Towards Detector Agnostic Fast Calorimetry Simulation

Renato Cardoso[1], Peter McKeown[1], Mikolaj Piorczynski[1], Piyush Raikwar[1], Kyongmin Yeo[2], Anna Zaborowska[1]

[1]CERN, Geneva, Switzerland
[2]IBM T. J. Watson Research Center, Yorktown Heights, NY USA

ML4Jets 2024
LPNHE, Paris, France

07.11.2024

Motivation

- Development of machine learning models for fast shower simulation is computationally expensive.
- Moreover, designing model for *each experiment* requires dedicated expertise.

Make FastSim easily available without access to ML expertise.

Motivation

- Development of machine learning models for fast shower simulation is computationally expensive.
- Moreover, designing model for *each experiment* requires dedicated expertise.

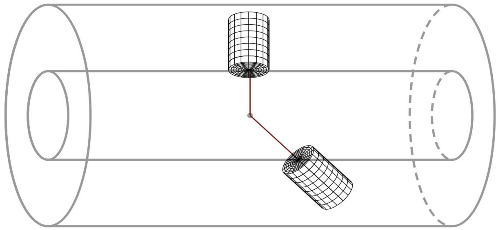
Make FastSim easily available without access to ML expertise.

1. *Generic energy scoring mesh [guide]*
   - Collect energy irrespective of the detector geometry.
   - Ready to use models. (Requires training)
2. *Generalizable ML model*
   - **Train once** on very large & diverse datasets to learn rich representations.
   - **Then adapt** to new detectors, quickly.

The dataset

Energy scoring

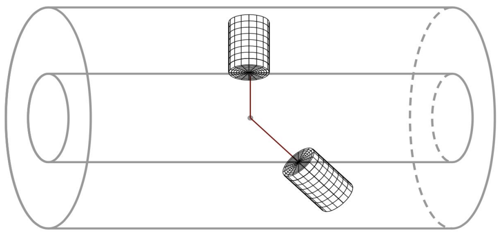
A detector agnostic mesh is constructed to contain the largest shower (CaloChallenge)

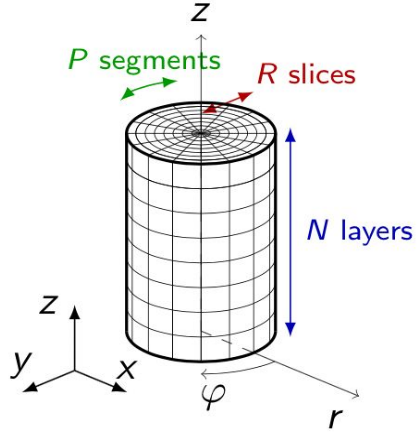


- The mesh aligns with the direction of incident particle.
    - The direction, i.e., the angles are recorded.

Energy scoring

A detector agnostic mesh is constructed to contain the largest shower (CaloChallenge)





- The mesh aligns with the direction of incident particle.
  - The direction, i.e., the angles are recorded.

$r \times \varphi \times z = 9 \times 16 \times 45$

- The size of the cells can vary across detectors according to its X_0 & R_M, but the number of cells remains constant[1].
- Explored in LHCb (CHEP'24 talk)

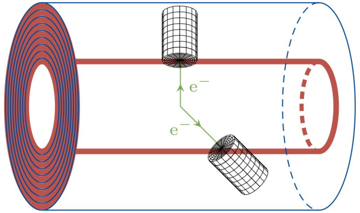
[1] i.e., for a particular model

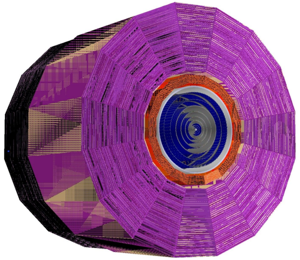
Dataset

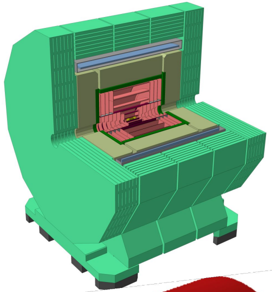
The dataset is constructed by simulating 1M single-photon showers based on the following conditions (continuous range):

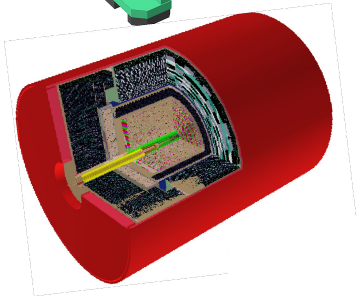
1. Energy (***e***): uniformly sampled from 1 GeV - 1 TeV[1]
2. Azimuthal angle (***φ***): 0 - 2π rad
3. Theta (***θ***): 0.87 - 0.27 rad

