Fast Machine-Learned Simulations for Future Detectors

Dylan Smith, UC Irvine

May 7, 2023

Abstract

In modern High Energy Physics (HEP) research, simulating detector response often present a large bottleneck in simulation time. Geant4, a popular simulation software toolkit, is used to model the microphysics of particle interactions and their products in particle physics experiments. While accurate, Geant4 is computationally expensive. Machine learning (ML) offers a faster alternative by learning a map between incident particles and detector response. However, current fast ML simulation techniques require using multiple models to simulate the varied response of each detector element. This work aims to develop a versatile ML-based model that can generate simulated data for any calorimeter, without being tuned to a specific geometry. During the 2022-2023 academic year, various models were identified and the most promising models were explored.

Contents

1 Introduction

The simulation of detector response to incident particles is a critical component of the analysis of data from particle physics experiments, which probe the underlying structure of nature at its most fundamental level [1] [2]. Geant4 is a widely-used simulation software toolkit in High Energy Physics (HEP) that models the microphysics of particle interactions and their products [3] [4] [5]. While it is the gold standard in terms of accuracy, Geant4 is computationally expensive. Machine learning (ML) provides a faster alternative by learning a map between incident particles and detector response. However, current fast ML simulation techniques generally require using hundreds of separately-trained models in order to describe the varied response of each detector element. My objective is to develop a versatile ML-based model that can generate simulated data for any calorimeter, without being tuned to a specific geometry. Such a model should have the flexibility to simulate both the irregular ATLAS calorimeter geometry and other detectors, such as CMS [6] or future detectors. During the 2022-2023 academic year, numerous models were investigated for this purpose, and several promising avenues identified.

2 Methods

Generative models learn to produce novel data similar to training examples by developing a model of the probability distribution underlying the training data. A *conditional* generative model learns to produce data as a function of some specified parameters. For generation of simulated detector responses, a conditional generative model would learn to produce novel detector responses conditioned on the incident particle type and energy. Within a single detector system, such as a calorimeter, the shape and size of the cells can vary dramatically, requiring individual generative models for each component. Training such a set of models is incredibly computationally expensive and requires significant human validation of the hundreds of generative networks. This strategy represents the state of the art within ATLAS[7]. If a generative model could generate simulated responses for an arbitrary geometry with a single model, it would require vastly reduced computational and human resources.

There are two possible strategies for training such a general-purpose generative model. The first, called *geometry agnostic*, aims to learn the underlying distribution of energy deposition as if the calorimeter had no segmentation (or equivalently, had infinitely fine segmentation). Realistic data samples would be generated by applying the geomtry's segmentation as a second step. Geometry-agnostic calorimeter data can be expressed as point-clouds, which consist of (approximately) continuous-valued points in space rather than specific cell responses.

The second strategy, called *geometry-aware*, uses a generative model which is conditioned on the detector geometry. Given many examples of responses across varying geometries, the model can learn how the response varies with geometry, and be capable of generating expected detector responses for any geometry. Geometry-aware training data is represented by binning the point-cloud hits into cells, which can vary in shape and size.

There are challenges associated with both approaches. The geometry-agnostic approach avoids the need to account for difference in segmentation, but the hits can occupy any point in continuous space along with having variable energy, so the network must learn correct energies and positions for each generated point. The geometry-aware approach only requires the model to learn the energies, as it is conditioned on the cell position and size, which are known beforehand even for generated samples. However, this means that the model must be trained on a wide range of potential geometries in order to generate new images, and it must be able to interpolate and generate images on geometries not present in the training data. There are several off-of-the-shelf models for learning on point-cloud data, but none currently for the cell approach, so these were developed based on existing models.

3 Data

The training dataset is generated using Geant4 with the calorimeter design created originally for the CaloGAN dataset [8]. The calorimeter has three layers: 'inner', 'middle', and 'outer' arranged along the z-axis with two orthogonal coordinates η and ϕ. Generating showers begins by shooting photons one at a time at the center of the calorimeter. As the photon interacts with the atoms of the detector material, it creates secondary particles that collide with atoms deeper in the detector. The result is a cascade of secondary products called a shower that develops sequentially through the calorimeter layers along the z-axis. 10k particle showers for photons with 65 GeV of energy were generated using Geant4.

