Normalizing Flows for Fragmentation and Hadronization

Ahmed Youssef
Dept. of Physics
University of Cincinnati
Cincinnati, Oh 45221
youssead@ucmail.uc.edu

Phil Ilten
Dept. of Physics
University of Cincinnati
Cincinnati, Oh 45221
philten@cern.ch

Tony Menzo
Dept. of Physics
University of Cincinnati
Cincinnati, Oh 45221
menzoad@mail.uc.edu

Stephen Mrenna
Fermilab
Batavia, IL 60510
mrenna@fnal.gov

Manuel Szewc
Dept. of Physics
University of Cincinnati
Cincinnati, Oh 45221
manuelsz24@gmail.com

Michael K. Wilkinson
Dept. of Physics
University of Cincinnati
Cincinnati, Oh 45221
michael.k.wilkinson@gmail.com

Jure Zupan
Dept. of Physics
University of Cincinnati
Cincinnati, Oh 45221
zupanje@ucmail.uc.edu

Abstract

Hadronization is an important step in Monte Carlo event generators, where quarks and gluons are bound into physically observable hadrons. Previous work has demonstrated first steps towards a machine-learning (ML) based simulation of the hadronization process. However, the presented architectures are limited to producing only pions as hadron emissions. In this work we use normalizing flows to overcome this limitation. We use masked autoregressive flows as a generator for the kinematic distributions in the hadronization pipeline. We condition normalizing flows (NFs) on different hadron masses and initial configuration energies, which allows for the emission of hadrons with arbitrary masses. The NF generated kinematic distributions match the PYTHIA generated ones well. In this paper we present our preliminary results.

1 Introduction

Particle collisions in collider experiments are simulated with Monte Carlo event generators, which can be factorized into three steps: (i) generation of the hard process (the particle collision); (ii) parton shower (the evolution from high energy to low energy, in which quarks and gluon are created); and (iii) hadronization (the combination of quarks and gluons to observable particles, hadrons). While the first two steps are perturbative in their nature, and thus under good theoretical control with significant efforts devoted to improving the precision even further [1–4], the hadronization step is inherently non-perturbative, i.e. it does not follow first principle and thus can not be described theoretically. Therefore, phenomenological models with many free parameters are typically used instead, which oftentimes do not cover the entire underlying physics well.
The two main models used in simulating hadronization are the Lund string model [5–7], where partons are connected via QCD color strings with a linear potential, which are iteratively split into

Machine Learning and the Physical Sciences workshop, NeurIPS 2022.

hadrons, and the cluster model [8–10], where partons are pre-confined into proto-clusters, which then decayed into hadrons via sequential two-body decays. Both models have limitations. The string model requires over $\mathcal{O}(20)$ parameters to describe the hadronization, and has some challenges describing baryon production. The cluster model has fewer parameters, but the decays of large clusters can lead to phenomenological problems such as predicting heavy baryon distributions which do not match data well. Widely used event generators which simulate the above steps are PYTHIA [11], HERWIG [12], and SHERPA [13].

In this project we propose a machine-learning (ML) based simulation for hadronization, aiming to replace or complement the common used phenomological models used in event generators. Generative Adversial Networks (GANs) [14], Variational Auto-Encoders (VAEs) [15–17] and Normalizing Flows (NFs) [18] have demonstrated the ability for ML to generate convincing physical observables. In addition, conditional generative models provide more flexibility and control of the output [19, 20]. Previous work demonstrated that GANs - used for the cluster model - [21] and different versions of VAE-like sliced-Wasserstein Auto Encoders (SWAEs) - used for the Lund string model - [22] have the potential to simulate parts of the hadronization process well. However, the presented models were limited to the emission of only pions as a final state particle, and the training of the kinematic distributions was performed separately with no correlation between the p_z and p_T distribution.

In this paper we present an updated version of MLHAD [22] utilizing NFs as a generator for the kinematic distribution, which overcomes these limitations. This new architecture is able to learn correlations between the kinematic distributions and is not limited to the emission of only pions, but rather can produce the emission of hadrons with arbitrary mass. We demonstrate these capabilities by training it on specially prepared PYTHIA hadronization outputs with an explicit infra-red (IR) cut-off. In sec. 4 we discuss possible extensions and planned future work.

2 Architecture and method

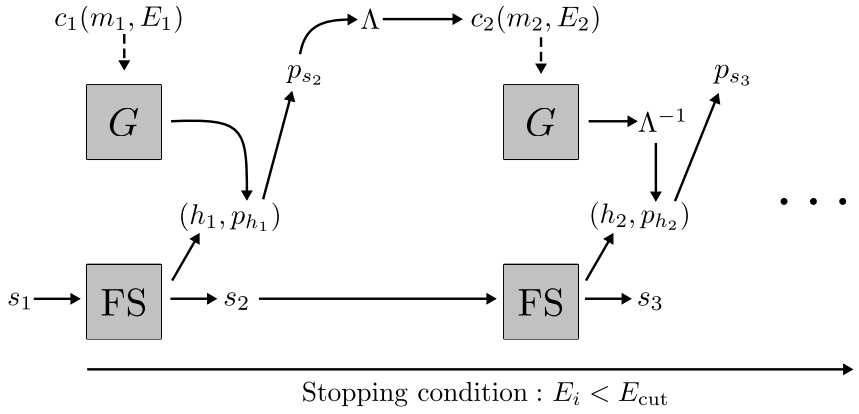

Figure 1: An illustration of using NF as a generator in the hadronization chains. G is the generator, which is in our case the NF. In principle any generative model, which generates the kinematics well can be used.

