Implicit Quantile Neural Networks for Jet Simulation and Correction

Braden Kronheim
Department of Physics
University of Maryland
College Park, MD 20742
bkronhei@umd.edu

Michelle P. Kuchera
Department of Physics
Davidson College
Davidson, NC 28035
mikuchera@davidson.edu

Harrison B. Prosper
Department of Physics
Florida State University
Tallahassee, FL 32306
hprosper@fsu.edu

Raghuram Ramanujan
Department of Mathematics & Computer Science
Davidson College
Davidson, NC 28035
raramanujan@davidson.edu

Abstract

Reliable modeling of conditional densities is important for quantitative scientific fields such as particle physics. In domains outside physics, implicit quantile neural networks (IQN) have been shown to provide accurate models of conditional densities. We present a successful application of IQNs to jet simulation and correction using the tools and simulated data from the Compact Muon Solenoid (CMS) Open Data portal.

1 Introduction

The ability to model and sample from multi-dimensional conditional densities is of critical importance in particle physics where almost all phenomena are subject to random fluctuations. One example is the interaction of a jet, a collimated collection of particles (see, for example, [1]), with a particle detector. The manner in which a jet interacts with a detector is usually modeled using a Monte Carlo method based on the widely used `GEANT4` toolkit [2]. `GEANT4` provides a high-fidelity stochastic simulation of particle interactions with matter, but comes with a high computational cost. In this paper, we demonstrate how given a large sample of jets simulated with `GEANT4`, the jet response function — which maps jets from the particle generation level (i.e., before the particles enter the particle detectors) to the observed jets (i.e., jets recorded in the detectors) — can be accurately modeled with an Implicit Quantile Neural Network (IQN). Moreover, the IQN performs this mapping at a significantly faster rate than the original `GEANT4`-based simulation. In addition, an IQN can model the inverse problem: the distribution of particle generation-level jets that could have yielded a given observed jet. The efficacy of IQNs in modeling multi-dimensional conditional densities is illustrated using simulated jet data provided by the CMS Collaboration at the CERN Open Data Portal[1].

[1] `https://opendata.cern.ch/`

Fourth Workshop on Machine Learning and the Physical Sciences (NeurIPS 2021).

2 Implicit Quantile Neural Networks

Given a training dataset comprising inputs $\boldsymbol{x}$ and a target y, the *quantile regression* problem is to construct an estimator $f(\boldsymbol{x};\boldsymbol{\theta})$ with parameters $\boldsymbol{\theta}$ of the quantile function, which is the inverse of the cumulative distribution function $F(y)$, conditioned on $\boldsymbol{x}$. We denote a specific quantile by $\tau = F(y)$.

This problem can be cast as an optimization problem [3] in which the average loss,

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{t:y_t \geq f(\boldsymbol{x}_t;\boldsymbol{\theta})} \tau|y_t - f(\boldsymbol{x}_t;\boldsymbol{\theta})| + \sum_{t:y_t < f(\boldsymbol{x}_t;\boldsymbol{\theta})} (1-\tau)|y_t - f(\boldsymbol{x}_t;\boldsymbol{\theta})|, \tag{1}$$

over a set of training examples $(\boldsymbol{x}_t, y_t)$ is minimized.

The use of neural networks as the estimator, $f(\boldsymbol{x};\boldsymbol{\theta})$, in Eq. 1 was first studied by White [4] and Taylor [5]. In high-energy physics, deep neural networks that optimize this "quantile loss" have been used to model detector response and to perform jet reconstruction [6, 7]. In those works, researchers focused on accurately modeling specific quantiles of interest (for example, $\tau = 0.25$). In contrast, we model the full quantile function using deep neural networks by extending the network architecture to include the quantile τ as an input — an approach first described by Ostrovski *et al.* [8]. That work additionally introduced a method to handle multi-dimensional probability densities of the form $p(y^{(1)}, y^{(2)}, \cdots, y^{(n)}|\boldsymbol{x})$ as follows

$$p(y^{(1)}, y^{(2)}, \ldots, y^{(n)}|\boldsymbol{x}) = p(y^{(1)}|\boldsymbol{x}) \prod_{i=2}^{n} p(y^{(i)}|\boldsymbol{x}, y^{(1)}, \ldots, y^{(i-1)}), \tag{2}$$

where $p(y^{(1)}|\boldsymbol{x}), p(y^{(2)}|\boldsymbol{x}, y^{(1)}), p(y^{(3)}|\boldsymbol{x}, y^{(1)}, y^{(2)}), \ldots, p(y^{(n)}|\boldsymbol{x}, y^{(1)}, y^{(2)}, \ldots, y^{(n-1)})$ are 1-dimensional conditional densities.

