IRMP-CP3-26-01, MBI-ML-26-01

A universal vision transformer for fast calorimeter simulations

Luigi Favaro[1], **Andrea Giammanco**[1] **and Claudius Krause**[2]

[1]Centre for Cosmology, Particle Physics and Phenomenology (CP3),
Université catholique de Louvain, Louvain-la-Neuve, Belgium

[2]Marietta Blau Institute for Particle Physics (MBI Vienna),
Austrian Academy of Sciences (ÖAW), Austria

E-mail: luigi.favaro@uclouvain.be, andrea.giammanco@cern.ch,
claudius.krause@oeaw.ac.at

Abstract. The high-dimensional complex nature of detectors makes fast calorimeter simulations a prime application for modern generative machine learning. Vision transformers (ViTs) can emulate the Geant4 response with unmatched accuracy and are not limited to regular geometries. Starting from the CaloDREAM architecture, we demonstrate the robustness and scalability of ViTs on regular and irregular geometries, and multiple detectors. Our results show that ViTs generate electromagnetic and hadronic showers statistically indistinguishable from Geant4 in multiple evaluation metrics, while maintaining the generation time in the $\mathcal{O}(10-100)$ ms on a single GPU. Furthermore, we show that pretraining on a large dataset and fine-tuning on the target geometry leads to reduced training costs and higher data efficiency, or altogether improves the fidelity of generated showers.

Keywords: fast detector simulations, generative networks, conditional flow matching, transformers

Submitted to Machine Learning: Science and Technology

1 Introduction

Particle Physics is a numerically intensive, data-processing and simulation-heavy science. The large experiments ATLAS and CMS at the Large Hadron Collider (LHC) record data at a staggering rate of several GB/s [1], totaling now one Exabyte in recorded data [2] at CERN. Simulation of collisions, the major backbone of statistical data analyses, needs to keep up with the amount of data as well. The upcoming runs of the LHC and the high-luminosity phase will increase the amount of required computing even further. To fully exploit the data and learn about the underlying laws of nature, it is therefore essential to develop efficient algorithms in every part of the analyses or simulation chains. Modern machine learning (ML) has the potential to contribute substantially to this endeavor [3, 4], for example, by accelerating computationally intensive bottlenecks and opening up new avenues for efficient analyses [5]. The rise of generative ML in computer science has introduced many new ideas for more efficient simulation in high-energy physics (HEP) in recent years, in particular for amplitude evaluation [6, 7, 8], phase space generation [9, 10, 11, 12, 13], (end-to-end) event generation [14, 15, 16, 17, 18, 19], and detector simulation (see [20, 21] for recent reviews on the vast literature). First-principled simulations of the latter are computationally expensive and comprise a significant portion of the overall computing budget [22, 23], so any kind of improvement has a direct and strong impact on the global computing efficiency. In addition to being faster than the traditional simulation based on Geant4 [24, 25, 26], generative networks have the capability to oversample, meaning they can amplify the statistical properties of the training dataset [27, 28, 29, 30, 31]. Alternatively, individual steps of the simulation chain [3] can be combined into a single generative network [32, 33, 34].

Despite all these advantages, generative ML networks for fast simulation are still computationally expensive. First, the training data need to be generated by traditional simulations, and second, the generative networks need to be trained. When switching to a new detector layout or even just changing the voxelization of a given geometry, the generative networks need to be retrained completely. It is therefore highly beneficial to make the overall training of the networks more efficient. For example, the voxelization that is applied to the raw hits can be adopted to better reflect the types of showers under consideration. Areas of larger activity would have a finer read-out and areas with less activity would be coarse [35]. As a result, the number of voxels to be considered in the subsequent training can be reduced, and the networks can be smaller. However, state of the art ML networks often assume regular geometries and mapping irregular voxelizations to regular space comes at increased computational costs.

As another alternative, one can keep the general setup the same, but investigate how the training of generative networks can be made more efficient. For this, we start with an observation regarding calorimeter showers from different incident particles, detector geometries, and detector materials: Even though the specific details are very different, the showers still have many things in common:

- Sparsity: A single shower deposits energy only in a fraction of the voxels, and most of the voxels will not receive an energy deposition at all.
- Dynamic range: The energy that is deposited in the voxels spans several orders of magnitude and, in general, scales with the incident energy. Since a common approach to calorimeter simulation with generative ML is to split off the scale of the energy that is deposited from the normalized shower shapes [36], the latter becomes less sensitive to the incident energies.
- Spatial correlations: Showers emerge as spatially connected clusters, in some cases (depending on incident particle and detector materials), even tracks of individual particles become visible, thereby strongly correlating the energy depositions of nearby voxels.
- Central activity: Especially for electromagnetic showers, the main activity will be at

the center of the considered volume and will form a single cluster.

Motivated by these observations and how different these distributions are from the standard normal distributions we usually assume in the latent space of the generative networks, we investigate transfer learning for calorimeter showers: instead of training the generative network from scratch, i.e. randomly initialized network weights, we start the training from weights that were previously optimized for another dataset. While this could still involve different incident particles, detector materials, detector layouts, or voxelizations, the differences in the pre-processed distributions between the datasets are still rather small. Fine-tuning is therefore more efficient because the network has to learn a smaller shift in how the data are mapped to the Gaussian latent space.

While similar in spirit to foundation models [37, 38, 39, 40, 41, 42, 43, 44, 45], which are pretrained once to a big dataset and then fine-tuned to the application at hand, we prefer the term transfer learning in this case, as we only consider generative tasks (on different datasets) and not other tasks like classification or regression as one usually would for a foundation model. Transfer learning was studied before in HEP [46, 47, 48, 49, 50], but mainly in the context of classification tasks. In the context of detector simulation, transfer learning has recently also been studied for detectors with fixed voxelization in [51] and point-cloud architectures in [52]. In this work, we propose a general fine-tuning strategy for vision transformers applicable to any detector geometry and across particle types. Additionally, we present complete benchmark results on the LEMURS dataset [53] used for pretraining and discuss the importance of evaluation metrics for a proper evaluation of the efficiency gains.

The paper is organized as follows. In Section 2, we introduce conditional flow matching and CaloDREAM, the generative networks we study. Section 3 discusses the evaluation metrics we employ to study the performance of the generative architectures. Section 4 and Section 5 show the performance of our generative network on datasets of regular and irregular geometries, respectively. In Section 6 we finally get to discuss the transfer learning and show how the pretrained CaloDREAM adapts to new datasets. We conclude in Section 7. In the appendices, we show additional high-level feature distributions of the datasets as well as a hyperparameter study.

2 Methods

2.1 Conditional Flow Matching

Our approach uses continuous normalizing flows trained with Conditional Flow Matching (CFM) [54] as the underlining generative network. This class of generative networks parametrizes the transition from the data to the latent space as an ordinary differential equation (ODE):

$$\frac{dx(t)}{dt} = v(x(t), t) \qquad \text{with} \qquad x \in \mathbb{R}^d \,, \tag{1}$$

where the velocity field $v(x(t),t) \in \mathbb{R}^d$ matches the dimensionality of the data. A differential equation of this form can be related to the underlying density through the continuity equation

$$\frac{\partial p(x,t)}{\partial t} + \nabla_x \left[p(x,t)v(x,t)\right] = 0 \; . \tag{2}$$

The continuous transformation of the density $p(x,t)$, parametrized by t, should satisfy the boundary conditions

$$p(x,t) \to \begin{cases} \mathcal{N}(x|0,1) & t \to 0 \\ p_\text{data}(x) & t \to 1 \; . \end{cases} \tag{3}$$

CFM is a simple prescription to train continuous normalizing flows upon selecting a conditional target trajectory. A standard choice is a linear trajectory of the form

$$\begin{aligned} &x(t|\epsilon, x_0) = (1-t)\epsilon + t x_0, \quad \text{with} \\ &\epsilon \sim \mathcal{N}(0,1), \quad x_0 \sim p_\text{data}(x) \; . \end{aligned} \tag{4}$$

Given Gaussian distributed random numbers, a training dataset, and uniformly distributed times, we can approximate the true velocity field with a neural network $v_\Theta(x(t),t)$. Using linear trajectories, the network is optimized using a simple mean-squared error loss of the form

$$\mathcal{L}_\text{CFM} = \left\langle \left[v_\Theta(x(t|\epsilon,x_0),t) - (x-\epsilon)\right]^2 \right\rangle_{t\sim U(0,1),\, \epsilon\sim\mathcal{N},\, x\sim p_\text{data}} . \tag{5}$$

Conditional probability distributions can be learned by allowing v_Θ to depend on additional inputs. Sampling from the trained network requires solving the ODE with the learned velocity field,

$$x(t=1) = x(t=0) + \int_0^1 \mathrm{d}t \; v_\Theta(x(t),t) \tag{6}$$

This is typically done numerically with standard ODE solvers such as Runge-Kutta methods, or more advanced Bespoke samplers[55].

2.2 CaloDREAM

We briefly review the core concepts of CaloDREAM [56] before discussing the extended setting needed for our studies. CaloDREAM combines two neural networks to generate the final calorimeter showers. An "energy network" produces the layer energy ratio variables u, as defined in [36], conditioned on the incident energy E_inc of the incoming particle. More generally, we refer to the set of global incident particle conditions as C. These may contain the incident energy, but also the incoming direction of the particle parametrized with the azimuthal and polar angles, and the particle type. The "shape network" learns the conditional distribution for the normalized voxels x, given all the other energy and conditional variables. We train the two networks independently, and the generation process follows the sequential steps:

$$\begin{aligned} u &\sim p_{\Theta_e}(u|C) && \text{energy variables,} \\ x &\sim p_{\Theta_s}(x|C,u) && \text{normalized voxels,} \end{aligned} \tag{7}$$

where Θ_e and Θ_s are the learnable weights of the energy and shape network, respectively. An analytic formula allows for the extraction of the layer energies from the generated ratios, which are used to rescaled the normalized voxels. A similar factorization with a ViT shape network can also be done with normalizing flow for faster generation at the cost of accuracy [57].

Energy network. The original CaloDREAM energy network built an embedding vector for each energy ratio from the incident energy and the energy ratios from the previous layers. Sampling from this network required the sequential generation of layer energies, hence solving an ODE for each energy ratio. We accelerate the generation, especially for small batch sizes, by avoiding this autoregressive sampling in favor of a parallel sampling of the full velocity vector. The construction of the conditional information is unchanged. A transformer encoder-decoder network takes as inputs the energy u-vector and the global shower information C. The encoder processes an embedding vector of fixed size l constructed as

$$s_i = [C_i, \text{onehot}(C_i), \vec{0}] \,, \qquad s_i \in \mathbb{R}^l \,, \tag{8}$$

where the index i runs over the number of conditions, the one-hot encoding vector specifies the encoded global variable, and the zeros act as a zero-padding vector. Similarly, the input vector for the decoder network is constructed as

$$t_i = [u_i, \text{onehot}(u_i), \vec{0}] \,, \qquad t_i \in \mathbb{R}^l \,. \tag{9}$$

The final output embedding vector is computed from a cross-attention step between the encoder and decoder latent vectors. Unlike the original CaloDREAM, we encode the conditional information in a single step, which means that the vector c is written as

$$c(u_0(t), \dots, u_N(t), C) \,, \tag{10}$$

where N is the number of detector layers. We finally pass this information to a multi-layer perceptron (MLP) with two linear layers and a sigmoid linear unit non-linearity, which predicts the full velocity field. Altogether, at each time step, we require a single network prediction during both training and inference, which can be written as

$$v_{\Theta_e}(u(t), C, t) = \text{MLP}\left(c(u_0(t), \dots, u_N(t), C), \mathcal{F}(t)\right) \,, \tag{11}$$

where $\mathcal{F}$ indicates a Gaussian Fourier projection, typically used to encode the time information [58]. The training hyperparameters of the energy network are given in Appendix A.

Shape network. The shape network contains the majority of the learnable weights and constitutes the most expensive component in terms of training data and resources. We use a Vision Transformer (ViT), based on [59], extended to operate on three-dimensional inputs. A ViT divides the calorimeter into patches, each of which contains an exclusive set of voxels. The large-scale architecture consists of a series of transformer blocks which perform a residual self-attention operation between embedded patches, followed by a dense network. This part of the neural network can be considered "universal" as it handles a variable number of patches, and it is common to any detector. The remaining elements of the ViT serve to embed the patches and the conditions in a common latent space, and to map the final representation to the predicted velocity field $v_{\Theta_s}(x(t), t, C, u)$. Before entering the universal block, we first create the patches and then embed them in a latent vector with a simple linear projection. Two more embedding networks perform the same operation on the physical conditions (u, C) and time step t. More details on the embedding steps of the input voxels are discussed in the following section. These two latent vectors are added to construct a single conditional vector, then encoded in the transformer blocks through learnable affine transformations. The modulations, self-attention, and the non-linear operations in the universal block closely follow the description presented in [56]. Figure 1 shows a visualization of the main components of the shape network and its universal block.

