Simulation of Electron-Proton Scattering Events by a Feature-Augmented and Transformed Generative Adversarial Network (FAT-GAN)

Yasir Alanazi[1] , **Nobuo Sato**[2] , **Tianbo Liu**[2] , **Wally Melnitchouk**[2] , **Pawel Ambrozewicz**[2] , **Florian Hauenstein**[3] , **Michelle P. Kuchera**[4] , **Evan Pritchard**[4] , **Michael Robertson**[5] , **Ryan Strauss**[5] , **Luisa Velasco**[6] and **Yaohang Li**[1]

[1]Department of Computer Science, Old Dominion University, Norfolk, Virginia 23529
[2]Jefferson Lab, Newport News, Virginia 23606
[3]Department of Physics, Old Dominion University, Norfolk, Virginia 23529
[4]Department of Physics, Davidson College, Davidson, North Carolina 28035
[5]Department of Mathematics and Computer Science, Davidson College, Davidson, North Carolina 28035
[6]Department of Physics, University of Dallas, Irving, Texas 75062
yalan001@odu.edu, {nsato, liutb, wmelnitc, pawel, hauenst}@jlab.org, {mikuchera, evpritchard, mirobertson, rystrauss}@davidson.edu, lvelasco@udallas.edu, yaohang@cs.odu.edu

Abstract

We apply generative adversarial network (GAN) technology to build an event generator that simulates particle production in electron-proton scattering that is free of theoretical assumptions about underlying particle dynamics. The difficulty of efficiently training a GAN event simulator lies in learning the complicated patterns of the distributions of the particles physical properties. We develop a GAN that selects a set of transformed features from particle momenta that can be generated easily by the generator, and uses these to produce a set of augmented features that improve the sensitivity of the discriminator. The new Feature-Augmented and Transformed GAN (FAT-GAN) is able to faithfully reproduce the distribution of final state electron momenta in inclusive electron scattering, without the need for input derived from domain-based theoretical assumptions. The developed technology can play a significant role in boosting the science of existing and future accelerator facilities, such as the Electron-Ion Collider.

1 Introduction

High-energy scattering reactions typically produce collections of particles in the final state whose momentum distributions are governed by fundamental femtometer-scale physics. One of the major goals of existing lepton-hadron scattering facilities, such as COMPASS at CERN and Jefferson Lab, as well as the future Electron-Ion Collider, is to determine the three-dimensional distributions of the hadrons' elementary quark and gluon constituent from measurements of particle production cross sections. Unfortunately, since quarks and gluons are not directly detectable experimentally, their properties must be inferred indirectly from the observed particle spectra within the theoretical framework of factorization in Quantum Chromodynamics (QCD) [Collins *et al.*, 1989].

Since the early 1970s, Monte Carlo event generators (MCEGs) have played a vital role in facilitating studies of high-energy scattering reactions. From the experimental perspective, MCEGs are crucial for understanding the complex arrays of detectors that measure the energies and momenta of final state particles. The construction of existing MCEGs, such as Pythia [Sjostrand *et al.*, 2008], Herwig [Bahr and others, 2008] or Sherpa [Gleisberg *et al.*, 2009], has been driven by a combination of high-precision data from previous experiments and inputs from theory. The latter have involved a mix of perturbative QCD methods, describing the dynamics of quarks and gluons at short distances, and phenomenological models that map the transition from quark and gluon degrees of freedom to the observable hadrons ("hadronization"). An MCEG can in principle be viewed as a form of a "data compatification tool", encapsulating enormous amounts of data collected from multiple experiments, which can be regenerated from the MCEG itself. On the other hand, the reliance of existing MCEGs on theoretical assumptions of factorization and on hadronization models limits their ability to capture the full range of possible correlations between produced particles' momenta and spins.

In this work we suggest a new strategy for constructing MCEGs using modern machine learning methods involving GANs [Goodfellow *et al.*, 2014] that can learn to generate particles in specific reactions, such as electron-proton scattering, without recourse to theoretical assumptions about femtometer-scale physics. A particular feature of GANs is their ability to generate synthetic data, such as images, by learning from real samples without knowledge of the underlying laws of the original system. The aim is to implicitly learn the primordial distributions over data which are difficult to model with an explicit parametrization for the underlying law. We propose to explore this aspect by treating the events characterized by final state particle momenta in high-energy reactions as the "images".

Typically, a GAN model is composed of a generator and a discriminator. The generator transforms random white noise through a deep neural network to produce candidate samples from the target distribution, while the discriminator learns through another deep neural network to differentiate the true samples from those produced by the generator. The GAN evolves as the generator and discriminator compete adversarially, alternatively updating their parameters during the training process. Eventually, the GAN is able to approximate the underlying cumulative distribution function (CDF) and inverse CDF transformation, and thus sample the target distribution given sufficiently large neural networks, sample size, and long enough computation time.

