

Foundation model framework for all tasks involving jet physics

Wahid Bhimji¹, Chris Harris¹, Vinicius Mikuni^{2,*} and Benjamin Nachman^{3,4,†}

¹*NERSC, Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA*

²*Nagoya University, Kobayashi-Maskawa Institute, Aichi 464-8602, Japan*

³*Department of Particle Physics and Astrophysics, Stanford University, Stanford, California 94305, USA*

⁴*Fundamental Physics Directorate, SLAC National Accelerator Laboratory, Menlo Park, California 94025, USA*



(Received 11 December 2025; accepted 4 February 2026; published 26 February 2026)

Foundation models use large datasets to build an effective representation of data that can be deployed on diverse downstream tasks. Previous research developed the OmniLearn foundation model for jet physics, using unique properties of particle physics, and showed that it could significantly advance discovery potential across collider experiments. This paper introduces a major upgrade, resulting in the OmniLearned framework. This framework has three new elements: (1) updates to the model architecture and training, (2) using over 1×10^9 jets used for training, and (3) providing well-documented software for accessing all datasets and models. We demonstrate OmniLearned with three representative tasks: top-quark jet tagging with the community Delphes-based benchmark dataset, b tagging with ATLAS full simulation, and anomaly detection with CMS experimental data. In each case, OmniLearned is the state of the art, further expanding the discovery potential of past, current, and future collider experiments.

DOI: [10.1103/knmd-f5jm](https://doi.org/10.1103/knmd-f5jm)

I. INTRODUCTION

Hadronic jets are the result of high-energy quarks and gluons. These objects are ubiquitous in reactions at colliders and elsewhere that involve momentum transfers well above 1 GeV. The radiation pattern within jets encodes information about their origin as well as emergent properties of the strong force. For decades, researchers proposed physically inspired observables to characterize this complex structure [1–3]. Deep learning-based taggers are now standard, but it is challenging to assemble enough examples jets to train state-of-the-art machine learning models for each of the myriad tasks involving jets.

A solution to this challenge is to pretrain an expressive machine learning model on a large number of related examples and then fine-tune the model for downstream tasks. Since the learned representation can be used for many goals, it serves as a foundation model. Most research related to foundation models in particle physics has focused on tokenized models trained via self-supervised learning [4–10], the same setup used to build large language models.

These approaches learn useful representations, but do not make full use of the continuity of the data and/or the label information available from simulations.

An alternative approach is to train using classification and generation objectives that are similar in nature to the ones employed by typical downstream tasks. This is the core philosophy of the OmniLearn foundation model for jets [11,12]. For that model, a transformer-based architecture was trained to classify and generate jets from the 100×106 jet Delphes [13] fast-simulation-based JetClass dataset with ten classes [14]. The model was then fine-tuned on a number of downstream tasks using Geant [15]-based full simulations from ATLAS, CMS, and H1. In all cases, OmniLearn matched or exceeded the state of the art. For example, OmniLearn can significantly reduce the number of computationally expensive, fully simulated examples required to train jet taggers, can accelerate and stabilize neural likelihood-ratio estimation, and can facilitate full phase-space anomaly detection.

In this paper, we present a significant upgrade of OmniLearn. This includes a number of updates to the architecture and training to improve performance. These updates also allow for more per-jet-constituent information to be incorporated into the model as well as for datasets without labels to be used in the foundation model training. The upgraded model, called OmniLearned, is now trained with over 1×10^9 jets (10x the previous model). This is a significant milestone on its own and utilizes the scaling properties of transformers. Alongside this paper, we have prepared software for readily accessing

* Contact author: vmikuni@hepl.phys.nagoya-u.ac.jp

† Contact author: nachman@stanford.edu

Published by the American Physical Society under the terms of the [Creative Commons Attribution 4.0 International](https://creativecommons.org/licenses/by/4.0/) license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI. Funded by SCOAP³.

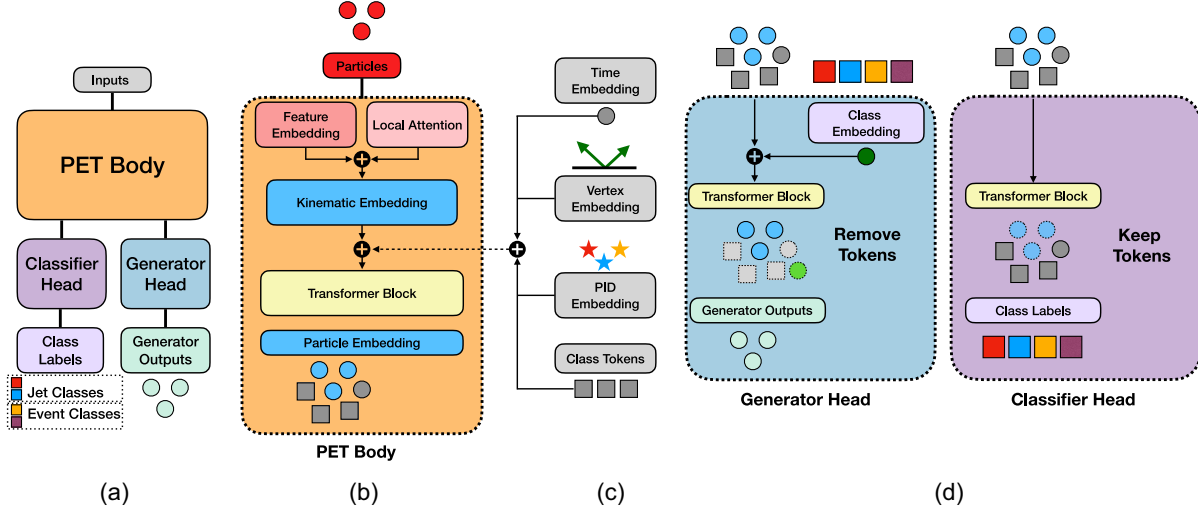


FIG. 1. Neural network architecture used to train OmniLearned. The general model architecture (a) consists of the new PET v2 body (b), input specific embedding (c), and task-specific blocks (d). See the text for more details.