Patching irregular geometries. Representing a shower as a regular grid introduces inefficiencies when modeling calorimeters with high granularity. The required spatial resolution translates into a large number of voxels, most of which have no energy deposition. A more computationally effective approach optimizes the geometry in the detector such that there are no biases in the downstream analysis while minimizing the number of voxels [35]. Such optimization results in an irregular grid of voxels in the direction of propagation of the shower. The structure of a ViT is not limited to a regular grid. We show how a ViT is extended to irregular geometries as long as a function which transforms voxels into patches can be defined. The CaloChallenge-ds1 [60] and CaloHadronic [61] datasets are examples of geometries with varying number of voxels per layer, or sub-detector. We handle these cases by defining the length of a patch P_{tot} and allowing for grouping adjacent voxels according to the calorimeter grid. If there are N different grids, we define the patch sizes

$$\{(P_{x_i}, P_{y_i}, P_{z_i})\}_{i=0}^{N} , \qquad \text{such that} \qquad P_{x_i} P_{y_i} P_{z_i} = P_{\text{tot}} , \tag{12}$$

for each set i. Here, the coordinates (x, y, z) characterize the geometry and P_k represents the number of voxels selected for the k coordinate. Since transformers are permutation-invariant, the next embedding step consists of applying a positional embedding to the created patches. This makes the transformer aware of the position of the patches, and hence of the spatial location of the calorimeter cells that are part of the patch. Such positional embedding should also respect the varying geometry and the corresponding patching. Therefore, we introduce a three-dimensional sine embedding with learnable

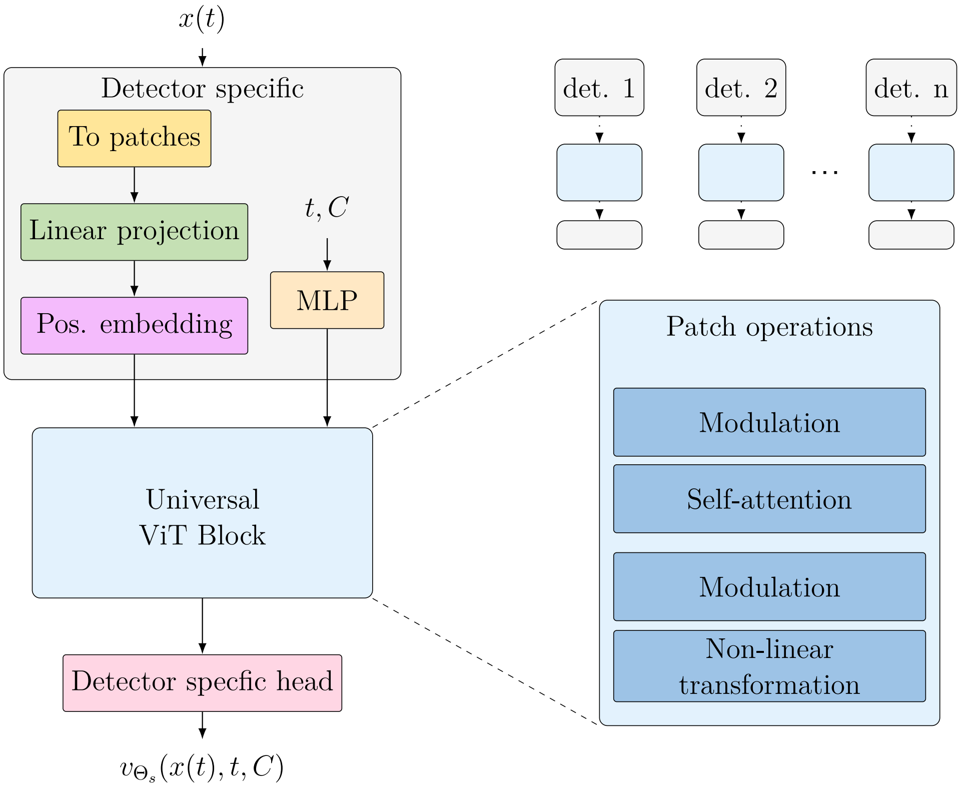


Figure 1. Schematic diagram of the vision transformer [59], which highlights the detector-specific and the universal part of the architecture. The color coded detector-specific steps (see text for more details) indicate the components which may be reinitialized during fine-tuning. The universal ViT block only contains learnable transformations at patch-level objects. These weights, trained on a large corpus of data, can learn general features of calorimeter showers which are used as initializations for other detectors.

frequencies. Let $(X_{n_i}, Y_{n_i}, Z_{n_i})$ be the number of patches along each direction for grid i, to incorporate spatial information into the transformer, we construct a three-dimensional positional encoding that respects the heterogeneous detector layout. For each grid, we define a cumulative coordinate in the depth direction

$$z \in \left\{0, \frac{1}{L}, \dots, \frac{L-1}{L}\right\}, \qquad L = \sum_{i=0}^{N} Z_{n_i} \;, \tag{13}$$

and two local coordinates in the transverse directions

$$x \in \left\{0, \frac{1}{X_{n_i}}, \dots, \frac{X_{n_i}-1}{X_{n_i}}\right\}, \qquad y \in \left\{0, \frac{1}{Y_{n_i}}, \dots, \frac{Y_{n_i}-1}{Y_{n_i}}\right\} \;. \tag{14}$$

Finally, the 3D meshgrid (x, y, z) is multiplied with a set of learnable frequencies ω_d initialized from a Gaussian distribution,

$$\omega_d = 2\pi f_d \;, \quad f_d \sim \mathcal{N}(0, 1) \;, \quad f_d \in \mathbb{R}^{D/6} \;, \tag{15}$$

where D is the latent dimensionality of a single patch. Therefore, the positional embedding vector, added to the inputs, is

$$\mathrm{P} = \left[\sin(xw^T), \cos(xw^T), \sin(yw^T), \cos(yw^T), \sin(zw^T), \cos(zw^T)\right] . \qquad (16)$$

Fine-tuning of pretrained networks. In the fine-tuning setting, our goal is to train a neural network on a large dataset and "finetune", in a second training step, on a smaller dataset sampled from the target distribution. A successful fine-tuning will show better data efficiency, meaning that the neural network achieves better generalization at fixed resources. If the pretraining and target detectors are the same, the fine-tuning step is straightforward: all pretrained weights can be preserved, and only the optimization continues on the target dataset. However, if the detector geometry differs between the pretraining and fine-tuning datasets, several components of the architecture may no longer match in dimensionality. This makes part of the pretrained parameters incompatible and requires careful reinitialization. In practice, we have to address the following:

- Embedding layer: the optimal patch size can be different between datasets and we have to realign the training. We find a simple interpolation between the original dimensionality and the final one to work well in practice;
- Embedding of the conditions: the energy ratios and the global conditions may follow a different distribution depending on the incident particle, the detector specifics, and the incident energy. We devise a preprocessing which keeps the boundary of the distributions unchanged. This choice minimizes the distributional shift in the fine-tuning step;
- Final layer: a change in the dimensionality of the final velocity vector requires a reinitialization of the final "head" layer. This step is similar to foundation models [44] where the final head contains few learnable parameters and spatial information, and can be retrained quickly, while the pretrained transformer backbone provides a strong inductive bias;
- Positional embedding: the learnable position embedding is reinitialized if the total number of patches changes. In particular, we recompute the spatial grid and reinitialize the learnable frequencies.

The specific choices adopted in the various datasets are discussed in the corresponding result sections. In general, we train reinitialized layers with a learning rate five times larger than the base one. We posit that general features, e.g the sparsity, of calorimeter showers are encoded in the large ViT backbone during pretraining. The second fine-tuning step can leverage these general aspects of calorimeter showers for a more efficient training dynamic. Even if embedding and final layers may have to be reinitialized, the majority of the learnable weights (>99%), which are still optimized during fine-tuning, are contained in the universal backbone.

3 Evaluation metrics

High-level features. The simplest measure for the fidelity of the generated calorimeter showers is the definition of high-level observables. Following [21], we calculate a large set of observables that characterize the shape of showers, together with energy-related observables that instead focus on the overall energy distribution across layers. The energy-related observables are:

- the total energy deposited in the calorimeter obtained as the sum of all the energy deposits, E_{tot}, divided by the incident energy of the incoming particle, E_{inc};
- E_i: the total energy deposition in layer i;
- shower profiles in the depth or transverse direction: we calculate the average energy deposition over the dataset at each layer in the z direction,

$$\langle E_i \rangle_x = \frac{\sum_{j=1}^{N} E_{ij}}{N} \, , \tag{17}$$

  and each radius r_i,

$$\langle E(r_i) \rangle_x = \frac{\sum_{j=1}^{N} E(r_i)_j}{N} \, , \tag{18}$$

  where $E(r_i)$ indicates the sum of the energy depositions at radius r_i, the index j runs over the showers and N is the size of the sample.

Shower shape observables are sensitive to the distribution of energy within one layer, hence these are useful to evaluate the large ViT shape network. For example, our set includes:

- $\langle \xi \rangle$: the shower center of energy calculated from the energy deposits x_i along the axis ξ in the physical space of the detector, defined as

$$\langle \xi \rangle = \frac{\sum_i \xi_i \cdot x_i}{\sum_i x_i} \, . \tag{19}$$

- $\sigma_{\langle \xi \rangle}$: the width of the center of energy calculated from the same quantities and defined as

$$\sigma_{\langle \xi \rangle} = \sqrt{\frac{\sum_i \xi_i^2 \cdot x_i}{\sum_i x_i} - \langle \xi \rangle^2} \, . \tag{20}$$

We calculate these observables to define either a set of features sensitive to layer-wise mismodelings or global shapes for the entire geometry. Additionally, we compute the sparsity of the showers defined as the active voxels with energy deposition $x_i > x_{\text{th}}$.

Distance-based metrics. Distance-based metrics like the Kernel-Physics-Distance (KPD) and Fréchet-Physics-Distance (FPD) were introduced in [62] as a contribution to the CaloChallenge [21] and are based on metrics used in computer science, the Fréchet Inception distance (FID) [63] and kernel Inception distance (KID) [64]. The Fréchet distances are the Wasserstein 2 distances of the multivariate Gaussians fitted to features of generated and reference data. For images, the features are taken from the penultimate layer of a pretrained Inception-V3 [65] classifier. In the physics case, they are taken as the high-level features defined earlier. The kernel distances work on the same features (pretrained classifier activations or physical, high-level observables) and determine the distance between the generated and reference set using the kernel-based Maximum-Mean-Discrepancy (MMD). Since it tends to correlate rather strongly with the Fréchet distances [21], we report only the FPD scores in our discussion below.

Our evaluation uses the implementation of FPD as provided by the JetNet package [66], with the same hyperparameter settings as used in the CaloChallenge [21] evaluation.

Neural classifiers. Classifiers are the most powerful evaluation metrics on the market. Given samples from two distributions, a neural classifier approximates the likelihood ratio, hence the optimal statistic for a simple two-hypothesis test, between the two. A simple 1D metric which can be extracted from a classifier is the area under the curve (AUC) score [36]. Alternatively, a full phase space reweighting function can be estimated from the histogram of the learned approximate likelihood ratio [67]. Although a powerful metric, the optimal training of a classifier is hard to achieve, especially in high-dimensional spaces. To mitigate this issue, we train multiple classifiers as done in [21]. We include a "high-level" classifier trained on a set of high-level features, a "low-level" classifier trained on the full set of voxels, and a "ResNet" convolutional architecture, which takes as inputs the same low-level information, but it has a stronger inductive bias towards identifying spatial mismodelings. All classifiers are trained to distinguish between a test Geant4 and a generated sample with the same number of showers. The input features, the classifier hyperparameters, and the best network selection follow the prescription used in the CaloChallenge [21].

Generation time. In a fast detector simulation setting, the optimal working point for a generative surrogate also depends on the generation speed of the neural network. We measure the generation time both on CPU and GPU. On GPU, we assume that it is possible to parallelize the event generation and use a reference batch size of 100. The timing includes overheads from the initialization of the CUDA kernels and the time required to move each batch from GPU back to CPU. For all the tests, we use a single NVIDIA A100 GPU. Even though differentiable emulators greatly benefit from GPU parallelism, we report the CPU generation time as well. The CPU used is an AMD EPYC Zen 3 Milan, from which we allocate a single core and thread.

4 Regular geometries

Our first step is to benchmark the universal backbone architecture on well-known datasets. We define a common architecture and train an energy and shape network for each dataset. Details on the parameters of the neural network are described in Appendix A.

CaloChallenge datasets-2/3. The public datasets [60, 68, 69] used for the Fast Calorimeter Simulation Challenge [21] are the ideal testbeds since they provide the latest comparison to multiple generative networks. Our starting points are the regular geometries developed for dataset-2 and 3. The simulation contains incoming electrons interacting with layers of alternating active silicon (thickness 0.3 mm) and inactive tungsten absorber layers (thickness 1.4 mm) at $\eta = 0$. The energy in the absorber layers is voxelized in a space with 45 layers, with binning in the radial, r, and angular, α, directions. Dataset-2 contains a total of 144 bins per layer, divided into 16×9 angular and radial voxels, while dataset-3 contains a total of 900 voxels per layer arranged into a 50×18 spatial grid. For both datasets, the minimum readout energy, which is also used as a threshold for calculating the sparsity, is $x_{\text{th}} = 15.15$ keV. The initial condition of the shower is characterized solely by its incident energy, which is log-uniformly distributed in $E_{\text{inc}} = 1 \ldots 1000$ GeV.

Starting from the reference CaloChallenge datasets already studied in [56], we provide a summary of the refined performance based on the neural network described in the previous section. A detailed description of the datasets and their evaluation have already been done in [21]. Therefore, we discuss the generation performance only in terms of neural network classifiers, since they provide stronger discrimination performance, FPD metric, and generation time. The high-level features histograms are provided in Appendix B.

In Table 1 (left), we summarize the AUC scores from the three neural network classifiers. As expected, we reproduce similar results to the original CaloDREAM for ds2. For ds3, we improve the performance for all three classifiers, especially for the low-level one. In [56], fully training the network for ds3 exceeded the available computational resources, limiting the size of the neural network and inhibiting complete convergence. We instead fix the architecture of the ViT and adjust the size of one patch to reduce the number of embedded patches in the transformer layers. We create patches with size $(z, \alpha, r) = (3, 10, 3)$ for a total of 450 patches for ds3, which reduces the number of patches by a factor of ~ 3 compared to [56]. Although smaller patches provide better resolution, we find that reducing the training cost while reaching a better convergence can ultimately improve the generation performance. The up-to-date generation performance, evaluated in terms of AUC score, shows that the high-level features are almost indistinguishable from Geant4, also for ds3. Among the classifiers that look at the entire shower, only the more advanced ResNet extracts mismodeled generated features. We remark the improvement over CaloDREAM by including the AUC scores reported in [21]. We observe similar improvements in the FPD metric. In particular, the CaloDREAM++ samples for ds2 are statistically indistinguishable from the Geant4 reference.

