BUFF: Boosted Decision Tree based Ultra-Fast Flow matching

Cheng Jiang*

School of Physics and Astronomy, University of Edinburgh, EH9 3FD, Edinburgh, United Kingdom

Sitian Qian†

School of Physics, Peking University, 100871, Beijing, China

Huilin Qu‡

CERN, EP Department, CH-1121 Geneva 23, Switzerland

(Dated: April 30, 2024)

Tabular data stands out as one of the most frequently encountered types in high energy physics. Unlike commonly homogeneous data such as pixelated images, simulating high-dimensional tabular data and accurately capturing their correlations are often quite challenging, even with the most advanced architectures. Based on the findings that tree-based models surpass the performance of deep learning models for tasks specific to tabular data, we adopt the very recent generative modeling class named conditional flow matching and employ different techniques to integrate the usage of Gradient Boosted Trees. The performances are evaluated for various tasks on different analysis level with several public datasets. We demonstrate the training and inference time of most high-level simulation tasks can achieve speedup by orders of magnitude. The application can be extended to low-level feature simulation and conditioned generations with competitive performance.

I. INTRODUCTION

The collisions happening at the Large Hadron Collider (LHC) [1], following the nature of quantum mechanics, are intrinsically probabilistic. This necessitates a vast number of events to acquire meaningful information that can validate theoretical predictions. Oftentimes, we rely on the large-scale Monte Carlo (MC) event production [2] with a dedicated, and probably lengthy production pipeline to perform complicated calculations of the Standard Model. In the upcoming High-Luminosity LHC (HL-LHC) era [3], it is expected that the trend for these simulations will shift towards a more CPU-intensive environment [4], given the much larger amount of data-taking. It is crucial to prevent the possible limitation in physics output due to insufficient statistics arising from constraint computational budget.

The successful application of deep generative models as fast surrogate models in collider physics allows for the rapid generation of particle collision data, drastically reducing event generation period without sacrificing accuracy. Diffusion models (DM) [5–7] excels in generating high-quality, detailed samples by iteratively denoising random noise into clean output, showcasing impressive capabilities in tasks like fast calorimeter simulation [8–13]. One common challenge of those DM applications is that they only reach the ideal performance after relatively long generation time. Many studies adopt the novel distillation techniques with pre-trained model such as progressive distillation and consistency model [14–16].

On the other hand, normalizing flows (NFs) [17, 18] stand out for their ability to model complex distributions with a small amount of time, making them especially powerful in fields requiring precise probability density estimation. Different generative models have emerged in various applications [19–23]. For the MC event generator, several applications focus on conditional phase space sampling [24–30]. Flow-based model could also produce good quality of shower rapidly [31–35]. Those models are applied extensively to handle tasks such as unfolding [36, 37], shower shape correction, anomaly detection [38–41] and quark gluon tagging [42].

Whereas, NF intrinsically can not reach desirable performance once moving to higher dimensions [43, 44]. Furthermore, most architectures adopted into high energy physics (HEP) are tailored for image synthesis. The majority of quantities or variables needed at the analysis level of event content are designed to be sensitive to certain phase spaces in one channel or aim to explore the deeper structure of hadronic objects. These variables have shapes far more complex than a simple Gaussian and are often treated as tabular data. Current approaches, which utilize the most advanced architectures to simulate tabular data in high dimensions and precisely capture their correlations, can be notably challenging, especially when considering the significantly longer training and validation times required for each update in analysis.

Recently, a new type of generative modelling class was introduced, inspired from the score matching diffusion model [5] and optimal transport [45], conditional flow matching (CFM) [46] offers a simulation-free approach to directly match the vector field, demonstrating good scalability to very high dimensions. In this work, we used several techniques mentioned from the study [47] to replace the traditional Neural Network backbone CFM with Gradient Boosted Trees (GBT) [48, 49]. We named

---

* C.Jiang-19@sms.ed.ac.uk
† stqian@pku.edu.cn
‡ huilin.qu@cern.ch

the framework Boosted Decision Tree based Ultra-Fast Flow matching (BUFF) with the model flowBDT. The reasonale behind using tree-based CFM is that the vector fields and conditional probability paths generated from N-dimensional tabular data should also hold the similar structures inherent to them. Several comprehensive studies [50, 51] indicate that tree-based models surpass most of deep learning models in performance for tasks specific to tabular data, such as classification and regression. In Section II, we delve into the workings of the Tree-based Conditional Flow Matching and share some improved understanding of this method, alongside its novel applications. Section III describes the datasets utilized in our study. In Section IV, we present the performance outcomes across various tasks using diverse datasets and a range of comprehensive metrics. Lastly, Section V offers potential applications and future directions for this line of research.

II. IMPROVED UNDERSTANDING FOR TREE-BASED FLOW MATCHING

Suppose $f : \mathbb{R}^d \to \mathbb{R}^d$ is a continuous differentiable map between target distribution p_1 and base distribution $x \sim q_0$, which usually is multidimensional Gaussian. Normalizing flow (NF) requires each composition of the inverse map $f(x) = f_{T-1} \circ ... \circ f_0(x)$ to have valid Jacobian determinants during sampling phase. By the change-of-variable rules, the log-likelihood loss for NF training is written as [43]:

$$
\begin{aligned}
p_1(x_T) &= q_0(f^{-1}(x_T)) \, |\det[\frac{\partial f^{-1}}{\partial x_t}(x_T)]| \\
&= \frac{q_0(x)}{|\det[\frac{\partial f}{\partial x}(x)]|}, \quad x = f^{-1}(x_T)
\end{aligned}
\tag{1a}
$$

$$
\log p_1(x_T) = \log q_0(x) - \sum_{t=0}^{T} \log |\det[\frac{\partial f_t}{\partial x_t}]| \tag{1b}
$$

From the formulation for the training loss, there are two drawbacks for transitional NF: the strict requirement for invertible map f and inefficient Jacobican determinant calculation. These would make NF predicts worse when scaling up to very high dimensions. Later, continuous normalizing flow (CNF) is proposed to replace the second term in Equation.2 using the property that the trace of the Jacobian matrix is equivalent to the log probability of the continuous random variable:

$$
\log p_1(x_T) = \log q_0(x) - \int_{t=0}^{T} \text{Tr}(\frac{\partial f}{\partial x_t}) dt \tag{2}
$$

This allows CNF to have a more efficient training stage given trace is generally easier to solve. However, the

vector field from a flow is still integrated over time (i.e. $\frac{\partial x}{\partial t} = f(x_t, t)$). The computational demanding training and bad scalability problems insist in these simulation-based approaches.

Flow matching proposes a simulation-free loss function which directly matches the probability path $p_t(x)$ and vector field $u_t(x)$ for time steps t uniformly distributed between 0 and 1. For any learnable vector field $v_t(x)$ parameter θ from a model with stochastic regression objective, the loss defined as [44]:

$$
\mathcal{L}_{FM}(\theta) = \mathbb{E}_{t,p_t(x)} ||v_\theta(t, x) - u_t(x)||^2 \tag{3}
$$

To compute this loss function needs the explicit knowledge of the target vector field u_t. One could use the conditioning variable z to define the CFM objective [46]:

$$
p_t(x) = \int p_t(x|z)q(z) \mathrm{d}z \tag{4a}
$$

$$
\mathcal{L}_{FM}(\theta) = \mathbb{E}_{t,q(z),p_t(x|z)} ||v_\theta(t, x) - u_t(x|z)||^2 \tag{4b}
$$

Both loss functions should hold the same gradient w.r.t. weights θ.

