Computing and Software for Big Science (2021) 5:13
https://doi.org/10.1007/s41781-021-00056-0

ORIGINAL ARTICLE

Getting High: High Fidelity Simulation of High Granularity Calorimeters with High Speed

Erik Buhmann[1] · Sascha Diefenbacher[1] · Engin Eren[2] · Frank Gaede[2] · Gregor Kasieczka[1] · Anatolii Korol[3] · Katja Krüger[2]

Received: 28 May 2020 / Accepted: 2 March 2021 / Published online: 26 May 2021

Abstract
Accurate simulation of physical processes is crucial for the success of modern particle physics. However, simulating the development and interaction of particle showers with calorimeter detectors is a time consuming process and drives the computing needs of large experiments at the LHC and future colliders. Recently, generative machine learning models based on deep neural networks have shown promise in speeding up this task by several orders of magnitude. We investigate the use of a new architecture—the Bounded Information Bottleneck Autoencoder—for modelling electromagnetic showers in the central region of the Silicon-Tungsten calorimeter of the proposed International Large Detector. Combined with a novel second post-processing network, this approach achieves an accurate simulation of differential distributions including for the first time the shape of the minimum-ionizing-particle peak compared to a full Geant4 simulation for a high-granularity calorimeter with 27k simulated channels. The results are validated by comparing to established architectures. Our results further strengthen the case of using generative networks for fast simulation and demonstrate that physically relevant differential distributions can be described with high accuracy.

Keywords Deep learning · Generative models · Calorimeter · Simulation · High granularity · GAN · WGAN · BIB-AE

Introduction

Precisely measuring nature's fundamental parameters and discovering new elementary particles in modern high energy physics is only made possible by our deep mathematical understanding of the Standard Model and our ability to reliably simulate interactions of these particles with complex detectors. While essential for our scientific progress, the production of these simulations is increasingly costly. This cost is already a potential bottleneck at the LHC, and the problem will be exacerbated by higher luminosity, larger amounts of pile-up and more complex and granular detectors at the high-luminosity LHC and planned future colliders. A promising way to accelerate the simulation is offered by generative machine learning models and was pioneered in Ref. [1]. The present work focuses on simulating a very high-resolution calorimeter prototype with greater fidelity of physically relevant distributions, paving the road for practical applications[1].

Advanced machine learning methods, based on deep neural networks, are rapidly transforming and improving the way to explore the fundamental interactions of nature in particle physics—see for example Ref. [2] for a recent overview of neural network architectures developed to identify hadronically decaying top quarks. However, we are only beginning to explore the potential benefits from unsupervised techniques designed to model the underlying high-dimensional density distribution of data. This allows, e.g., anomaly detection algorithms to identify signals from new physics

✉ Frank Gaede
frank.gaede@desy.de

Sascha Diefenbacher
sascha.daniel.diefenbacher@uni-hamburg.de

Engin Eren
engin.eren@desy.de

[1] Institut für Experimentalphysik, Universität Hamburg, Hamburg, Germany

[2] Deutsches Elektronen-Synchrotron, Hamburg, Germany

[3] Taras Shevchenko National University of Kyiv, Kyiv, Ukraine

[1] Implementations of the network architectures as well as instructions to produce training data are available on https://github.com/FLC-QU-hep/getting_high.

theories without making specific model assumptions [3–12]. Furthermore, once the phase space density is encoded in a neural network, it can be sampled from very efficiently. This makes synthetic models of particle interactions many orders of magnitude faster than classical approaches, where for example for a particle showering in a calorimeter many secondary shower particles have to be created and individually tracked through the material of the detector according to the underlying physics processes.

Calorimeters are a crucial part of experiments in high energy physics, where the incident primary particles create showers of secondary particles in dense materials that are used to measure the energy. In sandwich calorimeters, layers of dense materials are interleaved with sensitive layers recording energy depositions from secondary shower particles mostly from ionization. The details of the shower development via creation of secondary particles as well as their energy loss is typically simulated with great accuracy using the Geant4 [13] toolkit.

The crucial role of calorimeter simulation as a time-consuming bottleneck in the simulation chain at the LHC is well established. For example, the ATLAS experiment uses more than half of its total CPU time on the LHC Computing Grid for Monte Carlo simulation, which in turn is entirely dominated by the calorimeter simulation [14].

While generative neural network techniques promise enormous speed-ups for simulating the calorimeter response, it is of extreme importance that all relevant physical shower properties are reproduced accurately in great detail. This is particularly challenging for highly granular calorimeters, with a much higher spatial resolution, foreseen for most future colliders. Such concepts, as developed for the International Linear Collider (ILC), are also being used to upgrade detectors at the LHC for upcoming data-taking periods. One prominent example is the calorimeter endcap upgrade of the CMS experiment [15] with about 6 million readout channels. These factors make the timely development of precise simulation tools for high-resolution detectors relevant and motivate our investigation of a prototype calorimeter for the International Large Detector (ILD).

Outside of particle physics, generative adversarial neural networks [16] (GANs) have been used to produce synthetic data—such as photo-realistic images [17]—with great success. A traditional GAN consists of two networks, a generator and a discriminator separating artificial samples from real ones, which are trained against each other. An alternative to GANs for simulation are variational autoencoders [18] (VAE). A VAE consists of an encoder mapping from input data to a latent space, and a decoder, which maps from the latent space to data. If the probability distribution in latent space is known, it can be sampled from and used to generate synthetic data. A third path towards generative models is offered by normalizing flows [19–23]. In such models, a simple base probability distribution is transformed by a series of invertible mappings into a complex shape.

Recently, a novel architecture unifying several generative models such as GANs, VAEs, and others was proposed: the Bounded-Information-Bottleneck autoencoder (BIB-AE) [24]. We will show that by using a modified BIB-AE for generation we can accurately model all tested relevant physics distributions to a higher degree than achieved by traditional GANs. A detailed introduction to this architecture is provided in Sect. 3.3.

Specifically in particle physics, first results for the simulation of calorimeters focused on GANs achieved an impressive speed-up by up to five orders of magnitude compared to Geant4 [1, 25, 26]. Similarly, an approach using a Wasserstein-GAN (WGAN) architecture achieved realistic modeling of particle showers in air-shower detectors [27] and a high granularity sampling calorimeter [28]. In the context of future colliders, an architecture inspired by GANs was used for the fast simulation of showers in a high granularity electromagnetic calorimeter [29]. Generative models based on VAE and WGAN architectures were studied for concrete application by the ATLAS collaboration [30–32].

Beyond producing calorimeter showers, generative models in HEP have also been explored for modeling muon interactions with a dense target [33], parton showers [34–37], phase space integration [38–41], event generation [42–47], event subtraction [48] and unfolding [49].

The rest of this paper is organised as follows: in Sect. 2 we introduce the concrete problem and training data, in Sect. 3 the used generative architectures are discussed, and in Sect. 4 the obtained results are presented and compared. Finally, Sect. 5 provides conclusions and outlook.

Data Set

The ILD [50] detector is one of two detector concepts proposed for the ILC. It is optimized for Particle Flow, an algorithm that aims at reconstructing every individual particle in order to optimize the overall detector resolution. ILD combines high-precision tracking and vertexing capabilities with very good hermiticity and highly granular electromagnetic and hadronic calorimeters. For this study, one of the two proposed electromagnetic calorimeters for ILD, the Si-W ECal is chosen. It consists of 30 active silicon layers in a tungsten absorber stack with 20 layers of 2.1 mm followed by ten layers of 4.2 mm thickness respectively. The silicon sensors have $5 \times 5\,\text{mm}^2$ cell sizes. Throughout this work, we project the sensors onto a rectangular grid of $30 \times 30 \times 30$ cells. Each cell in this grid corresponds to exactly one sensor. As the underlying geometry of sensors in a realistic calorimeter prototype is not exactly regular, we will encounter some effects of this staggering. This makes the learning task more

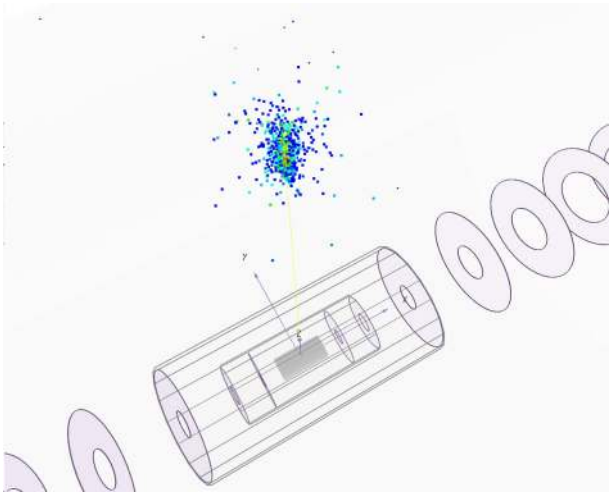

Fig. 1 A simulated 60 GeV photon shower in the ILD detector, as used in the training data

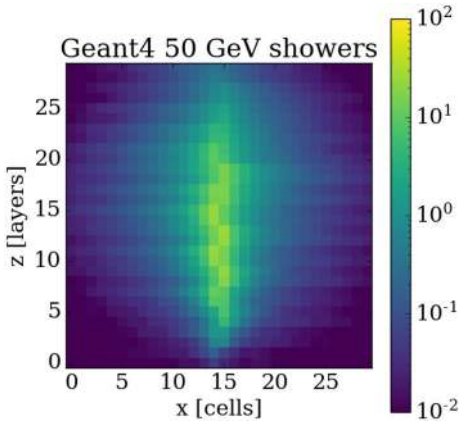

Fig. 2 Overlay of 2000 projections of 50 GeV Geant4 photon showers along the y direction

challenging for the network, but does not pose a fundamental problem. Architectures that more accurately encode irregular calorimeter geometries in neural networks exist [51], but are not the focus of this work.

ILD uses the iLCSoft [52] ecosystem for detector simulation, reconstruction and analysis. For the full simulation with Geant4, a detailed and realistic detector model implemented in DD4hep [53] is used. The training data of photon showers in the ILD ECal are simulated with Geant4 version 10.4 (with QGSP_ BERT physics list) and DD4hep version 1.11. The photons are shot at perpendicular incident angle into the ECal barrel with energies uniformly[2] distributed between 10 and 100 GeV. All incident photons are aimed at the $x-y$ center of the grid—i.e., at the point in the middle between the four most central cells of the front layer. An example event display showing such a photon shower is depicted in Fig. 1.