[1] 1 GeV - 100 GeV for FCCee detectors









Dataset

The dataset is constructed by simulating 1M single-photon showers based on the following conditions (continuous range):

1. Energy (***e***): uniformly sampled from 1 GeV - 1 TeV[1]
2. Azimuthal angle (***φ***): 0 - 2π rad
3. Theta (***θ***): 0.87 - 0.27 rad

Repeat for multiple detectors: 2 x Par04 (SiW & SciPb), ODD, FCCeeCLD & FCCeeALLEGRO

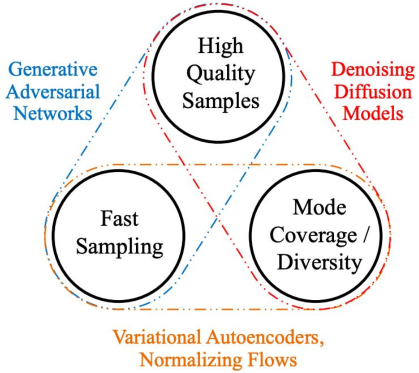
Experimental stats:

- 1M showers per detector
- Training/Validation split - 900K/100K
- We test on multiple sets of point conditions, e.g., 50 GeV at 1.57 & 0 rad (1K showers each)

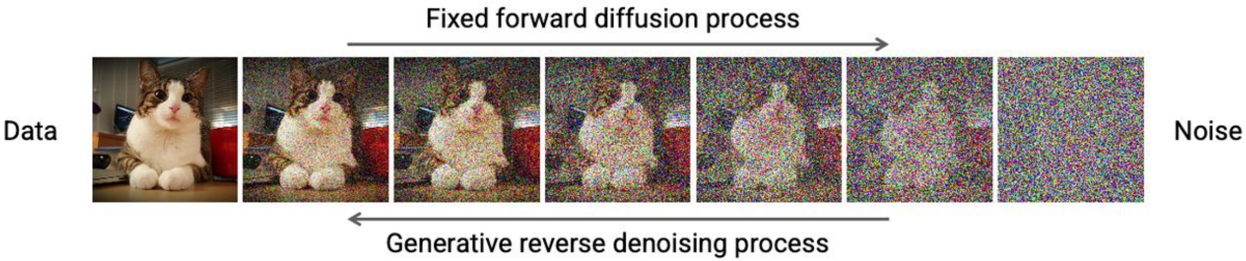
[1] 1 GeV - 100 GeV for FCCee detectors

The ML model

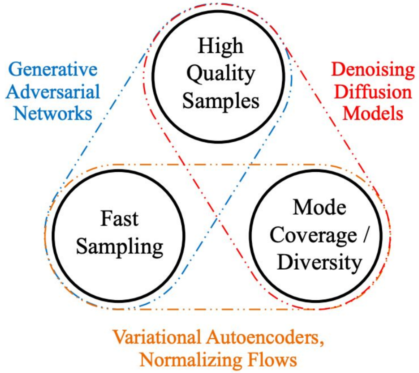
Generative model



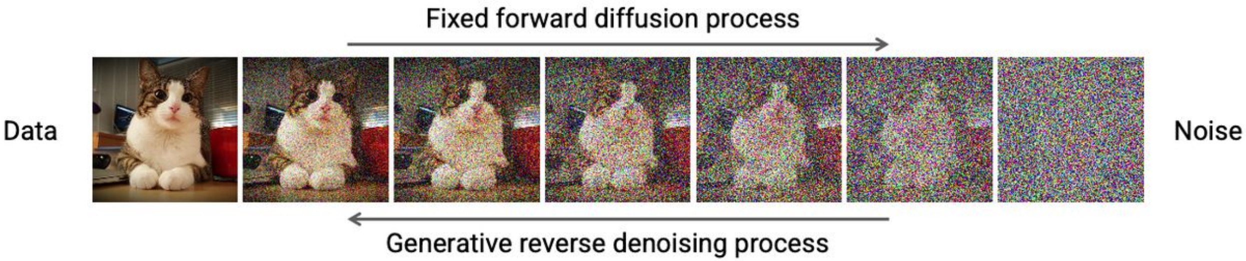
We use a **diffusion model** for higher accuracy and higher diversity.