There exists a specific, non-universal scenario in which GBT can be utilized here. Since GBT classes are non-differentiable, they cannot be directly applied in generative models such as Generative adversarial networks [52] or variational autoencoders [53], which rely on composing multiple models during training. In typical neural network-based flow matching, we employ Stochastic Gradient Descent (SGD) [54] with random sampling to minimize the expectations from Equation.4 and Equation.6 over a mini-batch. Additionally, we adapt the setup for GBT to 'mimic' batch training as practiced in deep learning models. The data points are duplicated repeatedly to approximate the expectation calculated over all possible pairs between input and target sample. We use the linear interpolation and independent conditional flow matching to build the training dataset of vector fields at each noise level, there will be different interpolant class for specific objectives [46, 55]. A small individual GBT model with 3-4 depth and 50-100 estimators is used on each time step. The training time is directly proportional to both the tree depth and number of estimators in one GBT model. Compared with other deep learning model, the extremely fast training and inference time of those light GBT model can provide a robust performance with only fraction of time for high-level physics task.

Since different models are trained for each noise level, the number of time steps during inference is fixed. Nevertheless, we retain the flexibility to adjust the sampler for solving ODE. The crucial aspect that needs investigation is whether those different models can predict with equivalent accuracy for ODE calculations with high-order numerical methods across various time steps. In our research, we modify the original setup: instead of employing the first-order Euler solver, we evaluate the performance using the second-order Midpoint solver, as well

as higher-order solvers such as Runge-Kutta [56], Tsitouras, and Dormand-Prince [57]. The solvers that best fit our tasks are identified as the Midpoint solver at 30-40 steps and the Dormand-Prince solver at 15-30 steps. Consequently, we have selected these two solvers as our defaults. In addition to this, we find that varying the number of output on each tree of the model significantly reduces the sampling time. We keep those hyperparameters small enough to achieve decent performance with just negligible time typically required.

III. DATASET

A. JetNet

Around 200,000 events of *JetNet* dataset [58] are simulated at leading order using `Madgraph` [59]. Each of them focus on a relatively narrow kinematic range so that relative transverse momenta of the generated parton and gauge boson are within a small window. Those parton-level events undergo the hadronization process via `Pythia` [60]. The transverse momentum (p^T) of particle jets is around 1 TeV. Jets are clustered by anti$-k_T$ algorithm with a cone size of $R = 0.4$.

The first 30 particles with highest p^T inside the jets are taken into account. Four particle-level variables are available: the relative location w.r.t. jets on angular coordinate frame: $\eta^{rel} = \eta^{particle} - \eta^{jet}$, mod $(\phi^{rel} = \phi^{particle} - \phi^{jet}, 2\pi)$, relative transverse momentum of the particle defined by $p_T^{rel} = p_T^{particle}/p_T^{jet}$. To distinguish the jets with fewer than 30 particles, fourth variable is tailored to tell if the particle is genuine or zero-padded.

Different types of jets are provided in the dataset, either from gluons, light or top quarks, or vector bosons. The gluon and light quarks usually have simple topology. We choose top quark jets to derive the high level features as they should have the complex topology to capture the inner correlations among subjets and structural distributions for the global point cloud.

B. CaloChallenge

The Fast Calorimeter Simulation Challenge 2022 [61] aims to create and evaluate a rapid and highly accurate calorimeter simulation by benchmarking deep learning methods. The datasets use the GEANT4 [62] framework to simulate different kinds of particle showers inside either a irregular or concentric calorimeter.

In each dataset, showers are simulated to travel along the z-axis within a cylindrical detector. Cells in the radial r and angular α bins are arranged on each sampling layer of the calorimeter. *CaloChallenge* dataset 1 contains photon shower simulated by `Geant4` with 368 voxels on 5 different calorimeter layers. Layer 1 and 2 are designed to have more granular layout than other three layers. By assigning the different voxel energies and locations according to the specific r and α bins, one can reconstruct the total deposited energy of the shower for the particle with incident energies from 256 MeV up to 4 TeV. Even though the other two *CaloChallenge* datasets are more granular than the first dataset, dataset 1 provides a better shower description within irregular geometry, and also a smoother translation to tabular data.

C. Jet Datasets for Unfolding

The dataset used for unfolding task as described in Section. IV C is from [63]. It contains QCD jets from Z + jets events that are simulated by either `Herwig` [64] or `Pythia` with different tunes and parton distribution functions. The jets are clustered by anti$-k_T$ algorithm within $R = 0.4$. We consider event to have the leading jet with a Z boson $p^T > 200$ GeV to make the event topology unaffected by the limited acceptance. Multiple datasets is required to perform the unfolding between full particle phase spaces and particle flow objects. The detector distortions are simulated by fast detector simulator of the CMS detector through `Delphes` [65] with concrete particle flow reconstruction. Six different quantities derived from the substructure of the leading jet are used to evaluate the performance.

D. Schrödinger Bridge Refinement

Schrödinger Bridge Refinement involves two distinct datasets of electron particles traveling through an electromagnetic calorimeter. The conditional input dataset `GFlash` [66–68] approximates electron showers using a parameterized fast simulation. The formulation is based on the knowledge of longitudinal and radial showers in the sampling calorimeter. We keep the Moliére radii and radiation length as their default values. Additionally, the `GFlash` procedures account for the effects of sampling geometry and materials on shower shapes.

Refining showers from the fast simulation to match the full simulation target from `Geant4` requires an injective mapping between them. Therefore, we ensure that the incident particle energies of both simulations remain the same for each training event. The electron shower has a regular size of 10×10 calorimeter cells with fixed incident energies of 50 GeV. The core idea for this task is to use the approximate shower as input conditions which should contain more comprehensive information about geometry and material.