The incoming photon enters from the bottom at $z = 0$ and traverses along the z-axis, hitting cells in the center of the $x-y$ plane. No variations of the incident angle and impact point are performed in this study. The overlay of 2000 showers summed over the y-axis is shown in Fig. 2. As can be seen, the cells in the ILD ECal are staggered due to the specific barrel geometry. The whole data set for training consists of 950k showers with continuous energies between 10 and 100 GeV. For the evaluations we generated additional, statistically independent, sets of events: 40k events uniformly distributed between 10–100 GeV and 4k events each at discrete energies in steps of 10 GeV between 20 and 90 GeV.

Generative Models

Generative models are designed to learn an underlying data distribution in a way that allows later sampling and thereby producing new examples. In the following, we first present two approaches—GAN and WGAN—which represent the state-of-the-art in generating calorimeter data and which we use to benchmark our results. We then introduce BIB-AE as a novel approach to this problem and discuss further refinement methods to improve the quality of generated data.

Generative Adversarial Network

The GAN architecture was proposed in 2014 [16] and had remarkable success in a number of generative tasks. It introduces generative models by an adversarial process, in which a generator G competes against an adversary (or discriminator) D. The goal of this framework is to train G in order to generate samples $\widetilde{x} = G(z)$ out of noise z, which are indistinguishable from real samples x. The adversary network D is trained to maximize the probability of correctly classifying whether or not a sample came from real data using the binary cross-entropy. The generator, on the other hand, is trained to fool the adversary D. This is represented by the loss function as

$$L = \min_G \max_D \mathbb{E}[\log D(x)] + \mathbb{E}[\log(1 - D(G(z)))], \quad (1)$$

and a schematic of the GAN training is provided in Fig. 3 (top).

For practical applications, the GAN needs to simulate showers of a specific energy. To this end, we parameterise

[2] Due to technical issues with the Geant4 generation step, the produced sample has a difference in statistics of 1% between the lowest and highest energies.

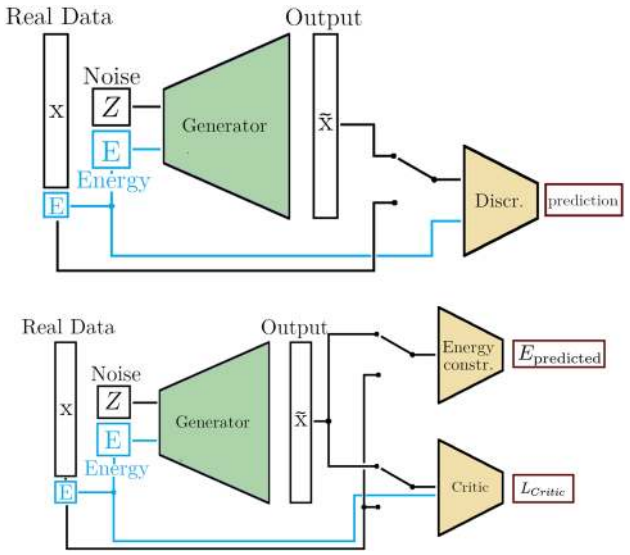

Fig. 3 Overview of the GAN (top) and WGAN (bottom) architectures. The blue line shows where the true energy is used as an input. The loss functions and feedback loops are explained in the text

generator and discriminator as functions of the photon energy E [54]. In general, we attempted to minimally modify the CaloGAN formulation [26] to work with the present dataset.

The original formulation of a GAN produces a generator that minimizes the Jensen–Shannon divergence between true and generated data. In general, the training of GANs is known to be technically challenging and subject to instabilities [55]. Recent progress on generative models improves upon this by modifying the learning objective.

Wasserstein-GAN

One alternative to classical GAN training is to use the Wasserstein-1 distance, also known as earth mover's distance, as a loss function. This distance evaluates dissimilarity between two multi-dimensional distributions and informally gives the cost expectation for moving a mass of probability along optimal transportation paths [56]. Using the Kantorovich-Rubinstein duality, the Wasserstein loss can be calculated as

$$L = \sup_{f \in \mathrm{Lip}_1} \{\mathbb{E}[f(x)] - \mathbb{E}[f(\tilde{x})]\}. \tag{2}$$

The supremum is over all 1-Lipschitz functions f, which is approximated by a discriminator network D during the adversarial training. This discriminator is called *critic* since it is trained to estimate the Wasserstein distance between real and generated images.

In order to enforce the 1-Lipschitz constraint on the critic [57], a gradient penalty term should be added to (2), yielding the critic loss function:

$$\begin{aligned} L_{\text{Critic}} &= \mathbb{E}[D(G(z))] - \mathbb{E}[D(x)] \\ &\quad + \lambda\ \mathbb{E}[(\parallel \nabla_{\hat{x}} D(\hat{x}) \parallel_2 - 1)^2], \end{aligned} \tag{3}$$

where λ is a hyper parameter for scaling the gradient penalty. The term $\hat{x}$ is a mixture of real data x and generated $G(z)$ showers. Following [57], it is sampled uniformly along linear interpolations between x and $G(z)$.

Finally, we again need to ensure that generated showers accurately resemble photons of the requested energy. We achieve this by parameterising the generator and critic networks in E and by adding a constrainer [28] network a. The loss function for the generator then reads:

$$\begin{aligned} L_{\text{Generator}} &= -\mathbb{E}[D(\tilde{x}, E)] \\ &\quad + \kappa \cdot \mathbb{E}[\left|(a(\tilde{x}) - E)^2 - (a(x) - E)^2\right|], \end{aligned} \tag{4}$$

where $\tilde{x}$ are generated showers and κ is the relative strength of the conditioning term. This combined network is illustrated in Fig. 3. The constrainer network is trained solely on the Geant4 showers; its weights are fixed during the generator training. We use the mean absolute error (L1) as loss[3]:

$$L_{\text{Constrainer}} = |E - a(x)|. \tag{5}$$

Bounded Information Bottleneck-Autoencoder

Autoencoder architectures map input to output data via a latent space. Using a structured latent space allows for later sampling and thereby generation of new data. The BIB-AE [24] architecture was introduced as a theoretical overarching generative model. Most commonly employed generative models—e.g. GAN [16], VAE [18], and adversarial autoencoder (AAE) [58]—can be seen as different subsets of the BIB-AE. This leads to better control over the latent space distributions and promises better generative performance and interpretability. In the following, we focus on the practical advantage gained from utilizing the individual BIB-AE components and refer to the original publication [24] for an information-theoretical discussion.

As it is an overarching model, an instructive way for describing the base BIB-AE framework is by taking a VAE and expanding upon it. A default VAE consist of four general components: an encoder, a decoder, a latent-space regularized by the Kullback–Leibler divergence (KLD), and an L_N-norm to determine the difference between the original and the reconstructed data. These components are all present as well in the BIB-AE setup. Additionally,

[3] Using L1 loss here gives better performance than L2, as L2 seems to introduce too large a penalisation for the occasionally expected outliers in the total energy sum due to the finite calorimeter resolution.

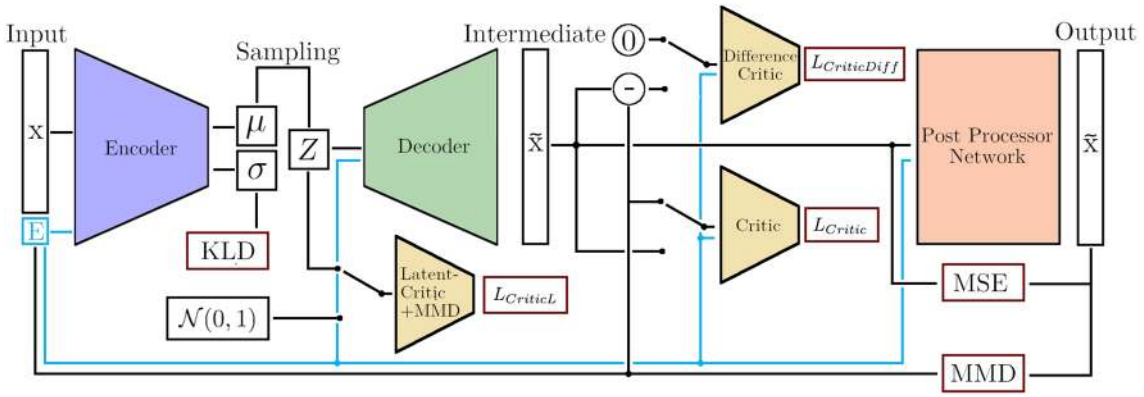

Fig. 4 Diagram of the BIB-AE architecture, including the additional MMD term defined in Sect. 3.4 and the Post Processor Network defined in Sect. 3.5. The blue line shows where the true energy is used as an input. The loss functions and feedback loops are explained in the text

one introduces a GAN-like adversarial network, trained to distinguish between real and reconstructed data, as well as a sampling based method of regularizing the latent space, such as another adversarial network or a maximum mean discrepancy (MMD, as described in the next section) term. In total this adds up to four loss terms: the KLD on the latent space, the sampling regularization on the latent space, the L_N-norm on the reconstructed samples and the adversary on the reconstructed samples. The guiding principle behind this is that the two latent space and the two reconstruction losses complement each other and, in combination, allow the network to learn a more detailed description of the data. Specifically looking at the two reconstruction terms we have, on the one hand, the adversarial network: from tests on utilizing GANs for shower generation we know that such adversarial networks are uniquely qualified to teach a generator to reproduce realistic looking individual showers. On the other hand, we have the L_N-norm: while our trials with pure VAE setups have shown that L_N-norms have great difficulty capturing the finer structures of the electromagnetic showers, an L_N -norm also forces the encoder-decoder structure to have an expressive latent space, as the original images could not be reconstructed without any latent space information. Therefore, the adversarial network forces the individual images to look realistic, while the L_N-norm forces latent space utilization, thereby improving how well the overall properties of the data set are reproduced. The latent space loss terms have a similar interaction. Here the KLD term regularizes our complete latent space by reducing the difference between the average latent space distribution and a normal Gaussian. The KLD is, however, largely blind to the shape of the individual latent space dimensions, as it only cares about the average. The sampling based latent space regularization term fills this niche by looking at every latent space dimension individually.

Our specific implementation of the BIB-AE framework is shown in Fig. 4. For our sampling based latent regularization we use both an adversary and an MMD term. The adversaries are implemented as critics trained with gradient penalty, similar to the WGAN approach. The main difference in our setup compared to the one described in [24] is that we replaced the L_N-norm with a third critic, trained to minimize the difference between input and reconstruction. We chose this because we found that using the L_N-norm to compare the input and the reconstructed output resulted in smeared out images.