Generative model



We use a **diffusion model** for higher accuracy and higher diversity.



As for the architecture, we apply **transformer** blocks.

- A **generalized architecture** that works with any type of data, e.g., text, images, audio, etc.
- Models long-range dependencies (**Attention** mechanism).

Diffusion process

We use **EDM** (Elucidating the Design Space of Diffusion-Based Generative Models) which improves over DDPM by:

- Loss reweighting over different timesteps via signal-to-noise ratio (continuous-time diffusion process)
- Reducing input variance introduced by diffusion process by scaling the input
- New samplers for faster inference. We use stochastic 2^{nd}-order Heun sampler with 32 steps for sampling
- Noise distribution that approaches dataset mean at x^T

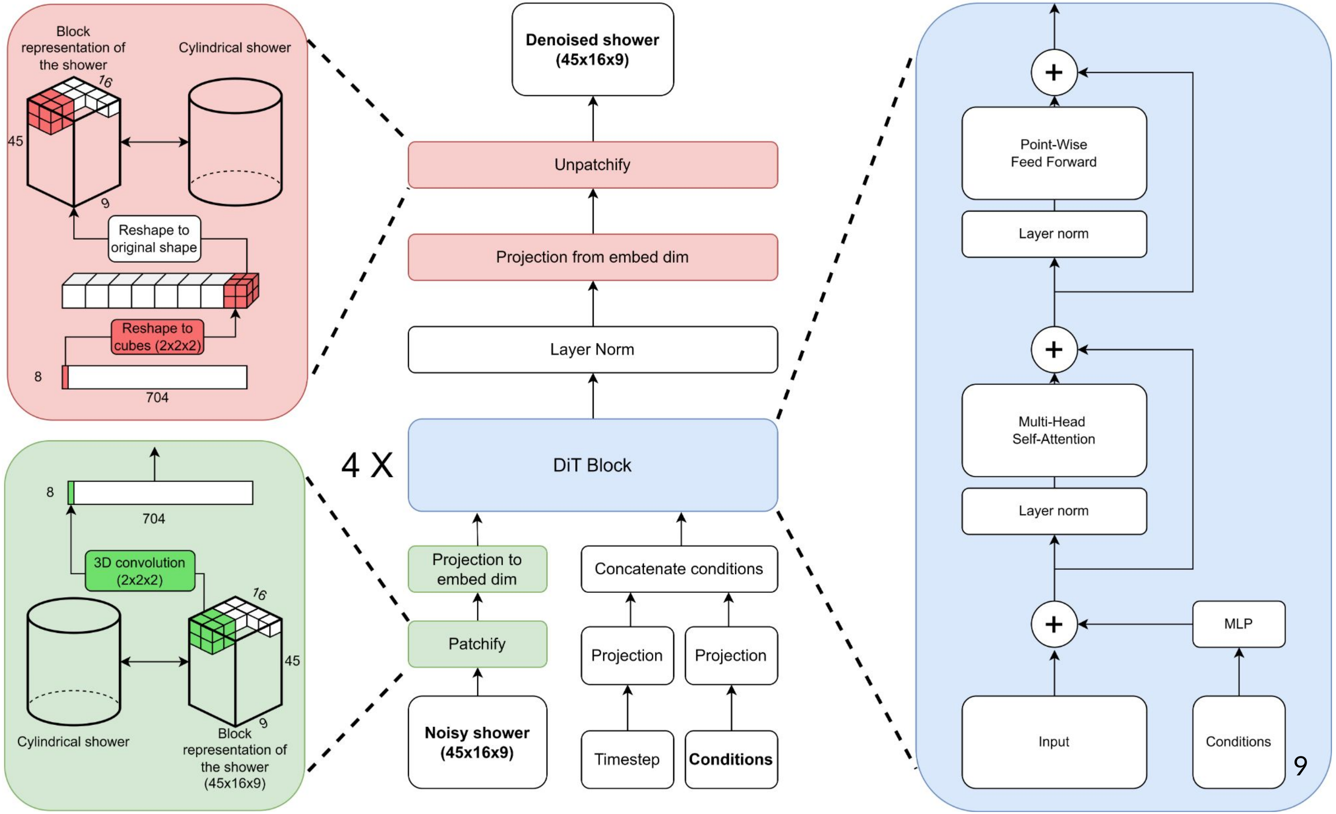
https://arxiv.org/abs/2206.00364

Model architecture

We use **DiT** blocks with modified patching and positional embedding more suited for our 3-dimensional “images”