IV. TASKS

Based on the new design and knowledge improvement of the model, various potential HEP applications will be shown in this section. We first demonstrate the baseline
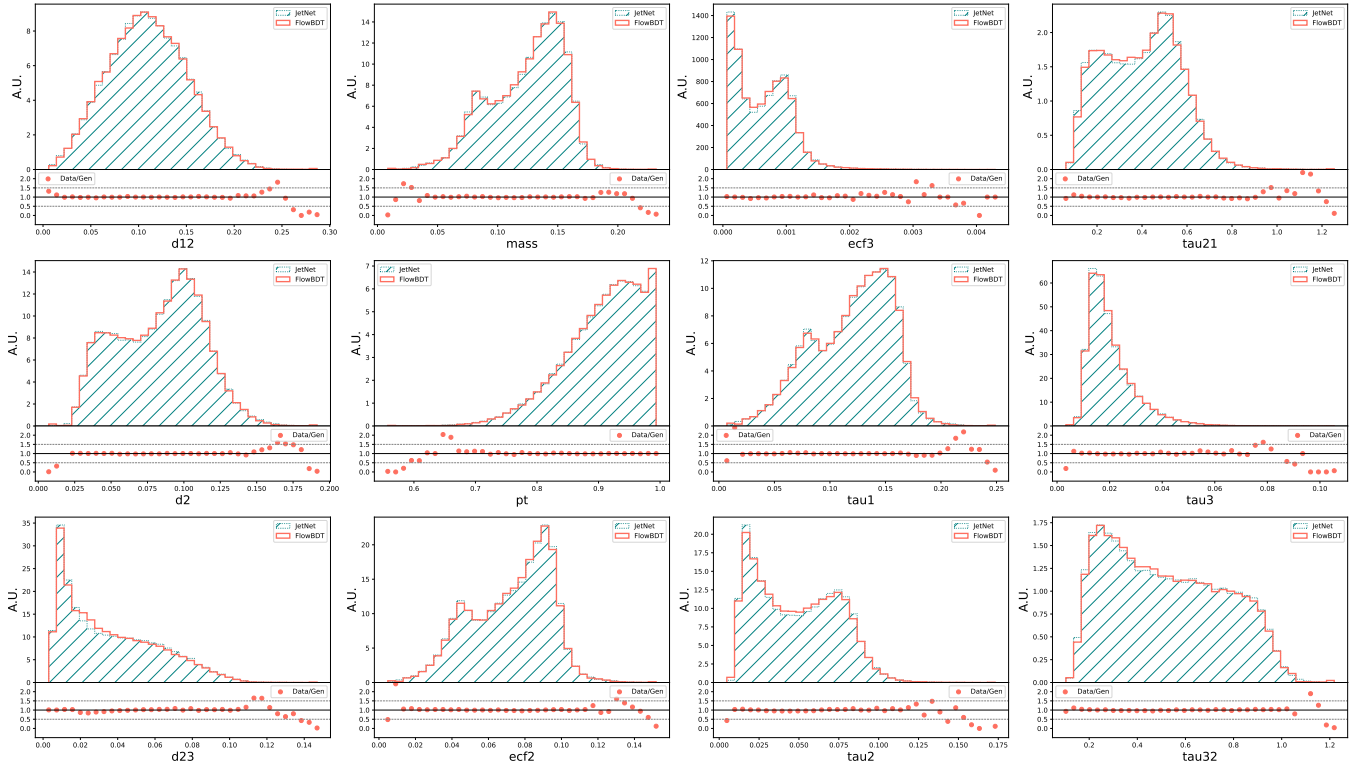
FIG. 1. The histogram and ratio plots for generated simultanously and original jet variables. Shaded: from JetNet, solid line: from flowBDT

performance task for end to end fast simulation as done with traditional normalizing flow [69] in section IV A. Then, section IV B shows the advantages of flow matching and GBT model by moving to a much higher dimension regime. flowBDT can still give decent performance for simulating hundreds of cells within irregular geometry and particles inside jets. Section IV C illustrates the model is also fully capable to do conditioned generation and distribution mapping from non-tractable prior distribution.

A. End to end fast simulation

Most of the high level variables used in analysis-level are derived via a long simulation chain. Those variables are chosen among a larger subset of variables after digitization and reconstruction. Having sufficient statistics for those high-level variables is crucial for many analyses, while deriving those quantities starting from the generator level would be computationally demanding. The possible approach is to directly generate those high-level observables according to the information from full simulation. Skipping those intermediate steps would reduce the simulation time significantly.

Whereas many variables used in the analysis have complicated distribution, the generative model has to learn to simulate a N dimensional distribution far away from

pure Gaussian form. Capturing the intricate shapes and correlations of these distributions poses a considerable challenge. We studied the specific case by simulating four vector, substructure, and correlation function variables of highly boosted objects such as top jets. The total number of 12 variables calculated from 3 subjets of pseudo-jets by *JetNet* are considered, specifically, The transverse momentum and mass of the jet p^{T}, m, the N-subjetiness for exclusive n-jet τ_1, τ_2, τ_3, the N-subjetiness ratio $\tau_{21} = \frac{\tau_2}{\tau_1}$, $\tau_{32} = \frac{\tau_3}{\tau_2}$, Energy splitting function defined by the splitting scale for n-jet exclusive clustering d_{12}, d_{23}, n$^{\mathrm{th}}$ order Energy correlation function ecf_2, ecf_3 calculated from p^{T} and ΔR, finally $d_2 = \frac{ecf_3 \sum p^{\mathrm{T}}}{ecf_2^2}$.

The Dormand-Prince and Midpoint solver within 20-30 steps could already give satisfactory results for generating 100k events with all variables. Fig.1 suggests most of variables with complicated shapes have great agreement between target and generated sample. The best performance on histogram is seen for substructures and functions related to first two subjets, especially distributions with two asymmetric peaks. Despite a slight misalignment in the histograms of the last subjet, flowBDT can efficiently handle well for the tasks around 10-20 dimensions with both few-steps low and high order ODE solver.

The evaluation metrics for binned and unbinned distribution for those variables are also provided in Table. I. Almost all the features sampled from the model give the
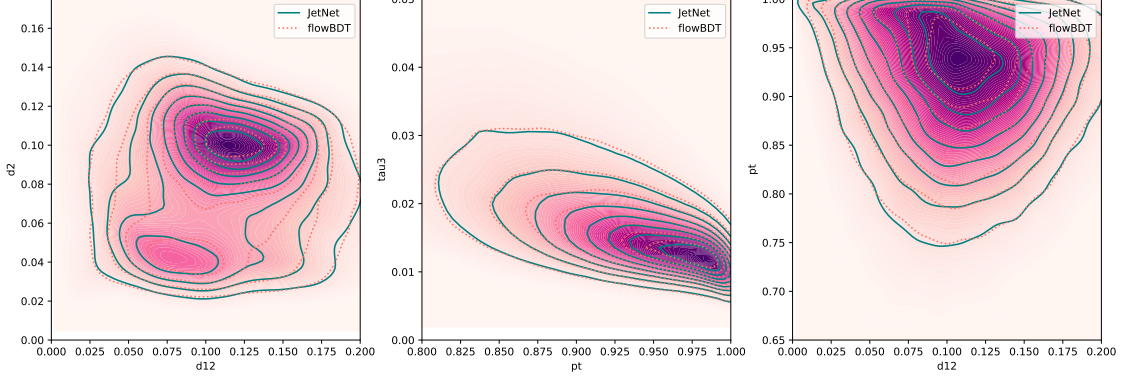
FIG. 2. The kernel density estimation contour plot for generated and original 100k jet variables. Solid: from JetNet, dotted: from flowBDT

Metrics	Sep power×100	EMD×10
d_{12}	0.059	0.0030
d_2	0.14	0.0037
d_{23}	0.0057	0.095
τ_1	0.096	0.042
τ_2	0.15	0.0070
τ_3	0.065	0.009
$mass$	0.086	0.0030
p^{T}	0.051	0.0085
τ_{21}	0.045	0.020
ecf_2	0.084	0.0027
ecf_3	0.0001	0.15
τ_{32}	0.0097	0.031

TABLE I. Evaluation metrics for samples generated from flowBDT. The separation power is calculated from the difference of binned histograms between generated and original sample. Earth Mover's Distance, or Wasserstein Distance is derived from unbinned distributions.

impressive performance on f-divergence metric (i.e. triangular discriminator [70, 71]). The Earth Mover's Distance [72, 73] maintains very low value for majority of generated features on unbinned distribution listed on Table. I. To do a comprehensive evaluation, variable simulated from our model should also have good pairwise correlation. We compare the correlation of generated sample with the original ones. Correlation contour plots are estimated by the Gaussian kernel. Representative correlations among four vector, substructure and energy splitting functions (d_1, d_{12}, p^{T}, τ_3) are shown in Fig.2. Surprisingly, we find the model can have almost the same shape on every level of the contour lines, even for plots with multiple peak rings.