For the precise implementation of the loss functions we define the encoder network N, the decoder network D, the latent critic C_L, the critic network C, and the difference critic C_D. The loss function for the latent critic C_L is given by

$$L_{C_L} = \mathbb{E}[C_L(N_E(x))] - \mathbb{E}[C_L(\mathcal{N}(0,1))] + \lambda\, \mathbb{E}[(\| \nabla_{\hat{x}} C_L(\hat{x}) \|_2 - 1)^2]. \tag{6}$$

Here $\hat{x}$ is a mixture of the encoded input image $N(x)$ and samples from a normal distribution $\mathcal{N}(0,1))$ and the E subscript indicates that the network receives the photon energy label as an input. The loss function for the main critic C is given by

$$L_C = \mathbb{E}[C_E(D_E(N_E(x)))] - \mathbb{E}[C_E(x)] + \lambda\, \mathbb{E}[(\| \nabla_{\hat{x}} C_E(\hat{x}) \|_2 - 1)^2]. \tag{7}$$

Where $\hat{x}$ is a mixture of the reconstructed image $D(N(x))$ and the original images x. Finally, the loss function for the difference critic C_D is given by

$$L_{C_D} = \mathbb{E}[C_{D,E}(D_E(N_E(x)) - x)] - \mathbb{E}[C_{D,E}(x - x = 0)] + \lambda\, \mathbb{E}[(\| \nabla_{\hat{x}} C_{D,E}(\hat{x}) \|_2 - 1)^2]. \tag{8}$$

Where $\hat{x}$ is a mixture of the difference $D(N(x)) - x$ and the difference $x - x = 0$. With different β factors giving the relative weights for the individual loss terms, the combined loss

for the encoder and decoder parts of the BIB-AE can be expressed as:

$$\begin{aligned} L_{\text{BIB-AE}} = &-\beta_{C_L} \cdot \mathbb{E}[C_L(N_E(x))] \\ &- \beta_C \cdot \mathbb{E}[C_E(D_E(N_E(x)))] \\ &- \beta_{C_D} \cdot \mathbb{E}[C_{D,E}(D_E(N_E(x)) - x)] \\ &+ \beta_{\text{KLD}} \cdot \text{KLD}(N_E(x)) \\ &+ \beta_{\text{MMD}} \cdot \text{MMD}(N_E(x), \mathcal{N}(0,1))). \end{aligned} \tag{9}$$

Maximum Mean Discrepancy

One major challenge in generating realistic photon showers is the spectrum of the individual cell energies, which is shown in Fig. 6 (left) in Sect. 4. The real spectrum shows an edge around the energy that a minimal ionizing particle (MIP) would deposit. Since the well-defined energy deposition of a MIP is often used to calibrate a calorimeter, we cannot simply ignore it. However, we found that purely adversarial based methods tend to smooth out this and other similar low energy features, an observation in line with other efforts to use generative networks for shower simulation [28]. A way of dealing with this is using MMD [59] to compare and minimize the distance between the real (D_R) and fake (D_F) hit-energy distributions:

$$\text{MMD}(D_R, D_F) = \langle k(x,x') \rangle + \langle k(y,y') \rangle - 2\langle k(x,y) \rangle, \tag{10}$$

where x and y are samples drawn from D_R and D_F respectively and k is any positive definite kernel function. MMD based losses have previously been used in the generation of LHC events [46].

A naive implementation of the MMD would be to compare every pixel value from a real shower with every value from a generated shower. This approach is however not feasible since it would involve computing Eq. (10) approximately $(30^3)^2$ times for each shower. To make the MMD calculation tractable, we introduce a novel version of the MMD, termed Sorted-Kernel-MMD. We first sort both, real and generated, hit-energies in descending order, and then take the n highest fake energies and compare them to the n highest real energies. Following this we move the n-sized comparison window by m and recompute the MMD. This process is repeated $\frac{N}{m}$-times, where N is the total number of pixels one wants to compare. The advantage of this approach is two-fold, for one the number of computations is linear in N, as opposed to the naive implementation which shows quadratic behavior. The second advantage is that energies will only be compared to similar values, thereby incentivising the model to fine-tune the energy. Specifically, the values $m = 25$, and $n = 100$ are used and we chose $N = 2000$, as this is approximately the maximum occupancy observed in our training data before any low energy cutoffs. In our experiments, adding this MMD term with the kernel function

$$k(x,x') = e^{-\alpha(x^2+x'^2-2xx')} \tag{11}$$

with $\alpha = 200$ to the loss term of either a GAN or a BIB-AE fixes the per-cell hit energy spectrum to be near identical to the training data. This however comes at a price, as the additional pixels with the energies used to fix the spectrum are often placed in unphysical locations, specifically at the edges of the $30 \times 30 \times 30$ cube.

Post Processing

In the previous section we found that using an MMD term in the loss function represents a trade off between correctly reproducing either the hit energy spectrum or the shower shape. To solve this, we split the problem into two networks that are applied consecutively but trained with different loss functions. The first network is a GAN or BIB-AE trained without the MMD term. This produces showers with correct shapes, but an incorrect hit-energy spectrum. The second network then takes these showers as its input and applies a series of convolutions with kernel size one. Therefore this second network can only modify the values of existing pixels, but not easily add or remove pixels. This second network, here called Post Processor Network, is trained using only the MMD term to fix the hit energy spectrum, and the mean squared error (MSE) between the input and output images, ensuring the change from the Post Processor Network is as minimal as possible.

Results

In the following we present the ability of our generative models to accurately predict a number of per-shower variables as well as global observables and analyse the achievable gain in computing performance. We include our implementation of a simple GAN (Sect. 3.1), a WGAN with additional energy constrainer (Sect. 3.2), and a BIB-AE with energy-MMD and post processing (Sects. 3.3, 3.4 and 3.5). A detailed discussion of the architectures and training hyper parameters can be found in Appendix A. All architectures are trained on the same sample of 950k Geant4 showers. Tests are either shown for the full momentum range (labeled *full spectrum*) or for specific shower energies (labeled with the incident photon energy in GeV).

Physics Performance

We first verify in Fig. 5 that the showers generated by all network architectures visually appear to be acceptable

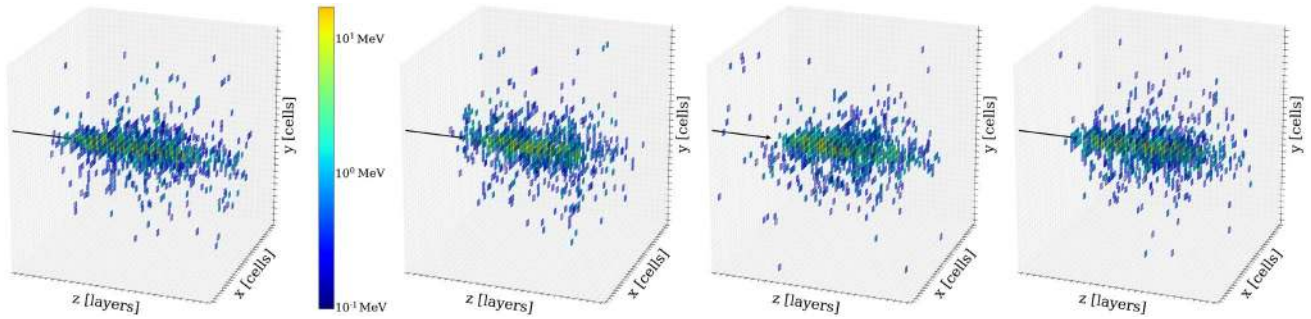

Fig. 5 Examples of individual 50 GeV photon showers generated by Geant4 (left), the GAN (center left), WGAN (center right), and BIB-AE (right) architectures. Colors encode the deposited energy per cell

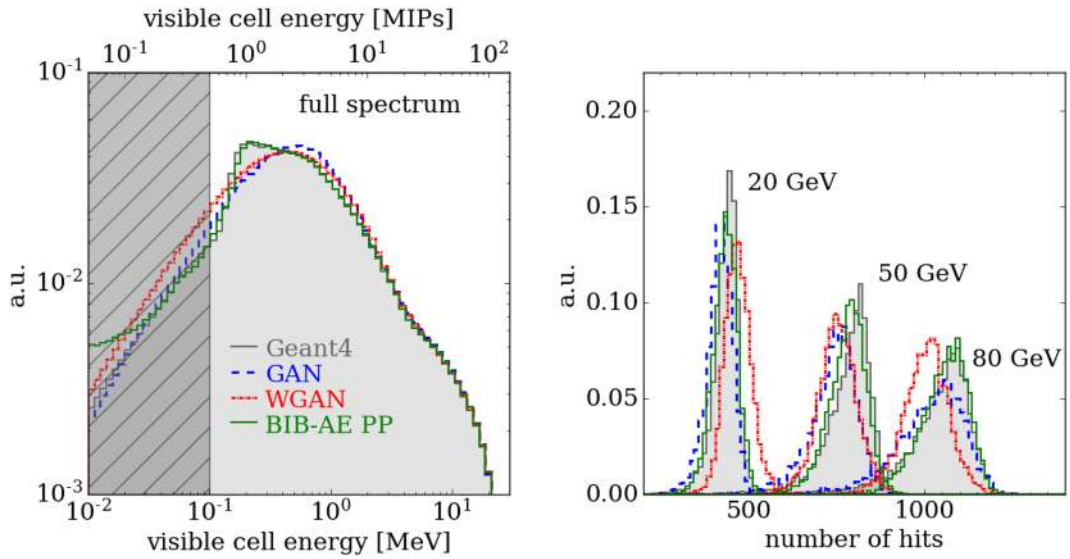

Fig. 6 Differential distributions comparing the per-cell energy (left) and the number of hits above 0.1 MeV (right) between Geant4 and the different generative models. Shown are Geant4 (grey, filled), our GAN setup (blue, dashed), our WGAN (red, dotted) and the BIB-AE (green, solid). The energy per-cell is measured in MeV for the bottom axis and in multiples of the expected energy deposit of a minimum ionizing particle (MIP) for the top axis

compared to Geant4. Were we attempting to generate *cute cat pictures*, our work would be done already at this point. Alas, these shower images are eventually to be used as realistic substitutes in physics analyses so we need to pay careful attention to relevant differential distributions and correlations.

In Fig. 6 a comparison between two differential distributions for all studied architectures and Geant4 is shown. The left plot compares the per-cell hit-energy spectrum averaged over showers for the full spectrum of photon energies. We observe that while the high-energy hits are well described by all generative models, both GAN and WGAN fail to capture the bump around 0.2 MeV. The BIB-AE is able to replicate this feature thanks to the Post Processor Network.[4] This energy corresponds to the most probable energy loss of a MIP passing a silicon sensor of the ILD Si-W ECal at perpendicular incident angle. Since this is a well-defined energy, it can be used in highly granular calorimeters for the equalisation of the cell response as well as for setting an absolute energy scale. It also leads to a sharp rise in the spectrum, as lower energies can only be deposited by ionizing particles that pass only a fraction of the thickness at the edges of sensitive cells or that are stopped. The region below half a MIP, corresponding to around 0.1 MeV, is shaded in dark grey. These cell energies are very small and therefore

[4] We studied applying post processing to the WGAN architecture as well. This is discussed in Sect. 4.2.

