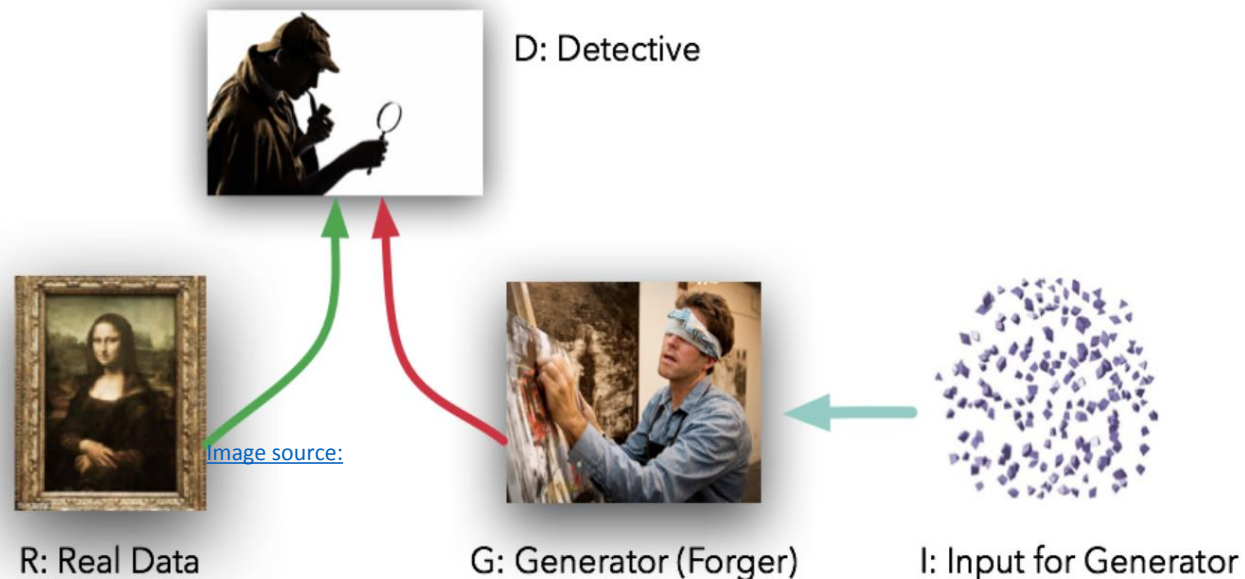


# Generative adversarial networks

*Simultaneously train two networks that compete and cooperate with each other:*

Generator G generates data from random noise

Discriminator D learns how to distinguish real data from generated data



<https://arxiv.org/pdf/1701.00160v1.pdf>

The counterfeiter/detective case

Counterfeiter shows the Monalisa

Detective says it is fake and gives feedback

Counterfeiter makes new Monalisa based on feedback

Iterate until detective is fooled