Automated detector simulation and reconstruction parametrization using machine learning

D. Benjamin,[1] S. Chekanov,[1] W. Hopkins,[1] Y. Li,[2] and J. R. Love[1]

[1] *High Energy Physics Division, Argonne National Laboratory, 9700 S. Cass Avenue, Argonne, IL 60439, USA*

[2] *Computational Science Division, Argonne National Laboratory, 9700 S. Cass Avenue, Argonne, IL 60439, USA*

(Dated: February 4, 2020)

Abstract

Rapidly applying the effects of detector respone to physics objects (e.g. electrons, muons, showers of particles) is essential in high energy physics. Currently available tools for the transformation from truth-level physics objects to reconstructed detector-level physics objects involve manually defining resolution functions. These resolution functions are typically derived in bins of variables that are correlated with the resolution (e.g. pseudorapidity and transverse momentum). This process is time consuming, requires manual updates when detector conditions change, and can miss important correlations. Machine learning offers a way to automate the process of building these truth-to-reconstructed object transformation functions and can capture complex correlation of these functions for any given set of input variables. Such machine learning algorithms, with sufficient optimization, could have a wide range of applications: improving phenomenological studies by using a better detector representation, allowing for more efficient production of Geant4 simulation by only simulating events within an interesting part of phase space, and studies on future experimental sensitivity to new physics.

I. INTRODUCTION

A cornerstone of particle collision experiments is the Monte Carlo (MC) simulation of physics processes resulting from collisions of high-energy particles, followed by the simulation of detector responses and object reconstruction. The MC simulation produces objects (jets, electrons, muons, etc) with properties (four momenta, particle types) which entirely depend on the physics processes occurring. These objects are commonly referred to as "truth" objects. These objects are altered by interactions with the detector and are reconstructed with experimental algorithms. Such objects, that have undergone a transformation due to detector interactions and reconstruction will be referred to as "reco" objects in this paper.

With the increased complexity of such experiments, such as those at the Large Hadron Collider (LHC), the detector simulations become increasing complex and time consuming. Parameterized detector simulations, such as Delphes [1], have been proven to be a vital tool for physics performance and phenomological studies (i.e. to estimate the sensitivity of an experiment to a new physics model). An approximation of the detector responses and experimental object reconstruction can, however, also be performed by neural networks (NN) trained using the Geant4-based simulations that have gone through an experiment's reconstruction algorithm. This NN could then computationally rapidly transform truth MC objects (jets and other identified particles) to objects modified by a detector and experimental reconstruction algorithms.

The main advantage of a detector parametrization based on machine learning (ML), as compared to a manually constructed parametrization such as Delphes, is that a neural network can automatically learn the features introduced by detailed full simulations avoiding the need to handcraft parameters to represent resolutions and inefficiencies. An NN trained using realistic detector simulation could memorize the transformation from the truth to the reco quantities without manual binning of quantities that are correlated to the transformation by analyzers. Another advantage is that the NN approach can introduce a complex interdependence of variables which is currently difficult to implement in parameterized simulations. Finally, since the underlying libraries used for ML (e.g. Keras [2], pyTorch [3], etc) are optimized for a wide range of hardware, an NN-based truth-to-reco transformation would be able to run efficiently on heterogeneous hardware resources (resources that use a varied set of processors such as GPUs and CPUs).

As a first step towards parameterized detector simulations with ML, it is instructive to investigate how a transformation from the truth to reco objects can be performed, leaving aside the question of introducing objects that are created by misreconstructions or objects that are lost due to inefficiencies.

