Title

CaloScore v2: single-shot calorimeter shower simulation with diffusion models

Permalink

https://escholarship.org/uc/item/4f2364vx

Journal

Journal of Instrumentation, 19(02)

ISSN

1748-0221

Authors

Mikuni, Vinicius
Nachman, Benjamin

Publication Date

2024-02-01

DOI

10.1088/1748-0221/19/02/p02001

CaloScore v2: Single-shot Calorimeter Shower Simulation with Diffusion Models

Vinicius Mikuni[1, *] and Benjamin Nachman[2, 3, †]

[1] *National Energy Research Scientific Computing Center, Berkeley Lab, Berkeley, CA 94720, USA*
[2] *Physics Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA*
[3] *Berkeley Institute for Data Science, University of California, Berkeley, CA 94720, USA*

Diffusion generative models are promising alternatives for fast surrogate models, producing high-fidelity physics simulations. However, the generation time often requires an expensive denoising process with hundreds of function evaluations, restricting the current applicability of these models in a realistic setting. In this work, we report updates on the CaloScore architecture, detailing the changes in the diffusion process, which produces higher quality samples, and the use of progressive distillation, resulting in a diffusion model capable of generating new samples with a single function evaluation. We demonstrate these improvements using the Calorimeter Simulation Challenge 2022 dataset.

I. INTRODUCTION

Deep generative models are a disruptive technology, enhancing many aspects of every day life and basic science research. In high energy physics, calorimeter simulations have been a benchmark for new deep generative models since their first application. Detailed physics simulations of particle showers in calorimeters are often prohibitively slow due to the large number of secondary particles produced as the primary particle is stopped inside the detector material. Bespoke and often proprietary fast simulations have been developed for many cases, but they are usually derived using low-dimensional heuristics. Deep learning has the potential to match the quality of detailed simulations in their full high-dimensional representation while also matching the speed of classical fast simulations. Automated, high-fidelity and fast calorimeter simulations can enhance the science of existing detectors and catalyze the development of better detectors at future experiments.

The application of deep generative models to calorimeter simulation began with CaloGAN [2, 3]. Since that time, Generative Adversarial Networks (GANs) [1] [2–17], Variational Autoencoders [18] [15, 16, 19–21], Normalizing Flows (NFs) [22] [23–29], and Diffusion Models [30] [31–33] have been applied to this problem. This research entered a precision era with the first NF application (CaloFlow) [23], which showed that even a post-hoc classifier had difficulty distinguishing physics from machine learning simulators. A number of related innovations in the CaloFlow series are motivating for our work including teacher-student training [24] and factorizing into energy/layer and shape/layer. Deep generative models are also now being used in practice. The ATLAS experiment has integrated a GAN into its fast simulation, which has improved the modeling of hadronic final states [17]. Simulations from ATLAS, combined with additional samples from more granular hypothetical detectors, form the CaloChallenge [34], a data challenge to compare a diverse set of models on the same calorimeter simulations. The score-based model CaloScore [31] was the first model deployed on all three datasets from the CaloChallenge. Since that time, NF-based approaches have also been studied for CaloChallenge datasets 1 [26, 28] and 2-3 [25]. As the performance of these approaches has not been quantified with exactly the same metric[1], it is hard to know which is 'best', but it is clear that they are all able to accurately describe various aspects of the complex calorimeter showers.

Since the publication of CaloScore, we have improved the performance significantly by introducing a number of innovations. Collectively, these updates constitute CaloScore v2, which represents the state of the art in calorimeter emulation. Improvements to the architecture and training procedure result in a model that has significantly better fidelity and is much faster than the original CaloScore. One aspect of CaloScore v2 is progressive distillation to reduce the number of timesteps in the diffusion process, with one step already achieving reasonable fidelity. Additionally, we modify the diffusion process to decrease the loss variance during training, and separate the task of determining the total energy deposition with the voxel generation through an additional generative model. Altogether, CaloScore v2 is essentially a new model built on the foundation of the original CaloScore – where we demonstrated that diffusion models are a compromise between flexibility (easy for GANs, hard for NFs) and robustness (easy for NFs, hard for GANs) – with qualitatively superior performance than its predecessor.

This paper is organized as follows. Section II introduces Diffusion Models and Sec. III describes how we sample from a trained model. The three CaloChallenge datasets are detailed in Sec. IV. The properties of CaloScore v2 are provided in Sec. V before present-

[*] vmikuni@lbl.gov
[†] bpnachman@lbl.gov

[1] A forthcoming CaloChallenge review paper will do this carefully, also including many methods that have not (yet) been published as standalone papers.

ing numerical results in Sec. VI. The paper ends with conclusions and outlook in Sec. VII.

II. DIFFUSION MODELS

Diffusion generative models apply perturbations to the data to slowly corrupt the initial dataset into a tractable noise distribution. The generation step aims to reverse this processes, starting from a noise distribution that is denoised towards realistic examples of the data to be generated. The time-dependent perturbation can be described by the following stochastic differential equation (SDE):

$$\mathrm{d}\mathbf{x} = f(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{dw}. \tag{1}$$

In this equation, the data $\mathbf{x} \in \mathbb{R}^d$ are perturbed over a time parameter $t \in [0, 1]$ with perturbation parameters defined by the choice of drift and diffusion coefficients $f(\mathbf{x}, t) \in \mathbb{R}^d$ and $g(t) \in \mathbb{R}$, respectively. The stochastic term is identified by the Wiener process, or Brownian motion, $w(t) \in \mathbb{R}^d$, often sampled from a normal distribution with the same dimension as the data. To reverse this processes towards the generation of new data, the reverse stochastic differential equation needs to be solved, described by

$$\mathrm{d}\mathbf{x} = [f(\mathbf{x}, t) - g(t)^2\nabla_x \log p(\mathbf{x})]\mathrm{d}t + g(t)\mathrm{d}\bar{\mathrm{w}}. \tag{2}$$

While the forward SDE is easy to solve, the reverse process requires the knowledge of the term $\nabla_x \log p(\mathbf{x})$, also known as the score function of the data. Since $\mathbf{x}$ is high-dimensional, the probability density of the data $p(\mathbf{x})$ is often intractable, and similarly, the score function cannot be easily estimated. Alternatively, the authors in [35] have shown that, in the limit of small noise perturbations, learning the score function of perturbed data is equivalent to learning the score function of the data itself. This observation motivates the loss function

$$\mathcal{L} = \frac{1}{2}\mathbb{E}_{\mathbf{x}_t,t}\left[\lambda(t)\left\|s_\theta(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t}\log q(\mathbf{x}_t|\mathbf{x})\right\|_2^2\right]. \tag{3}$$

