Lund jet images from generative and cycle-consistent adversarial networks

Stefano Carrazza[1] **and Frédéric A. Dreyer**[2]
[1] TIF Lab, Dipartimento di Fisica, Università degli Studi di Milano and INFN Milan
[2] Rudolf Peierls Centre for Theoretical Physics, University of Oxford

Abstract

We introduce a generative model to simulate radiation patterns within a jet using the Lund jet plane. We show that using an appropriate neural network architecture with a stochastic generation of images, it is possible to construct a generative model which retrieves the underlying two-dimensional distribution to within a few percent. We compare our model with several alternative state-of-the-art generative techniques. Finally, we show how a mapping can be created between different categories of jets, and use this method to retroactively change simulation settings on an existing sample. These results provide a framework for significantly reducing simulation times through fast inference of the neural network as well as for data augmentation of physical measurements.

1 Introduction

One of the most common objects emerging from hadron collisions at particle colliders such as the Large Hadron Collider (LHC) are jets. These are loosely interpreted as collimated bunches of energetic particles emerging from the interactions of quarks and gluons, the fundamental constituents of the proton [1, 2], and usually defined through a sequential recombination algorithm [3–5].

In recent years, there has been considerable interest in applications of generative adversarial networks (GAN) [6] and variational autoencoders (VAE) [7] to particle physics, where such generative models can be used to significantly reduce the computing ressources required to simulate realistic LHC data [8–15]. In these proceedings and its companion paper [16], we introduce a generative model to create new samples of the substructure of a jet from existing data. We use the Lund jet plane [17] as a visual representation of the clustering history of a jet, which provides an efficient encoding of radiation patterns and can be directly measured experimentally [18]. The Lund jet image is used to train a Least Square GAN (LSGAN) [19] to reproduce simulated data within a few percent accuracy. Finally, we show how a cycle-consistent adversarial network (CycleGAN) [20] can be used to create mappings between different categories of jets. We apply this framework to retroactively change the parameters of the parton shower on an event, adding non-perturbative effects to an existing sample.

2 Generating jets

To describe the radiation patterns of a jet, we will use the primary Lund plane representation [17], which can be projected onto a two-dimensional image that serves as input to a neural network.

The Lund jet plane is constructed by reclustering a jet's constituents with the Cambridge-Aachen (C/A) algorithm [4, 21]. This algorithm sequentially recombines pairs of particles that have the minimal $\Delta_{ij}^2 = (y_i - y_j)^2 + (\phi_i - \phi_j)^2$ value, where y_i and ϕ_i are the rapidity and azimuth of particle i.

This clustering sequence can be used to construct an $n \times n$ pixel image describing the radiation patterns of the initial jet. We iterate in reverse through the clustering sequence, labelling the momenta

Second Workshop on Machine Learning and the Physical Sciences (NeurIPS 2019), Vancouver, Canada.

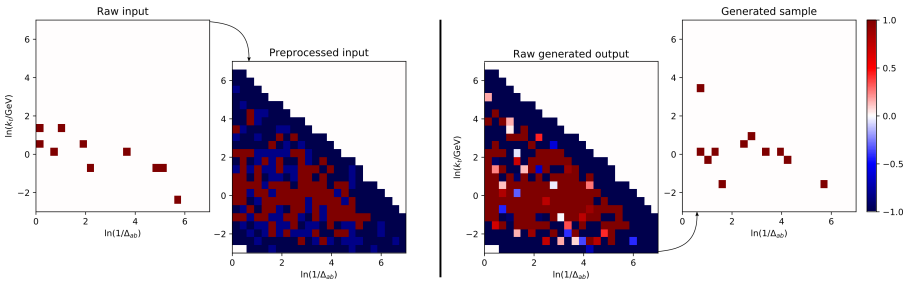


Figure 1: Left two figures: Sample input images before and after preprocessing. Right two: sample generated by the LSGAN and the corresponding Lund image.

of the two branches of a declustering as p_a and p_b, ordered in transverse momentum such that $p_{t,a} > p_{t,b}$. This procedure follows the harder branch a and at each step we activate the pixel on the image corresponding to the coordinates $(\ln \Delta_{ab}, \ln k_t)$, where $k_t = p_{t,b}\Delta_{ab}$ is the transverse momentum of particle b relative to $p_a + p_b$.

The data sample used in this article consists of 500k jets, generated using the dijet process in Pythia v8.223, and clustered using the anti-k_t algorithm [5, 22] with radius $R = 1.0$. Unless specified otherwise, results use the Delphes v3.4.1 fast detector simulation, with the `delphes_card_CMS_NoFastJet.tcl` card to simulate both detector effects and particle flow reconstruction. The simulated jets are then converted to Lund images with 24×24 pixels each, where a pixel is set to one if there is a corresponding $(\ln \Delta_{ab}, \ln k_t)$ value in the primary declustering sequence, otherwise it is left at zero.

