ANL-HEP-158616

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: March 30, 2020)

Abstract

Rapidly applying the effects of detector response 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 transformations and can capture complex correlation 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 final-state particles (hadrons, leptons, photons) with properties (four momenta, flavor) which entirely depend on the physics processes occurring. These particles, as well as more complex derived objects (such as jets), 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 high-energy collider 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). Producing an approximation of the detector in Delphes is a laborious process as developers must manually define resolution functions based on additional studies that involve analyses of experimental data or complex detector simulations used by experiments. An approximation of the detector responses and experimental object reconstruction can, however, also be performed with machine learning (ML) by training a neural network (NN) using the Geant4-based simulations that have gone through an experiment's reconstruction algorithm. Such an NN could then automatically learn the transformation functions from truth MC objects (jets and particles) to objects modified by a detector and experimental reconstruction algorithms.

The main advantage of a detector parametrization based on ML, as compared to a manually-constructed analytic 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 learn the transformation from the truth to the reco quantities without dedicated studies of resolution functions and could automatically be updated when there are changes to the detector or the reconstructions algorithms. Another advantage is that the NN approach can introduce a complex interdependence of variables which is currently difficult to implement in parameterized simulations.

The organization of this paper is as follows. In Section II, the traditional method of parametrizing detector and reconstruction effects are described. Section III contains a description of the samples used to train the NN. The details of the NNs structure and parameters are reported in Sections IV. Finally, the results from the trained NN are presented in Section V.

II. TRADITIONAL PARAMETERIZED FAST SIMULATIONS

In abstract terms, a typical variable ξ_i^{reco} that characterizes a reconstructed particle/jet, such as transverse momentum (${p_{\mathrm{T}}}^{\mathrm{reco}}$) or pseudorapidity ($\eta^{\mathrm{reco}}$), can be viewed as the result of a multivariate transform, F, of the original variable ξ_1^{truth} at truth level:

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

Generally, such a transform depends on several other variables ξ_2^{truth} .. $\xi_{\mathrm{N}}^{\mathrm{truth}}$ characterizing this (or other) objects at truth level. For example, the extent at which jet transverse momentum, p_{T}, is modified by a detector depends on the original truth-level transverse momentum ($\xi_1^{\mathrm{truth}} = p_T^{\mathrm{truth}}$), pseudorapidity ($\xi_2^{\mathrm{truth}} = \eta^{\mathrm{truth}}$), and other effects that can be inferred from truth quantities. Similarly, if particular detector regions in the azimuthal angle (ϕ) have low efficiency, 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 $N_{\mathrm{b}}^{\mathrm{k}}$, where k=1,2,3,... is the number of correlated variables describing truth-level objects and N_{b} is the number of histogram bins for the distributions $(\xi^{\mathrm{reco}} - \xi^{\mathrm{truth}})/\xi^{\mathrm{truth}}$, 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 $N_{\mathrm{b}}^{\mathrm{k}}$ since each object at the truth level should be placed inside the grid defined by N_{b} bins. Therefore, complex parameterisations of resolutions and efficiencies for $N > 2$ becomes computationally intensive.

III. MONTE CARLO SIMULATED EVENT SAMPLES

Monte Carlo events used for this analysis were produced using the Madgraph generator [2]. 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 [3] using the anti-k_t algorithm [4] 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 [5]. 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 $\Delta R \equiv \sqrt{\Delta\phi^2 + \Delta\eta^2}$, where $\Delta\phi \equiv \phi^{\mathrm{truth}} - \phi^{\mathrm{reco}}$ and $\Delta\eta \equiv \eta^{\mathrm{truth}} - \eta^{\mathrm{reco}}$, 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 in Delphes. Only matched jets are used 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 1.

To facilitate training and to smooth the hypersurface formed by the input variables, 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 jets that are within the 1st and 99th percentile of the $(p_{\mathrm{T}}^{\mathrm{reco}} - p_{\mathrm{T}}^{\mathrm{truth}})/p_{\mathrm{T}}^{\mathrm{truth}}$ distribution are considered.

