Flash Simulation at the LHC with Variational Autoencoders

Submitted by: Glenn Anta Bucagu

Supervisor: Dr. Maurizio Pierini

27th August 2021

1 Summary

We investigate the use of variational autoencoders in flash simulation of proton-proton collision events at the LHC. Given a dataset of over 2 million events, with 9 generator-level features and their software reconstructions, we train a variational autoencoder to reconstruct these features. Through analysis of distributions and residuals, we assess the validity of this flash simulation method, identify weaknesses and propose solutions that could be applied in the future to a wide range of LHC experiments.

Keywords— Deep leaarning, LHC, Flash Simulation, Generative Models, Variational Autoencoder

Contents

2 Introduction

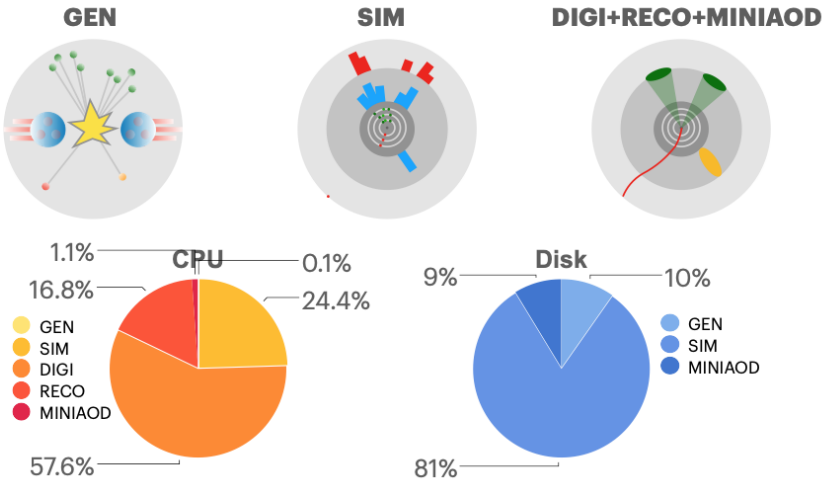
2.1 Current Context

CERN's Large Hadron Collider (LHC) is the site of several experiments. In particular, high energy proton-proton collisions are studied to re-enforce our current understanding of particle physics and possibly make new discoveries. In most cases, such experiments follow a data-driven methodology, meaning that conclusions are entirely drawn from data analysis and interpretation directly from the experiments themselves. In the case of the LHC, it is useful to generate synthetic data from simulated proton-proton collisions. CERN relies on experiment-specific software derived from the specialized software like GEANT4 [3]. By leveraging Monte Carlo simulation techniques, GEANT4 produces highly accurate simulated data. Previous experiments on the LHC have demonstrated a simulation error on the scale of a few percent. This state of the art accuracy does come at a cost, particularly in terms of storage and power consumption. With the upcoming High Luminostiy Large Hadron Collider (HL-LHC) producing data at a larger scale, it is not sustainable for GEANT4 to generate high accuracy data at the same rate as the LHC generates real data. The relative lack of synthetic data has been shown to lead to relatively large systematic uncertainties in LHC experiment analyses [1]. This is a particular problem for high precision measurements of Standard Model Processes with large datasets. The scientific community is therefore called to reduce computing resources for simulation workflows by at least one order of magnitude, to avoid offsetting the expected accuracy increase from the HL-LHC.

2.2 Data Generation Workflow

Consider an event simulation workflow at the Compact Muon Solenoid (CMS) experiment.

Figure 1: Event generation workflow at the CMS experiment



In figure 1, we observe different phases of the CMS event simulation workflow. First, proton-proton collisions are simulated up to the production of stable and observable particles. We call this generator-level events (GEN). This is effectively what a perfect detector would observe. The detector response is then simulated by the GEANT4 library (SIM). The resulting energy deposits are converted into digital signals (DIGI) which are reconstructed by the GEANT4 library (RECO). The signals account for detector imperfections and limited experimental resolution by incorporating some uncertainties [2]. It is at this stage that high-level objects like jets are created. Beginning with the RECO data format, a reduced analysis format (MINIAOD) is derived. The bottom of figure 1 illustrates the computing resource breakdown for the generation workflow of the CMS experiment in terms of CPU and Disk usage.