II. TRADITIONAL PARAMETERIZED FAST SIMULATIONS

In abstract terms, a typical variable $\xi_i^{\rm reco}$ that characterizes a reconstructed particle/jet, such as transverse momentum (${p_{\rm T}}^{\rm reco}$) or pseudorapidity ($\eta^{\rm reco}$), can be viewed as the result of a multivariate transform, F, of the original variable $\xi_1^{\rm truth}$ at truth level:

$$\xi_1^{\rm reco} = F(\xi_1^{\rm truth}, \xi_2^{\rm truth}, \xi_3^{\rm truth}, ...\xi_{\rm N}^{\rm truth}).$$

Generally, such a transform depends on several other variables $\xi_2^{\rm truth}$ .. $\xi_{\rm N}^{\rm truth}$ characterizing this (or other) objects at truth level. For example, the extent at which jet transverse momentum, $p_{\rm T}$ is modified by a detector depends on the original truth-level transverse momentum ($\xi_1^{\rm truth} = p_T^{\rm truth}$), pseudorapidity ($\xi_2^{\rm truth} = \eta^{\rm truth}$), and other effects that can be inferred from truth quantities. Similarly, if particular detector modules in the azimuthal angle (ϕ) are not active, this would introduce an additional dependence of this transform on ϕ.

Typical parameterized simulations ignore the full range of correlations between the truth-level variables. In most cases, the above transform is reduced to a single variable, or two (as in the case of Delphes simulations where the energy resolution of clusters depends on the original energies of particles and their positions in η). In order to take into account correlations between multiple parameters characterizing transformations to reconstruction objects, a grid in the hypercube with the dimension $N_{\rm b}^{\rm N}$, where $N_{\rm b}$ is the number of histogram bins for the distributions $(\xi^{\rm reco} - \xi^{\rm truth})/\xi^{\rm truth}$, representing the "resolution", must be created. This methodology results in a large number of histograms when there are many correlated variables that affect the resolution.

It should be pointed out that the calculation speed for parameterized simulations of one variable that depends on N other variables at the truth level is proportional to $N_{\rm b}^{\rm N}$ since each object at the truth level should be placed inside the grid defined by $N_{\rm b}$ bins. Therefore, complex parameterisations of resolutions and efficiencies for $N > 2$ becomes computationally intensive.

III. JET TRUTH-TO-RECO TRANSFORMATION WITH ML

To test the viability of using ML to transform truth objects to reco objects, we studied the truth-to-reco transformation for jets. Jet truth-level quantities, such as jet p_{T}, η, ϕ and jet mass (m) are used as training inputs to an NN while the output is an array of nodes that represent the binned probability density function (PDF) of the resolution for a single variable (such as jet p_{T}). Additional input variables could be any variable that can influence the resolution of a jet, such as jet flavor at the truth level, jet radius, etc. Figure 1 shows a schematic representation of the NN architecture for modelling the detector response for a single output variable. The aim is to have the NN learn the shape of the resolution PDF, for example for the p_{T}, depending on other input variables such as the η of the object. A binned output (multi-categorization) was used so that the precision of the resolution PDF modelling can be chosen.

