Federated Learning Strategies of Generative Adversarial Networks for High Energy Physics Calorimeter Simulation

Mohamed A. Hemdan [1] José Cabrero-Holgueras [2,3] Sofia Vallecorsa [2]

[1]The American University in Cairo, Cairo, Egypt, mashrafhemdan@aucegypt.edu
[2]CERN, Geneva, Switzerland, Sofia.Vallecorsa@cern.ch
[3]Universidad Carlos III de Madrid, Madrid, Spain, jose@cabreroholgueras.com

Abstract. Calorimeter simulation using Monte-Carlo is a computationally demanding task both in terms of computing and storage and relies on the use of distributed computing clusters. Generative Adversarial Networks (GANs) provide a more efficient alternative by capturing the complex underlying data distribution and generating synthetic data rapidly. In most cases, existing prototypes simulate the traversal of individual particles through small detector volumes, and thus, the training datasets are of a manageable size and confined to a single cluster. The next natural step in this line of research is to train larger generative models that incorporate more complex physics processes and detectors (including various sub-detector combinations). This would increase the training dataset size, and most likely require merging independently produced samples. Given that Monte Carlo data is generated and distributed across multiple cluster nodes (such as those within the WLCG tiered infrastructure), the communication overhead involved in transferring data between servers for centralized training would inevitably rise. To address this issue, we propose federated learning as a decentralized training solution, wherein a group of collaborators trains a model by sharing training updates with an aggregator. Our approach combines federated learning with GAN-based calorimeter simulation by locally training individual GAN models on different portions of the Monte Carlo data and aggregating the resulting models using the *FedAvg* algorithm. Our experiments demonstrate the effectiveness of our approach for GAN-based calorimeter simulation, achieving comparable simulation accuracy. Our approach could contribute to developing more accurate and scalable simulation methods for particle physics experiments.

1. Introduction
Generative Adversarial Networks (GANs) have shown great potential in High Energy Physics (HEP) due to their ability to capture complex data distributions. They have been used extensively in the simulations of energy showers in particle detectors [1, 2, 3]. They were also used in other HEP applications, including cosmic ray interaction simulation and adversarial optimization [4]. However, GANs are characterized by being data-hungry as they rely heavily on large amounts of diverse and high-quality data to produce high-fidelity samples. In particular in cases in which the synthetic dataset should be used for high-precision analyses. Losing such a requirement creates uncertainty in the quality of the generated samples, especially if the underlying distribution is very complex, which is a typical example in HEP simulations [5]. On the other hand, increasing the training set size is not always feasible. With the advancement of increasingly intricate models,

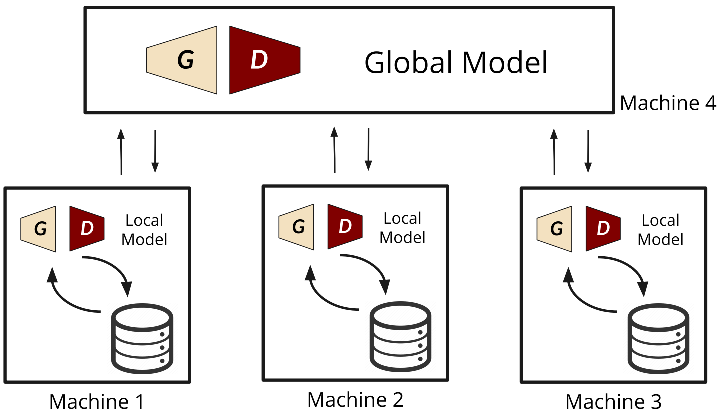

Figure 1: FL Training: Clients share local model parameters with the centralized machine. Parameters are aggregated to update the global model, which is then sent back to clients to update their local models.

communication constraints may arise among cluster nodes that possess the full list of datasets necessary for training. For instance, on the LHC Computing Grid (WLCG), data is generally dispersed across numerous tiers within the infrastructure.