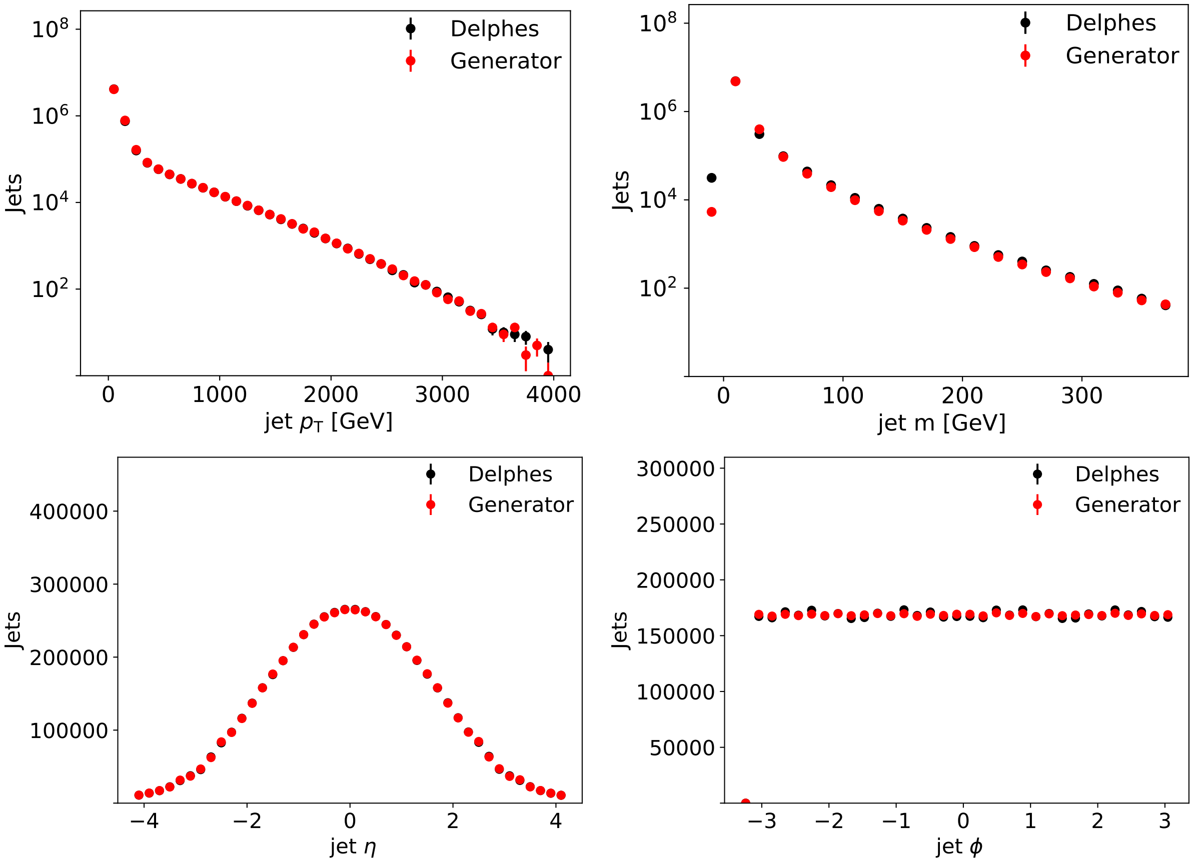


FIG. 1. Distributions for input variables for truth (red) and reco quantities (black).

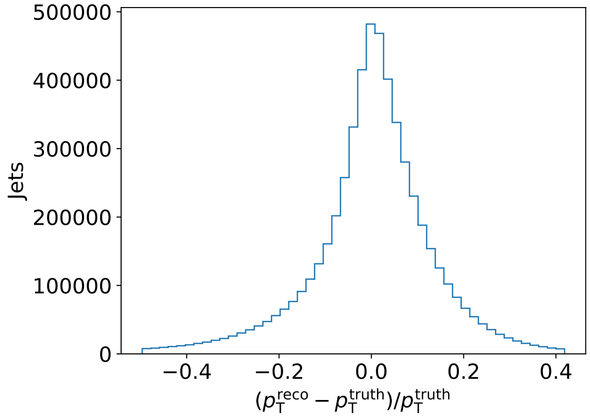


FIG. 2. Resolution of the p_T (relative differences between truth and reco p_T).

IV. 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 can consist of any variable that can influence the resolution of a jet, such as jet flavor at the truth level, jet radius, etc. Figure 3 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.