2.3 The Role of Deep Learning

In recent years, deep learning (DL) generative algorithms have been proposed as potential alternatives to speed up GEANT4 [4]. For example, one can use an image representation of LHC collisions as input for some generative model. This model would reconstruct the input and therefore bypass GEANT4 when simulating detector response. While these models are promising, there is still work to be done to fully integrate them in the centralized computing infrastructure of typical LHC experiments. For example, a DL model would need to account for the irregular geometry of a typical detector by avoiding the collision-as-image paradigm. The DL model would also need to deliver a dataset in a format compatible with downstream reconstruction software.

Some studies have investigate a novel approach: instead of training DL models for broad generation tasks, one could instead design analysis-specific generators, with the aim of learning and reconstructing arrays of physical quantities relevant to a given experiment. This method effectively reduces each event representation to a vector of

meaningful measurements. A successful model could generate a relatively large number of events, in a relatively short amount of time with relatively small storage requirements and minimal data pre-processing. The two main generative models considered were generative adversarial networks (GANs) and variational autoencoders (VAEs). On a fundamental level, each of these algorithms learns the multi-dimensional probability density function (PDF) that governs the array of physical quantities, and uses this PDF to reconstruct data. There does appear to be a trade-off between statistical precision (a quantity known to decrease as the amount of generated events increases) and the systematic uncertainty that comes from a less than perfect construction of the underlying multi-dimensional PDF. The amount of training data tends to be the limiting factor in the performance of generative models. By balancing the statistical uncertainty when augmenting the dataset and the systematic uncertainties associated with the DL model's learning process, one can hope to achieve good results. The idea of what is "good" is application-specific, but this generally determines whether a GAN or VAE is a computationally convenient alternative to GEANT4 for example.

3 Objective

We are given a benchmark dataset of 9 physical quantities obtained from W+1 jet events produced in $\sqrt{s} = 13$ TeV proton-proton collision events. In particular, we have the generator-level features $\vec{x_G}$ and the corresponding software reconstruction $\vec{x_R}$. We aim to train a VAE to produce $\vec{x_{DL}}$, a reconstruction that is purely a function of $\vec{x_G}$ and $\vec{x_R}$. A successful model could lay the foundation for future, more sophisticated use of generative models in particle physics.

In High Energy Physics applications, large values of $\vec{x_G}$ values can generally be obtained in relatively short time periods, we claim that the DL approach would result in a considerable decrease in computing resources. As opposed to storing individual collision data on the scale of $\mathcal{O}$(10kB) for analysis-ready object collections to $\mathcal{O}$(1MB) for raw data, one would only be need to work with a relatively limited set of relevant physical quantities.

4 The Dataset

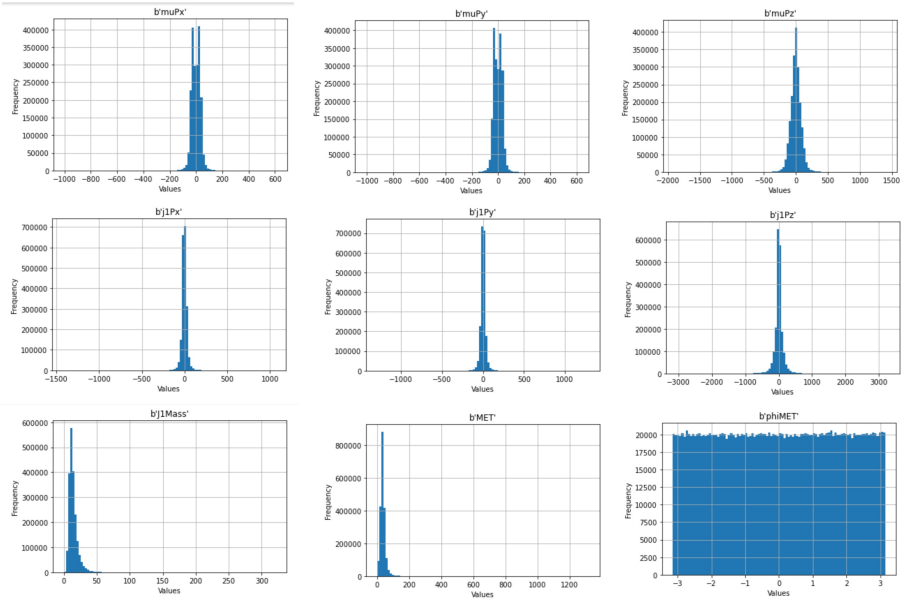
Our feature vector $\vec{x}$ contains the following nine quantities:

- Muon momentum in Cartesian coordinates: p_x^μ, p_y^μ and p_z^μ.
- Jet momentum in Cartesian coordinates: p_x^j, p_y^j and p_z^j.
- Logarithm of the jet mass: $\log M_j$.
- Missing transverse energy in polar coordinates: MET and ϕ_{MET}.