In order to apply Ostrovski *et al.*'s method to the jet quantile regression problem, a single training example $(p_T, \eta, \phi, m) \to (p'_T, \eta', \phi', m')$ is unrolled into the four training examples

$$\begin{aligned}
(p_T, \eta, \phi, m, 1, 0, 0, 0, 0, 0, 0) &\to (p'_T), \\
(p_T, \eta, \phi, m, 0, 1, 0, 0, p'_T, 0, 0) &\to (\eta'), \\
(p_T, \eta, \phi, m, 0, 0, 1, 0, p'_T, \eta', 0) &\to (\phi'), \\
(p_T, \eta, \phi, m, 0, 0, 0, 1, p'_T, \eta', \phi') &\to (m'),
\end{aligned} \tag{3}$$

where p_T, η, ϕ, m are the jet transverse momentum, pseudo-rapidity, azimuthal angle, and mass, respectively. The one-hot encoding after p_T, η, ϕ, m in Eq. (3) specifies which target is associated with the given unrolled example. To accommodate the form of the data in Eq. (3), we use a model $f(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}', \tau; \boldsymbol{\theta})$ that includes inputs $\boldsymbol{z}$ and $\boldsymbol{y}'$ for the one-hot encoding and the partially specified target vector, respectively. The quantile τ is also an input, albeit one that is generated on-the-fly at training time for each example at each epoch, by drawing an independent sample from $U(0, 1)$.

We train a dense, feed-forward network on batches of unrolled examples (augmented with an independent τ sample) sampled from the training set. At inference time, the trained model is used in an autoregressive manner: the unrolled examples, each with the desired quantile, are provided in the order shown in Eq. (3) with the quantities p'_T, η', ϕ' now the values *predicted* by the trained model, rather than the values from the test data.

Since the trained model approximates four quantile functions — one quantile function at a time depending on which one-hot encoding is used — the model is an implicit approximation of the multi-dimensional conditional density $p(\boldsymbol{y}|\boldsymbol{x})$. The 1-dimensional conditional densities from which $p(\boldsymbol{y}|\boldsymbol{x})$ is formed using Eq. (2) can be computed from $p(y^n|\boldsymbol{x}, y^{(1)}, \cdots, y^{(n-1)}) = (\partial f_n/\partial\tau)^{-1}$, where f_n is the model supplied with the n^{th} one-hot encoding.

One problem that often arises in quantile regression is *quantile crossing*, where the approximation to the quantile function is not monotonic. Prior work has attempted to mitigate this problem by imposing constraints on the model architecture or by optimizing novel loss functions [9, 10]. Tagasovska and Lopez-Paz, however, observed that the problem becomes significantly less pronounced when the full quantile function is approximated [11]. In this paper, we propose a regularized average loss

$$\mathcal{L}' = \mathcal{L} + \lambda \cdot \mathbb{1}[-f'](f')^2$$

that further alleviates this problem, where $f' = \partial f/\partial\tau$ and λ is a hyperparameter. By penalizing negative gradients of f with respect to τ, the regularization term favors solutions that are monotonically non-decreasing. This is equivalent to the condition $(f')^{-1} = p(y|\boldsymbol{x}) > 0$.

