Geometry-aware Autoregressive Models for Calorimeter Shower Simulations

Junze Liu[*1], **Aishik Ghosh**[*1,2], **Dylan Smith**[*1], **Pierre Baldi**[1], **Daniel Whiteson**[1]
[1]University of California, Irvine
[2]Lawrence Berkeley National Laboratory, Berkeley
`{junzel1, aishikg, dylanrs, pfbaldi, daniel}@uci.edu`

Abstract

Calorimeter shower simulations are often the bottleneck in simulation time for particle physics detectors. A lot of effort is currently spent on optimising generative architectures for specific detector geometries, which generalise poorly. We develop a geometry-aware autoregressive model on a range of calorimeter geometries such that the model learns to adapt its energy deposition depending on the size and position of the cells. This is a key proof-of-concept step towards building a model that can generalize to new unseen calorimeter geometries with little to no additional training. Such a model can replace the hundreds of generative models used for calorimeter simulation a Large Hadron Collider experiment. For the study of future detectors, such a model will dramatically reduce the large upfront investment usually needed tp generate simulations.

1 Introduction

Deep generative models have become essential in solving the fast simulation needs in particle physics [1–5]. Of particular concern are calorimeter simulations, which take up a large component of the computing budget of physics experiments [5]. While the same base architecture is found to perform reasonably on a multitude of natural image datasets, this has not been the case for calorimeter data, which are three dimensional images where the resolution of the image is determined by the size of the calorimeter cells (analogous to pixels). A significant amount of research time and computing resources are spent to develop generative architectures from scratch for different calorimeter geometries [6–9] (the geometrical arrangement of the cells, which may have different shapes and sizes). These studies have usually ignored challenging transition regions of the calorimeter where the geometry abruptly changes (for example the cells become wider in one dimension at a transition point), or dealt with them by training hundreds of networks to simulate different geometries within the same calorimeter [5]. Developing an architecture that can generalise to multiple calorimeters would significantly cut down on the R&D time currently invested to develop geometry specific models from scratch and validate them. Such a generalisable architecture would make future detector studies cheaper by dramatically reducing the upfront computational investment required to simulate a new detector design, since they can currently be done only using the slow (albeit accurate) first principles based simulation software (Geant [10–12]). When the difference in geometry is only in the segmentation of the detector, the model could zero-shot (without any additional training) adapt to the new geometry, and for calorimeter designs with a different composition of interaction material, it could act as a foundational (pre-trained) model that is tuned to the new calorimeter with far fewer training samples compared to what would be required to develop a model from scratch.

To solve this task, we investigate a geometry-aware autoregressive generative model which learns to adapt its energy deposition from cell to cell depending on its size and position, as well as the

[*]These authors contributed equally to this work

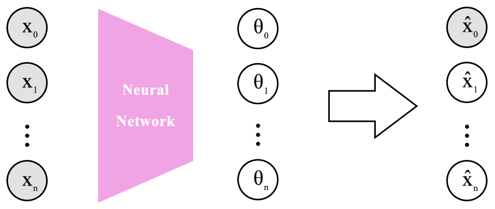
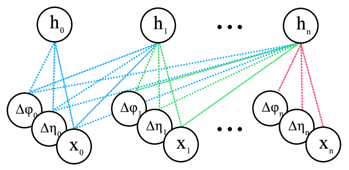
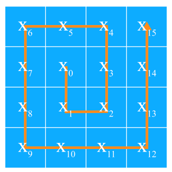


Figure 1: The architecture of one out of three Discrete Mixture Model Autoregressive Model (D+D ARM) we implement. The true starting value x_0 is given when generating the samples. θ_i is learned during the training and used to sample the energy deposit of each cell in calorimeter simulation.



(a) D+D ARM with cell sizes as extra inputs



(b) The spiral path

Figure 2: (a) The cell's size in η and ϕ is added as the extra input to provide the geometry information. All three inputs of each cell (energy deposit, size in η, and size in ϕ) are aggregated to hidden nodes according to the input ordering; (b) the 2D energy deposit matrix is flattened using a spiral path.

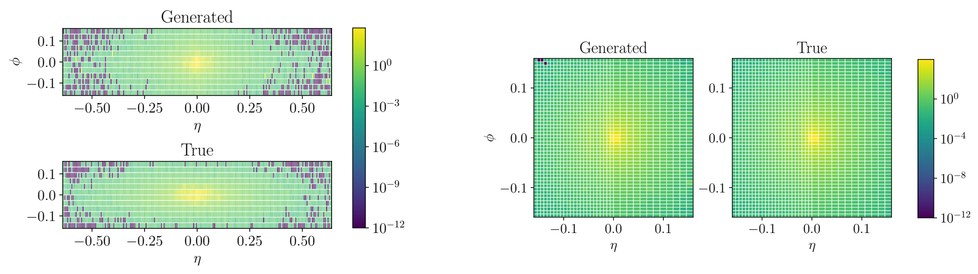
energy distribution in its neighbouring cells. Deep autoregressive models (ARMs) have demonstrated impressive performance for modeling the sparsity in images, thus are well suited for this task [13–15]. The model is trained on a range of calorimeter geometries, including the particularly challenging transition regions. We show that our model can adapt to these differences and reproduce the impact the geometry of a calorimeter has on important physics variables. This is a key step towards building a generalisable architecture that can adapt to new unseen calorimeter designs with little to no additional training.