4.1 Pre-processing

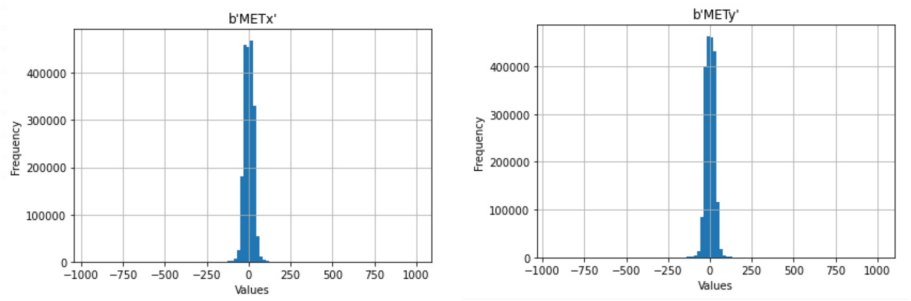
We study the individual distributions of each of the given 9 variates.

Figure 2: Distributions of raw generator-level input features



In figure 2, we see the muon momentum x and y components appear to follow bimodal distributions. The z component appears to be unimodal. All three coordinates appear approximately Gaussian and centred around 0. The jet momentum coordinates also appear somewhat Gaussian and centred near 0. In contrast, the logarithm of the jet mass and the missing energy magnitude appear to be bounded below, unimodal and skewed. Finally, the angular component of the missing tranverse energy appears to be uniform continuous in the range $[-\pi, \pi]$. Bimodal distributions, bounded distributions and uniform distributions are likely to be an issue for a VAE. This is because non-continuous, hard edges in the distribution are difficult to learn via continuous processes like the propagation techniques employed by VAEs. We have several techniques to attack this problem. For the missing energy, we can convert MET and ϕ_{MET} into Cartesian coordinates. For the bimodal and bounded distributions, we could either employ pre-processing techniques like power transformations or quantile transformations to map these distributions to Gaussian distributions. In the case of our benchmark problems, it turns out the VAE performance is unchanged, but this does not discredit these methods in a general case. In a more sophisticated setting, one could use normalizing flows to apply a series of invertible mappings that transform our difficult-to-learn distributions into more continuous distributions.

Figure 3: Distributions of x and y components of missing transverse energy



In our case, we only address the uniform distributions to yield the plots in figure figure 3.

5 Model Training

The benchmark dataset contains just over 2 million events, which are partitioned into 60% for training, 20% for hyperparamter tuning and 20% for validation and assessment. Prior to training, both $\vec{x_G}$ and $\vec{x_R}$ are scaled using a standard scaler. This reduces each distribution to 0 mean and unit variance, ultimately speeding up the training process. Our encoder contains 4 layers, the first with 9 neurons and others each with 64 neurons, batch normalization and ELU activation. Our decoder also contains 5 layers each with 64 neurons (except the last which contains 9 neurons), batch normalization and ELU activation. The final layer has a linear activation function. VAEs are strictly unsupervised, but we have an input $\vec{x_G}$ and target $\vec{x_R}$. Despite the high accuracy of the software used to reconstruct $\vec{x_R}$ from $\vec{x_G}$, there is still an average error of a few percent between input and target. We decide to incorporate a loss function that accounts for this discrepancy:

$$\mathcal{L}(\vec{x_{DL}}, \vec{x_G}, \vec{x_R}) = \beta \frac{1}{9} \frac{1}{n} \sum_{j=1}^{9} \sum_{i=1}^{n} (x_{DL}^{i,j} - x_R^{i,j})^2 + (1-\beta)\mathbb{E} \log \left(\frac{Q(z|\vec{x_G})}{P(z|\vec{x_G}} \right) \tag{1}$$

where z is the latent variable obtained via the encoder. $P(z|\vec{x_G}$ is the true PDF of z conditional on $\vec{x_G}$ and $Q(z|\vec{x_G}$ is the PDF of z conditional on $\vec{x_G}$ estimated by the VAE.