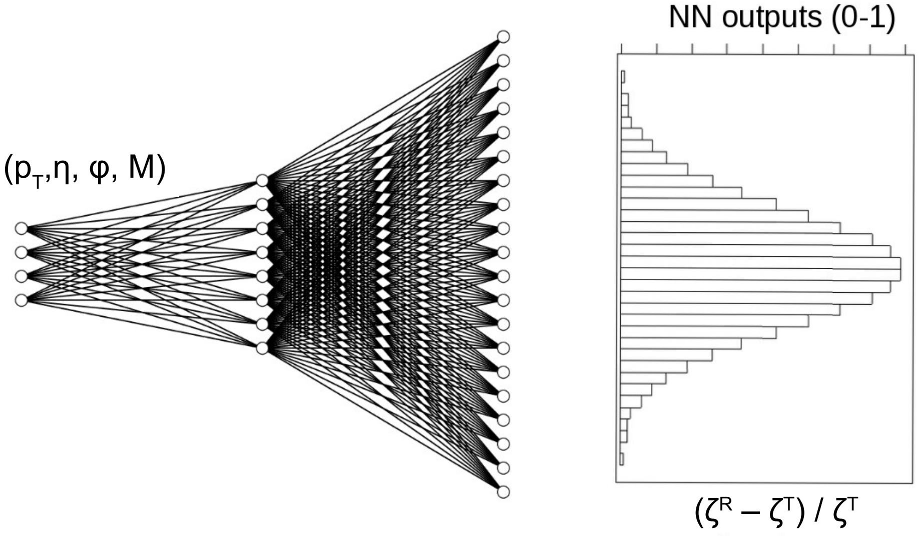


FIG. 3. A schematic representation of the NN architecture for modelling the detector response and effects of reconstruction algorithms on truth-level input variables. The output nodes of this NN represent a binned PDF for the resolution of single variable, e.g. $(p_{\mathrm{T}}^{\mathrm{reco}} - p_{\mathrm{T}}^{\mathrm{truth}})/p_{\mathrm{T}}^{\mathrm{truth}}$, while other variables, such as η, ϕ, mass (m), are auxiliary variables that affect $(p_{\mathrm{T}}^{\mathrm{reco}} - p_{\mathrm{T}}^{\mathrm{truth}})/p_{\mathrm{T}}^{\mathrm{truth}}$.

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. As far as the authors are aware, this is the first attempt in using an NN to learn the transformation of truth physics objects into reco objects.

An NN is trained with four input parameters, the scaled jet p_{T}, η, ϕ, and m. The NN consists of five fully connected layers with 100 nodes each, with each node having a rectifier linear unit (ReLu) activation function. The output layer has 400 nodes and a softmax activation function. The NN is trained using the Adam [6] optimizer with a learning rate of 10^{-4} and is implemented using Keras with a TensorFlow [7] backend. Finally, categorical cross-entropy is used as the loss function.

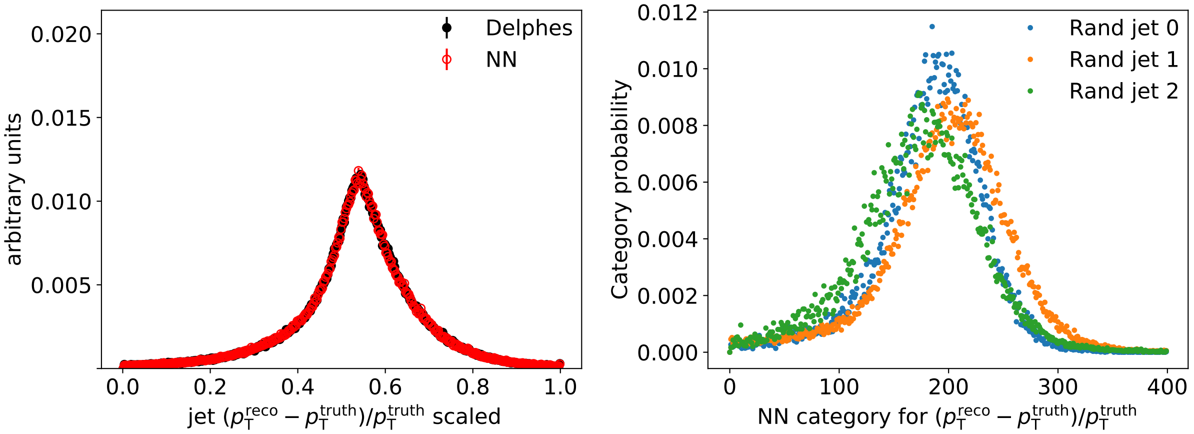
To optimize the NN, several output layer configurations are tested by varying the number of output nodes from 50 to 500, all with a softmax activation function. The configuration with 400 output nodes resulted in the best performance, measured by how well the NN could mimic the Delphes p_{T} spectrum and resolution (see below for details), with the least number of total NN parameters. The figure of merit for choosing the number of output nodes was the maximal deviation of the ratio between the p_{T} spectrum produced by the NN and the p_{T} spectrum produced by Delphes (see the bottom panel of Figure 5 for the final ratios). The configuration with the smallest maximal deviation with the least number of nodes was chosen (the deviation didn't improve when going to 500 nodes).