The neural network $s_\theta(\mathbf{x}_t, t)$ with trainable parameters θ takes as input data $\mathbf{x}_t$ that has been perturbed at time t. The weight parameter $\lambda(t)$ is a positive function used to determine the importance of each term in the loss function over time. By considering Gaussian perturbation $q(\mathbf{x}_t|\mathbf{x}) = \mathcal{N}(\mathbf{x}_t, \alpha_t\mathbf{x}, \sigma_t^2\mathbf{I})$, the score function of the perturbed data $\mathbf{x}_t = \alpha_t\mathbf{x} + \sigma_t\epsilon$, $\epsilon \sim \mathcal{N}(0, 1)^d$ is identified as:

$$\nabla_{\mathbf{x}_t}\log q(\mathbf{x}_t|x) = \frac{\mathbf{x} - \mathbf{x}_t}{\sigma_t^2} \sim \frac{\mathcal{N}(0, 1)^d}{\sigma_t}. \tag{4}$$

In the original CaloScore implementation, $\lambda(t) \equiv \sigma_t^2$, which improves the training stability by removing the σ-dependence of the perturbed score function in Eq. 4. While the direct prediction of the score function is beneficial, recent works have moved towards learning different representations of Eq. 3. The reason for this change is explained by the high variance of the signal-to-noise-ratio (SNR) distribution α_t/σ_t. At the beginning the diffusion process, at time values near zero, the standard deviation of the perturbation is designed to be small, leading to large values of SNR. Conversely, at time values near one, the perturbation is the largest to ensure that any prior data distribution is diffused towards a normal distribution at the end of the diffusion process, leading to small values of SNR. Since we expect $\sigma_t s_\theta(\mathbf{x}_t, t) \sim \mathcal{N}(0, 1)^d$, the expected values of $s_\theta(\mathbf{x}_t, t)$ also show high variance, requiring $s_\theta(\mathbf{x}_t, t)$ to spam a wide range of values. In CaloScore v2, we instead opt for a so-called velocity implementation, introduced in [36] that defines a target $\mathbf{v}_t \equiv \alpha_t\epsilon - \sigma_t\mathbf{x}$ which modifies the loss function in Eq. 3 to introduce the updated loss as

$$\mathcal{L} = \mathbb{E}_{\mathbf{x}_t,t}\left\|\mathbf{v}_t - \mathbf{v}_\theta(\mathbf{x}_t, t)\right\|^2, \tag{5}$$

with a neural network trained to learn directly the velocity parameter while taking as inputs the time and the perturbed data. From this implementation, we can still identify the approximation of the score function of the perturbed data as

$$s_\theta(\mathbf{x}_t, t) = \mathbf{x}_t - \frac{\alpha_t}{\sigma_t}\mathbf{v}_\theta(\mathbf{x}_t, t), \tag{6}$$

with the advantage of having the velocity parameter with similar range over the entire time interval of the diffusion process.

The choice of the drift and diffusion coefficients $f(\mathbf{x}, t)$ and $g(t)$ are also important parameters of the diffusion process. In CaloScore, different choices of parameters were investigated yielding similar performance. The parameters of the perturbation α and σ can also be used to define $f(\mathbf{x}, t)$ and $g(t)$ with

$$\begin{aligned} f(\mathbf{x}, t) &= \frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\mathbf{x}_t \\ g^2(t) &= \frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t} - 2\frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\sigma_t^2. \end{aligned} \tag{7}$$

For CaloScore v2, we choose to focus on the variance preserving (VP) implementation which additionally requires $\sigma_t^2 + \alpha_t^2 = 1$. A cosine schedule is used with $\alpha_t = \cos(0.5\pi t)$ and $\sigma_t = \sin(0.5\pi t)$. This choice is in contrast with the previous β-parameterization used in CaloScore , where $f(\mathbf{x}, t) = -\frac{1}{2}\beta(t)\mathbf{x}$ and $g(t) = \sqrt{\beta(t)}$ with $\beta(t) = \beta_{\min} + t\left(\beta_{\max} - \beta_{\min}\right)$ with $\beta_{\min} = 0.1$ and $\beta_{\max} = 20$. This update is motivated by the use of the progressive distillation method, explained in further detail in Section III.

III. SAMPLE GENERATION

With the approximation of the score function of the data, different methods an be employed to generate new observations. Stochastic solvers can be used to solve

Eq. 2. In CaloScore, sampling is performed using the Euler-Maruyama algorithm [37] followed by an additional corrector step that uses the Langevin MCMC approach [38, 39] to increase the sampling quality. This approach, however requires the discretization over time of the reverse SDE in Eq. 2 to be of $\mathcal{O}(100)$ to generate high fidelity calorimeter images. The large number of discretization steps is a natural consequence of the stochastic nature of the equation to be solved, since the precision in this case is determined by the magnitude of the stochastic noise added in each step. On the other hand, the reverse SDE admits a deterministic solution [35] of the form

$$\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = f(\mathbf{x}, t) - \frac{1}{2}g(t)^2\nabla_x \log q(\mathbf{x}_t)\,, \tag{8}$$

which can be solved with fewer time steps, while also providing an unique mapping between points of the initial noise distribution and the generated data. While Eq. 8 can be solved as is with direct integration, the authors of Ref. [40] propose a different deterministic sampler named DDIM, also shown to represent an integration rule for Eq. 8, but requiring fewer time steps to achieve the same level of precision. In the DDIM solver, the update rule is then specified by:

$$\mathbf{x}_s = \alpha_s \mathbf{x}_\theta(\mathbf{x}_t, t) + \sigma_s \frac{\mathbf{x}_t - \alpha_t \mathbf{x}_\theta(\mathbf{x}_t, t)}{\sigma_t}, \tag{9}$$

for time $s < t$ and position prediction $\mathbf{x}_\theta(\mathbf{x}_t, t) = \alpha_t \mathbf{x}_t - \sigma_t \mathbf{v}_\theta(\mathbf{x}_t, t)$. Additionally, the choice of the DDIM sampler is also motivated by the use of progressive distillation [36]. The idea of progressive distillation is to introduce a second model whose task is to learn to halve the number of time steps required by the DDIM solver using a trained diffusion model as a guide. In this approach, the trained diffusion model ("teacher") is used to initialize a "student" model. During training, the goal of the student model is to denoise data $\mathbf{x}_t$ towards a target $\widetilde{\mathbf{x}}_t$. The difference is that $\widetilde{\mathbf{x}}_t$ does not represent the clean data $(\mathbf{x})$, but is instead one that makes a single student DDIM step to match two teacher DDIM steps. After the student model is trained, generation can be performed using half the number of time steps compared to the teacher model. This process is then repeated multiple times, with the student at the end of each iteration becoming the new teacher. In CaloScore v2, the initial diffusion model uses 512 time steps to ensure precision and is distilled multiple times with results using 8 time steps and a single time step reported.