		Classifier AUC			FPD×10^3
		High-level	Low-level	ResNet	
Geant4	ds2-e$^-$	0.499(2)	0.500(2)	0.500(4)	10.7(8)
	ds3-e$^-$	0.500(3)	0.498(2)	0.499(2)	8.7(5)
CaloDREAM	ds2-e$^-$	0.521(2)	0.531(3)	0.681(15)	25(1)
	ds3-e$^-$	0.524(4)	0.630(5)	0.802(14)	21(1)
CaloDREAM++	ds2-e$^-$	0.511(1)	0.516(1)	0.683(9)	10.7(4)
	ds3-e$^-$	0.515(1)	0.524(1)	0.799(9)	18.9(4)

Table 1. Summary of evaluation metrics for the baseline networks on regular geometries. The AUC score of neural classifiers, as defined in the text, and the FPD score confirm the higher fidelity of the improved CaloDREAM network. The given classifier uncertainties are the standard deviations of 5 independent classifier trainings, while we report the FPD uncertainty estimate from a single sample.

Gen. time [ms per shower]		
CaloChallenge	ds2-e$^-$	ds3-e$^-$
CPU batch 1 (1 RK4 step)	8.09(3)×10^2	1.39(4)×10^3
GPU batch 100 (1 RK4 step)	11.0(2)	12.5(5)
CPU batch 1 (full gen)	5.43(4)×10^3	1.630(1)×10^4
GPU batch 100 (full gen)	34(1)	96(1)

Table 2. Generation time on the CaloChallenge-ds2/3 datasets on GPU, with batch size 100, and CPU, with batch size 1 and on a single-core and thread machine.

The generation time for CFM networks depends on the number of function evaluations used to numerically solve the integral in Eq. (6). We observe that the fidelity of generated samples plateaus at ∼20 function evaluations. In Table 2, we report the generation time for the full generation and for a single step of the Runge-Kutta 4 (RK4) solver. The CPU time should be considered as a standard reference number for fully sequential generation. Generation on GPUs fully exploits the benefit of modern deep learning, and it shows that generation time can be easily kept below 100 ms. Single-step generation can further reduce the sampling time with an accuracy tradeoff. We leave the study of strategies for the reduction of the function evaluations to the future. Here, we observe that the single-step generation suffers from overheads from initialization of the CUDA kernels, as it does not show the expected 20× gain.

LEMURS dataset. The LEMURS dataset [53] is an extended set built upon the CaloChallenge-ds2. It contains showers produced by a single particle interacting

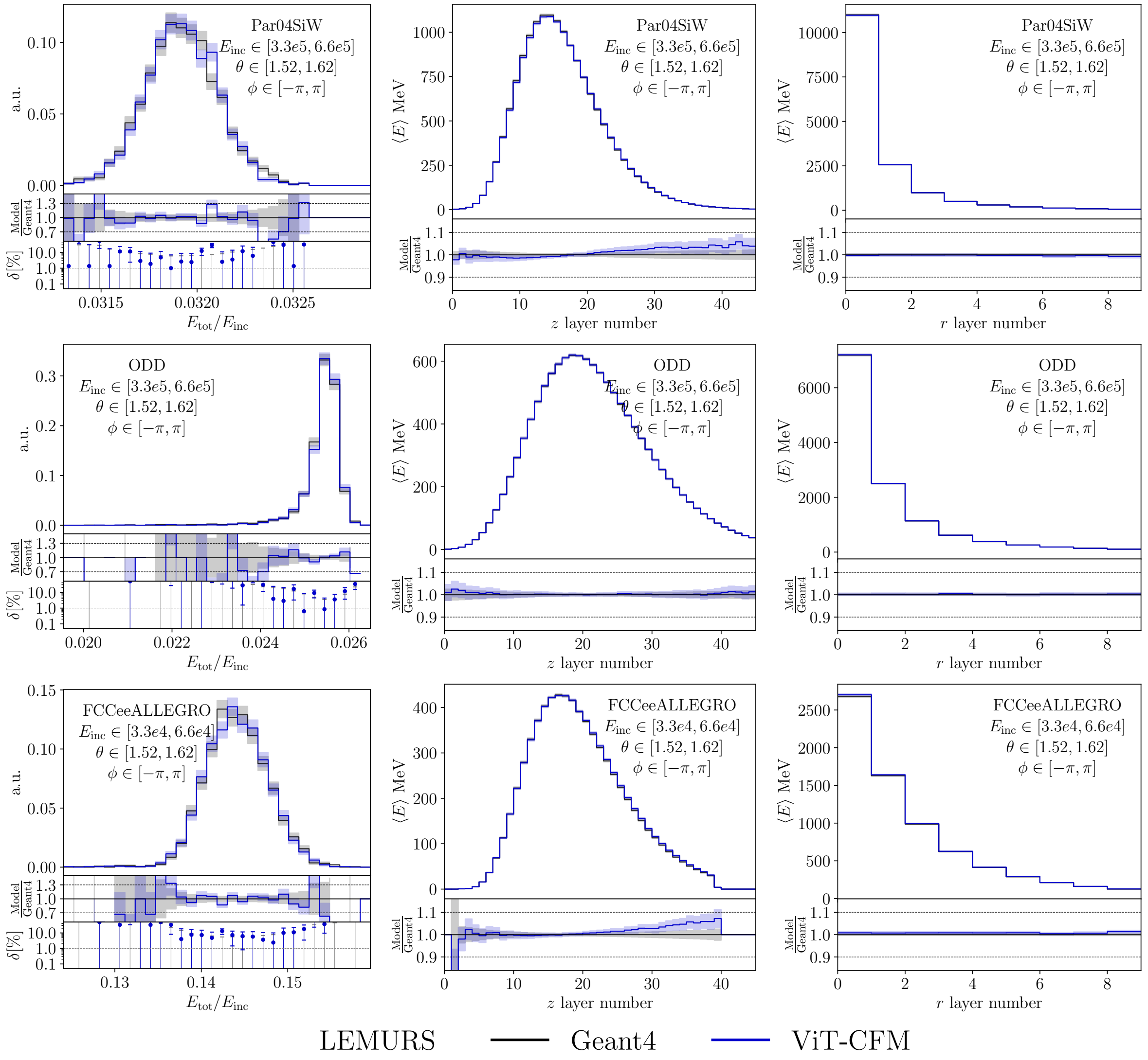


Figure 2. Summary of the sliced evaluation for $E_{\text{inc}} \in [0.33 \cdot E_{\text{i-max}}, 0.66 \cdot E_{\text{i-max}})$ GeV, $\theta \in [1.52, 1.62))$, where $E_{\text{i-max}}$ is 100 GeV for the FCC detectors and 1TeV for the Par04 and ODD detectors. We show the visible energy, the energy profile in the z direction, and the energy profile in the radial direction: (top) Par04SiW, (middle) ODD, and (bottom) FCCeeALLEGRO detectors.

with a voxelized representation of a calorimeter. Similarly to the CaloChallenge, the voxelization consists of a cylinder with segmentation in the depth direction z and in the transversal polar coordinates (r, α). The LEMURS dataset contains a total of 5M showers evenly divided into five different detectors: Par04SiW, Par04SciPb, ODD, FCCeeCLD, and FCCeeALLEGRO. The Par04 geometry is the starting point for the CalChallenge-ds2/3 studies and, in this dataset, is simulated with two possible compositions as active and passive materials: silicon-tungsten (SiW) and scintillators-lead (SciPb). The other three detectors are more realistic and are taken from the Open Data Detector [70] (ODD), and two detector proposals for the Future Circular

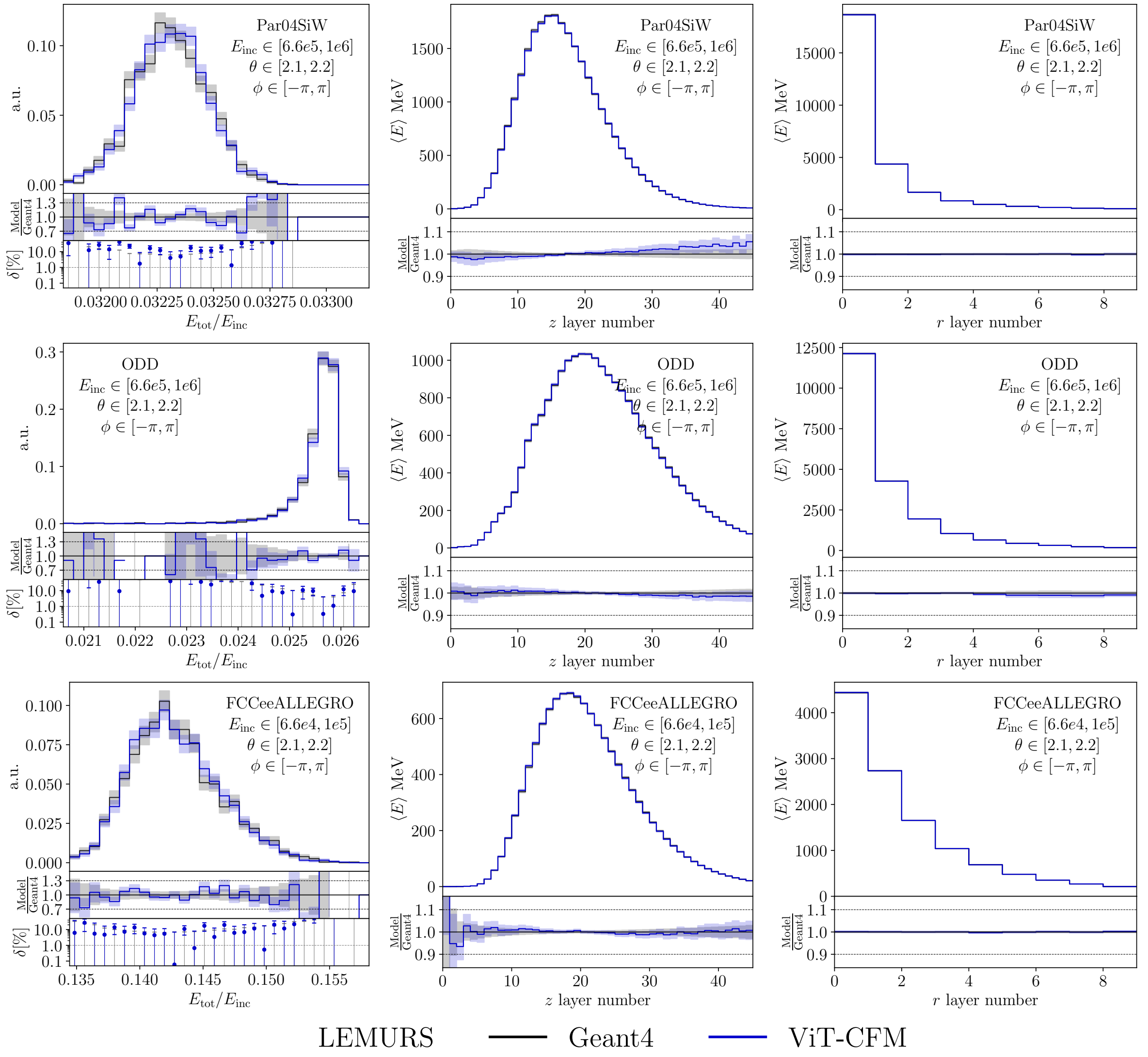


Figure 3. Summary of the sliced evaluation for $E_{\text{inc}} \in [0.66 \cdot E_{\text{i-max}}, E_{\text{i-max}})$ TeV, $\theta \in [2.1, 2.2))$, where $E_{\text{i-max}}$ is 100 GeV for the FCC detectors and 1TeV for the Par04 and ODD detectors. We show the visible energy, the energy profile in the z direction, and the energy profile in the radial direction: (top) Par04SiW, (middle) ODD, and (bottom) FCCeeALLEGRO detectors.

Collider [71], namely the CLIC-like detector [72] (CLD) and the ALLEGRO [73] designs. The full description of the detector geometry can be found in [53]. Showers entering the detectors are simulated at different incident energies and detector locations. The complete information of the incident particle is described by the incident energy E_{inc} and the polar and azimuthal angles (ϕ, θ) in the detector reference frame. The global conditions are independently sampled from

$$E_{\text{inc}} \sim \mathcal{U}(1, 10^3)\,\text{GeV}, \qquad \cos\theta \sim \mathcal{U}(\cos(0.87), \cos(2.27))\,\text{rad}, \\ \text{and} \quad \phi \sim \mathcal{U}(-\pi, \pi)\,\text{rad}\,. \tag{21}$$

The size and breadth of detectors contained in the LEMURS dataset make it the prime candidate for studies on transfer learning and multi-purpose training of generative networks.

For the LEMURS dataset, we train a single vision transformer extended to accommodate multiple conditions as input. We use the kinematic variables of the incoming particle, $E_{\text{inc}}, \theta, \phi$, and the detector-specific label introduced as a one-hot encoded vector. We append the energy variables to the same condition vector, for a total of 53 conditions. The energy network is also extended to depend on the full set of variables defining the incident particle. However, since it is easier to train and much smaller than the ViT, we use five different energy networks, one for each detector. We split the evaluation of the generative network into two parts:

- A sliced evaluation: we compare to high-level features calculated from 2.5k samples generated in a slice in (E_{inc}, θ), while being inclusive in ϕ. We define three linearly spaced bins in E_{inc} with equal size. In θ we instead select two narrower regions, namely $\theta \in \{[1.52, 1.62), [2.1, 2.2)\}$.
- Full evaluation: we train neural classifiers on a large sample that covers the full range of conditions.