As demonstrated above, flowBDT generated sample in very good quality within negligible amount of inference time, 0.4-0.8 ms for each event. Not only that, the training is incredibly fast for low dimensional tasks. We separate each time step with tiny BDT trees, trained on single thread of each CPU core. For 120k events, the training time is around 3-4 minutes per core.

B. Low level simulation for cells and jet constituents

To fully explore the potential of our model, we increase the dimension of the input features to the next magnitude. In the study, we will focus on the dataset 1 which contains layers with different granularities. There are ways to map the features into a latent space, or treat them as flatten array to deal with irregular geometry. The latter does not usually take the shower as 3-D nor structured data. Following by this logic, all the voxel energies now are treated as tabular data.

Output of GBT regression model by default need to be $(N, 1)$, thus number of voxels per shower corresponds to the number of GBTs required for training. Instead, we choose to let each tree has one output so that the number of trees are reduced to only one GBT model per time step. This also makes 5-8 times speedup while inferencing with indistinguishable performance as default setting.

In Fig. 3, we sum up the output raw voxels according to α and r bins of different layers. The flowBDT predicts very well on more granular layers as layer 1 and 2. However, a shortfall in simulating this metric becomes apparent in the energy distribution tails of layer 0 and in instances where showers with low energy deposited on the last layer.

Shower response E_{ratio} as an important quantity used in calibrating objects is shown in Fig. 4. The E_{ratio} generated by the model exhibits a shape and peak width closely mirroring those produced by Geant4. A similar correspondence is observed for another variable concerning the shower location, the center of energy in the η and ϕ directions: $\bar{\eta} = \frac{\langle \eta_i E_i \rangle}{\sum E_i}$ for cell location η_i, as illustrated in Figure. 5.
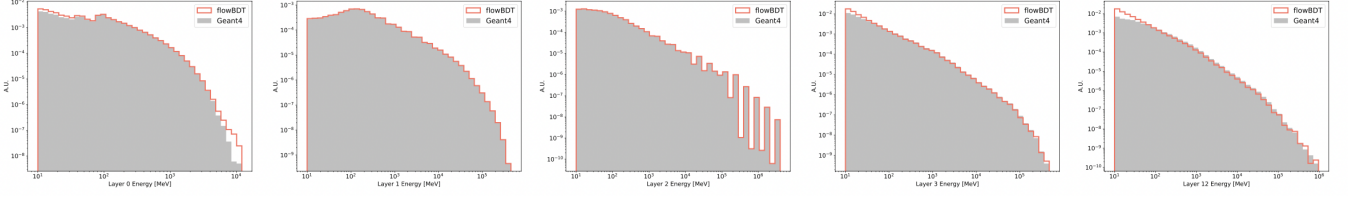
FIG. 3. Different calorimeter layer energies of the photon sample generated by flowBDT and Geant4.
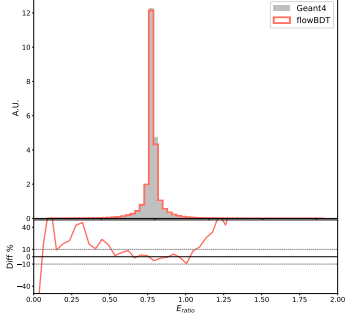

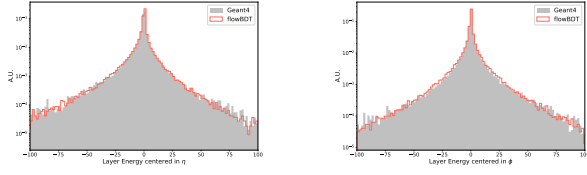FIG. 4. Shower response histogram and ratio plot of the photon sample generated by flowBDT and Geant4.

FIG. 5. Center of energy in η and ϕ direction on second layer of the photon shower generated by flowBDT and Geant4

The same treatment is applied to the JetNet dataset. In this context, the goal is to regress the raw particle-level four-vector data from gluon-initiated and top jets. Generated relative p^{T} distribution has outstanding alignment with the histogram. A small deficiency was observed on the tails of relative location distribution compared to the original dataset, as illustrated in Figure 7. This discrepancy means the limitations of the flowBDT model when applied to more structured datasets, such as point clouds that inherently contain spatial relationships and geometry.

C. Conditional generation

Most generative models encounter similar challenges when attempting to generate high-quality samples from non-Gaussian priors in large dimensions or from distributions that are significantly different from the target distribution. In this section, we will show how conditional generation tackled by the model in both low or high dimensions.

Due to the restricted resolution and acceptance of detectors, the distributions of detector-level quantities diverge from those at the particle level. One important technique used in collider physics is unfolding which corrects these detector effects. Machine learning-based unfolding methods [36, 37, 63] offer significant advantages for unbinned and high-dimensional differential cross-section measurements.

The public jet dataset with several MC generator/tune pairs is used, we consider the leading jet width and mass, multiplicity, N-subjetiness ratio between leading and subleading jets $\tau_{21} = \frac{\tau_2}{\tau_1}$, the normalized groomed mass $\ln \rho = \frac{\ln m_{SD}^2}{p_T^2}$ and momentum fraction z_g for Soft Drop grooming algorithm with specific angularity and cut [74, 75]. To assess the performance of the conditioned generation, we compare with the histograms generated from a Gaussian prior using both binned and unbinned metrics. As shown in Table.2, we found that the separation power and EMD from the non-Gaussian prior demonstrate better performance for the majority of the time. More importantly, starting from a reconstructed level prior provides substantial benefits, as it captures the correlation of the target observables much more effectively than a multidimensional Gaussian would. If we define a relative improvement on correlation difference to be ratio of the sum of absolute values of those differences with data between simulation and generated sample:

$$\mathcal{I}_{rel} = \frac{\sum_{i=1}^{n} Corr_{data,i} - Corr_{sim,i}}{\sum_{i=1}^{n} Corr_{data,i} - Corr_{gen,i}} \quad (5)$$

The relative improvement for conditional generation is around 3040%, which is much higher than 414% improvement calculated from unconditional one. But still both cases show a huge enhancement on general correlation as shown in Fig. 9.

