A Fast Particles Collision Event Generation Model Using a Conditional Variational Autoencoder conjugated with Multi Layer Perceptron Regression Network

Reza Shokrzad
s1056369

(Dated: December 3, 2022)

A high-energy collision in particle physics is called an event that is simulated in this study. We generate post-collision data, including energy-momentum 4-vector, and mass of new particles, with the Monte Carlo approach based on system energy conservation and relativistic dispersion relation in particle physics. Then, we train an MLP regression network to predict masses of particles given a corresponding 4-vector. The network could achieve RMSE 6.20 and 6.28 for the event particles that are promising for the masses with means about 100. Additionally, we propose a conditional Variational Auto Encoder to facilitate the generation process whose input is the masses of the regressor. The cVAE model outputs the synthetic 4-vector and the result indicates that the method can be help in generating data fast.

I. Problem Description

In particle physics, a fundamental interaction that occurs between subatomic particles in a short moment is called an event. An event leads to scattered, destroyed, or converted particles into several other particles. Creating events is necessary for physicians to answer such questions what are the particles made of, or what creates the interactions of matter? Ion implantation in the semiconductor industry and modification of surface properties of many materials are examples of industrial applications of such events [1]. Due to uncertainty in quantum physics, physicists use hypothetical predictions to clarify the experimental data following a collision. Any deviations from the theory can be a signal of New Physics that shows the importance of studying the field. To charge particles such as protons or electrons, at high speeds, close to the speed of light, an accelerator is needed. The Large Hadron Collider (LHC) is the most notable modern particle accelerator, which boosts the speed of particles and holds mentioned events. A collision event generates new scattered debris, which is tractable by giant detectors. The detecting subsystems are allocated in six different layers around a collision with the purpose of recording paths, momentum, and energy of new particles. A set including operation, maintenance, used material, power consumption, and human resources makes the process challenging and costly. Thus, simulating such a process can bridge the gap between mathematically challenging Quantum Field Theory and measurements at collider experiments [2].

The Monte Carlo (MC) sampling technique is traditionally the most popular method in quantum that has been applied to develop several accurate simulation tools for instance, FLUKA to track arbitrary heavy-ion species [3] , and GEANT4 the state-of-the-art toolkit the passage of particles through matter [4]. As a promising approach, MC is also widely used for event simulation. Additionally, Machine Learning (ML) generative models have been applied to reduce the cost and time of simulation by traditional methods for the last decade. Martínez J. et. al, propose a ML model based on Generative Adversarial Networks (GANs) [5] to simulate a full-event at the LHC and their application to pileup description [6]. Besides, Touranakou M. et. al, succeed to train a Variational Autoencoders (VAEs) [7] network that performs the state-of-the-art precision on the jet four-momentum [8].

This study simulates an electron-positron reaction that produces two new particles, as seen in Figure 1. We consider MC sampling as the first step to studying such a mentioned event. We train a Multi Layer Perceptron (MLP) network to predict invariant masses given by energy and momentum. Since Monte Carlo simulations are computationally expensive due to complicated detector geometries and diverse physical processes [9], we propose a conditional VAE to make this process faster. Thus, the model is built and trained given the regression output masses, to make a synthetic 4-vector of a particle (E, p_x, p_y, p_z) given the masses from the previous step (i.e. the MLP). The output reveals that the proposed method can generate data fast. The rest of the paper is organized as follows. In section 2, background knowledge about the MC approach, MLP regression network, and cVAE is presented. The detail of proposed methods are found in section 3. Section 4 presents the experiments' results to evaluate the proposed methods. The paper ends with the conclusion presented in section 5.

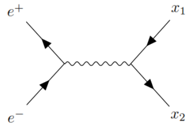

FIG. 1: An electron-positron collision event. Only two new generated particle is a simplification.

