Generative Models for Fast Cluster Simulations in the TPC for the ALICE Experiment

Kamil Deja[1], Tomasz Trzciński[1] and Łukasz Graczykowski[2]
for the ALICE Collaboration

[1]Institute of Computer Science, [2]Faculty of Physics
Warsaw University of Technology, Poland,
`kdeja@stud.elka.pw.edu.pl`, `t.trzcinski@ii.pw.edu.pl`,
`lukasz.graczykowski@pw.edu.pl`

Abstract. Simulating the possible detector response is a key component of every high-energy physics experiment. The methods used currently for this purpose provide high-fidelity results. However, this precision comes at a price of a high computational cost, which renders those methods infeasible to be used in other applications, e.g. data quality assurance. In this work, we present a proof-of-concept solution for generating the possible responses of detector clusters to particle collisions, using the real-life example of the Time Projection Chamber (TPC) in the ALICE experiment at CERN. We introduce this solution as a first step towards a semi-real-time anomaly detection tool. It's essential component is a generative model that allows to simulate synthetic data points that bear high similarity to the real data. Leveraging recent advancements in machine learning, we propose to use state-of-the-art generative models, namely Variational Autoencoders (VAE) and Generative Adversarial Networks (GAN), that prove their usefulness and efficiency in the context of computer vision and image processing. The main advantage offered by those methods is a significant speedup in the execution time, reaching up to the factor of 10^3 with respect to the GEANT3, a currently used cluster simulation tool. Nevertheless, this computational speedup comes at a price of a lower simulation quality. In this work we show quantitative and qualitative limitations of currently available generative models. We also propose several further steps that will allow to improve the accuracy of the models and lead to the deployment of anomaly detection mechanism based on generative models in a production environment of the TPC detector.

1 Introduction

High-energy physics (HEP) experiments, including those conducted at the Large Hadron Collider (LHC) [1], rely heavily on detailed simulations of the detector response in order to accurately compare recorded data with theoretical Monte Carlo models. This simulated data is used to understand the physics behind the real collisions registered in the experiment. The simulations are also used to adjust the experimental data for detector's inefficiencies and various conditions observed in the detector during the data taking procedure.

Traditional approaches to the simulation of the detector response use Monte Carlo methods to generate collisions. To properly simulate events, those methods need to synthesize propagation of every simulated particle through the detector's medium and model every interaction with the experimental apparatus using a transport package, usually GEANT3 [2], GEANT4 [3] or FLUKA [4]. This approach comes with disadvantages. First of all, those packages require specific implementations of the interactions of highly energetic particles with matter that occur inside the detector. In practice, the physics regulating those interactions is complex and often difficult to simulate. Secondly, the simulation methodology that relies on Monte Carlo methods is computationally expensive and scales linearly with the amount of simulated collisions.

To address the above shortcomings of a Monte Carlo-based simulation, we propose a fundamentally different approach to simulations in high-energy physics experiments that relies on generative models. The proof-of-concept solution, that implements this approach is based on the recently proposed models, such as Variational Autoencoders [5] and Generative Adversarial Networks [6]. The main idea behind using those models for simulations is to learn how to synthesize the situation of the detector after a collision from the data, instead of modeling every interaction of the particles.

Although this approach implies several limitations, especially linked to the inability of modeling the entire collision data, we believe that the proposed method can be successfully applied in other high-energy physics applications, *e.g.* anomaly detection.This is mainly thanks to the computation efficiency of the proposed approach. We will address the problem of modeling the entire collision data in the future work.

As a first step towards implementing our solution in production, we perform a set of experiments that aim at simulating clusters of the Time Projection Chamber (TPC) [7, 8] detector of the ALICE (A Large Ion Collider Experiment) [9] experiment at the LHC. We present quantitative, qualitative and computational cost evaluation of our method for several generative models and their variations, including standard Variational Autoencoder, Deep Convolutional Generative Adversarial Networks [10] and Long Short Term Memory [11] based networks. We also show how to improve the results obtained from those models using a recently proposed training method called a Progressive GAN [12].