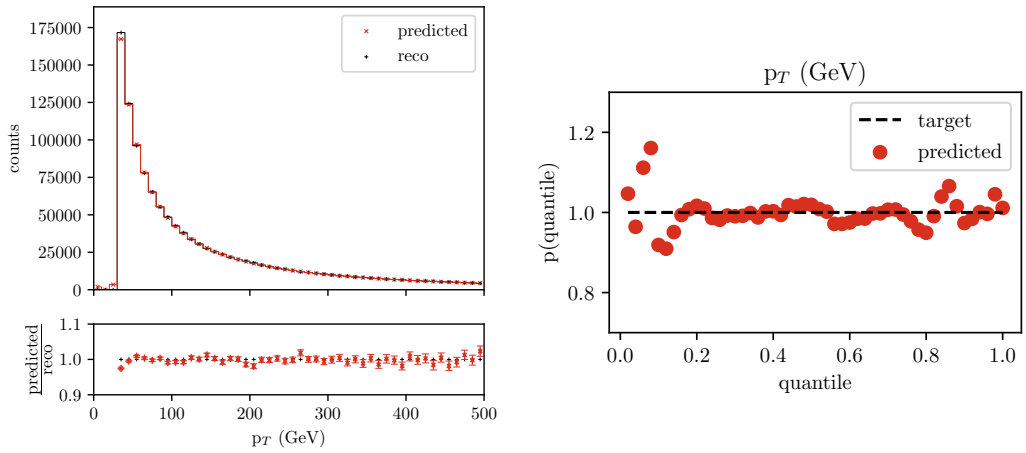

Figure 1: (Left) The predicted marginal raw reco-jet p_T spectrum is compared with the raw reco-jet p_T data from the test set. The predicted histogram displays the median of 1000 predicted marginal distributions. The ratio of the displayed spectra is shown in the lower plot. The upper and lower uncertainties are the 0.84 and 0.16 quantiles of the bin count distributions. (Right) The distribution of quantiles, each computed from the true reco-jet transverse momenta using the trained model.

3 Applications

3.1 Jets

In this study, IQNs are used to create a fast simulation of jets using the publicly available tools and simulated data provided by the CMS Collaboration at the CERN Open Data Portal. Simulation of particle collisions (typically, proton-proton collisions) at the Large Hadron Collider (LHC)[2] begin with the simulation of collisions between proton constituents — quarks and gluons — which are collectively referred to as partons. Partons are not directly observed; therefore, their transformation into observable particles called hadrons (via a process called hadronization) must also be simulated. A particle clustering algorithm, called anti-k_T [12], is then used to cluster the hadrons into jets. In a simulation, the clustering can be performed either on the particles that would be identified from measured tracks and energy deposits in particle detectors, or on the particles before they interact with the detectors. The jets created using the latter method are generally referred to as particle *generator* (or gen-) jets, while those created using the former method are called *reconstructed* (or reco-) jets. In the examples studied here, gen-jets will be considered the ground truth, i.e., the jets that would be observed with noise-free detectors. The reco-jets are usually subjected to corrections that render their characteristics as similar as possible to those of the gen-jets, on average, using techniques based on experimental and simulated data [1]. The reco-jets without these corrections will be called the *raw* reco-jets, while those with the corrections will simply be called reco-jets.

The simulated CMS jet data [13], as well as the analysis code (which was modified for this study), come from the CERN Open Data Portal [14] with Creative Commons CC0 waivers and GPL3 licenses respectively. Approximately 3.9 million simulated jets were extracted from the portal of which $\sim$ 1.3 million were set aside as test data. Of the remaining $\sim$ 2.6 million examples, $\sim$ 2.34 million examples were used as training data, leaving the remaining $\sim 260,000$ examples for model validation and hyperparameter tuning. The complete code to reproduce the results in this paper can be found at the linked Github repository[3] with a GPL3 license.

3.2 Detector-Level Jet Simulation

Jet simulation entails mapping particle-level jets (gen-jets) to the corrected jets (reco-jets) that are observed in particle detectors. The IQN used for this mapping problem is a 5-layer, fully-connected

[2] `https://home.cern/science/accelerators/large-hadron-collider`
[3] `https://github.com/alpha-davidson/IQNs-for-Jets`

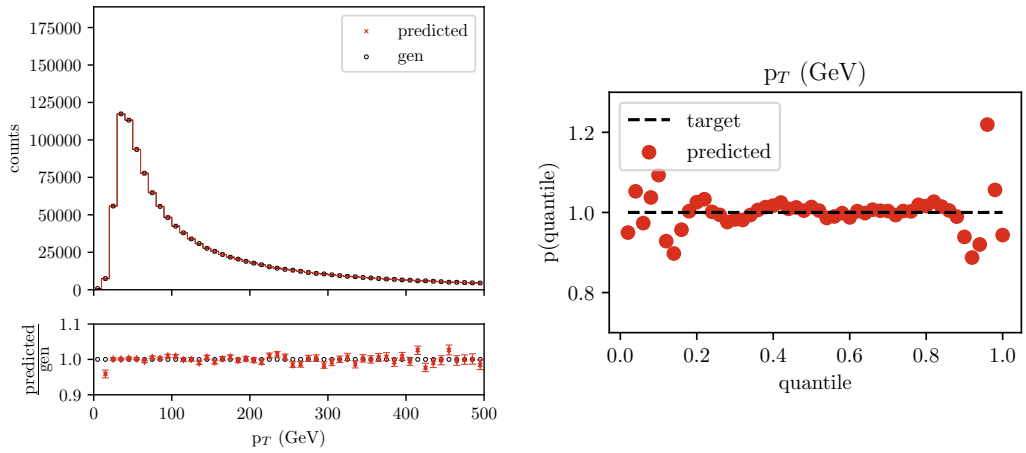

Figure 2: (Left) The predicted gen-jet p_T spectrum compared with the true gen-jet p_T spectrum. Both spectra are integrated over the raw reco-jet quantities. The ratio of the spectra is shown in the lower plot. The uncertainties reflect the statistical fluctuations in the 1000 predicted spectra. (Right) The distribution of quantiles, each computed from the gen-jet transverse momenta in the test set using the trained model.