Although GANs have demonstrated impressive results in generating near-realistic images [Karras *et al.*, 2018], music [Mogren, 2016], and videos [Clark *et al.*, 2019], training a successful GAN model is known to be notoriously difficult. Many GAN models suffer from major problems including mode collapse, non-convergence, model parameter oscillation, destabilization, vanishing gradient, and overfitting due to unbalanced generator/discriminator combinations. Approaches and techniques to address these general problems have been proposed and discussed in a number of publications [Salimans *et al.*, 2016; Arora and Zhang, 2017; Arjovsky and Bottou, 2017].

Using GANs to simulate events from particle reactions poses additional challenges for machine learning. Unlike many GAN applications, such as generating realistic and sharp looking images, where the distribution agreement between the GAN-generated samples and the true ones is often not strictly enforced, GANs for generating physical events are required to model the distributions of event features and their correlations sufficiently precisely for the nature of particle reactions to be faithfully replicated. Furthermore, events generated by GANs should not violate basic physical laws, such as baryon number, charge and momentum conservation. In order for the GAN event generator to work, a careful choice of architecture, representations, features, parameter initialization, and selection of hyper-parameters is required.

In this paper we describe the development of a GAN event generator to simulate particles in inclusive electron-proton scattering. As a proof of concept, we restrict ourselves to building a GAN that only learns how to generate specific kinds of particles in the final state, while ignoring other particles. We shall refer to such a generator as an "inclusive" generator, to distinguish it from an "exclusive" generator that generates the full spectrum of particles in the final sate. For testing and validation purposes we first use the existing theory-based Pythia8 MCEG [Sjostrand *et al.*, 2008] to generate the training samples to train GAN. Then we train and validate GAN on data from real-life electron scattering experiment.

Our analysis shows that the difficulty of successfully training a GAN event generator lies in the complicated patterns of distributions of the physical properties of the generated particles. To improve the GAN training, we develop a Feature-Augmented and Transformed GAN (FAT-GAN), using a set of transformed features from the particle physical properties as the generated features of the generator. These transformed features represent the underlying degrees of freedom of the particles, which can simplify the learning of the generator, while avoiding generation of unphysical events. A set of augmented features is further derived from the generated features to improve the sensitivity of the discriminator. We also compare the efficiency of using Cartesian coordinates versus spherical coordinates to represent generated features in the FAT-GAN. With well-selected coordinate systems and features, our GAN model is able to regenerate particle momenta so that it mimics all the relevant momentum correlations as observed in the the original MCEG.

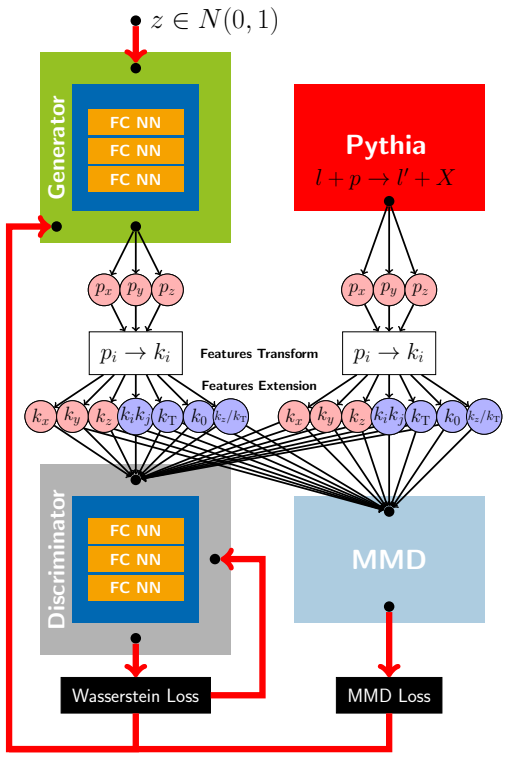

Figure 1: Architecture of the inclusive FAT-GAN event generator (see text for details).

2 Related Work

In the literature, GANs have been used in a variety of applications at the Large Hadron Collider (LHC), such as simulating energy deposits of final state particles [Paganini *et al.*, 2018a; Paganini *et al.*, 2018b] and jets [de Oliveira *et al.*, 2017; Musella and Pandolfi, 2018], accelerating importance sampling Monte Carlo integration [Bendavid, 2017], simulating data collections of beam studies [Erdmann *et al.*, 2019], and reconstructing cosmic ray-induced air showers [Erdmann *et al.*, 2018]. These efforts have focused on image representations of the aggregated experimental data as opposed to individual event-based samples. GANs have also been used to invert the detector effects such as in [Datta *et al.*, 2018] and [Bellagente *et al.*, 2020]. A crucial question about whether the events generated by GANs can add statistical precision beyond the training samples was investigated in [Butter *et al.*,

2020].

Recently there have been several attempts to investigate the possibility of directly training GANs at the event level in proton-proton collisions, such as those at the LHC. [Hashemi *et al.*, 2019] applied a GAN to produce muon four-momenta in $Z \to \mu^+\mu^-$ events generated by Pythia. Although agreeing well in the reduced dataset, their GAN model fails to reproduce certain feature distributions. [Otten *et al.*, 2019] reported a less satisfactory performance when a GAN was based on fully-connected deep networks in the study of the two-body decay processes $e^+e^- \to Z \to l^+l^-$ and $pp \to t\bar{t}$. [Butter *et al.*, 2019] applied a GAN to simulate the $2 \to 6$ particle production process $pp \to t\bar{t} \to (bq\bar{q}')(\bar{b}\bar{q}q')$, incorporating maximum mean discrepancy (MMD) [Gretton *et al.*, 2007] to resolve sharp local features. [Di Sipio *et al.*, 2020] implemented a dijet-GAN based on convolutional neural networks (CNNs) to simulate the production of pairs of jets at the LHC.