The mean square error component ensures the VAE prediction $\vec{x_{DL}}$ is close to the software reconstruction. The Kullback-Leibler Divergence component ensures the distribution of $\vec{x_{DL}}$ resembles the distribution of the generator-level input. For our particular case, we train our base model with $\beta = 0.5$, but this is a hyperparameter that can be tuned via grid search, random search or Bayesian optimization. Another important hyperparameter is the latent space, which is 12 for our base model and later tuned via Bayesian optimization in the GPyOpt library to 18. We train the VAE in the tensorflow convolutional setting, using the Adam optimizer, with an epoch-dependent learning rate of $0.01/(1 + n_{epoch})$ with $n_{epoch} = 100$ epochs. The data are

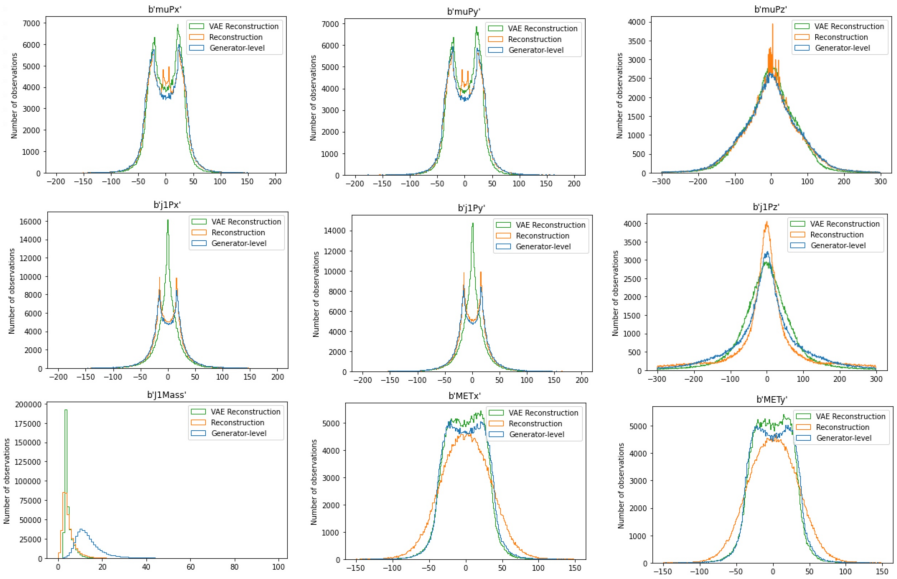
arranged in batches of size 128 and shuffled after reach round of training to avoid overtraining. We also incorporate early stopping, a mechanism that halts model training if the validation loss fails to decrease after 20 consecutive epochs. The VAE parameters with the lowest validation loss are selected.

6 Results

6.1 Preliminary Predictions

The trained VAE is used to generate samples of reconstructed events from generator-level and software reconstructions. We evaluate the model's performance by comparing the output distributions with those obtained by software on the same generator-level events. We also look at the relative residual distributions.

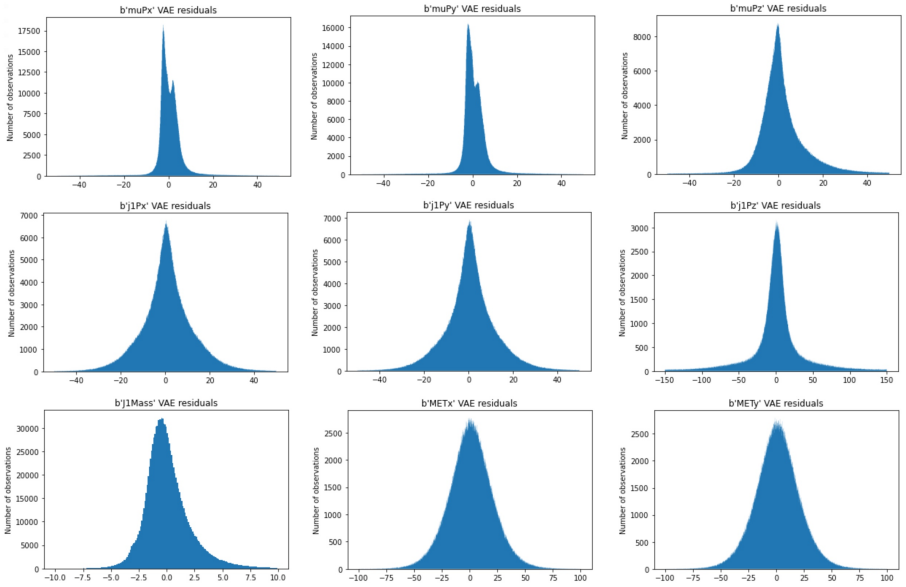
Figure 4: Distributions of VAE Predictions



