Grant Agreement No: 101004761

AIDAinnova

Advancement and Innovation for Detectors at Accelerators

Horizon 2020 Research Infrastructures project AIDAINNOVA

DELIVERABLE REPORT

FAST SHOWER SIMULATION IN GEANT4

DELIVERABLE: D12.2

Document identifier:	AIDAinnova_D12.2
Due date of deliverable:	End of Month 45 (Dec 24)
Report release date:	20/12/2024
Work package:	WP12: Software for Future Detectors
Lead beneficiary:	CERN
Document status:	Final

Abstract:

Detailed simulation of showers in calorimeters is often the most time-consuming part of computing for high energy physics (HEP) experiments. Instead of the expensive multi-step particle tracking computation, one can develop models that generate the energy deposits in the calorimeters according to a parameterised model. Machine learning (ML) techniques provide an advanced technique that can encapsulate a very sophisticated parameterisation and thus reproduce particle showers. We describe different ML models that we have developed either as generic detector-independent parameterisation, tested on e.g Future Circular Collider (FCC) detectors, or specifically designed for the International Large Detector (ILD). These models need to be integrated within the C++ framework of the experiment, as a part of the Geant4 simulation, replacing computationally costly parts of the simulation. For this purpose, a Geant4 example has been released with the detailed implementation of the necessary components, and, following it, a DD4hep-specific implementation has been prepared in the form of a DDFastShowerML library, released with the Key4hep stack, allowing all future experiments to employ ML models for fast shower simulation.

AIDAinnova Consortium, 2024

For more information on AIDAinnova, its partners and contributors please see http://aidainnova.web.cern.ch/

The Advancement and Innovation for Detectors at Accelerators (AIDAinnova) project has received funding from the European Union's Horizon 2020 Research and Innovation programme under Grant Agreement no. 101004761. AIDAinnova began in April 2021 and will run for 4 years.

Delivery Slip

	Name	Partner	Date
Authored by	A. Zaborowska, P. McKeown, P. Raikwar	CERN	06/12/2024
Edited by	I. Zaborowska [Task coordinator]	CERN	09/12/2024
Reviewed by	G.A, Stewart [WP coordinator] Frank Gaede[WP coordinator]	CERN DESY	13/12/2024
Approved by	Paolo Giacomelli [Scientific coordinator] Steering Committee	INFN	20/12/2024

TABLE OF CONTENTS

Contents

Executive summary

This report documents the components developed in order to run a fast simulation of calorimeters for future collider experiments. This includes different machine learning (ML) models as well as the infrastructure that enables the use of fast simulation in the frameworks of future experiments, integrating fast simulation with the full simulation performed with Geant4.

The first section introduces the overall workflow of the development and use of fast simulation, showing how all the components are interlinked.

The second section is dedicated to all the ML models that we developed, presenting those that are general and detector-independent (Par04VAE, and CaloDiT), as well as models developed specifically for the International Large Detector (BIBAE, and CaloClouds).

The third section describes the integration of these parameterised models into the simulation framework that was implemented in the form of a Geant4 example. In order to facilitate the use of ML models for future experiments, we also present the new DDFastShowerML library, released in Key4hep, which allows everyone to use our open-source models within a full simulation application, and can facilitate the integration of any new models in the future.

1. INTRODUCTION

Simulation plays a vital role in high energy physics (HEP) experiments. It is used heavily to optimise the design of future detectors and is necessary to compare against collision data when the experiment is in operation. The computing resources dedicated to experiments are typically dominated by the simulation of the passage of particles through the detector, with the majority of the simulation time being spent in the calorimeter system [1]. This can be easily explained by the number of particles produced in the resulting cascades in the calorimeter (either electromagnetic or hadronic), with each particle tracked step-by-step by a standard toolkit for simulation of particles, Geant4 [2]. In order to reduce this use of resources, this computing-heavy part of the cascade (shower) simulation can be parameterised, directly creating the energy deposits in the detector.

Different parameterisation models have been developed in the community for shower simulation over the past years. Recently, most of them are machine-learning (ML) models, with the summary of the latest models presented in [3, 4]. The goal is to find a model that is sufficiently accurate and sufficiently fast, with different requirements depending on the target detector granularity. In section 2 we present four models that we have developed: Par04VAE, suitable for low-granularity detectors, and CaloDiT, BIBAE, and CaloClouds, for high-granularity calorimetry.

All parameterisation models use a detailed simulation to create the training data for the model: the values of the energy deposited in the detector, alongside the position of those deposits. Given the high energies of the incident particles considered for the parameterisation (up to 1 TeV for proton-proton collisions, and up to 100 GeV for electron-positron collisions), it is unrealistic to consider every single energy deposit from the detailed simulation of such energetic showers. Voxelisation, or clustering, of individual deposits must be performed, and the resulting output used to create datasets that are used to train the ML models. Voxelised data can be seen as a 3D picture of the shower (either in Cartesian or cylindrical coordinates, Fig. 1.1(left, middle)) with a fixed number of voxels, including values of 0 in voxels where no energy was deposited, and the sum of all contained energy deposits within other voxels. Clustering allows energy deposits that lie close to one another to be grouped (with various options for the clustering algorithm), with the possibility that the output contains a variable number of energy values each positioned at the centre of each cluster. This is termed a point cloud and depicted in Fig. 1.1 (right). Different ML models are designed for either fixed-length or variable inputs. The goal of those models is to learn to generate the showers that mimic the input training data.