IV. FAST CALORIMETER SIMULATION CHALLENGE 2022

The performance of CaloScore v2 is evaluated using the datasets released for the Fast Calorimeter Simulation Challenge 2022 [17, 41–43]. Three datasets are provided, representing calorimeter shower simulations with Geant4 [44] of different detector geometries and number of detector components. Dataset 1 [42] is based on the ATLAS open dataset [17, 45] and is similar to the current ATLAS detector calorimeter geometry. While samples consisting of both photons and pions are provided, we evaluate our model using only the photon dataset. The voxelization procedure is defined such that it reduces the amount of empty voxels, while maintaining high fidelity compared to the full simulation. The downside of this approach is that the geometrical information present in the original detector layout is also reduced since each voxel now covers a different area depending on the number of detector components merged during the voxelization. A total of 368 voxels are then left to describe the full detector slice. Photon energies are provided in this configuration for 15 incident energies ranging from 256 MeV up to 4 TeV in steps given by powers of two. For each generated energy, 10k samples are provided with this number reduced at higher energies due to long simulation times, resulting in a total of 121k used during training.

Datasets 2 [43] and 3 [41] contain each 100k examples to be used for training and are simulated using a common detector layout but with different voxelization granularity. The detector simulated has a concentric cylinder geometry with 45 layers, where each layer consists of active (silicon) and passive (tungsten) material, simulated with Geant4. Simulations for electrons are generated at the detector surface with initial energy sampled from a log-uniform distribution ranging from 1 GeV to 1 TeV. In dataset 2, each layer consists of 144 readout cells, with 9 in the radial and 16 in the angular directions. Dataset 3 is more granular, consisting of 900 readout cells in each layer, with 18 in the radial and 50 in the angular directions with a total of 6480 and 40500 voxels, respectively.

Since the initial representation of the datasets 2 and 3 are given in cylindrical coordinates, a preprocessing step was used in CaloScore to convert the datasets to Cartesian coordinates. This choice avoids the need for the generative model to learn the periodic boundary conditions in the α direction, while also centralizing the detector readouts. Unfortunately, this transformation is not reversible, since multiple voxels can be mapped to a single voxel in the new Cartesian representation[2]. This choice limited the comparison of CaloScore with other generative models and has now been abandoned.

Similarly, CaloScore v2 uses a different data preprocessing to improve the fidelity of the generative model compared to CaloScore. In the original CaloScore, each voxel energy E_v is normalized by the value of the energy of the incident particle E_0 times a factor f used to fix the energy scale normalization caused by the sampling fraction of the detector and ensure the normal-

[2] A one-to-one assignment between the two sets of coordinates is possible, but requires the distance interval in Cartesian coordinates to follow a non-linear function since the transformation of coordinates is itself non-linear.

ized voxel energy $E'_v = \frac{E_v}{fE_0}$ lies between 0 and 1. In CaloScore v2, we instead split the generation into two tasks: one that generates the overall deposited energy per layer of the calorimeter, and one that generates normalized voxel distributions. For the second task, the preprocessing is changed, with the normalization factor for each voxel to be the deposited energy per layer instead of the incident particle energy. For the first task, to determine the overall energy per layer, we first divide the deposited energy per layer by the initial particle energy, multiplied by the factor f. The additional preprocessing steps applied to the data are then identical to CaloScore. The normalized energy depositions are then transformed to logit-space, similarly to the strategy used in CaloFlow. The log-transformed value u_v is defined as:

$$u_v = \log\left(\frac{x}{1-x}\right), x = \alpha + (1-2\alpha)E'_v. \tag{10}$$

The value α in Eq. 10 is set to 10^{-6} and avoids a discontinuity when $E'_v = 0$. The generated particle energy, used as a conditional input to the model, is also transformed before training. The transformed conditional energy u_0 is defined as:

$$u_0 = \frac{e_0 - e_{\text{min}}}{e_{\text{max}} - e_{\text{min}}}, \tag{11}$$

where e_{min} and e_{max} are the minimum and maximum energies available in the dataset used for the training. Last, all voxels and energy depositions per layer are standardized to have mean zero and unit variance across all training samples.

V. MODEL ARCHITECTURE AND TRAINING DETAILS

In the previous CaloScore implementation, the transformation to Cartesian coordinates resulted in a model that could be efficiently learned using few convolutional layers with large kernel sizes, implemented with a U-net [46] network architecture. In CaloScore v2, we employ a similar U-net architecture, but include additional attention layers. More specifically, datasets 2 and 3 have the number of spatial components in each dimension reduced by a factor 2 every other convolutional layer (resulting in a factor $2 \times 2 \times 2 = 8$ reduction) with fixed kernel size set to 3. This process is repeated 3 times, with lowest dimensional representation reduced by a factor 512 compared to the initial number of voxels. The 3D convolution operations use 32, 64, and 96 hidden nodes with swish [47] activation function. The attention layer is only used at the lowest dimensional representation, with data patches determined by the flattened array describing the data at the lowest dimensionality. The upsampling section of the architecture is a mirrored version, with dimensions increased by a factor 8 every other layer. Skip connections between the downsampling and upsampling sides of the architecture are combined with a concatenation operation, completing the architecture. Conditional information consisting of the time information, incident particle energy, and deposited energy per layer (in case of the diffusion model trained to generate normalized voxels), are included through an addition operation after every convolutional layer. A trainable embedding of the conditional features is created by a fully connected layer over the conditional inputs. The output size is fixed to match the expected output size of the convolutional layers. For dataset 1, the strategy is similar. The number of voxels to be simulated are reduced by a factor 2 every other layer, with this process repeated 4 times and overall reduction of factor 16 compared to the initial size. The number of hidden nodes for the 1D convolutional layers is then chosen to be 16, 32, 64, and 96 for each fixed dimensionality. Since this dataset is smaller compared to datasets 2 and 3, attention layers are used in all lower dimensional representations of the initial data.

A second diffusion model is introduced in CaloScore v2, tasked to learn only the energy deposition per layer. The model used to train the diffusion model is based on the ResNet [48] architecture, consisting of multiple fully connected layers with additional skip connections. The number of ResNet layers is set to 3 in both datasets with 128 hidden nodes in dataset 1 and 1024 in datasets 2 and 3. Additional choices of hyperparameters such as overall number of layers and hidden node sizes were tested and did not yield noticeable improvements.

The training is carried out using the Perlmutter supercomputer interfaced with the Horovod package [49] for distributed training. 16 NVIDIA A100 GPUs are used simultaneously during training, while a single GPU is used for evaluation and timing comparison. All models are trained for up to 250 epochs with a cosine learning rate schedule [50] with initial learning rate of 16×10^{-4}. If the loss function does not decrease for 30 consecutive epochs, evaluated in a separate testing set representing 20% to the sample size, the training is stopped. The implementation of all models is carried out using Keras backend [51] with TensorFlow [52].