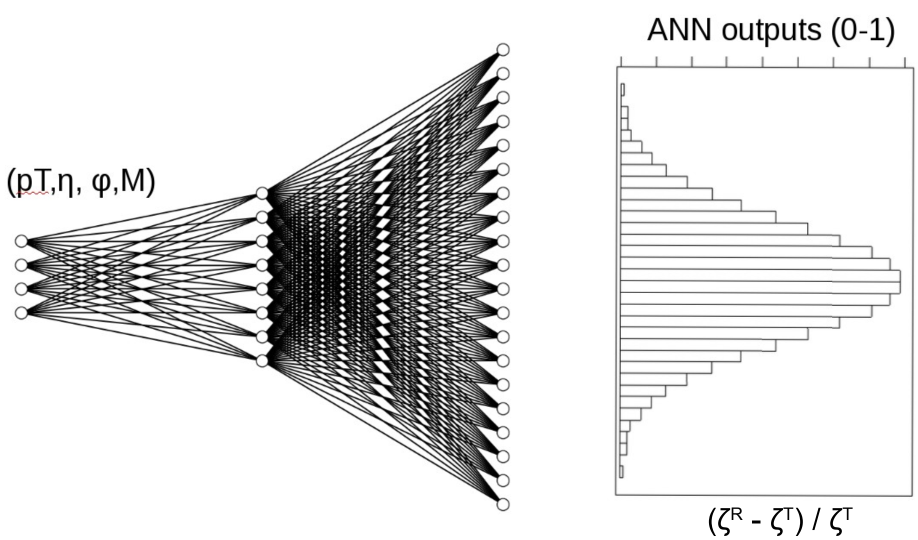


FIG. 1. A schematic representation of the NN architecture for modelling the detector response and affect of reconstruction algorithms on truth-level input variables. The output nodes of this NN represent a binned PDF for the resolution of single variable, e.g. $(p_{\mathrm{T}}^{\mathrm{reco}} - p_{\mathrm{T}}^{\mathrm{truth}})/p_{\mathrm{T}}^{\mathrm{truth}}$.

IV. MONTE CARLO SIMULATED EVENT SAMPLES

Monte Carlo events used for this analysis were produced using the Madgraph generator [4]. The simulated processes are a combination of equal event samples with top pair production ($t\bar{t}$) and photons produced in association with jets (γ+jets), which give a high rate of jets in different environments. Hadronic jets were reconstructed with the FastJet package [5] using the anti-k_t algorithm [6] with a distance parameter of 0.4. The detector simulation was performed with the Delphes package with a detector geometry which is similar to the ATLAS geometry. The event samples used for the following study are available from the HepSim database [7]. In this paper, only the transformation of p_{T} from truth jets (which have truth particle constituents) to reconstructed jets (which have calorimeter cell constituent) was performed, however the methodology should be object and parameter agnostic. Only truth jets which are matched to a reconstructed Delphes jet are considered in this study. For the matching criteria the reconstructed jet that has the smallest $\Delta R = \sqrt{\Delta\phi^2 + \Delta\eta^2}$, where $\Delta\phi = \phi^{\mathrm{truth}} - \phi^{\mathrm{reco}}$ and $\Delta\eta = \eta^{\mathrm{truth}} - \eta^{\mathrm{reco}}$, with respect to the truth jet is chosen. If this minimum ΔR is greater than 0.2, the truth jet is discarded. No other requirements are made on truth and reconstructed Delphes jets other than the $p_{\mathrm{T}} > 15$ GeV requirement made by Delphes. Only matched jets are used for this study since the aim of the study is to test whether an NN can learn changes in detector resolution as a function of kinematic properties of the jet (e.g. p_{T}, η, ϕ, m). The final number of training jets used is two million while 500,000 jets were used as an independent test sample. The distributions of quantities used as the input for the NN, p_{T}, η ϕ, m, are shown in Figure 2.

To facilitate gradient descent in all direction of the input variable space, the input variables are scaled to be in the range [0,1]. This avoids the p_{T} and the mass from having a disproportional affect on the training of the NN. The output variable, $(p_{\mathrm{T}}^{\mathrm{reco}} - p_{\mathrm{T}}^{\mathrm{truth}})/p_{\mathrm{T}}^{\mathrm{truth}}$, is also scaled to have values between 0 and 1. Only objects that are within the 1^{st} and 99^{th} percentile of the $(p_{\mathrm{T}}^{\mathrm{reco}} - p_{\mathrm{T}}^{\mathrm{truth}})/p_{\mathrm{T}}^{\mathrm{truth}}$ distribution are considered since objects outside this range are typically not used in physics analyses.