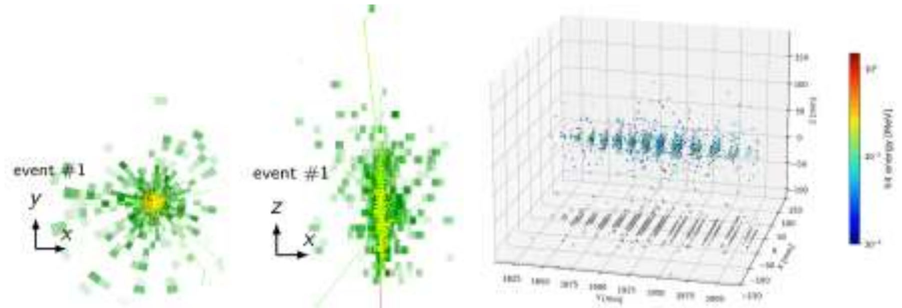

Figure 1.1: Photon shower (left, middle) in the voxelised cylindrical mesh and (right) represented as a point cloud in the ILD ECAL.

The showers that a model generates, no matter in which representation, need to be placed back into the detector, so that an assignment of energy values to the physical cells in the detector can be made. This step enables a fair measurement of the speed-up of the fast simulation, as it replaces the detailed (full) simulation by producing an output with an identical form (i.e. a collection of hits in the sensitive detector). The time spent to assign the deposits may be non-negligible, setting an upper limit on the potential speed-up [7]. It also enables a full validation to be performed, at reconstruction level and with an analysis of key physics observables, which will be presented for the BIBAE model in Sec. 2.3. Most models developed in the community are only validated at the simulation level, and often in their Python training environment. This does not account for many detector effects, as well as preventing any benchmarks including multiple particles, which hides what is of actual importance for a given detector. Inference performed in these Python environments, often with unrealistically large batches of showers, gives a misleading impression of the actual speed-up. The estimation of per-event batch sizes, which seamlessly fit in the current frameworks of experiments is given in [7]. Especially for electron-positron collisions, there is little potential for gain if current processing workflows are employed in the future. This highlights the importance of studying ML fast sim models in the simulation frameworks of experiments, which enables conclusions to be drawn about how events should be processed. The integration of models into simulation frameworks is presented in Sec. 3. First integration into the standalone Geant4 application is covered, in the form of an example called Par04 [5], which was described in detail in the Milestone report [6]. Then, an integration with the DD4hep toolkit is described, in the package called DDFastShowerML [2.2.13]. While the Par04 example shows how to produce the voxelised data, integrate a VAE model with several inference libraries, and run inference within Geant4, the DDFastShowerML focuses on the pieces needed for integration of the models, i.e., inference, within the realistic and irregular geometries that can be defined in DD4hep. In order to produce voxelised datasets as it is done in Par04, a small package called ddfastsim has been implemented [8].

The datasets produced for the development of the model described in Sec. 2.1 (ParVAE) were published [9] and served as the benchmark datasets for the CaloChallenge [3]. They contain voxelised energy deposits, with the voxelisation done in cylindrical coordinates, as depicted in Fig. 1.1. The cylinder axis is aligned with the momentum direction of the incident particle with smaller voxels in the centre of the shower, where energy density is highest. The size of each voxel along the axis (longitudinally), and in the radial direction is linked to the radiation length, and the Moliere radius of the absorber, respectively. This basic principle makes electromagnetic showers in various materials look similar, which motivates the development of a general model that is pre-trained on different datasets, and can be quickly adapted to a new detector. This can save substantial computing resources and allow Geant4 to offer generic fast simulation techniques alongside the detailed simulation, as presented in [10].

2. MACHINE LEARNING MODELS FOR SHOWER PARAMETERISATION

2.1. PAR04VAE

The first model developed and published in the context of AIDAinnova is the variational autoencoder (VAE) that is released together with the example of Geant4 called Par04 [5]. The model was described in detail in the milestone report of AIDAinnova [6]. This model produces a rather smeared distribution of energy in the generated showers, however it was still successfully adopted by the LHCb Collaboration and integrated within their simulation framework. With the additional sampling of energy deposits they observe a very good agreement of reconstructed variables, and are ready to use that model in production [11].

2.1.1. Generic model: adaptation to other detectors

The capability of Par04VAE to adapt to unseen detectors, a.k.a. MetaHEP, has been studied and summarised in [12]. The speed-up factor that was observed on the previously untested detector is more than 500. This greatly reduces the computational resources necessary to train the models.

2.2. CALODIT

CaloDiT was developed in collaboration with IBM and CERN IT Openlab and is another model designed to provide a generic model for fast shower simulation. Inspired by so-called "foundation models", CaloDiT takes a different direction than MetaHEP. It has the same goal of being easy to adapt to new detectors, which will aid in reducing computational and manpower resources. Whereas MetaHEP's design of a generalizable model is algorithmically-driven, CaloDiT is data-driven. Recently, there has been an emergence of foundation models in the ML literature which excel at performing a variety of tasks, even new ones, with only a slight amount of tuning, making these models useful for general purposes. Some examples include GPT-3 [13], Imagen [14], etc. The idea behind them is to train on a large and diverse dataset to learn robust representations of data that can be used for multiple tasks. This requirement fits in quite well given that our datasets are based on a simulator. First, we explain the model for a single detector (Par04) and discuss the multi-geometry training and adaptation in the next subsection. CaloDiT updates were presented at ACAT'24 [15] and ML4Jets'24 [10].

Usually, the data representation of the showers varies with the detector, based on physical cells. However, since we are interested in applying the same model to multiple geometries, we need to capture the shower energies irrespective of the physical attributes of the detector. To achieve this, we use the virtual cylindrical mesh from Par04. Along with varying the range of energy of the incident photon, we also vary the angles covering most of the regions in the detector to make the dataset more diverse. 1M photon showers were simulated in the Par04-SiW detector with the same granularity as the CaloChallenge Dataset-2 [3]. Thus, the dataset differs in the number of conditions and number of samples when compared to the CaloChallenge dataset.