For the point-cloud approach, the data is left in this form and the point-cloud models learn the data as collections of points in $\eta - \phi$ space per layer, with each point having some energy E. For the cell approach, these point-clouds are binned into cells of varying granularities in $\eta - \phi$ space that represent geometries present in the ATLAS detector. The different training geometries are represented as (n_η, n_ϕ), where n_η is the number of cells spanning the η-direction and n_ϕ is the number of cells spanning the ϕ-direction. The cell size is uniform for each geometry except for images of size (36,48) in the middle layer; there are 12 cells in negative η and 24 cells in positive η to represent the transition region of the calorimeter (See Fig. 1 for example). The subset of possible geometries per layer are shown in Table 1.

For the model to successfully simulate calorimeter images, the distributions of important physical quantities calculated from generated images must reproduce the distributions present in the training set. These distributions include the energy-weighted average mean and shower width of the hit in $\eta - \phi$ space per layer. The incident energy of the particle, must be conserved as well.

Layer	**Possible Geometries**
Inner	(48,4), (48,12), (48,24), (48,48), (192,4), (192,12), (192,24), (192,48)
Middle	(12,12), (48, 24), (48,48), (36,48)*
Outer	(24,24)

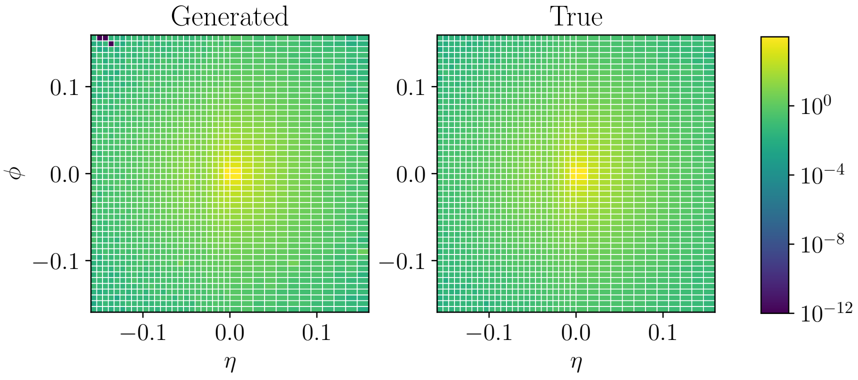


Figure 1: Mean training and generated images in the transition geometry for the S-ARM architecture

Table 1: Table of geometries that cell-approach models are trained on. (36,48) is the transition region geometry described in the Data section.

4 Model Architectures

Two approaches were studied: the point-cloud approach and the cell approach. Initial exploratory ideas will be discussed before moving on to more promising directions of simulation.

4.1 Geometry-Agnostic Simulation

Two algorithms were investigated to simulate calorimeter images in the point-cloud representation. These models feature two distinct generative model architectures: continuous normalizing flows (CNFs) and variational autoencoders (VAEs).

4.1.1 PointFlow

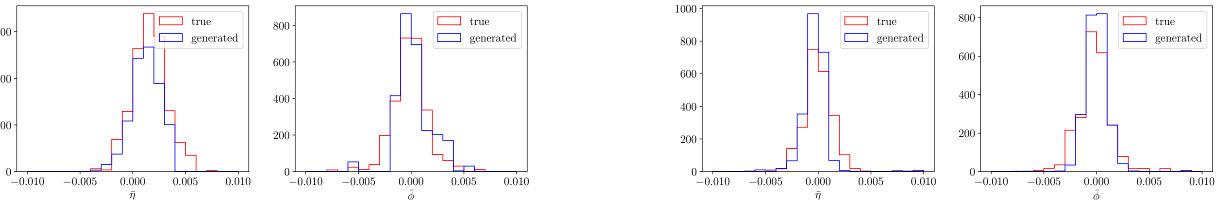
One approach that was explored was PointFlow [9], a CNF neural network architecture. PointFlow applies a series of invertible mappings to a simple random distribution such as a Gaussian, and the repeated mappings transform the simple distribution to an arbitrarily complicated distribution that models the training data distribution. During training, the model learns the parameters of the invertible mappings by maximizing the likelihood of the observed data. Once the model is trained, it can be used to generate new samples by inverting the flow of the learned transformations.

PointFlow was able to produce images that qualitatively looked quite similar to the training images, but was unable to reproduce the underlying physics distributions of the training set. In order to address this, a model using the VAE architecture was evaluated and compared to PointFlow.