We will train a neural network to create new samples from this reference data using generative adversarial networks [23]. They are constructed using both a generator G and discriminator D, which are competing against each other through a value function $V(G, D)$

$$\min_G \max_D V(D, G) = \mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]\,, \tag{1}$$

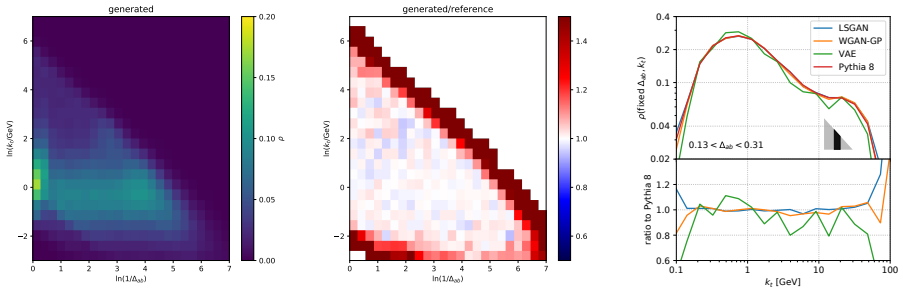
where we defined $p_z(z)$ as a prior on input noise variables. Thus D is trained in order to maximise the probability of correctly distinguishing the training examples and the samples from G, while the latter is trained to minimise $\log(1 - D(G(z)))$. The generator's distribution p_g optimises equation (1) when $p_g = p_{\text{data}}$, so that the generator learns how to generate new samples from z.

In practice, we found improved performance when using a Least Square Generative Adversarial Network (LSGAN) [19], a specific class of GAN which uses a least squares loss function for the discriminator. The main advantage of the LSGAN over the original GAN framework is a more stable training process, due to an absence of vanishing gradients. In addition, we include a minibatch discrimination layer [24] to avoid a collapse of the generator.

The LSGAN is trained on the full sample of QCD Lund jet images. In order to overcome the limitation of GANs due to the sparse and discrete nature of Lund images, we first re-sample our initial data set into batches of n_{avg} and create a new set of 500k images, each consisting of the average of n_{avg} initial input images, which are reinterpreted as physical events through a random sampling, where the pixel value is interpreted as the probability that the pixel is activated. A further data preprocessing step before training the LSGAN consists in rescaling the pixel intensities to be in the $[-1, 1]$ range, and masking entries outside of the kinematic limit of the Lund plane. The images are then whitened using zero-phase components analysis (ZCA) whitening [25]. The optimal choice of hyperparameters is determined using the distributed asynchronous hyperparameter optimisation library `hyperopt` [26].

In figure 1 the first two images illustrate an example of input image before and after preprocessing while the last two images represent the raw output from the LSGAN model and the corresponding Lund image.

The final averaged results for the Lund jet plane density is shown in figure 2a. The ratio of the generated sample and the reference data is shown in 2b, where we can observe a good agreement between the reference and the artificial sample generated by the gLund model. The model is able to reproduce the underlying distribution to within a 3-5% accuracy in the bulk region of the Lund image. Larger discrepancies are visible at the boundaries of the Lund image and are due the vanishing



(a) Generated (b) Ratio generated/reference (c) Δ_{ab} slice

Figure 2: (a) Average Lund jet plane density for a data set generated by the gLund model and (b) ratio between the generated and reference sample. (c) Slice along k_t with $0.13 < \Delta_{ab} < 0.31$

pixel intensities. In practice this model provides a new approach to reduce Monte Carlo simulation time for jet substructure applications as well as a framework for data augmentation. Finally, in figure 2c we show the distribution obtained for a slice of fixed Δ_{ab} size, cutting along the Lund jet plane vertically. The reference Pythia 8 sample is shown in red, with alternative models based on a variational autoencoder [7, 27, 28] and a Wasserstein GAN [29, 30] shown in green and orange respectively. The lower panel gives the ratio of the different models to the reference Pythia 8 curve, showing very good performance for the LSGAN and WGAN-GP models, which are able to reproduce the data within a few percent.

3 Reinterpreting events using image-to-image translations

Let us now introduce a novel application of domain mappings to reinterpret existing event samples. To this end, we implement a cycle-consistent adversarial network (CycleGAN) [20], which is an unsupervised learning approach to create translations between images from a source domain to a target domain.

Using as input Lund images generated through different processes or generator settings, one can use this technique to create mappings between different types of jet. As an example, we will consider a mapping from parton-level to detector-level images.