We consider a simplified Lund string model to demonstrate the performance of our ML architecture. Hadronization describes the combination of quarks and gluons into hadrons at the non-perturbative scale. In the Lund string model, the quarks are connected via a QCD color string/tube; in particular we are looking at a $q_i\bar{q}_i$ fragmentation event in the center-of-mass frame with quark-flavor index i and initial energy E. The string breaks iteratively and emits a hadron h per break. In each step the quark flavors and kinematics are conserved. Fig. 1 illustrates the architecture we are using to describe the Lund string model.

FS is the simplified PYTHIA flavor selector, which takes as input the initial string quark-flavor s_1 and returns the emitted hadron flavor h_1 and the new string-end flavor s_2. G is the generator, which samples the kinematics of the hadron and due to energy-momentum conservation, updates the four momentum of the new string end. Here we can use NFs as the kinematics generator G. This process repeats until the energy of the string falls below a predetermined cut-off value, E_{cut}, which is set to 5 GeV in our case. Before each hadron emission, the string fragments are boosted to their

center-of-mass frame using a Lorentz transformation Λ. The architecture is flexible such that we can exchange components easily. In this section we discuss conditional NFs as a generative model for the hadron kinematic.

2.1 NF as kinematic generator

NFs are ML architectures which perform a bijective transformation between two spaces, by first sampling a random vector u from a simple distribution (usually a Gaussian) $u \sim \pi(u)$. The sample is transformed by a neural network f to obtain a data point x from the desired and usually more complicated distribution p(x), where f is designed to be invertable. We can calculate the density of x by finding the corresponding random vector u by a change of variables:

$$p(x) = \pi(u)\left|\det\left(\frac{\partial f}{\partial u}\right)\right|^{-1}, \tag{1}$$

where $u = f^{-1}(x)$. To use NFs as a generative model, one starts with a sample u from the base distribution and uses eq. 1 to map to the target distribution. Autoregressive flows use a transformation $f(x_i)$, where x_i depends only on the previous coordinates $x_1, ..., x_{i-1}$, which leads to a triangular Jacobian matrix, and subsequently results in the more efficient computation of the determinant. To ensure the ability to learn high dimensional transformations with a complex target distribution, multiple layers of these bijective transformations are used; in our case we use five. One can choose different transformations as long as they are sufficiently expressive. In the results presented in this paper we use masked autoregressive flows (MAF) [23] with an affine [24] transformation. However, to decrease the sampling time one can use inverse autoregressive flows (IAF) [25].

2.2 Training Data

The training data is a set of PYTHIA generated first-hadron emissions for different initial string energies. We can reduce our process to a two variable problem by aligning the z axis of the coordinate system with the direction of the initial string and assuming axial symmetry in PYTHIA. We can then reconstruct the complete four momentum of a particle with the two momentum components p_z and p_T.

The NF is trained simultaneously on the p_z and p_T, which allows the NF to learn correlations between the two variables, such that the training data set contains samples from a 2D distribution of the transverse momentum and z component of the momentum $\mathbf{x}_i = \{p'^{(i)}_{z,k}, p^{(i)}_{T,k}\}$. In addition we condition in each step on the initial string energy and the hadron mass. This allows us to generate the kinematics for emissions with all possible hadron masses. The energy and mass conditions are transformed in the range $0 < c < 1$ as demonstrated in [22]. The loss function is the negative log likelihood of the learned probability distribution $p(x)$, which is minimized during the training process. We use the nflows package[1] [26] for the implementation and train for 5×10^6 iterations.

To demonstrate the performance, we train the NF on data for masses ranging from 0.1 GeV to 1 GeV and energies $E = 100, 400, 700, 1000$ GeV, such that the model can be provided with both a mass and energy label. This can be extended to larger mass ranges, 0.1 GeV to 20 GeV, over which one can cover any possible hadron emission.

3 Results

As one can see in fig. 11 the NF trained model is able to accurately reproduce first-emission kinematics of PYTHIA for a hadronized quark anti-quark system in the center-of-mass frame of the string. In addition, the NF is able to interpolate well for mass labels which were not included in the training, but fall between the training labels, see app. A.2. The NF is also able to extrapolate to mass labels outside the training range. In app. A.1 one can see the results for different labels we trained on and in app. A.2 are the results for mass labels that were not trained on, either interpolated or extrapolated. Fig. 3 shows the heat map of p_z and p_T for PYTHIA and NF generated kinematics, which demonstrates that the NFs are able to learn correlations between the two variables.