To our knowledge, this work represents the first attempt to use machine-learning generative models for clusters simulation in any high-energy physics experiment currently operating at CERN. The main contribution of this paper is a prototype of a proof-of-concept method for event simulation in the TPC detector that relies on generative models. We provide here an extensive overview of the recently proposed generative models that can be used in this context, as well as a detailed evaluation of their current performances when applied to cluster simulation. Evaluation results show a promising speedup over currently used Monte Carlo based simulation methods reaching up to 10^3, however, at the expense of simulation quality. We address this problem in the future work and show how to improve the generative models in the context of cluster simulation.

The remainder of this paper is organized as follows. In the next section, we describe related works. We then overview generative models that can be used in the proposed solutions and present the core idea behind its adaptation. Finally, we describe the dataset and present the results of our evaluation. In the last section, we conclude this work and outline the next steps.

2 Related Work

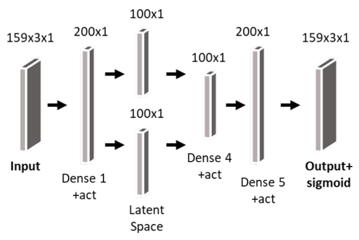
Currently used methods start from a collision simulation using a Monte Carlo generator, than different transport packages such as GEANT3 [2], GEANT4 [3] or FLUKA [4] are used to synthesize propagation of particles. At first, they provide space points where a given energetic particle has deposited some or its energy, as well as simulate the secondary particle showers in the detectors which leave their signal. Those space points have to be further translated to a simulated electronic signal from the detector sensors, as it would appear in a real collision. In the tracking detectors, those electronic signals, called *digits*, are then used to calculate *clusters*, a set of space-time digits from the adjacent sensitive elements of the detector that were presumably generated by the same particle crossing these elements.

Machine learning generative models, which we intend to use in the context of high-energy physics, have already proven their potential in numerous research areas related to text-to-image generation [13], conditional image generation for criminal identification purposes [14] or even drug design [15]. Their main research directions, however, are still restricted to applications related to images and videos. The task we aim to address in this paper, differs significantly from those applications. Simulating precise responses of a detector require high-fidelity of the obtained results, measured both qualitatively and quantitatively. This is not the case for artificially generated images or videos where subjective assessments are often the only way to evaluate generated output.

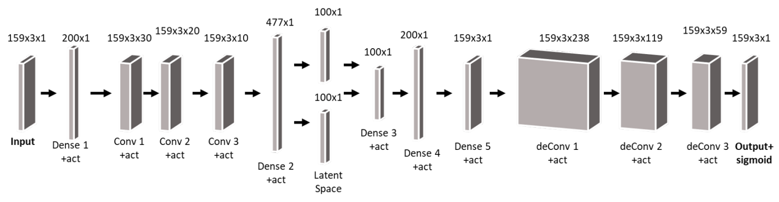
Despite those problems, there are several attempts to use Generative Adversarial Networks for simulating parts of the physics processes, like the very first one in the field: calorimeters response simulation [16]. In that work, the authors generate particle showers in electromagnetic calorimeters. Following the LAGAN processing [17], they adapt a DCGAN architecture to generate a number of 2D images, which they merge, to produce a final 3D Calorimeter response. Although our work draws inspiration from [16], we tackle a different problem related to detector's response simulation, while [16] focuses on generating particle showers.

3 Generative models

In this section we overview two generative machine learning models that we use in our solution and their architectures. First, we describe a Variational Autoencoder (VAE), and propose two different models of a VAE that fit our task. We use



(a) VAE



(b) cVAE

Fig. 1. Variational Autoencoders architectures evaluated in the context of detector's response simulation. Each block represent a network layer with it's size above.

them as an evaluation baseline for the second method which is based on the Generative Adversarial Network (GAN) algorithm. We present those methods in the following sections. We also outline an adaptation of a recently proposed progressive GAN training algorithm [12] for cluster simulation, which according to our studies performs the best out of all evaluated approaches.

3.1 Variational Autoencoder

Variational Autoencoder is a machine learning method that relies on an autoencoder idea proposed in [18]. It is implemented through an artificial neural network that can be used for dimensionality reduction and data compression. The simplest architecture of an autoencoder consists of an input layer with N neurons, where N is a dimensionality of the data, a hidden layer with $K < N$ neurons and the output layer with N neurons, exactly the same size as the input. The goal of an autoencoder is to re-generate the same output as received on the input, while internally reducing the latent representation to a lower number of dimensions.