neural network with 50 nodes per layer. We use the Leaky ReLU activation function and train the network using AMSGrad [15] using only CPU support. A typical training run lasted about 24 hours.

We assess the effectiveness of the trained model by comparing the predicted marginal density of each component of the reco-jet four-momentum (p'_T, η', ϕ', m') with the corresponding true reco-jet marginal density from the data, both integrated over the gen-jet quantities. Specifically, we use the trained model to generate 1000 reco-jet four-momenta predictions for each of the 10^6 gen-jets from the test set. From these 10^9 predictions conditioned on the gen-jet inputs, we derive 1000 marginal distributions for each jet parameter, by aggregating over randomly sampled batches of size 10^6. We present the median of these 1000 marginal distributions for p_T in the left panel of Fig. 1, alongside the marginal distribution computed from the test data. We see that the predicted marginal distributions are nearly modeled within uncertainty. We obtained similar results for the other three jet variables as well, but have omitted them in the interest of space.

Additionally, we perform the following closure test to validate our approach. We construct a cumulative distribution function (cdf) $F(y')$ for each of the 10^6 gen-jets in the test set, where $y' \in \{p'_T, \eta', \phi', m'\}$, for each component of the reco-jet momentum vector from our model's 10^9 predictions. We then use this induced cdf to map each reco-jet measurement from the test set to a quantile. If the IQN's modeling of the quantile function, and therefore the conditional distribution, is accurate, then we should expect the distribution of these quantiles to follow $U(0, 1)$. We see that this is indeed the case, as shown in the right panel of Fig. 1. Once again, we have chosen to only present the results for p_T and omit the qualitatively similar results obtained on the other three variables.

3.3 Jet Inverse Problem

An IQN can also be used to to solve the jet inverse problem: what gen-jets are consistent with being the progenitors of a given raw reco-jet? Such a function could be useful, for example, in estimating the particle-level spectrum of a jet quantity from the corresponding observed spectrum. In Fig. 2, we show the marginal densities for the predicted gen-jet p_T spectra, the test gen-jet data, and their ratio, computed using the protocol outlined in Sec. 3.2, i.e. using 10^9 predictions divided into 1000 batches of 10^6 each. The figure also shows the distribution of quantiles, this time computed from the true gen-jet p_T. The predicted spectra are again nearly within uncertainty across p_T, and a closure test is performed with similar results as the forward predictions.

4 Conclusions

In this work, we presented an application of IQNs to the tasks of jet correction and simulation. Our IQN architecture comprises a feed-forward, fully-connected neural network, which is straightforward to train, particularly when compared to other generative modeling techniques such as GANs that tend to be more unstable and require more careful tuning. Our approach utilizes a novel regularization term that helps mitigate quantile crossing, without interfering with the convergence of the network. The trained IQNs approximate the marginal densities of the jet variables nearly within the uncertainty of the predictions, across the parameter space. Confirming that the multi-dimensional conditional densities $p(\boldsymbol{y}|\boldsymbol{x})$ are modeled correctly beyond the correspondence shown in the closure tests presents additional challenges, as those distributions are not explicitly present in the data, and is the focus of ongoing study.

5 Impact Statement

Conditional densities are ubiquitous in particle physics. For example, they appear in statistical models $p(\boldsymbol{d}|\boldsymbol{\mu},\boldsymbol{\nu})$, where $\boldsymbol{d}$ are observable data and $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ are parameters of interest and nuisance parameters, respectively. They also appear in response functions $r(\boldsymbol{y}|\boldsymbol{x})$ that appear in multi-dimensional integrals of the form $o(\boldsymbol{y}) = \int r(\boldsymbol{y}|\boldsymbol{x})\, u(\boldsymbol{x})\, d\boldsymbol{x}$ that map an unobserved spectrum $u(\boldsymbol{x})$ to an observed spectrum $o(\boldsymbol{y})$ of which the jet 4-momenta spectra are a typical example. There is a renewed push in particle physics to publish full statistical models. IQNs provide a simple and effective way to both encapsulate statistical models and compute them very quickly, as well as to model the numerous response functions that appear in the analysis of particle physics data at the Large Hadron Collider and other particle physics research facilities. Finally, IQNs could be the basis of very fast detector simulations that are automatically abstracted from existing and future high-fidelity `GEANT4`-based event simulations.