Hyperparams:

- Embed dim = 144
- MLP ratio = 4x
- 4 DiT blocks



https://arxiv.org/abs/2212.09748

Model architecture

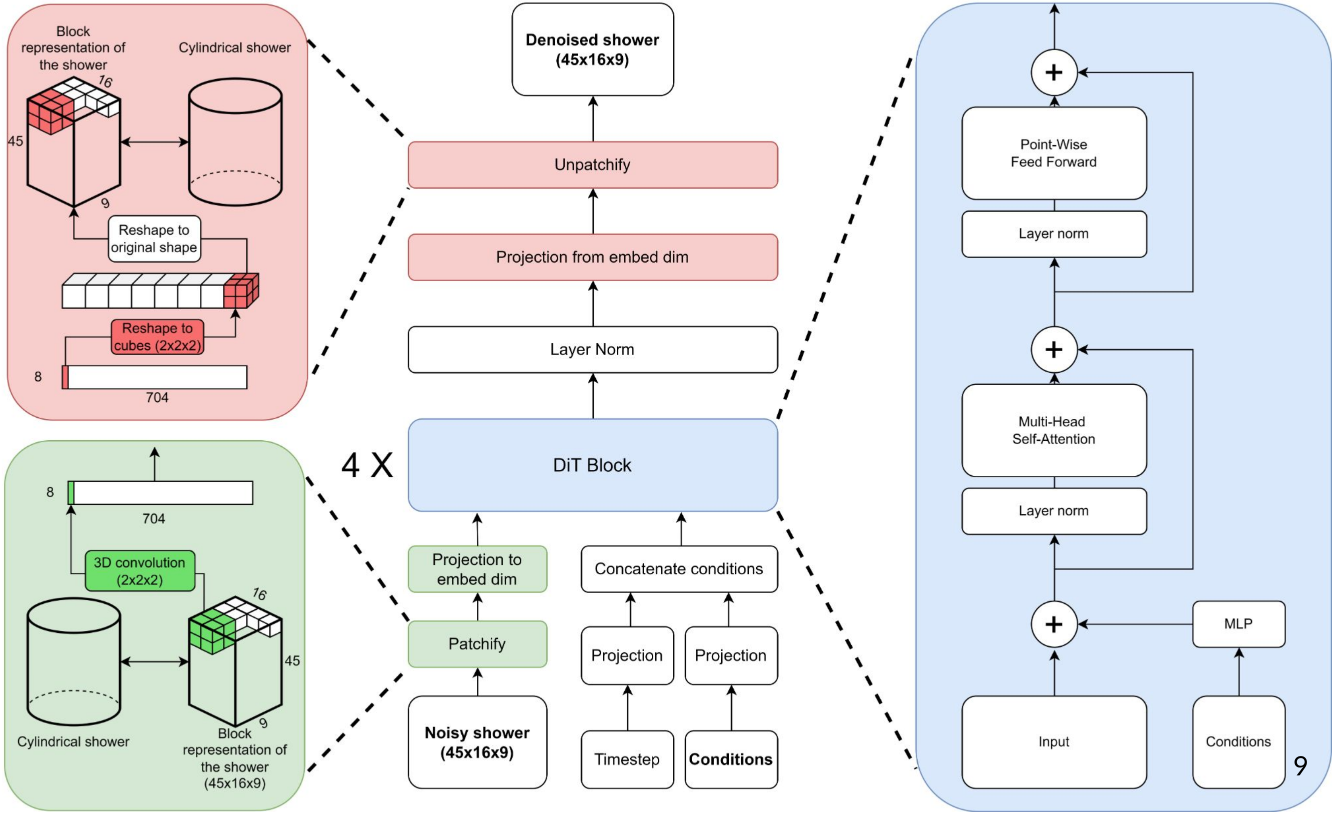
We use **DiT** blocks with modified patching and positional embedding more suited for our 3-dimensional "images"

Hyperparams:

- Embed dim = 144
- MLP ratio = 4x
- 4 DiT blocks

Preprocessing:

- Showers: log & **"global norm"**
- ***e*, *θ***: linear scaling
- ***φ***: sinusoidal
- **Detector: K+1 dim one-hot encoding**



https://arxiv.org/abs/2212.09748

The results

Experiments

1. Training the diffusion model on single detector
    - Trained on Par04-SiW detector
    - *The results are significantly better than non-diffusion models*
2. Multi-geometry training
    - Detectors - Par04-SiW, Par04-SciPb, ODD & FCCeeCLD
    - *Addition of detectors did not affect the accuracy*
3. Adaptation
    - Finetune the pre-trained model on the desired detector - FCCeeALLEGRO
    - This can be any detector, the model does not have to know it beforehand