The Variational Autoencoder (VAE) [5] extends this idea by focusing on the generalisation part of the encoder architecture.VAE re-generates new samples straight from the latent space by modifying the distribution of noise populated in the hidden layer. To ensure that it generates meaningful results, additional normalization constraint is added to the training loss function. It forces a neural network to ensure a normal distribution of the values retrieved as the output of the hidden layer neurons.

In this work, we focus on two architectures of Variational Autoencoders that can be applied in the context of detector's response simulation. First of them, dubbed simply VAE, is a fully connected network with one hidden layer. Second one, which we call cVAE, is a convolutional model with three convolutional and deconvolutional layers in encoder and decoder (generator) layers. Fig. 1 shows both architectures.

3.2 Generative Adversarial Networks

A second family of generative machine learning methods discussed in this work is based on the Generative Adversarial Network [6]. The main concept behind this unsupervised generative model is to train two neural networks to play a *min-max game* between each other.The first one – a *generator* – is trained to generate new data points whose distribution resembles the distribution of real data. The second one – trained as a *discriminator* – aims at predicting whether a given example comes from a real or synthetic distribution. During the training process, both networks are updated iteratively, one at a time, which leads to a simultaneous optimization of loss functions of both generator and discriminator. The *min-max* loss function introduced in [6], encompassing both networks can be written in the following manner presented in the Eq. 1:

$$\underset{G}{min}\,\underset{D}{max}\,V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))], \quad (1)$$

where $p_{data}(x)$ is the distribution of real data, x is a sample from p_{data}, $p(z)$ is a distribution of a noise vector z input into a generator and G and D represent a discriminator.

Architectures The general framework of Generative Adversarial Networks is rather flexible in terms of the networks architecture used as a generator and discriminator.Hence, we propose several architectures that fit to the context of simulating detector's response in a high-energy physics experiment.

More precisely, we evaluate the following models:

– **Multi-layered Perceptron (MLP)**: a baseline Generative Adversarial Network that consists of 4 fully connected layers used as a discriminator network with a generator built exactly in the reverse symmetrical way,
– **Recurrent GAN based on Long-Short Term Memory (LSTM)**: a recurrent network that comprises LSTM units proposed in [11],
– **Deep Convolutional Generative Adversarial Network (DCGAN)**: inspired by [13] multi layered network, with two dimensional convolutional/deconvolutional layers, as shown in Fig. 2.

Progressive GAN A typical approach to increase the performance of neural networks is to add more layers to the model, hence increasing the depth of

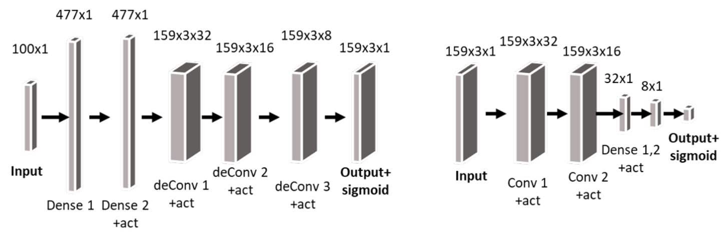


Fig. 2. Architecture of the DCGAN model. Each block represent a network layer with it's size above.

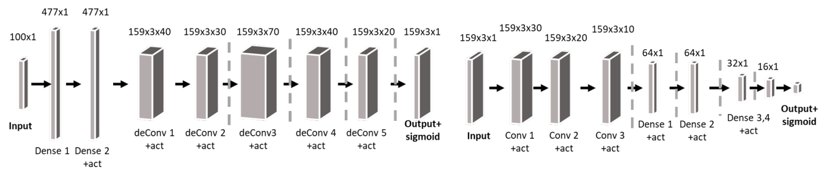


Fig. 3. Architecture for the Progressive GAN model. Dotted lines represent consecutive grow of the network with progressive steps. Each block represent a network layer with it's size above.

a resulting neural network. This, however, comes at a price of more complex training procedure. To evaluate how much performance gain we can obtain by growing our network, we rely on a recently proposed progressive GANs training method introduced in [12].