As for the model, following recent advancements in the generative ML literature, we choose the diffusion process as the generative process for CaloDiT. These diffusion-based models are shown to be easy to optimize while offering superior results than variational autoencoder or generative adversarial networks [16]. However, diffusion-based models tend to be slower because of the iterative process of transforming a sample from a known distribution to the desired data distribution. To mitigate this, we use consistency distillation [17] to reduce these iterative steps from 32 to just a single step. This distillation process requires the trained standard diffusion model and outputs a distilled model that is significantly faster with a minimal accuracy loss.

Given the non-trivial data representation of the showers, we adopt a transformer-based architecture [18] as a transformer in a generic architecture for almost any kind of data [19]. In particular, the architecture (Figure 2.1.1) is based on DiT-blocks [20] with a few modifications to allow for a 3-dimensional "shower image".

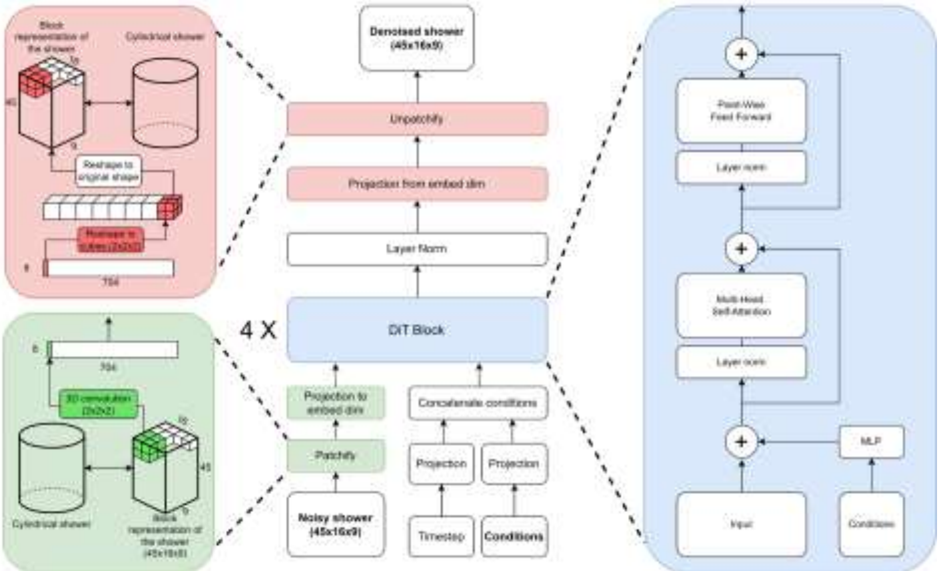

Figure 2.1.1: The architecture of CaloDiT. A noisy shower (Gaussian noise) is passed to the model (stack of multiple DiT blocks), and the output is a denoised shower (middle). Splitting the 3D shower into smaller 3D patches and back (left). DiT block where sample conditions are passed along with timestep condition (right).

It is straightforward to optimize diffusion models compared to GANs while offering significantly better modelling of shower observables compared to VAEs. We see a very good overlap between CaloDiT and Geant4, especially in the distribution of voxel energies in the virtual mesh. Getting this distribution accurately was not feasible for Par04VAE and the previous VQVAE-based model [21]. Other shower observables are also very well modelled. We do see a minor degradation in accuracy in the distilled model, but with the benefit of an almost 32 times speedup in generation time. These shower observables are shown in Figure 2.1.2. Further, we also do a benchmark on the inference timing of CaloDiT, shown in Figure 2.1.3. CaloDiT is slightly slower than Geant4 on a single CPU thread in the small low-energy region, but offers significant speedups at higher energies. This speedup also considers the time to map the energy back into the physical detector cells, as per [22].

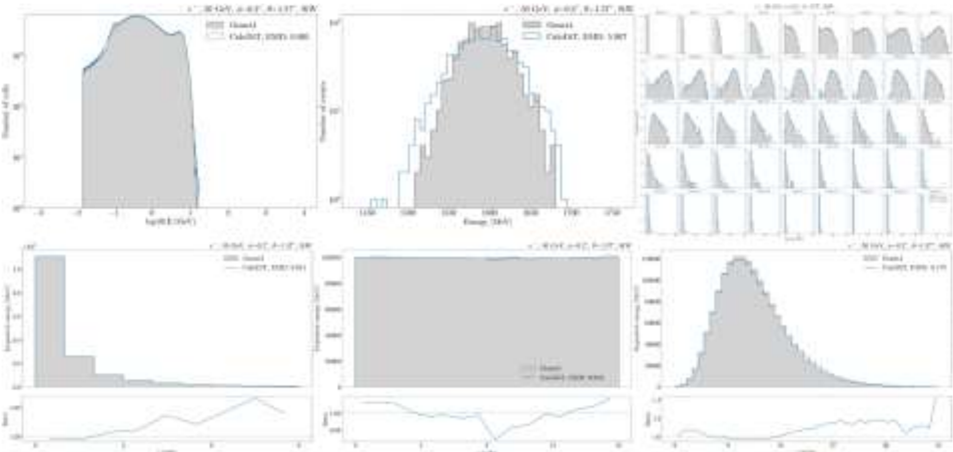

Figure 2.1.2: The shower observables in the virtual mesh for the Par04-SiW detector with the CaloDiT distilled version. Voxels energy, total energy, layer-wise energy, lateral profile, phi profile, and longitudinal profile (from top-left to bottom-right) show good agreement with Geant4.