all of the train and test datasets in a unified framework as well as well-documented code for training/fine-tuning the machine learning models. The new framework is demonstrated on three key tasks. First, the models are compared with the current state of the art on the community benchmark top quark and quark-versus-gluon tagging datasets. Second, we are able to improve the performance of b tagging with ATLAS public simulations. As part of this task, we also show how to repurpose parts of the foundation model for tasks different than their original intention (per-track classification from jet generator). Lastly, we show how the top quark can be effectively rediscovered using anomaly detection deployed on CMS open data. Along the way, we show how the outputs of the foundation model can also be used directly (without fine-tuning) for anomaly detection when the target is relatively close to a class in the training dataset.

The paper is organized as follows. Section II describes the improvements to the point edge transformer (PET) model that forms the backbone of OmniLearned. Section III then introduces the dataset used to train OmniLearned with over 1×10^9 jets. Results for different tasks are then described in Sec. IV for classification using several benchmarks, and Sec. V on performing anomaly detection using the CMS Open Data. The paper ends with conclusions and outlook in Sec. VI.

II. OmniLearned

In the previous OmniLearn model, a transformer network combined with a graph neural network named point-edge transformer (PET) was used as the backbone architecture. This choice naturally addresses the data structure of jets as point clouds with varying number of particles, an approach that was successfully used for multiple tasks in collider physics [14,16–22]. Building on the success of the PET model, we further improve the architecture (to PET v2) by

modifying the following key properties to enhance expressiveness and flexibility to the model. The new model architecture is shown in Fig. 1.

A. Input features

In the previous OmniLearn implementation, the inputs to the model were the kinematic information of each particle clustered in the jet. While only four numbers are necessary to completely describe the momentum of each particle, redundant features were added to accelerate training convergence. Such features included the difference in pseudorapidity η and azimuthal angle ϕ between each particle and the jet axis, together with the absolute distance Δ_R , calculated using the same η and ϕ values. In OmiLearned, we maintain the difference in η and ϕ between the particle and jet axis as inputs but only add two additional features, consisting of the logarithm of the transverse momentum $\log p_T$ and energy $\log E$. We also remove the preprocessing step, previously used to standardize the mean and variance of the input features, since this choice of representation already leads to each input roughly centered at zero with $\mathcal{O}(1)$ variance. That leaves us with the minimal set of inputs supported by OmniLearned as $(\Delta\eta, \Delta\phi, \log p_T, \log E)$.

B. Encoding additional information

While the kinematic information for each particle is always present in all datasets considered in this work, some datasets also provide additional information such as particle identification (PID) or vertex information for charged particles. In the previous OmniLearn model, we addressed this challenge by creating specific embedding blocks to address additional features. During training, the additional features would be randomly dropped with fixed probability and all additional features were replaced by zeros. This strategy is similar to dropout [23] and allows the

model to learn a useful representation in the presence and absence of these additional features. We maintain the same successful methodology but modify the exact implementation of each network block. First, the PID information is handled separately from the vertex information. Since the PID is naturally a discrete feature used to label a particle as an electron, muon, photon, charged or neutral hadron, we use a simple lookup table to embed the PID information, similarly to how tokenized words are encoded in language models. The vertex information is also handled separately from the other features and is encoded using two multilayer perceptron (MLP) layers with Gaussian error linear unit (GELU) [24] nonlinearity. The result of the encoded PID and vertex features are added to the kinematic features such that in the absence of either PID or vertex information the corresponding encoding is set to zero. When adapting the model to new datasets, if the dataset contains either PID or vertex information, the corresponding encoding blocks are loaded from the base model; otherwise they are safely ignored. The time parameter used to condition the diffusion generation (more in Sec. II F) is also embed in the model using Fourier features [25] followed by MLP layers with GELU activation. Instead of adding the time embedding to the overall latent representation we include the time information as an additional point in the point cloud. This choice, paired with the use of transformer blocks, improves the conditioning of the diffusion model with the time parameter, improving the generation quality during downstream tasks.

C. Local attention layers

The use of local information improves the ability of the model to grasp correlations between particles inside the jets. These local correlations can be incorporated using implementations similar to dynamic graph convolution layers where the neighborhood is defined using a k -nearest neighbor algorithm. For each particle x_i in the jet, we calculate pairwise features $f(x_i, x_j)$ with respect to each of the j neighbors. In the original OmniLearn model, the function $f(x_i, x_j) = [x_i, x_i - x_j]$ was chosen. MLP layers then used the pairwise features as inputs before the average across all neighbors was calculated. While this implementation has been successful in encoding the local relationship between particles, we perform two modifications that further improve the expressiveness and scalability of these operations. First, inspired by Ref. [26] we include physics-inspired interaction terms between each particle and their k -nearest neighbors. In particular, we modify the function $f(x_i, x_j)$ to be,

$$f(x_i, x_j) = [x_i - x_j, \log m(x_i, x_j), \log \Delta_R(x_i, x_j), \log[\min\{p_{Ti}, p_{Tj}\} \times \Delta_R(x_i, x_j)]], \quad (1)$$

where $m(x_i, x_j)$ is the invariant mass of the four-vector sum and $\Delta_R(x_i, x_j)$ is the distance in the pseudorapidity-azimuthal plane between particles x_i and x_j . The second modification is to replace the simple average with a learnable weighted average. In practice, we implement a transformer block where the attention mechanisms attend to only particle x_i and their k neighbors. Since attention is a computationally expensive operation, we set $k = 10$, which leads to the overall complexity of $N \times k \times k$. Since we later consider jets with up to 150 particles, the final complexity of the local attention layers is roughly equivalent to the one obtained by having a transformer block where all particles attend to all other particles. The use of attention layers instead of a simple average allows the model to learn the importance of each neighbor when creating the local embedding while also becoming more robust against variations in the overall particle multiplicity in the jet.