The sliced evaluation allows us to evaluate the generation performance in narrower phase space regions. Although the LEMURS dataset provides test samples of 1k showers at fixed conditions, we notice that the training dataset contains many fewer events for some of the working points. This results in large oversampling factors, sometimes up to a factor of 100. While exploring the amplification capability of neural networks is an important question, in this work, we limit the evaluation to tests with statistical powers that do not exceed the number of samples seen during training. Comparing a generated narrow slice to fixed test conditions is also not feasible because of biased selection and the physics effect. Therefore, we prefer to extract test slices from a held-out fraction of the training dataset. Each slice contains approximately 2.5k showers. Slicing comes at its own risks: large slices integrate physics behaviors at different conditions, which might dilute mismodeling in particular narrow regions.

Figure 2 and Figure 3 compare the generative network to Geant4 for two phase space regions: central showers with incident energy in the medium energy bin, and lateral showers from the highest incident energy bin. In this analysis, we highlight the visible energy and the shower profiles for the three most different geometries. Additional high-level features are provided in Appendix B. The energy network, which solely defines the generation quality of the visible energy and the z shower profile, shows good agreement with Geant4 in both phase space regions. In particular, for the FCCeeALLEGRO detector, the energy network correctly models the transition from central showers with a sharp energy cut in the last five layers to lateral showers with a smooth energy dependence. We only note a small tendency to produce higher-energetic showers, which could be related to the small number of samples in the slice. Comparing to [51], this highlights that a factorization into two networks can drastically improve the

	Classifier AUC		FPD	
	High-level	ResNet	Geant4	ViT-CFM
Par04SiW	0.503(1)	0.530(5)	1.0033(2)	1.0036(2)
Par04SciPb	0.503(1)	0.55(1)	1.0033(4)	1.0033(2)
ODD	0.502(1)	0.544(4)	1.0043(2)	1.0055(2)
FCCeeCLD	0.509(2)	0.559(5)	1.0040(2)	1.0045(2)
FCCeeALLEGRO	0.507(2)	0.688(16)	1.0035(2)	1.0045(3)

Table 3. Summary table of neural classifier AUC scores. Each classifier is trained to distinguish the Geant4 ground truth from a single detector sample from a total of 200k showers. We also report the FPD scores obtained from the JetNet library [62, 74].

energy reconstruction of showers while still capturing angular dependencies. We observe a similar accuracy for the radial shower profile, an observable which depends on both the energy and the shape network. For the shower profiles, the reported error bars are the standard deviation of the mean for each bin. Hence, they should not be considered as estimates of statistical deviations.

For the integrated evaluation, the large size of the sample allows us to train large neural network classifiers for a holistic evaluation. Table 3 shows the AUC score of the high-level and ResNet classifiers. We exclude the low-level classifiers since they take as inputs the same information used for the ResNet classifier, but they show lower sensitivity to mismodeled showers, as already seen in the CaloChallenge studies. The AUC and FPD scores both show good agreement with Geant4 across all the detectors. The only exception is the ResNet classifier for the FCCeeALLEGRO, which shows a larger AUC score. This is indeed expected since this is the most complex detector with strong dependencies on the position of the showers, as it was already observed with the sliced evaluation. While the FPD metric scores all networks as Geant4-like, the ResNet is able to identify differences between the generated and reference samples.

The architecture and the training hyperparameters are adopted from the CaloChallenge evaluation. We include histograms of additional high-level features for both evaluations in Appendix B.

5 Irregular geometries

Next, we show the generation performance on irregular geometries. In the following, we only require a fixed total patch size and a segmentation of the voxelized space divisible by that number.

CaloChallenge-ds1. The two remaining datasets of the CaloChallenge are of lower dimensionality but geometrically irregular. They provide calorimeter showers for central photons and charged pions originally used in AtlFast3 [75]. The detector geometries for photon and pion showers have five and seven layers, respectively, with the number of radial and angular bins

$$\begin{aligned} \text{photons} \qquad & 8 \times 1,\ 16 \times 10,\ 19 \times 10,\ 5 \times 1,\ 5 \times 1 \\ \text{pions} \qquad & 8 \times 1,\ 10 \times 10,\ 10 \times 10,\ 5 \times 1,\ 15 \times 10,\ 16 \times 10,\ 10 \times 1\ , \end{aligned} \tag{22}$$

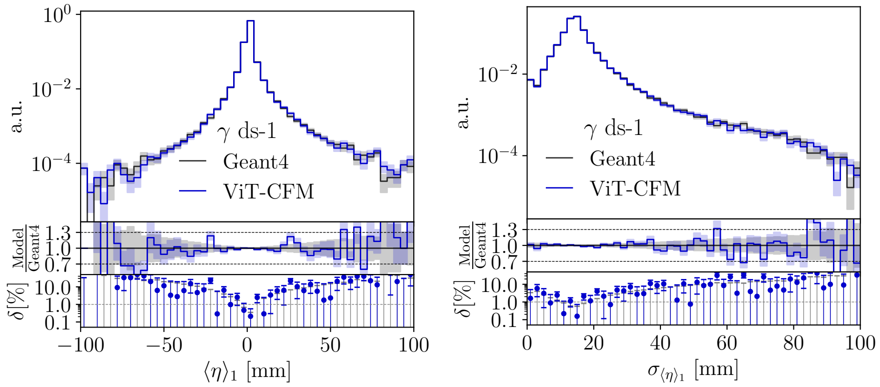
totaling 368 voxels for photons and 533 voxels for pions. The incident energies of both photons and pions are 15 discrete values with increasing power of two, namely $E_{\text{inc}} = 256$ MeV ... 4.2 TeV. The energy E_{inc} refers to the momentum of the particles; this has implications for charged pions at low energy, which have non-negligible mass. Due to the Geant4 generation time constraints, the sample sizes for larger energies are smaller. Details are given in Table 4.

The CaloChallenge-ds1s are a special example of irregular geometries, since there are layers with a single angular bin which cannot be readily patchified. In this case, we add a minimal number of bins to reach the required patch size dimension. In both cases, we use a patch size of five and therefore add four additional bins per radial voxel in the layers without angular sectioning. These additional bins do not carry energy and are part of a single patch in the neural network. Therefore, they do not introduce biases in the encoding and the overall generation process. After generation, they are collapsed into a single bin with energy equal to the maximum generated energy deposition. This approach mildly increases the number of voxels, but we expect it not to occur for high-granular geometries where a common divisor is easier to find. We train a vision transformer on the ds1-γ and ds1-π^+ datasets. The increase in the total number of voxels is of a factor ~ 1.19 for ds1-γ and ~ 1.17 for ds1-π^+.

Figure 4 and Figure 5 show a summary evaluation of the generation performance. We highlight the center of energy and the width of the center of energy for the first segmented layer. For both incident particles, the neural network reproduces the Geant4 response up to statistical uncertainties. The AUC scores of the high- and low-level classifiers provide a holistic view of the entire phase space. For ds1-γ, both classifiers confirm that the network emulates Geant4 almost perfectly. For ds1-π^+, the high-level features show a similar level of agreement, but we observe a higher level of mismodelling from the low-level classifier. While hadronic showers are more complex to model, this discrepancy arises from the usage of a different version of Geant4 in the generation of the test sample [21]. The CaloChallenge found a low (high)-level AUC of 0.609(4) (0.558(2)) when comparing Geant4 train and test sets [21]. Nonetheless, these results

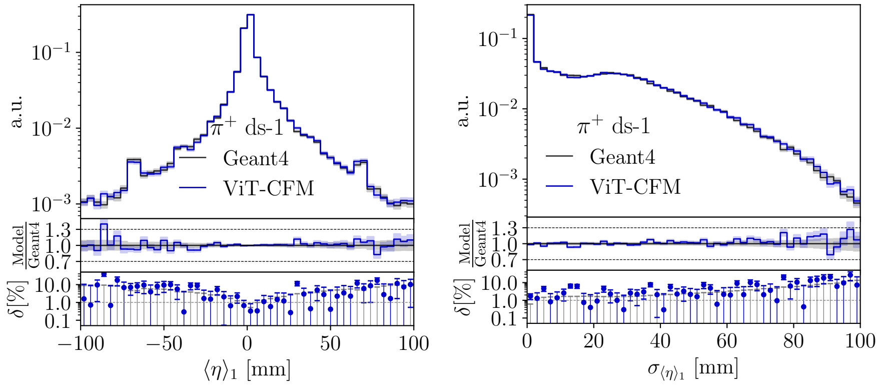
E_{inc}	256 MeV ... 131 GeV	262 GeV	0.524 TeV	1.04 TeV	2.1 TeV	4.2 TeV
photons	10000 per energy	10000	5000	3000	2000	1000
pions	10000 per energy	9800	5000	3000	2000	1000

Table 4. Sample sizes for different incident energies in dataset 1.



Classifier	ds1-γ
Low-level	0.518(1)
High-level	0.504(1)
Gen. time [ms per shower]	
CPU batch 1	3.8×10^3
GPU batch 100	24.36(5)

Figure 4. Summary of the evaluation on the CaloChallenge-ds1-γ dataset. We show the center of energy and the shower width in layer-1, the AUC scores of a low- and high-level neural classifier, and the generation time on CPU, with batch size 1, and GPU, with batch size 100.



Classifier	ds1-π^+
Low-level	0.633(3)
High-level	0.520(1)
Gen. time [ms per shower]	
CPU batch 1	5.7×10^3
GPU batch 100	30.6(3)

Figure 5. Summary of the evaluation on the CaloChallenge-ds1-π^+ dataset. We show the center of energy and the shower width in layer-1, the AUC scores of a low- and high-level neural classifier, and the generation time on CPU, with batch size 1, and GPU, with batch size 100.

improve over all the other submissions to the CaloChallenge.

The lower dimensionality of the dataset implies that a smaller number of patches is processed by the ViT. We observe this effect in the generation time; even though the neural network has the same number of parameters, the generation time for the full generation of a shower for ds1 is shorter than that of the regular ds2 and ds3.

CaloHadronic dataset. Originally generated for [61], this dataset contains showers produced from an incident π^+. Unlike the cylindrical geometry of the other datasets, the detector is represented in cartesian coordinates, and it corresponds to the high-granular calorimeter proposed for the International Linear Collider [76]. The incoming particle is orthogonal to the calorimeter, and it carries incident energy sampled from

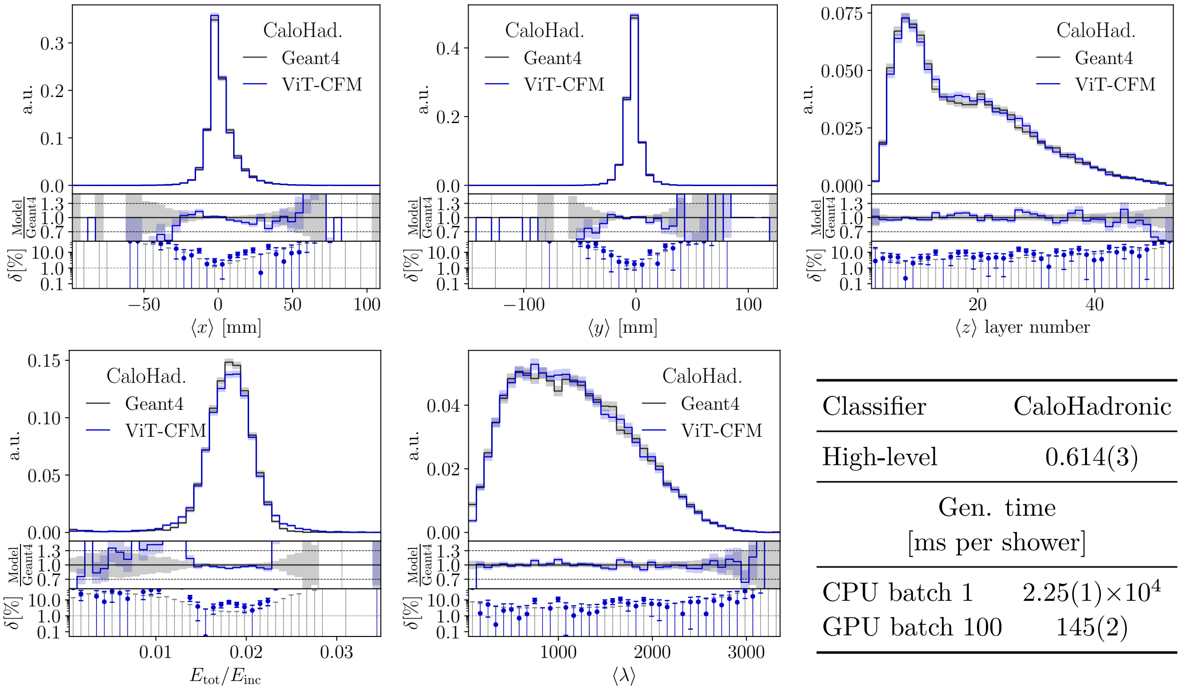
$$E_{\text{inc}} \sim \mathcal{U}(10, 90)\,\text{GeV}\;. \tag{23}$$

The main challenges coming from this dataset are the extremely high granularity, the separation between the electromagnetic and hadronic calorimeters, and the intrinsic complex nature of hadronic showers. Our aim is to showcase a more realistic example where different granularities can arise and show how a ViT can deal with them. Therefore, we only use 100k showers to avoid the computationally expensive training on the full dataset. We further simplify the generation task in two ways. First, in [61] the original calorimeter is up-sampled by a factor of three in each of the x-y axes, thus increasing the generation space by a factor of nine. We avoid the complication of blindly increasing the number of voxels and use the original cell size for the hadronic calorimeter (HCal). Therefore, the HCal contains 43200 voxels organized as $(x, y, z) = (30, 30, 48)$, where each cell has a lateral width of 30 mm. Second, the electromagnetic (ECal) section is much more granular and sparse. Instead of using the original ECal voxelization $(x, y, z) = (180, 180, 30)$, we down-sample with a sum-pooling operation down to 2250 voxels organized as $(x, y, z) = (15, 15, 10)$, where each cell has size 5.1×12 mm. Here we do not perform the up-sampling operation which can be learned, together with the down-sampling step, with an autoencoder-like structure, especially if the showers are incredibly sparse [56, 77, 78]. We create this grid representation for 100k showers. We use 80k showers for training and validation, and the remaining 20k for testing.