4.1.2 SetVAE

SetVAE [10] is a neural network architecture that is designed to work with point-cloud data, similar to PointFlow. SetVAE uses a VAE architecture, which is made up of two neural networks: the encoder and the decoder. The encoder maps the training data to a lower-dimensional latent space, while the decoder maps the latent space back to the original image dimension.

The network's goal is to learn a mapping from the latent space to the training dataset. This is achieved by ensuring that the outputs of the decoder match the training images using the mean-squared error. The encoder learns values that are interpreted as the mean and standard deviation of the latent distribution, which is then sampled stochastically. During training, the latent space is constrained to be normally distributed, making it easier to sample from than an unconstrained latent space distribution.



(a) Energy-weighted average η and ϕ for (36,48) images.

(b) Energy-weighted average η and ϕ for 48x48 images.

Figure 2: Energy-weighted average η and ϕ values for (36,48) and (48,48) images.

The aim is for the latent space to encode the original training distribution so well that randomly sampling from the latent space and then transforming via the decoder results in simulated images that resemble the training images. To produce generated images, a vector is sampled from $\mathcal{N}(0, I)$, which is then decoded by the decoder.

Like PointFlow, SetVAE could generate calorimeter images that look qualitatively similar to the images they were trained on, but the model could not reproduce the underlying physics distributions. With two off-of-the-shelf SOTA models underperforming, the focus was shifted to geometry-aware models to improve performance.

4.2 Geometry-Aware Simulation

Geometry-aware models, unlike point-cloud models, receive the cell geometry of the calorimeter image to condition on during training. Generally, it is far too expensive for the network to consider the details of every cell individually, so certain methods must be used to make this geometry conditioning efficient. For example, autoregressive and transformer models take the features from each cell in sequence, using aggregated information learned from previous cells to predict the next cell in the sequence. Both autoregressive models and transformers will be discussed.

4.2.1 Sparse Autoregressive Model

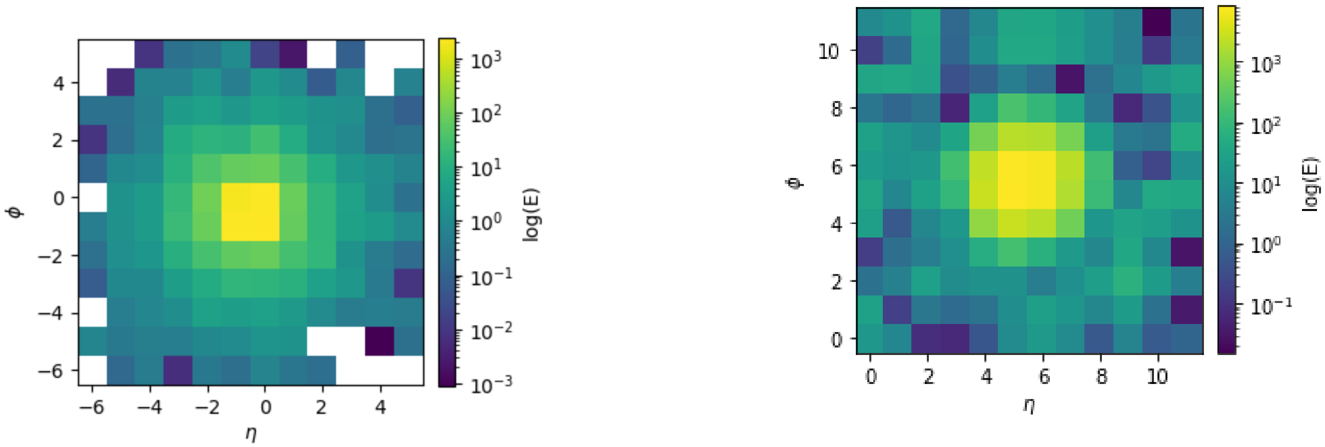
The Sparse Autoregressive Model (S-ARM) [11] framework is composed of three auto-regressive models (ARMs) trained separately for each layer. Autoregressive models learn to predict the value for the energy of a cell using not only information about the cell from other training samples, but also by aggregating information from previous cells. The model keeps a history of the previous cells' energies and uses this history to predict future cells in the sequence one at a time. The cell size $(\Delta\eta, \Delta\phi)$ are given as conditioning features , where $\Delta\eta$ is the length along the $\eta-$axis and $\Delta\phi$ is the length along the $\phi-$axis. These conditions are fixed and the model does not attempt to learn their distribution like it does with the energy, which allows the ARMs to learn the energy distribution as a function of these geometry features.