VI. RESULTS

We evaluate the performance of CaloScore v2 using the metrics available for the evaluation of the Fast Calorimeter Simulation Challenge 2022, as well as additional studies to quantify the agreement of different physics distributions with the original Geant simulations. Since dataset 3 before distillation is much slower than datasets 1 and 2, only distilled results with 8 and a single time step are reported.

Distributions of the total energy deposition and number of calorimeter hits are presented in Fig. 1. A hit both in the Geant simulation and from generated samples is defined by any energy deposition above 0.1 keV in dataset 1 and 15.1 kev in datasets 2 and 3.

We observe a good agreement between generated samples from CaloScore v2 compared to the full simulation. In particular, CaloScore v2 after the distillation to a single time step is still able to retain precision. In dataset 1, energies are generated in specific energy intervals, leading to discontinuous values of energy depositions. We compare the results for the total deposited energy with between CaloScore and CaloScore v2 using the 1-Wasserstein distance, referred as the Earth mover's distance (EMD), between generated samples and the Geant simulation. Results for the EMD obtained using the total deposited energy are listed in Table. I, with Wasserstein GAN (WGAN-GP) and CaloScore values taken from the best performing models presented in [31].

TABLE I. Comparison of the earth mover's distance calculated using the total deposited energy. Values for CaloScore are selected from the best performing model reported in [31].

Model	EMD		
	dataset 1	dataset 2	dataset 3
CaloScore	1.52	1.8	3.17
WGAN-GP	21.55	5.95	13.29
CaloScore v2	0.21	0.13	-
CaloScore v2 8 steps	0.33	0.15	0.25
CaloScore v2 1 step	0.35	0.19	0.40

We observe a significant improvement compared to CaloScore with EMD values decreasing by a factor 10 compared to previous results. This improvement stems from the independent determination of the energy depositions per layer achieved by the separate diffusion process. Reduced fidelity is observed in the distilled models compared to baseline CaloScore v2. Nevertheless, even the single-shot model still improves upon CaloScore.

Next, we study the mean deposited energy versus r, α, and layer number presented in Fig. 2.

The mean energy as a function of layer number is determined by the independent diffusion model, making it insensitive to the modeling of individual voxels. In contrast, the distributions concerning r and α are sensitive to the modeling of the individual voxels, with agreement within 10% observed in all distillation levels.

Additionally, we investigate the angular distributions of the calorimeter showers in datasets 2 and 3 in terms of the shower width, shown in Fig. 3. The shower width σ_i with $x_i, i \in [1,2]$ representing the r- and α- coordinates is calculated as:

$$\sigma_i = \sqrt{\langle x_i^2 \rangle - \langle x_i \rangle^2}, \tag{12}$$

with energy-weighted mean defined as

$$\langle x_i \rangle = \frac{\sum_j x_{i,j} E_j}{\sum_j E_j}. \tag{13}$$

The agreement of CaloScore v2 is often within 10% compared to the Geant simulation, with exception to the tails of the distribution located in the later layers of the detector with differences more pronounced for the distilled models compared to the baseline CaloScore v2.

We also perform a visual inspection for datasets 2 and 3 using samples generated by CaloScore v2 by looking at the average energy deposition per voxel for 10,000 showers in layers 10 and 44, the layers with the highest and lowest average energy deposition, respectively. Results are shown in Fig. 4.

In layer 10, the majority of the energy deposition is located near $r = 0$ since incident particles are generated at the center and orthogonal to the detector plane. As the electromagnetic shower evolves, the interactions with the detector material result in more energy deposited away from the center, with layer 44 showing the majority of the energy depositions spread over higher values of r. In all cases, the CaloScore v2 samples are able to reproduce the correct trend and do not seem to create any noticeable mismodeling.

Finally, we investigate the energy conditioning of the model by comparing the distributions of the deposited energy and the energy of the incident particle. Results are presented in Fig. 5.

In all cases the CaloScore v2 model is able to correctly reproduce the deposited energy with spread similar to the one observed by the Geant samples.

We also perform the so-called "classifier test" where a binary classifier is trained to distinguish generated samples from the samples produced by Geant, as used in the construction of GANs and proposed as a post-hoc metric in [23]. We use the official classifier and training schedule provided by the challenge to evaluate the results shown in Tables II and III for classifiers trained using either lower level inputs or high level distributions respectively. A similar classifier test was carried out when evaluating the original CaloScore performance, where an AUC of around 98% was observed for all datasets. We should also point out that not only the classifier architecture, number of training epochs, and learning rates were different, but the initial preprocessing to convert datasets 2 and 3 to Cartesian coordinates makes the comparison of this metric deceptive. On the other hand, CaloScore v2 shows much lower values for both AUC and JSD metrics, showing a large improvement compared to the previous model. After distillation, we observe a degradation of the AUC and JSD in all datasets. Nevertheless, even the single-shot model observes AUC values significantly lower than 1 in all datasets. Direct comparisons with other models submitted to the Fast Calorimeter Challenge containing additional metrics will be made available in the forthcoming review paper.

We investigate the generation time required by CaloScore v2 and compare with previous results reported using the same hardware setup in Table. IV. The baseline CaloScore v2 uses 512 time steps, value 5 times bigger than the original CaloScore, leading to slower generation times. On the other hand, the distilled model with 8 time steps and the single-shot