Federated learning (FL) is a technique for training machine learning (ML) models on distributed data across many clients [6]. A centralized node sends copies of the global model parameters to each client, which trains the model copy on local data. These local models are then sent back to the centralized node for aggregation (e.g., averaging model parameters). Through many rounds, the global model is gradually updated to learn and perform well on the entire distributed data. FL enables models to train on larger data sets and reduces biases associated with locally trained models. Different approaches have been proposed for the federated training of GANs like FedAvg [6], and MD-GAN [7]. In our experiments, we consider FedAvg, in which both the generator and the discriminator are located on the client and the server, since it proves to work well with independent and identically distributed (iid) data [6].

In this paper, we propose a federated learning approach for GAN-based calorimeter simulation in particle physics where we have adapted the 3DGAN [2] model to federated learning and analyzed the changes with respect to centralized training. We experimented with different hyper-parameters (e.g., number of clients, local epochs, rounds) and studied their effect on model performance. Our results demonstrate that the federated learning approach reaches an equivalent performance to the centralized scenario. The paper is structured as follows. Section 2 provides background on federated learning, Section 3 presents our main findings, and Section 4 provides a discussion of the results and conclusion.

2. Background

Federated learning is based on training an ML model on data from different clients, as shown in Fig. 1. We used the FedAvg algorithm for model aggregation[6]. The FedAvg algorithm supposes there exist K clients, n the total number of data points on all clients, P_j the data existing on client j, and n_j the number of samples at client j. We want to minimize the overall loss f, which decomposes into the following.

$$\min_{w\in R^d} f(w) \qquad s.t. \qquad f(w) = \sum_{j=1}^{K} \frac{n_j}{n} F_j(w), \qquad F_j(w) = \frac{1}{n_j} \sum_{i\in P_j} f_i(w)$$

Where $f(w)$ is the overall loss function for model parameters w, $F_j(w)$ is the total loss at client j and $f_i(w)$ is the loss of sample i. The 3DGAN model uses a weighted sum of individual losses[**?**

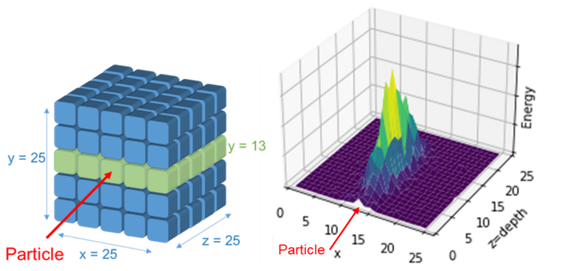

Figure 2: 3D Representation and Particle Shower development at $y = 13$ [2]

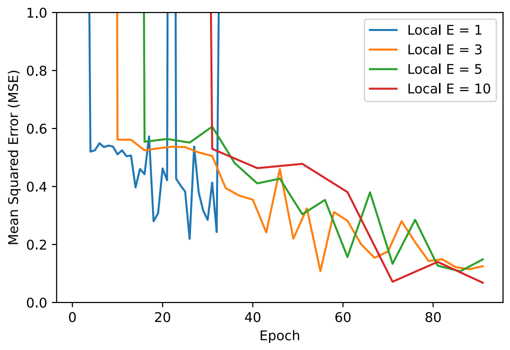

Figure 3: MSE Validation Loss per epoch for different number of local epochs

]. To simplify our evaluation, we used the mean squared error (MSE) between the generated and actual energy depositions as the loss function $f(w)$. Thus, we try to obtain similar data distributions. In each round, the generator and the discriminator of each client are trained concurrently in a Min-Max game.

3. Evaluation

To carry out the experiments, we used data from 20000 energy showers generated using Monte Carlo. An example of the energy shower, its 3D representation, and its energy deposition are shown in Fig. 2. To simulate federated learning, we distributed the data equally across a number of different clients. We adapted the 3DGAN model to federated learning and experimented with different hyperparameters, namely the number of local epochs and the number of clients, due to their significant effect on convergence rate and training time. The experiments run on a server node with an Intel(R) Xeon(R) Silver 4216 CPU @ 2.10GHz with 64 cores, 196 GB of RAM, and 4 Nvidia Tesla V100S GPUs.