As presented in Fig. 2, the DCGAN model we evaluate is rather shallow, *i.e.* it consists of only 3 convolutional/deconvolutional layers. We therefore employ the progressive training method for our DCGAN and grow our network by gradually increasing the number of stacked layers. This way, we are able to enhance the resolution of the resulting simulation by gradually increasing the number of possible locations of simulated cluster responses. This is of our particular interest, since in the context of high-energy physics the resolution of the simulated cluster's response has to be relatively high. Our DCGAN is therefore grown in 5 progressive steps, ending with 6 convolutional/deconvolutional layers. Fig. 3 shows the final architecture obtained by the training method.

Implementation details To train all networks, we use dropout [19], Leaky ReLu activation [20], and batch normalization [21]. As our loss function, we take a binary cross-entropy loss. We optimize our loss function with ADAM, a stochastic optimization method [22]. We initialize network weights with the glorot algorithm [23]. The neural networks are implemented in Python using Keras library [24] with TensorFlow backend [25]. Our implementations are based on those enlisted in the large scale study [26].

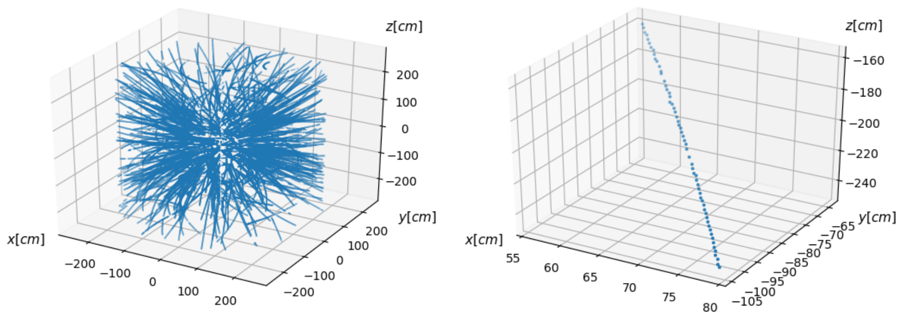


Fig. 4. Input data examples: **(left)** a full simulation of space points occurring after particles collision in the TPC detector of the ALICE Experiment; **(right)** clusters generated by a single particle.

4 Dataset

To evaluate our generative models, we use a dataset of 3D trajectories of particles after collision generated with a Monte Carlo simulation method of PYTHIA 6.4 [27] Perugia-0 [28] model. Our dataset therefore contains a sample of proton–proton collisions at the center of mass energy of $\sqrt{s} = 7$ TeV. After the simulated collision, the generated particles were transported, using GEANT3 [2] package, through the detector medium so that the full simulation of the detector response as well as the track reconstruction were performed. The trajectories correspond to the real traces observed in the TPC detector of the ALICE Experiment at LHC, CERN and the experimental conditions corresponded to 2010 data-taking period of the experiment. Fig. 4 shows sample visualisation of the particle trajectories from the collected dataset. Since the size of a dataset generated this way results in petabytes of data, we randomly sample it to contain around 100 events corresponding to approximately 60.000 data samples.

Each data sample contains a series of 3D coordinates (x, y, z) corresponding to the trajectory of a given particle after the collision. The minimal and maximum values of the coordinates depend on the detector size, which is a cylinder-shaped structure of approximately $5m \times 5m \times 5m$. Since the collision does not have to occurs in the central point of the detector, the minimal distance from the [0,0,0] point, where the trajectories are collected, is 848 mm. The precision of the recorded coordinates is limited to the resolution of the TPC read-out pads, which is between 0.8 to 1.2 mm, depending on the size of the pad [7]. For normalization purposes, the input data coordinates are scaled to fit the $[0, 1]$ interval in each dimension. We also apply other normalization procedures, such as zero-padding the samples for particle trajectories consisting less than 159 points (maximal value in our dataset). Although additional characteristics of the particles can be registered, *e.g.* its energy and speed, for the purpose of the evaluation presented in this work we restrict our data samples to 3D coordinate values.