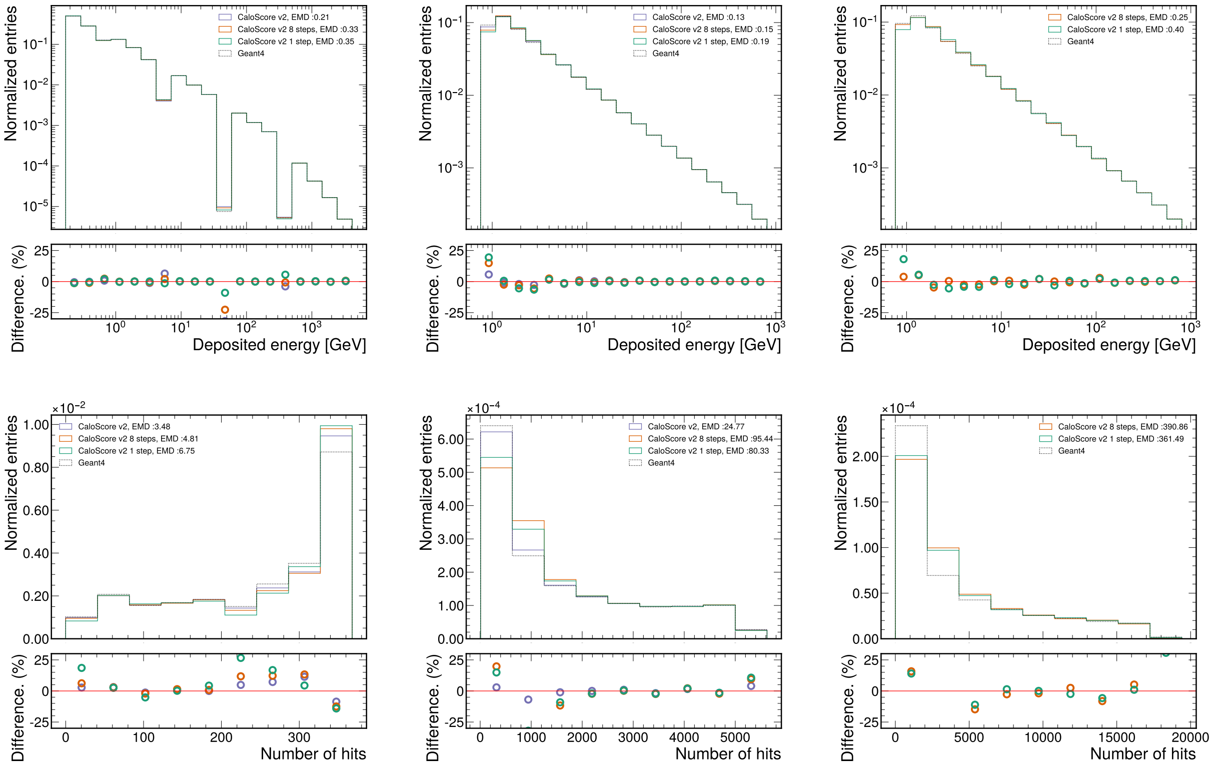

FIG. 1. Comparison of the sum of all voxel energies (top) and number of hits (bottom) for datasets 1 (left), 2 (middle), and 3 (right). The Earth mover's distance (EMD) between each distribution and the Geant distribution is also provided.

TABLE II. Area under the ROC curve (AUC) and Jensen-Shannon divergence (JSD) calculated based on the classifier trained using low level information.

Model	AUC/JSD		
	dataset 1	dataset 2	dataset 3
CaloScore v2	0.758/0.155	0.597/0.023	-
CaloScore v2 8 steps	0.815/0.242	0.709/0.106	0.670/0.075
CaloScore v2 1 step	0.878/0.367	0.755/0.157	0.6974/0.1002

TABLE III. Area under the ROC curve (AUC) and Jensen-Shannon divergence (JSD) calculated based on the classifier trainined using high level information.

Model	AUC/JSD		
	dataset 1	dataset 2	dataset 3
CaloScore v2	0.587/0.047	0.622/0.039	-
CaloScore v2 8 steps	0.6278/0.066	0.833/0.282	0.851/0.310
CaloScore v2 1 step	0.714/0.136	0.846/0.305	0.880/0.376

model decrease significantly the amount of time required, even compared to the initial CaloScore implementation and with a WGAN with similar overall architecture as CaloScore. The reason for the reduction comes from CaloScore v2 utilizing convolutional layers with smaller kernel sizes as opposed to CaloScore. The larger kernel sizes were required to achieve a more precise result, since the transformation to cartesian coordinates increased the data sparsity, leading to a larger fraction of voxels without energy depositions.

TABLE IV. Number of dimensions, trainable parameter, and time to generate 100 new calorimeter showers for each dataset studied in this work. Generation times for Geant are based on the average time required to generate samples over the energy range provided.

Model	Time to 100 showers [s]		
	dataset 1	dataset 2	dataset 3
CaloScore	4.0	5.8	33.4
WGAN-GP	1.3	1.33	2.06
Geant	$\mathcal{O}(10^2 - 10^3)$	$\mathcal{O}(10^4)$	$\mathcal{O}(10^4)$
CaloScore v2	4.0	27.8	73.7
CaloScore v2 8 steps	0.05	0.33	1.71
CaloScore v2 1 step	0.002	0.010	0.011

VII. CONCLUSIONS

In this work we introduced CaloScore v2 as a follow up to CaloScore, a diffusion generative model for calorimeter shower simulation. Compared to its predecessor, CaloScore v2 brings several changes to both increase the fidelity of the simulation and decrease the

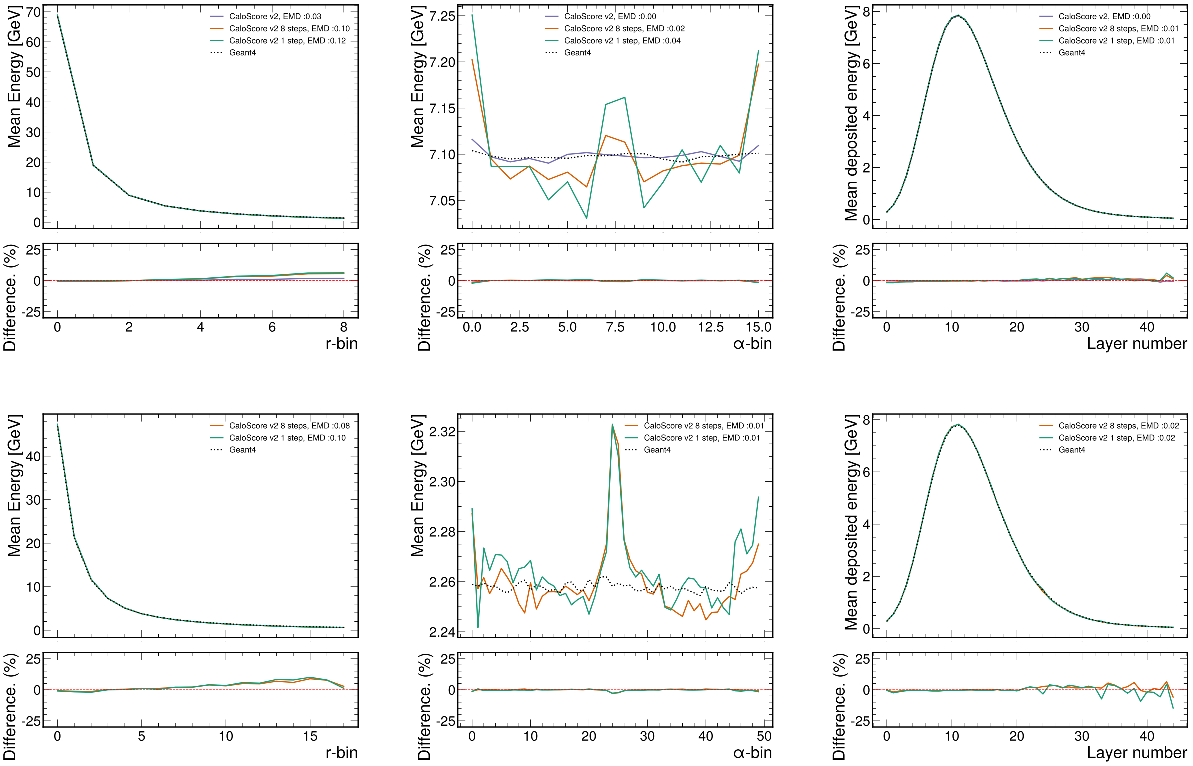

FIG. 2. Comparison of the average deposited energies in the r- (left), α- (middle), and z-coordinates (right) for datasets 2 (top) and 3 (bottom). The Earth mover's distance (EMD) between each distribution and the Geant distribution is also provided.