Distillation

- To make the sampling process of the diffusion model faster

Experiments

1. Training the diffusion model on single detector
   - Trained on Par04-SiW detector
   - *The results are significantly better than non-diffusion models*

Product-side {
2. Multi-geometry training
   - Detectors - Par04-SiW, Par04-SciPb, ODD & FCCeeCLD
   - *Addition of detectors did not affect the accuracy*

Client-side {
3. Adaptation
   - Finetune the pre-trained model on the desired detector - FCCeeALLEGRO
   - This can be any detector, the model does not have to know it beforehand

Product/**Client-side** {

Distillation
   - To make the sampling process of the diffusion model faster

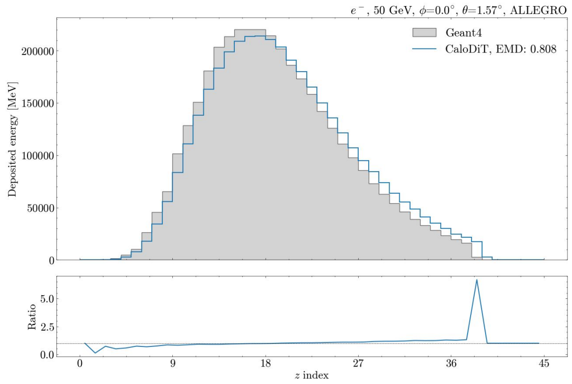
Adaptation vs Scratch

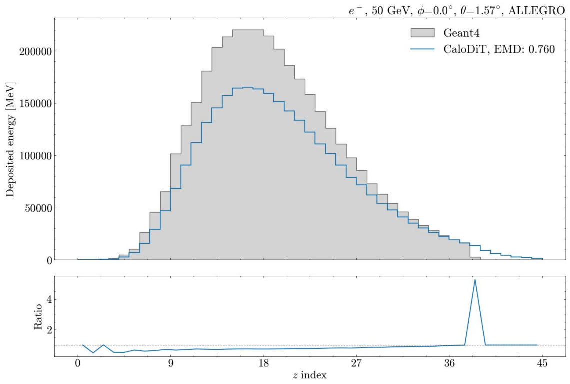
SGD steps

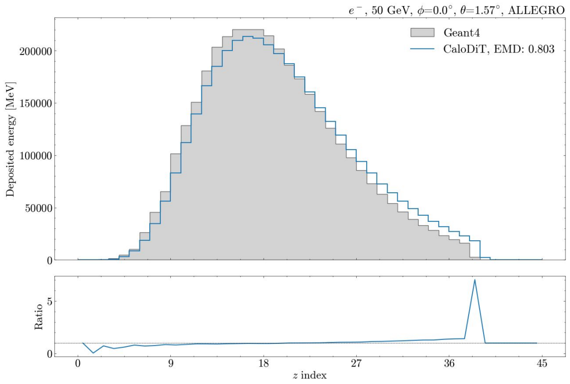
Adaptation @ 10K

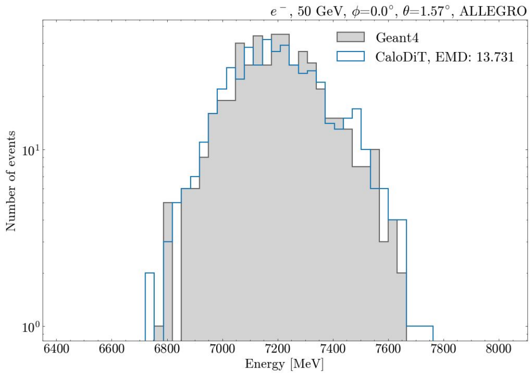
Scratch @ 10K

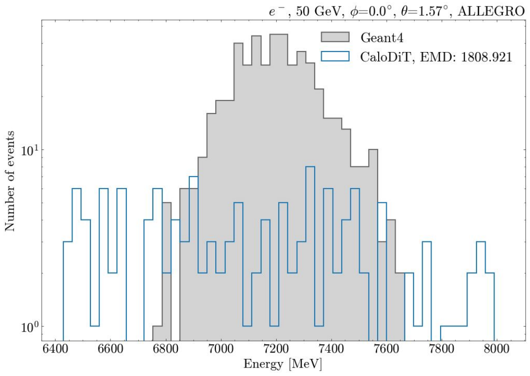
Scratch @ 100K

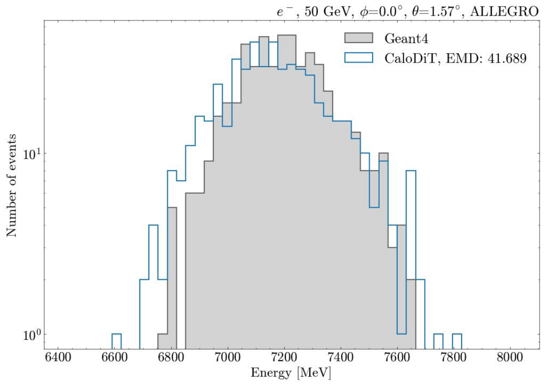










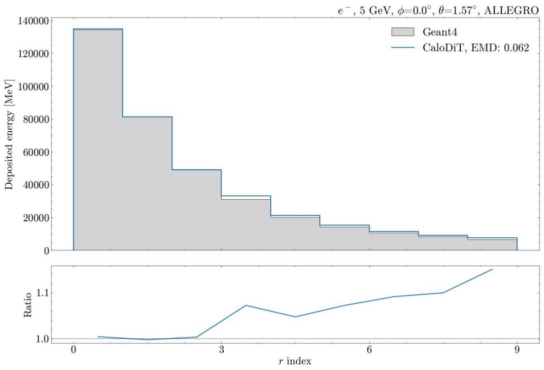


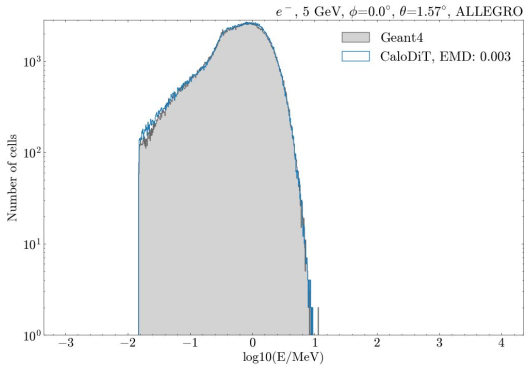
Note: Both adaptation and training from scratch is done on 100K samples

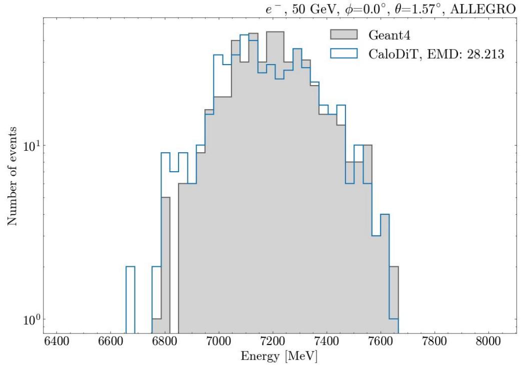
~10x less steps

What if just 1K samples?