II. Preliminary knowledge

This section introduces essential definitions and knowledge related to this study. Simulations of nature and, in particular, processes in particle physics require expensive computations and can sometimes take much longer than scientists can afford. This reason magnifies the importance of computer-based simulations. A practical tool of particle physics, Monte Carlo has been a method that uses probability and randomness since the early 1970s [10]. The approach is to randomly choose a draw from a feasible range by considering criteria to accept or reject the sample with a certain probability. ML-based generative models, along with advances in ML, have become a severe facilitator to MC in the task of simulation reactions in high-energy events recently. While MC tools simulate physics events from first principles, ML models are data-driven learning from sample events. So far, GANs [5], VAEs [7], and Normalizing Flows (NFs) [11] are the most popular generative ML models have been applied for particle physics [10]. Since among the proposed methods, we use VAEs, especially cVAEs, in this work, a profound explanation of the VAE mechanism is provided.

VAE, a generative neural network, comprises an encoder block Ψ and a decoder block Φ, two simple or even deep neural networks. The aim of VAE is to learn the input distribution and generate samples after the training. Encoder Ψ projects the events into latent variables z, a bottleneck called latent space. Then, Φ reconstructs the events from z where z is forced to follow a standard normal (Gaussian) distribution. The loss function for the VAE is then derived from variational inference [12] by minimizing the Kullback–Leibler divergence (KLD) between the posterior $p(z|x)$ and the encoded prior distribution $q(z) = N(0, 1)$:

$$\mathcal{L}_{VAE} = \|x - \Phi(\Psi(x))\|^2 + \eta KLD(q(z)\|p(z|x)), \quad (1)$$

where the first term represents the reconstruction error, the second term calculates KLD, and η is the harmonic parameter to balance the two. Besides, cVAE is a variant of VAEs in which the input is conditioned, and this input condition influences the generated data. For instance, the ground truth of data can be a conditioned input in a classification problem. Figure 2 shows the architecture of cVAEs, which is fed by a parameterized distribution and an input condition. The encoder and decoder blocks are trained jointly with the objective of loss minimization. In this work, we use masses of two particles obtained by the MLP regression network as the input condition, and the output generates two 4-vector, i.e. (E, p_x, p_y, p_z) for each particle. An example of cVAEs can be found in [13].

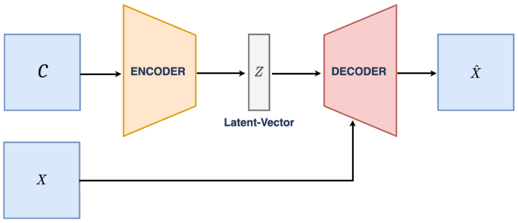

FIG. 2: The architecture of cVAEs network.

III. Methodology

The experimental physics community is a resource consuming task since it is now relying on energies that gigantic particle accelerators cannot even reach. In addition, new accelerators will cost billions of dollars at a minimum. Thanks to growing computer power in recent years, computer simulation is feasible for scientists to simulate experimental research. Likewise, in the task of collision event simulation, applying computer-based approaches is utilized in this study.

A. Monte Carlo Sampling

We use MC sampling for this experiment in which two children particles are sampled under the condition provided in the equation 2, called **energy–momentum relation**, or **relativistic dispersion relation**.

$$E^2 = (Pc)^2 + (m_0c^2)^2, \quad (2)$$

where E, m_0, and P indicate total energy, invariant mass, and momentum of a particle, respectively, and the constant c is the speed of light. For simplification, c is considered one ($c = 1$) in this work that must preserve the equation 3.

$$E^2 = P^2 + M^2. \quad (3)$$

Notable, momentum-energy is a four-vector in space-time (equation 4) in special relativity.

$$P = (p^0, p^1, p^2, p^3) = (\frac{E}{c}, p_x, p_y, p_z) \quad (4)$$

Since a generated particle has a momentum 4-vector and its corresponding mass, a five-value vector for each particle should be obtained (E, p_x, p_y, p_z, M). Therefore, the final version of the made dataset has ten features (five elements for each offspring particle). Algorithm 1 implements the MC method to generate particle x_1 and its antiparticle x_2. First, masses of particles are sampled uniformly in the range of (10, 150). Then, we sample spherical elements for particle x_1, i.e. θ_1, ϕ_1, r_1 from uniform distributions in range of $(0, \pi)$, $(0, 2\pi)$ and $(0, E_{total})$, respectively. Corresponded mentioned spherical elements with Cartesian follow by the below equations.