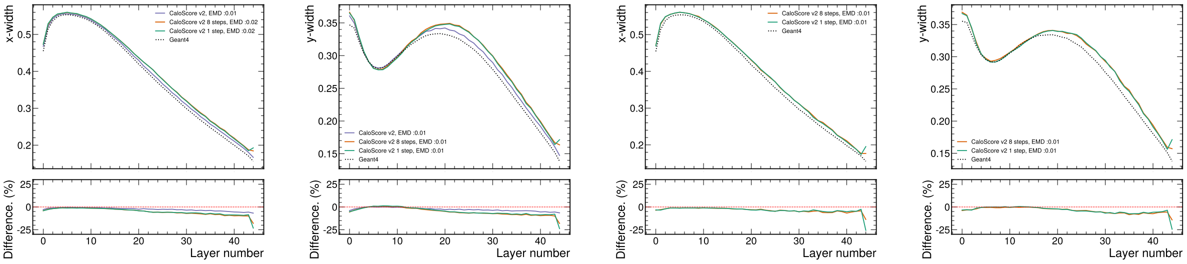

FIG. 3. Comparison of the particle shower width in the r- and α- directions in datasets 2 (first two figures from the left) and 3 (last two figures from the left). The Earth mover's distance (EMD) between each distribution and the Geant distribution is also provided.

time required for the sampling of new observations.

We evaluate the performance of CaloScore v2 using the simulated samples created for the Fast Calorimeter Simulation Challenge 2022. We separate the generation process into two problems: generating the overall energy deposition in each layer of the calorimeter and generating the normalized voxel response. This modification improves the quality of the generated samples with better estimation of the overall energy deposition. Similarly, modifications to the network architecture through the use of attention layers increased the model performance without resulting in slower sampling times. Indeed, a single evaluation of CaloScore v2 is now faster compared to a single evaluation of CaloScore.

The sampling speed has also been reduced compared to CaloScore by a factor 500-2000 through the additional use of progressive distillation, a technique to iteratively reduce the number of time steps required during sampling. With this technique, we are able to reduce the generation to a single time step, resulting in the first single-shot diffusion models for detector simulation in collider physics. While the single-shot diffusion model shows a degradation in fidelity compared to the baseline CaloScore v2, we still observe an

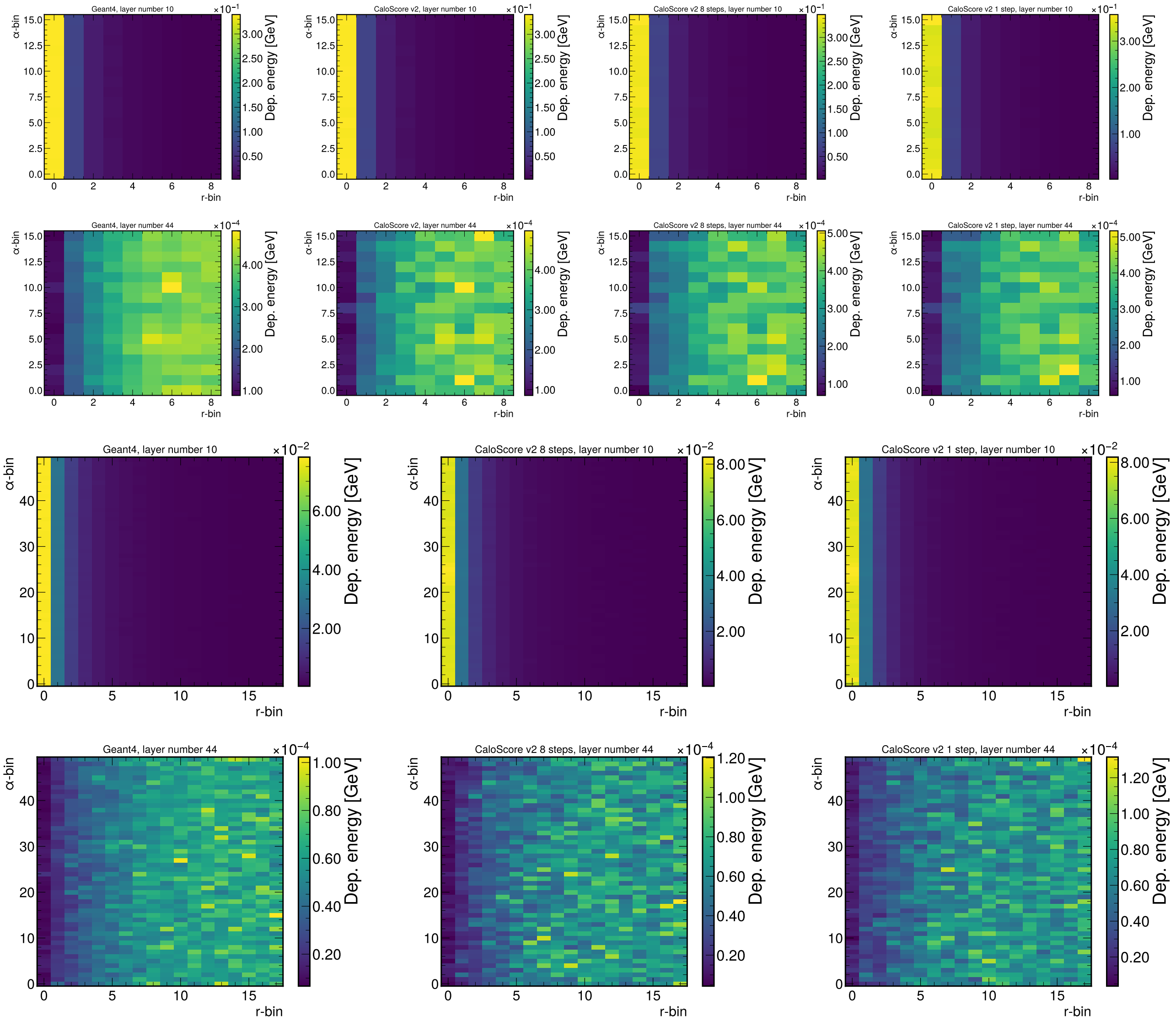

FIG. 4. The 2-dimensional distribution of the mean deposited energy in layers with highest (first and third rows) and lowest (second and fourth rows) mean energy depositions in datasets 2 (first two rows) and 3 (last two rows). Simulated samples from Geant are shown in the first column, compared with CaloScore v2 using different number of sampling time steps.

overall good performance, also evidenced by the classifier test which is not able to distinguish samples from Geant and CaloScore v2 with perfect accuracy.