Although each of the above GANs was designed to simulate events in different reactions, they all face similar challenges of learning the unique feature distributions of the events, which often exhibit sharp edges, spikes, multiple peaks, and large variations. Building a generator capable of generating these distributions and a discriminator sensitive to their features are the keys to the success of GAN-based MCEGs.

3 Methods

3.1 Data Descriptions

We use GANs to mimic two samples of inclusive electron-proton scattering events: the first is generated from the Pythia MCEG [Sjostrand *et al.*, 2008] at a center-of-mass energy of 100 GeV, and the second is experimental data from CLAS at Jefferson Lab with beam energy of 5.5 GeV. In our initial analysis, we train the GAN only on the scattered electron momenta. Events are represented as an array of the electron four-momenta $p_\mu = (E; \boldsymbol{p})$, where in Cartesian coordinates the three-momentum is given by $\boldsymbol{p} = (p_x, p_y, p_z)$, and the energy is given by $E = \sqrt{p_x^2 + p_y^2 + p_z^2 + m^2}$, where m is the electron mass. Throughout this paper we will work in units of GeV for all momentum and energy variables.

3.2 GAN Architecture

The generator produces both generated features and augmented features. The three generated features describe the three degrees of freedom of the scattered electron (components of the momentum $\boldsymbol{p}$), while the augmented features, such as energy E, transverse momentum $p_T = \sqrt{p_x^2 + p_y^2}$, and the longitudinal to transverse momentum ratio p_z/p_T, can be calculated from the generated features to represent other physical characteristics of the particle. The augmented features are used to improve the sensitivity of the discriminator.

The input to the generator "G" is a 100-dimensional white noise array centered at 0 with unit standard deviation. The generator network consists of 5 hidden dense layers, each with 512 neurons, activated by a leaky Rectified Linear Unit (ReLU) function. The last hidden layer is fully connected to a three-neuron output, activated by a linear function representing the generated features. A customized Lambda layer is then incorporated to calculate the augmented features from the generated features. Inspired by the idea of importance sampling [Ji and Li, 2016], the augmented features are carefully selected to improve the sensitivity of the discriminator in distinguishing the GAN-generated events from the Pythia input. These augmented features, together with the generated features from the generator, are concatenated and fed to the discriminator as input.

As for the generator, the neural network in the discriminator "D" also consists of 5 hidden dense layers, each with 512 neurons, activated by a leaky ReLU function. To avoid overfitting in classification, a 10% dropout rate is applied to each hidden layer. The last hidden layer is fully connected to a single-neuron output, activated by a linear function, where "1" indicates a true event and "0" is a fake event. The overall architecture of the inclusive GAN event generator is illustrated in Fig. 1.

3.3 Loss Functions

The discriminator D is trained to give $D(\boldsymbol{p}) = 1$ for each sample $\boldsymbol{p}$ generated by Pythia, and $D(\widetilde{\boldsymbol{p}}) = 0$ for each sample $\widetilde{\boldsymbol{p}}$ produced by the generator. The discriminator is optimized against the Wasserstein loss with gradient penalty [Gulrajani *et al.*, 2017] to improve training stability and reduce the likeliness of mode collapse. The loss function L_D of the discriminator is defined as

$$\begin{aligned} L_D = &\big(\mathbb{E}[D(\widetilde{\boldsymbol{p}})] - \mathbb{E}[D(\boldsymbol{p})]\big) \\ &+ \lambda\, \mathbb{E}_{\widehat{\boldsymbol{p}} \sim P_{\widehat{\boldsymbol{p}}}}\big[(\|\nabla_{\widehat{\boldsymbol{p}}} D(\widehat{\boldsymbol{p}})\|_2 - 1)^2\big], \end{aligned} \tag{1}$$

where $\mathbb{E}$ denotes the expectation value. The first term in Eq. (1) measures the Wasserstein distance [M. Arjovsky and Bottou, 2017]. The second term is the gradient penalty, where $\widehat{\boldsymbol{p}}$ is a random sample from $P_{\widehat{\boldsymbol{p}}}$, defined by a uniform distribution along the straight lines between pairs of samples from the true Pythia event data and the generator's output. The coefficient λ is a harmonic parameter to balance the Wasserstein distance and the gradient penalty.