2 Dataset

The training dataset is generated using Geant4 [10–12] with the calorimeter design created originally for CaloGAN dataset [16], but we modified the segmentation of the detector for our study. The calorimeter has three layers, 'inner', 'middle', and 'outer', denoted by $z = \{1, 2, 3\}$ and two orthogonal coordinates, η and ϕ. It records a three dimensional image but with no time information. To generate particle showers, we shoot single photons at the centre ($\eta = 0$, $\phi = 0$) of the calorimeter. The photon generates a shower of particles as it interacts with the material, and the shower develops sequentially through the calorimeter layers. 10,000 particle showers for photons with 65 GeV of energy were generated at (pseudo-)point-cloud level using Geant4 and then binned into cells of varying granularities in η-ϕ space. The innermost layer is generated by starting from the default dimension of (12, 12) and uniformly sampling multiplicative factors along each dimension: 4 and 16 in η and $^1/_3$, 1, 2, and 4 in ϕ. The middle calorimeter layer was generated by uniformly sampling layer sizes of (12, 12), (48, 24), (48,48), and (36, 48), wherein the final geometry has 12 cells on the left half of the image and 24 cells on the right half in η (this represents the transition region of a full calorimeter) ; all other geometries have uniform cell sizes across each image. Finally, the outermost layer are all of the same shape, (24, 24), as this is the sparsest layer. These geometries were chosen to resemble the range of calorimeter segmentation that exist in the ATLAS experiment [17] electromagnetic calorimeter.

3 Methods

We design a framework based on the autoregressive model (ARM) as an efficient data generator for simulated multi-layer calorimeter data. The framework is composed of three Discrete Mixture Model (D+D) ARMs trained separately for each layer (Figure 1). Each D+D ARM is built based on Masked Autoencoder for Distribution Estimation (MADE) [18]. While traditional auto-regressive models generate each cell in the output sequentially, MADE generates all desired parameters with a single pass through the regular autoencoder. This feature allows us to exploit the parallel computation



(a) Inner layer - (192, 12)

(b) Middle layer - (36, 48)

Figure 3: The average generated data is similar to the average original data. (a) The generated and true average inner layer of (192, 12), and (b) the middle layer of (36, 48).

on GPUs for faster training. The D+D ARM for the generation of the inner layer consists of one masked fully connected layer and one 1D convolutional layer, and the D+D ARMs for the generation of the middle layer and the outer layer consist of five masked fully connected layers and one 1D convolutional layer. We use GELU [19] as the activation function. Then, we have a following softmax layer to classify the value into N categories, where N is between 0 and the max cell value in the sample. Each generated energy deposit value is modeled as a categorical distribution.

In addition to the original energy deposits, the cell sizes in η and ϕ dimensions are also given as inputs to the D+D ARM, enabling the model to adaptively generate the energy distribution for different geometries (Figure 2a). During training, the inputs of each ARM includes a matrix of the original simulated calorimeter data with energy deposits in cells and two matrices of each cell's size in η and ϕ. We start the generation from the central cell because the shower typically deposits the most energy near the center, and this acts as a good starting point for the ARMs. In additional, we give the cell size ($\Delta\eta$, $\Delta\phi$) as additional inputs, where $\Delta\eta$ is the length along the η-axis and $\Delta\phi$ is the length along the ϕ-axis. This allows the ARMs to learn the energy distribution as a function of these geometry features. At test time, we give the networks a value for the central cell as a starting point, which is sampled from a known prior distribution, and two matrices which describe the size of each cell in η and ϕ. The trained D+D ARM then generates the energy deposits in the rest cells by sampling from the learnt multinomial probability distribution.