Looking at figure 4, we see the muon momentum coordinates are well predicted by the VAE as the distributions overlap over the whole domain. There is some statistically significant noise, particularly around 0, but this can be attributed to the software reconstruction mechanism which accounts for random errors stemming from resolution limits and detector imperfections. In contrast, the jet x and y coordinates are not well predicted, particularly around 0. This can be, in part, attributed to the very high peaks in the two modes for each distribution. This is equivalent to a hard edge which is difficult for a VAE to learn in general. The z coordinate relatively well-predicted, but it achieves a relatively low peak around 0 unlike the software reconstruction and generator-level measurements. The logarithm jet mass is not well predicted specifically for measurements close to 0. This can be, in part, attributed to the discrepancy

between the generator-level input and the reconstructed target, as well as the hard edge at 0. Finally, the missing energy components appear to be equally well predicted, despite the slight discrepancy in the distributions of the generator-level and software reconstruction.

Figure 5: VAE Residual Distributions



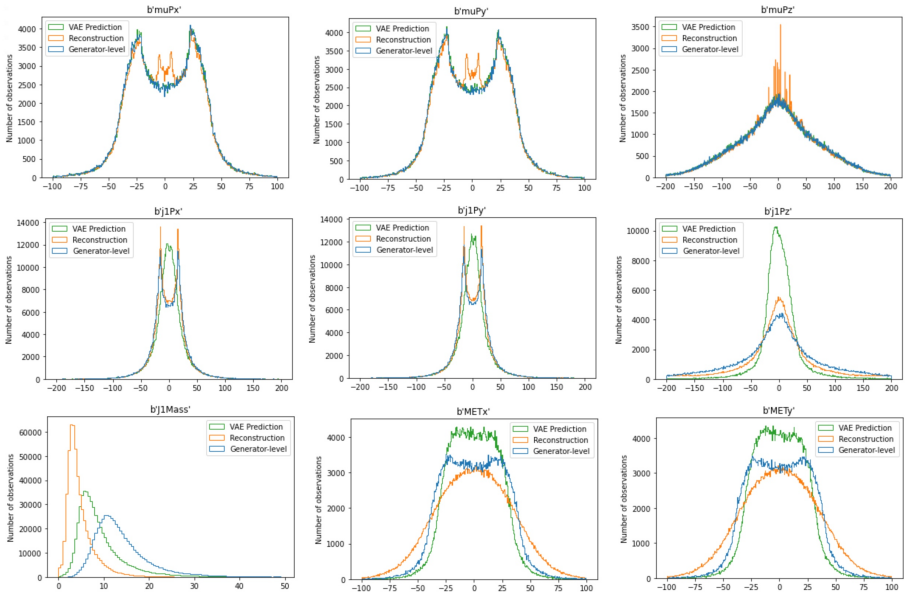
Looking at figure 5, we see that all residual settings are close, if not, Gaussian. The muon momenta x and y components appear to have very sharp peaks, with some bimodal behavior. This is to be expected since the generator-level and software reconstructions follow bimodal distributions. The distribution of the z component of the muon momentum appears slightly skewed positively, suggesting there may be a slight systematic error in the VAE. The residuals for the jet momentum components seem reasonably symmetric and close to Gaussian with 0 mean. The residual for the x and y components of the missing transverse energy seem reasonably Gaussian and 0-centred, suggesting their distributions are well-understood by the VAE mechanism. The residuals for the logarithm of the jet mass follow a slightly positively skewed distribution. This, in conjunction with the plot of the out-of-sample VAE reconstructions in figure 4 suggest this variate is poorly predicted by the VAE. One alternative could be to use another function of the jet mass. The logarithm is bounded below for positive masses and this introduces the hard edge problem we previously addressed. One could use a function that spreads out jet mass values continuously over a symmetric range for example. Such distributions seem to lend themselves well to VAE reconstructions.

