Journal of Instrumentation

PAPER • OPEN ACCESS

CaloClouds II: ultra-fast geometry-independent highly-granular calorimeter simulation

To cite this article: Erik Buhmann *et al* 2024 *JINST* **19** P04020

Published by IOP Publishing for Sissa Medialab

Received: *December 4, 2023*
Revised: *January 23, 2024*
Accepted: *February 20, 2024*
Published: *April 18, 2024*

CaloClouds II: ultra-fast geometry-independent highly-granular calorimeter simulation

Erik Buhmann,[a] **Frank Gaede**,[b,c] **Gregor Kasieczka**,[a,c] **Anatolii Korol**,[b,*] **William Korcari**,[a] **Katja Krüger**[b] **and Peter McKeown**[b]

[a] *Institut für Experimentalphysik, Universität Hamburg, Luruper Chaussee 149, 22607 Hamburg, Germany*
[b] *Deutsches Elektronen-Synchrotron DESY, Notkestr. 85, 22607 Hamburg, Germany*
[c] *Center for Data and Computing in Natural Sciences CDCS, Deutsches Elektronen-Synchrotron DESY, Notkestr. 85, 22607 Hamburg, Germany*

E-mail: anatolii.korol@desy.de

Abstract: Fast simulation of the energy depositions in high-granular detectors is needed for future collider experiments at ever-increasing luminosities. Generative machine learning (ML) models have been shown to speed up and augment the traditional simulation chain in physics analysis. However, the majority of previous efforts were limited to models relying on fixed, regular detector readout geometries. A major advancement is the recently introduced CaloClouds model, a geometry-independent diffusion model, which generates calorimeter showers as point clouds for the electromagnetic calorimeter of the envisioned International Large Detector (ILD).

In this work, we introduce CaloClouds II which features a number of key improvements. This includes continuous time score-based modelling, which allows for a 25-step sampling with comparable fidelity to CaloClouds while yielding a 6× speed-up over Geant4 on a single CPU (5× over CaloClouds). We further distill the diffusion model into a consistency model allowing for accurate sampling in a single step and resulting in a 46× speed-up over Geant4 (37× over CaloClouds). This constitutes the first application of consistency distillation for the generation of calorimeter showers.

Keywords: Simulation methods and programs; Analysis and statistical methods; Calorimeter methods

ArXiv ePrint: 2309.05704

*Corresponding author.

https://doi.org/10.1088/1748-0221/19/04/P04020

2024 JINST 19 P04020

Contents

1 Introduction

Accurate simulations of particle physics experiments are crucial for comparing theory predictions with experimental results. With the planned high luminosity upgrade to the Large Hadron Collider (LHC) [1] and other envisioned collider experiments like those at the International Linear Collider (ILC) [2], experimental data is going to be taken at ever increasing rates. The amount of simulated events needs to keep up with these rates, which is difficult to achieve with current Monte Carlo simulations and the projected computing budgets at large experiments [3, 4].

Detector simulations, such as the simulation of the sensor response in highly granular calorimeters, can be augmented or sped up by employing modern generative machine learning methods [5–8]. Recent studies have explored the simulation of calorimeter showers with various generative models such as generative adversarial networks (GANs) [5, 9–19], autoencoders and their variants [20–25], and normalizing flows [26–33]. Additionally, diffusion models [34–38], also referred to as score-based generative models, have been shown to provide very high fidelity on calorimeter data [39–43]. Beyond detector simulation, generative models have, for example, also been explored as event generators [44–50] and parton shower simulators [51–63].

Most previous generative calorimeter models rely on a fixed data geometry, representing calorimeter showers as 3-dimensional images with the energy as the "color" channel and each pixel representing a calorimeter sensor. Modern high granularity calorimeters consist of many thousands of sensor cells or more (e.g. 6 million for the planned CMS HGCal [64]), but a given shower often deposits energy in only a small fraction of cells resulting in very sparse 3D image representations. Hence, it is much more computationally efficient to only simulate the actual energy depositions with

2024 JINST 19 P04020

a generative model. This can be achieved by describing the shower with only the coordinates and energies deposited — i.e. a *point cloud*. Such a multidimensional calorimeter point cloud can be represented by four features, the three-dimensional spatial coordinates and the cell energy, with the number of points equivalent to the number of cells containing hits.

In addition to computational efficiency, such point cloud showers have the major advantage that they can represent not only cell energies, but also much more granular GEANT4 step information, i.e. simulated energy depositions in the material, not accessible in experiments. Such GEANT4 step point clouds are largely independent of the cell structure within a layer of a given calorimeter, effectively allowing the translation-invariant projection of the shower into any part of the calorimeter, regardless of cell type. These projections with GEANT4 step point clouds are less likely to produce artifacts due to gaps or cell staggering than cell-level point clouds would, resulting in a largely geometry-independent description of the calorimeter shower. This approach is complimentary to a geometry-aware model [65], which is trained with a dataset containing various calorimeter geometries.

Previous point cloud and graph generative models explored in particle physics [55, 58–61, 63, 66] were only used for relatively small numbers of points. However, energetic calorimeter showers in high granularity calorimeters consist of $O(1000)$ points. To generate such showers, we recently introduced CALOCLOUDS [40] a generative model able to accurately generate photon showers in the form of point clouds with several thousands of points (namely clustered GEANT4 steps), in order to achieve geometry-independence. Since then, a specific comparison between a generative model for fixed geometry and a generative model for point cloud structured calorimeter showers on cell-level was performed in ref. [41].

This CALOCLOUDS architecture consists of multiple sub-models with a diffusion model (see section 3.1 for details) at its core. Most diffusion models, including the one used in CALOCLOUDS, are currently held back by their slow sampling speed, as many evaluation steps have to be performed to generate events. However, recent advances in computer vision achieve very high generative fidelity on natural image data with $O(10)$ model evaluations using advanced training paradigms and novel ordinary and stochastic differential equation solvers [67–70]. In this work, we first leverage recent advances in the training and sampling procedure of diffusion models in order to generate samples with the CALOCLOUDS II model[1] using much fewer model evaluations than the original CALOCLOUDS model, by following the diffusion paradigm introduced in ref. [68].

Another research direction to speed up generative models is the distillation of diffusion models into models which require significantly fewer function evaluations during sampling than the original model [71–75]. Recently, consistency models have been introduced as a novel kind of generative model allowing for single and multi-step data generation [76]. These consistency models can either be trained ab-initio or distilled from an already trained diffusion model. We demonstrate the ability to distill our diffusion model into a consistency model, thereby allowing data generation with a single model evaluation leading to further speed-ups.

In summary, the proposed CALOCLOUDS II contains the following adjustments:

1. The previously used discrete-time diffusion process is replaced with the continuous-time diffusion paradigm introduced in ref. [68]. This allows for fewer diffusion iterations during sampling.

[1]The code is available at https://github.com/FLC-QU-hep/CaloClouds-2.

2024 JINST 19 P04020

2. The common latent space is removed as we have noticed no advantage for the generative fidelity when generating photon calorimeter showers. This removal yields a simplified model architecture and improved training and sampling speeds.

3. We add a calibration to the energy per calorimeter layer as well as applying a calibration to the center of gravity in the X- and Y-direction of the generated point cloud showers. This replaces the previous total energy calibration and improves the generative fidelity in the longitudinal energy distribution.

4. Further, we apply consistency distillation to distill the diffusion model into a consistency model [76], allowing single step generation and therefore greatly improved sampling speed. We refer to this model as CaloClouds II (CM).

In section 2 we describe the point cloud dataset used for training and evaluation. The diffusion paradigm and model components of the CaloClouds II model are explained in section 3. We compare the generative fidelity of CaloClouds II and its variant to the original CaloClouds model in section 4 and draw our conclusions in section 5.

2 Data samples

To compare the performance of our improved CaloClouds II model we use the same dataset as in ref. [40]. The data describes a calorimeter shower in the form of a point cloud. Each calorimeter shower consists of energy depositions of photons showering in a section of the high-granular electromagnetic calorimeter (ECAL) of the envisioned International Large Detector (ILD) [77]. As a sampling calorimeter, it consists of 30 layers with passive tungsten material and active silicon sensors. All individual silicon layers consist of small 5×5 mm readout cells with a thickness of 0.5 mm. Between the first 20 active layers in the longitudinal direction there are passive layers with a thickness of 2.1 mm and between the remaining 10 layers the passive layers have a thickness of 4.2 mm. We simulated the dataset with Geant4 Version 10.4 (using the QGSP_BERT physics list) implemented in the iLCSoft framework [78]. The simulated geometric model is implemented in DD4hep [79] and includes realistic gaps between the sensors and position dependent irregularities. More simulation details can be found in ref. [40].

During the full Geant4 simulation up to 40,000 individual energy depositions originating from secondary particles traversing the active sensor material are registered (depending on the incident photon energy). These energy depositions are commonly referred to as Geant4 *steps*. All steps that fall into the volume of the same sensor are subsequently summed, resulting in the energy deposited in a cell *hit*. These cell hits (up to 1,500 at 90 GeV) are then used for downstream analysis as it is the same low-level information that is measurable in a real experimental setting.

Ideally, a generative model should produce cell-level hits to make the full Geant4 simulation more computationally efficient. Cell-level information is also generated in all other approaches for fast calorimeter shower simulation with generative machine learning models. However, generating discrete cell hits directly in the form of a point cloud is challenging, as minor imperfections such as generating multiple points in the same calorimeter cell can heavily impact the generative fidelity in various high level observables like the total number of cell hits N_{hits}.

2024 JINST 19 P04020

Therefore, it could be advantageous to generate point clouds not on hit-level but on GEANT4 step-level, i.e. many simulated very granular energy depositions per cell, resulting in much larger point clouds where points are continuously distributed in space (as opposed to discrete cell hits). Yet, we found generating a point cloud with up to 40,000 steps prohibitively expensive from a computational point-of-view. Additionally such a high resolution is not necessary for good generative fidelity. Therefore in ref. [40] we introduced a middle ground: we cluster the up to 40,000 GEANT4 steps into up to 6,000 *points*. For this clustering, the steps are grouped into their layer and their energy is binned in an ultra-high granularity grid with 36× higher granularity than the cell resolution, resulting in a square grid size of $0.83 \times 0.83\,\mathrm{mm}^2$. This results in a clustered point cloud of up to 6,000 points — sufficiently small to be generated with the CALOCLOUDS model, yet distributed in discrete positions with sufficiently small separation so as to be approximately a continuous point distribution in 3D space.