To deploy our system, a Python-based tool has been developed that enables the training of federated learning (FL) models on remote machines. This tool utilizes Ansible to distribute the client and server programs to the designated machines. The programs operate within Docker containers and communicate with each other through the network to exchange model updates and parameters. A monitoring container receives performance metrics from the clients and servers and provides them to the user upon request. The tool comprises two distinct Python packages: one leveraging the Flower library[8] for managing federated training and parameter updates for both the client and server programs [9], and another dedicated to the building and deployment of the containers [10].

3.1. Local Epochs

Local epochs (E) refer to the number of iterations in which a client trains its local model on its own data before transmitting the updated model to the central server. To examine the impact of varying E on the performance of the global model, we conducted a series of experiments involving three clients, each with a different number of E, as illustrated in Fig. 3. To ensure that the participating devices' computational resources were not overwhelmed, we limited E to 10 or fewer. Our findings indicate that all plots exhibit a decreasing trend in the loss function over the epochs. However, we observed that increasing E results in a more consistent learning curve, which can be attributed to clients having more time to converge to a more accurate solution that better reflects their data distribution.

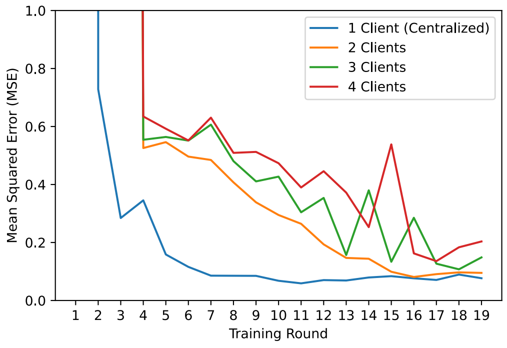

Figure 4: MSE validation loss per epoch evaluated for different numbers of clients

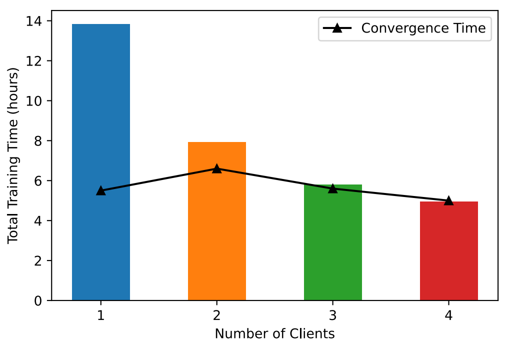

Figure 5: Total Training Time and Convergence Time for different numbers of clients

3.2. Number of Clients

The efficacy of the central model is contingent upon the distribution of data that is partitioned among multiple clients. To investigate the impact of data partitioning on performance, we conducted several experiments with varying numbers of clients C. The mean squared error (MSE) loss was monitored for different numbers of clients to assess the effect of data distribution on performance. Results depicted in Fig. 4 reveal that experiments involving fewer clients exhibit faster convergence in training rounds compared to those involving a larger number of clients.

3.3. Training Time

Increasing the number of clients results in data parallelization, reducing the overall training time. To assess this phenomenon, we conducted experiments to measure the training time and time-to-convergence for each number of clients. Our findings, as depicted in Fig.5, indicate that the overall training time decreases as the number of clients increases. For example, training time for two clients is significantly lower than the centralized approach by 44%. However, we observed a slight increase in convergence time when moving from one client to two clients. Nevertheless, the overall decrease in training time offsets this effect, resulting in reduced convergence time. Remarkably, at four clients, convergence time decreased by 10% compared to the centralized approach.

4. Discussion and Conclusions

Federated and centralized learning are different. In centralized learning, only one client/machine is used for training. In contrast, federated learning uses many clients for training a global model, which is constructed by aggregating the local models from each client. It is highly efficient as it only exchanges model parameters compared to centralized learning, which exchanges the actual data. These local models are then updated every E number of epochs, known as a round. During each round, each local model converges with the local data, producing biased models. With many clients, each having a different model, aggregation can hardly produce a generalizable model, slowing down convergence. For example, as the number of clients increases (e.g., 2-4 clients, shown in Fig. 4), the convergence rate decreases. However, such delay is catered for by the parallelism of local epochs, which reduces the overall training time, as shown in Fig. 5.