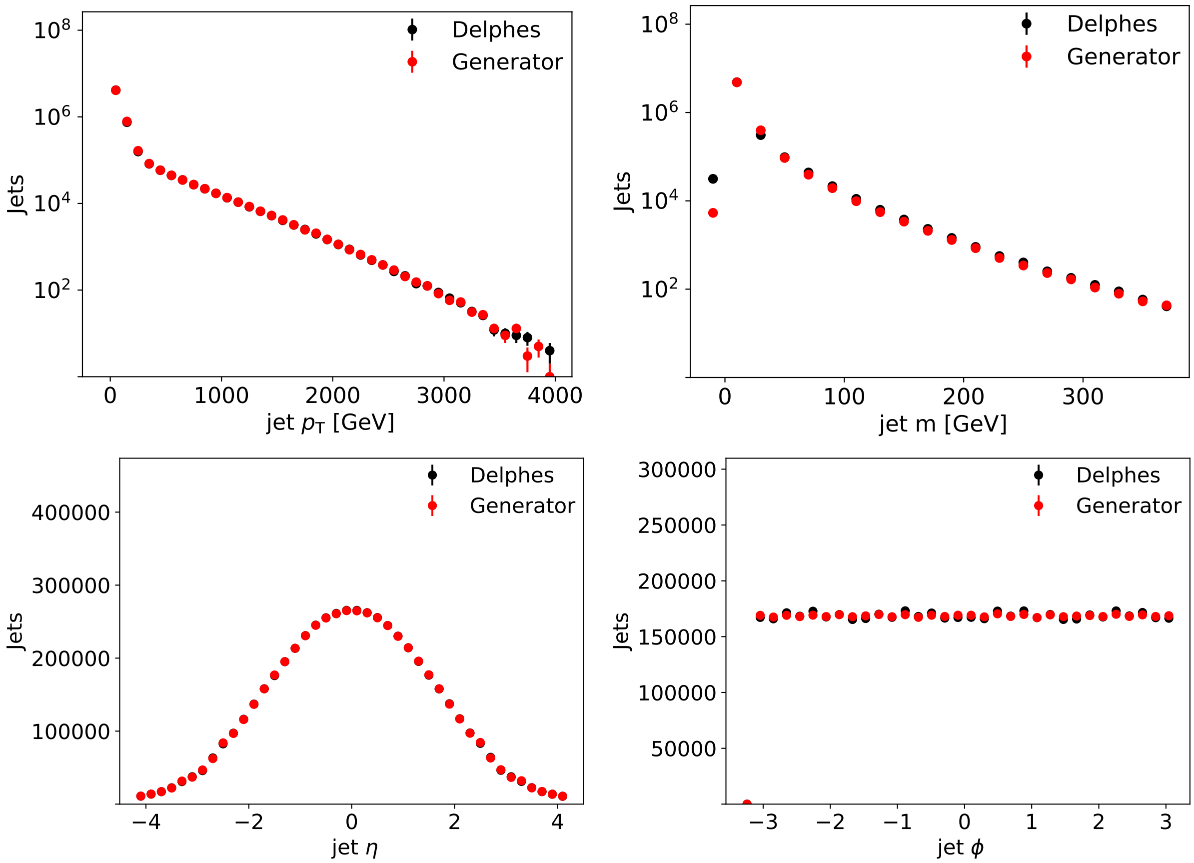


FIG. 2. Distributions for input variables for truth (red) and reco quantities (black).

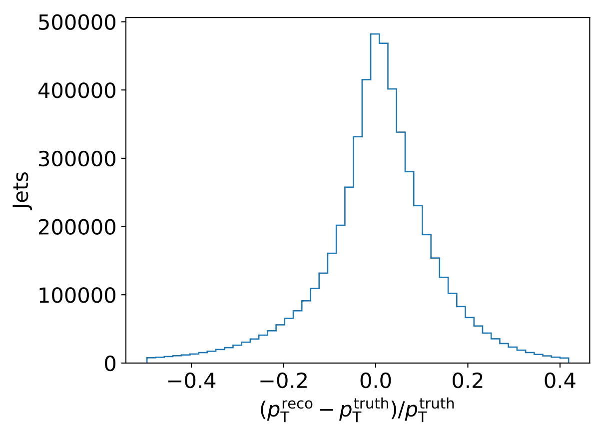


FIG. 3. Resolution of the p_{T} (relative differences between truth and reco p_{T}).

V. NEURAL NETWORK STRUCTURES

An NN is trained with four input parameters, the scaled p_{T}, η, ϕ, and m, and consist of five layers with 100 nodes each and with each node having a rectifier linear unit (ReLu) activation function. Several output layer configurations were tested including having 100, 200, 300, 400, and 500 output nodes, all with a softmax activation function. The 400 output nodes configuration resulted in the best performance, measured by how well the NN could mimic the Delphes p_{T} spectrum and resolution (see below for details), with the least number of total NN parameters.