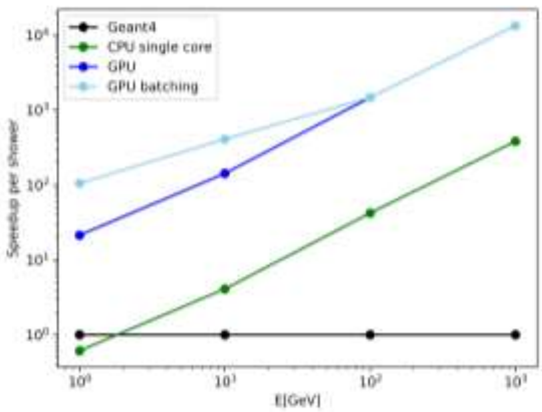

Figure 2.1.3: CaloDiT speedup relative to Geant4 on CPU single-core (green), GPU (blue), GPU with batching (sky blue).

2.2.1. Generic model: adaptation to other detectors

In order to generalise the model to other detectors, the model needs to understand how showers differ between detectors. We choose a rather simple way to do so, providing the detector condition to the model as a category. First, the pretraining is done on *K* categories, and then the model is adapted to the $K+1^{th}$ detector, which is the new target detector. In our case, we utilize 4 detectors - Par04SiW, Par04SciPb, ODD, FCCeeCLD for pretraining, and FCCeeALLEGRO as the target for adaptation.

We generate the dataset in a similar fashion to the previous subsection but use ddFastSim [8] which makes use of DD4Hep to simulate showers in different detectors. Adaptation refers to the finetuning of an ML model with a lower learning rate than normal. This avoids the loss of information obtained during the pretraining phase by not aggressively changing the parameters of the model. Since CaloDiT has two phases at the outset (standard diffusion + distillation), the adaptation first requires the finetuning of the standard diffusion model, followed by the finetuning of the distilled model on the new detector dataset.

The results indicate significantly fewer gradient descent steps are required when the pretrained model is used, as opposed to training from scratch (Figure 2.2.1 shows 10x). Furthermore, good results can be obtained using a significantly smaller dataset, which otherwise is not possible if the model is trained from scratch (typically also 10x). When combined, and depending on how much the desired detector deviates from the pretraining detector, the total reduction in computing resources can range from 25-100 times.

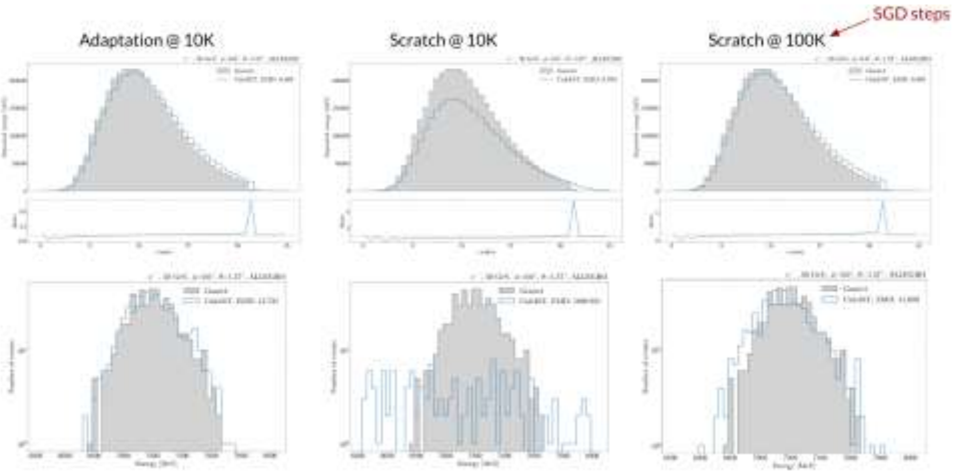

Figure 2.2.1: Adaptation of CaloDiT on the FCCeeALLEGRO detector. We find 10x fewer optimization steps (training steps) are required when we use a pretrained CaloDiT for adaptation (left), compared to training from scratch (right). If the same number of optimization steps are taken for training from scratch, the model performs poorly (middle).

Finally, CaloDiT is converted to ONNX and TorchScript formats for inference in C++-based environments. One such integration is with the Par04 example in Geant4 [24], where it is readily available to generate Par04SiW showers and the same model can be adapted to any other desired detector using the provided scripts. It will be released with the next Geant4 release in 2025. Another integration is done using DDFastShowerML [23], where CaloDiT is used to generate FCCeeCLD showers (see section 3.2).

2.3. BIBAE

The Bounded Information Bottleneck Autoencoder (BIBAE), was introduced in order to provide an information-theoretical framework that encompasses many of the common Generative Adversarial (GAN) and Variational Autoencoder (VAE) architectures [25]. The BIBAE was first applied to the task of fast calorimeter shower simulation in [26], in particular for showers initiated by photons hitting orthogonally the face of the ILD highly granular ECAL represented as a regular grid (or 'shower image'). Several application specific adaptations to the architecture were necessary to achieve a high fidelity modelling of key shower observables. In particular, a dedicated post processor network, that was applied in a second training step to the showers, allowed the model to achieve a more accurate modelling of the cell energy spectrum than was possible with previous models. A schematic of the architecture is shown in Figure 2.3.1.

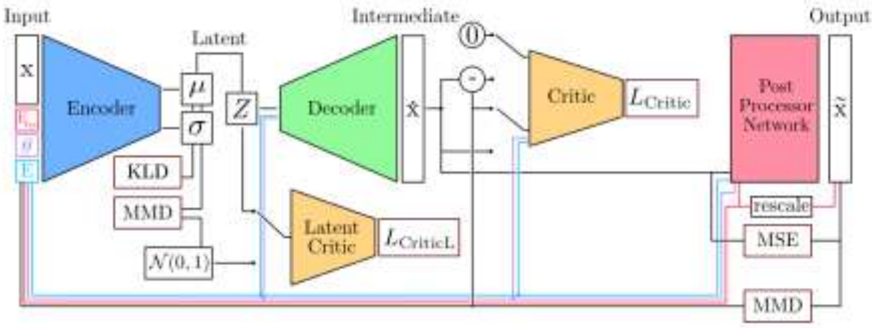

Figure 2.3.1: Schematic of the BIBAE architecture, consisting of both VAE-like and GAN-like components, together with a dedicated Post Processor Network. Figure taken from [27].