Additionally, for the NN training, several batch-size and number-of-epoch combinations are probed to test whether these parameters cause a change in sensitivity to small subsamples (specifically, forward jets) of the training sample. Changing the batch size without modifying the number of epochs would result in a change in the total number updates to the NN parameters during back propagation. To avoid this, the number of backpropagations (N_{bp}) are 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. The performance of the resulting NNs is evaluated by comparing the Kolmogorov-Smirnov (KS) test statistic for the $(p_{\mathrm{T}}^{\mathrm{reco}} - p_{\mathrm{T}}^{\mathrm{truth}})/p_{\mathrm{T}}^{\mathrm{truth}}$ of forward jets in Delphes and for the NN-produced jets. Batch size and number of epochs of 5, 10, 20, 100, 200, 1000 are tested resulting negligible differences in the KS test statistic for the different NNs.

Finally, the loss function for both the training (2 million jets) and test sample (500,000 jets) is evaluated for each epoch. The values of the loss function for the training and testing sample are found to be within 1% of each other at the end of training.

V. 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 shape of the distribution.



(a)

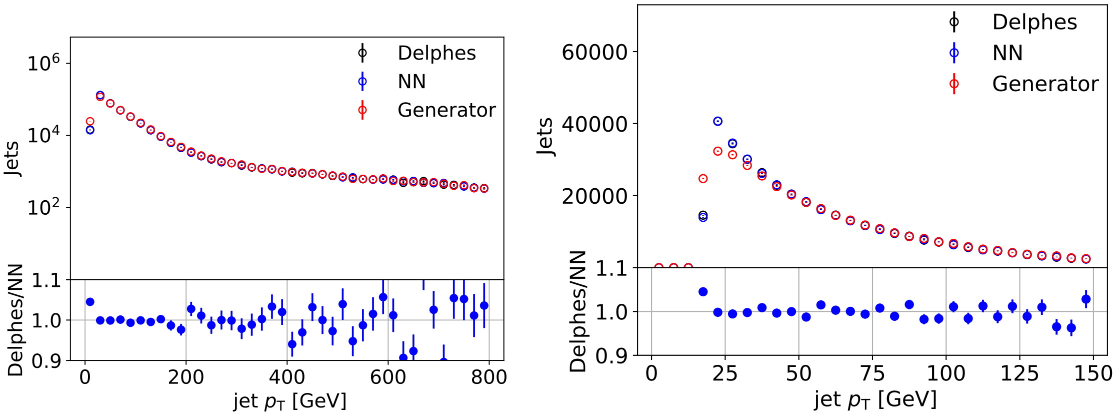
(b)

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

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 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 1.

The mean and standard deviation of the resolution (demonstrated in Figure 4) as a



(a)

(b)

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

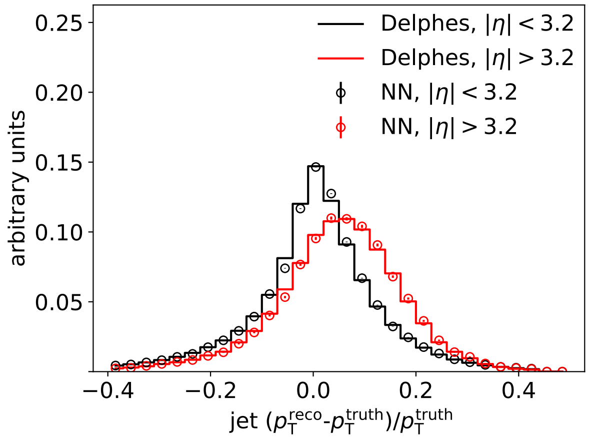


FIG. 6. Jet p_T resolution for the training sample for both the central and forward region.

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. This difference between the means has a small effect on truth-to-reco transformation due to its small size relative to 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 for a large range in p_T.