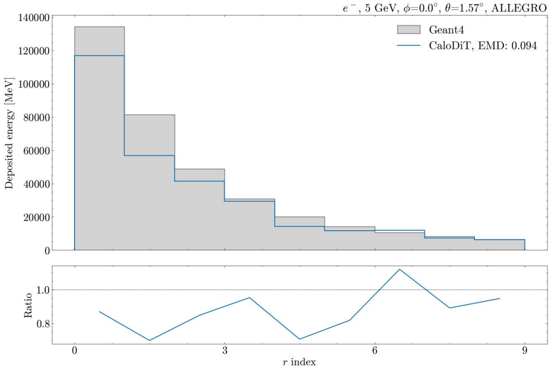
Adaptation @ 10K

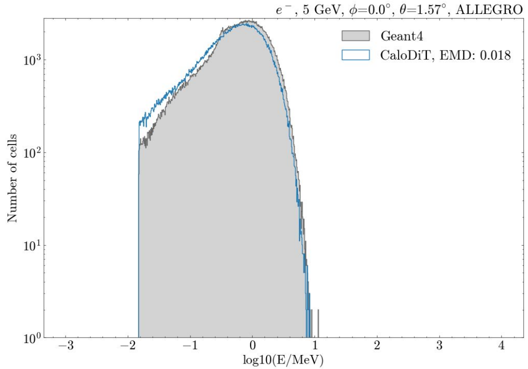


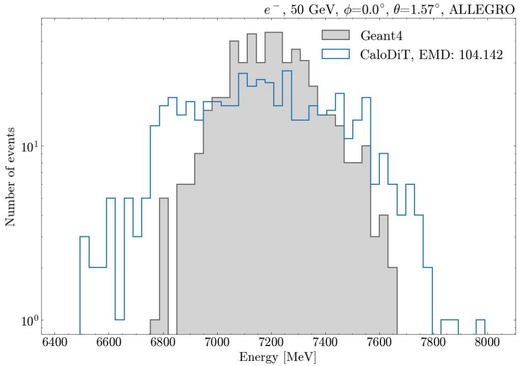




Scratch @ 100K







Note: Dataset size would depend on the complexity of the detector. This is an example based on a preliminary study **Needs a lot less data**

Speeding up the inference

Taking longer steps

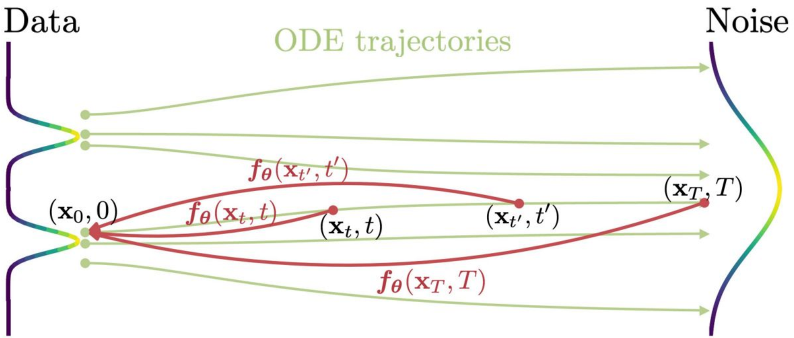
Diffusion models need to iterate over multiple diffusion steps, which leads to slow sampling process.

- Reduce the number of steps to reach the final image given the noise
  - Faster samplers, e.g., DDIM, DPM-Solver++
  - Distillation, e.g., Progressive distillation to condense information
- First step, EDM - more stable diffusion process
- Second step, **consistency distillation**

Taking longer steps

Diffusion models need to iterate over multiple diffusion steps, which leads to slow sampling process.

- Reduce the number of steps to reach the final image given the noise
  - Faster samplers, e.g., DDIM, DPM-Solver++
  - Distillation, e.g., Progressive distillation to condense information
- First step, EDM - more stable diffusion process
- Second step, **consistency distillation**