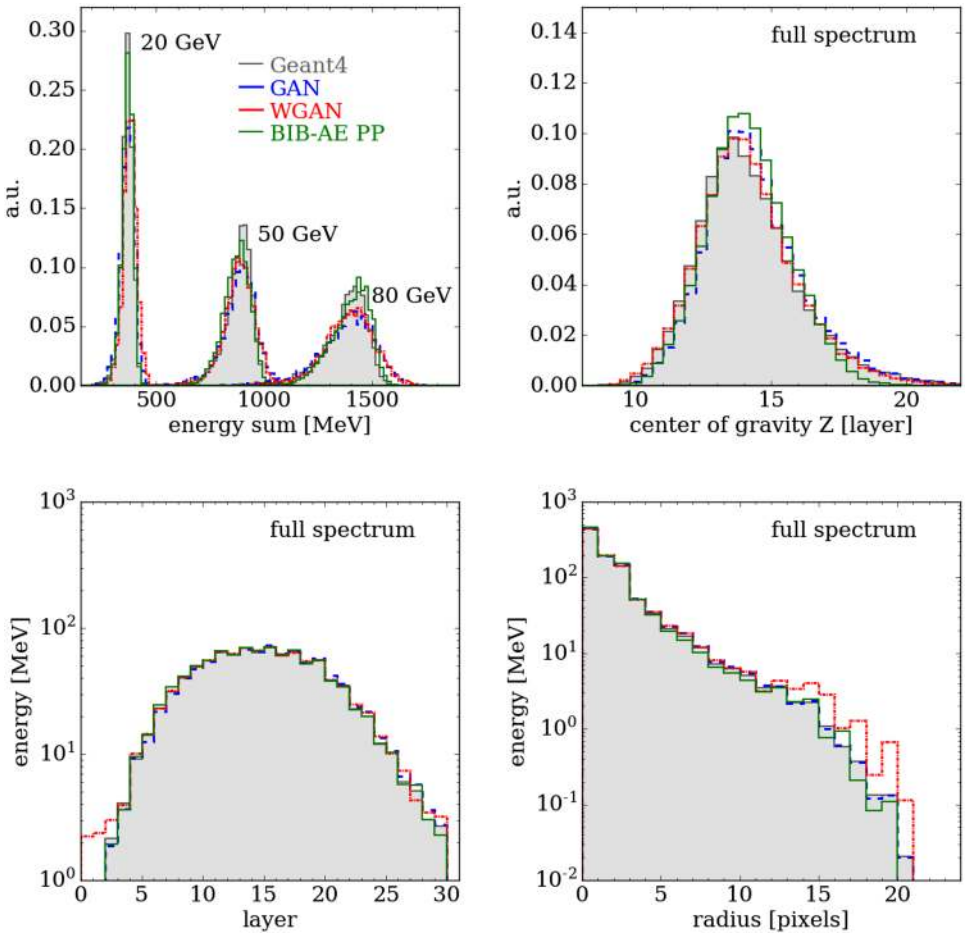

Fig. 7 Additional differential distributions comparing physical observables between Geant4 and the different generative models. Shown are Geant4 (grey, filled), our GAN setup (blue, dashed), our WGAN (red, dotted) and the BIB-AE with Post Processing (green, solid)

will be discarded in a realistic calorimeter, as their signal to noise ratio is too low. For the following discussion cell energies below 0.1 MeV will therefore not be considered and only cells above this cut-off are included in all other performance plots and distributions.

Next, the plot on the right shows the number of hits for three discrete photon energies (20 GeV, 50 GeV, and 80 GeV). Here, the GAN and WGAN setups slightly underestimate the total number of hits, while the BIB-AE accurately models the mean and width of the distribution. This behavior can be traced back to the left plot. Since we apply a cutoff removing hits below 0.1 MeV, a model that does not correctly reproduce the hit-energy spectrum around the cut-off will have difficulties correctly describing the number of hits.

Additional distributions are shown in Fig. 7. The top left depicts the visible energy distribution for the same three discrete photon energies. Both, the shape, center and width of the peak are well reproduced for all models. Due to the sampling nature of the calorimeter under study, the visible energy is of course much lower than the incoming photons' energy.

In the top right and bottom two plots we compare the spatial properties of the generated showers. First, on the top right, the position of the center of gravity along the z axis is shown. The Geant4 distribution is well modelled by the GANs, however there are slight deviations for the BIB-AE. A detailed investigation of this discrepancy showed that the z axis center of gravity is largely encoded in a single latent space variable. A mismatch between the observed latent distribution for real samples and the normal distribution drawn from when generating new samples directly translates into the observed difference. Sampling from a modified distribution would remove the problem.

Finally, the two plots on the bottom show the longitudinal and radial energy distributions. We see that while all models are able the reproduce the bulk of the distributions very well, deviations for the WGAN appear around the edges.

We next test how well the relation of visible energy to the incident photon energy is reproduced. To this end we use a Geant4 sample where we simulated photons at discrete energies ranging from 20 to 90 GeV in 10 GeV steps. We then use our models to generate showers for these energies and calculate the mean and root-mean-square of the 90% core of the distribution, labeled μ_{90} and σ_{90} respectively, for all sets of showers. The results are shown in Fig. 8. Overall the mean (left) is correctly modelled, showing only deviations in

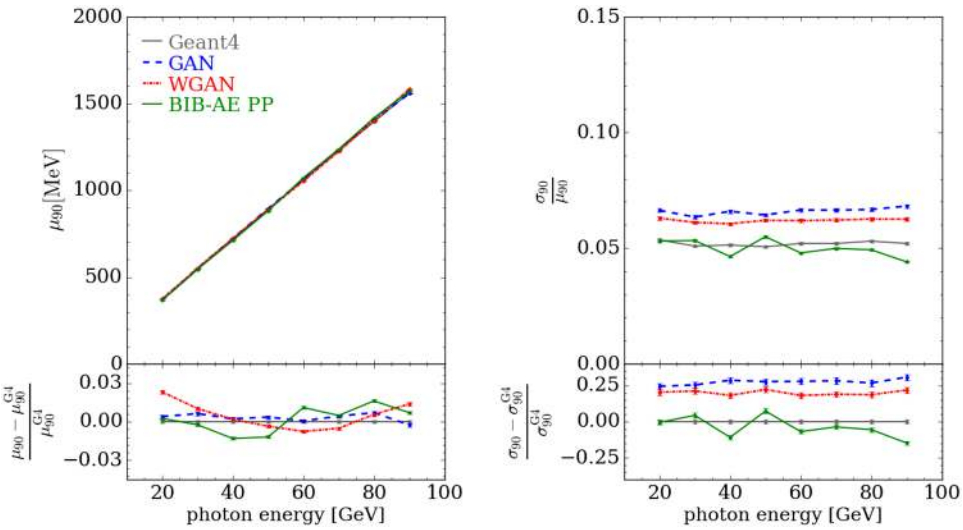

Fig. 8 Plot of mean (μ_{90}, left) and relative width (σ_{90}/μ_{90}, right) of the energy deposited in the calorimeter for various incident particle energies. In order to avoid edge effects, the phase space boundary regions of 10 and 100 GeV are removed for the response and resolution studies. In the bottom panels, the relative offset of these quantities with respect to the Geant4 simulation is shown

the order of one to two percent. The relative width, σ_{90}/μ_{90} (right) looks worse: GAN and WGAN overestimate the Geant4 value at all energies. While the BIB-AE on average correctly models the width, it still shows deviations of up to ten percent at high energies. Note that the width cannot be interpreted as energy resolution of the calorimeter due to the two different absorber thicknesses used in the ECal, requiring different calibrations.

Finally, we verify whether correlations between individual shower properties present in Geant4 are correctly reproduced by our generative setups. The properties chosen for this are: The first and second moments in x, y and z direction, labeled as $m_{1,x}$ through $m_{2,z}$, the visible energy deposited in the calorimeter E_{vis}, the energy of the simulated incident particle E_{inc}, the number of hits n_{hit}, and the ratio between the energy deposited in the 1st/2nd/3rd third of the calorimeter and the total visible energy, labeled E_1/E_{vis} through E_3/E_{vis}. The results are shown in Fig. 9. The top left plot shows the correlations for Geant4 showers. We then present the difference to Geant4 for the GAN (top right), WGAN (bottom left), and BIB-AE (bottom right). The smallest differences are observed for the GAN (absolute maximum difference of 0.2), followed BIB-AE (0.36) and WGAN (0.57).

Figure 10 shows examples of 2D scatter plots: the number of hits and the visible energy (top row) as well as the center of gravity and the visible energy (bottom row). These allow us insight into the full correlations between these variables beyond the simple correlation coefficients. Similar to Fig. 9 we see that the GAN matches the Geant4 correlations exceptionally well, while the WGAN and the BIB-AE display some slight correlation mis-matching. The discrepancy in the BIB-AE center of gravity and visible energy correlation can be traced back to the mismodelling of the center of gravity as seen in Fig. 7.

The distributions of physical observables shown above are expected to be the major factor for assessing the quality of a simulation tool. While the correlations are also useful as they provide additional insight, our main focus when evaluating network performance are the physics distributions.

The Importance of Post Processing

In the previous section we demonstrated that our proposed architecture—the BIB-AE with a post processor network—achieved excellent performance in simulating important calorimeter observables. In the following, we will dissect this improvement. To this end we compare a WGAN trained with an additional simple MMD kernel (labelled WGAN MMD), a WGAN trained with the full post processing (labelled WGAN PP), a BIB-AE without post processing (labelled BIB-AE) to Geant4 and to the combined BIB-AE network including post processing (labelled BIB-AE PP) from the main text. We do not investigate a simple GAN with post processing as we expect it to exhibit largely the same behaviour as the WGAN.

In Fig. 11 we show the performance of these approaches. The top left panel of Fig. 11 demonstrates that removing post-processing from the BIB-AE leads to a smeared out MIP peak, while adding the simple MMD term or the more complex post processing to the WGAN result in good modelling of the per-cell hit energy spectrum. However, now this improvement comes at a price: the distribution of the number of hits (top right) is too narrow compared to Geant4 and the longitudinal (bottom center) and radial (bottom right) energy profiles are described badly as additional energy is deposited at the edges of the shower. Especially noticeable is the additional energy in the first and last layers. This would be problematic for standard reconstruction methods that rely on the precise position of the shower start and end.