From a machine learning standpoint, this paper introduces a novel regularization term to enforce the requirement that the neural network learn a monotonically increasing quantile function. This modification to the standard quantile regression loss function is potentially useful to any application that requires a model to produce a statistically meaningful quantile function and cumulative distribution function post-training.

Acknowledgments and Disclosure of Funding

This work was supported by the National Science Foundation under Cooperative Agreement OAC-1836650.

References

[1] Ariel Schwartzman. Jet energy calibration at the lhc. *International Journal of Modern Physics A*, 30(31), 11 2015.

[2] J. Allison, K. Amako, John Apostolakis, Pedro Arce, Makoto Asai, T. Aso, Enrico Bagli, A. Bagulya, S. Banerjee, Guy Barrand, B.R. Beck, A.G. Bogdanov, D. Brandt, Jeremy Brown, H. Burkhardt, Ph Canal, Daniel Ott, Stephane Chauvie, K. Cho, and H. Yoshida. Recent developments in geant4. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 835, 07 2016.

[3] Roger Koenker and Gilbert Bassett. Regression quantiles. *Econometrica*, 46(1):33, 1978.

[4] Halbert White. Nonparametric estimation of conditional quantiles using neural networks. In Connie Page and Raoul LePage, editors, *Computing Science and Statistics*, pages 190–199, New York, NY, 1992. Springer New York.

[5] James W. Taylor. A quantile regression neural network approach to estimating the conditional density of multiperiod returns. *Journal of Forecasting*, 19(4):299–311, 2000.

[6] S. Cheong, A. Cukierman, B. Nachman, M. Safdari, and A. Schwartzman. Parametrizing the detector response with neural networks. *Journal of Instrumentation*, 15(01), 1 2020.

[7] A. M. Sirunyan et al. A deep neural network for simultaneous estimation of b jet energy and resolution. *Computing and Software for Big Science*, 4(1):10, Oct 2020.

[8] Georg Ostrovski, Will Dabney, and Remi Munos. Autoregressive quantile networks for generative modeling. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3936–3945. PMLR, 10–15 Jul 2018.

[9] Alex J. Cannon. Non-crossing nonlinear regression quantiles by monotone composite quantile regression neural network, with application to rainfall extremes. *Stochastic Environmental Research and Risk Assessment*, 32(11):3207–3225, 2018.

[10] Sang Jun Moon, Jong-June Jeon, Jason Sang Hun Lee, and Yongdai Kim. Learning multiple quantiles with neural networks. *Journal of Computational and Graphical Statistics*, 0(0):1–11, 2021.

[11] Natasa Tagasovska and David Lopez-Paz. Single-model uncertainties for deep learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[12] Matteo Cacciari, Gavin P Salam, and Gregory Soyez. The anti-ktjet clustering algorithm. *Journal of High Energy Physics*, 2008(04):063–063, apr 2008.

[13] CMS Collaboration. Simulated dataset QCD_Pt-15to7000_TuneCUETP8M1_Flat_13TeV_pythia8 in MINIAODSIM format for 2016 collision data. *CERN Open Data Portal*, 2019.

[14] Kimmo Kallonen. JetNtupleProducerTool - jet tuple producer from CMS run2 miniaod. *CERN Open Data Portal*, 2019.

[15] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *International Conference on Learning Representations*, 2018.

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] The limitations of the models trained are discussed in section 3.2
   (c) Did you discuss any potential negative societal impacts of your work? [No] This paper considers a novel application of an existing machine learning technique to particle physics data; any negative societal impacts would be indirect.
   (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] We link to the GitHub repository containing our code in Section 3.1.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] The most important details are presented in Sections 2, 3.1 and 3.2. Complete training details are provided as part of the code repository (link is provided in Section 3.1). The motivation for choosing specific hyperparameters is not discussed, because the method is fairly robust to a wide range of hyperparameter choices.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Uncertainties are reported in Figures 1 and 2.

(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)? [Yes] See Section 3.2.

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] See Section 3.1.
   (b) Did you mention the license of the assets? [Yes] See Section 3.1.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] See Section 3.1.
   (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]