The calorimeter data is carefully prepared to fit in our models. The 2D matrix is flattened using a spiral path counterclockwise to fit in our D+D ARM, so the neighboring cells share similar energy deposit values (Figure 2b). We discretize the energy deposit value by rounding it to the nearest integer. The energy depositions vary by several orders of magnitude from one cell to another, and preprocessing them by a power transformation, $\hat{x} = x^{1/p}$, was found to improve training. The optimal value for the hyper-parameter p was found to be $p = 2$. We use a splitting ratio 0.8/0.1/0.1 to split the dataset to a training set, a validation set, and a test set. The models were trained in PyTorch [20] up to 100 epochs using the Adam optimizer [21] and a negative log likelihood as the loss function on two NVIDIA RTX 3090 GPUs. The performance is evaluated on the test set by computing physics observables from the raw Geant4 and ARM generated images and comparing their distributions. The physics variables include energy weighted mean and standard deviations in η and ϕ directions, and the correlation between lateral (η, ϕ) distributions between the calorimeter layers in the z direction. We also visually inspect individual samples, and the average images for each geometry.

4 Results

Our D+D ARMs show promising performance in the ability to adapt data generation to various calorimeter geometries. We study the performance in two aspects: qualitative and quantitative analyses. For qualitative analysis, we visualize the average sample obtained by averaging all generated samples. The average sample of each layer generated by our D+D ARMs is compared with the original average sample. Figure 3 shows two example geometries, (192, 12) and (36, 48), from the inner layer and middle layer respectively. The samples of (36, 48) from the middle layer are particularly challenging because they have nonuniform cell sizes along the η direction. The results show that our geometry-aware D+D ARMs are able to learn to generate data with more energy deposit in larger cells while less in smaller cells with in the same sample. For quantitative analysis, we show three physics variables: energy weighted means, shower widths, and correlations between layers. The

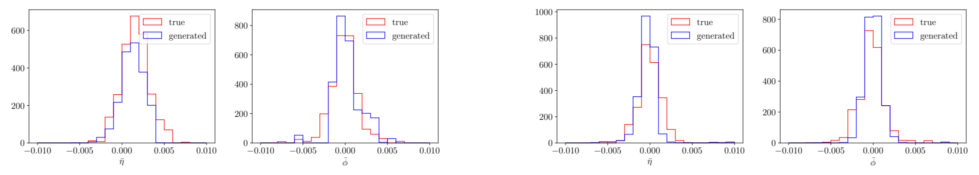

(a) Middle layer - (36, 48) (b) Middle layer - (48, 48)

Figure 4: Distribution of the energy weighted means of samples of (a) shape (36, 48) and (b) shape (48, 48) in the middle layer. The generated data and the original data share a similar distribution.

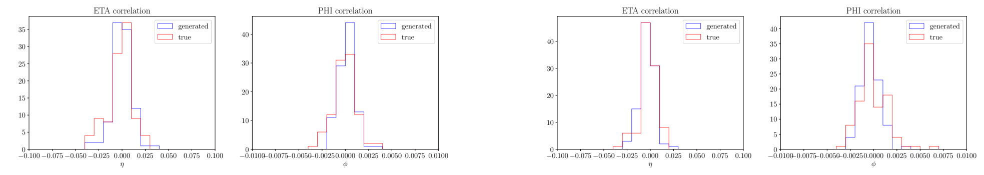

(a) Inner (48, 12) & middle (12, 12) layer (b) Inner (48, 4) & middle (36, 48) layer

Figure 5: Distribution of the distance of energy weighted means between (a) samples of (48, 12) in the inner layer and (12, 12) in the middle layer, and (b) samples of shape (48, 4) in the inner layer and (36, 48) in the middle layer. The generated data and the original data share a similar distribution.

energy weighted mean is summed over all cells in an image as Eq. 1 in η (and likewise in ϕ):

$$\bar{\eta} = \frac{\sum_i \eta_i E_i}{\sum_i E_i} \tag{1}$$

where E_i is the energy deposit of the i^{th} cell. The shower width is as Eq. 2 in η (and likewise in ϕ):

$$\sigma_\eta = \sqrt{\frac{\sum_i E_i(\eta_i - \bar{\eta})^2}{\frac{(M-1)}{M}\sum_i E_i}} \tag{2}$$

where M is the number of cells with non-zero energy. The distributions of these two parameters calculated from all generated and original samples of each geometry are visualized side by side to study the performance of our D+D ARMs in statistics (Figure 4). The slight off-set to the right of the peak in the η distribution is a feature of transition regions of calorimeters and therefore expected for the geometry (36, 48) of the middle layer. We choose samples of the shape (36, 48) and (48, 48) from the middle layer to show the robustness of our D+D ARMs in the generation of different geometries regardless of the varying cell size. At last, we show the correlation of samples between two layers with different geometries by calculating the distance between their energy weighted means (Figure 5). The distributions of correlations are visualized between samples of (48, 12) from the inner layer and (12, 12) from the middle layer, as well as samples of (48, 4) from the inner layer and (36, 48) from the middle layer. The generated samples share similar distribution as the original samples, indicating that our D+D ARMs learns the correlation between layers.