Finally, progressive distillation shows that single-shot diffusion models can be achieved for fast and high fidelity simulation in collider physics. This observation motivates future work on reducing the performance degradation during the distillation process to retain the same level of precision as the initial diffusion model. Alternatively, smaller model architectures may be able to reduce even further the evaluation time by exploring additional symmetries present in the data.

CODE AVAILABILITY

The code used to produce all results presented in this paper are available at https://github.com/ViniciusMikuni/CaloScoreV2

ACKNOWLEDGMENTS

VM and BN are supported by the U.S. Department of Energy (DOE), Office of Science under contract DE-AC02-05CH11231. This research used resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under

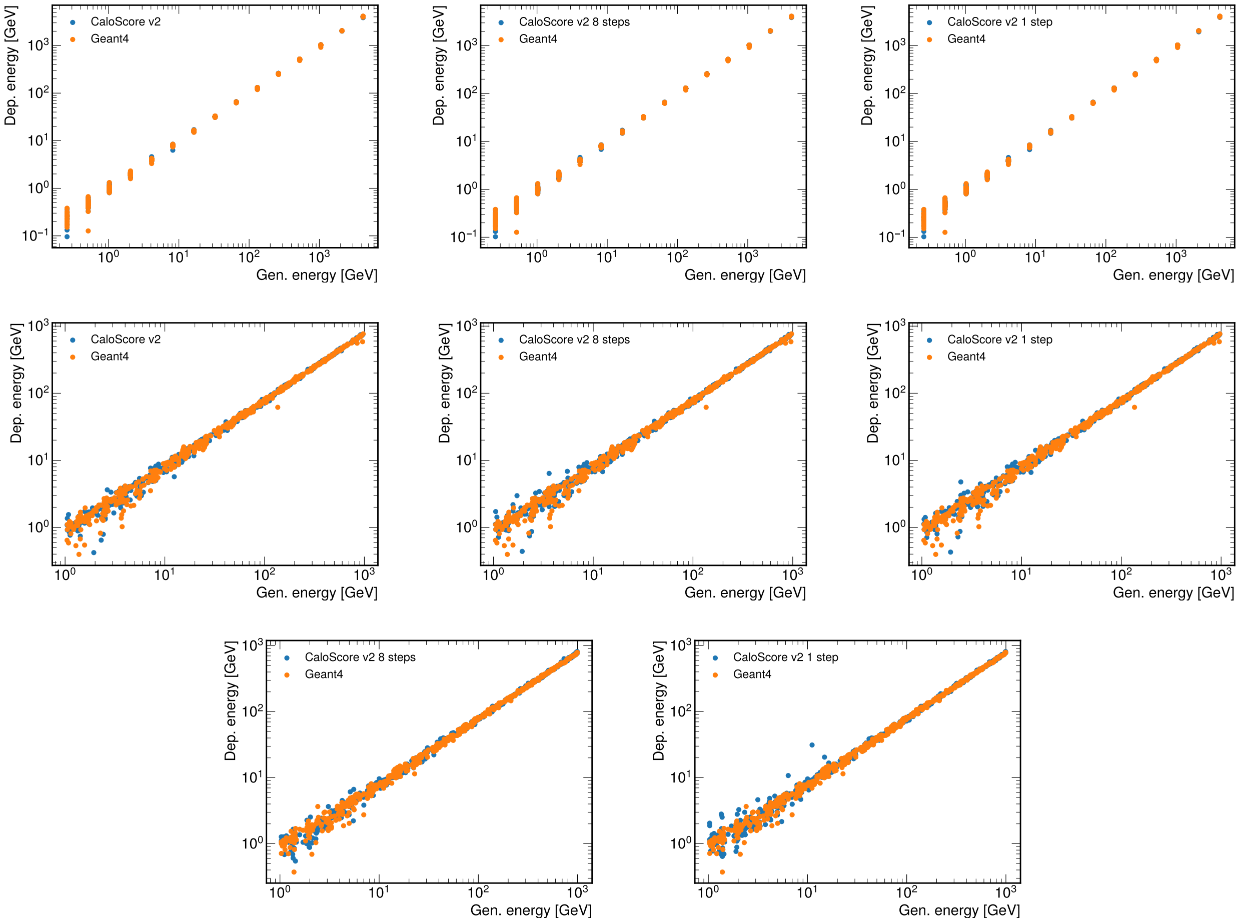

FIG. 5. Deposited energy versus generated energy in Geant (orange) and CaloScore v2 (blue) using different number of sampling steps. First row samples are generated using energies from dataset 1, second row from dataset 2 and third row from dataset 3.

Contract No. DE-AC02-05CH11231 using NERSC award HEP-ERCAP0021099.

[1] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, arXiv e-prints , arXiv:1406.2661 (2014), arXiv:1406.2661 [stat.ML].

[2] M. Paganini, L. de Oliveira, and B. Nachman, Phys. Rev. Lett. **120**, 042003 (2018), arXiv:1705.02355 [hep-ex].

[3] M. Paganini, L. de Oliveira, and B. Nachman, Phys. Rev. D **97**, 014021 (2018), arXiv:1712.10321 [hep-ex].

[4] L. de Oliveira, M. Paganini, and B. Nachman, J. Phys. Conf. Ser. **1085**, 042017 (2018), arXiv:1711.08813 [hep-ex].

[5] M. Erdmann, L. Geiger, J. Glombitza, and D. Schmidt, Comput. Softw. Big Sci. **2**, 4 (2018), arXiv:1802.03325 [astro-ph.IM].

[6] M. Erdmann, J. Glombitza, and T. Quast, Comput. Softw. Big Sci. **3**, 4 (2019), arXiv:1807.01954 [physics.ins-det].

[7] D. Belayneh *et al.*, (2019), 10.1140/epjc/s10052-020-8251-9, arXiv:1912.06794 [physics.ins-det].

[8] S. Vallecorsa, F. Carminati, and G. Khattak, Proceedings, 23rd International Conference on Computing in High Energy and Nuclear Physics (CHEP 2018): Sofia, Bulgaria, July 9-13, 2018 **214**, 02010 (2019).

[9] C. Ahdida *et al.* (SHiP), (2019), 10.1088/1748-0221/14/11/P11028, arXiv:1909.04451 [physics.ins-det].