[1] Corresponding github repo and licence can be found in: `https://github.com/bayesiains/nflows`

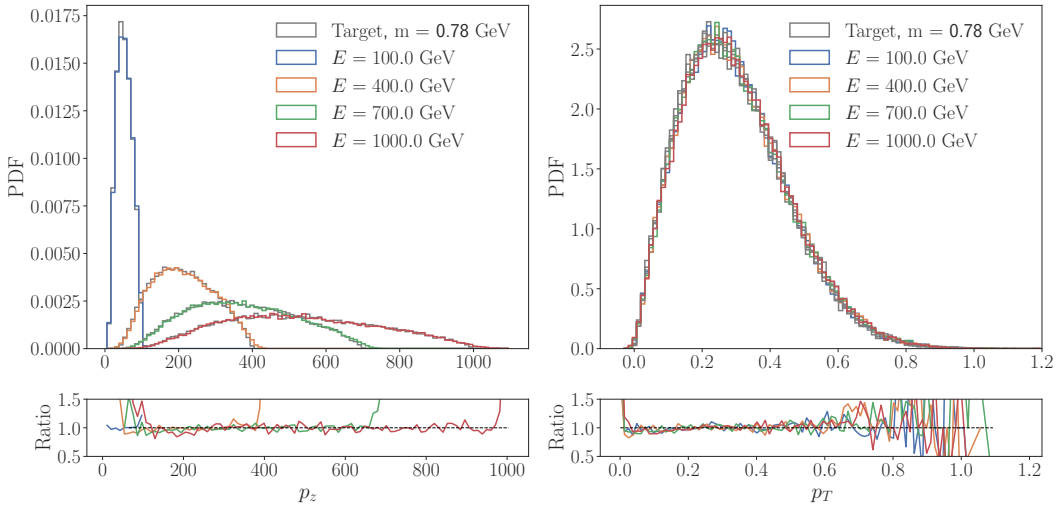

Figure 2: The NF generated p_z and p_T distributions for first-hadron emissions with string energies of (label 1) $E = 100, 400, 700, 1000$ GeV, and hadron masses of (label 2) $m = 0.78$ GeV, which the NF was trained with, compared to the PYTHIA generated target distribution (grey), as well as the ratios of the NF generated to PYTHIA generated distributions.

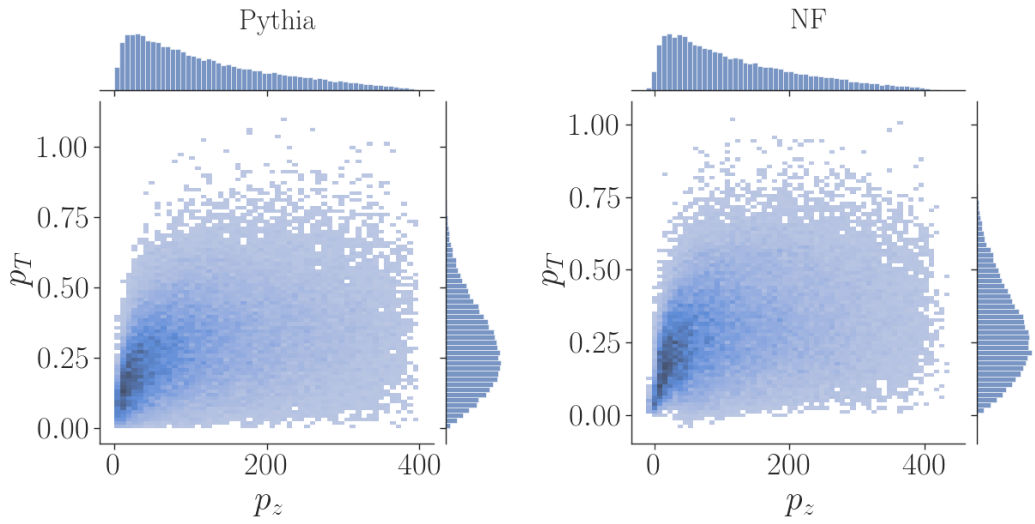

Figure 3: Heat map illustrating the correlation between p_z and p_T for the PYTHIA (left) and NF (right) generated kinematics.

The model can be trained on mass labels corresponding to hadron masses ranging from 0.1 to 20 GeV. With this range we can sample the kinematics for all physical hadrons that can be emitted since for the kinematics only the mass of the hadron is needed to distinguish them.

In fig. 4 we show the average number of hadron multiplicities as a function of initial string energy, using PYTHIA or MLHAD with NFs fragmentation chain. The updated version of MLHAD with NFs shows greater agreement than the previous version utilizing the cSWAE architecture presented in [22] (fig. 12).