In addition to a computationally efficient simulation, this makes the generated calorimeter point cloud largely geometry-independent of the actual cell layout of the calorimeter, unlike point clouds based on cell-level energy depositions. This ultra-high granularity calorimeter point cloud can be projected into any part of the calorimeter (except changing its depth), without introducing reconstruction artifacts due to for example gaps and cell staggering, as successfully shown in ref. [40].

To produce the training set, a total of 524,000 showers were generated with GEANT4, with an incident energy uniformly sampled between 10 and 90 GeV. Additionally, multiple test sets were generated: 40,000 showers uniformly distributed in energy for the figures shown in section 4.1; 2,000 showers for the single energy plots at 10, 50, and 90 GeV; and 500,000 showers for calculating the evaluation metrics and the classifier score in section 4.2 and section 4.3.

Each point of the point cloud has four features: the X- and Y-position (transverse to the incident particle direction), the Z-position (parallel to the incident particle direction), and the *energy*. As a pre-processing step, the passive material regions are removed such that the point locations in the dataset also become continuous in the longitudinal Z-axis. The position features, X, Y, and Z, are each normalized to the range $[-1, 1]$. The energy feature of the 4d point cloud is given in MeV.

As it is important for downstream analyses to accurately simulate the behaviour of photon showers on the level of the physical geometry, i.e. at cell level, all results shown in section 4.1 to 4.3 are on cell-level. To this end, the calorimeter point cloud — with either up to 40,000 points for GEANT4 or with up to 6,000 points for those generated with CALOCLOUDS/CALOCLOUDS II— are binned to the realistic ILD ECAL layout (including detector irregularities and gaps) with $30 \times 30 \times 30$ calorimeter cells.

3 Generative model

The CALOCLOUDS II model is an improved version of the original CALOCLOUDS architecture from ref. [40]. First, we revisit the main model components of the CALOCLOUDS model, afterwards we outline the improvements made in CALOCLOUDS II.

CALOCLOUDS is a combination of two normalizing flows [80], a VAE-like encoder [81], and a discrete time Denoising Diffusion Probabilistic Model (DDPM) [37]. Specifically, it consists of the *Shower Flow*, a normalizing flow generating conditioning and calibration features; the *EPiC Encoder*, an encoder based on Equivariant Point Cloud (EPiC) layers [59] to encode calorimeter showers during training into a latent space for model conditioning; the *Latent Flow*, a normalizing flow trained to model the encoded latent space during sampling; and a diffusion model, called *PointWise Net*, which

2024 JINST 19 P04020

is a DDPM-based diffusion model generating each point independent and identically distributed (i.i.d.) based on a common latent space, incident energy and number of points conditioning. The models are implemented using PyTorch 1.13 [82].

In the following, we outline the differences between CaloClouds and CaloClouds II. The largest conceptual difference is the change of the diffusion paradigm. We move from a discrete time diffusion process (DDPM), in which the training and sampling is performed with the same number of diffusion steps, to a continuous time diffusion paradigm based on ref. [68], sometimes referred to as *EDM* diffusion or *k-diffusion*. This EDM diffusion allows for training a continuous time score function, which can be used to denoise any noise level, thereby separating the training and sampling procedure and allowing for sampling with various ordinary differential equation (ODE) and stochastic differential equation (SDE) solvers and different step sizes. Crucially, it allows to trade off sampling speed and sampling fidelity without retraining. We find good performance with the 2^{nd}-order Heun ODE solver and the step size parameterisation suggested in ref. [68]. Additional details on the diffusion paradigm is given in the following section 3.1.

As a second change, CaloClouds II simplifies the original model. We noticed that for the photon calorimeter shower point clouds we are generating in this study, the shared latent space between points is not necessary for high generative fidelity. Therefore the latent dimensionality can be set to zero, so the *EPiC Encoder* and the *Latent Flow* are removed. By discarding it we achieve a simpler model as well as improved training and sampling efficiency.

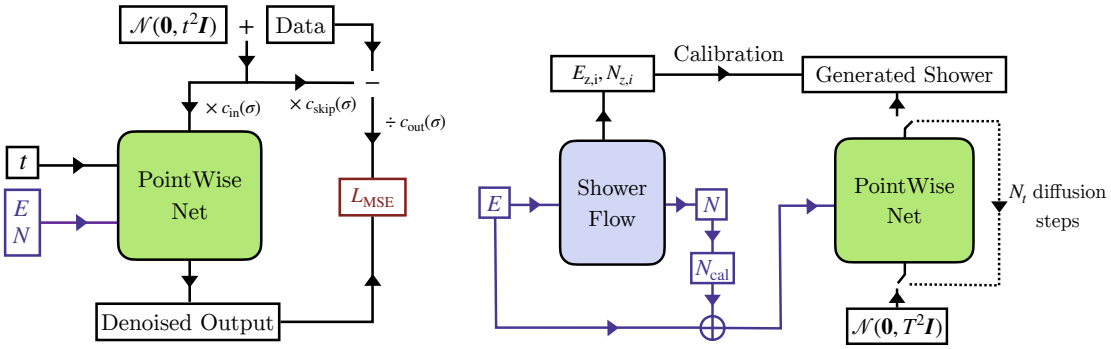
Next, the *Shower Flow* for generating conditioning and calibration features is expanded to generate the total number of points, total visible energy, the relative number of points and energy of each calorimeter layer in the Z-direction, as well as the center of gravity in the X- and Y-direction. This flow is conditioned on the incident particle energy only. The total number of points generated per shower is used — together with the incident particle energy — for the conditioning of the PointWise Net diffusion model.

Overall, the Shower Flow is composed of ten blocks, each with seven coupling layers [83, 84] conditioned on the incident particle energy. It is implemented using the Pyro package [85]. The Shower Flow is trained once for 350k iterations and used for all three models (CaloClouds, CaloClouds II, and CaloClouds II (CM)) compared in section 4.

The post-diffusion calibration expands upon the calibration in ref. [40]: the number of hits per layer is calibrated by ordering all points in the Z-coordinate and setting iteratively the first $N_{z,i=1}$ points to $z_i = 1$ (first layer), the second $N_{z,i=2}$ points to $z_i = 2$ (second layer) and so on until the 30^{th} layer. Afterwards, we calibrate the total layer energy by re-weighting each point energy to sum up to the predicted layer energy $E_{\text{pred},i}$. Finally, we calculate the center of gravity in X and $Y-$direction of the point cloud and subtract its difference in comparison to the predicted center of gravity from each point's $X-$ and $Y-$ coordinate to calibrate the overall point cloud center of gravity in these two dimensions. Note that we further set points with negative generated energy to zero.

During sampling, the number of points predicted by the Shower Flow is calibrated before being used for the conditioning of the Latent Flow and the PointWise Net. The calibrated number of points is given by $N_{\text{cal}} = N_{\text{uncal}} \cdot p_{\text{gen}}(p_{\text{data}}(N_{\text{uncal}}))$, where p_{data} is a cubic polynomial fit of the ratio of the number of points $N_{\text{uncal, G4}}$ to the number of cell hits $N_{\text{cell, G4}}$ of the Geant4 showers and p_{gen} is a fit of the ratio of number of cell hits $N_{\text{cell, gen}}$ to the (uncalibrated) number of points $N_{\text{uncal, gen}}$ of a given model. Hence, this polynomial fit p_{gen} is performed for each model separately. More details

2024 JINST 19 P04020



(a) Training (b) Sampling

Figure 1. Illustration of the training and sampling procedure of the CaloClouds II model. **(a)** During training a random continuous time step t is trained conditioned on the shower energy E and number of points N. The loss, L_{MSE}, is approximated by a simple mean squared error (MSE) between the noised data and the denoised output. The scaling functions c_{in}, c_{out}, and c_{skip} are defined following eq. 3.4. **(b)** During sampling the E-conditional Shower Flow generates N as well as shower observables for calibration. After a N calibration the PointWise Net denoises iteratively noise $\mathcal{N}(\mathbf{0}, T^2\boldsymbol{I})$ into a calorimeter shower. When sampling with CaloClouds II (CM) only one denoising step is performed.

on the model components and the calibrations can be found in ref. [40]. A schematic overview of the training and sampling procedure is shown in figure 1.

In the following section 3.1 we describe the continuous time diffusion paradigm implemented in the CaloClouds II model and in section 3.2 we outline its distillation into a consistency model, referred to as CaloClouds II (CM). Both models use the same model components outlined above. Details on the training and sampling hyperparameters are outlined in section 3.3.

3.1 Diffusion model

The diffusion model [34] used in the CaloClouds model is a Denoising Diffusion Probabilistic Model (DDPM) with the same discrete time steps during model training and sampling [37, 86]. Since the introduction of DDPM, subsequent works, i.e. refs. [38, 68, 87], have shown that it is advantageous to train a diffusion model with continuous time conditioning. This allows for a more flexible sampling regime for which various SDE and ODE solvers with either a fixed or an adaptive number of solving steps can be applied.