$$r = \sqrt{p_x^2 + p_y^2 + p_z^2} \quad (5)$$

$$\theta = \arccos(\frac{p_z}{r}) \quad (6)$$

$$\phi = \arctan(\frac{p_y}{p_x}) \quad (7)$$

Algorithm 1 Monte Carlo Sampling Events

```
Require: $Nr.instance = 10^5$, $E_{total} = 400$
  for in range(Nr.instance) do
    while $True$ do
      $M_1 \leftarrow uniform(10, 150)$
      $M_2 \leftarrow uniform(10, 150)$
      $\theta_1 \leftarrow uniform(0, \pi)$                ▷ Polar angel
      $\phi_1 \leftarrow uniform(0, 2\pi)$             ▷ Azimuthal angel
      $r_1 \leftarrow uniform(0, E_{total})$
      $p_{1x}, p_{1y}, p_{1z} \leftarrow to\_Cartesian(r_1, \theta_1, \phi_1)$
      $P_1 \leftarrow \sqrt{p_{1x}^2 + p_{1y}^2 + p_{1z}^2}$
      $E_1 \leftarrow \sqrt{M_1^2 + P_1^2}$
      $E_2 \leftarrow E_{total} - E_1$
      $\theta_2, \phi_2 \leftarrow (\pi - \theta_1), (\pi + \phi_1)$
      $P_2 \leftarrow \sqrt{E_2^2 - M_2^2}$
      $p_{2x}, p_{2y}, p_{2z} \leftarrow to\_Cartesian(P_2, \theta_2, \phi_2)$
      if $E_1 \geq E_{total}$ or $(E_2^2 - M_2^2) < 0$ then
        $miss_{cnt} + +$
      else if  then
        Break
      end if
    end while
  end for
```

There are different types of detectors, such as Atlas, LHCb, CMS, and ALIS. A practical challenge in LHC is the non-optimal precision of the detectors performance [14]. Since the detectors record the events with some errors, we smear data by a multiplicative Gaussian noise to simulate the detection error.

B. Multi-layer Perceptron Regressor

In this subsection, we propose to train an MLP regression network to predict the masses of the two particles given the energy and momentum as the input data. We split the MC data to obtain the train-validation set. Afterward, the network is trained on the train set. We later use the predicted masses of validation data as a cVAE network. Specifically, the network's input is a 4-vector (E, p_x, p_y, p_z) of two particles, and the output is their masses. To estimate the uncertainty of the prediction, we use Root Mean Square (RMSE) as follows.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}}, \tag{8}$$

where $\hat{y}_i$ is predicted value and n is the number of data.

C. conditional Variational Auto Encoder (cVAE)

In high-energy particle physics, numerical simulations require enormous computational power. Due to simulation speed and budget, the corresponding scientific progress is limited. Thus, we propose an ML-based generative model, cVAE, to promote the data generation process, where the network's input is masses, and its output is energy-momentum 4-vector of two distinct particles. The condition input is masses that come from the regression model, and the additional input of the decoder is 4-vector from smeared MC samples. Figure 3 illustrates the architecture of the encoder, decoder and the whole VAE's network.

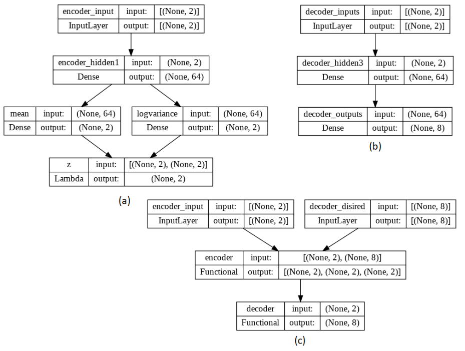
encoder_input | input: | [(None, 2)]
InputLayer | output: | [(None, 2)]
encoder_hidden1 | input: | (None, 2)
Dense | output: | (None, 64)
mean | input: | (None, 64)
Dense | output: | (None, 2)
logvariance | input: | (None, 64)
Dense | output: | (None, 2)
z | input: | [(None, 2), (None, 2)]
Lambda | output: | (None, 2)
(a)
decoder_inputs | input: | [(None, 2)]
InputLayer | output: | [(None, 2)]
decoder_hidden3 | input: | (None, 2)
Dense | output: | (None, 64)
decoder_outputs | input: | (None, 64)
Dense | output: | (None, 8)
(b)
encoder_input | input: | [(None, 2)]
InputLayer | output: | [(None, 2)]
decoder_disired | input: | [(None, 8)]
InputLayer | output: | [(None, 8)]
encoder | input: | [(None, 2), (None, 8)]
Functional | output: | [(None, 2), (None, 2), (None, 2)]
decoder | input: | (None, 2)
Functional | output: | (None, 8)
(c)