D. Global attention layers

The use of transformer layers [27] is now the default architecture used both in collider physics and elsewhere due to the scalability of transformer models when trained with large amounts of data. In the previous PET implementation, a generic transformer block was used. In the new implementation, we modify the attention mechanism to add the same set of physics inspired interaction terms to the attention matrix. Effectively, we modify the attention matrix by adding a bias term that encodes the same physics quantities from Eq. (1), now calculated between each pair of particles. In addition, we replace the standard layer normalization operations with a learnable hyperbolic tangent operation [28], which is shown to improve stability even when considering dozens of transformer blocks.

E. Task specific heads

The joint data representation is used as input to task specific blocks that aim to solve complementary tasks in the form of jet classification and jet generation. In the previous OmniLearn implementation, the classification head used a classification token, implemented as a learnable token tasked to summarize the information of all particles inside the jet. This additional token was treated in the same way as the other particles in the jet and attended to all particles during transformer blocks. We noticed that adding more learnable tokens was helpful, resulting in each token learning to summarize different particle information, similar to a multihead algorithm. In our new implementation, we use five learnable tokens that are now added to the set of particles as part of the general embedding model instead of only being added at the classification block. For jet classification, the tokens themselves are used to determine the classification outputs, while for jet generation, the additional tokens are removed before the output layer.

F. Loss function

The loss introduced in OmniLearn consisted of a combination of the classification and generation tasks. To combine the two representations, an additional loss term took the perturbed inputs, designed for the training of the diffusion models, through the classification head. The overall loss function was,

$$\mathcal{L} = \mathcal{L}_{\text{class}} + \mathcal{L}_{\text{gen}} + \mathcal{L}_{\text{class smear}} \quad (2)$$

$$= \text{CE}(y, y_{\text{pred}}) + \|v - v_{\text{pred}}\|^2 + \alpha^2 \text{CE}(y, \hat{y}_{\text{pred}}). \quad (3)$$

The cross-entropy (CE) term is calculated using the true jet labels and the output of the classification head. In the new implementation, we would also like to include jets where the truth label is not present, which is the case for data collected by experiments. In the absence of truth labels, we design a separate classification task, where the goal is to classify jets coming from different datasets. In this case, we break down the output of the classifier head into two, one used for supervised jet flavor classification, and one used for sample classification, where the cross entropy loss for both classification tasks is added equally in the loss function.

We also modify the training of the generative part based on diffusion models. During the diffusion model training, particles clustered inside the jet were first perturbed using a time-dependent Gaussian value,

$$z(t) = \alpha(t)x + \sigma(t)\epsilon, \quad (4)$$

where $\epsilon \sim \mathcal{N}(0, 1)$ and α and σ determine the mixture between clean data x and noise ϵ . The goal of the diffusion model is then to take as input $z(t)$ and predict the velocity term $v = \alpha(t)\epsilon - \sigma(t)x$ introduced in Ref. [29], a linear combination of the noise and the clean data. This approach has been very successful at generating jets [30,31] with high fidelity. Recently, an alternative parametrization has been introduced in the form of flow matching models [32], where the perturbed data is used as inputs to a model that instead predicts the so-called *velocity field* $u = \epsilon - x$. The velocity field estimates the path that connects the noise distribution to the data as the solution of an ordinary differential equation. Even though diffusion and flow matching models are presented as different philosophies, they are equivalent. In fact, for the choice of $\alpha(t) = 1 - t$ and $\sigma(t) = t$, often used in flow matching models, the perturbed data used as input for the flow matching training is the same as the one from Eq. (4). Moreover, the diffusion loss function defined as,

$$\mathcal{L}_D = \|v_{\theta}(z_t, t) - v\|^2, \quad (5)$$

for a neural network model $v_{\theta}(z_t, t)$ with trainable parameters θ is related to the flow matching loss function

TABLE I. List of open datasets used during the pretraining of OmniLearn and the amount of events available for training, testing, and validation.

Dataset	Training	Validation	Test
JetClass [14]	100M	20M	5M
JetClass2 [33]	200M	600k	600k
Aspen open jets [34]	125M	25.7M	26.6M
ATLAS top tagging [35]	178M	20M	2.2M
H1 DIS	42.2M	872k	255k
CMS QCD	239M	17.5M	16M
CMS BSM	173.5M	17M	17M
Total	1057.7M	101.8M	67.6M

$$\mathcal{L}_F = \|u_{\theta}(z_t, t) - u\|^2 = w(t)\|v_{\theta}(z_t, t) - v\|^2, \quad (6)$$

through a time-dependent weight $w(t) = \frac{[\sigma + \alpha]^2}{\sigma^2 + \alpha^2}$.