To ensure that the distributions of the event features created by the generator match with those of Pythia, we incorporate a two-sample test based on kernel MMD in our inclusive GAN event generator. To compare two distributions, the MMD employs a kernel-based statistical test method to determine if the two samples are drawn from different distributions. As a result, the loss function L_G of the generator G includes a Wasserstein distance term from the discriminator D and an MMD term [Li *et al.*, 2017],

$$L_G = -\mathbb{E}[D(\widetilde{\boldsymbol{p}})] + \eta\, \mathrm{MMD}^2(\boldsymbol{p}, \widetilde{\boldsymbol{p}}), \tag{2}$$

where η is the balancing hyperparameter. The MMD term is defined as

$$\begin{aligned} \mathrm{MMD}^2(\boldsymbol{p}, \widetilde{\boldsymbol{p}}) = &\ \mathbb{E}_{\boldsymbol{p}_a, \boldsymbol{p}_{a'} \sim P_{\boldsymbol{p}}}[k(\boldsymbol{p}_a, \boldsymbol{p}_{a'})] \\ &+ \mathbb{E}_{\boldsymbol{p}_b, \boldsymbol{p}_{b'} \sim P_{\widetilde{\boldsymbol{p}}}}[k(\boldsymbol{p}_b, \boldsymbol{p}_{b'})] \\ &- 2\, \mathbb{E}_{\boldsymbol{p}_a \sim P_{\boldsymbol{p}}, \boldsymbol{p}_b \sim P_{\widetilde{\boldsymbol{p}}}}[k(\boldsymbol{p}_a, \boldsymbol{p}_b)], \end{aligned} \tag{3}$$

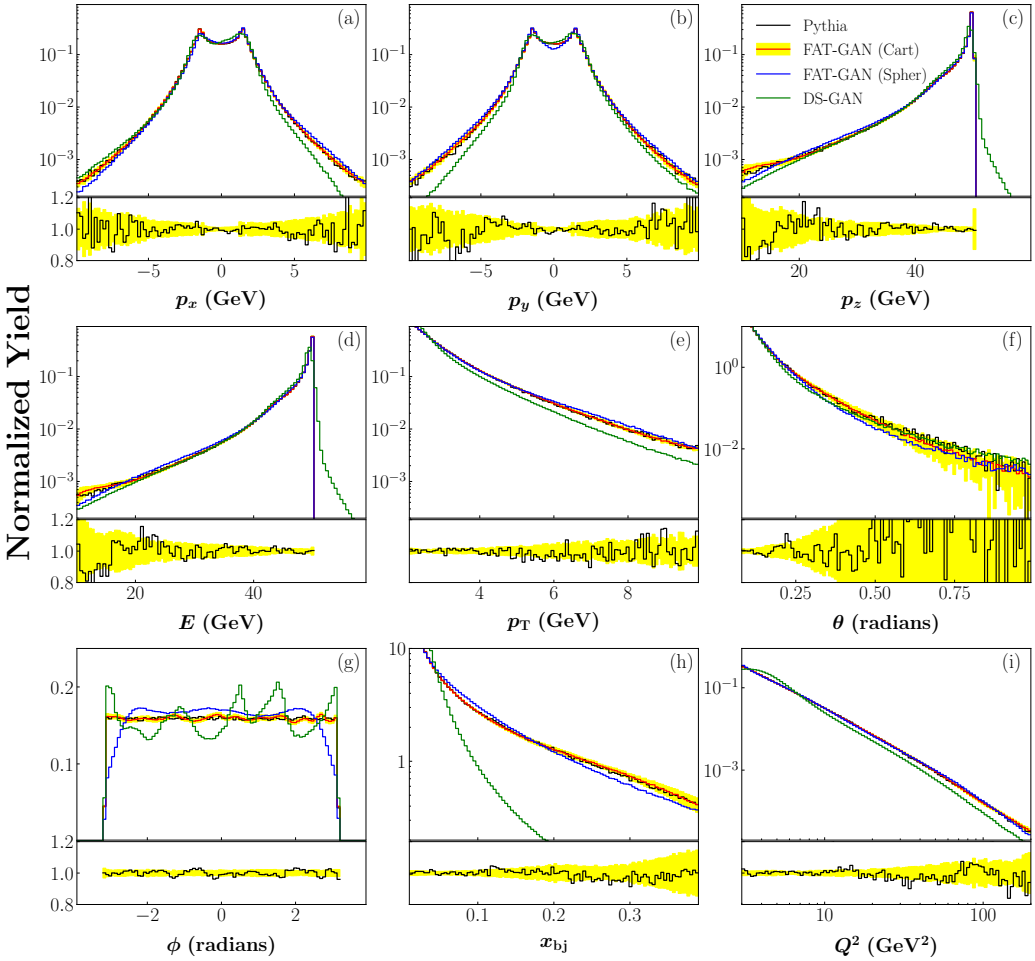

Figure 2: Distributions of physical properties of the scattered electron, p_x, p_y, p_z, E, p_T, θ, ϕ, $x_{\rm bj}$ and Q^2 (see text), generated by Pythia (black lines), FAT-GAN (Cart) (red lines and yellow bands), FAT-GAN (Spher) (blue lines), and DS-GAN (green lines). The ratio of the FAT-GAN (Cart) to Pythia yields is shown at the bottom of each panel.

where $k(\boldsymbol{p}_a, \boldsymbol{p}_b)$ is a positive definite kernel function. Here we select a Gaussian kernel such that $k(\boldsymbol{p}_a, \boldsymbol{p}_b) = \exp[-(\boldsymbol{p}_a - \boldsymbol{p}_b)^2/2\sigma^2]$, where σ is the hyperparameter determining the MMD resolution, tuned to the same order of magnitude as the width of the event features.