We set the total patch size to $P_{\text{tot}} = 75$ with segmentation $P_{\text{ecal}} = (5, 5, 3)$ and $P_{\text{hcal}} = (3, 5, 5)$ for the electromagnetic and hadronic calorimeter, respectively. The flexibility in the patch selection allows for the straightforward transformation to patch space embeddings without adding unphysical layers which would be needed to divide the detector into patches with a single segmentation.

To evaluate the generative network, we follow a methodology similar to [61]. We first calculate global high-level features for the entire detector. The top row of Fig. 6 shows the center of energy in the x and y directions in units of the real detector size, and the center of energy in the z direction as a function of the layer number. The bottom row shows the ratio $E_{\text{tot}}/E_{\text{inc}}$ and the average number of hits $\langle\lambda\rangle$. Finally, we show, in table, the AUC score of a neural classifier trained on the five observables together with the layer energies, and the generation time on CPU and GPU. The marginal distributions, as well as the neural classifier, confirm that the main features of the hadronic showers are well-captured by the generative network, with larger deviations for low-energy and low-occupancy showers, otherwise within statistical uncertainties. For all the results, the Geant4 reference corresponds to a test sample containing 20k showers, which is compared to a generated sample with the same statistic, and we apply a low-energy threshold of $x_{\text{th}} = 1$ keV. Given the small size of the test sample, we refrain from training a classifier on the low-level energy deposits. The details of the classifier implementation and training are reported in Appendix A.

The generation time on both CPU and GPU increases due to the larger number of patches compared to the CaloChallenge-ds3. Nevertheless, the GPU generation time is still at $\mathcal{O}(100)$ ms.



Classifier	CaloHadronic
High-level	0.614(3)
Gen. time [ms per shower]	
CPU batch 1	$2.25(1)\times10^4$
GPU batch 100	145(2)

Figure 6. Summary of the evaluation on the CaloHadronic dataset. We show the center of energy in the three coordinates x,y, and z for the entire calorimeter, the total visible energy E_{tot}, and the average occupancy $\langle\lambda\rangle$ in the detector. In the table, we report the AUC score of a neural classifier and the generation time on CPU and GPU.

6 Transfer learning

We present two clear cases where fine-tuning can be used to reduce the cost of training ViTs for fast detector simulation. First, we use the large LEMURS dataset as a pretraining objective during which the network learns multi-detector responses and angular dependencies. The target fine-tuning is the CaloChallenge-ds2 dataset. For the fine-tuning phase, we set the angular conditions to $\theta = 1.57$ and $\phi = 0.0$. Since the detector matches the Par04 setting, we set the one-hot detector encoding equal to the Par04SiW detector. The one-hot encoding is an effective entry point for the introduction of prior knowledge of the detector configuration. For instance, a third detector with the Par04 geometry can be encoded as a weighted combination of the two materials already seen during training. In this case example, the fine-tuning step specializes the network to a single angular condition and adjusts the response from a photon initiated shower to an electron one. The second study is a superresolution from the CaloChallenge-ds2, or LEMURS, voxelization to detectors with higher resolutions. For this study, we reinitialize all the components that encode and decode the shower information in the ViT, as described in Section 2. This choice corresponds to the "Full fine-tuned" strategy adopted for point clouds in [52]. Fine-tuning the entire backbone is strictly more expressive than adjusting a subset of weights. Since our networks require a manageable amount of resources, we do not explore other more parameter-efficient training strategies.

The principal figures of merit for evaluating the training efficiency of generative networks are the amount of data and the computational resources used during training. We explore two scenarios: one where we fix the amount of available data and we vary the number of training iterations, and a second where we scan over the size of the training dataset.

Training iterations The baseline training of Section 4 converges after approximately 800k iterations. Here, we aim to find the minimum number of training iterations needed to obtain the same performance with the fine-tuned network. Figure 7 summarizes the efficiency gain from utilizing a pretrained network. On the left, we compare a ViT trained from scratch against a network trained on LEMURS and then fine-tuned on the CaloChallenge-ds2. We use the ResNet classifier AUC score as our evaluation metric, as it demonstrated the strongest discriminative performance in our baseline studies. On the right, we repeat the same study for the superresolution study. In both cases, the fine-tuned network converges more rapidly, showing an AUC score significantly smaller than 1.0 already after 100k iterations. The performance of the network trained from scratch is reached with roughly 400k training iterations, therefore converging in half the training time. Faster convergence is particularly beneficial if multiple neural networks have to be trained because the large pretraining phase happens only once. As an example, the calorimetric module of AtlFast3 [75] contains 300 neural networks trained with different particle types and at different pseudo-rapidity angles. On our hardware, the pretraining phase becomes subdominant already after six trainings. We remark that the selection of the metric can affect the evaluation of the fine-tuning step and the following efficiency gain claims. In Fig. 8, we show the same scan over the training iterations but we use the FPD score as the evaluation metric. Notice that generation quality converges to the optimal value much more quickly than the ResNet classifier. This highlights the importance of selecting the most discriminative metric if a holistic evaluation of the network is of interest.

Dataset size Generating Geant4 data is the most computationally intensive step of the simulation chain, and reducing the training data necessary to reach the target accuracy and generalizability is of large interest for a fast simulation surrogate. We explore the effect of fine-tuning on three datasets: first on the CaloChallenge-ds2/3 as in the previous section, then on the CaloHadronic dataset. We perform a full performance scan over training dataset sizes only for the CaloChallenge-ds2, since the smaller dimensionality allows for faster training. Given our available computational resources, it was unfeasible to carry out similar studies for the higher resolution datasets. Therefore, we look for performance improvements using the full training dataset. We use as a performance metric the one that demonstrated the best discriminative power from the previous tests; for the CaloChallenge datasets, we report the ResNet classifier

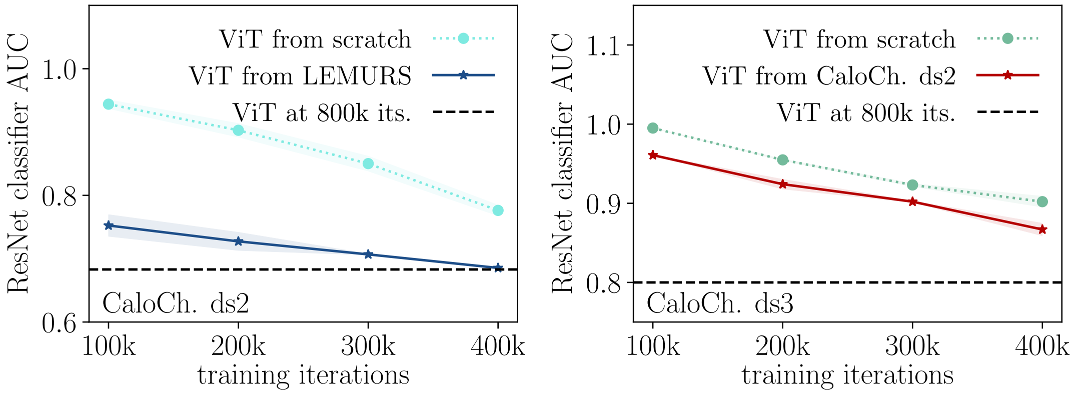


Figure 7. Efficiency gain of fine-tuned networks for varying training iterations. The ResNet classifier AUC score of fine-tuned networks is significantly lower than that of the networks trained from scratch. (left) Pretraining on the LEMURS dataset, fine-tuned on the CaloChallenge-ds2. (right) Pretraining on the CaloChallenge-ds2, fine-tuned on the CaloChallenge-ds3.

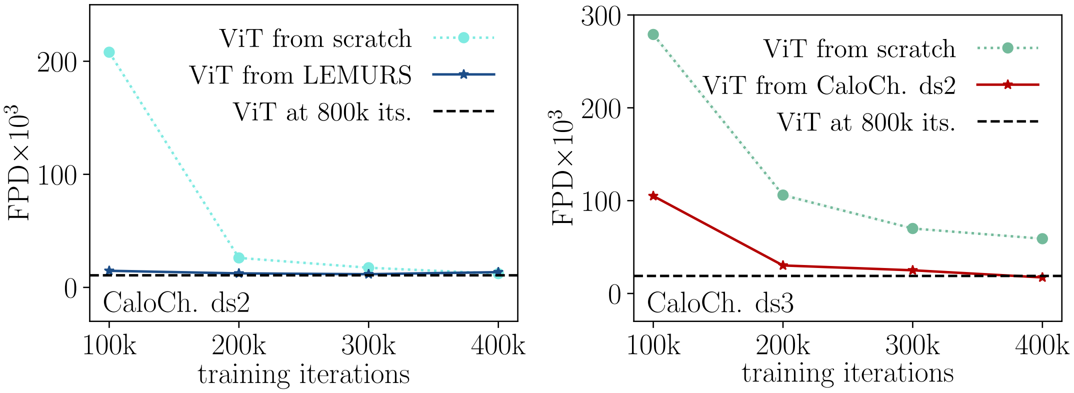
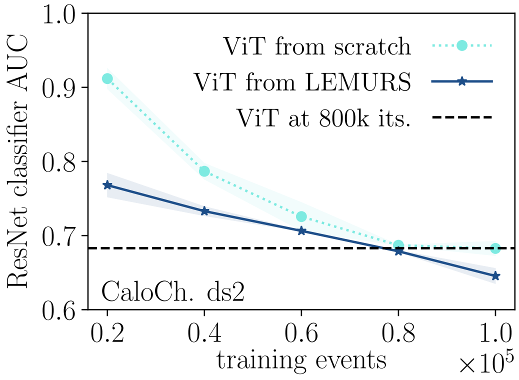


Figure 8. Efficiency gain of fine-tuned networks for varying training iterations. The FPD score of fine-tuned networks converges to the best value observed from a complete training in significantly less iterations than the networks trained from scratch. Notice that the the ResNet classifier in Fig. 7 indicates smaller gains compared to the FPD. (left) Pretraining on the LEMURS dataset, fine-tuned on the CaloChallenge-ds2. (right) Pretraining on the CaloChallenge-ds2, fine-tuned on the CaloChallenge-ds3.

AUC score, while for CaloHadronic, we use the neural classifier trained on the five high-level observables. Figure 9 (left) shows the ResNet AUC score for the pretrained ViT on LEMURS and fine-tuned to the CaloChallenge-ds2. To train the classifier, we always generate 100k showers, therefore, we are also testing the generalizability of the generative architecture at smaller training dataset fractions. Our results show that the fine-tuned network provides better performance, hence generalizability, for each training dataset size. Fine-tuned networks converge much more rapidly; we avoid overfitting by increasing the weight decay parameter as the size of the training dataset diminishes. In



	from scratch	fine-tuned
	ResNet cls. AUC	
CaloCh. ds2	0.683(9)	0.645(9)
CaloCh. ds3	0.799(9)	0.789(9)
	High-level cls. AUC	
CaloHadronic	0.614(3)	0.600(4)

Figure 9. (left) Efficiency gain of fine-tuned networks for varying dataset sizes. fine-tuned networks better generalize as demonstrated by a ResNet classifier always trained with a total of 200k showers. Pretraining on the LEMURS dataset, fine-tuning on the CaloChallenge-ds2. (right) Summary table of classifier AUC scores after fine-tuning on the entire training dataset for the corresponding detectors.

particular, at 100k training showers, the fine-tuned ViT is significantly better than the network trained from scratch, setting a new state-of-the-art for the CaloChallenge. The high-granular ds3 shows a smaller gain. Even though the ViT converges more rapidly, as shown in Fig. 7, the performance plateaus at an AUC of 0.80. This could be an indication that the neural network expressivity is the limiting factor. We observe a similar behavior for the CaloHadronic dataset, where the high-level classifier shows a better agreement with Geant4 for the fine-tuned ViT.

7 Conclusions

Machine learning emulators are effective solutions to accelerate slow simulators. Fast calorimeter simulation is the most striking and compelling example at HEP collider experiments, such as those at the LHC, ILC, or the FCC. Once ensured that the generation speed of generative networks is sufficient, maximizing the accuracy of the surrogate calorimeter showers becomes the real challenge. Especially for high-granular detectors, the computational cost of training deep learning networks becomes prohibitively expensive.

We have proposed a vision transformer architecture that can be readily applied to any voxelized detector geometry. From the performant CaloDREAM architecture, we have defined an efficient and flexible patching for arbitrary detector layouts, including optimized and full detector voxelizations. This enables a reduction in both computational and data requirements without sacrificing expressiveness. Our results show excellent agreement with Geant4 on the studied datasets; multiple evaluation metrics indicate that the generated samples are indistinguishable from Geant4, while also improving over previous benchmarks.

We have further introduced a pretraining strategy for ViTs and demonstrated that pretraining, followed by a targeted fine-tuning on the CaloChallenge ds2/3 significantly reduces the resources needed to match the performance of the network trained from scratch on the most sensitive evaluation metric. Additionally, for the large LEMURS pretraining, we have shown better data efficiency and better generalization of the fine-tuned network. This is the first application of patch-based fast calorimeter simulation to an entire detector with both an electromagnetic and a hadronic calorimeter. We believe our studies will open the way towards the full implementation and deployment of transformer-based emulators in the community.

Code and data availability. Together with this document, we include a public release of the code at https://github.com/luigifvr/vit4hep. We also publish the complete set of high-level features and samples used during evaluation at 10.5281/zenodo.18071948.

Acknowledgments

We would like to thank Sofia Palacios Schweitzer and Ayodele Ore for comments on the manuscript and, together with Jonas Spinner, for many useful discussions. We would also like to thank Martina Mozzanica for support with the CaloHadronic dataset, and Anna Zaborowska and Peter Mckeown for helpful discussions on the LEMURS dataset.

L.F. is supported by the Fonds de la Recherche Scientifique - FNRS under Grant No. 4.4503.16.