Even though both formulations lead to similar losses, the training convergence rate and sample quality between diffusion and flow matching models can be different, with flow matching models often leading to better generation quality with a similar number of sampling steps to the diffusion training. For this reason, we change the velocity prediction from OmniLearn to the flow matching objective in Eq. (6) resulting in the combined loss,

$$\mathcal{L} = \text{CE}(y, y_{\text{pred}}) + \|u - u_{\text{pred}}\|^2 + \alpha^2 \text{CE}(y, \hat{y}_{\text{pred}}). \quad (7)$$

This change also leads to the modification to the function α in Eq. (3) that multiplies the cross entropy term evaluated over perturbed inputs.

III. 1×10^9 DATASET

The growth of the machine learning community for high energy physics closely follows the increase in availability of open datasets used for benchmarking and research. Multiple datasets for jet physics have been made available, each hosted on different platforms and saved with different data formats. As part of the OmniLearn release, we also provide a software package that automatically accesses and downloads all datasets used in this work to unify them in the same format. The datasets used for the pretraining of OmniLearn are listed in Table I.

The combined dataset size amounts to more than 1×10^9 jets. A total of 210 classes are created across all datasets, 200 of them targeting specific jet flavors and ten of them attached to specific datasets. The first dataset included in the pretraining is the JetClass dataset [14] consisting of ten different jet classes simulated using MADGRAPH5_aMC@NLO [36] for the matrix element calculation and Pythia 8 [37,38] for parton showering and hadronization. Delphes 3.4.3 [13,39,40] is used to simulate detector effects with the CMS detector configuration. Jets are clustered using the anti- k_t algorithm with radius parameter of $R = 0.8$ [41–43].

Only jets with transverse momentum between 500 and 1000 GeV and pseudorapidity $|\eta| < 2.0$ are considered. The JetClass dataset was upgraded with additional 188 classes in the JetClass 2 dataset [33], consisting of fine-grained labels for different combinations of initial partons, but using the same generators, detector simulation, and jet clustering setup as JetClass. The ATLAS Top Tagging dataset [35] released by the ATLAS Collaboration consists of top quarks and quantum chromodynamics (QCD) non-resonant jets simulated using the ATLAS detector simulation based on Geant4 [15,44]. Events are generated with Pythia8 using the NNPDF2.3LO [45] set of parton distribution functions and the A14 [46] set of tuned parameters. Pileup effects are simulated by overlaying inelastic interactions on top of the underlying hard scattering process based on the 2017 data taking period. Jet constituents are reconstructed based on the Unified Flow Object algorithm [47]. Jets are clustered using anti- k_t algorithm with $R = 1.0$ with additional Soft-Drop [48] and pileup mitigation [49–51] algorithms applied. Compared to the previous version used by OmniLearn, the ATLAS Top Tagging dataset was updated to also contain systematic variations of the nominal samples used for the simulations. These variations cover the same set of uncertainties commonly considered for physics analysis at the LHC. Since we only use this dataset for the pretraining of the model, we include both nominal and varied datasets together during training, without explicitly making a distinction between systematic variations and nominal samples. These events also receive the same label assigned for top quarks and QCD events in the JetClass dataset for consistency. To increase the variety of jet examples at different regions of the phase space we also include jets simulated across different collision systems. We use simulations of neutral-current deep inelastic scattering (DIS) provided by the H1 Collaboration and generated using two different generators: the Rapgap 3.1 [52] and Djangoh 1.4 [53] generators for electron-proton collisions with electron and proton beam energies of 27.6 GeV and 920 GeV, respectively. The detector simulation of the H1 detector is performed using the Geant3 [54,55] package. An energy-flow algorithm [56–58] is then used to reconstruct the particles clustered into jets using the k_t algorithm with $R = 1.0$. Since these events target a different physics process, we assign distinct labels to each simulation, separately from the other labels considered so far. The last set of events used for the pretraining of OmniLearn are processed from the open dataset released by the CMS Collaboration, consisting of teal or simulated proton-proton collisions collected in 2016. The preprocessed data events were also released as part of the Aspen Open Jets [34] dataset. Additional simulations from the CMS Collaboration for QCD and several potential new physics processes were also produced using the same framework. The new physics samples were chosen such that they individually yield a large amount of jets in the final state, while providing complementary information based on

the datasets already considered. The selected samples are vectorlike quark production with a B' and T' , charged Higgs production, bulk Graviton and radion production, supersymmetry (SUSY) in the context of the next to minimal supersymmetric Standard Model, SUSY with displaced vertices, Z' production, and a narrow resonance X that decays to a resonant pair YY , which in turn decay to pairs of jets. These new physics samples combined with the QCD and the experimental data collected by the CMS Collaboration form the additional 10 classes used for sample classification during the pre-training phase.

In all datasets, up to 150 particles are saved per jet to be used during training. The training is carried out on the Perlmutter Supercomputer [59] using from 32 to 512 GPUs simultaneously. For different model sizes, the local batch was chosen based on the memory available for each GPU. The global batch size, defined as the local batch size multiplied by the number of GPUs, is kept the same for all models and equal to 4096. All pretrained models are trained for three full passes of the 1×10^9 dataset. OmniLearn is implemented in PYTORCH [60]. The cosine learning rate schedule [61] is used with an initial learning rate of 1×10^{-5} and decreased to 10^{-6} until the end of the training. The LION optimizer [62] is used with parameters $\beta_1 = 0.95$ and $\beta_2 = 0.98$. The fine-tuning of OmniLearn across different datasets and tasks is performed by setting the learning rate of all network weights to be a factor 5 smaller than the output layer. We investigate three model sizes for OmniLearn named small (-s), medium (-m), and large (-l) models with roughly 3M, 58M, and 460M trainable weights, respectively. In Appendix the details of each model size is provided. Some hyperparameter optimization of all parameters stated above suggested that large gains are not readily available from modest tweaks to the setup.