In a shift of focus, the model is utilized to refine and simulate calorimeter showers. The conditional input is derived from parameterized shower approximations produced by fast simulations. In typical machine learning models, we often use discrete conditions such as energy and coordinates for training. However, some attributes, such as material type, are difficult to define numerically. Using an approximate shower as a condition offers a novel
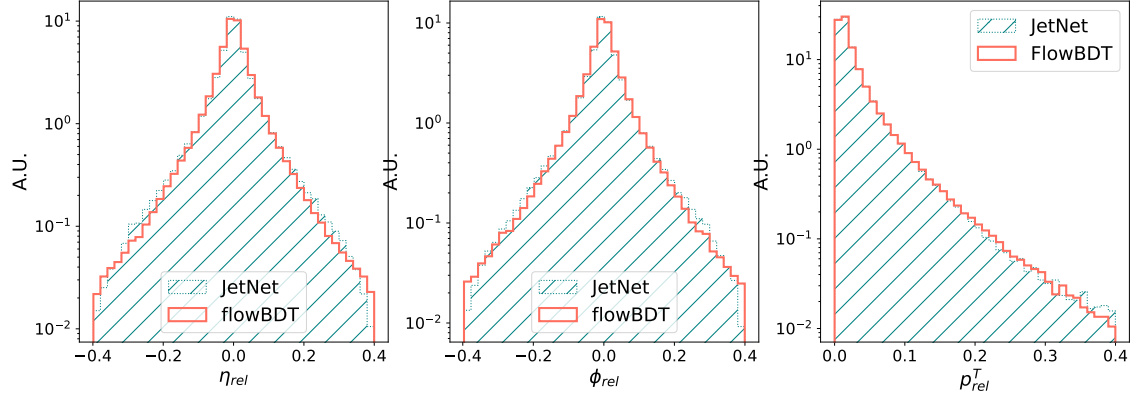
FIG. 6. Relative location and p^{T} of the jet constituents inside gluon-initiated jets from flowBDT and JetNet.
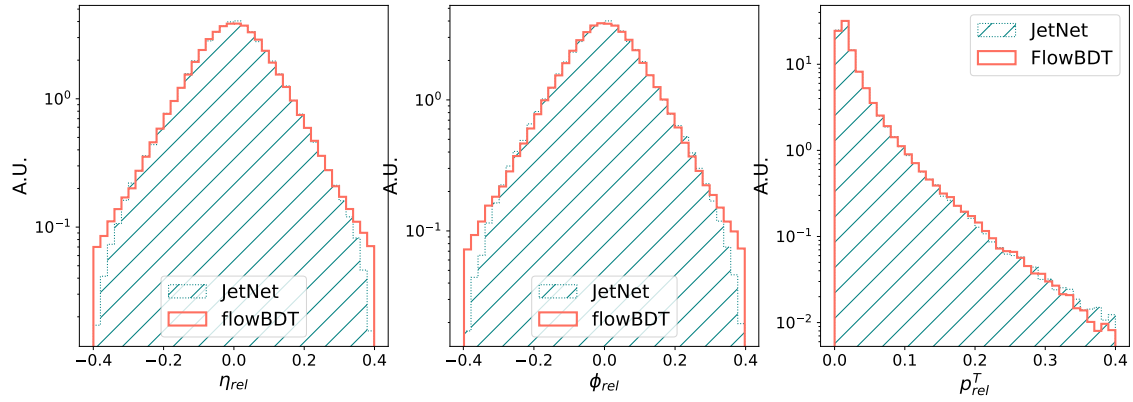

FIG. 7. Relative location and p^{T} of the jet constituents inside top quark jets from flowBDT and JetNet.
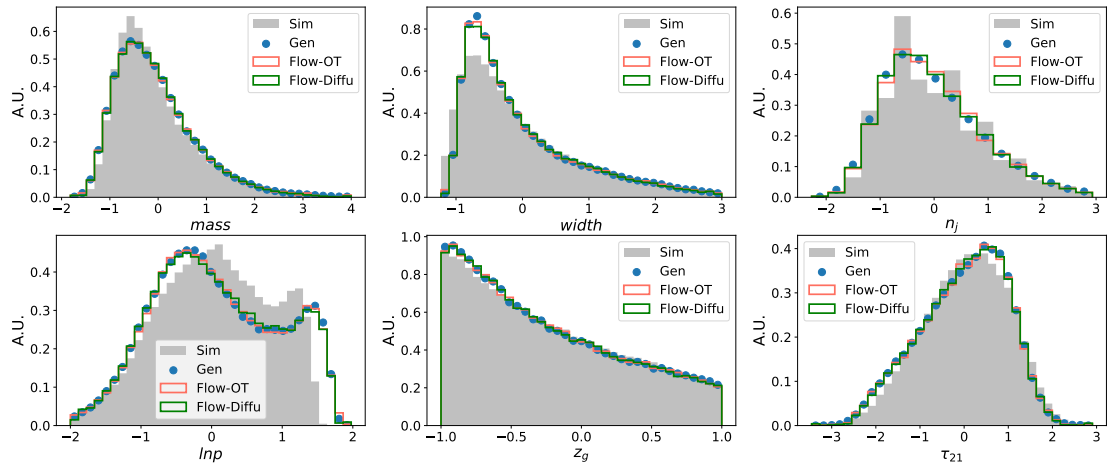

FIG. 8. Comparison plots of the histograms for 6 observables from leading jets for unfolding. Flow-Diffu: from Gaussian prior, Flow-OT: from conditioned prior, Gen: target unfolded observables, Sim: from reconstructed events.
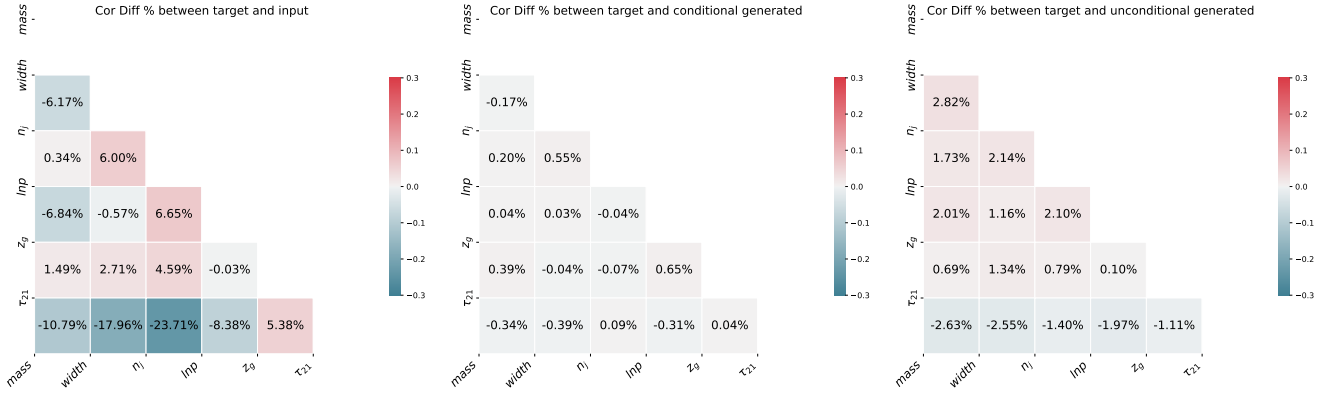
FIG. 9. Comparison plots of the correlation difference heatmaps for 6 observables from leading jets for unfolding. Left: Between target data and original simulation level input, middle: Between target data and generation from conditioned prior, right: Between target data and generation from Gaussian prior.