In the following, we outline the key parts of a diffusion model based on the paradigm outlined in ref. [68]. The training of a diffusion model starts by diffusing a data distribution $p_{\text{data}}(x)$ with an SDE [38] in the forward direction ("data" → "noise") via

$$\mathrm{d}\mathbf{x}_t = \boldsymbol{\mu}\left(\mathbf{x}_t, t\right)\mathrm{d}t + \sigma(t)\mathrm{d}\mathbf{w}_t, \tag{3.1}$$

where t is a fixed time step defined in the interval $t \in [0, T]$ with $T > 0$ as a hyperparameter. $\boldsymbol{\mu}(\cdot, \cdot)$ and $\sigma(\cdot)$ denote the *drift* and *diffusion* coefficients, and $\mathbf{w}_{t\in[0,T]}$ is the standard Brownian motion. The distribution of $\mathbf{x}_t \sim p_t(\mathbf{x}) = p_{\text{data}}(\mathbf{x}) * \mathcal{N}(\mathbf{0}, T^2\boldsymbol{I})$ ($*$ as the convolution operator) and at time step zero it is identical to the data distribution $p_0(\mathbf{x}) = p_{\text{data}}(\mathbf{x})$. When reversing this diffusion process ("noise" → "data"), a so called *probability flow ODE* emerges with a solution trajectory

2024 JINST 19 P04020

sampled at time step t given by

$$\mathrm{d}\mathbf{x}_t = \left[\boldsymbol{\mu}\left(\mathbf{x}_t, t\right) - \frac{1}{2}\sigma(t)^2 \nabla \log p_t\left(\mathbf{x}_t\right) \right] \mathrm{d}t, \tag{3.2}$$

with $\nabla \log p_t\left(\mathbf{x}_t\right)$ as the *score function* of $p_t(\mathbf{x})$. As suggested in ref. [68], we set the coefficients in the SDE in eq. 3.1 to $\boldsymbol{\mu}(\mathbf{x}, t) = 0$ and $\sigma(t) = \sqrt{2t}$ to ensure that $p_T(\mathbf{x})$ is close to the distribution $\mathcal{N}(\mathbf{0}, T^2\boldsymbol{I})$. Since the exact analytical score function is usually unknown, we train a neural network with weights ϕ as a score model $\boldsymbol{s}_\phi(\mathbf{x}, t) \approx \nabla \log p_t\left(\mathbf{x}_t\right)$ to get the empirical probability flow ODE:

$$\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = -t\boldsymbol{s}_\phi\left(\mathbf{x}_t, t\right) \tag{3.3}$$

For the purpose of numerically stable scaling behaviour, we follow ref. [68] and actually train a separate network $\boldsymbol{d}_\phi$ with t-dependent skip connections from which $\boldsymbol{s}_\phi$ is derived:

$$\boldsymbol{s}_\theta(\mathbf{x}, t) = c_{\text{skip}}(t)\,\mathbf{x} + c_{\text{out}}(t)\,\boldsymbol{d}_\theta\left(c_{\text{in}}(t)\,\mathbf{x}, t\right) \tag{3.4}$$

The coefficients are time dependent and control the skip connection via $c_{\text{skip}} = \sigma_{\text{data}}^2/\left(\sigma_{\text{data}}^2 + t^2\right)$, the input scaling via $c_{\text{in}} = t \cdot \sigma_{\mathbf{data}}/\sqrt{\sigma_{\text{data}}^2 + t^2}$ and the output scaling via $c_{\text{out}} = 1/\sqrt{\sigma_{\text{data}}^2 + t^2}$. The hyperparameter σ_{data} corresponds roughly to the variance of $p_{\text{data}}(\mathbf{x})$ and is set to $\sigma_{\text{data}} = 0.5$. During training a random time step is drawn from the continuous noise distribution $\ln(t) = \mathcal{N}\left(P_{\text{mean}}, P_{\text{std}}^2\right)$, with $P_{\text{mean}} = -1.2$ and $P_{\text{std}} = 1.2$ (the default parameters chosen in ref. [68]), and the loss is given by:

$$\mathbb{E}_{t,\mathbf{x}_t,\mathbf{x}_0}\left[\|\boldsymbol{s}_\theta(\mathbf{x}_t, t) - \mathbf{x}_0\|_2^2\right] \tag{3.5}$$

An illustration of this training process can be found in figure 1(a).

For sampling from the trained score model, one samples from noise at time step T as $\hat{\mathbf{x}}_T \sim \mathcal{N}(\mathbf{0}, T^2\boldsymbol{I})$ and integrates the probability flow ODE in eq. 3.2 over discrete time steps backwards in time using a numerical ODE solver. This results in a sample $\hat{\mathbf{x}}_0$ which provides a good approximation of a sample from the data distribution $p_{\text{data}}(\mathbf{x})$. In practice the solver is usually stopped at a small positive value $\epsilon > 0$ to avoid numerical instabilities resulting in the approximate sample $\hat{\mathbf{x}}_\epsilon \approx \hat{\mathbf{x}}_0$. For our sampling, we use the suggested values and step scheduling from ref. [68] with $T = 80$ and $\epsilon = 0.002$, and apply the 2$^{\text{nd}}$ order Heun ODE solver.

3.2 Consistency model

Consistency Models (CM) [76] are a recently introduced generative architecture. They allow for single-step or multi-step generation with the same model and can be trained standalone or distilled from a diffusion model that has already been trained. A *consistency model* $\boldsymbol{f}_\Phi$ with weights Φ is trained to estimate the *consistency function* $\boldsymbol{f}$ from data. The consistency function is defined as $\boldsymbol{f} : (\mathbf{x}_t, t) \rightarrow \mathbf{x}_\epsilon$ and is *self-consistent* in the sense that any pair of $(\mathbf{x}_t, t)$ belong to the same probability flow ODE trajectory. This means that the result of a function evaluation at any point on this trajectory leads to the same result, i.e. $\boldsymbol{f}(\mathbf{x}_t, t) = \boldsymbol{f}(\mathbf{x}_{t'}, t')$ for all $t, t' \in [\epsilon, T]$. The time interval describes the minimum noise at time step ϵ and the maximum noise at time T.

For sampling from a trained consistency model in a single model pass, one initializes $\hat{\mathbf{x}}_T \sim \mathcal{N}(\mathbf{0}, T^2\boldsymbol{I})$ and performs one function evaluation to get $\hat{\mathbf{x}}_\epsilon = \boldsymbol{f}_\Phi(\mathbf{x}_T, T)$. It is also possible to sample

2024 JINST 19 P04020

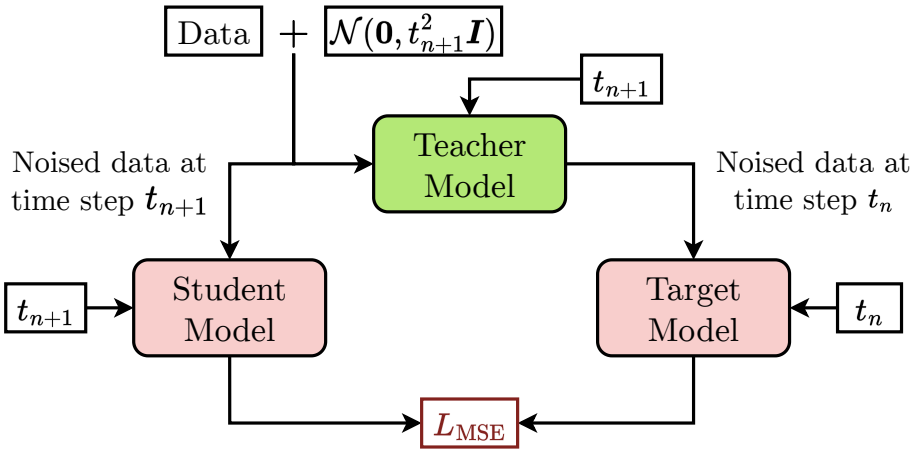


Figure 2. Illustration of the consistency distillation process distilling the diffusion model of CaloClouds II (teacher model) into a consistency model (student and target model). The student model is updated via gradient descent and the target model is updated as an exponential moving average of the student model weights.

with multiple model passes by first evaluating $\boldsymbol{f}_\Phi(\mathbf{x}_T, T)$, and then adding noise again from $\mathcal{N}(\mathbf{0}, t^2\boldsymbol{I})$ to denoise a second time. This can be done in an alternating fashion for an arbitrary number of steps. Often multi-step generation appears to improve sample fidelity [63, 68], however we are able to achieve comparable fidelity to the original diffusion model with only a single model evaluation and therefore limit ourselves to this most efficient scenario.

In line with ref. [76], we found improved training fidelity when distilling the consistency model from a diffusion model instead of training it individually. For this purpose we distill the consistency model $\boldsymbol{f}_\Phi(\mathbf{x}, t)$ from the diffusion model $\boldsymbol{s}_\phi(\mathbf{x}, t)$ based on the PointWise Net of CaloClouds II introduced in the previous section 3.1. The distillation is performed by separating the continuous time space $[\epsilon, T]$ into $N-1$ sub intervals (we use $N = 18$). The interval boundaries are determined by the same step size parameterisation as in the diffusion model sampling formulation [68]. During training a random boundary time step $t_{n\in[1,N]}$ is chosen to perform the distillation. We refer to the original diffusion model here as the *teacher* model $\boldsymbol{s}_\phi(\mathbf{x}, t)$ and to the distilled consistency model during distillation as the *student* model $\boldsymbol{f}_\Phi(\mathbf{x}, t)$. Additionally, we call the final distilled consistency model the *target* model $\boldsymbol{f}_{\Phi^-}(\mathbf{x}, t)$. We use the self-consistency property of the consistency model for training since it requires a well trained model to obey $\boldsymbol{f}_\Phi(\mathbf{x}_{t_{n+1}}, t_{n+1}) = \boldsymbol{f}_\Phi(\mathbf{x}_{t_n}, t_n)$. The neighboring points $(\mathbf{x}_{t_{n+1}}, \mathbf{x}_{t_n})$ on the probability flow ODE trajectory are obtained by sampling $\mathbf{x} \sim p_{\text{data}}$, adding noise to it to get $\mathbf{x}_{t_{n+1}} \sim \mathcal{N}(\mathbf{x}, t_{n+1}^2\boldsymbol{I})$ and performing one ODE solver step with the teacher diffusion model to compute $\mathbf{x}_{t_n} = \boldsymbol{s}_\phi(\mathbf{x}_{t_{n+1}}, t_{n+1})$. This allows the student consistency model $\boldsymbol{f}_\Phi(\mathbf{x}, t)$ to be trained with the loss:

$$\mathbb{E}_{t,\mathbf{x}_t,\mathbf{x}_0}\left[\left\|\boldsymbol{f}_\Phi(\mathbf{x}_{t_{n+1}}, t_{n+1}) - \boldsymbol{f}_{\Phi^-}(\mathbf{x}_{t_n}, t_n)\right\|_2^2\right] \tag{3.6}$$

The target model $\boldsymbol{f}_{\Phi^-}(\mathbf{x}_t, t)$ weights Φ^- are updated after every iteration as a running average of the student model weights Φ. An overview of the distillation procedure is illustrated in figure 2.