IV. JET CLASSIFICATION

The initial evaluation of OmniLearn on jet classification is carried out using two widely used benchmark datasets in collider physics: top quark tagging [63] and quark/gluon [16] classification. In the top quark tagging dataset, events are simulated using Pythia 8 and Delphes with the ATLAS detector configuration. The background process consists of nonresonant jets produced via QCD and the signal consists of top quark pair production with all-hadronic decays. Jet constituents are clustered using the anti- k_t algorithm with $R = 0.8$ and all jets in the range $550 \text{ GeV} < p_T < 650 \text{ GeV}$ and $|\eta| < 2$ are saved. While multiple datasets in the pretraining data use top quarks and QCD jets, this particular dataset uses a different detector configuration and narrower p_T window. The quark/gluon dataset consists of stable particles, without detector simulation, clustered into jets. Neutrinos are excluded and the anti- k_t algorithm with radius $R = 0.4$ is used. The quark-initiated sample (signal) is generated using a $Z(\nu\nu) + q$ while the gluon-initiated data (background) are generated using $Z(\nu\nu) + g$ processes.

TABLE II. Comparison between the performance reported for different classification algorithms on the top tagging dataset. The uncertainty quoted corresponds to the standard deviation of five trainings with different random weight initialization. If the uncertainty is not quoted then the variation is negligible compared to the expected value. Bold results represent the algorithm with highest performance.

	Acc	AUC	$1/\epsilon_B$	
			$\epsilon_S = 0.5$	$\epsilon_S = 0.3$
ResNeXt-50 [19]	0.936	0.9837	302 ± 5	1147 ± 58
P-CNN [19]	0.930	0.9803	201 ± 4	759 ± 24
PFN [16]	...	0.9819	247 ± 3	888 ± 17
ParticleNet [19]	0.940	0.9858	397 ± 7	1615 ± 93
JEDI-net [18]	0.9300	0.9807	...	774.6
PCT [22]	0.940	0.9855	392 ± 11	1559 ± 98
LGN [64]	0.929	0.964	...	435 ± 95
rPCN [20]	...	0.9845	364 ± 9	1642 ± 93
LorentzNet [65]	0.942	0.9868	498 ± 18	2195 ± 173
PELICAN [66]	0.9425	0.9869	...	2289 ± 204
ParT [14]	0.940	0.9858	413 ± 16	1602 ± 81
ParT-f.t. [14]	0.944	0.9877	691 ± 15	2766 ± 130
Mixer [67]	...	0.9859	416	...
MIParT [26]	0.942	0.9868	505 ± 8	2010 ± 97
MIParT-f.t. [26]	0.944	0.9878	640 ± 10	2789 ± 133
L-GATr [68]	0.9423	0.9870	540 ± 20	2240 ± 70
L-GATr-f.t. [68]	0.9446	0.9879	651 ± 11	2894 ± 84
PET [11,12]	0.938	0.9848	340 ± 12	1318 ± 39
OmniLearn [11,12]	0.942	0.9872	568 ± 9	2647 ± 192
PET v2- S	0.9427	0.987	505 ± 14	2167 ± 153
OmniLearned-S	0.944	0.9875	565 ± 12	2637 ± 128
PET v2-M	0.9423	0.987	482 ± 11	1861 ± 61
OmniLearned-M	0.944	0.9880	656 ± 12	3208 ± 176
OmniLearned-L	0.944	0.9880	688 ± 9	3486 ± 157

Both samples are generated using Pythia 8. Jets are required to have transverse momentum $p_T \in [500, 550]$ GeV and rapidity $|y| < 1.7$. OmniLearned is then fine-tuned on each dataset and compared against the results obtained by the original OmniLearn model and multiple alternatives. The results for different quality metrics are listed in Tables II and III.

Notably, the improved PET v2 model shows significantly better results compared to the previous iteration, matching and sometimes surpassing state-of-the-art models also trained from scratch on the same dataset. After fine-tuning, the performance of OmniLearned increases considerably and surpasses all previous benchmarks. Moreover, we observe a strong relationship between the model size of OmniLearned and the fine-tuning performance, with bigger models showing better results after fine-tuning. We also provide a visualization of the performance obtained by different algorithms in Fig. 2 for the background rejection at 30% signal efficiency.

The last classification task we investigate uses the dataset released by the ATLAS Collaboration for flavor tagging [69,70]. Smaller radius jets with $R = 0.4$ are clustered

TABLE III. Comparison between the performance reported for different classification algorithms on the quark and gluon dataset. The uncertainty quoted corresponds to the standard deviation of nine trainings with different random weight initialization. If the uncertainty is not quoted then the variation is negligible compared to the expected value. Bold results represent the algorithm with highest performance.

	Acc	AUC	$1/\epsilon_B$	
			$\epsilon_S = 0.5$	$\epsilon_S = 0.3$
P-CNN [19]	0.827	0.9002	34.7	91.0
PFN [16]	...	0.9005	34.7 ± 0.4	...
ParticleNet [19]	0.840	0.9116	39.8 ± 0.2	98.6 ± 1.3
rPCN [20]	...	0.9081	38.6 ± 0.5	...
ParT [14]	0.840	0.9121	41.3 ± 0.3	101.2 ± 1.1
ParT-f.t. [14]	0.843	0.9151	42.4 ± 0.2	107.9 ± 0.5
PET [11,12]	0.837	0.9110	39.92 ± 0.1	104.9 ± 1.5
OmniLearn [11,12]	0.844	0.9159	43.7 ± 0.3	107.7 ± 1.5
PET v2-S	0.842	0.9137	41.7 ± 0.4	104.4 ± 1.3
OmniLearned-S	0.844	0.9153	42.9 ± 0.2	108.0 ± 0.5
PET v2-M	0.841	0.9135	41.3 ± 0.6	103.8 ± 1.1
OmniLearned-M	0.845	0.9162	43.2 ± 0.1	111.2 ± 1.5