In the course of this project, this model was extended in two key directions. In the first instance, the model was adapted to hadronic shower simulation. Hadronic showers are significantly more complex than electromagnetic showers, due to the richer phenomenology present in hadronic interactions, as well as the inclusion of an electromagnetic component. This results in much more complicated topologies in a highly granular calorimeter, with branch-like structures being exhibited, and also significantly larger variations between individual events. A data set was created consisting of hadronic showers initiated by charged pions of varying energy incident perpendicular to the face of the highly granular ILD Analogue Hadronic Calorimeter (HCAL). To study the performance of the BIBAE model on this more challenging task, numerous adaptations to the model were necessary, including additional loss terms, and modifications to the post processing, among others. A comparison was drawn between Geant4, the BIBAE and a Wasserstein GAN (WGAN) trained on the same dataset. The BIBAE was again shown to provide a superior modelling of these hadronic showers compared to the WGAN, for example, providing a superior description of the cell energy spectrum, as shown in Figure 2.3.2. Additionally, the showers were interfaced with the state-of-the-art PandoraPFA reconstruction algorithm, by means of a mapping to the detector geometry, and the reconstruction performance studied. This work represents a first application of a generative model to the simulation of hadronic showers in a highly granular calorimeter [28,29].

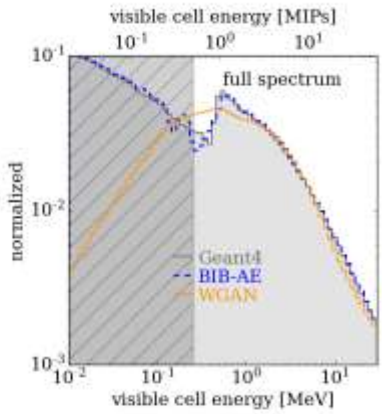

Figure 2.3.2: Cell energy spectrum for pion showers in the ILD HCAL. The BIBAE model (blue) provides a superior modelling of the Geant4 ground truth (grey) than the WGAN model (orange). Figure taken from [28] .

The second research direction pursued was the extension of the BIBAE model developed for electromagnetic shower simulation to enable its incorporation in a realistic full event simulation. To this end, it is necessary for a model to provide an appropriate detector response for particles incident with a varying energy and also at different angles. A dataset was created consisting of photons with varying incident energies and angles fired into the face of the ILD ECAL. Adaptations to the BIBAE were necessary to handle the larger phase space inherent in the dataset, and included the addition of a hybrid approach, whereby the use of a Normalising Flow (NF) model enabled a significant improvement in the modelling of the visible energy. A thorough study was conducted into the performance of the model in terms of key calorimetric observables before and after interfacing with PandoraPFA via a mapping to the detector geometry. The results of this study are published in [27].

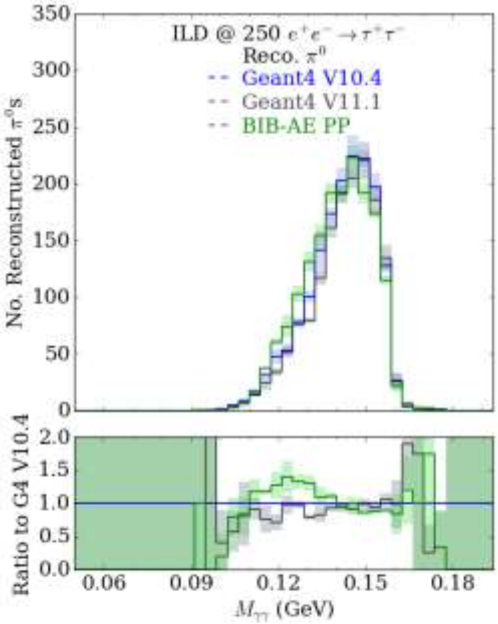

Figure 2.3.3: Number of π^0s reconstructed as a function of the invariant mass of the two photons combined into the π^0 for the process $e^+ e^- \to \tau^+\tau^-$. Distributions are shown for two Geant4 versions (blue, grey) and for a simulation incorporating a BIBAE model (green). Each distribution is the average over three different random seeds used in the simulation. Figure taken from [30].

The model was then incorporated into the DDFastShowerML library (see Section 3.2), which enabled the inclusion of the model in a full event simulation, and subsequent physics studies after interfacing with a complete event reconstruction chain. The performance of the BIBAE model was studied for use in electromagnetic shower simulation of the process $e^+ e^- \to \tau^+\tau^-$, where the τ lepton decayed hadronically. The BIBAE was found to provide a modelling of key physics observables, such as the invariant mass of neutral pions produced in the tau decays, shown in Figure 2.3.3, to a similar level to the variations present between Geant4 versions. This study formed part of a doctoral thesis, which has been published [30].