4 Conclusion and Outlook

The updated version of MLHAD utilizing NF, appears to be well suited for modeling the non-perturbative process of hadronization. This was demonstrated by training an NF architecture on

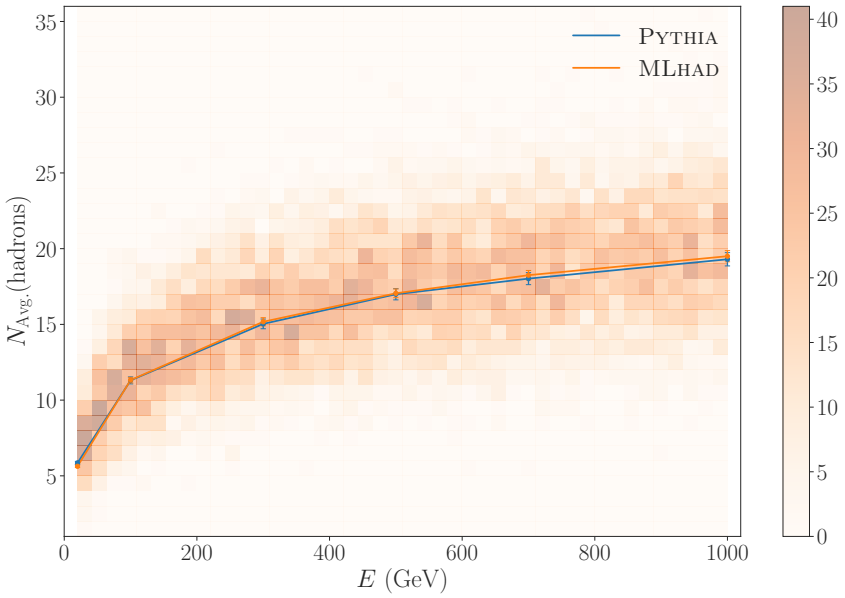

Figure 4: Average number of hadrons produced in the hadronization chain as function of initial string energy using MLHAD (orange) and PYTHIA (blue) for 2×10^4 fragmentation chains and only pions as emitted hadrons.

a simplified PYTHIA hadronization model, limited to string ends made out light quark flavors. It overcomes the limitation of allowing only pions as emitted hadrons, presented in [22, 21], and is able to simulate all possible hadron emissions. In addition, our NF is able to be trained on p_z and p_T simultaneously, such that possible correlations between these two variables can be learned.

Due the architecture of the hadronization pipeline, see fig. 1, the NF model can be easily implemented in the fragmentation chain. In fig 4 we generate a hadronization chain and compare the average hadron multiplicity of PYTHIA with the ML model. The updated MLHAD version utilizing NFs shows a better performance than the previous version. However, despite these encouraging results from masked autoregressive flows (MAF), we may switch to a different NF in the future due to the slow sampling speed of the MAF.
Another advantage of the presented architecture is that the exact probability distribution of the drawn samples is known. Given the sample, one can compute the exact likelihood that would be generated by the model. Future work will include uncertainty estimation of hadronization models.
Looking forward, the training of the ML architecture presented in this work will be performed on physically accessible observables. Possible candidates can be found here [27–32]. Many of these observable will be accessible due to open-data efforts from different collaborations. The presented architecture will be used in a pipeline to perform training on experimental data. The code for MLHAD can be found in `https://gitlab.com/uchep/mlhad`.

Acknowledgment

We thank Mike Williams for the suggestion of using normalizing flows. AY, JZ, and TM acknowledge support in part by the DOE grant de-sc0011784 and NSF OAC-2103889. PI and MW are supported in part by NSF OAC-2103889. SM is partially supported by the DOE/HEP QuantISED program grant “HEP Machine Learning and Optimization Go Quantum”, identification number 0000240323.

Broader Impact

In high energy particle physics, characterizing the uncertainty of phenomenological models is critical for interpreting results. To date, common hadronization models have not included uncertainty estimates. This NF architecture will allow us to explore such uncertainties with a well-defined methodology. While the resulting hadronization model may not be applicable to the larger community, the methodology for interpreting the estimated uncertainty will provide an important benchmark across a wide range of ML applications. Additionally, we will work towards interpretation of the model for a specific physical process. Such interpretation and uncertainty estimation of ML architectures will play a crucial role in understanding the ethical implications of broader-ranging ML applications such as self-driving cars, parole determination, or healthcare.

References

[1] R. Frederix, S. Frixione, V. Hirschi, D. Pagani, H. S. Shao, and M. Zaro. The automation of next-to-leading order electroweak calculations. *JHEP*, 07:185, 2018. doi: 10.1007/JHEP11(2021)085. [Erratum: JHEP 11, 085 (2021)].

[2] Johannes Bellm, Stefan Gieseke, and Simon Plätzer. Merging NLO Multi-jet Calculations with Improved Unitarization. *Eur. Phys. J. C*, 78(3):244, 2018. doi: 10.1140/epjc/s10052-018-5723-2.