Observables(Sep power$\times 100$ / EMD$\times 10$)	Simulation	Conditional	Unconditional
Jet mass	0.496/0.369	**0.072/0.038**	0.091/0.084
Jet width	1.770/0.406	0.044/**0.031**	**0.034**/0.098
N constituents	1.129/0.405	0.10/0.28	**0.07/0.26**
$\ln_\rho$	3.684/0.916	**0.053/0.046**	0.10/0.082
z_g	0.047/0.173	**0.010/0.029**	0.013/0.030
τ_{21}	0.558/0.593	**0.095/0.054**	0.14/0.093

TABLE II. Comparison table for evaluation metrics over binned and unbinned distributions of 6 unfolding observables between data and original simulation, generated sample with and without conditional inputs

way to integrate these factors to guide the model with more explicit information inside the calorimeter.

Again, we flatten the calorimeter cells before training. Although this increases the training time compared to tasks with fewer dimensions, the computational cost is still generally smaller than most models we have encountered. Figure 10 shows the histogram of the deposited energy sum for the electron shower. The generated histogram displays a peak width very similar to the original, but with sharper tails at the lower energy end.
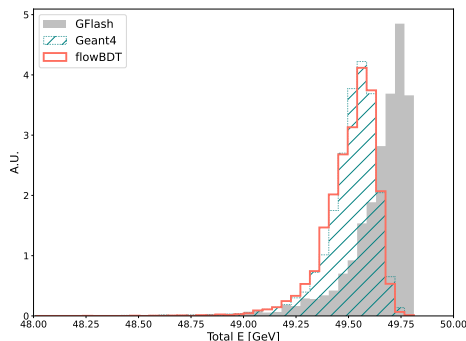

FIG. 10. Total energy deposited for 50 GeV electron shower in the 10 * 10 calorimeter. GFlash: from classical fast simulation, flowBDT: generated sample, Geant4: from full simulation

V. CONCLUSION AND OUTLOOKS

In this work, we have introduced BUFF, a new framework with applications of flow matching and extensive usage of GBT models to accomplish various tasks on different levels of analysis in HEP. We have developed a novel model called flowBDT, which features several enhancements over the original, such as the use of higher-order solvers to reduce training time, distinct GBT training objectives to handle very high-dimensional tasks with a 5-8 times speedup in sampling, and the benefits of conditional generations. The negligible inference time of lightweight GBT models significantly shortens the generation time for high-level tasks compared to traditional flow matching. BUFF is indeed well-suited for rapid training across multiple CPU cores.

We evaluated the generated high-level features for complicated top jet topology using both binned and unbinned metrics. The results, including the f-divergence metric and Earth Mover's Distance, are impressive for most features. We greatly increase the dimensionality for the simulation of low-level calorimeter cells and jet constituents. Our model still shows very decent result for those tasks. Then, we demonstrated some advantages of conditional generation on both high and low levels. For high level tasks such as unfolding, the relative improvement on cor-

relation difference with target from conditional generations is over 30 times higher than before. The conditions can also be the cells from an approximate shower that takes geometry and material information into account.

The model outperforms most tasks specific to tabular data than typical flow-based models. Looking ahead, we aim to extend the application of our model to include a variety of complex tasks such as anomaly detection, quark-gluon tagging, and others. Additionally, for low-level simulations, we will explore alternative strategies to further enhance the efficiency, aiming to significantly speed up both the training and inference time.

VI. ACKNOWLEDGEMENTS

Appendix A: Sampling time metric

We listed two plots in Fig. 11 to graphically demonstrate how should we choose the batch size of generated sample given different dimensonalities. Since multiple GBT models are used during sampling stage, each model needs a short period time to be "awake". This is a non-negligible factor if our sampling time is already below millisecond scale. Any small batch size of generated sample would often required a larger sampling time per feature or per event. For example, the total generation time only scales up to 2 times when increasing from 100 to 1000 events, and up to 4-5 times when scaling from 1000 to 10,000 events. This demonstrates the model's efficient scaling behavior, which is crucial for processing large datasets. Noticeably, for restricting the number of output on each tree would result in a faster inference, this is especially obvious for higher dimensions. As we see in the right plot in Fig. 11, an event with 368 features only consumes 2.5 times more CPU cost than an event with 90 features.
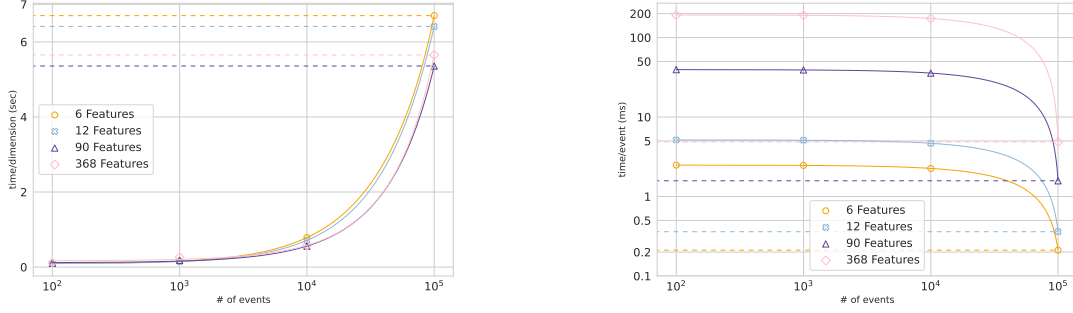
FIG. 11. Sampling time per feature (left) and per event (right) for number of events generated at once with different dimensionalities.

[1] L. Evans and P. Bryant, LHC Machine, Journal of Instrumentation **3** (08), S08001.

[2] A. Buckley, J. Butterworth, S. Gieseke, D. Grellscheid, S. Höche, H. Hoeth, F. Krauss, L. Lönnblad, E. Nurse, P. Richardson, S. Schumann, M. H. Seymour, T. Sjöstrand, P. Skands, and B. Webber, General-purpose event generators for lhc physics, Physics Reports **504**, 145–233 (2011).

[3] G. Apollinari, O. Brüning, T. Nakamoto, and L. Rossi, High Luminosity Large Hadron Collider HL-LHC (2015).

[4] A. Valassi, E. Yazgan, J. McFayden, S. Amoroso, J. Bendavid, A. Buckley, M. Cacciari, T. Childers, V. Ciulli, R. Frederix, S. Frixione, F. Giuli, A. Grohsjean, C. Gütschow, S. Höche, W. Hopkins, P. Ilten, D. Konstantinov, F. Krauss, Q. Li, L. Lönnblad, F. Maltoni, M. Mangano, Z. Marshall, O. Mattelaer, J. Fernandez Menendez, S. Mrenna, S. Muralidharan, T. Neumann, S. Plätzer, S. Prestel, S. Roiser, M. Schönherr, H. Schulz, M. Schulz, E. Sexton-Kennedy, F. Siegert, A. Siódmok, and G. A. Stewart, Challenges in monte carlo event generator software for high-luminosity lhc, Computing and Software for Big Science **5**, 10.1007/s41781-021-00055-1 (2021).

[5] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, Score-based generative modeling through stochastic differential equations (2021), arXiv:2011.13456 [cs.LG].