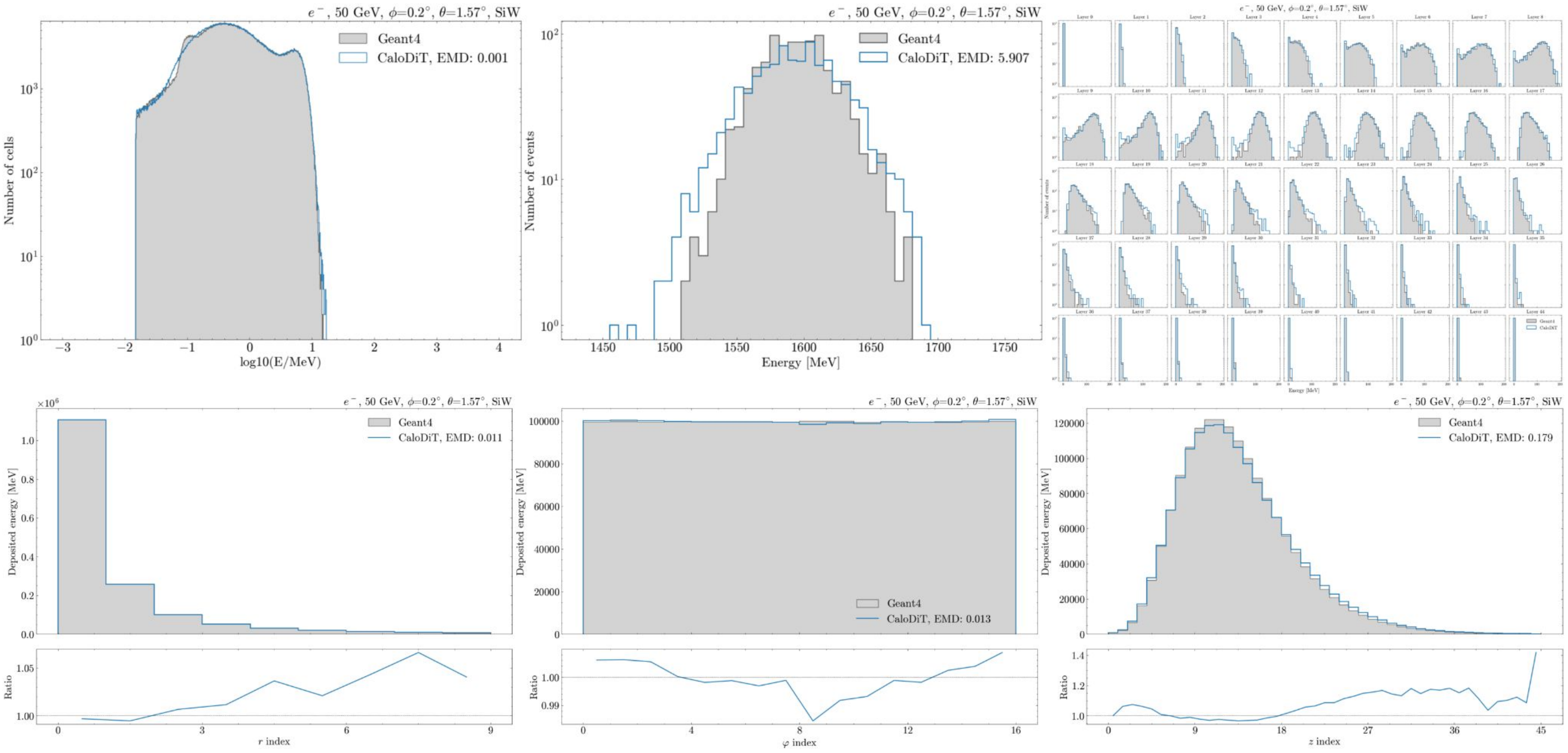


Map any point on the ODE trajectory to a fixed initial point, thus achieving *consistency*

- Single-step diffusion model

https://arxiv.org/abs/2303.01469

Results - Distilled model



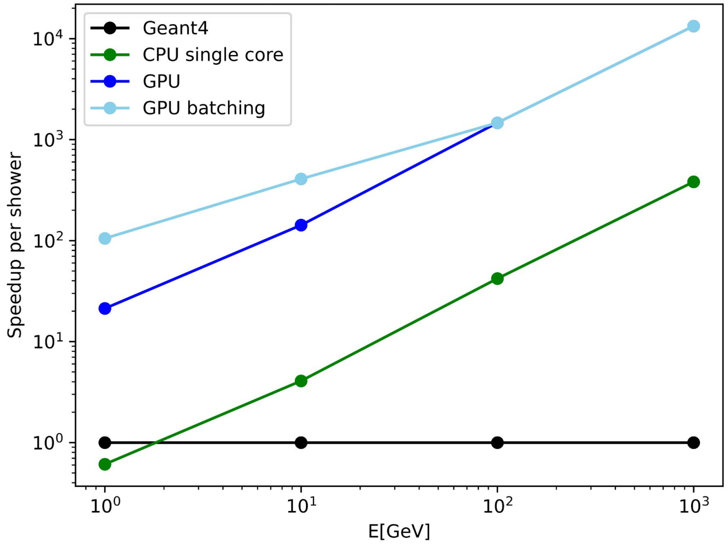
Note: Distillation is done separately for now on Par04 detector

Timings

- We benchmark our distilled model with single diffusion step relative to Geant4
- Geant4 times are based on γ interactions with Par04 geometry
- Placement of hits and batch size is taken into consideration. More details here

Hardware:

- CPU - AMD EPYC 9334
- GPU - NVIDIA RTX 6000 Ada



Speedup relative to Geant4

Conclusion & Next steps

- We present a detector agnostic fastsim model, easily adaptable to new detectors
- The results are highly promising which significantly reduces the required statistics, and training time from days to just a couple of hours[1]
- We get to an impressively low number of diffusion steps with the distilled model, which make this on par with VAE, GANs

What's next?

- Tuning the model architecture. Exploring lightweight attention mechanisms
- Investigating how distillation affects adaptation
- Looking at reconstruction and physics level observables
- Testing our framework in experiments
    - The mesh is already implemented in Gaussino and DD4hep
    - Work started for ATLAS

[1] Depending on the model size of course

Thank you for listening!

Questions?

piyush.raikwar@cern.ch

Backup

Pipeline for the client

1. User has access to a pre-trained model
2. User adapts the given model to their geometry
3. User distills the adapted model to make is faster
4. The model is ready