2024 JINST 19 P04020

3.3 Training and Sampling

The diffusion model in CaloClouds II was trained for 2M iterations with a batch size of 128 using the Adam optimizer [88] with a fixed learning rate of 10^{-4}. As the final model, we use an exponential moving average (EMA) of the model weights. We scan several values for the number of ODE solver steps N and find $N = 13$ optimal as with fewer steps than this, the generative fidelity as probed by the correct learning of physically relevant shower shapes with CaloClouds II deteriorates. This results in $2N - 1$ diffusion model evaluations since the last step of the Heun ODE solver does not perform a 2nd order correction. Compared to CaloClouds with 100 function evaluations this already hints at a significant computational speed-up.

The diffusion model used in CaloClouds II was distilled into a consistency model for CaloClouds II (CM) by using the Adam optimizer with a fixed learning rate of 10^{-4} for 1M iterations with a batch size of 256. Notably, only a single training is necessary for distilling a model which is able to perform single step generation, as opposed to the multiple trainings required for e.g. progressive distillation [42, 66, 72].

4 Results

In the following, we compare the original CaloClouds model with the improved CaloClouds II model and its distilled variant CaloClouds II (CM). To achieve a fair comparison between the three models, we use the same training of the Shower Flow and the same calibration procedure for all three models. Hence, the Shower Flow from the CaloClouds II model was also used for generating samples with the CaloClouds model — a slight modification compared to the original CaloClouds model in ref. [40]. This means that the samples generated with the CaloClouds model also include the energy per layer and center of gravity in X and Y calibration. For the Latent Flow and the PointWise Net of CaloClouds the same model weights as in ref. [40] were used.

We first show the performance of our generative models based on the same observables as discussed in ref. [40] in section 4.1. Next, in section 4.2, we quantify the performance of the models with multiple Wasserstein-distance-based scores for the usual set of calorimeter shower observables and in section 4.3 we use a classifier to distinguish between simulated Geant4 showers and generated showers based on the calculated shower observables. Finally in section 4.4 we benchmark the computational efficiency of our models and compare them to the baseline simulation timing with Geant4.

4.1 Physics performance

In this section, we compare various calorimeter shower distributions from ref. [40] between the Geant4 test set and datasets generated using CaloClouds, CaloClouds II, and CaloClouds II (CM). First, we compare various cell-level and shower observables calculated from the model generated showers to Geant4 simulations with samples of incident photons with energies uniformly distributed between 10 and 90 GeV (also referred to as *full spectrum*). In figure 3 we investigate three representations of the energy distributed in the calorimeter cells, namely the per-cell energy distribution (left), the radial shower profile (center) and the longitudinal shower profile (right). The per-cell energy distribution contains the energy of the cells of all showers in the test dataset. The peak of the distribution at about 0.2 MeV corresponds to the most probable energy deposition of a minimum ionising particle (MIP) in the silicon sensor. For downstream analyses a cell energy cut at half a MIP is applied, since

2024 JINST 19 P04020

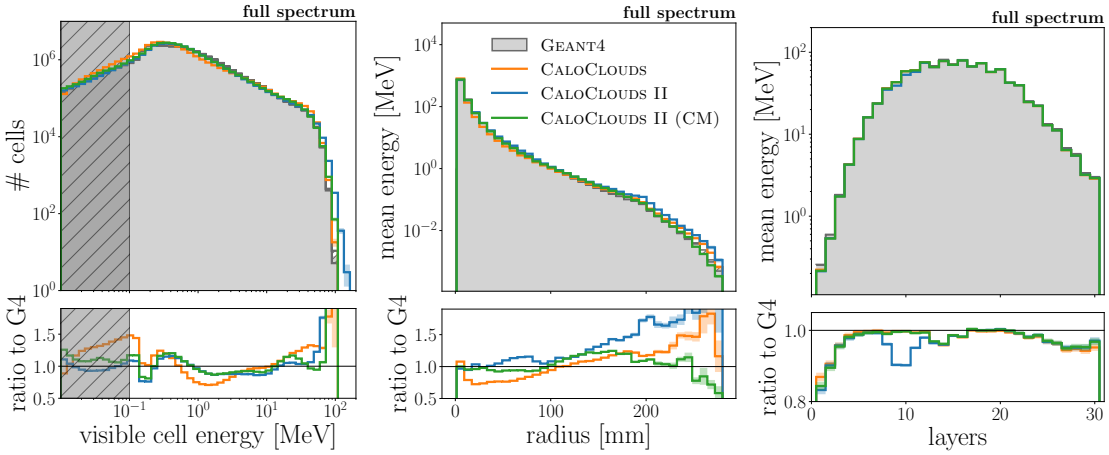


Figure 3. Histogram of the cell energies (left), radial shower profile (center), and longitudinal shower profile (right) for GEANT4, CALOCLOUDS, CALOCLOUDS II, and CALOCLOUDS II (CM). In the cell energy distribution, the region below 0.1 MeV is grayed out (see main text for details). All distributions are calculated with 40,000 events sampled with a uniform distribution of incident particle energies between 10 and 90 GeV. The bottom panel provides the ratio to GEANT4. The error band corresponds to the statistical uncertainty in each bin.

below this threshold the sensor response is indistinguishable from electronic noise. Hence this cut was applied to all showers when calculating the shower observables and scores in this section. All models describe the cell energy distribution reasonably well. For most of the range the CALOCLOUDS II models perform better than CALOCLOUDS, however there are a few outliers with energies which are too high produced by CALOCLOUDS II compared to the other two models.

The radial shower profile describes the average radial energy distribution around the central shower axis (in Z-direction) of the ECAL. Below a radius of about 180 mm, the distribution is well described by all three models, above 180 mm, the models deviate from GEANT4. Overall the CALOCLOUDS II (CM) model represents the GEANT4 distribution most closely. Note that this is a distribution that is not directly impacted by any of the post-diffusion calibrations performed and is therefore a good benchmark for the effectiveness of the point cloud diffusion approach alone.

The longitudinal shower profile describes how much energy is deposited on average in each of the 30 calorimeter layers. In the previous iteration of CALOCLOUDS it was not well modeled, but since we now calibrate the energy per layer with the improved Shower Flow for the generated point clouds it is well modelled. However, we observe deviations in the first few layers for all three models. Since they share the same Shower Flow, we expect future improvements in this model to translate to an improved longitudinal profile. Further, a small outlier can be seen for the CALOCLOUDS II model around the 10^{th} layer. The alternating higher and lower energy depositions per layer are due to the fact that for technical reasons, pairs of silicon sensors surrounding one tungsten absorber layer and facing opposite directions are installed into a tungsten structure with every other absorber layer. This results in the observed pair-wise difference in the sampling fraction between consecutive layers.

In figure 4 we show the center of gravity distribution $m_{1,i\in\{X,Y,Z\}}$ (the energy weighted shower centroid) in the X-, Y-, and Z-directions. Note that in the X- and Y-directions these distribution are calibrated for the original point cloud, before the cell-level observables are calculated. While the $m_{1,X}$ distribution is very well modelled by all three generative models, $m_{1,Y}$ is slightly shifted to

2024 JINST 19 P04020

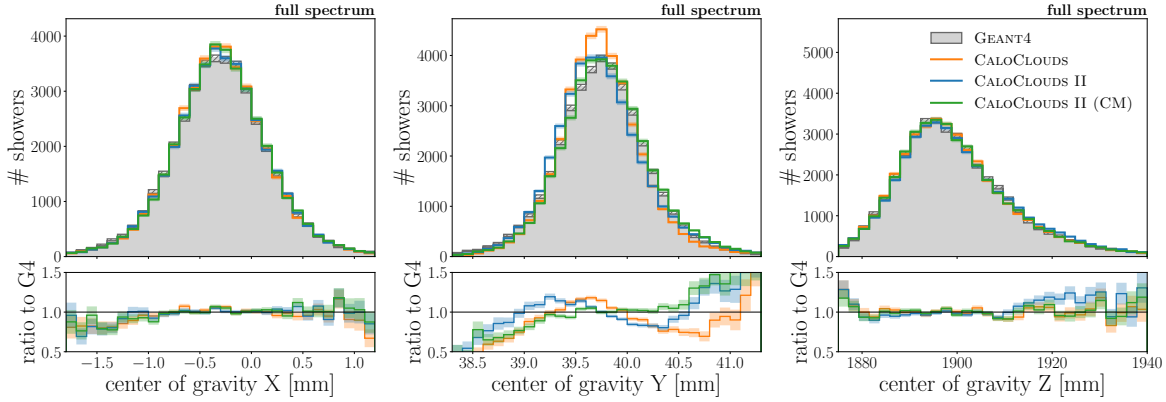


Figure 4. Position of the center of gravity of showers along the X (left), Y (center), and Z (right) directions. All distributions are calculated for 40,000 showers with a uniform distribution of incident particle energies between 10 and 90 GeV. The error band corresponds to the statistical uncertainty in each bin.

lower center of gravity values for all models with the CALOCLOUDS distribution additionally being marginally too narrow. The centers of gravity in X and Y behave slightly different as a magnetic field is simulated in the Y-direction and the active sensors are staggered in the Y-direction while they are all aligned in the X-direction. Due to the number of hits and energy per layer calibrations applied, the distribution of $m_{1,Z}$ is very well modelled. Only in the region around 1925 mm is the CALOCLOUDS II model slightly worse than the other two models. Overall, the three models are reasonably close to the GEANT4 simulation in all six observables.

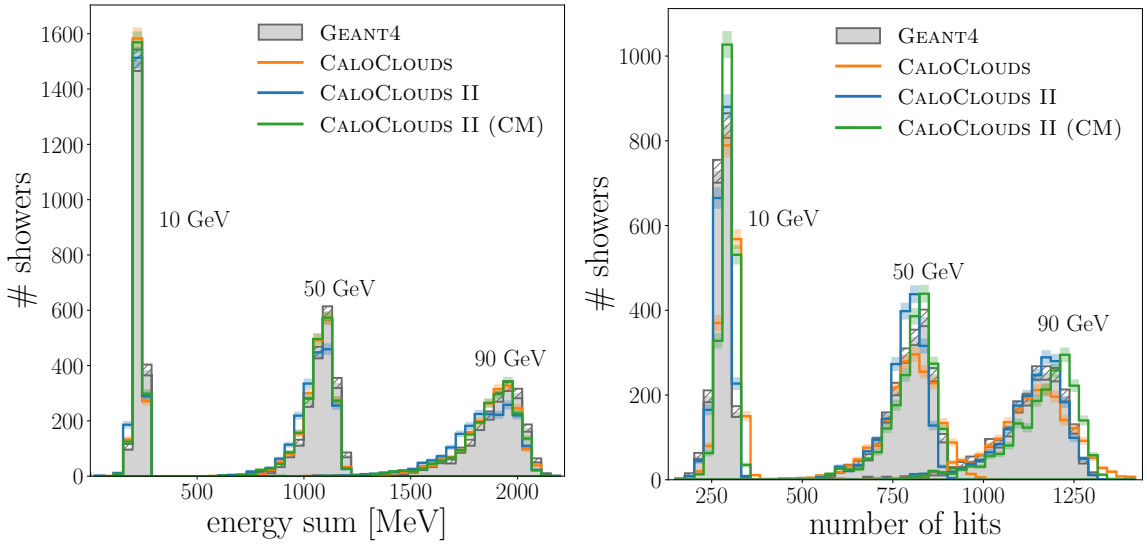


Figure 5. Visible energy sum (left) and the number of hits (right) distributions, for 10, 50, and 90 GeV showers. For each energy and model, 2,000 showers are shown. The error band corresponds to the statistical uncertainty in each bin.