2.4. CALOCLOUDS

The majority of models explored for fast calorimeter shower simulation in the literature rely on a fixed grid representation of a shower, including the BIBAE model described in Section 2.3. This approach has two key drawbacks. Firstly, the high level of sparsity in a shower means that relying on a representation of a fixed size wastes a significant fraction of computing power in a model on empty cells. Secondly, it can introduce artifacts in the projection from an irregular detector structure to a regular structure, if the model operates directly on the detector readout. The CaloClouds model makes use of a point cloud representation of a shower, which consists of the position of the point in 3D space, along with the corresponding energy deposited. A photon shower in the ILD ECAL represented as a point cloud is shown in Figure 1.1. Importantly, in order to achieve a high degree of geometry independence, the model is able to operate directly on information at a lower level than the actual readout. This information is accessible in the form of Monte Carlo steps present in the full simulation provided by Geant4. The CaloClouds model itself consists of a multi-step approach. The core of the model is a point cloud diffusion model, based on [31]. Around this core, additional components operate, including an NF, which generates a latent space for the model, as well as providing a number of points to sample for a given incident particle energy and predicting the energy deposited in each calorimeter layer per-shower. Several calibrations based on the output of this model are then applied to the model, to ensure a correct modelling of showers. The model was trained on a dataset of electromagnetic showers created by photons fired perpendicular to the face of the ILD ECAL, with a range of incident energies. The performance of the model was then evaluated on a range of calorimeter observables, with the high degree of geometry independence being established by comparing the performance at two different incident positions on the face of the calorimeter, shown in Figure 2.4.2. This was the first time a full calorimeter shower, consisting of a few thousand space points, was successfully simulated in the form of a point cloud with a generative model. The results of this study were published in [32].

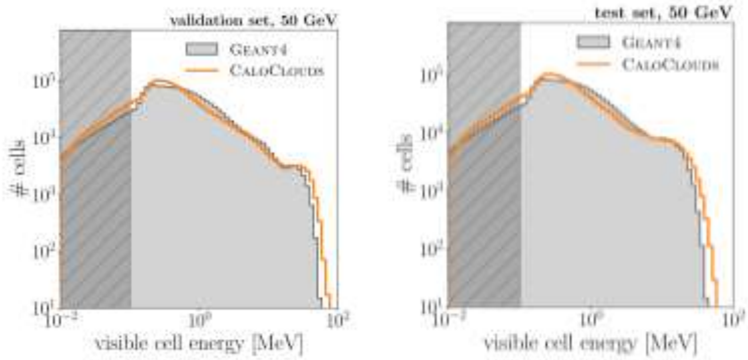

Figure 2.4.2: Cell energy spectrum for Geant4 (grey) and CaloClouds (orange) simulated showers at 50 GeV created at the same position used for training (validation, left) and a different position (test, left), with the CaloClouds generated point cloud translated to this position. Figure taken from [32].

While the original CaloClouds model is able to accurately simulate a highly granular calorimeter shower in the form of a point cloud, the model inference time (i.e. per-shower generation time) is not significantly faster than the Geant4 simulation time. This is a result of the diffusion model used, which relies on an iterative denoising procedure, requiring repeated sampling from the model. To tackle this problem, the diffusion process used was reformulated following [33], and a distillation procedure [34] applied to reduce the number of sampling steps required for the model to one. This allowed a speed up factor of 46x to be achieved over Geant4 for single shower generation on a single CPU core. The results of this study were published in [35].

Subsequently, the CaloClouds model has been extended in a similar fashion to the BIBAE, described in Section 2.3, to be able to provide an appropriate detector response for particles of various incident energies and under various incident angles to the calorimeter face. This has enabled the CaloClouds model to be incorporated into the DDFastShowerML library (Section 3.2).

2.4.1. CaloClouds on Par04 dataset

As part of the CaloChallenge 2022, the CaloClouds model was successfully applied to the third CaloChallenge dataset, which is derived from Par04. The main adaptation necessary for the model was related to the regular cell nature of the hits in the mesh, which made points more discretised than was the case for the original model, which was designed for operation on points derived from Geant4 steps. More details can be found in the final review of the CaloChallenge [3].

3. INTEGRATION OF MODELS

3.1. PAR04

The pipeline for the development of machine learning models has been released in the form of a Geant4 example, Par04. It demonstrates how to produce data, and how to run inference within the C++ toolkit. It has been described in detail in the AIDAinnova milestone report [1.6]. Numerous inference libraries have been implemented: lwtnn, ONXX runtime, LibTorch. LibTorch was developed during the first AIDAinnova hackathon in 2022, and the GPU inference was the topic of the second hackathon in 2023, both drawing on work from all the institutes collaborating on this task.

3.2. DDFASTSHOWERML

The DDFastShowerML library was developed following the example provided by Par04 to enable the use of ML fast simulation models with realistic detector geometries described with the DD4hep toolkit [36]. To enable the library to be generic, which eases the addition of new and varied models, the components of the library are factorised as far as possible. The core of the library is a class template with the following key components:

- ***Model***: The implementation of a specific model architecture. This component is responsible for the preparation of the input in the correct form for the model, as well as converting the output of the model to space points.
- ***Inference***: The component responsible for calling the inference library for the model. Currently, the library supports the use of ONNXRuntime and LibTorch
- ***Geometry***: The implementation of the detector geometry that allows the placement of space points produced by the model. This may, for example, include the positioning of calorimeter layers, as well as the conversion between global (envelope) and local coordinates. The local direction with respect to the calorimeter face is computed, in order to provide conditioning input for the model.

- ***Hit Maker***: A class provided by Geant4 which enables placement of energy deposits which lie in a sensitive region of the detector.
- ***Trigger***: An interface that lies on top of the fast simulation triggers present in Geant4 and DD4hep, and enables the termination of full physics simulation and the initiation of ML based simulation. This can also be used to exclude irregular regions of a realistic detector geometry, in which the use of the ML model may not be appropriate.

The library supports both the approaches to fast simulation presented in this report. In the first approach, following the work done for the ILD detector, showers can be placed into the detector based directly on the position of the layers in the calorimeter. In the second approach, showers can be placed into the calorimeter using the virtual cylindrical scoring mesh as used in Par04. In this case, a specific sensitive configuration of the detector geometry and corresponding sensitive detector action is used to place hits in the virtual mesh back into the detector readout. This approach has so far been tested with the CaloDiT model and the CLD detector proposed for use at FCC. Event displays of showers generated in DDFastShowerML using these two different approaches can be seen in Figure 3.2.1.