[6] J. Ho, A. Jain, and P. Abbeel, Denoising diffusion probabilistic models (2020), arXiv:2006.11239 [cs.LG].

[7] P. Dhariwal and A. Nichol, Diffusion models beat gans on image synthesis (2021), arXiv:2105.05233 [cs.LG].

[8] O. Amram and K. Pedro, Denoising diffusion models with geometry adaptation for high fidelity calorimeter simulation (2023), arXiv:2308.03876 [physics.ins-det].

[9] V. Mikuni and B. Nachman, Score-based generative models for calorimeter shower simulation, Physical Review D **106**, 10.1103/physrevd.106.092009 (2022).

[10] V. Mikuni and B. Nachman, Caloscore v2: Single-shot calorimeter shower simulation with diffusion models (2023), arXiv:2308.03847 [hep-ph].

[11] E. Buhmann, F. Gaede, G. Kasieczka, A. Korol, W. Korcari, K. Krüger, and P. McKeown, Caloclouds ii: Ultra-fast geometry-independent highly-granular calorimeter

simulation (2023), arXiv:2309.05704 [physics.ins-det].

[12] A. Butter, T. Jezo, M. Klasen, M. Kuschick, S. P. Schweitzer, and T. Plehn, Kicking it off(-shell) with direct diffusion (2024), arXiv:2311.17175 [hep-ph].

[13] M. Leigh, D. Sengupta, G. Quétant, J. A. Raine, K. Zoch, and T. Golling, Pc-jedi: Diffusion for particle cloud generation in high energy physics (2023), arXiv:2303.05376 [hep-ph].

[14] T. Salimans and J. Ho, Progressive distillation for fast sampling of diffusion models (2022), arXiv:2202.00512 [cs.LG].

[15] Y. Song, P. Dhariwal, M. Chen, and I. Sutskever, Consistency models (2023), arXiv:2303.01469 [cs.LG].

[16] Y. Song and P. Dhariwal, Improved techniques for training consistency models (2023), arXiv:2310.14189 [cs.LG].

[17] I. Kobyzev, S. J. Prince, and M. A. Brubaker, Normalizing flows: An introduction and review of current methods, IEEE Transactions on Pattern Analysis and Machine Intelligence **43**, 3964–3979 (2021).

[18] L. Ardizzone, J. Kruse, S. Wirkert, D. Rahner, E. W. Pellegrini, R. S. Klessen, L. Maier-Hein, C. Rother, and U. Köthe, Analyzing inverse problems with invertible neural networks (2019), arXiv:1808.04730 [cs.LG].

[19] M. Paganini, L. de Oliveira, and B. Nachman, Calo-GAN: Simulating 3D high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks, Physical Review D **97**, 10.1103/physrevd.97.014021 (2018).

[20] S. Bieringer, A. Butter, S. Diefenbacher, E. Eren, F. Gaede, D. Hundhausen, G. Kasieczka, B. Nachman, T. Plehn, and M. Trabs, Calomplification — the power of generative calorimeter models, Journal of Instrumentation **17** (09), P09028.

[21] S. Hoque, H. Jia, A. Abhishek, M. Fadaie, J. Q. Toledo-Marín, T. Vale, R. G. Melko, M. Swiatlowski, and W. T. Fedorko, Caloqvae : Simulating high-energy particle-calorimeter interactions using hybrid quantum-classical generative models (2023), arXiv:2312.03179 [hep-ex].

[22] M. F. Giannelli and R. Zhang, Caloshowergan, a generative adversarial networks model for fast calorimeter shower simulation (2023), arXiv:2309.06515 [physics.ins-det].

[23] M. A. W. Scham, D. Krücker, B. Käch, and K. Borras, Deeptreegan: Fast generation of high dimensional point clouds (2023), arXiv:2311.12616 [hep-ex].

[24] T. Heimel, R. Winterhalder, A. Butter, J. Isaacson, C. Krause, F. Maltoni, O. Mattelaer, and T. Plehn, Mad-NIS - Neural multi-channel importance sampling, SciPost Phys. **15**, 141 (2023), arXiv:2212.06172 [hep-ph].

[25] T. Heimel, N. Huetsch, F. Maltoni, O. Mattelaer, T. Plehn, and R. Winterhalder, The MadNIS Reloaded (2023), arXiv:2311.01548 [hep-ph].

[26] N. Deutschmann and N. Götz, Accelerating HEP simulations with Neural Importance Sampling, JHEP **03**, 083, arXiv:2401.09069 [hep-ph].

[27] T. Heimel, N. Huetsch, R. Winterhalder, T. Plehn, and A. Butter, Precision-Machine Learning for the Matrix Element Method (2023), arXiv:2310.07752 [hep-ph].

[28] C. Gao, S. Höche, J. Isaacson, C. Krause, and H. Schulz, Event generation with normalizing flows, Physical Review D **101**, 10.1103/physrevd.101.076002 (2020).

[29] C. Gao, J. Isaacson, and C. Krause, i-flow: High-dimensional integration and sampling with normalizing flows, Machine Learning: Science and Technology **1**, 045023 (2020).

[30] S. Bright-Thonney, P. Harris, P. McCormack, and S. Rothman, Chained quantile morphing with normalizing flows (2023), arXiv:2309.15912 [hep-ph].

[31] C. Krause and D. Shih, Fast and accurate simulations of calorimeter showers with normalizing flows, Physical Review D **107**, 10.1103/physrevd.107.113003 (2023).

[32] C. Krause and D. Shih, Caloflow ii: Even faster and still accurate generation of calorimeter showers with normalizing flows (2023), arXiv:2110.11377 [physics.ins-det].

[33] S. Schnake, D. Krücker, and K. Borras, CaloPointFlow II Generating Calorimeter Showers as Point Clouds (2024), arXiv:2403.15782 [physics.ins-det].

[34] S. Diefenbacher, E. Eren, F. Gaede, G. Kasieczka, C. Krause, I. Shekhzadeh, and D. Shih, L2lflows: generating high-fidelity 3d calorimeter images, Journal of Instrumentation **18** (10), P10017.

[35] I. Pang, J. A. Raine, and D. Shih, Supercalo: Calorimeter shower super-resolution (2024), arXiv:2308.11700 [physics.ins-det].

[36] M. Backes, A. Butter, M. Dunford, and B. Malaescu, An unfolding method based on conditional Invertible Neural Networks (cINN) using iterative training (2022), arXiv:2212.08674 [hep-ph].

[37] S. Diefenbacher, G.-H. Liu, V. Mikuni, B. Nachman, and W. Nie, Improving generative model-based unfolding with schrödinger bridges (2023), arXiv:2308.12351 [hep-ph].

[38] T. S. Roy and A. H. Vijay, A robust anomaly finder based on autoencoders (2020), arXiv:1903.02032 [hep-ph].

[39] B. Nachman and D. Shih, Anomaly detection with density estimation, Physical Review D **101**, 10.1103/physrevd.101.075042 (2020).

[40] C. Krause, B. Nachman, I. Pang, D. Shih, and Y. Zhu, Anomaly detection with flow-based fast calorimeter simulators (2023), arXiv:2312.11618 [hep-ph].