The impact of local epochs on model convergence is a crucial consideration in federated learning. Insufficient local epochs can prevent the model from effectively capturing the local data features. Consequently, the global model aggregated from the local models may not accurately represent the entire data and perform poorly on test data. For instance, a local epoch of E=1 can cause model divergence, as demonstrated in Fig. 3. In contrast, increasing the number of local epochs enables the model to effectively capture the underlying data distribution, leading to improved performance on new data. However, this comes at the cost of increased computational resources, which may be constrained in participating clients. Therefore, a tradeoff is necessary to determine the optimal number of local epochs. Our findings indicate that stable results can be obtained with local epochs ranging from 3 to 10. Furthermore, our model ensures privacy by securely storing local data at each client without any data exchange between the clients or the server.

Overall, federated learning can provide an effective strategy to increase the performance of current HEP machine-learning applications. It enables GANs to train on much larger datasets, reducing uncertainties in the generated data. In this work, we explored the adaptation of federated learning to GAN-based calorimeter simulation. We experimented with different hyperparameters and different numbers of clients. Our experiments showed more stable learning curves for a high number of local epochs and smaller training times for a large number of clients. Furthermore, our federated training approach has achieved equivalent performance to the centralized approach.

From a broader point of view, we consider this work an initial look into the more general strategy needed for training deep neural networks across a distributed infrastructure such as the LHC Computing Grid. Data (experimental or Monte Carlo-generated data) is stored across multiple nodes on the Grid, and the development of advanced Machine/Deep Learning architectures (including generative models) might require in the near future, the combination of different datasets scattered across multiple sites. Federated Learning could provide the tools needed to make this additional step toward the design of efficient strategies for distributed deep learning. However, studies are needed in order to understand FL integration into the WLCG infrastructure and, more in general, the High Energy Physics computing model. We believe our work is a first step in this direction.

References

[1] Hosein Hashemi and Claudius Krause. Deep Generative Models for Detector Signature Simulation: An Analytical Taxonomy. 12 2023.

[2] Florian Rehm, Sofia Vallecorsa, Kerstin Borras, and Dirk Krücker. Validation of deep convolutional generative adversarial networks for high energy physics calorimeter simulations, 2021.

[3] Andreas Adelmann et al. New directions for surrogate models and differentiable programming for High Energy Physics detector simulation. In *Snowmass 2021*, 3 2022.

[4] Gilles Louppe, Joeri Hermans, and Kyle Cranmer. Adversarial variational optimization of non-differentiable simulators. 2017.

[5] Konstantin Matchev, Alexander Roman, and Prasanth Shyamsundar. Uncertainties associated with GAN-generated datasets in high energy physics. *SciPost Physics*, 12(3), mar 2022.

[6] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. 2016.

[7] Corentin Hardy, Erwan Le Merrer, and Bruno Sericola. MD-GAN: Multi-discriminator generative adversarial networks for distributed datasets. In *2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, may 2019.

[8] Daniel J Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Javier Fernandez-Marques, Yan Gao, Lorenzo Sani, Hei Li Kwing, Titouan Parcollet, Pedro PB de Gusmão, and Nicholas D Lane. Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*, 2020.

[9] Mohamed Hemdan. fed-pack. `https://github.com/mashrafhemdan1/fed_pack`, 9 2022.

[10] Mohamed Hemdan. fed-deploy. `https://github.com/mashrafhemdan1/fed_deploy`, 9 2022.

[11] Eckhard Elsen. A roadmap for HEP software and computing R&D for the 2020s. *Computing and Software for Big Science*, 3(1), November 2019.

[12] Luke de Oliveira, Michela Paganini, and Benjamin Nachman. Learning particle physics by example: Location-aware generative adversarial networks for physics synthesis. *Computing and Software for Big Science*, 1(1), sep 2017.