[10] V. Chekalina, E. Orlova, F. Ratnikov, D. Ulyanov, A. Ustyuzhanin, and E. Zakharov, CHEP 2018 (2018), 10.1051/epjconf/201921402034, arXiv:1812.01319 [physics.data-an].

[11] F. Carminati, A. Gheata, G. Khattak, P. Mendez Lorenzo, S. Sharan, and S. Vallecorsa, Proceedings, 18th International Workshop on Advanced Computing and Analysis Techniques in Physics Research (ACAT 2017): Seattle, WA, USA, August 21-25, 2017 **1085**, 032016 (2018).

[12] S. Vallecorsa, Proceedings, 18th International Workshop on Advanced Computing and Analysis Techniques in Physics Research (ACAT 2017): Seattle, WA, USA, August 21-25, 2017 **1085**, 022005 (2018).

[13] P. Musella and F. Pandolfi, Comput. Softw. Big Sci. **2**, 8 (2018), arXiv:1805.00850 [hep-ex].

[14] K. Deja, T. Trzcinski, and u. Graczykowski, Proceedings, 23rd International Conference on Computing in High Energy and Nuclear Physics (CHEP 2018): Sofia, Bulgaria, July 9-13, 2018 **214**, 06003 (2019).

[15] (2022), arXiv:2210.06204 [hep-ex].

[16] ATL-SOFT-PUB-2018-001 (2018).

[17] G. Aad *et al.* (ATLAS), Comput. Softw. Big Sci. **6**, 7 (2022), arXiv:2109.02551 [hep-ex].

[18] D. P. Kingma and M. Welling, arXiv e-prints , arXiv:1312.6114 (2013), arXiv:1312.6114 [stat.ML].

[19] E. Buhmann, S. Diefenbacher, E. Eren, F. Gaede, G. Kasieczka, A. Korol, and K. Krüger, (2021), arXiv:2102.12491 [physics.ins-det].

[20] E. Buhmann, S. Diefenbacher, E. Eren, F. Gaede, D. Hundhausen, G. Kasieczka, W. Korcari, K. Krüger, P. McKeown, and L. Rustige, (2021), arXiv:2112.09709 [physics.ins-det].

[21] S. Diefenbacher, E. Eren, F. Gaede, G. Kasieczka, A. Korol, K. Krüger, P. McKeown, and L. Rustige, (2023), arXiv:2303.18150 [physics.ins-det].

[22] D. Rezende and S. Mohamed, in *Proceedings of the 32nd International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 37, edited by F. Bach and D. Blei (PMLR, Lille, France, 2015) pp. 1530–1538.

[23] C. Krause and D. Shih, (2021), arXiv:2106.05285 [physics.ins-det].

[24] C. Krause and D. Shih, (2021), arXiv:2110.11377 [physics.ins-det].

[25] M. R. Buckley, C. Krause, I. Pang, and D. Shih, (2023), arXiv:2305.11934 [physics.ins-det].

[26] C. Krause, I. Pang, and D. Shih, (2022), arXiv:2210.14245 [physics.ins-det].

[27] S. Diefenbacher, E. Eren, F. Gaede, G. Kasieczka, C. Krause, I. Shekhzadeh, and D. Shih, (2023), arXiv:2302.11594 [physics.ins-det].

[28] J. C. Cresswell, B. L. Ross, G. Loaiza-Ganem, H. Reyes-Gonzalez, M. Letizia, and A. L. Caterini, in *36th Conference on Neural Information Processing Systems* (2022) arXiv:2211.15380 [hep-ph].

[29] J. Liu, A. Ghosh, D. Smith, P. Baldi, and D. Whiteson, (2023), arXiv:2305.11531 [physics.ins-det].

[30] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, arXiv e-prints , arXiv:2011.13456 (2020), arXiv:2011.13456 [cs.LG].

[31] V. Mikuni and B. Nachman, Phys. Rev. D **106**, 092009 (2022), arXiv:2206.11898 [hep-ph].

[32] E. Buhmann, S. Diefenbacher, E. Eren, F. Gaede, G. Kasieczka, A. Korol, W. Korcari, K. Krüger, and P. McKeown, (2023), arXiv:2305.04847 [physics.ins-det].

[33] F. T. Acosta, V. Mikuni, B. Nachman, M. Arratia, K. Barish, B. Karki, R. Milton, P. Karande, and A. Angerami, (2023), arXiv:2307.04780 [cs.LG].

[34] "Fast calorimeter simulation challenge 2022," .

[35] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, ArXiv **abs/2011.13456** (2021).

[36] T. Salimans and J. Ho, in *International Conference on Learning Representations* (2022).

[37] P. E. Kloeden and E. Platen, in *Numerical Solution of Stochastic Differential Equations* (Springer, 1992) pp. 103–160.

[38] G. Parisi, Nucl. Phys. B **180**, 378 (1981).

[39] U. Grenander and M. I. Miller, Journal of the royal statistical society series b-methodological **56**, 549 (1994).

[40] J. Song, C. Meng, and S. Ermon, CoRR **abs/2010.02502** (2020), 2010.02502.

[41] M. Faucci Giannelli, G. Kasieczka, C. Krause, B. Nachman, D. Salamani, D. Shih, and A. Zaborowska, "Fast Calorimeter Simulation Challenge 2022 - Dataset 3," (2022).

[42] M. F. Giannelli, G. Kasieczka, C. Krause, B. Nachman, D. Salamani, D. Shih, and A. Zaborowska, "Fast Calorimeter Simulation Challenge 2022 - Dataset 1," (2022).

[43] M. Faucci Giannelli, G. Kasieczka, C. Krause, B. Nachman, D. Salamani, D. Shih, and A. Zaborowska, "Fast Calorimeter Simulation Challenge 2022 - Dataset 2," (2022).

[44] S. Agostinelli *et al.*, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **506**, 250 (2003).

[45] *Fast simulation of the ATLAS calorimeter system with Generative Adversarial Networks*, Tech. Rep. (CERN, Geneva, 2020).

[46] O. Ronneberger, P. Fischer, and T. Brox, in *International Conference on Medical image computing and computer-assisted intervention* (Springer, 2015) pp. 234–241.

[47] P. Ramachandran, B. Zoph, and Q. V. Le, arXiv preprint arXiv:1710.05941 (2017).

[48] K. He, X. Zhang, S. Ren, and J. Sun, in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016) pp. 770–778.

[49] A. Sergeev and M. D. Balso, arXiv preprint arXiv:1802.05799 (2018).

[50] I. Loshchilov and F. Hutter, CoRR **abs/1608.03983** (2016), 1608.03983.

[51] F. Chollet, "Keras," `https://github.com/fchollet/keras` (2017).

[52] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, in *OSDI*, Vol. 16 (2016) pp. 265–283.