6.2 Reformulating the problem

Our results show that a significant discrepancy between the generator-level data and reconstructed data can negatively affect VAE predictions, particularly because of the

way our loss function is formulated. In the spirit of classical regression problems, we could use $x\vec{_{DL}}$ as input to the VAE, and have $\vec{x_G} - \vec{x_R}$ as the target.

Figure 6: VAE Predictions Reformulated



In figure 6, we see an overall improvement in VAE predictions. In particular, the muon momentum distributions overlap perfectly, ignoring the effect of random noise around 0. The jet momenta predictions in the x and y coordinates improve drastically, but the harsh peaks around 0 are still not well predicted by the VAE. The jet momentum z coordinate is not as well predicted as in the previous problem formulation, particularly as the VAE tends to overestimate small quantities around 0. The jet mass also appears to be better predicted, but again this is limited by the drastic difference in the generator and reconstruction distributions. The missing transverse energy components seem to be predicted equally well in both paradigms.

7 Conclusions

We carried out a small scale test for a new flash simulation which exploits variational autoencoders to convert an analysis-specific representation of collision events at generator level to the corresponding reconstruction. The aim was to assess the validity of this data augmentation strategy by identifying strengths, weaknesses and ways the strategy could be generalized to more LHC experiments. If this approach is successful on a large scale, one could replace any request of N events with a request of $n < N$ events. The remaining $N - n$ events could be kept at generator level. Avoiding the detector simulation and reconstruction process for these $N - n$ events could save on scarce resources, particularly in the current context of the HL-LHC.

We demonstrated that although a simple VAE achieves a good performance, it requires careful data pre-processing. In particular, VAEs have been shown to be relatively poor at learning non-smooth distributions. Such distributions are likely to be exhibited in LHC experiments, so it is important to have solutions to this problem. One approach could be to use normalizing flows: use a deep learning model to transform the difficult-to-learn distribution into a more desirable distribution via a sequence of continuous and invertible mappings. Typically, one would aim for Gaussian distributions, but depending on the experiment in question, there may be better candidates. This would involve training another deep learning model, which would entail more resource allocation. Another approach would be the transformation of the difficult-to-learn distribution via a simple function. Depending on the experiment and data at hand, one could use power transformations, quantile transformations, polynomial transformations or change of coordinates. The difficulty here is that inverting the VAE predictions may disproportionately highlight the VAE's poor predictions. Nonetheless, this method is less resource-intensive than normalizing flows. In the case where there is a significant difference between the given generator level and reconstructed events, one could train the VAE to predict the difference between the input and target, as this has been shown to be successful in most cases. Even in the best of settings, there is likely to be some edge effects which are not well learned by the VAE. One approach could be to cherry-pick activation functions for such variables and emphasize the training process at the edge of the bounded domains. The downside is that this method is not easily generalize to any experiment.

8 Acknowledgements

This project would not have been possible without a dedicated effort from Dr. Maurizio Pierini and Mary Toulakanou. VAEs in particle physics are not thoroughly published in research, so attacking issues required experience, collaboration and creativity. Dr. Pierini's and Ms. Toulakanou's valuable insight helped me build good intuition with regard to the underlying VAE mechanism. I am forever grateful.

References

[1] Johannes Albrecht, Antonio Augusto Alves, Guilherme Amadio, Giuseppe Andronico, Nguyen Anh-Ky, Laurent Aphecetche, John Apostolakis, Makoto Asai, Luca Atzori, and et al. A roadmap for hep software and computing r&d for the 2020s. *Computing and Software for Big Science*, 3(1), Mar 2019.

[2] C. Chen, O. Cerri, T. Q. Nguyen, J. R. Vlimant, and M. Pierini. Analysis-specific fast simulation at the lhc with deep learning. *Computing and Software for Big Science*, 5(1), 2020.

[3] S. Agostinelli et al. Geant4—a simulation toolkit. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 506(3):250–303, 2003.

[4] Sydney Otten, Sascha Caron, Wieske de Swart, Melissa van Beekveld, Luc Hendriks, Caspar van Leeuwen, Damian Podareanu, Roberto Ruiz de Austri, and Rob Verheyen. Event generation and statistical sampling for physics with deep generative models and a density information buffer, 2021.