[3] John M. Campbell, Stefan Höche, Hai Tao Li, Christian T. Preuss, and Peter Skands. Towards NNLO+PS Matching with Sector Showers. 8 2021.

[4] Stefan Höche and Stefan Prestel. The midpoint between dipole and parton showers. *Eur. Phys. J. C*, 75(9):461, 2015. doi: 10.1140/epjc/s10052-015-3684-2.

[5] Bo Andersson, G. Gustafson, G. Ingelman, and T. Sjostrand. Parton Fragmentation and String Dynamics. *Phys. Rept.*, 97:31–145, 1983. doi: 10.1016/0370-1573(83)90080-7.

[6] Bo Andersson. The Lund model. *Camb. Monogr. Part. Phys. Nucl. Phys. Cosmol.*, 7:1–471, 1997.

[7] Silvia Ferreres-Solé and Torbjörn Sjöstrand. The space–time structure of hadronization in the Lund model. *Eur. Phys. J. C*, 78(11):983, 2018. doi: 10.1140/epjc/s10052-018-6459-8.

[8] Richard D. Field and Stephen Wolfram. A QCD Model for e+ e- Annihilation. *Nucl. Phys. B*, 213:65–84, 1983. doi: 10.1016/0550-3213(83)90175-X.

[9] Thomas D. Gottschalk. An Improved Description of Hadronization in the {QCD} Cluster Model for e^+e^- Annihilation. *Nucl. Phys. B*, 239:349–381, 1984. doi: 10.1016/0550-3213(84)90253-0.

[10] B.R. Webber. A QCD Model for Jet Fragmentation Including Soft Gluon Interference. *Nucl. Phys. B*, 238:492–528, 1984. doi: 10.1016/0550-3213(84)90333-X.

[11] Torbjörn Sjöstrand, Stefan Ask, Jesper R. Christiansen, Richard Corke, Nishita Desai, Philip Ilten, Stephen Mrenna, Stefan Prestel, Christine O. Rasmussen, and Peter Z. Skands. An introduction to PYTHIA 8.2. *Comput. Phys. Commun.*, 191:159–177, 2015. doi: 10.1016/j.cpc.2015.01.024.

[12] Johannes Bellm et al. Herwig 7.0/Herwig++ 3.0 release note. *Eur. Phys. J. C*, 76(4):196, 2016. doi: 10.1140/epjc/s10052-016-4018-8.

[13] Enrico Bothmann et al. Event Generation with Sherpa 2.2. *SciPost Phys.*, 7(3):034, 2019. doi: 10.21468/SciPostPhys.7.3.034.

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL `https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf`.

[15] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014.

[16] Soheil Kolouri, Charles E. Martin, and Gustavo K. Rohde. Sliced-wasserstein autoencoder: An embarrassingly simple generative model. *CoRR*, abs/1804.01947, 2018. URL http://arxiv.org/abs/1804.01947.

[17] Ilya O. Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schölkopf. Wasserstein auto-encoders. *CoRR*, abs/1711.01558, 2017. URL http://arxiv.org/abs/1711.01558.

[18] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 1530–1538. JMLR.org, 2015.

[19] Marco Bellagente, Anja Butter, Gregor Kasieczka, Tilman Plehn, Armand Rousselot, Ramon Winterhalder, Lynton Ardizzone, and Ullrich Köthe. Invertible Networks or Partons to Detector and Back Again. *SciPost Phys.*, 9:74, 2020. doi: 10.21468/SciPostPhys.9.5.074. URL https://scipost.org/10.21468/SciPostPhys.9.5.074.

[20] Marco Bellagente, Anja Butter, Gregor Kasieczka, Tilman Plehn, and Ramon Winterhalder. How to GAN away Detector Effects. *SciPost Phys.*, 8:70, 2020. doi: 10.21468/SciPostPhys.8.4.070. URL https://scipost.org/10.21468/SciPostPhys.8.4.070.

[21] Aishik Ghosh, Xiangyang Ju, Benjamin Nachman, and Andrzej Siodmok. Towards a deep learning model for hadronization, 2022. URL https://arxiv.org/abs/2203.12660.

[22] Phil Ilten, Tony Menzo, Ahmed Youssef, and Jure Zupan. Modeling hadronization using machine learning, 2022. URL https://arxiv.org/abs/2203.04983.

[23] George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/6c1da886822c67822bcf3679d04369fa-Paper.pdf.

[24] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. URL https://arxiv.org/abs/1410.8516.

[25] Diederik P. Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improving variational inference with inverse autoregressive flow, 2016. URL https://arxiv.org/abs/1606.04934.

[26] Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. nflows: normalizing flows in PyTorch, November 2020. URL https://doi.org/10.5281/zenodo.4296287.

[27] Yang-Ting Chien, Abhay Deshpande, Mriganka Mouli Mondal, and George Sterman. Probing hadronization with flavor correlations of leading particles in jets. *Phys. Rev. D*, 105(5):L051502, 2022. doi: 10.1103/PhysRevD.105.L051502.