Computational resources have been provided by the supercomputing facilities of the Université catholique de Louvain (CISM/UCL) and the Consortium des Équipements de Calcul Intensif en Fédération Wallonie Bruxelles (CÉCI) funded by the Fond de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under convention 2.5020.11 and by the Walloon Region. The present research benefited from computational resources made available on Lucia, the Tier-1 supercomputer of the Walloon Region, infrastructure funded by the Walloon Region under the grant agreement n°1910247.

Appendix A Training hyperparameters

Parameter	Energy network
Iterations	250k
LR sched.	cosine
Max LR	10^{-4}
Batch size	256
ODE solver	Runge-Kutta 4 (20 steps)
Dim embedding	64
Intermediate dim	1024
Num heads	4
Num layers	4

Table A1. Training and network parameters for the energy network of Section 2.

Parameter	Universal ViT block
Iterations	1.2M
Batch size	64
LR sched.	CosineAnnealingLR
Initial LR	10^{-4}
ODE solver	Runge-Kutta 4 (20 steps)
Embedding dimension	480
Attention heads	6
MLP hidden dimension	1920
Blocks	6

Table A2. Training and network parameters for the universal ViT block presented in Section 2.

Dataset	Patch sizes
	(z, r, α)
CaloChallenge-ds1γ	(1, 1, 5)
CaloChallenge-ds1π^+	(1, 1, 5)
CaloChallenge-ds2e^-	(3, 16, 1)
CaloChallenge-ds3e^-	(3, 10, 3)
LEMURS	(3, 16, 1)
CaloHadronic	(z, x, y)
ECal	(5, 5, 3)
HCal	(3, 5, 5)

Table A3. Patch sizes used for the CaloChallenge, LEMURS, and CaloHadronic datasets.

Parameter	Value
high-level and low-level	
Optimizer	Adam
Epochs	100
Number of layers	3
Hidden nodes	2048
ResNet	
Optimizer	AdamW
Epochs	50
Number of layers	18
Common training parameters	
Learning rate	$2 \cdot 10^{-4}$
Batch size	1000
Training samples	60k
Validation samples	20k
Testing samples	20k
Activation function	leaky ReLU

Table A4. Parameters for the high-level, low-level, and ResNet classifiers network.

Appendix B Additional LEMURS high-level features

We provide, in Figs. B2, B3, B4, B5, and B6, histograms of the high-level features for the layer with the largest energy deposition. These features, together with all the other layers, have been used to train the high-level classifier for the full LEMURS evaluation. The held-out dataset, used for this evaluation, corresponds to the last file of the LEMURS† training dataset. The FCCeeCLD evaluation is the only exception: we noticed that the distribution of the incident energies for the files with less than 100k events does not agree with a uniform distribution. We show the discrepancy between in Fig. B1 and, to avoid biased comparison, we use the first file as the Geant4 reference.

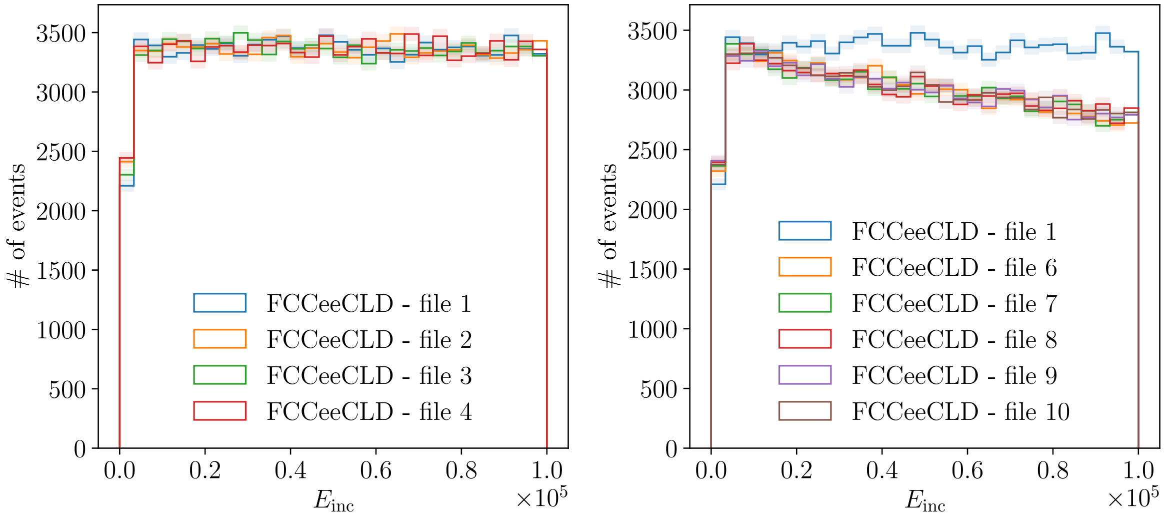


Figure B1. Distribution of the incident energies for the FCCeeCLD detector. The training files 1-4 (left) have the expected uniform distribution while the files 6-10 (right) have a smoothly falling behavior towards larger energies.

†Our studies utilize the 1.0.0 version of the LEMURS dataset.

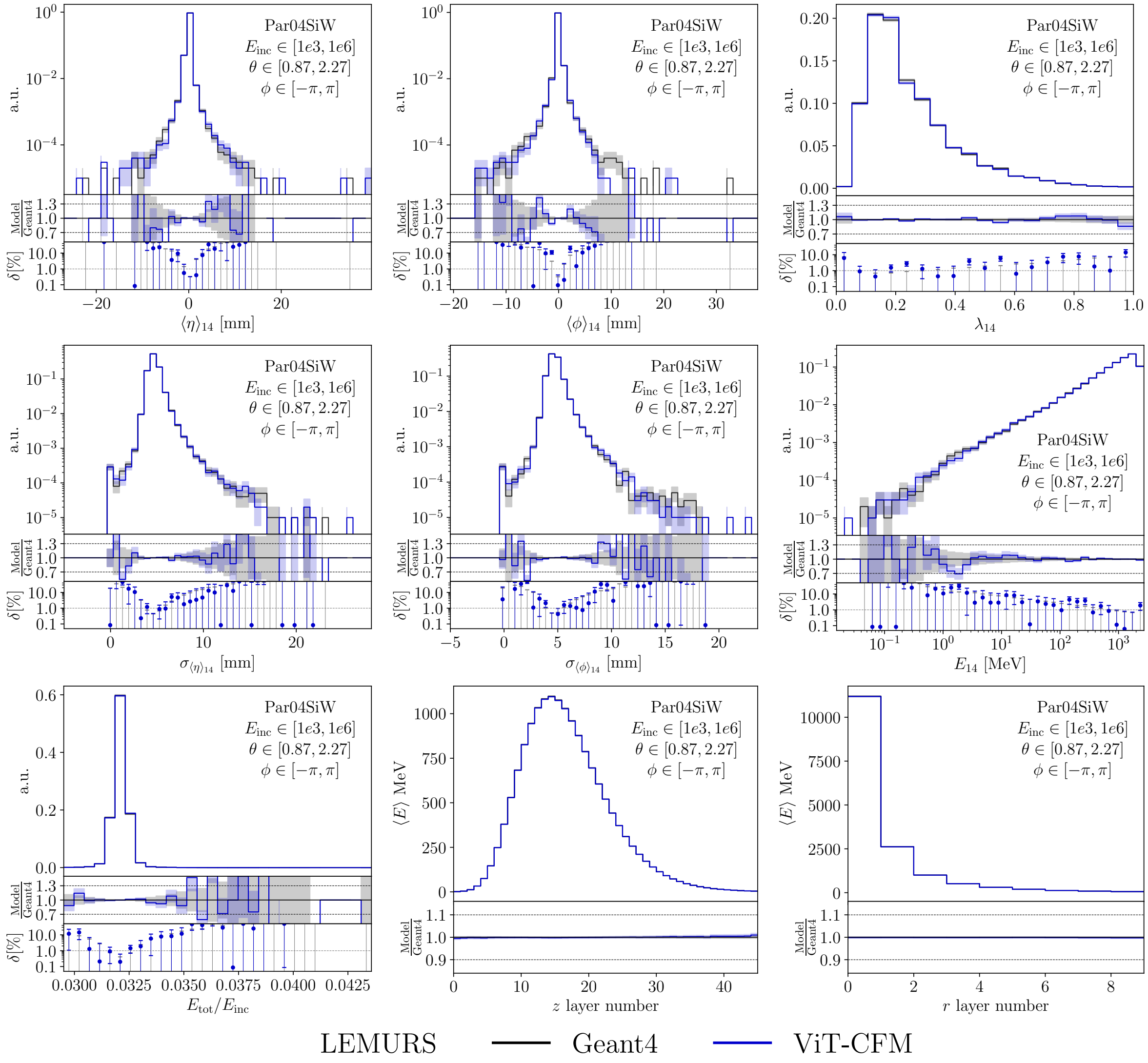


Figure B2. Summary of high-level observables for the Par04SiW detector. From left to right, top to bottom: center of energy in the η and ϕ directions, sparsity, width of the center of energy in the η and ϕ directions, layer energy depositions, energy ratio $E_{\text{inc}}/E_{\text{tot}}$, energy profiles in the z and r directions.

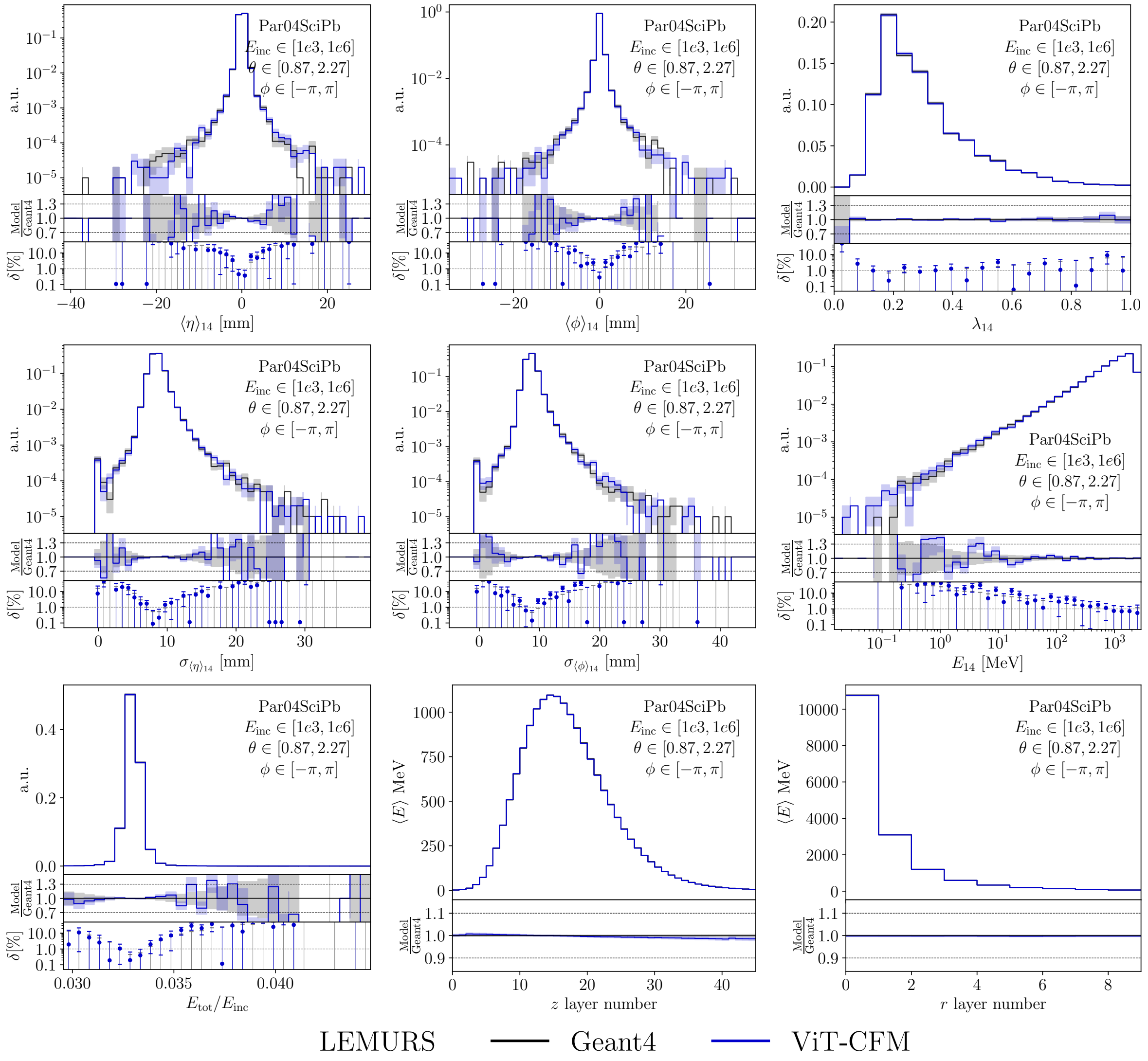


Figure B3. Summary of high-level observables for the Par04SciPb detector. From left to right, top to bottom: center of energy in the η and ϕ directions, sparsity, width of the center of energy in the η and ϕ directions, layer energy depositions, energy ratio $E_{\rm inc}/E_{\rm tot}$, energy profiles in the z and r directions.