Generation begins after the central cell is given to the network by sampling from a known prior distribution. The trained ARM generates the energy deposits in subsequent cells using previous cell energies. The 2D cell positions in (η, ϕ) per layer are flattened, producing a 1D sequential ordering that determines how the network iterates through an image. The pattern used is a spiral path counterclockwise pattern starting from the center cell.

The performance is evaluated on the test dataset by comparing physics distributions from the training and generated datasets. The distributions to evaluate against are the same as in the point-cloud approach: energy-weighted mean and standard deviations in η and ϕ directions. In evaluating the performance of the model, these distributions should reproduce the distributions of the training images for the same calorimeter geometry. For example, a shift in the average η histogram is expected as an artifact of the transition region geometry in the middle of the calorimeter, so generated images must be able to represent that shift in its distributions.

Results First results using this S-ARM model have been achieved by me and collaborators [12]. Fig. 1 shows mean training and generated images using S-ARM, and Fig. 2 shows average η and ϕ histograms for two training geometries. The distribution for the (36,48) geometry correctly represents the shifts caused by the transition region in Fig. 2a.

In this work, my primary responsibilities are data preprocessing, physics validation, and providing physics feedback during model development, while a student from the Computer Science department at UCI develops the framework for the model architecture and ran training. This work was presented at NeurIPS 2022 in New Orleans as a poster presentation and is ongoing to demonstrate its effectiveness in interpolating geometries the model was not trained on.



(a) Sample training 12x12 image

(b) Sample generated 12x12 image generated by GNN VAE model

Figure 3: Training and generated 12x12 sample calorimeter images

Being an autoregressive model, an ordering must be selected for the network to iterate through the cells in an image. S-ARM has been shown [11] to be very sensitive to the choice of ordering, as the model necessarily learns different correlations between cells when flattened to 1D. Other SOTA models work with graph-structured data, in which all possible connections between cells can be evaluated and learned by the model. I am currently working on developing cell-conditioned models that are not autoregressive, so a specific ordering is not imposed on how the model predicts cells.

4.2.2 Graph Networks

Graph Neural Networks (GNNs) are a category of deep learning models that specifically cater to data represented as unordered networks of nodes. GNNs aim to learn features at both the node and global graph level by gathering information from adjacent nodes and connections in the graph. This gathering can take various forms including message-passing, which is essentially a weighted sum over neighboring nodes, or more complex architectures like attention (further discussed in the section on Set Transformers).

In attempting to generate images with a GNN, a VAE model was developed that operates on graph-structured data. Cell-binned calorimeter images are converted to graphs, where each node is connected to every other node in a given image. Each node has five cell features: $[\eta, \phi, \Delta\eta, \Delta\phi, E]$, where E is the only learned feature, and the other four are used to condition the network in a similar fashion as S-ARM.

Results An example generated and training image are shown in Fig. 3. While the GNN can generate images qualitatively similar to the training data, the vanilla VAE was ineffective in attempting to reproduce physics distributions like mean and shower width. However, training on graphs is a natural way to handle varying image sizes, as no additional preprocessing needs to be done to train on variable graph sizes (S-ARM must use zero-padding to ensure all images are the same number of cells). More advanced machinery must be used in place of message-passing, which only aggregates information locally. Thus, I have been focusing on transformer models, which take the training data in a similar graph-type format, but uses the more powerful attention mechanism to aggregate information in an image.

4.2.3 Set Transformers

Transformers are a type of deep learning model that process graph-structured data using attention mechanisms, which enable the network to weigh the importance of each sequence element when making predictions or output. This is achieved by computing a weighted sum of the sequence with weights determined by comparing each element to others, resulting in self-attention when computed on an image with itself. However, self-attention can be computationally expensive, scaling at $O(N^2)$, where N is the number of cells in an image. To address this, induced self-attention can be used, which computes the attention between an image and a lower-dimensional representation of the image [13]. Transformers are faster to train and optimize compared to GNNs and the addition of induced self-attention enhances their efficiency.

Generative adversarial networks (GANs) are another type of generative model architecture that has been extensively studied. GANs consist of two networks, a generator, and a discriminator, competing against each other. The generator produces images that the discriminator tries to classify as either real or fake, and the generator attempts to improve by creating better images to fool the discriminator.