5 Conclusion and Future Directions

We study the first attempt of training a geometry-aware generative model to simulate a range of calorimeter geometries and demonstrate that the model is able to adapt its energy in response to different calorimeter geometries. This is the first step towards building a general purpose generative architecture that can handle new calorimeter segmentations and become a pre-trained base that can be quickly tuned to new calorimeter designs. Our proposed geometry-aware autoregressive model shows promising performance for this challenging task, with room for further improvement. The model learns a generalised way to simulate different calorimeter geometries by learning to use information about each individual cell; the cell size and position. Our experiments prove that deep autoregressive models are able to adaptively learn to generate the energy distribution when the cell size varies even within individual calorimeter layers. The method we implemented is still preliminary but shows us the direction for the next phase. Further work will focus on improving the fidelity of the generated images. We may enforce sparsity through Sparse Autoregressive Models (SARMS) [15] that use an extra parameter to model the sparsity in the data and generate continuous output, instead of discrete outputs we currently have. While we currently sample the energy of the central cell from a histogram, in the future this may be generated with a neural network. By making the generation of each layer fully conditional on other layers, we could further improve the correlation between the generated layers. Finally, we will study the zero-shot adaptation of the model for more and more challenging geometries.

Acknowledgement

JZ and PB are in part supported by grant ARO 76649-CS to PB. AG, DS, and DW are supported by the U.S. Department of Energy, Office of Science under grant DE-SC0009920. DS is additionally supported by a HEPCAT fellowship from the U.S. Department of Energy.

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] A. Butter and T. Plehn, "Generative Networks for LHC events," Aug. 2020. arXiv: `2008.08558 [hep-ph]`.

[3] K. Deja, T. Trzcinski, and Ł. Graczykowski, "Generative models for fast cluster simulations in the TPC for the ALICE experiment," *EPJ Web Conf.*, vol. 214, A. Forti, L. Betev, M. Litmaath, O. Smirnova, and P. Hristov, Eds., p. 06 003, 2019. DOI: `10.1051/epjconf/201921406003`.

[4] V. Chekalina, E. Orlova, F. Ratnikov, D. Ulyanov, A. Ustyuzhanin, and E. Zakharov, "Generative Models for Fast Calorimeter Simulation: the LHCb case," *EPJ Web Conf.*, vol. 214, A. Forti, L. Betev, M. Litmaath, O. Smirnova, and P. Hristov, Eds., p. 02 034, 2019. DOI: `10.1051/epjconf/201921402034`. arXiv: `1812.01319 [physics.data-an]`.

[5] G. Aad *et al.*, "AtlFast3: the next generation of fast simulation in ATLAS," *Comput. Softw. Big Sci.*, vol. 6, p. 7, 2022. DOI: `10.1007/s41781-021-00079-7`. arXiv: `2109.02551 [hep-ex]`.

[6] E. Buhmann *et al.*, "Getting High: High Fidelity Simulation of High Granularity Calorimeters with High Speed," *Comput. Softw. Big Sci.*, vol. 5, no. 1, p. 13, 2021. DOI: `10.1007/s41781-021-00056-0`. arXiv: `2005.05334 [physics.ins-det]`.

[7] C. Krause and D. Shih, "CaloFlow: Fast and Accurate Generation of Calorimeter Showers with Normalizing Flows," Jun. 2021. arXiv: `2106.05285 [physics.ins-det]`.

[8] V. Mikuni and B. Nachman, "Score-based Generative Models for Calorimeter Shower Simulation," Jun. 2022. arXiv: `2206.11898 [hep-ph]`.

[9] G. R. Khattak, S. Vallecorsa, F. Carminati, and G. M. Khan, "Fast simulation of a high granularity calorimeter by generative adversarial networks," *Eur. Phys. J. C*, vol. 82, no. 4, p. 386, 2022. DOI: `10.1140/epjc/s10052-022-10258-4`. arXiv: `2109.07388 [physics.ins-det]`.

[10] 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`.

[11] 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`.

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

[13] A. Van Den Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," in *International conference on machine learning*, PMLR, 2016, pp. 1747–1756.

[14] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma, "Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications," *arXiv preprint arXiv:1701.05517*, 2017.

[15] 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.

[16] 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`.

[17] G. Aad *et al.*, "The ATLAS Experiment at the CERN Large Hadron Collider," *JINST*, vol. 3, S08003, 2008. DOI: `10.1088/1748-0221/3/08/S08003`.

[18] M. Germain, K. Gregor, I. Murray, and H. Larochelle, "Made: Masked autoencoder for distribution estimation," in *International conference on machine learning*, PMLR, 2015, pp. 881–889.

[19] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv preprint arXiv:1606.08415*, 2016.

[20] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 5
   (c) Did you discuss any potential negative societal impacts of your work? [No] Our work mainly targets improving physics simulation efficiency.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] Our project is still on-going, and the code is frequently being updated.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 3
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 3
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]