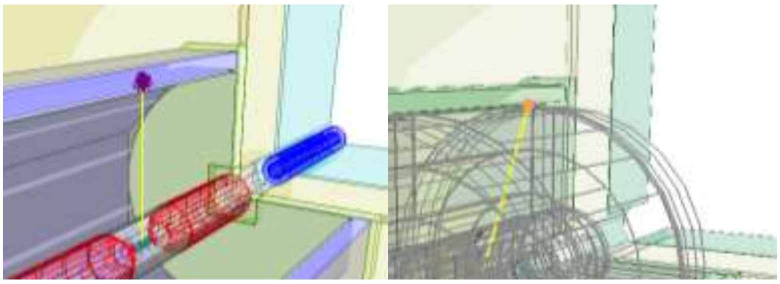

Figure 3.2.1: ML generated showers in DDFastShowerML. Left: BIBAE generated shower in the ILD detector using a layer-wise calorimeter placement (taken from [37]*). Right: CaloDiT shower generated in the CLD detector using the cylindrical scoring mesh and custom sensitive detector action.*

The library has been described in [37], and formed part of a doctoral thesis, which has been published [30].

4. CONCLUSIONS

In the context of AIDAinnova we have developed several ML models for the fast simulation of showers in calorimeters, allowing future detectors to test different models and choose the best suited one. While the Par04VAE model suffices for low-granularity calorimetry, other models need to be employed for highly granular detectors, which are those generally planned at future facilities. Thus far, the development of ML models for simulating hadronic showers has received little attention in the literature, in particular for highly granular calorimeters. In this context, we have applied the

BIBAE model to hadronic showers in a highly granular calorimeter for the first time. While this study shows significant potential, significant work lies ahead in order to be able to use an ML model for hadronic showers in a realistic simulation. In the first instance, the study presented only considered showers in a hadronic calorimeter, however, in reality, a hadronic shower can begin in the electromagnetic calorimeter and stretch into the hadronic calorimeter. This represents a significant challenge, as the readout granularity typically differs significantly between these two detectors. Potentially, a point cloud based model could be suited to address such a challenge. It would then also be necessary to develop a hadronic shower model that can also provide an appropriate detector response for particles at various angles, which has not yet been explored for hadronic showers. Other models we have developed, CaloDiT and CaloClouds, are both diffusion based models, but they operate on different data representations. The goal for CaloDiT is to provide a generic model that can be quickly adapted to new calorimeters, thus removing the need for computationally heavy training from scratch. This could prove useful for new detectors, as well for those that are not yet finalised and change often. CaloClouds provides a model able to more efficiently handle highly granular calorimeter showers than common approaches based on regular grids, as well as a route to gain a higher degree of independence from the detector geometry, thanks to the use of sub-readout level simulation information.

In order to complete a full validation of the models, and to allow their use in simulation production, we implemented a DD4hep-based library, DDFastShowerML, that integrates all of our models in the Geant4 simulation framework. This has been performed together with the subtask 1: *Turnkey Software* of WP12. Thanks to that integration, a thorough physics analysis was completed, testing BIBAE-generated electromagnetic showers produced in a full physics process. While the physics performance was not perfect, the deviations of the BIBAE based simulations from the Geant4 reference were found to be comparable to the differences between Geant4 versions. In the future, further physics processes should be explored to gauge model performance, along with more targeted multi-particle benchmarks that provide focused tests on physically relevant aspects of showers. Importantly, DDFastShowerML is designed in a way that allows new models to be easily integrated, serving future ML model developments. In the future, support for hadronic showers models should be incorporated, as well as support for batching of showers within an event together with GPU usage. This would help significantly to maximise the available simulation speedups relative to Geant4 full simulation. Additionally, the library has thus far only been used to simulate showers in the ECAL of ILD and CLD, which both utilise the same calorimeter technologies. In the future, it would be important to extend support to other calorimeter technologies, including, for example, the liquid argon calorimeter proposed for the ALLEGRO detector. This can be easily achieved with the CaloDiT model, as it utilises the underlying data structure independent of the detector, with the cylindrical mesh. The simulation-level variables achieved for ALLEGRO ECal are very good, but the ultimate validation can only be performed after the placement of deposits inside of the detector.

5. References

AIDA-credited Contributions:

[3] Krause, C. et al. (2024). CaloChallenge 2022: A Community Challenge for Fast Calorimeter Simulation. https://doi.org/10.48550/arXiv.2410.21611.

[6] Zaborowska, A. (2023) *AIDAInnova Milestone report. Prototype of ML based shower simulation*. Zenodo. doi: 10.5281/zenodo.7564612.

[7] Zaborowska, A. (2023) *Level up your performance calculation of the fast shower simulation model* [online]. Available from: https://indico.cern.ch/event/1253794/contributions/5588609/attachments/2748958/4784643/ml4jets_generalFastSim.pdf [Accessed 6 December 2024].

[10] Raikwar, P. (2024) Towards Detector Agnostic Fast Calorimetry Simulation. ML4Jets 2024. url: https://indico.cern.ch/event/1386125/contributions/6083371/.

[12] Salamani, D., Zaborowska, A., Pokorski, W. (2023). MetaHEP: Meta learning for fast shower simulation of high energy physics experiments. *Physics Letters B, 844, 138079.*

[15] Raikwar, P. (2024) CaloDiT: Diffusion with transformers for fast shower simulation. ACAT 2024. url: https://indico.cern.ch/event/1330797/contributions/5796591/.

[21] Raikwar, P. et al. (2024) Transformers for Generalized Fast Shower Simulation. In: EPJ Web of Conf. 295, p. 09039. doi: 10.1051/epjconf/202429509039. Url: https://doi.org/10.1051/epjconf/202429509039.

[22] Zaborowska, A. Fast simulation: status and use at FCC-ee. FCC Physics Workshop. url: https://indico.cern.ch/event/1307378/contributions/5729659/.