[28] Rabah Abdul Khalek, Valerio Bertone, and Emanuele R. Nocera. Determination of unpolarized pion fragmentation functions using semi-inclusive deep-inelastic-scattering data. *Phys. Rev. D*, 104(3):034007, 2021. doi: 10.1103/PhysRevD.104.034007.

[29] V. Bertone, N. P. Hartland, E. R. Nocera, J. Rojo, and L. Rottoli. Charged hadron fragmentation functions from collider data. *Eur. Phys. J. C*, 78(8):651, 2018. doi: 10.1140/epjc/s10052-018-6130-4.

[30] Maryam Soleymaninia, Hadi Hashamipour, and Hamzeh Khanpour. Neural network QCD analysis of charged hadron fragmentation functions in the presence of SIDIS data. *Phys. Rev. D*, 105(11):114018, 2022. doi: 10.1103/PhysRevD.105.114018.

[31] Hao Chen, Ian Moult, Jesse Thaler, and Hua Xing Zhu. Non-Gaussianities in collider energy flux. *JHEP*, 07:146, 2022. doi: 10.1007/JHEP07(2022)146.

[32] Patrick T. Komiske, Ian Moult, Jesse Thaler, and Hua Xing Zhu. Analyzing N-point Energy Correlators Inside Jets with CMS Open Data. 1 2022.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] see sec. 4
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] see sec. 2
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Plots for different labels

A.1 Known labels

In fig. 5 to 8 one can see the results on labels, which the flow was not trained trained on.

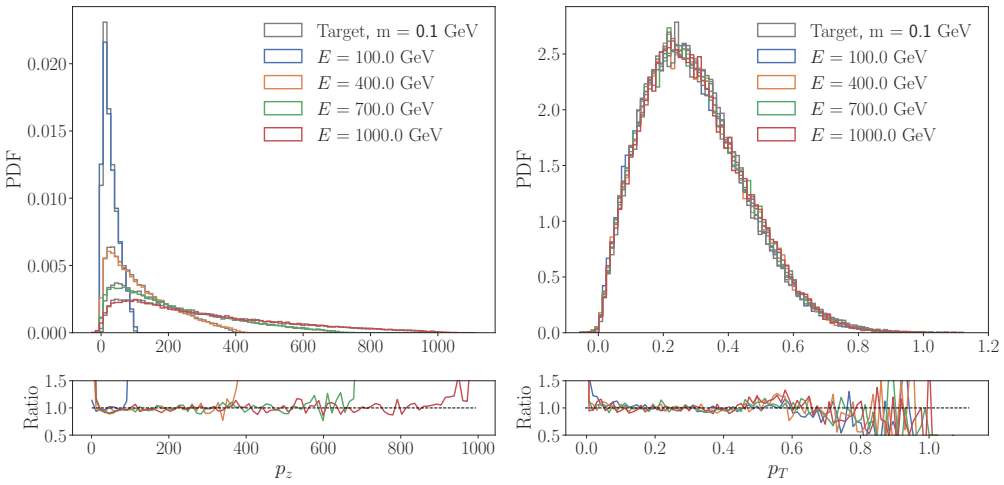

Figure 5

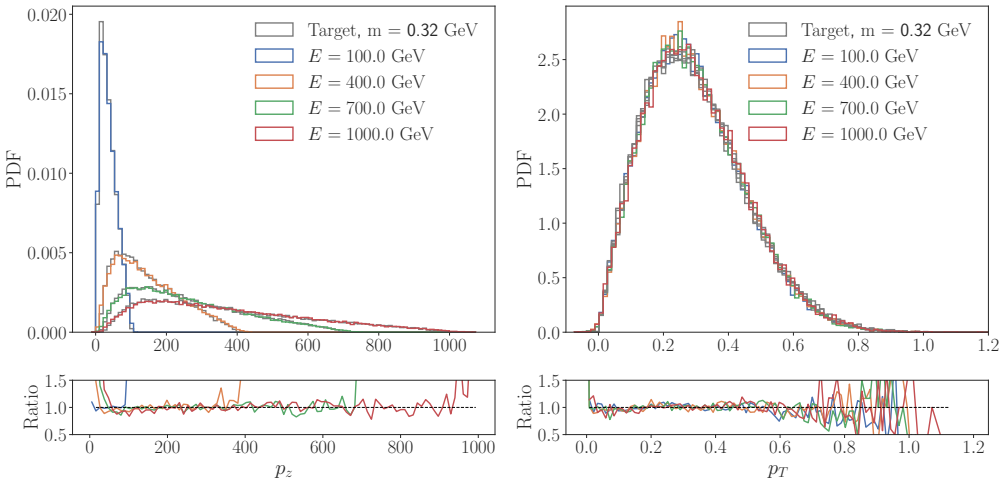

Figure 6

A.2 Unknown labels

In fig. 9 to 12 one can see the results on labeled trained on.