[41] T. Finke, M. Hein, G. Kasieczka, M. Krämer, A. Mück, P. Prangchaikul, T. Quadfasel, D. Shih, and M. Sommerhalder, Back to the roots: Tree-based algorithms for weakly supervised anomaly detection (2023), arXiv:2309.13111 [hep-ph].

[42] M. J. Dolan, J. Gargalionis, and A. Ore, Quark-versus-gluon tagging in cms open data with cwola and topicflow (2023), arXiv:2312.03434 [hep-ph].

[43] D. P. Kingma and P. Dhariwal, Glow: Generative flow with invertible 1x1 convolutions (2018), arXiv:1807.03039 [stat.ML].

[44] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, Flow matching for generative modeling (2023), arXiv:2210.02747 [cs.LG].

[45] G. Peyré and M. Cuturi, Computational optimal transport (2020), arXiv:1803.00567 [stat.ML].

[46] A. Tong, K. Fatras, N. Malkin, G. Huguet, Y. Zhang, J. Rector-Brooks, G. Wolf, and Y. Bengio, Improving and generalizing flow-based generative models with minibatch optimal transport (2024), arXiv:2302.00482 [cs.LG].

[47] A. Jolicoeur-Martineau, K. Fatras, and T. Kachman, Generating and imputing tabular data via diffusion and flow-based gradient-boosted trees (2024), arXiv:2309.09968 [cs.LG].

[48] T. Hastie, R. Tibshirani, and J. Friedman, Boosting and additive trees, in *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (Springer New York, New York, NY, 2009) pp. 337–387.

[49] L. Mason, J. Baxter, P. Bartlett, and M. Frean, Boosting algorithms as gradient descent, in *Advances in Neural Information Processing Systems*, Vol. 12, edited by S. Solla, T. Leen, and K. Müller (MIT Press, 1999).

[50] R. Shwartz-Ziv and A. Armon, Tabular data: Deep learning is not all you need (2021), arXiv:2106.03253 [cs.LG].

[51] L. Grinsztajn, E. Oyallon, and G. Varoquaux, Why do tree-based models still outperform deep learning on tabular data? (2022), arXiv:2207.08815 [cs.LG].

[52] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, Generative adversarial networks (2014), arXiv:1406.2661 [stat.ML].

[53] D. P. Kingma and M. Welling, Auto-encoding variational bayes (2022), arXiv:1312.6114 [stat.ML].

[54] S. Ruder, An overview of gradient descent optimization algorithms (2017), arXiv:1609.04747 [cs.LG].

[55] M. S. Albergo, N. M. Boffi, and E. Vanden-Eijnden, Stochastic interpolants: A unifying framework for flows and diffusions (2023), arXiv:2303.08797 [cs.LG].

[56] Runge–kutta methods, in *Numerical Methods for Ordinary Differential Equations* (John Wiley & Sons, Ltd, 2008) Chap. 3, pp. 137–316.

[57] M. Calvo, J. Montijano, and L. Randez, A fifth-order interpolant for the dormand and prince runge-kutta method, Journal of Computational and Applied Mathematics **29**, 91 (1990).

[58] R. Kansal, J. Duarte, H. Su, B. Orzari, T. Tomei, M. Pierini, M. Touranakou, J.-R. Vlimant, and D. Gunopulos, Particle cloud generation with message passing generative adversarial networks (2022), arXiv:2106.11535 [cs.LG].

[59] J. Alwall, M. Herquet, F. Maltoni, O. Mattelaer, and T. Stelzer, Madgraph 5: going beyond, Journal of High Energy Physics **2011**, 10.1007/jhep06(2011)128 (2011).

[60] T. Sjöstrand, S. Ask, J. R. Christiansen, R. Corke, N. Desai, P. Ilten, S. Mrenna, S. Prestel, C. O. Rasmussen, and P. Z. Skands, An introduction to pythia 8.2, Computer Physics Communications **191**, 159–177 (2015).

[61] M. F. Giannelli, G. Kasieczka, B. N. Claudius Krause, D. Salamani, D. Shih, and A. Zaborowska, Fast calorime-

ter simulation challenge 2022 (2022).

[62] S. Agostinelli *et al.* (GEANT4), GEANT4–a simulation toolkit, Nucl. Instrum. Meth. A **506**, 250 (2003).

[63] A. Andreassen, P. T. Komiske, E. M. Metodiev, B. Nachman, and J. Thaler, Omnifold: A method to simultaneously unfold all observables, Physical Review Letters **124**, 10.1103/physrevlett.124.182001 (2020).

[64] M. Bähr, S. Gieseke, M. A. Gigg, D. Grellscheid, K. Hamilton, O. Latunde-Dada, S. Plätzer, P. Richardson, M. H. Seymour, A. Sherstnev, and B. R. Webber, Herwig++ physics and manual, The European Physical Journal C **58**, 639–707 (2008).

[65] 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, Journal of High Energy Physics **2014**, 10.1007/jhep02(2014)057 (2014).

[66] S. Diefenbacher, V. Mikuni, and B. Nachman, Refining Fast Calorimeter Simulations with a Schrödinger Bridge (2023), arXiv:2308.12339 [physics.ins-det].

[67] G. Grindhammer, M. Rudowicz, and S. Peters, The fast simulation of electromagnetic and hadronic showers, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **290**, 469 (1990).

[68] G. Grindhammer and S. Peters, The parameterized simulation of electromagnetic showers in homogeneous and sampling calorimeters (2000), arXiv:hep-ex/0001020 [hep-ex].

[69] F. Vaselli, A. Rizzi, F. Cattafesta, and G. Cicconofri (CMS), *FlashSim prototype: an end-to-end fast simulation using Normalizing Flow*, Tech. Rep. (CERN, Geneva, 2023).

[70] F. Topsoe, Some inequalities for information divergence and related measures of discrimination, IEEE Transactions on Information Theory **46**, 1602 (2000).

[71] A. Hoecker, P. Speckmayer, J. Stelzer, J. Therhaag, E. von Toerne, H. Voss, M. Backes, T. Carli, O. Cohen, A. Christov, D. Dannheim, K. Danielowski, S. Henrot-Versille, M. Jachowski, K. Kraszewski, A. K. J. au2, M. Kruk, Y. Mahalalel, R. Ospanov, X. Prudent, A. Robert, D. Schouten, F. Tegenfeldt, A. Voigt, K. Voss, M. Wolter, and A. Zemla, Tmva - toolkit for multivariate data analysis (2009), arXiv:physics/0703039 [physics.data-an].

[72] L. V. Kantorovich, Mathematical methods of organizing and planning production, Management Science **6**, 366 (1960).

[73] M. Arjovsky, S. Chintala, and L. Bottou, Wasserstein generative adversarial networks, in *Proceedings of the 34th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 70, edited by D. Precup and Y. W. Teh (PMLR, 2017) pp. 214–223.

[74] A. J. Larkoski, S. Marzani, G. Soyez, and J. Thaler, Soft drop, Journal of High Energy Physics **2014**, 10.1007/jhep05(2014)146 (2014).

[75] M. Dasgupta, A. Fregoso, S. Marzani, and G. P. Salam, Towards an understanding of jet substructure, Journal of High Energy Physics **2013**, 10.1007/jhep09(2013)029 (2013).