Next, we investigate the models' fidelity for single incident photon energies of 10, 50, and 90 GeV. In figure 5 we show the distributions of the total visible energy (left) and the distributions of the number of hits (cells with deposited energy above the half MIP threshold) for the three single energy datasets

2024 JINST 19 P04020

of 2k showers each. The total energy is well represented by all three models. The number of hits on the other hand is one of the most difficult distributions to represent well with a point cloud generative model. Here high fidelity is still achieved by applying the number of points calibration discussed in section 3. Overall the CaloClouds distributions are slightly too wide as was observed already in ref. [40]. In comparison, CaloClouds II represents the shape of the distribution better, yet in particular for 10 and 90 GeV showers the mean is a bit too large for the CaloClouds II (CM) generated events. This is explainable due to the nature of the polynomial fit used for the number of points calibration. The fit does not perform very well at the edges of the incident energy space. It is known that extrapolation is difficult for generative models, therefore we conjecture that with a training set including lower and higher energies, the fidelity at 10 and 90 GeV would approach the performance at 50 GeV. Overall the CaloClouds II models perform very well and are comparable in fidelity to the CaloClouds model.

4.2 Evaluation Scores

We next investigate the performance of all three CaloClouds models by calculating scores from the high level calorimeter shower observables considered in the previous section. This allows us to put a number on the fidelity observed in plots presented in the previous section 4.2 and not only rely on comparing distributions by eye.

The following observables are considered in order to calculate the one-dimensional scores: the number of hits (cells with energy depositions above the half MIP threshold) N_{hits}, the sampling fraction (the ratio of the visible energy deposited in the calorimeter to the incident photon energy) $E_{\text{vis}}/E_{\text{inc}}$, the cell energy E_{cell} , the center of gravity in the X-, Y-, and Z-directions $m_{1,i\in\{X,Y,Z\}}$, and ten observables each for the longitudinal energy $E_{\text{long},i\in[1,10]}$ and for the radial energy $E_{\text{radial},i\in[1,10]}$. The ten observables for the longitudinal (radial) energy depositions are computed with the energy clustered in consecutive layers (concentric regions) such that on average all 10 observables $E_{\text{long},i\in[1,10]}$ and $E_{\text{radial},i\in[1,10]}$ are computed with the same statistics. Further details on these in total 20 observables can be found in appendix A.

To compare the distributions of these observables between Geant4 and the three generative models, we calculate the 1-Wasserstein distance W_1 — also known as the earth movers distance — between each pair of distributions. The advantages of the Wasserstein distance are that it is an unbinned estimator, for one-dimensional distributions it is computationally efficient to calculate, and no hyperparameter choices have to be made apart from the number of events used for comparison.

Following earlier works using Wasserstein distance based model evaluation scores to compare generative models [55, 58], we calculate the distance between observables calculated from 50k Geant4 and 50k model generated showers. This is done 10× for independent uniformly distributed samples and we report the mean and standard deviation of the scores in table 1. For this purpose, we simulated 500k Geant4 samples and generated 500k showers with each CaloClouds model. To have all scores in a similar order of magnitude, we standardize each observable before we calculate the W_1 score. For the layer energy and radial energy scores, $W_1^{E_{\text{long}}}$ and $W_1^{E_{\text{radial}}}$, we report the average Wasserstein distance over all ten bins. The hit energy score $W_1^{E_{\text{cell}}}$ is calculated for 50k cell hits. In addition to the generative model scores, we also calculate the scores for Geant4 itself, comparing 50k Geant4 showers to a separate set of 50k Geant4 showers.

As can be seen in table 1, most model scores are quite close together. We observe a few outliers, i.e. in the sampling fraction score $W_1^{E_{\text{vis}}/E_{\text{inc}}}$ the CaloClouds and CaloClouds II models are much better CaloClouds II model and in the radial energy score $W_1^{E_{\text{radial}}}$ the CaloClouds II models outperform

2024 JINST 19 P04020

CaloClouds, which is in line with the histogram shown in figure 3. Overall, CaloClouds II (CM) appears to produce higher fidelity showers than the other two models, since it has the best score in four of the scores and does not exhibit any large outliers compared to the other two models. However, as can also be seen in the histograms in section 4.1, none of the scores — with the exception of $W_1^{m_{1,Z}}$ — quite reaches the fidelity of the Geant4 truth itself. Hence we conclude that while all three models generate high fidelity ECAL showers, they should be further improved to match Geant4 exactly in the future.

Table 1. Model performance comparison with 1-Wasserstein distance based scores for various standardized shower observables. The values presented are the mean and standard deviation of 10 calculated scores comparing 50k Geant4 and 50k generated showers.

Simulator	$W_1^{N_{\text{hits}}}$ ($\times 10^{-3}$)	$W_1^{E_{\text{vis}}/E_{\text{inc}}}$ ($\times 10^{-3}$)	$W_1^{E_{\text{cell}}}$ ($\times 10^{-3}$)	$W_1^{E_{\text{long}}}$ ($\times 10^{-3}$)	$W_1^{E_{\text{radial}}}$ ($\times 10^{-3}$)	$W_1^{m_{1,X}}$ ($\times 10^{-3}$)	$W_1^{m_{1,Y}}$ ($\times 10^{-3}$)	$W_1^{m_{1,Z}}$ ($\times 10^{-3}$)
Geant4	0.7 ± 0.2	0.8 ± 0.2	0.9 ± 0.4	0.7 ± 0.8	0.7 ± 0.1	0.9 ± 0.1	1.1 ± 0.3	0.9 ± 0.3
CaloClouds	$\mathbf{2.5 \pm 0.3}$	11.4 ± 0.4	15.9 ± 0.7	$\mathbf{2.0 \pm 1.3}$	38.8 ± 1.4	4.0 ± 0.4	8.7 ± 0.3	1.4 ± 0.5
CaloClouds II	3.6 ± 0.5	26.4 ± 0.4	$\mathbf{15.3 \pm 0.6}$	3.7 ± 1.6	11.6 ± 1.5	$\mathbf{2.4 \pm 0.4}$	$\mathbf{7.6 \pm 0.2}$	3.9 ± 0.4
CaloClouds II (CM)	6.1 ± 0.7	$\mathbf{9.8 \pm 0.5}$	16.0 ± 0.7	$\mathbf{2.0 \pm 1.4}$	$\mathbf{8.3 \pm 1.9}$	3.0 ± 0.4	9.5 ± 0.6	$\mathbf{1.2 \pm 0.5}$

As a side note, the Wasserstein distance can be heavily impacted by outliers in the distributions. Therefore it does not always correlate well with the distribution shape observed in histograms. However, the scores complement the visual inspections of histograms and distributions shown in section 4.1 well.

While useful for comparing generative architectures, 1-Wasserstein distances only consider each dimension of the problem individually. Of course, a successful generative model should also accurately describe higher order correlations. We investigate this in the next section.

4.3 Classifier scores

We further compare the model generated showers to the Geant4 simulation by training a fully connected high-level classifier using the shower observables discussed in the previous section 4.2 to distinguish between model generated and Geant4 simulated showers. The 25 input shower observables are the ten radial and longitudinal energy observables, as well as the three center of gravity variables and the number of hits and total visible energy. For the datasets, we use 500k Geant4 showers and 500k showers generated by each generative model. A 80%, 10%, 10% data split is applied, resulting in a training set of 800k showers and a validation and test set with 100k showers each.

The classifier is implemented as a fully connected neural network with three layers (containing 32, 16, 8 nodes respectively) with LeakyReLU [89] activation functions, and one output node with a Sigmoid activation. It is trained with the Adam optimizer [88] for 10 epochs for each dataset using a binary cross-entropy loss. The final model epoch is chosen based on the lowest validation loss.

To evaluate the classifier we use the area under the receiver operating characteristic curve (AUC) score calculated on the test set. This kind of *classifier score* is also used in other publications evaluating generative models in high energy physics such as ref. [24, 27, 28, 30, 33, 58, 90]. In case the classifier can perfectly separate the Geant4 and model generated datasets, it will result in an AUC = 1.0. For a generated dataset that is indistinguishable from Geant4 simulation, we expect a confused classifier with an AUC = 0.5. Values in between are difficult to interpret in absolute terms, but can give a rough indication of how well the generative models are performing compared to each other. Note that its already not trivial to implement a generative model that achieves AUC values below 1.0.

2024 JINST 19 P04020

We trained the classifier ten times with a different train/ test/ validation data split each time. In table 2 we present the mean AUC and standard deviation of these ten classifier trainings. The CaloClouds generated dataset performs the worst, leading to an almost perfect classification with AUC = 0.999. The two CaloClouds II variants both have a better score clearly separated from an AUC = 1.0. With an AUC = 0.923 the CaloClouds II (CM) model performs slighly better than the CaloClouds II model. For most events, both models result in a separability from the Geant4 simulated showers, but constitute a clear improvement over the baseline CaloClouds implementation. The better performance of the CaloClouds II variants is likely due to the improved radial energy distribution, as we observed a rather large deviation in the $W_1^{E_{\text{radial}}}$ score and in the radial distributions in figure 6.

Table 2. Model performance comparison with area under the receiver operating characteristic curve (AUC) score.