As shown in Fig. 4, after the collision occurs, up to few hundreds particle trajectories can be observed inside the detector. Although we could attempt to simulate all of those trajectories at once using generative models, we postulate to generate separate trajectories for individual particles first and then merge them together to achieve the final goal. This approach allows us to better control the studies and evaluate the results iteratively, as more trajectories are added. Furthermore, the highest reported resolution of the output generated by the state-of-the-art generative models, such as VAEs and GANs, is 1024×1024 [12]. This resolution is relatively low, when compared to the results obtained in the high-energy physics experiments. To circumvent this limitation of the generative models, we transform our simulation problem into a so-called transactional form and simulate individual points with their (x, y, z) coordinates, which we can then link together to form a full trajectory.

5 Results

In this section, we evaluate the proposed methods based on generative models in terms of both quality of the generated results and the computational cost. As our baseline model, we use the Monte Carlo-based method currently used in the ALICE Experiment to generate the full simulation of the TPC response of the generated particle showers, propagated through the detectors medium by the GEANT3 package. We use the most recent implementation provided as part of the O2 software [8, 29].

To evaluate the quality of the trajectories generated by our method, we refer to the physical property of the particle tracks observed after the collision. When particles move through the detector medium, they form a track whose shape can be defined as a helix of particular parameters. We leverage this phenomenon and calculate for each method an evaluation metric that measures the distance of generated trajectory from a theoretically ideal shape of helix, approximated with an arc. We report this metric as a mean squared error (MSE) between the ideal helix shape and the simulated one and average it across all simulated particles. We give the result both in the coordinate unit system and metric system, scaled using detector's dimensions. Although this approach does not take into account the physical properties of the particles resulting in different parameters of the helix created by each particle, it allows us to compare several unsupervised simulation methods that are not conditioned on particle characteristics. The proposed metric validates the quality of the shape generated by the evaluated simulation methods and serves as a first approximation of a method performance. We intend to extend the proposed metric to handle more complex evaluation procedure in future work.

We also evaluate the computation cost of generating the results for various methods. To that end, we execute the code 10 times on a standalone machine with a Intel Core i7-6850K (3.60GHz) CPU (using single core, no GPU acceleration) and record the average execution time. We then normalize it with respect

Method	**MSE**	**MSE (mm)**	**speedup**
GEANT3 (*current simulation)*	0.00017	0.085	1
Random (*estimated*)	0.33000	166.155	N/A
GAN-MLP	0.11077	55.385	10^4
GAN-LSTM	0.10879	54.395	10^4
VAE	0.07483	37.415	10^4
DCGAN	0.05236	26.18	10^2
cVAE	0.02666	13.33	10
proGAN	**0.00176**	**0.88**	**30**

Table 1: Quality of the Generative models, and their performance comparing to the GEANT3 based simulation solution.

to the baseline method of simulation, namely GEANT3, which is currently used to simulate the detector's response.

The results of our evaluation are presented in Tab. 1. For the baseline simulation method GEANT3, the average mean squared error (MSE) is equal to 0.00017 or 0.085mm. When using a random generator to generate a set of 3D points, the estimated expected MSE is 0.33. Compared to those baselines, the best performing progressive DCGAN model (proGAN) achieves a satisfactory MSE of 0.00176. This result corresponds to a sub-centimetre accuracy of our method, while providing over 30-fold speedup with respect to the baseline method of simulation, a Monte Carlo-based GEANT3. We believe that this result is a promising sign and should inspire further research. Finally, one needs to remember that the goal of our simulation is to provide a benchmark set of positive results that will then be used to assess the health of a detector through a complimentary anomaly detection system. Therefore, the accuracy achieved by proGAN is a good step in this direction.

Regarding the other evaluated models, the Variational Autoencoder (VAE) achieves the MSE value of 0.075, while providing a speedup of over four orders of magnitude. The sample results generated using this approach are presented in Fig. 5(a). In fact, this is the least complex modal of all evaluated in this paper and therefore both qualitative and quantitative results it achieves are relatively good. Furthermore, we are able to improve the performance of the VAE by substituting the fully connected layers with the convolutional ones, as it is done in the convolutional Variational Autoencoder (cVAE). With this modification, we can reduce the MSE error to approximately 0.0266. However, because of a significant increase in model complexity, it is also considerably slower than the VAE. The observed speedup with respect to the GEANT3 solution reaches only one order of magnitude. As shown in Fig. 5(b), the qualitative results generated by the convolutional VAE appear to be more random, although their resolution is higher than in the case of VAE.