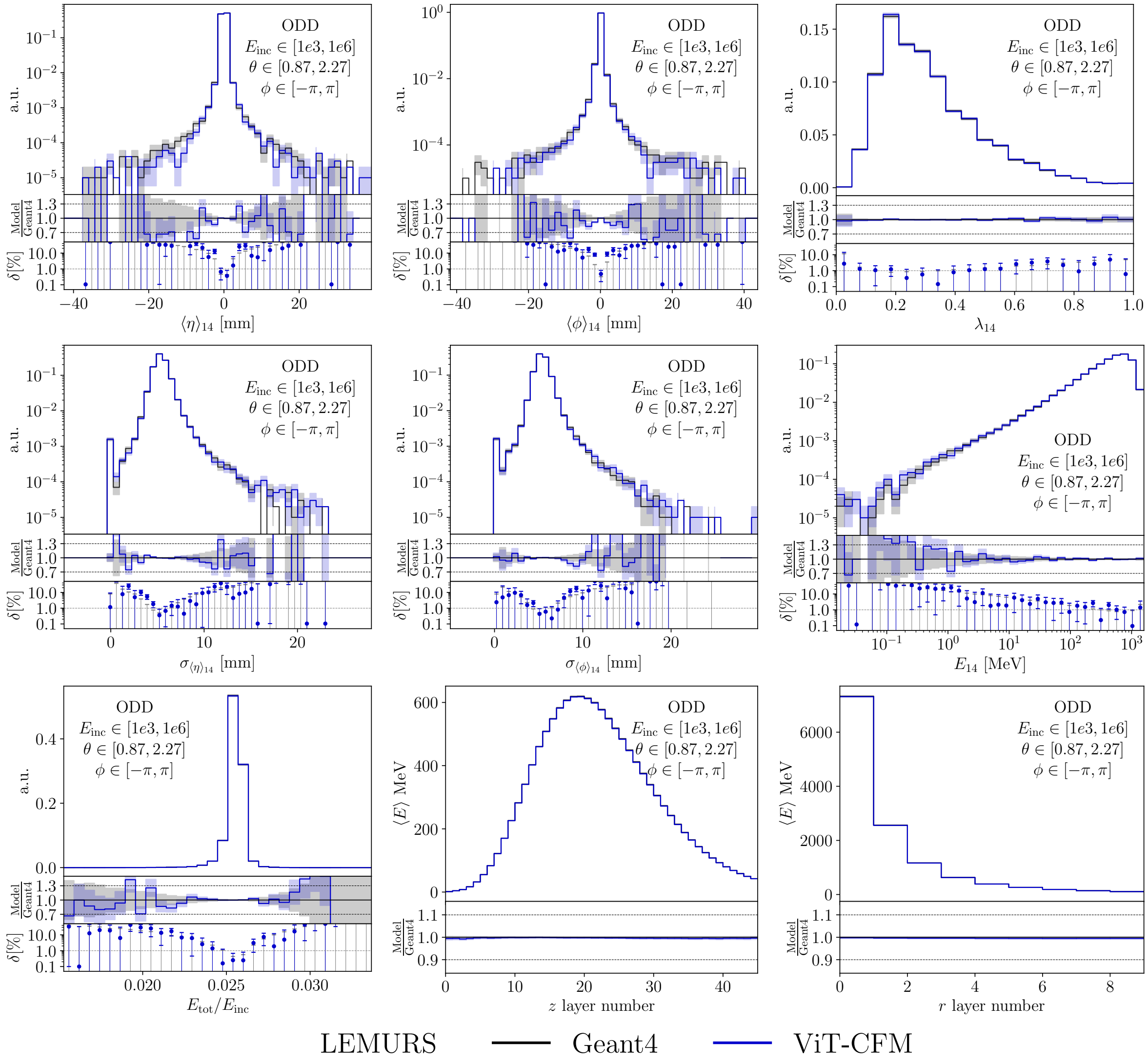


Figure B4. Summary of high-level observables for the ODD detector. From left to right, top to bottom: center of energy in the η and ϕ directions, sparsity, width of the center of energy in the η and ϕ directions, layer energy depositions, energy ratio $E_{\text{inc}}/E_{\text{tot}}$, energy profiles in the z and r directions.

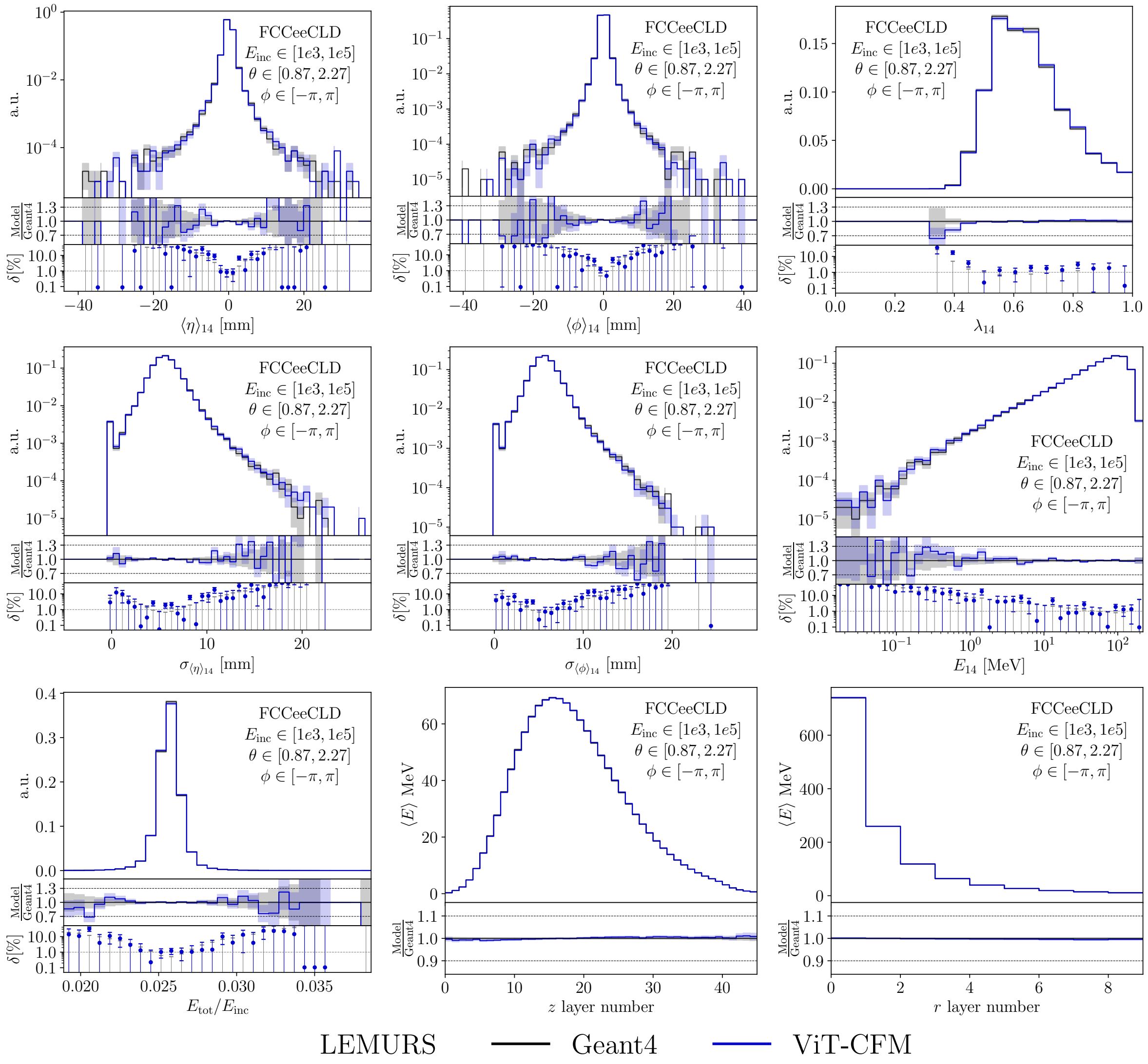


Figure B5. Summary of high-level observables for the FCCeeCLD detector. From left to right, top to bottom: center of energy in the η and ϕ directions, sparsity, width of the center of energy in the η and ϕ directions, layer energy depositions, energy ratio $E_{\rm inc}/E_{\rm tot}$, energy profiles in the z and r directions.

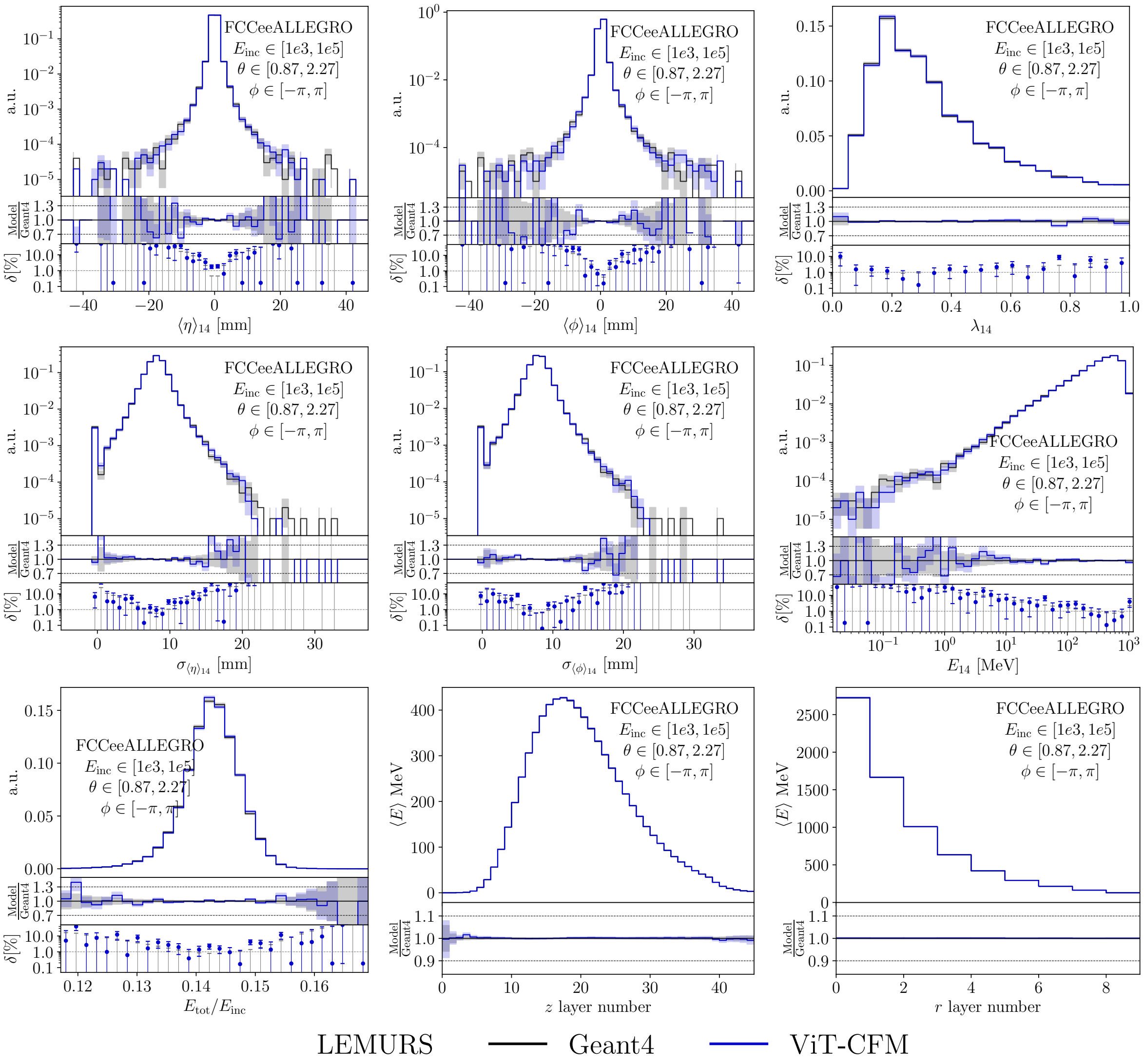


Figure B6. Summary of high-level observables for the FCCeeALLEGRO detector. From left to right, top to bottom: center of energy in the η and ϕ directions, sparsity, width of the center of energy in the η and ϕ directions, layer energy depositions, energy ratio $E_{\text{inc}}/E_{\text{tot}}$, energy profiles in the z and r directions.

Bibliography

[1] Albrecht J *et al.* 2025 *J. Phys. G* **52** 030501 (*Preprint* 2408.03881)

[2] Harris R CERN hits one exabyte of stored experimental data from the LHC URL https://home.cern/news/news/computing/cern-hits-one-exabyte-stored-experimental-data-lhc

[3] Badger S *et al.* 2023 *SciPost Phys.* **14** 079 (*Preprint* 2203.07460)

[4] Shanahan P *et al.* 2022 (*Preprint* 2209.07559)

[5] Cranmer K, Brehmer J and Louppe G 2020 *Proc. Nat. Acad. Sci.* **117** 30055–30062 (*Preprint* 1911.01429)

[6] Badger S and Bullock J 2020 *JHEP* **06** 114 (*Preprint* 2002.07516)

[7] Bahl H, Elmer N, Favaro L, Haußmann M, Plehn T and Winterhalder R 2025 *SciPost Phys. Core* **8** 073 (*Preprint* 2412.12069)

[8] Beccatini L, Maltoni F, Mattelaer O and Winterhalder R 2025 (*Preprint* 2512.11036)

[9] Gao C, Höche S, Isaacson J, Krause C and Schulz H 2020 *Phys. Rev. D* **101** 076002 (*Preprint* 2001.10028)

[10] Bothmann E, Janßen T, Knobbe M, Schmale T and Schumann S 2020 *SciPost Phys.* **8** 069 (*Preprint* 2001.05478)

[11] Heimel T, Winterhalder R, Butter A, Isaacson J, Krause C, Maltoni F, Mattelaer O and Plehn T 2023 *SciPost Phys.* **15** 141 (*Preprint* 2212.06172)

[12] Heimel T, Huetsch N, Maltoni F, Mattelaer O, Plehn T and Winterhalder R 2024 *SciPost Phys.* **17** 023 (*Preprint* 2311.01548)

[13] Heimel T, Mattelaer O, Plehn T and Winterhalder R 2025 *SciPost Phys.* **18** 017 (*Preprint* 2408.01486)

[14] Di Sipio R, Faucci Giannelli M, Ketabchi Haghighat S and Palazzo S 2019 *JHEP* **08** 110 (*Preprint* 1903.02433)