The cycle obtained for a CycleGAN trained on parton and detector-level images is shown in figure 3a, where an initial parton-level Lund image is transformed to a detector-level one, before being reverted again. The sampled image is shown in the bottom row.

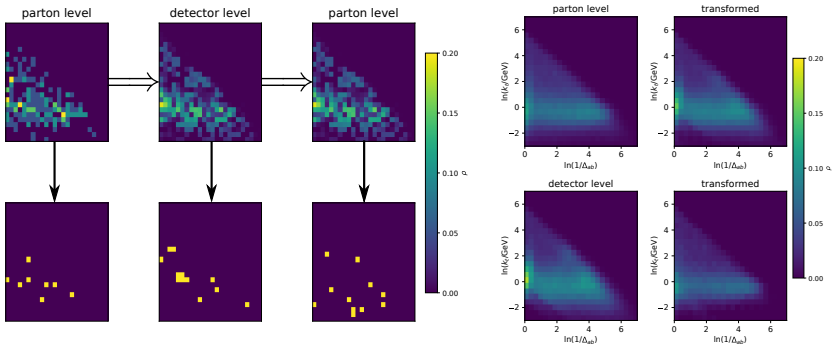
A CycleGAN learns mapping functions between two domains X and Y, using as input training samples from both domains. It creates an unpaired image-to-image translation by learning both a mapping $G : X \to Y$ and an inverse mapping $F : Y \to X$ which observes a forward cycle consistency $x \in X \to G(x) \to F(G(x)) \approx x$ as well as a backward cycle consistency $y \in Y \to F(y) \to G(F(y)) \approx y$. This behaviour is achieved through a cycle consistency loss

$$\mathcal{L}_{\text{cyc}}(G, F) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1] \,, \tag{2}$$

Additionally, the full objective $\mathcal{L}$ includes also adversarial losses to both mapping functions such that G is incentivized to generate images $G(x)$ that resemble images from Y, while the discriminator D_Y attempts to distinguish between translated and original samples. Thus, CycleGAN aims to find arguments solving $G^*, F^* = \arg\min_{G,F} \max_{D_X, D_Y} \mathcal{L}$.

We implemented a CycleGAN framework, labelled CycleJet, that can be used to create mappings between two domains of Lund images. By training a network on parton and detector-level images, this method can thus be used to retroactively add non-perturbative and detector effects to existing parton-level samples. Similarly, one can train a model using images generated through two different underlying processes, allowing for a mapping e.g. from QCD jets to W or top initiated jets.

In the first row of figure 3b we show results for an initial average parton-level sample before (left) and after (right) applying the parton-to-detector mapping encoded by the CycleJet model, while in



(a) Translation cycle (b) Transformed samples

Figure 3: (a) Top: transition from parton-level to delphes-level and back using CycleJet, bottom: corresponding sampled event. (b) Top: average of the parton-level sample before (left) and after (right) applying the parton-to-detector, bottom: delphes-level sample before and after applying the detector-to-parton.

the second row we perform the inverse operation by taking as input the dephes-level sample before (left) and after (right) applying the CycleJet detector-to-parton mapping. This example shows clearly the possibility to add non perturbative and detector effects to a parton level simulation within good accuracy. Similar mappings can be created with good accuracy e.g. to transform a sample of QCD jets into a W jet sample.

This method allows for the possibility to use such an approach to save CPU time for applying full detector simulations and non perturbative effects to parton level events. It is also possible to train the CycleJet model on Monte Carlo data and apply the corresponding mapping to real data.

4 Conclusions

We have conducted a careful study of generative models applied to jet substructure.

First, we trained a LSGAN model to generate new artificial samples of detector level Lund jet images. With this, we observed agreement to within a few percent accuracy with respect to the reference data. This new approach provides an efficient method for fast simulation of jet radiation patterns without requiring the long runtime of full Monte Carlo event generators. Another advantage consists in the possibility of this method to be applied to real collider data to generate accurate physical samples, as well as making it possible to avoid the necessity for large storage space by generating realistic samples on-the-fly.

Secondly, a CycleGAN model was constructed to map different jet configurations, allowing for the conversion of existing events. This procedure can be used to change Monte Carlo parameters such as the underlying process or the shower parameters. As examples we show how to add non perturbative and detector effects to a parton level simulation. As for the LSGAN, this method can be used to save CPU time by including full detector simulations and non perturbative effects to parton level events. Additionally, one could use CycleJet to transform real data using mappings trained on Monte Carlo samples or apply them to samples generated through gLund.

The full code and the pretrained models presented in this paper are available in [31, 32].