The combined network is trained adversarially for $200,000$ epochs. In each epoch, the combined network passes through the complete training dataset once. A large batch of $10,000$ events is employed to ensure that there are sufficient samples to calculate a stable MMD value in each batch. Each batch contains random examples from the Pythia event dataset. The optimizer is Adam [Kingma and Ba, 2014] with a 10^{-4} learning rate, $\beta_1 = 0.5$, and $\beta_2 = 0.9$. To balance the generator and discriminator training, the training ratio is set to 5.

3.4 Feature Representations

The physical observables characterizing the scattered electron properties are illustrated in Fig. 2, including the momentum and energy components of the electron four-vector p_μ [Fig. 2(a)–(e)], scattering angles [Fig. 2(f)–(g)], and derived quantities $x_{\rm bj}$ and Q^2 (see Sec. 4.2 below). The energy and momentum distributions generated by Pythia exhibit rather large variations, with the ratio between the most populated regions and those with rare events reaching up to $\sim 10^4$.

More seriously, a sharp edge in the E distribution arises from energy conservation, which restricts E to be less than the incident beam energy, $E_{\rm b}$. This sharp edge is very difficult for the inclusive GAN to learn, as unphysical events can be generated with $E > E_{\rm b}$, which the discriminator is not sensitive enough to differentiate from the eligible physical events, particularly when $E_{\rm b} - E$ is small. The sharp edge in the E distribution also leads to sharp edges in the p_z and θ distributions. The difficulty of learning sharp edge distributions has also been reported in [Hashemi *et al.*, 2019].

To address the problem of learning sharp edge distributions, we transform the momentum properties to specific generated features that allow their distributions to be generated more easily, while avoiding production of unphysical

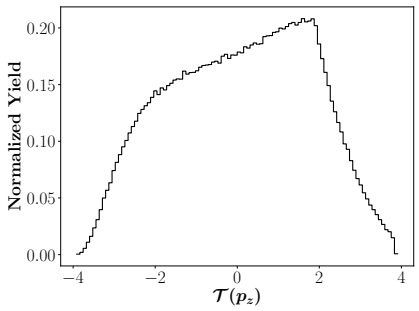

Figure 3: Distribution of the transformed feature $\mathcal{T}(p_z)$.

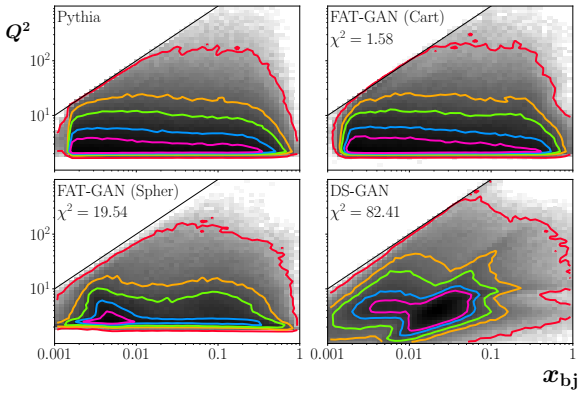

Figure 4: Joint distributions of Q^2 (in GeV^2) and $x_{\rm bj}$ for FAT-GAN (Cart), FAT-GAN (Spher) and DS-GAN compared to Pythia. The χ^2 values per bin are indicated in the GAN-generated panels.

particles. For the p_z distribution, instead of directly using p_z as a generated feature, we use the transformed variable $\mathcal{T}(p_z) = \log[(E_{\rm b} - p_z)/(1\ {\rm GeV})]$ in the generator. The original distribution with a sharp edge is now converted to a distribution that is more like a Gaussian, with significantly reduced variation, as illustrated in Fig. 3.

Although $\mathcal{T}(p_z)$ does not have actual physical meaning, the transformation ensures that the GAN will not generate events with unphysical p_z values. As well as making it easier for the generator to produce, the transformed distribution $\mathcal{T}(p_z)$ improves the sensitivity of the discriminator as a classifier. As a result, the generator learns to generate $(p_x, p_y, \mathcal{T}(p_z))$, and $p_z = \exp[(E_{\rm b} - \mathcal{T}(p_z))/(1\ {\rm GeV})]$ is later calculated by the customized Lambda layer as one of the augmented features.

In addition to p_z, the variables p_T, E, and p_z/p_T are also calculated in the customized Lambda layer as augmented features in order to improve the sensitivity of the discriminator. The generated features and augmented features are concatenated as the input to the discriminator.