[15] Verheyen R 2022 *SciPost Phys.* **13** 047 (*Preprint* 2205.01697)

[16] Butter A, Plehn T and Winterhalder R 2019 *SciPost Phys.* **7** 075 (*Preprint* 1907.03764)

[17] Butter A, Heimel T, Hummerich S, Krebs T, Plehn T, Rousselot A and Vent S 2023 *SciPost Phys.* **14** 078 (*Preprint* 2110.13632)

[18] Quétant G, Raine J A, Leigh M, Sengupta D and Golling T 2024 *Phys. Rev. D* **110** 076023 (*Preprint* 2406.13074)

[19] Leigh M, Sengupta D, Quétant G, Raine J A, Zoch K and Golling T 2024 *SciPost Phys.* **16** 018 (*Preprint* 2303.05376)

[20] Hashemi B and Krause C 2024 *Rev. Phys.* **12** 100092 (*Preprint* 2312.09597)

[21] Amram O *et al.* 2025 *Rept. Prog. Phys.* **88** 116201 (*Preprint* 2410.21611)

[22] ATLAS Collaboration 2022 ATLAS Software and Computing HL-LHC Roadmap Tech. rep. CERN Geneva URL https://cds.cern.ch/record/2802918

[23] CMS Offline Software and Computing 2022 CMS Phase-2 Computing Model: Update Document Tech. rep. CERN Geneva URL https://cds.cern.ch/record/2815292

[24] Agostinelli S, Allison J, Amako K, Apostolakis J, Araujo H, Arce P, Asai M, Axen D, Banerjee S, Barrand G, Behner F, Bellagamba L, Boudreau J, Broglia L, Brunengo A, Burkhardt H, Chauvie S, Chuma J, Chytracek R, Cooperman G, Cosmo G, Degtyarenko P and Dell'Acqua A 2003 *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **506** 250–303 ISSN 0168-9002 URL https://www.sciencedirect.com/science/article/pii/S0168900203013688

[25] Allison J, Amako K, Apostolakis J, Araujo H, Arce Dubois P, Asai M, Barrand G, Capra R, Chauvie S, Chytracek R, Cirrone G, Cooperman G, Cosmo G, Cuttone G, Daquino G, Donszelmann M, Dressel M, Folger G, Foppiano F, Generowicz J, Grichine V, Guatelli S, Gumplinger P, Heikkinen A, Hrivnacova I, Howard A, Incerti S, Ivanchenko V, Johnson T, Jones F, Koi T, Kokoulin R, Kossov M, Kurashige H, Lara V, Larsson S, Lei F, Link O, Longo F, Maire M, Mantero A, Mascialino B, McLaren I, Mendez Lorenzo P, Minamimoto K, Murakami K, Nieminen P, Pandola L, Parlati S, Peralta L, Perl J, Pfeiffer A, Pia M, Ribon A, Rodrigues P, Russo G, Sadilov S, Santin G, Sasaki T, Smith D, Starkov N, Tanaka S, Tcherniaev E, Tome B, Trindade A, Truscott P, Urban L, Verderi M, Walkden A, Wellisch J, Williams D, Wright D and Yoshida H 2006 *IEEE Transactions on Nuclear Science* **53** 270–278

[26] Allison J, Amako K, Apostolakis J, Arce P, Asai M, Aso T, Bagli E, Bagulya A, Banerjee S, Barrand G, Beck B, Bogdanov A, Brandt D, Brown J, Burkhardt H, Canal P, Cano-Ott D, Chauvie S, Cho K, Cirrone G, Cooperman G, Cortés-Giraldo M, Cosmo G, Cuttone G, Depaola G, Desorgher L, Dong X, Dotti A, Elvira V, Folger G, Francis Z, Galoyan A, Garnier L, Gayer M, Genser K, Grichine V, Guatelli S, Guèye P, Gumplinger P, Howard A, Hřivnáčová I, Hwang S, Incerti S, Ivanchenko A, Ivanchenko V, Jones F, Jun S, Kaitaniemi P, Karakatsanis N, Karamitros M, Kelsey M, Kimura A, Koi T, Kurashige H, Lechner A, Lee S, Longo F, Maire M, Mancusi D, Mantero A, Mendoza E, Morgan B, Murakami K, Nikitina T, Pandola L, Paprocki P, Perl J, Petrović I, Pia M, Pokorski W, Quesada J, Raine M, Reis M, Ribon A, Ristić Fira A, Romano F, Russo G, Santin G, Sasaki T, Sawkey D, Shin J, Strakovsky I, Taborda A, Tanaka S, Tomé B, Toshito T, Tran H, Truscott P, Urban L, Uzhinsky V, Verbeke J, Verderi M, Wendt B, Wenzel H, Wright D, Wright D, Yamashita T, Yarba J and Yoshida H 2016 *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **835** 186–225 ISSN 0168-9002 URL https://www.sciencedirect.com/science/article/pii/S0168900216306957

[27] Matchev K T, Roman A and Shyamsundar P 2022 *SciPost Phys.* **12** 104 (*Preprint* 2002.06307)

[28] Butter A, Diefenbacher S, Kasieczka G, Nachman B and Plehn T 2021 *SciPost Phys.* **10** 139 (*Preprint* 2008.06545)

[29] Bieringer S, Butter A, Diefenbacher S, Eren E, Gaede F, Hundhausen D, Kasieczka G, Nachman B, Plehn T and Trabs M 2022 *JINST* **17** P09028 (*Preprint* 2202.07352)

[30] Watts S J and Crow L 2025 *Mach. Learn. Sci. Tech.* **6** 025046 (*Preprint* 2412.18041)

[31] Bahl H, Diefenbacher S, Elmer N, Plehn T and Spinner J 2025 (*Preprint* 2509.08048)

[32] Vaselli F, Rizzi A, Cattafesta F and Cicconofri G (CMS) URL https://cds.cern.ch/record/2858890?ln=de

[33] Krammer N (CMS) 2025 *PoS* **LHCP2024** 280

[34] Mazurek M 2025 Machine learning in LHCb Simulation: From fast to flash *13th Large Hadron Collider Physics Conference* (*Preprint* 2511.02020)

[35] 2025 Photon showers in the ATLAS fast calorimeter simulation: A voxelized dataset with minimized information loss and improved ML models Tech. rep. CERN Geneva all figures including auxiliary figures are available at https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PUBNOTES/ATL-SOFT-PUB-2025-003 URL https://cds.cern.ch/record/2942061

[36] Krause C and Shih D 2023 *Phys. Rev. D* **107** 113003 (*Preprint* 2106.05285)

[37] Golling T, Heinrich L, Kagan M, Klein S, Leigh M, Osadchy M and Raine J A 2024 *Mach. Learn. Sci. Tech.* **5** 035074 (*Preprint* 2401.13537)

[38] Birk J, Hallin A and Kasieczka G 2024 *Mach. Learn. Sci. Tech.* **5** 035031 (*Preprint* 2403.05618)

[39] Mikuni V and Nachman B 2025 *Phys. Rev. D* **111** L051504 (*Preprint* 2404.16091)

[40] Wildridge A J, Rodgers J P, Colbert E M, yao Y, Jung A W and Liu M 2024 Bumblebee: Foundation Model for Particle Physics Discovery *Postponed: Machine Learning and the Physical Sciences: Workshop at NeurIPS 2024* (*Preprint* 2412.07867)

[41] Birk J, Gaede F, Hallin A, Kasieczka G, Mozzanica M and Rose H 2025 *JINST* **20** P07007 (*Preprint* 2501.05534)

[42] Mikuni V and Nachman B 2025 *Phys. Rev. D* **111** 054015 (*Preprint* 2502.14652)

[43] Tani L, Pata J and Birk J 2025 *SciPost Phys. Core* **8** 046 (*Preprint* 2503.19165)

[44] Hallin A 2025 Foundation models for high-energy physics *2nd European AI for Fundamental Physics Conference* (*Preprint* 2509.21434)

[45] Bhimji W, Harris C, Mikuni V and Nachman B 2025 (*Preprint* 2510.24066)

[46] Dreyer F A, Grabarczyk R and Monni P F 2022 *Eur. Phys. J. C* **82** 564 (*Preprint* 2203.06210)

[47] Chappell A and Whitehead L H 2022 *Eur. Phys. J. C* **82** 1099 (*Preprint* 2207.03139)

[48] Beauchesne H, Chen Z E and Chiang C W 2024 *JHEP* **02** 138 (*Preprint* 2312.06152)

[49] Mokhtar F, Pata J, Garcia D, Wulff E, Zhang M, Kagan M and Duarte J 2025 *Phys. Rev. D* **111** 092015 (*Preprint* 2503.00131)

[50] Bonilla J L, Graczyk K M, Ankowski A M, Banerjee R D, Kowal B E, Prasad H and Sobczyk J T 2025 (*Preprint* 2508.12987)

[51] Raikwar P, Zaborowska A, McKeown P, Cardoso R, Piorczynski M and Yeo K 2025 (*Preprint* 2509.07700)

[52] Gaede F, Kasieczka G and Valente L 2025 (*Preprint* 2512.00187)

[53] McKeown P, Raikwar P and Zaborowska A 2025 (*Preprint* 2509.05108)

[54] Lipman Y, Chen R T Q, Ben-Hamu H, Nickel M and Le M 2023 (*Preprint* 2210.02747) URL https://arxiv.org/abs/2210.02747

[55] Shaul N, Singer U, Chen R T Q, Le M, Thabet A, Pumarola A and Lipman Y 2024 Bespoke non-stationary solvers for fast sampling of diffusion and flow models (*Preprint* 2403.01329) URL https://arxiv.org/abs/2403.01329

[56] Favaro L, Ore A, Schweitzer S P and Plehn T 2025 *SciPost Phys.* **18** 088 (*Preprint* 2405.09629)

[57] Favaro L, Giammanco A and Krause C 2025 Fast, accurate, and precise detector simulation with vision transformers *2nd European AI for Fundamental Physics Conference* (*Preprint* 2509.25169)

[58] Tancik M, Srinivasan P P, Mildenhall B, Fridovich-Keil S, Raghavan N, Singhal U, Ramamoorthi R, Barron J T and Ng R 2020 Fourier features let networks learn high frequency functions in low dimensional domains (*Preprint* 2006.10739) URL https://arxiv.org/abs/2006.10739

[59] Peebles W and Xie S 2023 Scalable diffusion models with transformers (*Preprint* 2212.09748) URL https://arxiv.org/abs/2212.09748

[60] Giannelli M F, Kasieczka G, Krause C, Nachman B, Salamani D, Shih D and Zaborowska A 2023 Fast calorimeter simulation challenge 2022 - dataset 1 version 3 https://doi.org/10.5281/zenodo.8099322 URL https://doi.org/10.5281/zenodo.8099322

[61] Buss T, Gaede F, Kasieczka G, Korol A, Krüger K, McKeown P and Mozzanica M 2025 (*Preprint* 2506.21720)

[62] Kansal R, Li A, Duarte J, Chernyavskaya N, Pierini M, Orzari B and Tomei T 2023 *Phys. Rev. D* **107** 076017 (*Preprint* 2211.10295)

[63] Heusel M, Ramsauer H, Unterthiner T, Nessler B and Hochreiter S 2017 *Advances in Neural Information Processing Systems* **30** arXiv:1706.08500 (*Preprint* 1706.08500) URL https://proceedings.neurips.cc/paper_files/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf

[64] Bińkowski M, Sutherland D J, Arbel M and Gretton A 2018 *International Conference on Learning Representations* arXiv:1801.01401 (*Preprint* 1801.01401) URL https://openreview.net/forum?id=r1lUOzWCW

[65] Szegedy C, Vanhoucke V, Ioffe S, Shlens J and Wojna Z 2015 *arXiv e-prints* arXiv:1512.00567 (*Preprint* 1512.00567)

[66] Kansal R, Pareja C, Hao Z and Duarte J 2023 *Journal of Open Source Software* **8** 5789 URL https://joss.theoj.org/papers/10.21105/joss.05789

[67] Das R, Favaro L, Heimel T, Krause C, Plehn T and Shih D 2024 *SciPost Phys.* **16** 031 (*Preprint* 2305.16774)

[68] Giannelli M F, Kasieczka G, Krause C, Nachman B, Salamani D, Shih D and Zaborowska A 2022 Fast calorimeter simulation challenge 2022 - dataset 2 https://doi.org/10.5281/zenodo.6366271 URL https://doi.org/10.5281/zenodo.6366271

[69] Giannelli M F, Kasieczka G, Krause C, Nachman B, Salamani D, Shih D and Zaborowska A 2022 Fast calorimeter simulation challenge 2022 - dataset 3 https://doi.org/10.5281/zenodo.6366324 URL https://doi.org/10.5281/zenodo.6366324

[70] Allaire C, Gessinger P, Hdrinka J, Kiehn M, Kimpel F, Niermann J, Salzburger A and Sevova S 2022 Opendatadetector URL https://doi.org/10.5281/zenodo.6445359

[71] Abada A *et al.* (FCC) 2019 *Eur. Phys. J. C* **79** 474

[72] Bacchetta N *et al.* 2019 (*Preprint* 1911.12230)

[73] Mlynarikova M (ALLEGRO) 2025 *EPJ Web Conf.* **320** 00022

[74] Kansal R, Pareja C, Hao Z and Duarte J 2023 *J. Open Source Softw.* **8** 5789

[75] Aad G *et al.* (ATLAS) 2022 *Comput. Softw. Big Sci.* **6** 7 (*Preprint* 2109.02551)

[76] Aihara H *et al.* (ILC) 2019 (*Preprint* 1901.09829)

[77] Ernst F, Favaro L, Krause C, Plehn T and Shih D 2025 *SciPost Phys.* **18** 081 (*Preprint* 2312.09290)

[78] Toledo-Marin J Q *et al.* 2025 *npj Quantum Inf.* **11** 114 (*Preprint* 2410.22870)