Simulator	AUC
CaloClouds	0.999 ± 0.001
CaloClouds II	0.928 ± 0.001
CaloClouds II (CM)	**0.923 ± 0.001**

4.4 Timing

In this section, we benchmark the average time to produce a single calorimeter shower with the three models considered and investigate the speed-up over the baseline Geant4 simulation. The timing results are presented in table 3.

On both a single CPU and on an NVIDIA® A100 GPU we generated 25× 2,000 showers with the same uniform energy distribution between 10 and 90 GeV. We report the mean and standard deviation of generating these showers. In particular the timing on a single CPU is interesting for current applications of generative models in high energy physics, as CPUs are much more widely available than GPUs and the current computing infrastructure relies on simulations run on CPUs. Further, the

Table 3. Comparison of the computational performance of CaloClouds, CaloClouds II, and CaloClouds II (CM) to the baseline Geant4 simulator on a single core of an Intel® Xeon® CPU E5-2640 v4 (CPU) and on an NVIDIA® A100 with 40 GB of memory (GPU). 2,000 showers were generated with incident energy uniformly distributed between 10 and 90 GeV. Values presented are the means and standard deviations over 10 runs. The number of function evaluations (NFE) indicate the number of diffusion model passes.

Hardware	Simulator	NFE	Batch Size	Time / Shower [ms]	Speed-up
CPU	Geant4			3914.80 ± 74.09	×1
	CaloClouds	100	1	3146.71 ± 31.66	×1.2
	CaloClouds II	25	1	651.68 ± 4.21	×6.0
	CaloClouds II (CM)	1	1	84.35 ± 0.22	×46
GPU	CaloClouds	100	64	24.91 ± 0.72	×157
	CaloClouds II	25	64	6.12 ± 0.13	×640
	CaloClouds II (CM)	1	64	2.09 ± 0.13	×1873

2024 JINST 19 P04020

single CPU timing facilitates a direct comparison to the GEANT4 simulation. Here CALOCLOUDS already yields a speed up of 1.2×, but with less sampling steps CALOCLOUDS II achieves a speed up of 6.0×. However, when implementing the consistency distillation, we achieve a speed up of 46× with the CALOCLOUDS II (CM) model even surpassing previous generative models on the same kind of dataset such as the BIB-AE [20] by about a factor 5.

On an NVIDIA® A100 GPU the CALOCLOUDS model achieves a speed up of 157×, CALOCLOUDS II achieves 640×, and CALOCLOUDS II (CM) achieves 1873× speed up over the baseline GEANT4 simulation on a single CPU. Note that GEANT4 is currently not compatible with GPUs and that GPUs are significantly more expensive than CPUs.

For reference, the training of the CALOCLOUDS model on similar NVIDIA® A100 GPU hardware took around 80 hours for 800k iterations with a batch size of 128, while training of the CALOCLOUDS II model took around 50 hours for 2 million iterations with the same batch size. The consistency distillation for 1 million iterations with a batch size of 256 took about 100 hours.

The speed up between CALOCLOUDS and CALOCLOUDS II is the result of a combination of the improved diffusion paradigm requiring a reduced number of function evaluations as well as the removal of the latent flow. The speed up due to the consistency model in CALOCLOUDS II (CM) yields another large factor, since only a single model evaluation is performed. Both models would be slightly slower when applied in conjunction with the Latent Flow of the CALOCLOUDS model as one evaluation of the Latent Flow is about 50% slower than a single evaluation of the PointWise Net. For a large number of model passes of the PointWise Net in the diffusion framework, the efficiency of the Latent Flow is negligible. However when we consider CALOCLOUDS II (CM) with a single model pass, the application of the Latent Flow would have a noticeable impact on computational performance. Therefore, we removed the Latent Flow in favour of model efficiency as we did not see any improvement in generative fidelity when using it in the CALOCLOUDS II framework.

5 Conclusions

CALOCLOUDS was the first generative model to achieve high-fidelity highly-granular photon calorimeter showers in the form of point clouds with a number of points of $O(1000)$. Due to their sparsity, describing calorimeter showers as point clouds is computationally more efficient than describing them with fixed data structures, i.e. 3D images. Additionally, as the point clouds are based on clustered GEANT4 steps, they allow for a translation-invariant and geometry-independent shower representation. Such cell-geometry-independent models could be easily adapted for fast simulations of calorimeters with non-square cell geometries, i.e. hexagonal cells as used in the envisioned CMS HGCAL [64].

With CALOCLOUDS II we introduce a more streamlined version of CALOCLOUDS utilizing the advanced diffusion paradigm from ref. [68]. It allows for sampling with less model evaluations and for distillation into a consistency model. Using the consistency model in CALOCLOUDS II (CM), generation with a single model evaluation is possible and results in a greatly improved computational efficiency and a speed up of 46× over GEANT4 on a single CPU. This single event CPU performance is particularly promising for introducing a generative model into existing GEANT4-based simulation pipelines. As opposed to other diffusion distillation methods like progressive distillation, consistency distillation only requires a single training to distill the diffusion model in CALOCLOUDS II into a single step generative model, further emphasising the computational advantage of the models presented here. To our knowledge, this constitutes the first application of a consistency model to calorimeter data.

2024 JINST 19 P04020

We compare all three point cloud generative models using one-dimensional distributions and a classifier-based measure and find comparable performance with a slight advantage for the CaloClouds II variants. In particular, the CaloClouds II (CM) model exhibits superior performance while being significantly more computationally efficient. It is counter-intuitive, that a distilled consistency model outperforms the original diffusion model, however, it is known that ODE solvers might introduce errors in earlier denoising steps that are then propagated to the generated samples [68]. The consistency model avoids this since we use it for single-shot generation. Yet, slight deviations from the Geant4 simulations are still visible in various shower observables. Further improvements could likely be achieved by investigating more complex architectures for the diffusion model such as fast transformer implementations [91], equivariant point cloud (EPiC) layers [59], or cross-attention [92].

During the completion of this manuscript, another EDM diffusion based model with subsequent consistency distillation was shown to achieve good fidelity when generating particle jets in the form of point clouds with up to 150 points [63]. While technically a similar approach, in our case the consistency model does not lose generative fidelity compared to the diffusion model and we demonstrate the generation of two orders of magnitude more points (6000 vs 150).

In conclusion, the CaloClouds II model generates high fidelity electromagnetic showers when benchmarked on various shower observables against the baseline Geant4 simulation. In combination with consistency distillation the CaloClouds II (CM) model yields an accurate simulator, which is significantly faster than Geant4 on identical hardware. This constitutes an important step towards the integration of point-cloud based generative models in actual simulation workflows.

Acknowledgments

We thank Dirk Krücker for the valuable comments on the manuscript. This research was supported in part by the Maxwell computational resources operated at Deutsches Elektronen-Synchrotron DESY, Hamburg, Germany. This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under Grant Agreement No 101004761. We acknowledge support by the Deutsche Forschungsgemeinschaft under Germany's Excellence Strategy — EXC 2121 Quantum Universe — 390833306 and via the KISS consortium (05D23GU4, 13D22CH5) funded by the German Federal Ministry of Education and Research BMBF in the ErUM-Data action plan. E.B. is partially funded by a scholarship from the Friedrich Naumann Foundation for Freedom. A.K. has received support from the Helmholtz Initiative and Networking Fund's initiative for refugees as a refugee of the war in Ukraine.

A Radial and longitudinal energy observables

To explore the radial and longitudinal energy profile shown in figure 3 further and to calculate the evaluation scores in section 4.2, we define ten radial and longitudinal energy observables for the calorimeter showers.

Respectively, the ten observables are defined such that energy is clustered in each observable with an equal amount of statistics. Put differently, the energy is binned in ten quantiles with approximately the same number of cell hits in each quantile. The energy bins are defined by the quantiles calculated on the Geant4 test set with 40,000 events. While the bin edges are precisely defined for the radial energy, we round the bin edges of the longitudinal observables to the nearest layer integer number.

Histograms of the radial energy observables $E_{\text{radial},i\in[1,10]}$ are shown in figure 6 and of longitudinal energy observables $E_{\text{long},i\in[1,10]}$ in figure 7. The bin edges for all observables are given in table 4.

2024 JINST 19 P04020

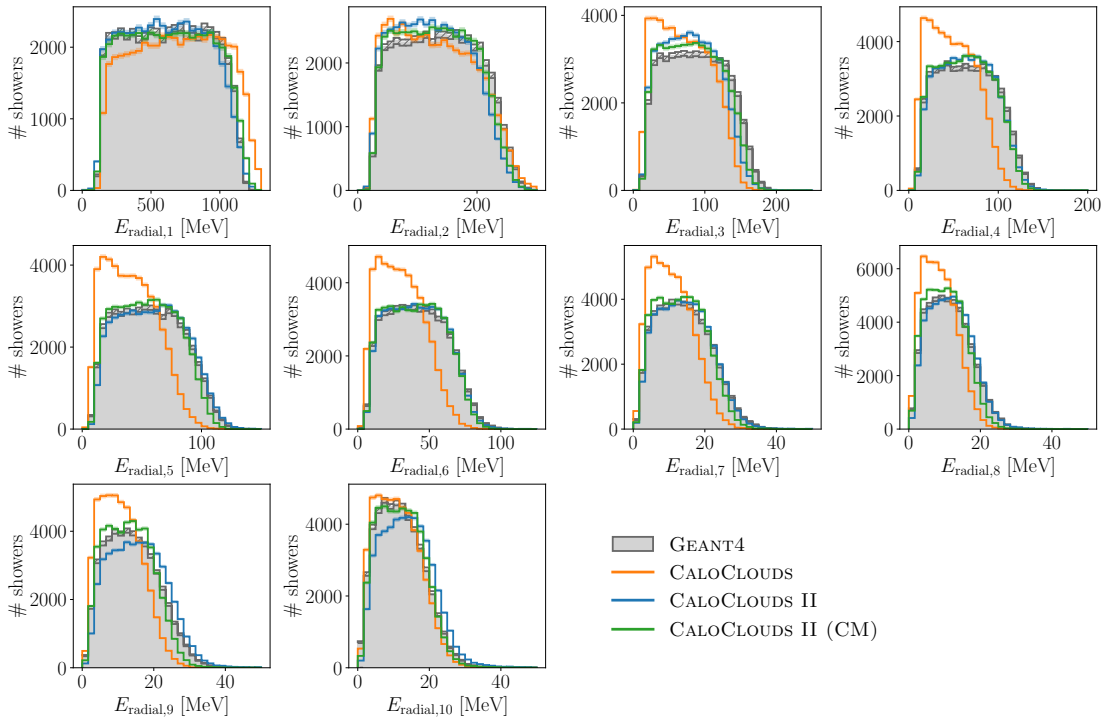


Figure 6. Radial energy observables for 50,000 showers. The error band corresponds to the statistical uncertainty in each bin.