FIG. 3: The architecture of top-left: the encoder, top-right: the decoder and bottom: VAE network.

IV. Result and Discussion

In this section, we investigate the performance of deep generative models in simulating energy-momentum 4-vector given by mass in particle collision events. The simulations are implemented in Python and Colab with its publicly available GPU processor to train the neural network. The number of samples is set to 100000 obtained from Monte Carlo with shape $(100000, 10)$ described in Table I for two phases, before and after smearing. Notable, MC experiment takes 20.12 seconds to run with the hit rate 76.55%.

The approach of smearing data is corrupting data with a multiplicative Gaussian noise that is applied %10 on Energy (using G(mean=1, std=0.1) distribution) and %5 on both polar (θ) and azimuthal (ϕ) angels (using G(1, 0.05) distribution). Figure 4 illustrates the energy histogram before and after smearing that shows that the energy conservation is kept before smearing. Both particles' energies have means of around 200, with the max just below 400. Due to Gaussian-based smearing, it is possible to see some energy greater than the system's total energy. However, this deviation violates energy conservation which is expected for some values due to inaccurate recordings by detectors. Generated data for masses of particles before and after smearing is shown in Figure 5. Since we sample both masses from a uniform distribution, we would expect flat histograms before smearing, but it appears a bit different. The reason is related to the order of sampling. Since we start to sample M_1, then we calculate E_1 followed by calculating E_2 based on energy conservation, the possibility of generating M_2 with high values became lower where M_1 had a higher value. To address this problem and make the two distributions more identical, for each sample, we flip a coin to swap the particle vectors leading to Figure 5. Due to using the normal distribution for smearing, flat mass distribution scatters from the average seen in the right-side diagram. As we can see, the values of masses exceed 150, which is anticipated because of multiplicative Gaussian noise. Besides, Figure 6 illustrates the polar and azimuthal angles

	metric	E_1	p_{1x}	p_{1y}	p_{1z}	M_1	E_2	p_{2x}	p_{2y}	p_{2z}	M_2
Sampled data	mean	185.6	-0.4	0.3	0.0	78.7	214.3	0.3	-0.4	-0.1	74.2
	std	81.4	91.3	91.3	128.9	40.2	81.4	106.5	106.7	150.6	40.0
	min	10.1	-364.5	-371.4	-382.0	10.0	10.8	-375.8	-381.3	-382.6	10.0
	max	389.1	367.1	368.8	376.6	149.9	389.8	374.2	373.4	381.2	149.9
After smear	mean	192.6	-0.6	1.1	0.4	96.9	223.9	0.0	0.4	0.3	100.4
	std	88.6	91.4	91.5	128.6	47.5	89.1	106.5	106.9	150.4	50.5
	min	8.2	-359.4	-365.3	-384.6	0.4	9.8	-368.3	-384.1	-382.5	0.3
	max	481.2	366.3	366.9	377.4	338.3	512.8	378.2	381.4	381.2	359.6

TABLE I: The simulated data with MC sampling and after smearing.

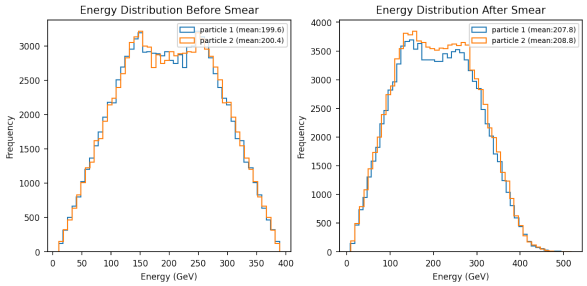

FIG. 4: The energy distribution of two new particles before and after smearing.