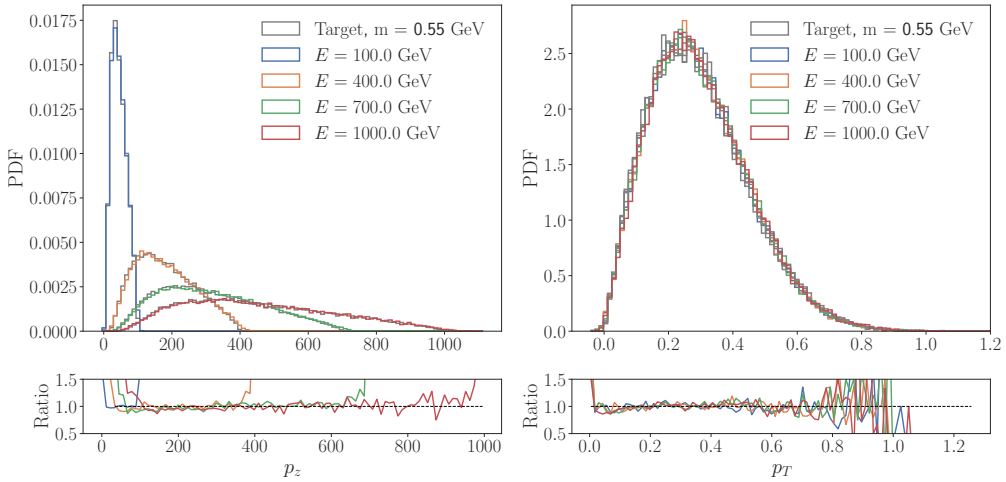

Figure 7

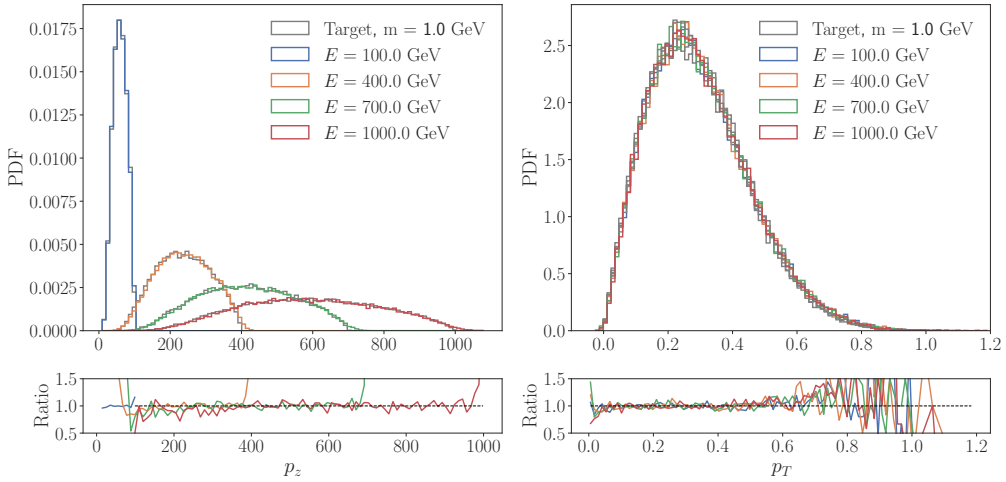

Figure 8

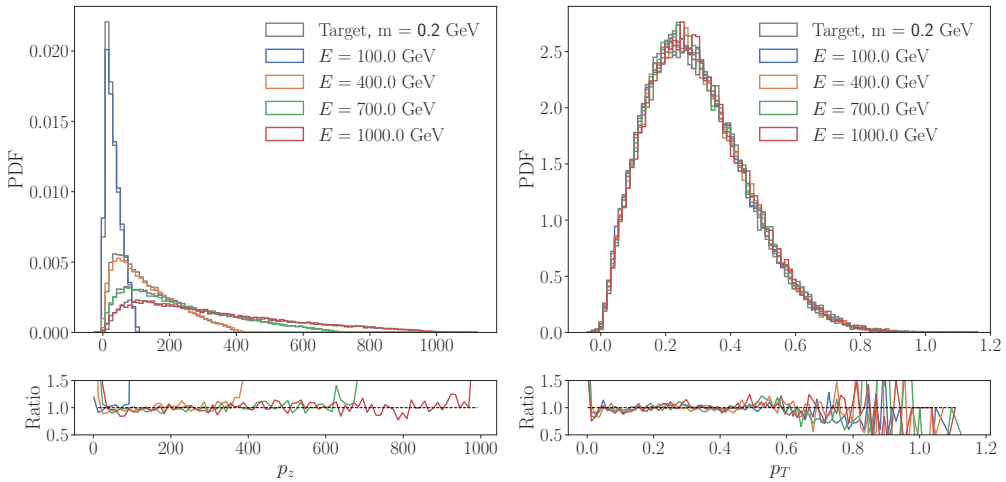

Figure 9