4 Results

In this section we compare the event feature distributions from the FAT-GAN and the Direct-Simulation GAN ("DS-GAN") that directly simulates the momenta (p_x, p_y, p_z). We also compare the efficiency of the FAT-GAN implementation in Cartesian coordinates ("FAT-GAN (Cart)") with the FAT-GAN in spherical coordinates ("FAT-GAN (Spher)"). The latter is an alternative representation to describe a particle in terms of the variables (E, θ, ϕ), where $\theta = \arctan(p_z/p_T)$ is the polar angle between p_z and the transverse plane, and $\phi = \arctan(p_y/p_x)$ is the azimuthal angle. The FAT-GAN (Spher) generates $(\mathcal{T}(E), \mathcal{T}(\theta), \phi)$ in spherical coordinates as the generated features, and then p_x, p_y, p_z, p_T, E, and p_z/p_T as the augmented features. The features $\mathcal{T}(E)$ and $\mathcal{T}(\theta)$ are converted from E and θ using a logarithmic transformation similar to that applied to p_z to remove the sharp edges in the distribution.

4.1 Feature Distributions

The event feature distributions from the DS-GAN, FAT-GAN (Cart), and FAT-GAN (Spher) are compared in Fig. 2 with those generated from Pythia. In general the DS-GAN results do not match as well with Pythia compared to FAT-GAN (Cart) and FAT-GAN (Spher). Moreover, the tails beyond the sharp edges in Fig. 2(c) and (d) indicate that some unphysical events are generated with $E > 50$ GeV. The FAT-GAN (Cart) yields match better with Pythia than do the FAT-GAN (Spher) yields, particularly for the ϕ distribution. The four momentum components (E; p_x, p_y, p_z), as well as p_T, θ and ϕ, are also well reproduced relative to Pythia, and have a minimal number of unphysical events.

Compared to the reaction events simulated in [Butter *et al.*, 2019] and [Di Sipio *et al.*, 2020], where the typical ratio between the peak and tail events is up to 10, the ratio in our scattering electron features can be up to 10^4. Nevertheless, even for the rare events that are 10^{-3} of the number of the peak events, the FAT-GAN (Cart) agrees well with Pythia in their distributions, including the symmetry of ϕ shown in Fig. 2.

4.2 Inter-correlations Between Event Features

In addition to the electron momentum and energy, we examine two additional physical quantities that are typically used to characterize electron scattering, namely the squared four-momentum of the exchanged virtual photon Q^2 and the Bjorken scaling variable $x_{\rm bj}$, neither of which are explicitly generated as features in the FAT-GAN. In terms of the beam and scattered electron momenta and energies, the photon virtuality can be written as $-Q^2 \equiv q \cdot q = q_0^2 - \boldsymbol{q}^2$, where $q_0 = E_{\rm b} - E$ and $\boldsymbol{q} = (-p_x, -p_y, \sqrt{E_{\rm b}^2 - m^2} - p_z)$ are the energy and three-momentum transfer, respectively. The Bjorken variable is defined by the dimensionless ratio

$$x_{\rm bj} = \frac{Q^2}{2P \cdot q}, \tag{4}$$

and kinematically ranges from 0 to 1. As shown in Fig. 2(h) and (i), the $x_{\rm bj}$ and Q^2 distributions from the DS-GAN deviate significantly from those generated by Pythia. In contrast, the FAT-GAN (Cart) yields match better than those from the DS-GAN and the FAT-GAN (Spher).

The Q^2–$x_{\rm bj}$ joint distributions, shown in Fig. 4, indicate that the FAT-GAN (Cart) gives a good match with the Pythia Q^2-$x_{\rm bj}$ joint distribution, as indicated by the contour lines, with a χ^2 value per bin of 1.58. In contrast, somewhat

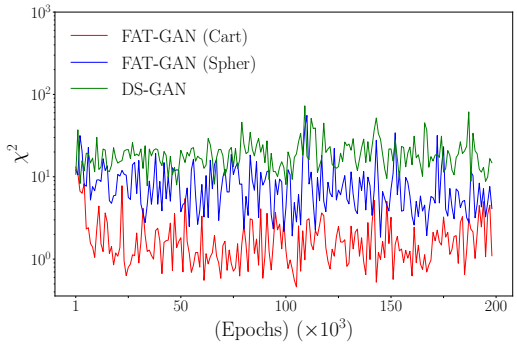

Figure 5: Comparison of χ^2 values for $x_{\rm bj}$ distributions of Pythia events and those generated by the FAT-GAN (Cart), FAT-GAN (Spher) and DS-GAN, with respect to the number of training epochs.

worse agreement ($\chi^2 = 19.54$) is observed for the FAT-GAN (Spher), and very poor agreement ($\chi^2 = 82.41$) for the DS-GAN. This indicates that the FAT-GAN model in Cartesian coordinates not only learns the four-momentum vector accurately, but also their inter-correlations.

4.3 Comparison of Coordinate Representations

The Cartesian representation and the spherical representation of a particle are equivalent physically. However, the Cartesian and spherical representations exhibit different learning efficiencies in the FAT-GAN. As shown in Figs. 2 and 4, the FAT-GAN (Cart) demonstrates better agreement in the distributions of physical properties than the FAT-GAN (Spher). In Fig. 5 the convergence of the FAT-GAN (Cart), FAT-GAN (Spher) and DS-GAN is compared, as measured by the χ^2 values for the $x_{\rm bj}$ distributions of Pythia and GAN-generated events along training epochs. Although both FAT-GANs yield better convergence than the DS-GAN, the FAT-GAN (Cart) demonstrates better efficiency and generally lower χ^2 values that are close to 1 than the FAT-GAN (Spher).