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

[28] Buhmann, E. et al. (2022) Hadrons, better, faster, stronger, *Mach. Learn.: Sci. Technol.*, 3 025014

[29] McKeown, P. et al. (2022) Generative Models for Fast Simulation of Electromagnetic and Hadronic Showers in Highly Granular Calorimeters, In: *41st International Conference on High Energy Physics*, 6-13 July, 2022, Bologna, Italy, *PoS ICHEP2022*, 236, pp. 6

[30] McKeown, P. (2024) *Development and Performance of a Fast Simulation Tool for Showers in High Granularity Calorimeters based on Deep Generative Models*, Hamburg, Germany: Verlag Deutsches Elektronen-Synchrotron DESY, DOI: 10.3204/PUBDB-2024-01825

[32] Buhmann, E. et al. (2023) CaloClouds: fast geometry-independent highly-granular calorimeter simulation, *JINST* 18 P11025

[35] Buhmann, E. et al. (2024) CaloClouds II: ultra-fast geometry-independent highly-granular calorimeter simulation, *JINST* 19 P04020

[37] McKeown, P. et al. (2023) Fast Simulation of Highly Granular Calorimeters with Generative Models: Towards a First Physics Application, In: *The European Physical Society Conference on High Energy Physics*, 21-25 August, 2023, Hamburg, Germany, *PoS EPS-HEP2023*, 568, pp. 6

Other references:

[1] Kreps, M. et al. (2024) *LHCb view - the future of simulation and simulation for the future* [online]. Available from:

Field Code Changed

https://indico.cern.ch/event/1475445/contributions/6235436/attachments/2972698/5232093/LHCbView-20241122.pdf [Accessed 5 December 2024].

[2] Allison, J., et al. (2016). Recent developments in Geant4. Nuclear Instruments and Methods in Physics Research Section A (835), 186–225. https://doi.org/10.1016/j.nima.2016.06.125

[4] Hashemi, B., and Krause, C. (2024) Deep generative models for detector signature simulation: A taxonomic review. *Reviews in Physics, 12*, p.100092.

[5] Geant4 collaboration. (2021) Par04 example: Machine learning for fast simulation in Geant4 [online]. Available from: https://gitlab.cern.ch/geant4/geant4/-/tree/master/examples/extended/parameterisations/Par04 [Accessed 6 December 2024].

[8] Zaborowska, A. (2024) *DDFastSim - data production in a cylindrical shower mesh* [online]. Available from: https://gitlab.cern.ch/fastsim/ddfastsim [Accessed 6 December 2024]

[9] Salamani, D. & Zaborowska, A. (2022). *High Granularity Electromagnetic Calorimeter Shower Images* [Data set]. Zenodo. https://doi.org/10.5281/zenodo.6082201

[11] Mazurek, M. (2024) *Generative AI for fast simulations in LHCb*. CHEP 2024. Accessible from: https://indico.cern.ch/event/1338689/contributions/6015805/ [Accessed 6 December 2024]

[13] Brown, T. B. et al. (2020) Language models are few-shot learners, arXiv:2005.14165.

[14] Saharia, C. et al. (2022) Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. arXiv: 2205.11487 [cs.CV]. url: https://arxiv.org/abs/2205.11487.

[16] Karras, T. et al. (2022) Elucidating the design space of diffusion-based generative models. In: Advances in neural information processing systems 35, pp. 26565–26577.

[17] Song, Y. et al. (2023) Consistency Models, arXiv:2303.01469

[18] Vaswani, A. (2017) Attention is all you need. In Advances in Neural Information Processing Systems.

[19] Reed, S., et al. (2022) A Generalist Agent, https://arxiv.org/abs/2205.06175.

[20] Peebles, W. and Xie, S. (2023) Scalable diffusion models with transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 4195–4205.

[23] DDfastShowerML library for ML model inference in DD4hep. url: https://gitlab.desy.de/ilcsoft/ddfastshowerml.

[24] Raikwar, P. and Zaborowska, A. "Par04 development repository: Machine learning for fast simulation in Geant4". url: https://gitlab.cern.ch/fastsim/par04/-/tree/CaloDiT_v1.

[25] Voloshynovskiy, S. et al. (2019) Information bottleneck through variational glasses, arXiv:1912.00830

[26] Buhmann, E. et al. (2021) Getting High: High Fidelity Simulation of High Granularity Calorimeters with High Speed, *Comput.Softw.Big Sci.*, 5 (1) 13

[31] Luo, S., Hu, W. (2021) Diffusion Probabilistic Models for 3D Point Cloud Generation, arXiv:2103.01458

[33] Karras, T. et al. (2022) Elucidating the Design Space of Diffusion-Based Generative Models, arXiv:2206.00364

[34] Song, Y. et al. (2023) Consistency Models, arXiv:2303.01469

[36] Frank, M. et al. (2014) DD4hep: A Detector Description Toolkit for High

Energy Physics Experiments, In: *20th International Conference on Computing in High Energy and Nuclear Physics*, 14-18 October, 2013, Amsterdam, The Netherlands, *J. Phys.: Conf. Ser.* 513 022010, pp.

Field Code Changed

ANNEX: GLOSSARY

Acronym	Definition
ML	Machine Learning
HEP	High energy physics
ILD	International Large Detector
FCC	Future Circular Collider
BIBAE	Bounded Information Bottleneck Autoencoder
GAN	Generative Adversarial Network
VAE	Variational Autoencoder
ECAL	Electromagnetic Calorimeter
HCAL	Hadronic Calorimeter
PFA	Particle Flow Algorithm
NF	Normalising Flow
DD4hep	Detector Description for High Energy Physics
CLD	CLIC-like Detector
CLIC	Compact Linear Collider
ODD	Open Data Detector