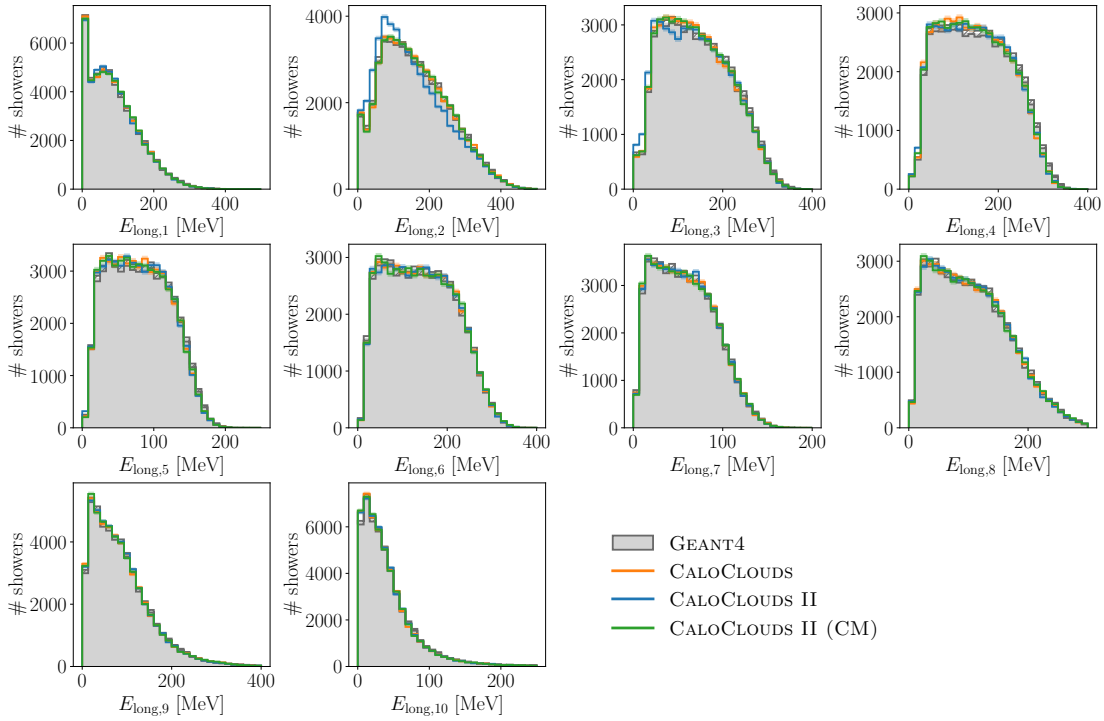


Figure 7. Longitudinal energy observables for 50,000 showers. The error band corresponds to the statistical uncertainty in each bin.

2024 JINST 19 P04020

Table 4. Bin edges for calculating the radial and longitudinal energy observables $E_{\text{radial},i\in[1,10]}$ and $E_{\text{long},i\in[1,10]}$. Determined for ten quantiles each including approximately the same number of cell hits. All bins are half-open, except the last bin.

Bin edges	0	1	2	3	4	5	6	7	8	9	10
Edges for $E_{\text{radial},i\in[1,10]}$ [mm]	0	6.6	9.8	13.0	17.0	23.4	33.6	40.1	48.5	68.8	300
Edges for $E_{\text{long},i\in[1,10]}$ [layer]	1	9	12	14	16	17	19	20	22	25	30

References

[1] I. Zurbano Fernandez et al., *High-Luminosity Large Hadron Collider (HL-LHC): Technical design report*, Tech. Rep. CERN-2020-010 CERN, Geneva, Switzerland (2020) [DOI:10.23731/CYRM-2020-0010].

[2] T. Behnke et al., *The International Linear Collider Technical Design Report - Volume 1: Executive Summary*, arXiv:1306.6327.

[3] HEP Software Foundation collaboration, *A Roadmap for HEP Software and Computing R&D for the 2020s*, *Comput. Softw. Big Sci.* **3** (2019) 7 [arXiv:1712.06982].

[4] A. Boehnlein et al., *HL-LHC Software and Computing Review Panel, 2nd Report*, Tech. Rep. CERN-LHCC-2022-007, CERN, Geneva (2022).

[5] M. Paganini, L. de Oliveira and B. Nachman, *Accelerating Science with Generative Adversarial Networks: An Application to 3D Particle Showers in Multilayer Calorimeters*, *Phys. Rev. Lett.* **120** (2018) 042003 [arXiv:1705.02355].

[6] A. Butter et al., *GANplifying event samples*, *SciPost Phys.* **10** (2021) 139 [arXiv:2008.06545].

[7] S. Bieringer et al., *Calomplification — the power of generative calorimeter models*, 2022 *JINST* **17** P09028 [arXiv:2202.07352].

[8] A. Adelmann et al., *New directions for surrogate models and differentiable programming for High Energy Physics detector simulation*, in the proceedings of the *Snowmass 2021*, Seattle, WA, U.S.A., 17–26 July 2022 [arXiv:2203.08806].

[9] M. Paganini, L. de Oliveira and B. Nachman, *CaloGAN: Simulating 3D high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks*, *Phys. Rev. D* **97** (2018) 014021 [arXiv:1712.10321].

[10] L. de Oliveira, M. Paganini and B. Nachman, *Controlling Physical Attributes in GAN-Accelerated Simulation of Electromagnetic Calorimeters*, *J. Phys. Conf. Ser.* **1085** (2018) 042017 [arXiv:1711.08813].

[11] M. Erdmann, L. Geiger, J. Glombitza and D. Schmidt, *Generating and refining particle detector simulations using the Wasserstein distance in adversarial networks*, *Comput. Softw. Big Sci.* **2** (2018) 4 [arXiv:1802.03325].

[12] M. Erdmann, J. Glombitza and T. Quast, *Precise simulation of electromagnetic calorimeter showers using a Wasserstein Generative Adversarial Network*, *Comput. Softw. Big Sci.* **3** (2019) 4 [arXiv:1807.01954].

[13] D. Belayneh et al., *Calorimetry with deep learning: particle simulation and reconstruction for collider physics*, *Eur. Phys. J. C* **80** (2020) 688 [arXiv:1912.06794].

[14] ATLAS collaboration, *Fast simulation of the ATLAS calorimeter system with Generative Adversarial Networks*, Tech. Rep. ATL-SOFT-PUB-2020-006, CERN, Geneva (2020).

2024 JINST 19 P04020

[15] F. Carminati et al., *Three dimensional Generative Adversarial Networks for fast simulation*, *J. Phys. Conf. Ser.* **1085** (2018) 032016.

[16] P. Musella and F. Pandolfi, *Fast and Accurate Simulation of Particle Detectors Using Generative Adversarial Networks*, *Comput. Softw. Big Sci.* **2** (2018) 8 [arXiv:1805.00850].

[17] ATLAS collaboration, *Deep generative models for fast shower simulation in ATLAS*, Tech. Rep. ATL-SOFT-PUB-2018-001, CERN, Geneva (2018).

[18] ATLAS collaboration, *AtlFast3: The Next Generation of Fast Simulation in ATLAS*, *Comput. Softw. Big Sci.* **6** (2022) 7 [arXiv:2109.02551].

[19] H. Hashemi et al., *Ultra-High-Resolution Detector Simulation with Intra-Event Aware GAN and Self-Supervised Relational Reasoning*, arXiv:2303.08046.

[20] E. Buhmann et al., *Getting High: High Fidelity Simulation of High Granularity Calorimeters with High Speed*, *Comput. Softw. Big Sci.* **5** (2021) 13 [arXiv:2005.05334].

[21] E. Buhmann et al., *Decoding Photons: Physics in the Latent Space of a BIB-AE Generative Network*, *EPJ Web Conf.* **251** (2021) 03003 [arXiv:2102.12491].

[22] E. Buhmann et al., *Hadrons, better, faster, stronger*, *Mach. Learn. Sci. Tech.* **3** (2022) 025014 [arXiv:2112.09709].

[23] ATLAS collaboration, *Deep generative models for fast photon shower simulation in ATLAS*, arXiv:2210.06204.

[24] J.C. Cresswell et al., *CaloMan: Fast generation of calorimeter showers with density estimation on learned manifolds*, in the proceedings of the *36th Conference on Neural Information Processing Systems: Workshop on Machine Learning and the Physical Sciences*, New Orleans, LO, U.S.A., 3 December 2022 [arXiv:2211.15380].

[25] S. Diefenbacher et al., *New angles on fast calorimeter shower simulation*, *Mach. Learn. Sci. Tech.* **4** (2023) 035044 [arXiv:2303.18150].

[26] C. Chen et al., *Analysis-Specific Fast Simulation at the LHC with Deep Learning*, *Comput. Softw. Big Sci.* **5** (2021) 15.

[27] C. Krause and D. Shih, *Fast and accurate simulations of calorimeter showers with normalizing flows*, *Phys. Rev. D* **107** (2023) 113003 [arXiv:2106.05285].

[28] C. Krause and D. Shih, *Accelerating accurate simulations of calorimeter showers with normalizing flows and probability density distillation*, *Phys. Rev. D* **107** (2023) 113004 [arXiv:2110.11377].

[29] S. Schnake, D. Krücker and K. Borras, *Generating Calorimeter Showers as Point Clouds*, in the proceedings of the *Machine Learning and the Physical Sciences Workshop*, New Orleans, LO, U.S.A., 3 December 2022.

[30] C. Krause, I. Pang and D. Shih, *CaloFlow for CaloChallenge Dataset 1*, arXiv:2210.14245.

[31] S. Diefenbacher et al., *L2LFlows: generating high-fidelity 3D calorimeter images*, 2023 *JINST* **18** P10017 [arXiv:2302.11594].