In an attempt to optimize the NN training, several batch-size and number-of-epoch combinations were used in an attempt to optimize the sensitivity to a small subsample (the forward jets) of the training sample. The number of backpropagations (N_{bp}) were held constant by keeping the ratio of the number of epochs (N_e) and batch size (N_b) constant since $N_{bp} = \frac{N_t}{N_b} N_e$ where N_t is the number of training jets. Batch size and number of epochs of 5, 10, 20, 100, 200, 1000 were tested resulting in similar performance of the NN.

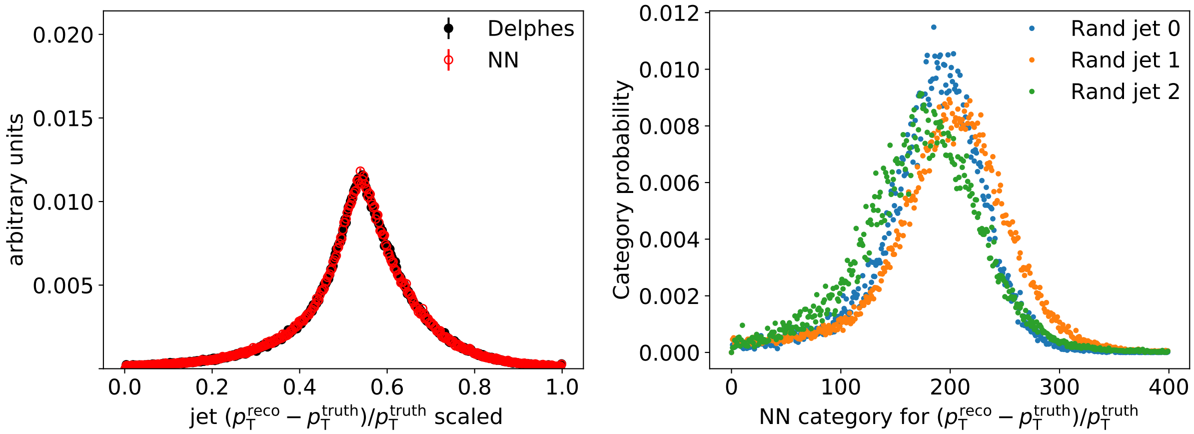
Finally, the NN is trained using the Adam [8] optimizer with a learning rate of 10^{-4} and is implemented using Keras with a TensorFlow [9] backend.

VI. RESULTS

After the NN has been trained to learn the PDF of $(p_{\mathrm{T}}^{\mathrm{reco}} - p_{\mathrm{T}}^{\mathrm{truth}})/p_{\mathrm{T}}^{\mathrm{truth}}$, the resulting learned PDF is compared to the Delphes PDF using the test sample in Figure 4a. Good agreement is observed between the Delphes and NN PDFs, showing that the NN has learned the bulk distribution.