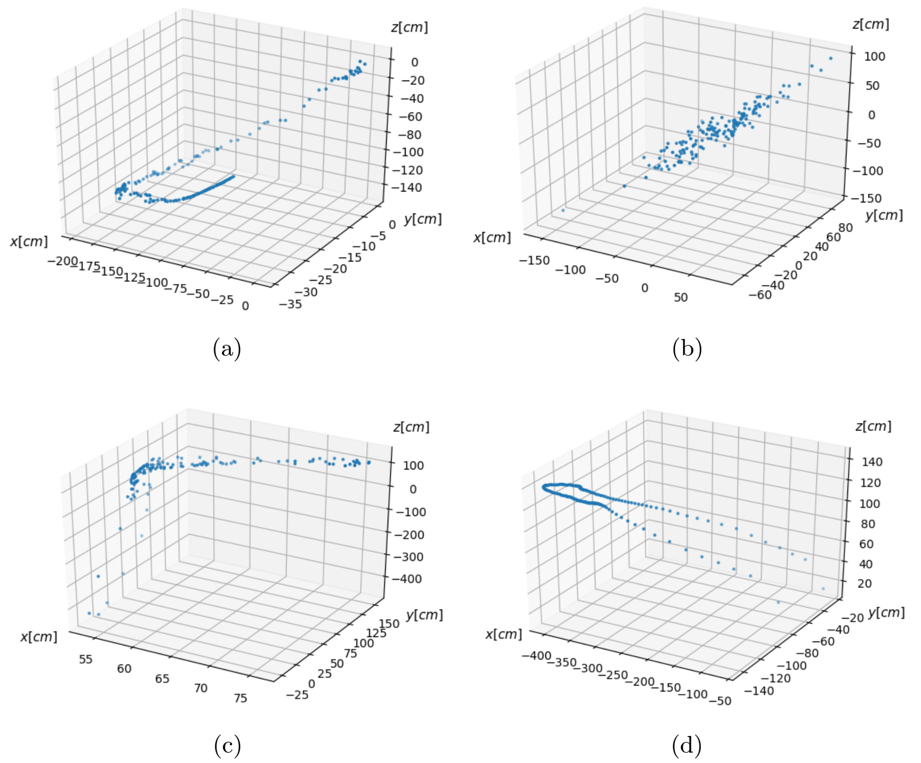


Fig. 5. Exemplar results generated by the **(a)** Variational Autoencoder (VAE), **(b)** convolutional Variational Autoencoder (cVAE),**(c)** the Deep Convolutional GAN (DC-GAN) and **(d)** the progressive DCGAN (proGAN) .

As far as the models based on the Generative Adversarial Networks are concerned, the results obtained by the Multilayer Perceptron (MLP) method indicate that this approach, as well as the recurrent network method based on the LSTM units, do not achieve sufficient accuracy. Although their speedup over the baseline simulation method reaches four orders of magnitude, the quantitative and qualitative results are only slightly better than those obtained by a random generator.

However, when analysing the results obtained by the Deep Convolutional GAN (DCGAN) model, we can see that extending neural network architectures with convolutional layers significantly reduces the MSE value, while still providing a reasonable speedup over the baseline that reaches two orders of magnitude. We explain that performance gain through the fact that convolutional layers can better simulate the working conditions of 3D simulation, as well as allow for deeper and more powerful models. The MSE value that is achieved by the evaluated DCGAN architecture is approximately 0.052 and is comparable to the