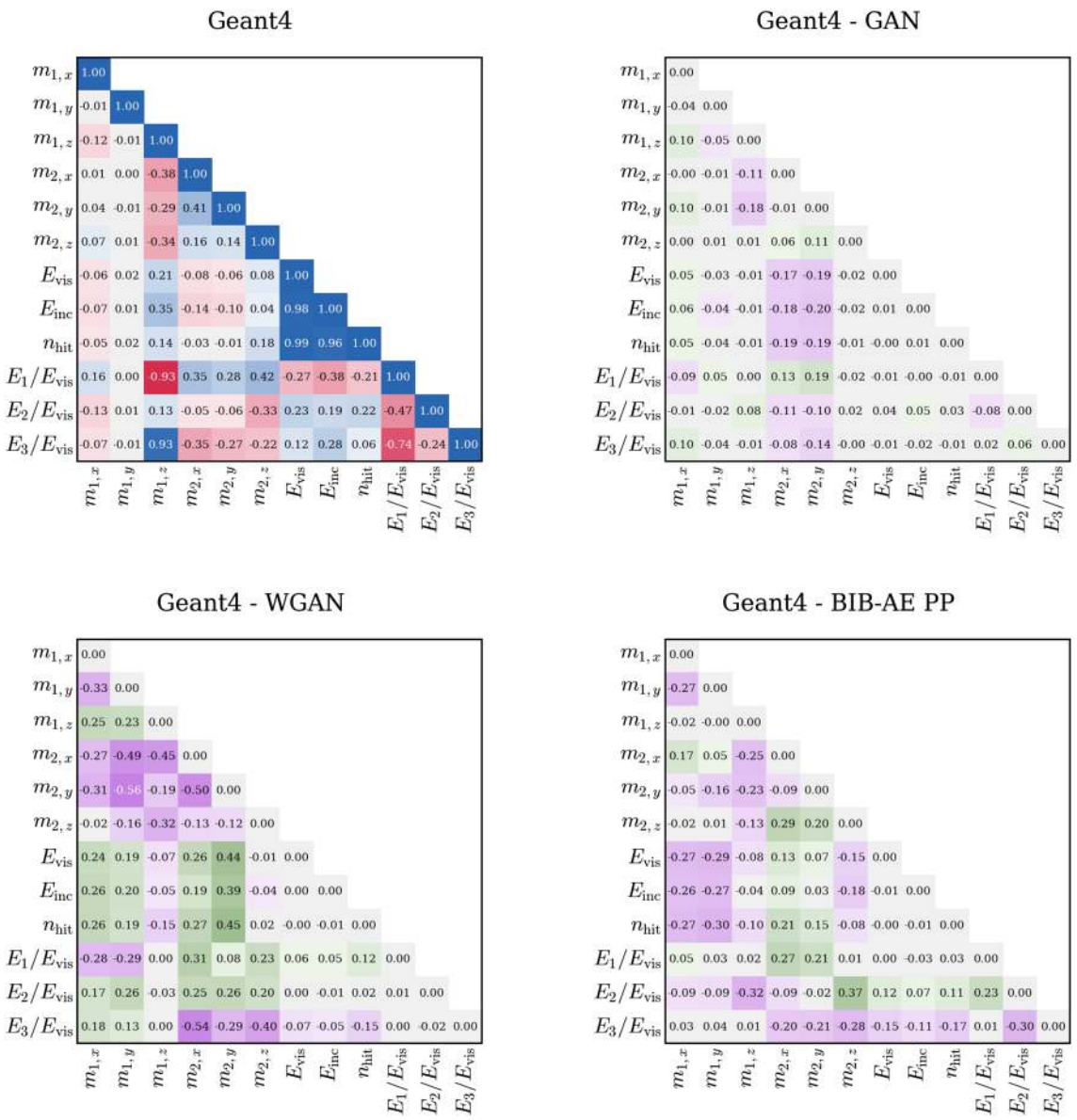

Fig. 9 Linear correlation coefficients between various quantities described in the text in Geant4 (top left). Difference between these correlations in Geant4 and GAN (top right), Geant4 and WGAN (bottom left), and Geant4 and BIB-AE with post processing (bottom right). The mean absolute differences compared to Geant4 are 0.058 for the GAN, 0.187 for the WGAN and 0.132 for the BIB-AE

These energy deposits along the image edges are the main reason why the BIB-AE Post Processor is implemented as a separate network rather than integrated in the main decoder structure. The latter would require applying the MMD loss to the entire decoder, which in our test led to energy deposits similar to what can be seen in the WGAN MMD line.

While we were not able to improve the WGAN approach via post processing, we are not aware of fundamental reasons why a better performance using a similar method should not be possible for GAN and WGAN based architectures as well. One reason why AE based architectures might allow better training of post processing steps is however the higher correlation between real input and fake samples via the latent space embedding. Nonetheless, the ability of the BIB-AE framework to make use of this post processing setup motivates future studies of this rather novel architecture for calorimeter shower generation.

Computational Performance

Beyond the physics performance of our generative models, discussed in the previous section, the major argument for these approaches is of course the potential gain in production time. To this end, we benchmark the per-shower

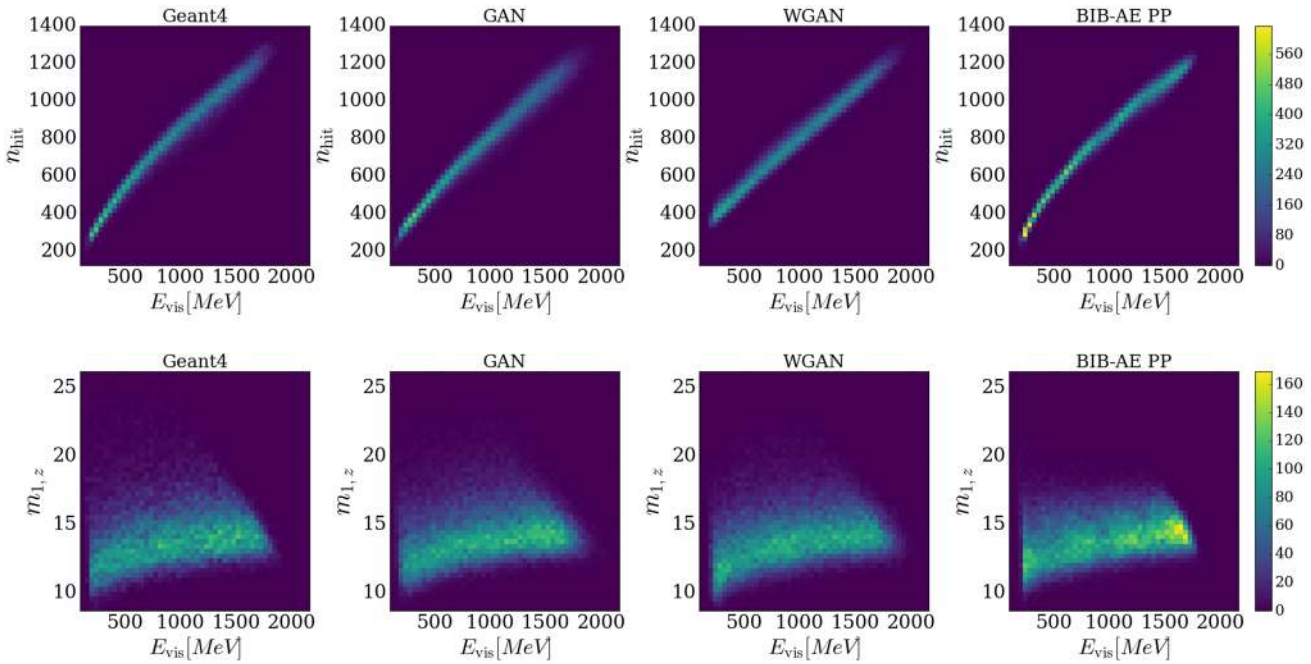

Fig. 10 Scatter plot showing the correlations between visible energy and number of hits (top) and visible energy and center of gravity (bottom)

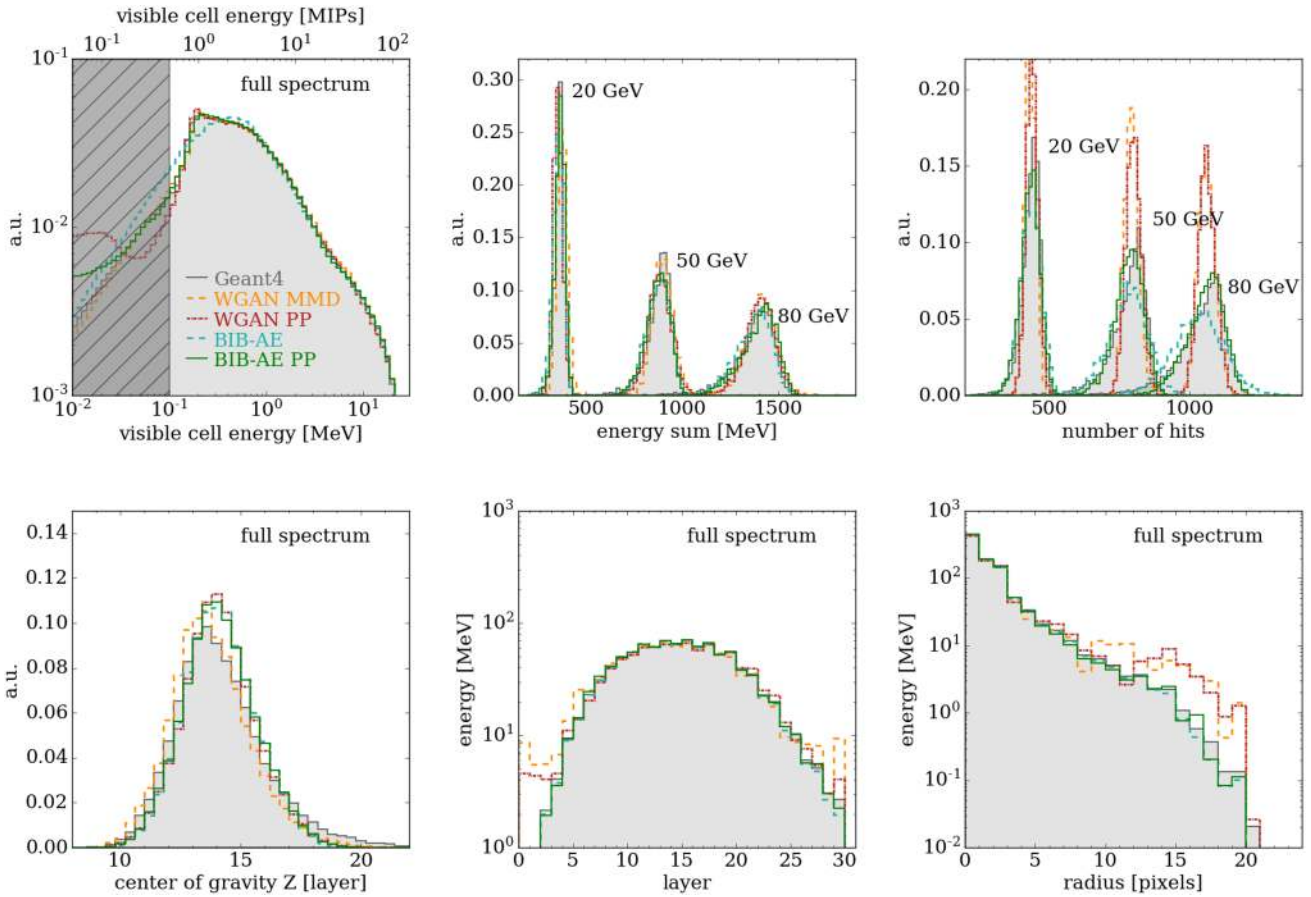

Fig. 11 Differential distributions comparing physics quantities between Geant4 and the different generative models. The energy per-cell is measured in MeV for the bottom axis and in multiples of the expected energy deposit of a minimum ionizing particle (MIP) for the top axis