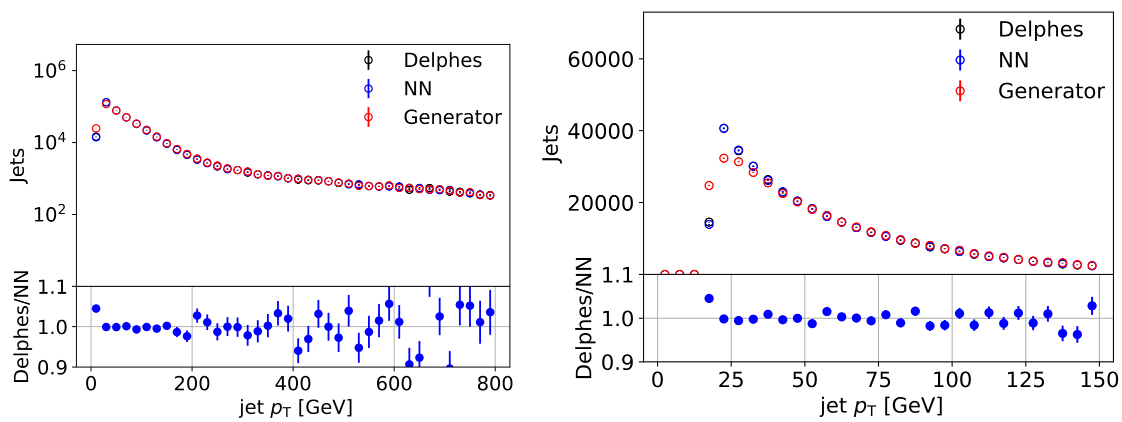
The NN output represents a binned PDF for each jet based on its input parameters (i.e. p_{T}, ϕ, η, and m). The PDFs for a set of three randomly selected jets are shown in Figure 4b which features shapes expected for typical resolution function with variations due to changes in jet input parameters. These PDFs are then randomly sampled to produce an NN jet that mimics the reco jet. A comparison of the NN-generated and Delphes jet p_{T} distribution for the test sample is shown in Figure 5. The NN reproduces the jet p_{T} distribution of Delphes within 5% for reconstructed jets with $p_{\mathrm{T}} > 20$ GeV.

To test whether the NN learned correlations between input parameters and the p_{T} resolution, the jets were divided into central ($|\eta| < 3.2$) and forward ($|\eta| > 3.2$) jets. The p_{T} resolution is then compared between the two regions for both the Delphes jets as well as the NN-generated jets. These two regions in the detector simulation have different calorimeter responses which results in different jet p_{T} resolutions in these two $|\eta|$ regions. The resulting



(a) (b)

FIG. 4. NN-generated jet $(p_T^{reco} - p_T^{truth})/p_T^{truth}$ compared to Delphes reco jet $(p_T^{reco} - p_T^{truth})/p_T^{truth}$ (a). Representative values of the NN output after training for three randomly selected truth jets which have different input values (b).



(a) (b)

FIG. 5. Delphes and NN-generated jet p_T disitributions for a wide (a) and narrow (b) p_T range.

resolutions for both regions are shown in Figure 6 using the training sample. The training sample was chosen for this comparison because forward jets make up a small subsample of all jets, as can be seen in Figure 2.

The mean and standard deviation of the resolution (shown, inclusively, in Figure 4) as a function of p_T is shown in Figure 7. The mean of the resolution for the NN is systematically higher than the resolution for Delphes but this effect is small when considering the width of the resolution. The standard deviation of the resolutions, however, are the same for the NN and Delphes across the p_T range showing that the NN accurately predicts the resolutions