Acknowledgments: F.D. is supported by the Science and Technology Facilities Council (STFC) under grant ST/P000770/1. S.C. is supported by the European Research Council under the European Union's Horizon 2020 research and innovation Programme (grant agreement number 740006).

References

[1] George F. Sterman and Steven Weinberg. Jets from Quantum Chromodynamics. *Phys. Rev. Lett.*, 39:1436, 1977.

[2] Gavin P. Salam. Towards Jetography. *Eur. Phys. J.*, C67:637–686, 2010.

[3] Stephen D. Ellis and Davison E. Soper. Successive combination jet algorithm for hadron collisions. *Phys. Rev.*, D48:3160–3166, 1993.

[4] Yuri L. Dokshitzer, G. D. Leder, S. Moretti, and B. R. Webber. Better jet clustering algorithms. *JHEP*, 08:001, 1997.

[5] Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. The anti-k_t jet clustering algorithm. *JHEP*, 04:063, 2008.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[7] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[8] Luke de Oliveira, Michela Paganini, and Benjamin Nachman. Learning Particle Physics by Example: Location-Aware Generative Adversarial Networks for Physics Synthesis. *Comput. Softw. Big Sci.*, 1(1):4, 2017.

[9] Michela Paganini, Luke de Oliveira, and Benjamin Nachman. Accelerating Science with Generative Adversarial Networks: An Application to 3D Particle Showers in Multilayer Calorimeters. *Phys. Rev. Lett.*, 120(4):042003, 2018.

[10] Michela Paganini, Luke de Oliveira, and Benjamin Nachman. CaloGAN : Simulating 3D high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks. *Phys. Rev.*, D97(1):014021, 2018.

[11] 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. 2019.

[12] Pasquale Musella and Francesco Pandolfi. Fast and Accurate Simulation of Particle Detectors Using Generative Adversarial Networks. *Comput. Softw. Big Sci.*, 2(1):8, 2018.

[13] Kaustuv Datta, Deepak Kar, and Debarati Roy. Unfolding with Generative Adversarial Networks. 2018.

[14] Riccardo Di Sipio, Michele Faucci Giannelli, Sana Ketabchi Haghighat, and Serena Palazzo. DijetGAN: A Generative-Adversarial Network Approach for the Simulation of QCD Dijet Events at the LHC. 2019.

[15] Anja Butter, Tilman Plehn, and Ramon Winterhalder. How to GAN LHC Events. 2019.

[16] Stefano Carrazza and Frédéric A. Dreyer. Lund jet images from generative and cycle-consistent adversarial networks. 2019.

[17] Frédéric A. Dreyer, Gavin P. Salam, and Grégory Soyez. The Lund Jet Plane. *JHEP*, 12:064, 2018.

[18] The ATLAS collaboration. Measurement of the Lund Jet Plane using charged particles with the ATLAS detector from 13 TeV proton–proton collisions. 2019.

[19] Xudong Mao, Qing Li, Haoran Xie, Raymond Y. K. Lau, and Zhen Wang. Multi-class generative adversarial networks with the L2 loss function. *CoRR*, abs/1611.04076, 2016.

[20] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networkss. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.

[21] M. Wobisch and T. Wengler. Hadronization corrections to jet cross-sections in deep inelastic scattering. In *Monte Carlo generators for HERA physics. Proceedings, Workshop, Hamburg, Germany, 1998-1999*, pages 270–279, 1998.

[22] Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. FastJet User Manual. *Eur. Phys. J.*, C72:1896, 2012.

[23] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 2672–2680. MIT Press, Cambridge, MA, USA, 2014.

[24] Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *CoRR*, abs/1606.03498, 2016.

[25] Anthony J. Bell and Terrence J. Sejnowski. The "independent components" of natural scenes are edge filters. *Vision Research*, 37(23):3327 – 3338, 1997.

[26] J. Bergstra, D. Yamins, and D. D. Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, pages I–115–I–123. JMLR.org, 2013.

[27] Irina Higgins, Loïc Matthey, Xavier Glorot, Arka Pal, Benigno Uria, Charles Blundell, Shakir Mohamed, and Alexander Lerchner. Early visual concept learning with unsupervised deep learning. *CoRR*, abs/1606.05579, 2016.

[28] Christopher P. Burgess, Irina Higgins, Arka Pal, Loïc Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β-vae. *CoRR*, abs/1804.03599, 2018.

[29] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 214–223, 2017.

[30] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein gans. *CoRR*, abs/1704.00028, 2017.

[31] Frédéric Dreyer and Stefano Carrazza. Jetsgame/glund v1.0.0, September 2019.

[32] Frédéric Dreyer and Stefano Carrazza. Jetsgame/cyclejet v1.0.0, September 2019.