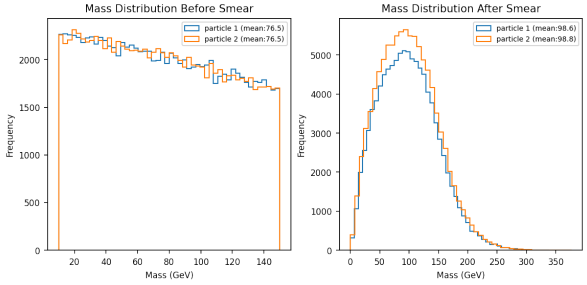

FIG. 5: The mass distribution of two new particles before and after smearing.

for both particles before and after smearing. The result of MC generated momentum is shown in Figure 7. The comparison of relativistic dispersion relation for generated data before and after in Figure 8 indicates that MC method is a proper approach for simulating process by keeping the rule $E^2 - M^2 - P^2$ near zero.

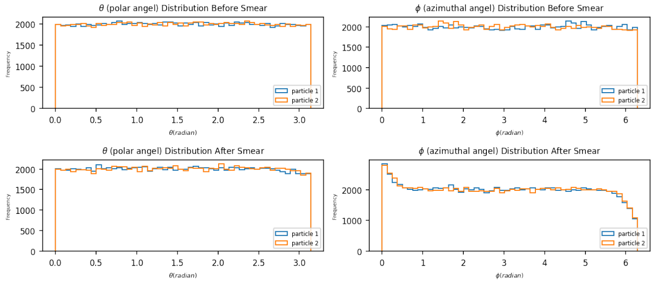

FIG. 6: Polar and azimuthal angles before and after smearing for MC generated particles.

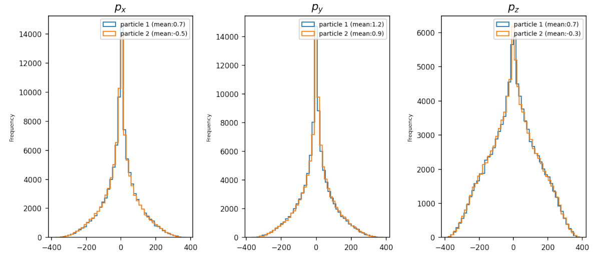

FIG. 7: Momentum elements distribution for both particles x_1 and x_2.

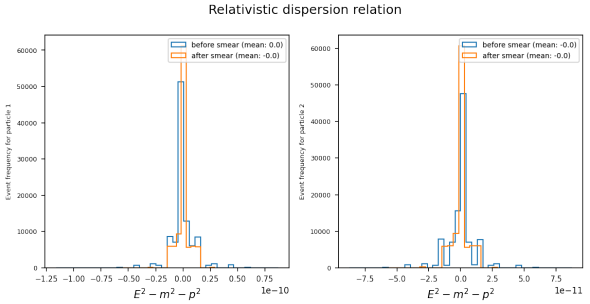

FIG. 8: Relativistic dispersion relation before and after smearing for both particles. Looking at the scale in x-axis reveals that the values are close to zero.

A regression network is trained that is a Multi-Layer Perceptron (MLP) which has four dense layers (8,706 parameters) with activation function *relu* [15] for three middle layers and *linear* for the output layer to predict the invariant masses of generated particles. The input shape is $(samples, 8)$ where features are 4-vectors of two particles, and the output shape is $(samples, 2)$ for two masses should be predicted. Data is split into train and test sets with the portion 70-30% to estimate the uncertainty of the prediction with test data. Hence, the network's performance is evaluated by RMSE between real and predicted masses on test set. RMSE for x_1 and x_2 are calculated 6.20 and 6.28, respectively, which appear promising enough (for smeared masses with the means around 100) to accept that a regressor can learn the pattern in MC data based on relativistic dispersion relation. Besides, Figure 9 shows a proper prediction nearly fitted on the identity line that proves the regressor learn pattern of data.