using the anti- k_t algorithm and simulated using top quark decays from POWHEG BOX [71–74] interfaced with Pythia 8 at next-to-leading-order accuracy. Reconstructed tracks are used as inputs for the training. Jets are matched to hadron-level jets if they are found in a radius of 0.3 and are assigned the parton flavor as a label. Four different classes are considered for the classification task: b(ottom) jets, c(harm) jets, light jets, and jets produced through hadronic decays from tau leptons. Besides the basic kinematic quantities used by OmniLearned, additional vertex information is available. To promote a fair comparison with current flavor tagging algorithms used by the ATLAS Collaboration, we include all features that are also used by the Collaboration’s current state of the art GN2 architecture, with detailed explanation for each feature presented in Ref. [69]. In addition to the main classification task, GN2 also includes auxiliary tasks that help improve the performance of the tagger. These include the prediction of the track origin and the identification of tracks sharing a common vertex. The first auxiliary task assigns a label to each track, similar to the PID information present in other datasets. When training OmniLearned, we perform the jet classification using the classification but repurpose the generation head to perform track classification, thus enabling the model to benefit from the additional auxiliary task. In this case, all the weights from the generation head are loaded during the fine-tuning process while the output layers are replaced to match the number of classes present in each classification task. Even though the diffusion task is different than the track classification we observe the use of the pretrained weights to still be beneficial compared to starting from random weights.

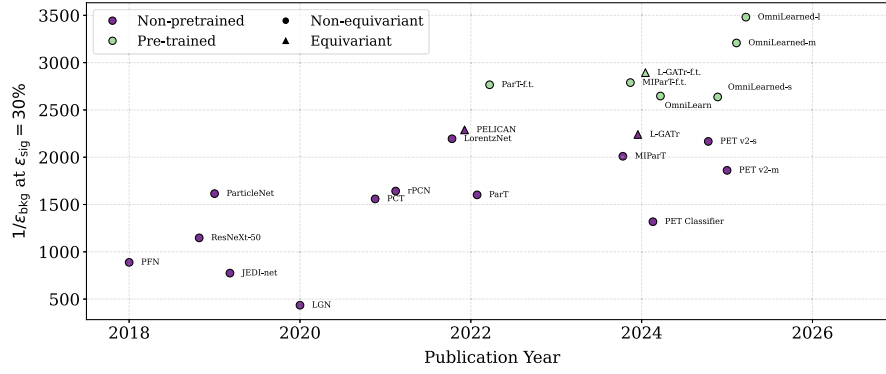


FIG. 2. Background rejection efficiency for a fixed signal efficiency of 30% in the community top tagging dataset.

All models are trained using the full dataset consisting of 168×106 jets and evaluated using the medium dataset consisting of 25.6×106 jets. From the trained classifiers, the output score of the network is used to determine the tagging response. Following the ATLAS Collaboration strategy, The b- and c-tagging discriminators are defined as,

$$D_b = \log \left(\frac{p_b}{f_c p_c + f_\tau p_\tau + (1 - f_c - f_\tau) p_u} \right),$$

$$D_c = \log \left(\frac{p_c}{f_b p_b + f_\tau p_\tau + (1 - f_b - f_\tau) p_u} \right), \quad (8)$$

with parameters $f_b = 0.3, f_c = 0.2, f_\tau = 0.01$ optimized for GN2. We build the same discriminator functions, but instead use $f_b = 0.2$, which improves rejection against light jets without impacting tagging performance for other classes.

Results of the training are listed in Table IV, where we present results for both b- and c-tagging. Results are compared with the ones reported by the ATLAS Collaboration trained using the same open dataset for reconstructed jets with $p_T > 20$ GeV.¹

All models show improved performance compared to the GN2 baseline and OmniLearned shows improved performance compared to the models trained from scratch. In particular, the impact of the fine-tuning is more pronounced for OmniLearned-S, improving the performance compared to the model trained from scratch by up to 15%. Although bigger models also show better performance, the impact of fine-tuning on performance is smaller. However, in this case, the fine-tuning requires less than half of the network updates to achieve convergence, making the fine-tuning 50% faster than training the medium model from scratch.

¹Notice that the numbers reported are based on the training performed using the open dataset only. Public results for the GN2 model use a dataset created from a combination of top quark decays and Z' events, with the latter not available at the time of the writing of this manuscript.

We also provide the results of the receiver operating characteristic curve in Fig. 3 for different values of the signal efficiency.

We observe noticeable improvements for b- and c tagging in terms of background rejection for all considered background types. In the case of b tagging, light-jet rejection improves by more than 50% below 80% b-tag efficiency while τ -jet rejection improves by a factor 2. Improvements to c-jet rejection are also observed to be more than 50% for b-jet efficiencies below 70%. Improvements were also observed in c tagging, where the light-jet rejection improves by 30% to 50% depending on the c-jet efficiency. For τ - and b jets the rejection compared to GN2 improves by around 20% and 5%–30%, respectively.