Combining the effectiveness of self-attention with GANs, the Augmenting Generative Adversarial Particle Transformer (GAPT) is a new generative model being explored to generate calorimeter images for geometry-aware training [14]. GAPT is based on the set transformer and is capable of operating on point-clouds [13]. The current work is focusing on conditioning the GAPT model on spatial information $(\eta, \phi, \Delta\eta, \Delta\phi)$ similar to S-ARM instead of using point-clouds.

5 Conclusion

Several ML algorithms were investigated and compared in order to produce a general-purpose, multi-geometry model. Two data organization philosophies, point-cloud and cell data, were investigated and compared across several generative models, such as CNFs, VAEs, and GANs; the instances using point-cloud were shown to be ineffective at modelling the underlying physics of the calorimeter images, and so focus has shifted to the geometry-aware approach in the future.

Calorimeter simulation is a vital aspect of high-energy physics in the future; as more powerful detectors are constructed, computational cost will certainly increase. Thus, it will become more crucial to have fast simulation to study particle interactions in the detector. The current model of training several hundred generative networks to simulate one detector is very costly and cannot be migrated easily to new detectors. Current ML tools such as transformers and GANs show incredible promise in simulating highly complex images and being able to keep up with the rapid expanse of detector physics.

References

[1] "Deep generative models for fast shower simulation in ATLAS," CERN, Geneva, Tech. Rep., Jul. 2018, All figures including auxiliary figures are available at https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PUBNOTES/ATL-SOFT-PUB-2018-001. [Online]. Available: https://cds.cern.ch/record/2630433.

[2] Ratnikov, Fedor, "Generative adversarial networks for lhcb fast simulation," *EPJ Web Conf.*, vol. 245, p. 02 026, 2020. DOI: 10.1051/epjconf/202024502026. [Online]. Available: https://doi.org/10.1051/epjconf/202024502026.

[3] S. Agostinelli *et al.*, "GEANT4–a simulation toolkit," *Nucl. Instrum. Meth. A*, vol. 506, pp. 250–303, 2003. DOI: 10.1016/S0168-9002(03)01368-8.

[4] J. Allison *et al.*, "Geant4 developments and applications," *IEEE Trans. Nucl. Sci.*, vol. 53, p. 270, 2006. DOI: 10.1109/TNS.2006.869826.

[5] J. Allison *et al.*, "Recent developments in Geant4," *Nucl. Instrum. Meth. A*, vol. 835, pp. 186–225, 2016. DOI: 10.1016/j.nima.2016.06.125.

[6] S. Chatrchyan *et al.*, "The CMS Experiment at the CERN LHC," *JINST*, vol. 3, S08004, 2008. DOI: 10.1088/1748-0221/3/08/S08004.

[7] A. Collaboration, *Deep generative models for fast photon shower simulation in atlas*, 2022. arXiv: 2210.06204 [hep-ex].

[8] M. Paganini, L. de Oliveira, and B. Nachman, "CaloGAN: Simulating 3d high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks," *Physical Review D*, vol. 97, no. 1, Jan. 2018. DOI: 10.1103/PhysRevD.97.014021.

[9] G. Yang, X. Huang, Z. Hao, M.-Y. Liu, S. Belongie, and B. Hariharan, *Pointflow: 3d point cloud generation with continuous normalizing flows*, 2019. arXiv: 1906.12320 [cs.CV].

[10] J. Kim, J. Yoo, J. Lee, and S. Hong, *Setvae: Learning hierarchical composition for generative modeling of set-structured data*, 2021. arXiv: 2103.15619 [cs.LG].

[11] Y. Lu, J. Collado, D. Whiteson, and P. Baldi, "Sparse autoregressive models for scalable generation of sparse images in particle physics," *Physical Review D*, vol. 103, no. 3, p. 036 012, 2021.

[12] J. Liu, A. Ghosh, D. Smith, P. Baldi, and D. Whiteson, *Geometry-aware autoregressive models for calorimeter shower simulations*, 2022. arXiv: 2212.08233 [physics.ins-det].

[13] K. Stelzner, K. Kersting, and A. Kosiorek, “Generative adversarial set transformers,” *Workshop on Object-Oriented Learning at ICML 2020*, 2020.

[14] R. Kansal *et al.*, “Evaluating generative models in high energy physics,” *Physical Review D*, vol. 107, no. 7, Apr. 2023. DOI: `10.1103/physrevd.107.076017`.