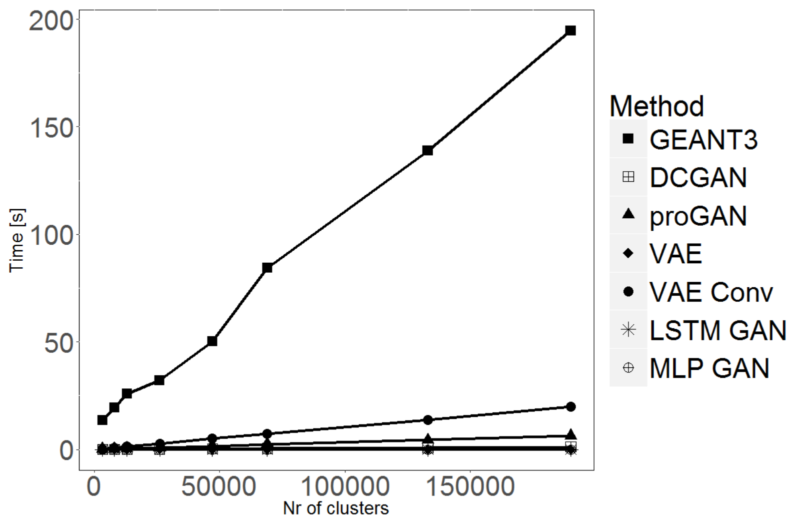


Fig. 6. Comparison of performance between GEANT 3 simulation and machine learning generative models.

results produced with VAE and cVAE architectures. The qualitative results can be seen in Fig. 5(c).

Extending the DCGAN model using a progressive training method (proGAN) allows us to improve the quality of the results and reach an unprecedented MSE value of 0.00176. The qualitative results generated by the proGAN also indicate that this method is a promising path of research, especially when additional conditioning on the particle type is added to the model. Although the computational complexity of the proGAN model is higher than the competing approaches, it still offers a massive 30-fold speedup over the baseline GEANT3 simulation method.

To emphasize the true potential of the simulation methods based on the generative models, we computed the execution times for different methods while increasing the number of simulated clusters, i.e. model outputs. Fig. 6 shows the results of this experiment. Although the computational cost of all the methods increases linearly with the number of simulated cluster responses, the improvement achieved by the generative models is massive and grows with increasing number of clusters. It is worth mentioning that the presented results are obtained for a single-core CPU computation, while additional hardware-based speedup of another order of magnitude was observed when using the GPU-based implementation for the neural network methods. Although Monte Carlo simulations can also benefit from such an acceleration, its iterative character does not allow the same improvement as in the case of deep neural networks.

6 Future work

In this work, we give an overview and preliminary evaluation results of possible machine learning generative methods that can be used for cluster simulation. We intend to investigate these approaches further and implement a fully working solution which can be deployed in production, *i.e.* as a part of the ALICE Experiment at LHC.

To that end, we plan to extend the described models by conditioning our model on the initial characteristics of the collision. More precisely, we envision extending the input to the model with initial particles momenta generated by PYTHIA or other currently used simulation tool. This shall reduce the variance of the model and increase the quality of the observed results.

On top of replacing the complex methods for clusters simulation that are currently in use, we want to employ the generative models as a part of the quality assurance tool. We draw our inspiration from the work of [30] where the Generative Adversarial Networks are used to discover anomalies in medical imaging data. We plan to design a similar solution for the high-energy physics experiments to discover faulty outputs of the particle detectors in a semi-real-time manner. To that end, we plan to use the efficient and highly effective generative models to simulate multiple healthy responses of the detector and compare the registered parameters of the detector with the simulated ones. If the synthetically generated responses do not agree with the real conditions, the model shall label the condition as an anomaly and alert the appropriate services.

7 Conclusion

In this work, we demonstrated the potential of machine learning generative methods namely Generative Adversarial Networks and Variational Autoencoders, for cluster simulation in the high-energy physics experiments. We proved the possible application of those methods using the example dataset generated for the TPC detector in the ALICE Experiment at LHC. We evaluated several architectures that are based on generative models, such as Variational Autoencoders and Generative Adversarial Networks, and compared their results in terms of quality and computational cost. We also adapted a progressive training technique to enhance the quality of generated samples. Although the quality of the best performing method is not yet comparable to the quality of currently used simulation methods, the computational speedup of the proposed method is unprecedented and reaches up to 10^4 when compared to the currently employed GEANT 3 simulation technique.

Acknowledgements

The authors acknowledge the support from the Polish National Science Centre grant no. UMO-2016/21/D/ST6/01946. The GPUs used in this work were funded by the grant of the Dean of the Faculty of Electronics and Information Technology at Warsaw University of Technology (project II/2017/GD/1).