TABLE IV. Comparison between the performance reported for different classification algorithms on the ATLAS flavor tagging dataset. Values reported for GN2 are obtained from the model trained on the same dataset for a fair comparison. While the results of a single model trained are displayed, we notice that the performance listed is stable against multiple runs of the same algorithms apart from GN2, whose baseline is taken from the public results.

b-tagging $1/\epsilon_B$ ($\epsilon_b = 70\%$)			
Algorithm	C jets	Light jets	τ jets
GN2	45.5	1097	245
PET v2-S	55.7	1512	409
PET v2-M	63.6	1772	494
OmniLearned-S	61.4	1726	459
OmniLearned-M	66.5	1853	493
C-tagging $1/\epsilon_B$ ($\epsilon_c = 30\%$)			
Algorithm	B jets	Light jets	τ jets
GN2	21.1	166	21.4
PET v2-S	22.4	206	24.1
PET v2-M	24.0	233	26.7
OmniLearned-S	23.6	221	26.0
OmniLearned-M	24.8	235	28.4

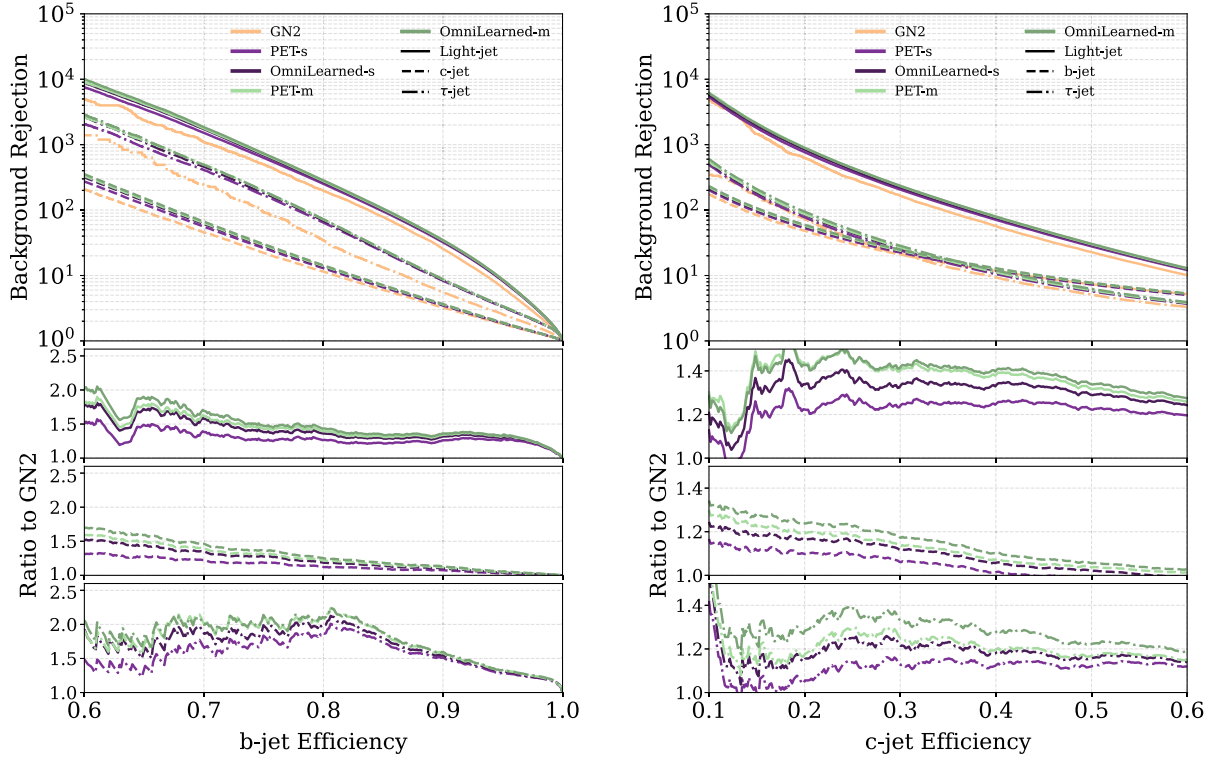


FIG. 3. Receiver operating characteristic for b- (left) and c tagging (right) for different algorithms. The ratio plots show the background rejection improvement compared to GN2 for different background jet classes.

V. ANOMALY DETECTION

Our last example integrates multiple components of OmniLearned into one. In particular, we consider weakly supervised, resonant anomaly detection [75–80]. This is a collection of tools for automatically searching for new physics in multidimensional feature spaces. Algorithms in this toolkit are composed of two parts: a mechanism for estimating the Standard Model background in an interval (signal region) of the reconstructed resonance mass and a procedure for determining how compatible the estimated data are with the real data. A well-studied benchmark approach is Cathode [81], which uses generative models to be able to sample background events conditioned on the reconstructed resonance mass. This model is trained outside the signal region (in the sideband region). Then, a classifier is trained to distinguish samples from the generative model and data in the signal region. Historically, this classifier was trained using a small set of high-level features, but OmniLearn showed that sensitivity to anomalies is also possible using the full phase space if² the classifier uses a foundation model [12]. After selecting events that the classifier flags as anomalous, a parametric fit is performed to quantify the statistical significance. Versions of

²Reference [82,83] showed a similar boost in sensitivity by pretraining and parametrizing the classifier in new physics properties.

this setup have been used by ATLAS [84,85] and CMS [86] to look for dijet resonances and by others to discover unisolated Upsilon's with CMS data [87] and cold stellar streams with Gaia data [88–92]. In this section, we deploy these methods to rediscover the top quark in the single, high- p_T jet channel.³ Interesting in its own right as a first search with single jets and as a new benchmark to test anomaly detection methods in data with a known resonance,⁴ this example demonstrates both the generative and discriminative capabilities of OmniLearned.