Overall, the FAT-GAN (Spher) is found to have a degraded performance compared to the FAT-GAN (Cart). The main reason is that the physical properties in spherical coordinates are less favorable than those in Cartesian coordinates for training the FAT-GAN. In particular, both the distributions of E and θ in spherical coordinates exhibit sharp edges (see Fig. 2(d) and (f)). Moreover, the ϕ distribution has shape boundaries at both ends, which poses additional complications for the GAN to learn.

4.4 FAT-GAN on Experimental Data

Having studied the performance of the FAT-GAN on synthetic Pythia data, we now compare the results of the FAT-GAN (Cart) simulations with inclusive scattering data for a 5.5 GeV electron beam impinging on a liquid hydrogen target in CLAS at Jefferson Lab [Mecking and others, 2003]. A clean event sample of 100k reconstructed electron events was used for the comparison with the trained FAT-GAN, illustrated in Fig. 6, which shows that the distributions of the events generated by FAT-GAN match well with those of the experimental data. Note, however, that in contrast to the smooth distributions from the Pythia MCEG in Fig. 2, the CLAS data here have

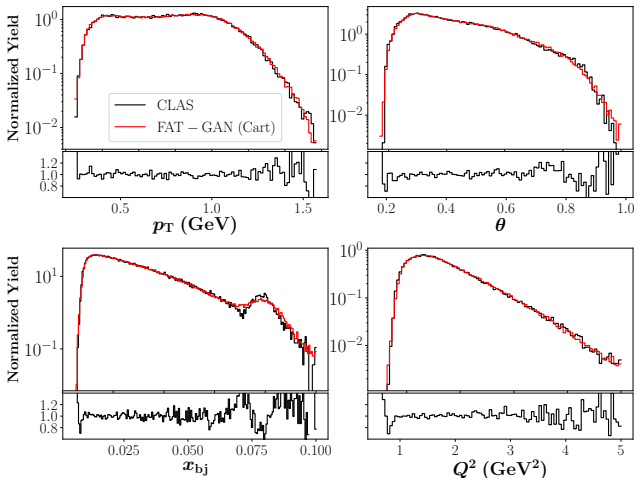

Figure 6: Comparison of the distributions of the physical properties p_T, θ, $x_{\rm bj}$ and Q^2 generated by FAT-GAN (Cart) with those from CLAS at Jefferson Lab (uncorrected for detector effects). The ratio of FAT-GAN (Cart) to CLAS data is shown at the bottom of each panel.

not been corrected for detector effects, and therefore display artifacts such as the shoulder in the $x_{\rm bj}$ distribution. Nevertheless, the comparison in Fig. 6 indicates that the physical properties p_T, θ, $x_{\rm bj}$, and Q^2, which are not directly used in the training, are correctly captured by the FAT-GAN trained on the experimental event data.

5 Conclusion

We have investigated the use of GANs to simulate electrons in the final state of high-energy inclusive electron-proton collisions. We report that selecting the appropriate features as generated features or augmented features plays a critical role in building a successful GAN event generator. While the physical properties of the particles often exhibit challenging distribution patterns for a GAN to learn, using transformed features enables the generator to more easily generate and avoid unphysical events. Augmenting in addition the feature space for the discriminator to become more sensitive, our FAT-GAN demonstrates good agreement with simulated as well as experimental data in mimicking event feature distributions and their inter-correlations. We also find that the selection of the appropriate coordinate representations impacts the GAN performance. Although the FAT-GANs presented in this paper are specific to electron-proton scattering, the feature selections and transformation strategy can be generalized to GANs for simulating other reactions under different conditions, as well as learning exclusive events.

The FAT-GAN package is available at https://github.com/ijcai2021/FAT-GAN.

Acknowledgements

We thank the CLAS Collaboration for providing the electron scattering data used in Fig. 4, and J. Qiu for helpful discussions. This work was supported by the LDRD project No. LDRD19-13, No. LDRD20-18, and No. LDRD21-22.

References

[Arjovsky and Bottou, 2017] M. Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. *ArXiv*, 1701.04862, 2017.

[Arora and Zhang, 2017] S. Arora and Y. Zhang. Do gans actually learn the distribution? an empirical study. *ArXiv*, 1706.08224, 2017.

[Bahr and others, 2008] M. Bahr et al. Herwig++ Physics and Manual. *Eur. Phys. J.*, C58:639–707, 2008.

[Bellagente *et al.*, 2020] M. Bellagente, A. Butter, G. Kasieczka, T. Plehn, and R. Winterhalder. How to gan away detector effects. *ArXiv*, 1912.00477, 2020.

[Bendavid, 2017] J. Bendavid. Efficient monte carlo integration using boosted decision trees and generative deep neural networks. *ArXiv*, 1707.00028, 2017.