Table 1 Overview of computational performance of WGAN and BIB-AE model, compared to Geant4 full simulation

Simulator	Hardware	Batch size	15 GeV	Speed-up	10–100 GeV Flat	Speed-up
Geant4	CPU	N/A	1445.05 ± 19.34 ms	–	4081.53 ± 169.92 ms	–
WGAN	CPU	1	64.34 ± 0.58 ms	×**23**	63.14 ± 0.34 ms	×**65**
		10	59.53 ± 0.45 ms	×**24**	56.65 ± 0.33 ms	×**72**
		100	58.31 ± 0.93 ms	×**25**	58.11 ± 0.13 ms	×**70**
		1000	57.99 ± 0.97 ms	×**25**	57.99 ± 0.18 ms	×**70**
BIB-AE	CPU	1	426.60 ± 3.27 ms	×**3**	426.32 ± 3.62 ms	×**10**
		10	422.60 ± 0.26 ms	×**3**	424.71 ± 3.53 ms	×**10**
		100	419.64 ± 0.07 ms	×**3**	418.04 ± 0.20 ms	×**10**
WGAN	GPU	1	3.24 ± 0.01 ms	×**446**	3.25 ± 0.01 ms	×**1256**
		10	6.13 ± 0.02 ms	×**236**	6.13 ± 0.02 ms	×**666**
		100	5.43 ± 0.01 ms	×**266**	5.43 ± 0.01 ms	×**752**
		1000	5.43 ± 0.01 ms	×**266**	5.43 ± 0.01 ms	×**752**
BIB-AE	GPU	1	3.14 ± 0.01 ms	×**460**	3.19 ± 0.01 ms	×**1279**
		10	1.56 ± 0.01 ms	×**926**	1.57 ± 0.01 ms	×**2600**
		100	1.42 ± 0.01 ms	×**1017**	1.42 ± 0.01 ms	×**2874**

Evaluated on both a single core of a Intel® Xeon® CPU E5-2640 v4 (CPU) and NVIDIA® V100 with 32 GB of memory (GPU). Numerical values represent the mean and standard deviation of 25 runs

generation time both on CPU and GPU hardware architectures. In Table 1, we provide the performance for 4 (3) batch sizes for the WGAN[5] (BIB-AE). We observe a speed-up by evaluating generative models on GPU vs. Geant4 on CPU of up to almost a factor of three thousand. Moreover, the evaluation time of our generative models is independent of the incident photon energy while this is not the case for the Geant4 simulation.

Conclusion

The accelerated simulation of calorimeters with generative deep neural networks is an active area of research. Early works [1, 25, 26] established generative networks as a fast and very promising tool for particle physics and simulated the positron, photon, and charged pion response of an idealised perfect calorimeter with three layers and a total of 504 cells ($3 \times 96, 12 \times 12$, and 12×6).

Using the WGAN architecture and an energy constrainer network [28] allowed the correct simulation of the observed total energy of electrons for a calorimeter consisting of seven layers with a total of 1260 cells (12×15 cells per layers). However, a mismodelling of individual cell energies below 10 MIPs, also leading to an observed deviation in the hit multiplicity distribution, was observed and studied. Our implementation of a WGAN based on [28] reproduces this effect (see Fig. 6 (left)). The proposed BIB-AE architecture with additional MMD loss term and Post Processor Network leads to a reliable description of low energy deposits.

The ATLAS collaboration also reported the accurate simulation of high-level observables for photons in a four-layer calorimeter segment with a total of 276 cells ($7 \times 3, 57 \times 4$, 7×7 and 7×5) using a VAE architecture [31] and 266 cells using a WGAN [32]. Recent progress was made applying a GAN architecture to simulating electrons in a high granularity calorimeter prototype [29]. The considered detector consists of 25 layers with 51×51 cells per layer, leading to a total of 65k cells to be simulated. On this very challenging problem, good agreement with Geant4 was achieved for a number of differential distributions and correlations of high-level observables. Specifically, the per-cell energy distribution was not reported, however the disagreement in the hit multiplicity again implies a mismodelling of the MIP peak region.

Our specific contribution is the first high fidelity simulation for a number of challenging quantities relevant for downstream analysis, including the overall energy response and per-cell energy distribution around the MIP peak, for a realistic high-granularity calorimeter. This is made possible by the first application of the BIB-AE architecture—unifying GAN and VAE approaches—in physics. Modifications to this architecture, specifically an additional kernel-based MMD loss term and a Post Processor Network, were developed. These improvements can potentially also be applied to other generative architectures and models. Planned future work includes the extension of this approach to also cover multiple particle types, incident positions and angles towards a complete, fast, and physically reliable synthetic calorimeter simulation.

[5] The time evaluation of the GAN network is not reported since the generator architecture is very similar to the WGAN.

Appendix: Network Architectures and Training Procedure

The network architectures of generative models have a large number of moving parts and the contributions from various generators, discriminators, and critics need to be carefully orchestrated to achieve good results. In the following we provide details of the implementation and training for the GAN, WGAN, and BIB-AE models. Due to the high computational cost of the studies—e.g., the BIB-AE was trained for a total of four days in parallel on four NVIDIA Tesla V100 (32 GB) GPUs—no systematic tuning of hyperparameters was performed. For all architectures a good modelling of the Geant4 training distributions was used as stopping criterion. All architectures are implemented in PyTorch [60] version 1.3.

GAN Training

Our implementation of the simple GAN is inspired by [1, 25, 26] and it should serve as an easy to implement baseline model consisting of a generator and a discriminator. In total, the generator has 1.5M trainable weights and the discriminator has 2.0M weights. We therefore did not consider additional modifications to the GAN approach such as training with a gradient penalty term.

The generator network of the GAN consists of 3-dimensional transposed convolution layers with batch normalization. It takes a noise vector of length 100, uniformly distributed from -1 to 1, and the true energy labels E as inputs. A first transposed convolution with a 4^3 kernel (stride 1) is applied to the noise vector multiplied by E. The main transposed convolution consists of four layers. The first three layers have a kernel size of 4^3 (stride 2) followed by batch normalization. The final layer has a kernel size of 3^3 (stride 1). All layers use ReLU [61] as activation function.

The discriminator uses five 3-dimensional convolution layers followed by two fully connected layers with 257 and 128 nodes respectively. The convolution layers use a 3^3 kernel. The stride is 2 for all convolutional layers. Batch normalisation [62] is applied after each convolution except in the first and last layer. We flatten the output of the convolutions and concatenate it the with input energy before passing it to the fully connected layers. Each fully connected layer except the final one uses LeakyReLU [63] (slope: -0.2) as an activation function. The activation in the final layer is sigmoid.

For training, we use the Adam optimizer [64] (learning rate 2×10^{-5}). The training process starts from updating the discriminator for real and fake showers. After that we freeze the parameters of the discriminator and update the generator with a new generated batch of fake showers. The generator and discriminator are trained alternating until the training is stopped after 125k weight updates—corresponding to approximately six epochs—when good modelling of the control distributions is achieved.

WGAN Training

The WGAN architecture, based on [27, 28], consists of three networks: one generator with 3.7M weights, one critic with 250k weights, and one constrainer network with 220k weights. The critic network starts with four 3D convolution layers with kernel sizes $(X, 2, 2)$ with $X = 10, 6, 4, 4$ which have 32, 64, 128, and 1 filters respectively. LayerNorm [65] layers are sandwiched between the convolutions. After the last convolution, the output is concatenated with the E vector required for $E-$conditioning. After that, it is flattened and fed into a fully connected network with 91, 100, 200, 100, 75, 1 nodes. Throughout the critic, LeakyReLU (slope: -0.2) is used as activation function.

The generator network takes a latent vector z (normally distributed with length 100) and true E labels as input and separately passes them through a 3D transposed convolution layer using a 4^3 kernel with 128 filters. After that, the outputs are concatenated and processed through a series of four 3D transposed convolution layers (kernel size 4^3 with filters of 256, 128, 64, 32). LayerNorm layers along with ReLU activation functions are used throughout the generator.

The energy-constrainer network is similar to the critic: three 3D convolutions with kernel sizes 3^3, 3^3 and 2^3 along with 16, 32, and 16 filters are used. The output is then fed into a fully connected network with 2000, 100, and 1 nodes. LayerNorm layers and LeakyReLU (slope: -0.2) are sandwiched in between convolutional layers.

The WGAN is trained for a total of 131k weight updates which corresponds to 20 epochs. The generator and critic network are trained using the Adam optimizer with an initial learning rate of 10^{-4}. The learning rate is decreased by a factor of 10 each after the first 50k and after a total of 100k iterations. For the critic, the initial learning rate is 10^{-5}. It is reduced by a factor of 10 after 50k iterations. Finally, the constrainer network is trained using stochastic gradient descent [66] with a learning rate of 10^{-5}. After 30k iterations, the constrainer weights are frozen. The training of the WGAN took one week on three NVIDIA Tesla V100 GPUs.

BIB-AE Training

Our implementation of the BIB-AE architecture consists of an encoder and a decoder, a latent space critic, a pair of critic and difference critic, and a network for post processing, and has 71M weights in total. Of these, 35M weights

are used by the encoder. This is a significantly larger number of weights than what can be found in the GAN and WGAN models, however this can largely be attributed to the use of fully connected layers in the BIB-AE, while both GANs are almost purely convolutions. Regardless of this weight discrepancy both models remain comparable, since their total computing time is in the same order of magnitude, as can be seen in Table 1.