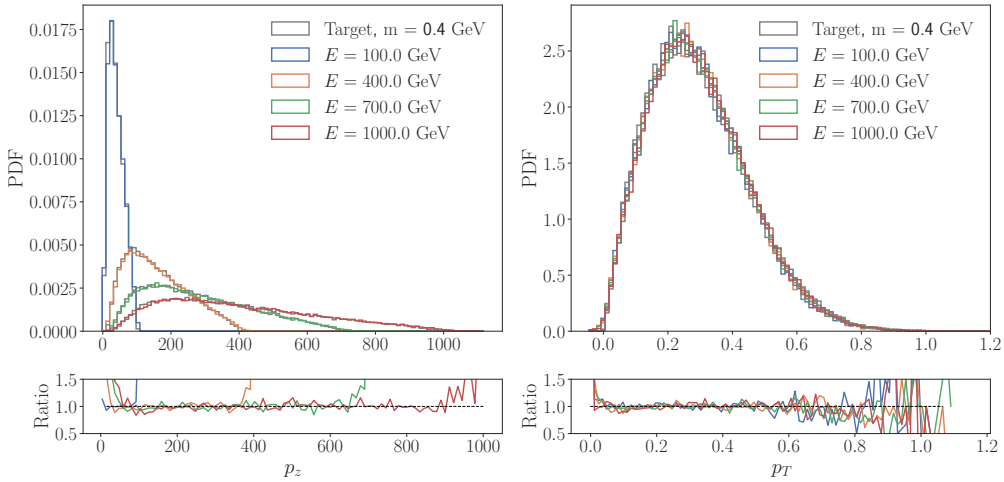

Figure 10

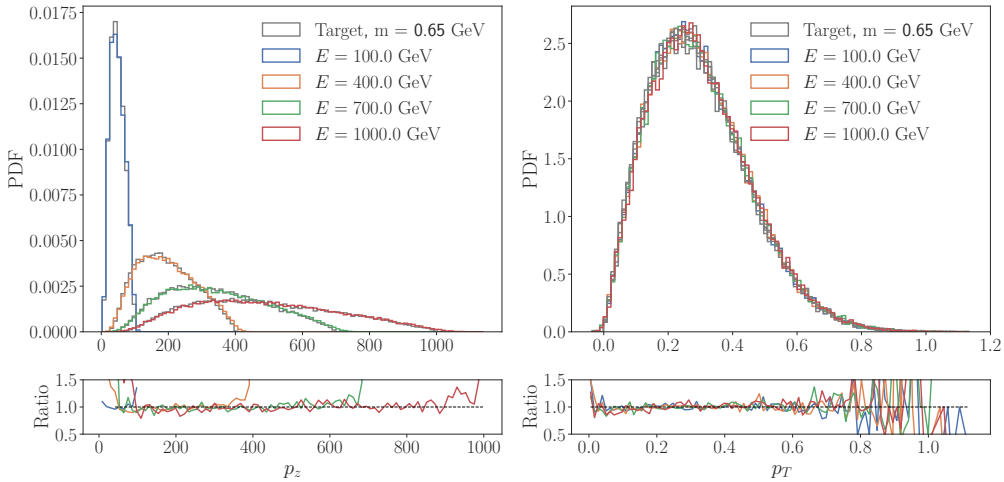

Figure 11

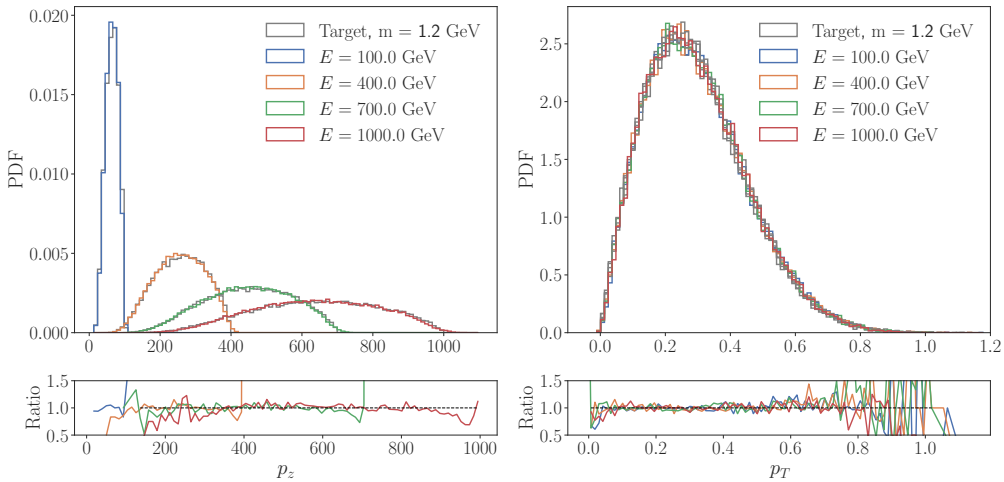

Figure 12