[Butter *et al.*, 2019] A. Butter, T. Plehn, and R. Winterhalder. How to gan lhc events. *ArXiv*, 1907.03764, 2019.

[Butter *et al.*, 2020] A. Butter, S. Diefenbacher, G. Kasieczka, B. Nachman, and T. Plehn. Ganplifying event samples. *ArXiv*, 2008.06545, 2020.

[Clark *et al.*, 2019] A. Clark, J. Donahue, and K. Simonyan. Adversarial video generation on complex datasets. *ArXiv*, 1907.06571, 2019.

[Collins *et al.*, 1989] J. C. Collins, D. E. Soper, and G. F. Sterman. Factorization of Hard Processes in QCD. *Adv. Ser. Direct. High Energy Phys.*, 5:1–91, 1989.

[Datta *et al.*, 2018] K. Datta, D. Kar, and D. Roy. Unfolding with generative adversarial networks. *ArXiv*, 1806.00433, 2018.

[de Oliveira *et al.*, 2017] L. de Oliveira, M. Paganini, and B. Nachman. Learning particle physics by example: Location-aware generative adversarial networks for physics synthesis. *ArXiv*, 1701.05927, 2017.

[Di Sipio *et al.*, 2020] R. Di Sipio, G. M. Faucci, H. S. Ketabchi, and S. Palazzo. DijetGAN: A Generative-Adversarial Network Approach for the Simulation of QCD Dijet Events at the LHC. *ArXiv*, 08:110, 2020.

[Erdmann *et al.*, 2018] M. Erdmann, J. Glombitza, and D. Walz. A deep learning-based reconstruction of cosmic ray-induced air showers. *Astro. Phys.*, 97:46–53, 2018.

[Erdmann *et al.*, 2019] M. Erdmann, J. Glombitza, and T. Quast. Precise simulation of electromagnetic calorimeter showers using a wasserstein generative adversarial network. *ArXiv*, 1807.01954, 2019.

[Gleisberg *et al.*, 2009] T. Gleisberg, S. Hoeche, F. Krauss, M. Schonherr, S. Schumann, F. Siegert, and J. Winter. Event generation with SHERPA 1.1. *JHEP*, 02:007, 2009.

[Goodfellow *et al.*, 2014] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *ArXiv*, 1406.2661, 2014.

[Gretton *et al.*, 2007] A. Gretton, K. Borgwardt, M. J. Rasch, B. Scholkopf, and A. J. Smola. A kernel method for the two-sample problem. *ArXiv*, 0805.2368, 2007.

[Gulrajani *et al.*, 2017] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of wasserstein gans. *ArXiv*, 1704.00028, 2017.

[Hashemi *et al.*, 2019] B. Hashemi, N. Amin, K. Datta, D. Olivito, and M. Pierini. Lhc analysis-specific datasets with generative adversarial networks. *ArXiv*, 1901.05282, 2019.

[Ji and Li, 2016] H. Ji and Y. Li. *Monte Carlo Methods and Their Applications in Big Data Analysis*, pages 125–139. 2016.

[Karras *et al.*, 2018] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. *ArXiv*, 1812.04948, 2018.

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

[Li *et al.*, 2017] C. Li, W. Chang, Y. Cheng, Y. Yang, and B. Póczos. Mmd gan: Towards deeper understanding of moment matching network. *ArXiv*, 1705.08584, 2017.

[M. Arjovsky and Bottou, 2017] S. Chintala M. Arjovsky and L. Bottou. Wasserstein gan. *ArXiv*, 1701.07875, 2017.

[Mecking and others, 2003] B. A. Mecking et al. The CEBAF Large Acceptance Spectrometer (CLAS). *Nucl. Instrum. Meth. A*, 503:513–553, 2003.

[Mogren, 2016] O. Mogren. C-rnn-gan: Continuous recurrent neural networks with adversarial training. *ArXiv*, 1611.09904, 2016.

[Musella and Pandolfi, 2018] P. Musella and F. Pandolfi. Fast and accurate simulation of particle detectors using generative adversarial networks. *ArXiv*, 1805.00850, 2018.

[Otten *et al.*, 2019] S. Otten, S. Caron, W. de Swart, M. van Beekveld, L. Hendriks, C. van Leeuwen, D. Podareanu, R. R. de Austri, and R. Verheyen. Event generation and statistical sampling for physics with deep generative models and a density information buffer. *ArXiv*, 1901.00875, 2019.

[Paganini *et al.*, 2018a] M. Paganini, L. de Oliveira, and B. Nachman. Accelerating science with generative adversarial networks: An application to 3d particle showers in multilayer calorimeters. *ArXiv*, 1705.02355, 2018.

[Paganini *et al.*, 2018b] M. Paganini, L. de Oliveira, and B. Nachman. Calogan Simulating 3d high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks. *ArXiv*, 97(1), 2018.

[Salimans *et al.*, 2016] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. *ArXiv*, 1606.03498, 2016.

[Sjostrand *et al.*, 2008] T. Sjostrand, S. Mrenna, and P. Z. Skands. A Brief Introduction to PYTHIA 8.1. *Comput. Phys. Commun.*, 178:852–867, 2008.