[32] A. Xu, S. Han, X. Ju and H. Wang, *Generative machine learning for detector response modeling with a conditional normalizing flow*, 2024 *JINST* **19** P02003 [arXiv:2303.10148].

[33] M.R. Buckley, C. Krause, I. Pang and D. Shih, *Inductive simulation of calorimeter showers with normalizing flows*, *Phys. Rev. D* **109** (2024) 033006 [arXiv:2305.11934].

[34] J. Sohl-Dickstein, E.A. Weiss, N. Maheswaranathan and S. Ganguli, *Deep Unsupervised Learning using Nonequilibrium Thermodynamics*, arXiv:1503.03585.

2024 JINST 19 P04020

[35] Y. Song and S. Ermon, *Generative Modeling by Estimating Gradients of the Data Distribution*, arXiv:1907.05600.

[36] Y. Song and S. Ermon, *Improved Techniques for Training Score-Based Generative Models*, arXiv:2006.09011.

[37] J. Ho, A. Jain and P. Abbeel, *Denoising Diffusion Probabilistic Models*, arXiv:2006.11239.

[38] Y. Song et al., *Score-Based Generative Modeling through Stochastic Differential Equations*, arXiv:2011.13456.

[39] V. Mikuni and B. Nachman, *Score-based generative models for calorimeter shower simulation*, *Phys. Rev. D* **106** (2022) 092009 [arXiv:2206.11898].

[40] E. Buhmann et al., *CaloClouds: fast geometry-independent highly-granular calorimeter simulation*, 2023 *JINST* **18** P11025 [arXiv:2305.04847].

[41] F.T. Acosta et al., *Comparison of Point Cloud and Image-based Models for Calorimeter Fast Simulation*, arXiv:2307.04780.

[42] V. Mikuni and B. Nachman, *CaloScore v2: single-shot calorimeter shower simulation with diffusion models*, 2024 *JINST* **19** P02001 [arXiv:2308.03847].

[43] O. Amram and K. Pedro, *Denoising diffusion models with geometry adaptation for high fidelity calorimeter simulation*, *Phys. Rev. D* **108** (2023) 072014 [arXiv:2308.03876].

[44] A. Butter, T. Plehn and R. Winterhalder, *How to GAN LHC Events*, *SciPost Phys.* **7** (2019) 075 [arXiv:1907.03764].

[45] B. Hashemi et al., *LHC analysis-specific datasets with Generative Adversarial Networks*, arXiv:1901.05282.

[46] R. Di Sipio, M. Faucci Giannelli, S. Ketabchi Haghighat and S. Palazzo, *DijetGAN: A Generative-Adversarial Network Approach for the Simulation of QCD Dijet Events at the LHC*, *JHEP* **08** (2019) 110 [arXiv:1903.02433].

[47] J. Arjona Martínez et al., *Particle Generative Adversarial Networks for full-event simulation at the LHC and their application to pileup description*, *J. Phys. Conf. Ser.* **1525** (2020) 012081 [arXiv:1912.02748].

[48] Y. Alanazi et al., *Simulation of electron-proton scattering events by a Feature-Augmented and Transformed Generative Adversarial Network (FAT-GAN)*, arXiv:2001.11103 [DOI:10.24963/ijcai.2021/293].

[49] S. Otten et al., *Event Generation and Statistical Sampling for Physics with Deep Generative Models and a Density Information Buffer*, *Nature Commun.* **12** (2021) 2985 [arXiv:1901.00875].

[50] A. Butter et al., *Generative networks for precision enthusiasts*, *SciPost Phys.* **14** (2023) 078 [arXiv:2110.13632].

[51] L. de Oliveira, M. Paganini and B. Nachman, *Learning Particle Physics by Example: Location-Aware Generative Adversarial Networks for Physics Synthesis*, *Comput. Softw. Big Sci.* **1** (2017) 4 [arXiv:1701.05927].

[52] A. Andreassen, I. Feige, C. Frye and M.D. Schwartz, *JUNIPR: a Framework for Unsupervised Machine Learning in Particle Physics*, *Eur. Phys. J. C* **79** (2019) 102 [arXiv:1804.09720].

[53] E. Bothmann and L. Debbio, *Reweighting a parton shower using a neural network: the final-state case*, *JHEP* **01** (2019) 033 [arXiv:1808.07802].

[54] K. Dohi, *Variational Autoencoders for Jet Simulation*, arXiv:2009.04842.

2024 JINST 19 P04020

[55] R. Kansal et al., *Particle Cloud Generation with Message Passing Generative Adversarial Networks*, in the proceedings of the *35th Conference on Neural Information Processing Systems*, Online Conference, Canada, 6–14 December 2021 [arXiv:2106.11535].

[56] B. Käch et al., *JetFlow: Generating Jets with Conditioned and Mass Constrained Normalising Flows*, arXiv:2211.13630.

[57] B. Käch, D. Krücker and I. Melzer-Pellmann, *Point Cloud Generation using Transformer Encoders and Normalising Flows*, arXiv:2211.13623.

[58] R. Kansal et al., *Evaluating generative models in high energy physics*, *Phys. Rev. D* **107** (2023) 076017 [arXiv:2211.10295].

[59] E. Buhmann, G. Kasieczka and J. Thaler, *EPiC-GAN: Equivariant point cloud generation for particle jets*, *SciPost Phys.* **15** (2023) 130 [arXiv:2301.08128].

[60] M. Leigh et al., *PC-JeDi: Diffusion for Particle Cloud Generation in High Energy Physics*, *SciPost Phys.* **16** (2024) 018 [arXiv:2303.05376].

[61] B. Käch and I. Melzer-Pellmann, *Attention to Mean-Fields for Particle Cloud Generation*, arXiv:2305.15254.

[62] A. Butter et al., *Jet Diffusion versus JetGPT – Modern Networks for the LHC*, arXiv:2305.10475.

[63] M. Leigh et al., *Faster diffusion model with improved quality for particle cloud generation*, *Phys. Rev. D* **109** (2024) 012010 [arXiv:2307.06836].

[64] CMS collaboration, *The Phase-2 Upgrade of the CMS Endcap Calorimeter*, Tech. Rep. CERN-LHCC-2017-023 CERN, Geneva, Switzerland (2017) [DOI:10.17181/CERN.IV8M.1JY2].

[65] J. Liu et al., *Generalizing to new geometries with Geometry-Aware Autoregressive Models (GAAMs) for fast calorimeter simulation*, 2023 *JINST* **18** P11003 [arXiv:2305.11531].

[66] V. Mikuni, B. Nachman and M. Pettee, *Fast point cloud generation with diffusion models in high energy physics*, *Phys. Rev. D* **108** (2023) 036025 [arXiv:2304.01266].

[67] Q. Zhang and Y. Chen, *Fast Sampling of Diffusion Models with Exponential Integrator*, arXiv:2204.13902.

[68] T. Karras, M. Aittala, T. Aila and S. Laine, *Elucidating the Design Space of Diffusion-Based Generative Models*, arXiv:2206.00364.

[69] C. Lu et al., *DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps*, arXiv:2206.00927.

[70] C. Lu et al., *DPM-Solver++: Fast Solver for Guided Sampling of Diffusion Probabilistic Models*, arXiv:2211.01095.

[71] E. Luhman and T. Luhman, *Knowledge Distillation in Iterative Generative Models for Improved Sampling Speed*, arXiv:2101.02388.

[72] T. Salimans and J. Ho, *Progressive Distillation for Fast Sampling of Diffusion Models*, arXiv:2202.00512.

[73] X. Liu, C. Gong and Q. Liu, *Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow*, arXiv:2209.03003.

[74] H. Zheng et al., *Fast Sampling of Diffusion Models via Operator Learning*, arXiv:2211.13449.

[75] D. Berthelot et al., *TRACT: Denoising Diffusion Models with Transitive Closure Time-Distillation*, arXiv:2303.04248.

2024 JINST 19 P04020

[76] Y. Song, P. Dhariwal, M. Chen and I. Sutskever, *Consistency Models*, arXiv:2303.01469.

[77] ILD Concept Group, *International Large Detector: Interim Design Report*, arXiv:2003.01116.

[78] iLCSoft Group, *iLCSoft Project Page*, https://ilcsoft.desy.de/portal.

[79] M. Frank, F. Gaede, C. Grefe and P. Mato, *DD4hep: A Detector Description Toolkit for High Energy Physics Experiments*, *J. Phys. Conf. Ser.* **513** (2014) 022010.

[80] D.P. Kingma et al., *Improving Variational Inference with Inverse Autoregressive Flow*, arXiv:1606.04934.

[81] D.P. Kingma and M. Welling, *Auto-Encoding Variational Bayes*, arXiv:1312.6114.

[82] A. Paszke et al., *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, arXiv:1912.01703.

[83] L. Dinh, J. Sohl-Dickstein and S. Bengio, *Density estimation using Real NVP*, arXiv:1605.08803.

[84] C. Durkan, A. Bekasov, I. Murray and G. Papamakarios, *Neural Spline Flows*, arXiv:1906.04032.

[85] E. Bingham et al., *Pyro: Deep Universal Probabilistic Programming*, *J. Mach. Learn. Res.* **20** (2019) 28 [arXiv:1810.09538].

[86] S. Luo and W. Hu, *Diffusion Probabilistic Models for 3D Point Cloud Generation*, arXiv:2103.01458.

[87] J. Song, C. Meng and S. Ermon, *Denoising Diffusion Implicit Models*, arXiv:2010.02502.

[88] D.P. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization*, in the proceedings of the *3rd International Conference on Learning Representations*, San Diego, CA, U.S.A., 7–9 May 2015 [arXiv:1412.6980].

[89] A.L. Maas, A.Y. Hannun, A.Y. Ng et al., *Rectifier nonlinearities improve neural network ac6–21 June oustic models*, in the proceedings of the *30th International Conference on Machine Learning*, Atlanta, GA, U.S.A., 2013.

[90] R. Das et al., *How to Understand Limitations of Generative Networks*, *SciPost Phys.* **16** (2024) 031 [arXiv:2305.16774].

[91] T. Dao et al., *FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness*, arXiv:2205.14135.

[92] A. Vaswani et al., *Attention Is All You Need*, in the proceedings of the *31st International Conference on Neural Information Processing Systems*, Long Beach, CA, U.S.A., 4–9 December 2017 [arXiv:1706.03762].

2024 JINST 19 P04020