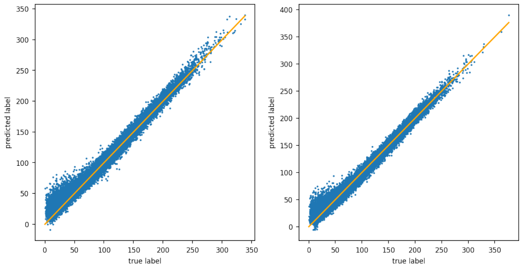

FIG. 9: Regression network prediction for masses of both particles (left: M_1 and right: M_2).

A generative model is built and trained to use it as event data generator. The input of the encoder is predicted masses (m_pred = *regressor_model.predict(X_test)*), and the output is the corresponding energy-momentum 4-vector for each particle. Therefore, the shape of input is $(100000, 2)$ and output's is $(100000, 8)$. We also add a dummy input (4-vector of corresponded m_pred), which goes forward through the network until the decoder is used to calculate the loss. We use *sigmoid* as activation function in cVAE and 'sigmoid' for hidden layers' activation function, 'Adam' as the optimizer with a learning rate 10^{-3}. A visualization of 2d latent space can be seen in Figure 10, indicating that the encoder successfully builds a specific manifold that sampling from this distribution might lead to purposed data.

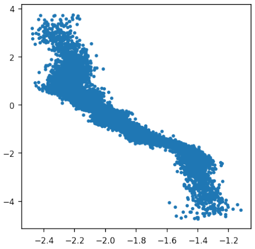

FIG. 10: The distribution of 2D latent space.

To evaluate the performance of VAE, we can look at energy distributions of real and fake data. Figure 11 shows that generated energies tend to real values. Besides, Figure 12 shows the distribution of real and fake masses for both particles. KL-divergence is the other evaluation metric used in this phase. The mentioned value for real and fake energies and masses are 0.14, 0.11, 1.26,

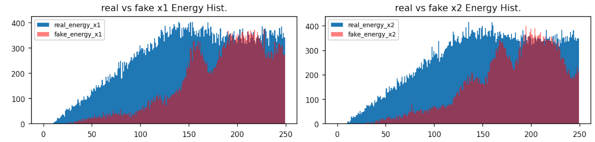

FIG. 11: Energy distribution of real and fake data.

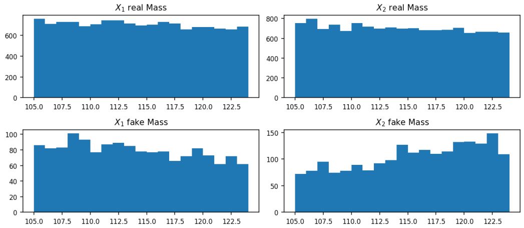

FIG. 12: Mass distribution of real and fake data.

and 1.19, respectively. These small values indicate the similarity of two distributions of real and fake data.

V. Conclusion

In this paper, we proposed a fast ML-based method, cVAE, to simulate particle collision events. Through a 3-pahse experiment, we first simulated an electron-positron collision event that led to two new particles with the MC approach and smeared them with multiplicative Gaussian noise. The result was promising, as the relativistic dispersion relation values were approximately zero before and after smearing. Afterward, we trained a regression network to predict masses of new particles given the MC energy-momentum 4-vector. The regressor performance was evaluated by RMSE, 6.20 and 6.28, for two new particles, proving the network is thriving in generating masses. Last but not least, we built a cVAE to generate 4-vector given the masses obtained by the regressor. The result showed that this model can be used for fast generating data.

In case of generative models with different shapes of input and output, conditional GANs is designed with this purpose can be applied for the future works.

A. Appendix

Figure 13 shows the architecture of the MLP regression network. Besides, the learning curve of the mentioned network is shown in Figure 14.

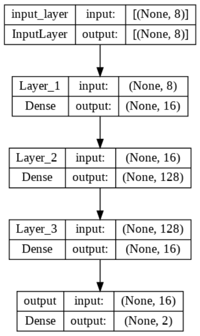

FIG. 13: a multi-layer perceptron neural network is trained for predicting masses of particles.

Lastly, Figure 15 shows the learning of cVAE that is converged.