References

1. L. Evans and P. Bryant, “LHC Machine,” *JINST*, vol. 3, p. S08001, 2008.
2. R. Brun, F. Bruyant, F. Carminati, S. Giani, M. Maire, A. McPherson, G. Patrick, and L. Urban, “GEANT Detector Description and Simulation Tool,” *CERN-W5013, CERN-W-5013, W5013, W-5013*, 1994.
3. S. Agostinelli *et al.*, “GEANT4: A Simulation toolkit,” *Nucl. Instrum. Meth.*, vol. A506, pp. 250–303, 2003.
4. A. Ferrari, P. R. Sala, A. Fasso, and J. Ranft, “FLUKA: A multi-particle transport code,” *CERN-2005-010, SLAC-R-773, INFN-TC-05-11*, 2005.
5. D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *CoRR*, vol. abs/1312.6114, 2013.
6. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *NIPS*, 2014.
7. G. Dellacasa *et al.*, “ALICE: Technical Design Report of the Time Projection Chamber,” *CERN-OPEN-2000-183, CERN-LHCC-2000-001*, 2000.
8. B. Abelev *et al.*, “Upgrade of the ALICE Experiment: Letter Of Intent,” *J. Phys.*, vol. G41, p. 087001, 2014.
9. K. Aamodt *et al.*, “The ALICE experiment at the CERN LHC,” *JINST*, vol. 3, p. S08002, 2008.
10. A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” *CoRR*, vol. abs/1511.06434, 2015.
11. S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
12. T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of gans for improved quality, stability, and variation,” *CoRR*, vol. abs/1710.10196, 2017.
13. S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, “Generative adversarial text to image synthesis,” *CoRR*, vol. abs/1605.05396, 2016.
14. X. Yan, J. Yang, K. Sohn, and H. Lee, “Attribute2image: Conditional image generation from visual attributes,” in *ECCV*, 2016.
15. A. Kadurin, A. Aliper, A. Kazennov, P. Mamoshina, Q. Vanhaelen, K. Khrabrov, and A. Zhavoronkov, “The cornucopia of meaningful leads: Applying deep adversarial autoencoders for new molecule development in oncology,” *Oncotarget*, vol. 8, no. 7, p. 10883, 2017.
16. M. Paganini, L. de Oliveira, and B. Nachman, “Calogan: Simulating 3d high energy particle showers in multi-layer electromagnetic calorimeters with generative adversarial networks,” *CoRR*, vol. abs/1705.02355, 2017.
17. L. de Oliveira, M. Paganini, and B. Nachman, “Learning particle physics by example: location-aware generative adversarial networks for physics synthesis,” *Computing and Software for Big Science*, vol. 1, p. 4, 2017.
18. G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
19. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *JMLR*, vol. 15, no. 1, pp. 1929–1958, 2014.
20. X. Glorot, A. Bordes, and Y. Bengio, “Deep sparse rectifier neural networks,” in *International Conference on Artificial Intelligence and Statistics*, pp. 315–323, 2011.
21. S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *ICML*, 2015.

22. D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *CoRR*, vol. abs/1412.6980, 2014.
23. X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *International Conference on Artificial Intelligence and Statistics*, pp. 249–256, 2010.
24. F. Chollet *et al.*, “Keras.” https://github.com/fchollet/keras, 2015.
25. M. Abadi *et al.*, “Tensorflow: A system for large-scale machine learning,” in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pp. 265–283, 2016.
26. M. Lucic, K. Kurach, M. Michalski, S. Gelly, and O. Bousquet, “Are gans created equal? a large-scale study,” *CoRR*, vol. abs/1711.10337, 2017.
27. T. Sjostrand, S. Mrenna, and P. Z. Skands, “PYTHIA 6.4 Physics and Manual,” *JHEP*, vol. 05, p. 026, 2006.
28. P. Z. Skands, “Tuning Monte Carlo Generators: The Perugia Tunes,” *Phys. Rev.*, vol. D82, p. 074018, 2010.
29. A. A. D. P. Suaide, C. A. G. Prado, T. Alt, L. Aphecetche, N. Agrawal, A. Avasthi, M. Bach, R. Bala, G. Barnafoldi, A. Bhasin, *et al.*, “O2: A novel combined online and offline computing system for the alice experiment after 2018,” in *Journal of Physics: Conference Series*, vol. 513, p. 012037, IOP Publishing, 2014.
30. T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, “Unsupervised anomaly detection with generative adversarial networks to guide marker discovery,” in *International Conference on Information Processing in Medical Imaging*, pp. 146–157, Springer, 2017.