The encoder consists of four 3-dimensional convolution layers with kernel size 4^3, 4^3, 4^3 and 3^3, stride 2, 2, 2 and 1 and 8, 16, 32 and 64 filters. After each convolution LayerNorm is applied. The final convolution has an output shape of $64 \times 5 \times 5 \times 5$. This output is flattened, concatenated with the true energy label, and passed to a series of dense layers with 8001, 4000, 32 and 2×24 nodes. The two sets of 24 final outputs are interpreted as μ and σ and are used to define 24 Gaussian distributions. We sample once from each Gaussian to form the latent representation of the input shower. These 24 values are passed to the decoder.

The decoder takes the 24 latent-samples and concatenates them with 488 points of random Gaussian noise as well as the true energy label. The resulting tensor is then passed to dense layers with 513, 768, 4000 and 8000 nodes. We reshape the output of the dense layers to $8 \times 10 \times 10 \times 10$. Using two transposed convolution layers with kernel sizes 3^3 and 3^3, strides 3 and 2, and 8 and 16 filters respectively this is upsampled to $16 \times 60 \times 60 \times 60$ and then reduced back down to $8 \times 30 \times 30 \times 30$ by a kernel-size 2^3, stride 2 convolution. This is followed by four more convolutions, all with kernel-size 3^3 and stride 1 with 8, 16, 32, and 1 filters respectively. Once again each (transposed) convolution except for the last one is followed by LayerNorm. Both encoder and decoder use LeakyReLU as intermediate activation functions. The final encoder layer has a linear, the final decoder layer a ReLU activation.

The BIB-AE latent space critic is a fully connected network with 1, 50, 100, 50, and 1 nodes using LeakyReLU activation. The critic is trained using samples from a Normal distribution as true data and using the latent space samples as fakes. Each of the 24 sampled latent space variables is passed individually to the critic.

The BIB-AE critic and difference critic are built as a combined network with four input streams. The first stream takes the $30 \times 30 \times 30$ shower image as input and applies three convolutions with kernel-size 3^3, 3^3, and 3^3, stride 2, 2, and 1, and 128, 128, and 128 filters, reducing the input to $128 \times 4 \times 4 \times 4$. The convolutions are interspersed with LayerNorms. The convolutional output is flattened and passed to a dense layer with 64 output nodes. The second stream is nearly identical to the first one, except the input is scaled by adding one and applying the natural logarithm. The third stream consists of a single dense layer with $30^3 = 27{,}000$ input and 64 output nodes. The input to this stream is the flattened difference between the reconstructed image and the original image. Finally, we use the true energy label as input to the fourth stream. It consists of one dense layer with one input and 64 outputs.

The 64 outputs from each of the four streams are concatenated and passed to a final set of dense layers with 256, 128, 128, 128, 1 nodes. We once again use LeakyReLU everywhere except for the final layer, which has a linear activation. During training the first two streams receive Geant4 images as real data and reconstructed images as fakes. The third stream receives Geant4-Geant4 as real and Geant4-reconstructed as fake. The fourth stream always receives the true energy label.

The Post Processor Network also has two streams. The first takes a $30 \times 30 \times 30$ image as its input and applies a kernel-size 1^3, stride 1 convolution with 128 filters. The second one takes the true energy label and the sum over all pixels in the input image as its input. These are passed to dense layers with 2, 64, 64, 64 nodes, the output of which is expanded to a $64 \times 30 \times 30 \times 30$ shape. The tensor is then concatenated along the filter dimension with the $128 \times 30 \times 30 \times 30$ output of the first stream. The combined object is passed to five more convolutions, all with kernel-size 1^3, stride 1 and 128, 128, 128, 128, and 1 filters. As before, convolutions are interspersed with LayerNorms. We use LeakyReLU save for the last layer which uses a linear activation. The use of kernel-size 1^3 means that the same function is applied to every pixel value. However the intermittent LayerNorms cause the precise functions to be different for each individual shower as well as for each pixel within the showers. As a result, each shower has its own set of 27,000 functions that behave very similarly, but are still tailored to each of the 27,000 possible pixel positions.

The setup is initially trained for 35 epochs without the Post Processor, the evolution of the individual loss contributions during this training is shown in Fig. 12. The initial learning rates are 0.5×10^{-3} for encoder, decoder and the critic, and 2.0×10^{-3} for the latent critic. All learning rates decay by 0.95 after each epoch. For each encoder/decoder update we update the critics five times. After these 35 epochs we train the Post Processor for one epoch using only the MSE term. This ensured the Post Processors baseline behaviour is to make as little changes to the images as possible. For three subsequent epochs the Post Processor is trained using a combination of MSE and MMD, with the same learning rate as the encoder/decoder. The initial 35 epochs of training took 3 days on four NVIDIA Tesla V100 (32 GB) GPUs and the Post Processor training lasted for one additional day. We save checkpoints after each epoch. A composite figure of merit combining a number of 1D distributions was used to evaluate when stopping was warranted and to select which checkpoint shows the best agreement with the training data.

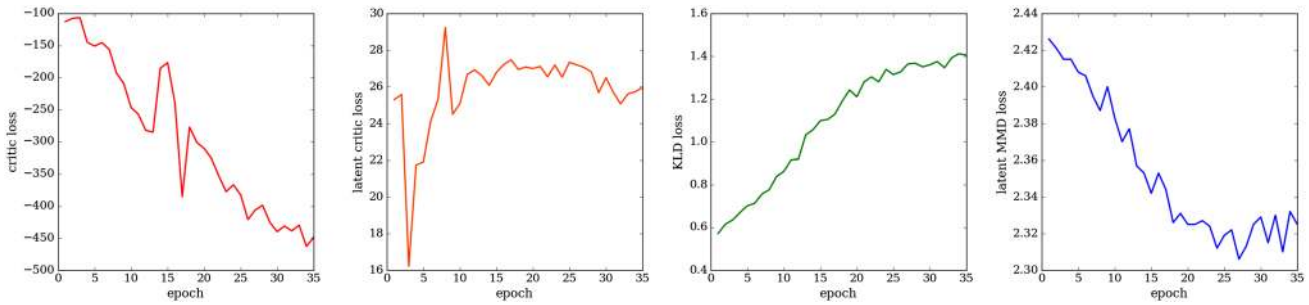

Fig. 12 Evolution of the individual loss contributions during the BIB-AE training. From left to right: critic loss, latent critic loss, KLD loss and latent MMD loss

Acknowledgements The authors would like to thank Martin Erdmann, Tobias Golling, Tilman Plehn, David Shih, and Slava Voloshynovskiy for encouraging discussions and for providing valuable feedback on the manuscript. We especially thank Ben Nachmann for his suggestions to improve the GAN training. We would also like to thank the Maxwell and National Analysis Facility (NAF) computing centers at DESY for the smooth operation and technical support. E. Buhmann is funded by the German Federal Ministry of Science and Research (BMBF) via *Verbundprojekts 05H2018 - R&D COMPUTING (Pilotmaßnahme ErUM-Data) Innovative Digitale Technologien für die Erforschung von Universum und Materie*. S. Diefenbacher is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy—EXC 2121 "Quantum Universe"—390833306. E. Eren is funded through the Helmholtz Innovation Pool project AMALEA that provided a stimulating scientific environment for parts of the research done here.

Funding Open Access funding enabled and organized by Projekt DEAL.

Data Availability Statement This manuscript has associated data in a data repository. [Authors' comment: Available at https://doi.org/10.5281/zenodo.3826103]

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Paganini M, de Oliveira L, Nachman B (2018) Accelerating science with generative adversarial networks: an application to 3D particle showers in multilayer calorimeters. Phys Rev Lett 120(4):042003. https://doi.org/10.1103/PhysRevLett.120.042003. arXiv:1705.02355 [hep-ex]
2. Kasieczka G, Plehn T et al (2019) The machine learning landscape of top taggers. SciPost Phys 7:014. https://doi.org/10.21468/SciPostPhys.7.1.014. arXiv:1902.09914 [hep-ph]
3. Heimel T, Kasieczka G, Plehn T, Thompson JM (2019) QCD or What? Sci Post Phys 6(3):030. https://doi.org/10.21468/SciPostPhys.6.3.030. arXiv:1808.08979 [hep-ph]
4. Farina M, Nakai Y, Shih D (2020) Searching for new physics with deep autoencoders. Phys Rev D 101(7):075021. https://doi.org/10.1103/PhysRevD.101.075021. arXiv:1808.08992 [hep-ph]
5. Cerri O, Nguyen TQ, Pierini M, Spiropulu M, Vlimant JR (2019) Variational Autoencoders for New Physics Mining at the Large Hadron Collider. JHEP 05:036. https://doi.org/10.1007/JHEP05(2019)036. arXiv:1811.10276 [hep-ex]
6. Collins JH, Howe K, Nachman B (2018) Anomaly detection for resonant new physics with machine learning. Phys Rev Lett 121(24):241803. https://doi.org/10.1103/PhysRevLett.121.241803. arXiv:1805.02664 [hep-ph]
7. Hajer J, Li YY, Liu T, Wang H (2020) Novelty detection meets collider physics. Phys Rev D 101(7):076015. https://doi.org/10.1103/PhysRevD.101.076015. arXiv:1807.10261 [hep-ph]
8. Amram O, Suarez CM (2020) Tag N' Train: a technique to train improved classifiers on unlabeled data. arXiv:2002.12376 [hep-ph]
9. Nachman B, Shih D (2020) Anomaly detection with density estimation. Phys Rev D 101:075042. https://doi.org/10.1103/PhysRevD.101.075042. arXiv:2001.04990 [hep-ph]
10. Andreassen A, Nachman B, Shih D (2020) Simulation assisted likelihood-free anomaly detection. arXiv:2001.05001 [hep-ph]
11. Knapp O, Dissertori G, Cerri O, Nguyen TQ, Vlimant JR, Pierini M (2020) Adversarially learned anomaly detection on CMS open data: re-discovering the top quark. arXiv:2005.01598 [hep-ex]
12. ATLAS Collaboration, Aad G, et al (2020) Dijet resonance search with weak supervision using $\sqrt{s} = 13$ TeV pp collisions in the ATLAS detector. arXiv:2005.02983 [hep-ex]
13. Agostinelli S et al (2003) Geant4–a simulation toolkit. Nucl Instrum Methods Phys Res Sect A Accelerators, Spectrom Detect Assoc Equip 506(3):250. https://doi.org/10.1016/S0168-9002(03)01368-8. http://www.sciencedirect.com/science/article/pii/S0168900203013688