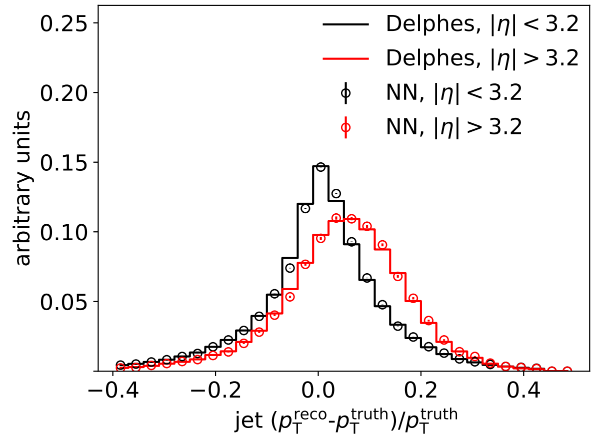
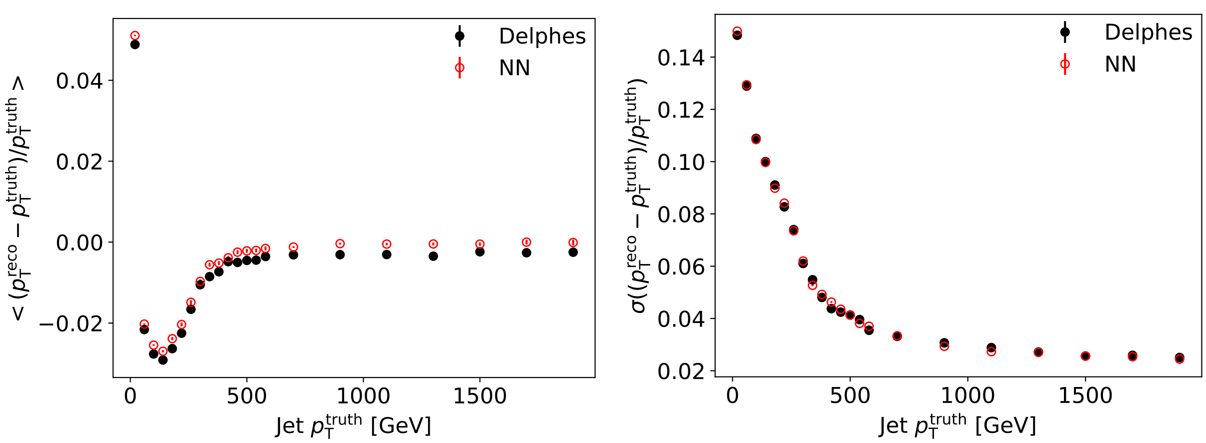


FIG. 6. Jet p_{T} resolution for the training sample for both the central and forward region.

for a large range in p_{T}.



(a) (b)

FIG. 7. The mean (a) and standard deviation of the jet p_{T} resolution for Delphes and NN-generated as a function truth jet p_{T}.

VII. CONCLUSION

A truth-level to reconstruction-level quantity transformation using a multi-categorizing NN is presented. This approach does not require the determination of analytic resolution functions since an NN can automatically learn the resolutions during the training procedure. The NN implementation presented effectively learned the truth-to-reconstruction transformation without requiring manual binning to capture the differences in resolutions of particular subsamples (i.e. central and forward jets). The automatic learning of correlations between the input variables and the resolution is one of the attractive features of using an ML-based transformation, allowing for rapid deployment of detector parametrizations.

Additional improvements could probably be made by including more information about the objects (e.g. whether a b-quark is present in a jet, kinematic information from other objects in the event) making this method more robust. This method should be easily extendable to additional reconstructed quantities and could be used to model the ATLAS and CMS detector. The method described in this paper allows for automated detector parametrization which can facilitate phenomological studies, efficient truth event selection, and upgrade studies.

ACKNOWLEDGMENTS

The submitted manuscript has been created by UChicago Argonne, LLC, Operator of Argonne National Laboratory ("Argonne"). Argonne, a U.S. Department of Energy Office of Science laboratory, is operated under Contract No. DE-AC02-06CH11357. The U.S. Government retains for itself, and others acting on its behalf, a paid-up nonexclusive, irrevocable worldwide license in said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the Government. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan. http://energy.gov/downloads/doe-public-access-plan. Argonne National Laboratory's work was funded by the U.S. Department of Energy, Office of High Energy Physics under contract DE-AC02-06CH11357.

[1] J. de Favereau, C. Delaere, P. Demin, A. Giammanco, V. Lemaître, A. Mertens, and M. Selvaggi. DELPHES 3, A modular framework for fast simulation of a generic collider experiment. *JHEP*, 02:057, 2014.

[2] François Chollet et al. Keras. https://keras.io, 2015.

[3] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit

Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[4] J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, et al. The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations. *JHEP*, 07:079, 2014.

[5] Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. FastJet User Manual. *Eur. Phys. J. C*, 72:1896, 2012.

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

[7] S.V. Chekanov. HepSim: a repository with predictions for high-energy physics experiments. *Advances in High Energy Physics*, 2015:136093, 2015. Available as `http://atlaswww.hep.anl.gov/hepsim/`.

[8] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv e-prints*, page arXiv:1412.6980, Dec 2014.

[9] Martín Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.