References

[1] Oscar Barbalat. *Applications of particle accelerators*. Tech. rep. CERN, 1990.

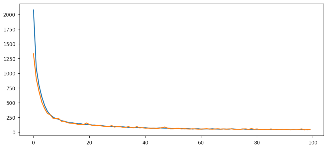

FIG. 14: The learning curve of regression model after 100 epochs.

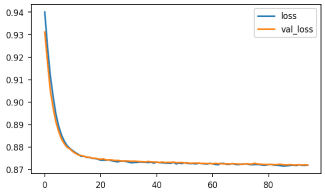

FIG. 15: The learning curve of cVAE model after 100 epochs.

[2] Christian Tobias Preuss. "Improving the Efficiency and Accuracy in Monte-Carlo Event Generation". PhD thesis. Monash University, 2021.

[3] N Fuster-Martínez, R Bruce, F Cerutti, R De Maria, P Hermes, A Lechner, A Mereghetti, J Molson, S Redaelli, E Skordis, et al. "Simulations of heavy-ion halo collimation at the CERN Large Hadron Collider: Benchmark with measurements and cleaning performance evaluation". In: *Physical Review Accelerators and Beams* 23.11 (2020), p. 111002.

[4] GEANT Collaboration, S Agostinelli, et al. "GEANT4–a simulation toolkit". In: *Nucl. Instrum. Meth. A* 506.25 (2003), p. 0.

[5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. *Generative Adversarial Networks*. 2014. DOI: `10.48550/ARXIV.1406.2661`. URL: `https://arxiv.org/abs/1406.2661`.

[6] Jesus Arjona Martínez, Thong Q Nguyen, Maurizio Pierini, Maria Spiropulu, and Jean-Roch Vlimant. "Particle Generative Adversarial Networks for full-event simulation at the LHC and their application to pileup description". In: *Journal of Physics: Conference Series*. Vol. 1525. 1. IOP Publishing. 2020, p. 012081.

[7] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[8] Mary Touranakou, Nadezda Chernyavskaya, Javier Duarte, Dimitrios Gunopulos, Raghav Kansal, Breno Orzari, Maurizio Pierini, Thiago Tomei, and Jean-Roch Vlimant. "Particle-based Fast Jet Simulation at the LHC with Variational Autoencoders". In: *arXiv preprint arXiv:2203.00520* (2022).

[9] Johannes Albrecht, Antonio Augusto Alves, Guilherme Amadio, Giuseppe Andronico, Nguyen Anh-Ky, Laurent Aphecetche, John Apostolakis, Makoto Asai, Luca Atzori, Marian Babik, et al. "A Roadmap for HEP Software and Computing R&D for the 2020s". In: *Computing and software for big science* 3.1 (2019), pp. 1–49.

[10] Yasir Alanazi, Nobuo Sato, Pawel Ambrozewicz, Astrid Hiller-Blin, Wally Melnitchouk, Marco Battaglieri, Tianbo Liu, and Yaohang Li. "A Survey of Machine Learning-Based Physics Event Generation". In: *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, 2021. DOI: `10.24963/ijcai.2021/588`. URL: `https://doi.org/10.24963%2Fijcai.2021%2F588`.

[11] Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. “Normalizing flows: An introduction and review of current methods”. In: *IEEE transactions on pattern analysis and machine intelligence* 43.11 (2020), pp. 3964–3979.

[12] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. “Variational inference: A review for statisticians”. In: *Journal of the American statistical Association* 112.518 (2017), pp. 859–877.

[13] Adrian Alan Pol, Victor Berger, Cecile Germain, Gianluca Cerminara, and Maurizio Pierini. “Anomaly detection with conditional variational autoencoders”. In: *2019 18th IEEE international conference on machine learning and applications (ICMLA)*. IEEE. 2019, pp. 1651–1657.

[14] Oliver Buchmüller. “LHC detectors: commissioning and early physics”. In: *Journal of Physics: Conference Series*. Vol. 110. 1. IOP Publishing. 2008, p. 012015.

[15] Abien Fred Agarap. “Deep learning using rectified linear units (relu)”. In: *arXiv preprint arXiv:1803.08375* (2018).