14. Jansky R (2015) The ATLAS Fast Monte Carlo production chain project. J Phys Conf. Ser 664(7):072024. https://doi.org/10.1088/1742-6596/664/7/072024
15. CMS Collaboration (2017) The Phase-2 Upgrade of the CMS Endcap Calorimeter. Tech. Rep. CERN-LHCC-2017-023. CMS-TDR-019, CERN, Geneva. https://cds.cern.ch/record/2293646
16. Goodfellow IJ et al (2014) Generative adversarial nets. In: Proceedings of the 27th International Conference on Neural Information Processing Systems—vol 2. NIPS'14, pp 2672–2680. arXiv:1406.2661 [stat.ML]. https://dl.acm.org/doi/10.5555/2969033.2969125
17. Karras T, Laine S, Aila T (2019) A style-based generator architecture for generative adversarial networks. In: 2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 4396–4405. https://doi.org/10.1109/CVPR.2019.00453. arXiv:1812.04948 [cs.NE]
18. Kingma DP, Welling M (2013) Auto-encoding variational Bayes. arXiv:1312.6114 [stat.ML]
19. Dinh L, Krueger D, Bengio Y (2014) NICE: non-linear independent components estimation. arXiv:1410.8516 [cs.LG]
20. Dinh L, Sohl-Dickstein J, Bengio S (2016) Density estimation using real NVP. arXiv:1605.08803 [cs.LG]
21. Rezende DJ, Mohamed S (2015) Variational inference with normalizing flows. In: Proceedings of the 32nd International Conference on International Conference on Machine Learning—vol 37. ICML'15, pp 1530–1538. arXiv:1505.05770 [stat.ML]
22. Papamakarios G, Nalisnick E, Rezende DJ, Mohamed S, Lakshminarayanan B (2019) Normalizing flows for probabilistic modeling and inference. arXiv:1912.02762 [stat.ML]
23. Brehmer J, Cranmer K (2020) Flows for simultaneous manifold learning and density estimation. arXiv:2003.13913 [stat.ML]
24. Voloshynovskiy S, Kondah M, Rezaeifar S, Taran O, Holotyak T, Rezende DJ (2019) Information bottleneck through variational glasses. arXiv:1912.00830 [cs.CV]
25. de Oliveira L, Paganini M, Nachman B (2017) Learning particle physics by example: location-aware generative adversarial networks for physics synthesis. Comput Softw Big Sci 1(1):4. https://doi.org/10.1007/s41781-017-0004-6. arXiv:1701.05927 [stat.ML]
26. Paganini M, de Oliveira L, Nachman B (2018) CaloGAN : simulating 3D high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks. Phys Rev D 97(1):014021. https://doi.org/10.1103/PhysRevD.97.014021. arXiv:1712.10321 [hep-ex]
27. Erdmann M, Geiger L, Glombitza J, Schmidt D (2018) Generating and refining particle detector simulations using the Wasserstein distance in adversarial networks. Comput Softw Big Sci 2(1):4. https://doi.org/10.1007/s41781-018-0008-x. arXiv:1802.03325 [astro-ph.IM]
28. Erdmann M, Glombitza J, Quast T (2019) Precise simulation of electromagnetic calorimeter showers using a Wasserstein generative adversarial network. Comput Softw Big Sci 3(1):4. https://doi.org/10.1007/s41781-018-0019-7. arXiv:1807.01954 [physics.ins-det]
29. Belayneh D et al (2019) Calorimetry with deep learning: particle simulation and reconstruction for collider physics. arXiv:1912.06794 [physics.ins-det]
30. ATLAS Collaboration (2018) Deep generative models for fast shower simulation in ATLAS. Tech. Rep. ATL-SOFT-PUB-2018-001, CERN, Geneva. http://cds.cern.ch/record/2630433
31. ATLAS Collaboration (2019) VAE for photon shower simulation in ATLAS. Tech. Rep. ATL-SOFT-SIM-2019-007, CERN. https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PLOTS/SIM-2019-007/
32. ATLAS Collaboration, Ghosh A (2019) Deep generative models for fast shower simulation in ATLAS. In: 19th International Workshop on Advanced Computing and Analysis Techniques in Physics Research, Saas Fee, Switzerland (ATL-SOFT-PROC-2019-007). https://cds.cern.ch/record/2680531
33. SHiP, Ahdida C et al (2019) Fast simulation of muons produced at the SHiP experiment using Generative Adversarial Networks. JINST 14:P11028. https://doi.org/10.1088/1748-0221/14/11/P11028. arXiv:1909.04451 [physics.ins-det]
34. Bothmann E, Debbio L (2019) Reweighting a parton shower using a neural network: the final-state case. JHEP 01:033. https://doi.org/10.1007/JHEP01(2019)033. arXiv:1808.07802 [hep-ph]
35. Monk J (2018) Deep learning as a Parton shower. JHEP 12:021. https://doi.org/10.1007/JHEP12(2018)021. arXiv:1807.03685 [hep-ph]
36. Andreassen A, Feige I, Frye C, Schwartz MD (2019) JUNIPR: a framework for unsupervised machine learning in particle physics. Eur Phys J C 79(2):102. https://doi.org/10.1140/epjc/s10052-019-6607-9. arXiv:1804.09720 [hep-ph]
37. Carrazza S, Dreyer FA (2019) Lund jet images from generative and cycle-consistent adversarial networks. Eur Phys J C 79(11):979. https://doi.org/10.1140/epjc/s10052-019-7501-1. arXiv:1909.01359 [hep-ph]
38. Badger S, Bullock J (2020) Using neural networks for efficient evaluation of high multiplicity scattering amplitudes. arXiv:2002.07516 [hep-ph]
39. Klimek MD, Perelstein M (2018) Neural network-based approach to phase space integration. arXiv:1810.11509 [hep-ph]
40. Bendavid J (2017) Efficient Monte Carlo integration using boosted decision trees and generative deep neural networks. arXiv:1707.00028 [hep-ph]
41. Bothmann E, Janßen T, Knobbe M, Schmale T, Schumann S (2020) Exploring phase space with neural importance sampling. arXiv:2001.05478 [hep-ph]
42. Musella P, Pandolfi F (2018) Fast and accurate simulation of particle detectors using generative adversarial networks. Comput Softw Big Sci. https://doi.org/10.1007/s41781-018-0015-y. arXiv:1805.00850 [hep-ex]
43. Otten S et al (2019) Event generation and statistical sampling for physics with deep generative models and a density information buffer. arXiv:1901.00875 [hep-ph]
44. Hashemi B, Amin N, Datta K, Olivito D, Pierini M (2019) LHC analysis-specific datasets with generative adversarial networks. arXiv:1901.05282 [hep-ex]
45. Di Sipio R, Faucci Giannelli M, Ketabchi Haghighat S, Palazzo S (2019) DijetGAN: a generative-adversarial network approach for the simulation of QCD Dijet events at the LHC. JHEP 08:110. https://doi.org/10.1007/JHEP08(2019)110. arXiv:1903.02433 [hep-ex]
46. Butter A, Plehn T, Winterhalder R (2019) How to GAN LHC Events. Sci Post Phys 7(6):075. https://doi.org/10.21468/SciPostPhys.7.6.075. arXiv:1907.03764 [hep-ph]
47. Gao C, Höche S, Isaacson J, Krause C, Schulz H (2020) Event generation with normalizing flows. Phys Rev D 101(7):076002. https://doi.org/10.1103/PhysRevD.101.076002. arXiv:2001.10028 [hep-ph]
48. Butter A, Plehn T, Winterhalder R (2019) How to GAN event subtraction. arXiv:1912.08824 [hep-ph]
49. Bellagente M, Butter A, Kasieczka G, Plehn T, Winterhalder R (2019) How to GAN away detector effects. arXiv:1912.00477 [hep-ph]
50. ILD Concept Group, Abramowicz H et al (2020) International large detector: interim design report. arXiv:2003.01116 [physics.ins-det]
51. Qasim SR, Kieseler J, Iiyama Y, Pierini M (2019) Learning representations of irregular particle-detector geometry with distance-weighted graph networks. Eur Phys J C 79(7):608. https://doi.

org/10.1140/epjc/s10052-019-7113-9. arXiv:1902.07987 [physics.data-an]
52. iLCSoft Project Page (2016). https://github.com/iLCSoft
53. Frank M, Gaede F, Grefe C, Mato P (2014) DD4hep: a detector description toolkit for high energy physics experiments. J Phys Conf Ser 513:022010. https://doi.org/10.1088/1742-6596/513/2/022010
54. Baldi P, Cranmer K, Faucett T, Sadowski P, Whiteson D (2016) Parameterized neural networks for high-energy physics. Eur Phys J C 76(5):235. https://doi.org/10.1140/epjc/s10052-016-4099-4. arXiv:1601.07913 [hep-ex]
55. Salimans T, Goodfellow I, Zaremba W, Cheung V, Radford A, Chen X (2016) Improved techniques for training GANs. arXiv:1606.03498 [cs.LG]
56. Cédric V (2009) Optimal transport: old and new. Springer, Berlin
57. Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, Courville A (2017) Improved training of Wasserstein GANs. Adv Neural Inf Process Syst 30:5767–5777. arXiv:1704.00028 [cs.LG]. http://papers.nips.cc/paper/7159-improved-training-of-wasserstein-gans.pdf
58. Makhzani A, Shlens J, Jaitly N, Goodfellow I, Frey B (2015) Adversarial autoencoders. arXiv:1511.05644 [cs.LG]
59. Gretton A, Borgwardt KM, Rasch MJ, Schölkopf B, Smola AJ (2008) A Kernel method for the two-sample problem. CoRR. arXiv:0805.2368 [cs.LG]
60. Paszke A et al (2019) PyTorch: an imperative style, high-performance deep learning library. Adv Neural Inf Process Syst 32:8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
61. Nair V, Hinton GE (2010) Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th International Conference on International Conference on Machine Learning. ICML'10, pp 807–814
62. Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. In: Proceedings of the 32nd International Conference on Machine Learning, vol 37. pp 448–456. arXiv:1502.03167 [cs.LG]
63. Maas AL, Hannun AY, Ng AY (2013) Rectifier nonlinearities improve neural network acoustic models. In: Proceedings of ICML workshop on deep learning for audio, speech and language processing
64. Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. arXiv:1412.6980 [cs.LG]
65. Ba JL, Kiros JR, Hinton GE (2016) Layer normalization. arXiv:1607.06450 [stat.ML]
66. Ruder S (2016) An overview of gradient descent optimization algorithms. arXiv:1609.04747