In order to produce an NN with optimal performance and a minimal amount of parameters, a genetic algorithm (GA) is used to optimize the NN hyperparameters. GAs are commonly used for NN hyperparameter optimizations [8] but are not commonly used in high energy physics. The number of layers, number of neurons in each layer, and choice of

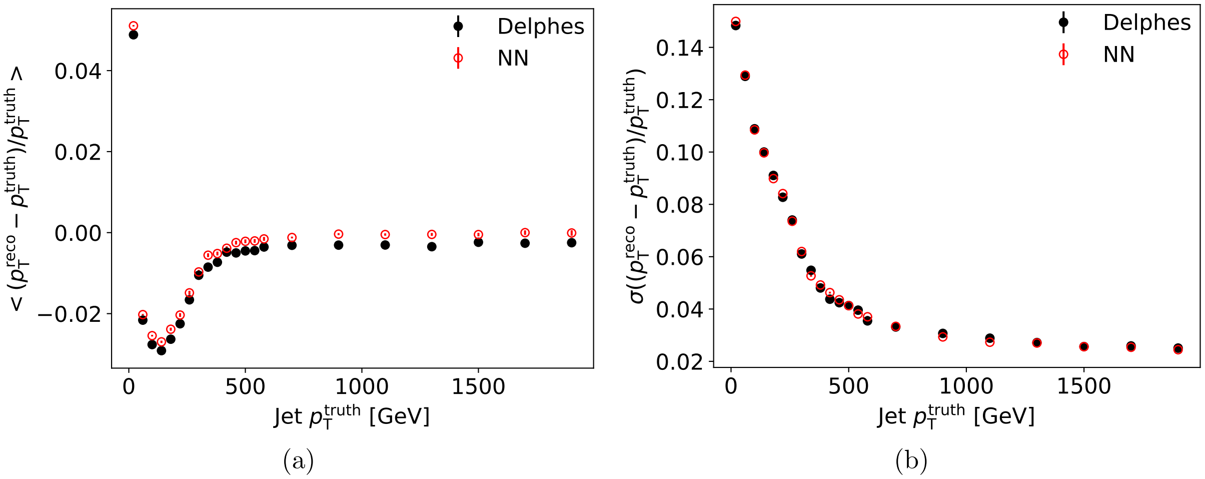


FIG. 7. The mean (a) and standard deviation of the jet p_T resolution for Delphes and NN-generated as a function truth jet p_T.

the learning rate are scanned to find the optimal configuration. The GA utilized is an evolutionary algorithm that mimics the process of natural selection and which was previously used in the determination of the parameters of a complex force field [9, 10]. During the optimization, the sets of parameters are ranked by their value of categorical accuracy. The final set of parameters selected by the GA had similar performance to the initial configuration. This is expected due to the small scale of our problem and because Delphes using Gaussian approximations of the resolution functions. The GA is tested in this scenario to prepare it for future studies where an NN will be optimized to learn non-linear and non-gaussian resolution functions typically seen in Geant4 simulations.

VI. CONCLUSION

A truth-level to reconstruction-level transformation using an NN classifier 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 jet p_T distribution produced by the NN is within 5% of the p_T distribution from Delphes for jets with $15 < p_T < 400$ GeV. The NN also 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). Finally, the NN was able to learn the both the mean and width of the resolution over a wide range of jet p_T(15 GeV to 2 TeV).

This automatic learning of correlations between the input variables (i.e. p_T and η) and the resolution is one of the attractive features of using an ML-based transformation, allowing for rapid deployment of detector parametrizations.

This method can be extended and improved 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) potentially making this method more robust. This method could be extendable to additional reconstructed quantities and could be used to automatically model the ATLAS and CMS detectors. Another improvement left for future studies is the use of mixture density networks [11] to estimate resolution functions as a combination of Gaussians. The method described in this paper allows for automated detector parametrization which can facilitate phenomological studies, efficient truth event selection, and upgrade studies.

ACKNOWLEDGMENTS

We gratefully acknowledge the computing resources provided on a high-performance computing cluster operated by the Laboratory Computing Resource Center at Argonne National Laboratory. 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] 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.

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

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

[5] 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/.

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

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

[8] James S. Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2546–2554. Curran Associates, Inc., 2011.

[9] Ying Li, Hui Li, Frank C. Pickard, Badri Narayanan, Fatih G. Sen, Maria K. Y. Chan, Subramanian K. R. S. Sankaranarayanan, Bernard R. Brooks, and Benoît Roux. Machine learning force field parameters from ab initio data. *Journal of Chemical Theory and Computation*, 13(9):4492–4503, 2017. PMID: 28800233.

[10] Md Mahbubul Islam, Grigory Kolesov, Toon Verstraelen, Efthimios Kaxiras, and Adri C. T. van Duin. ereaxff: A pseudoclassical treatment of explicit electrons within reactive force field simulations. *Journal of Chemical Theory and Computation*, 12(8):3463–3472, 2016. PMID: 27399177.

[11] Christopher M. Bishop. Mixture density networks. Technical report, 1994.