For this study, we use the Aspen Open Jets dataset, which contains jets collected by the CMS Collaboration during the 2016 data taking period. We consider the problem of identifying top quark initiated jets, a physics process expected to consist of only 0.1% of the jets contained in the dataset [34]. We consider jets with p_T greater than 500 GeV and define the signal region consisting of jets with Soft Drop mass [48] between 140 GeV and 220 GeV. The diffusion head in OmniLearned is then fine-tuned using the sidebands, including as jets with mass

³Reference [93] also used anomaly detection to rediscover the top quark, but deployed unsupervised learning and required simulations to estimate the nontop quark background.

⁴Reference [87] also provided such a dataset for the Upsilon. Given the many challenges experienced in practice [84–86] that were absent from phenomenology studies, it is crucial to have multiple such examples.

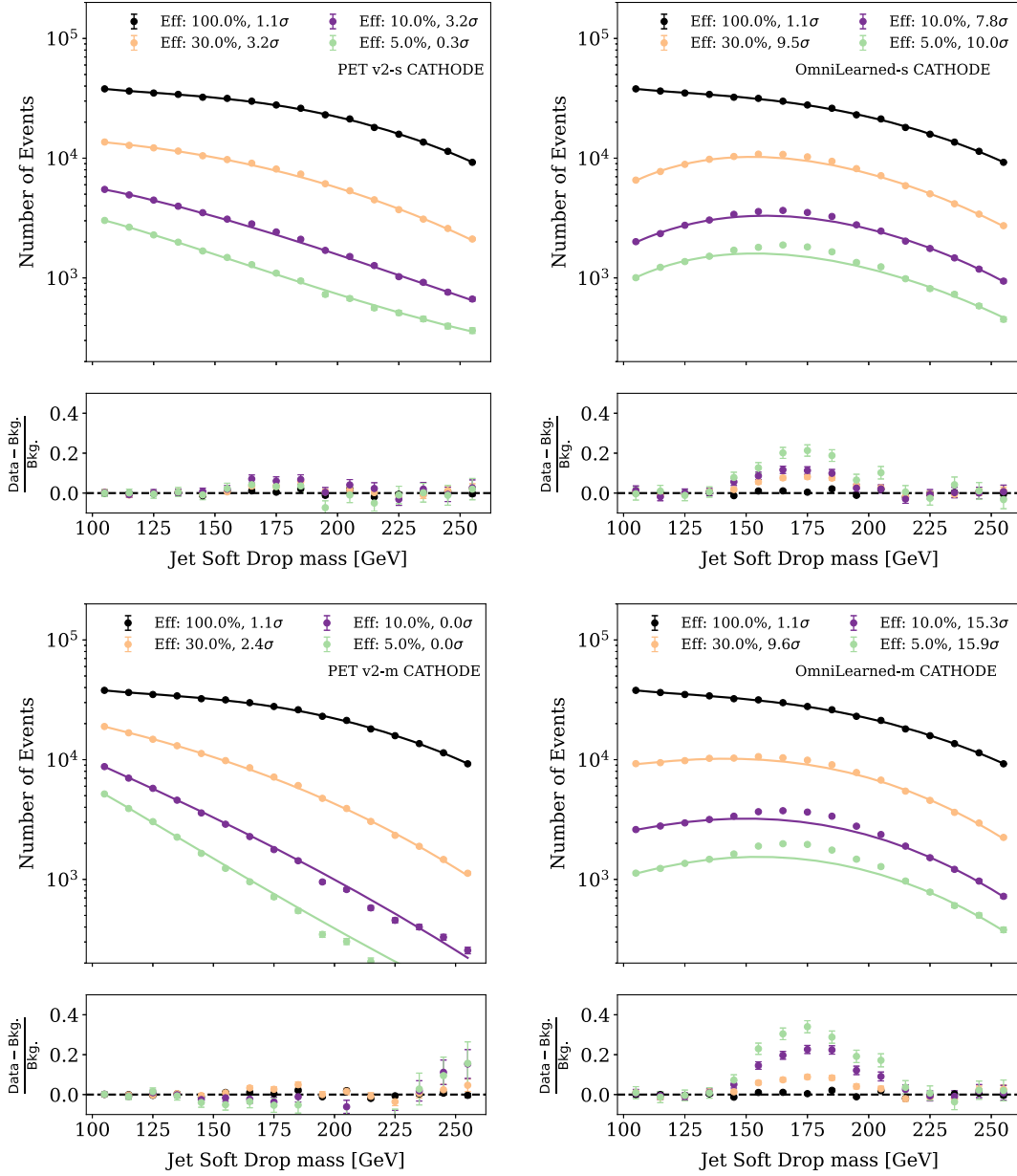


FIG. 4. Anomaly detection results using the CMS Open Data. Different model sizes (rows) for models trained from scratch (left column) or fine-tuned with OmniLearned (right column) are shown. Different thresholds of the anomaly score, resulting in different data efficiencies, are shown together with the expected sensitivity.

greater than 100 GeV and smaller than 300 GeV, while excluding the signal region. While the pretraining of OmniLearned does not include conditioning information, we encode the additional jet information consisting of the jet p_T , jet invariant and Soft Drop mass, and particle multiplicity as an additional “particle” to be concatenated with the real particles inside the jet after an embedding layer. The embedding layer is simply a fully connected layer with a nonlinear activation function that maps the inputs and their original dimensionality to the internal dimensionality of the representation used by the model.

To sample entire jets from the diffusion model, we first need to generate the conditioning information. To do this, we train a second generative model using the sidebands. First, we estimate the distribution of the Soft Drop mass using the sidebands, modeled as a smoothly falling distribution with four free parameters, commonly used in dijet resonance searches [94–96], parametrized as,

$$\frac{dN}{dm_j} = p_0 \frac{(1-x)^{p_1}}{x^{p_2+p_3 \ln(x)}}, \quad x = m_j/600 \text{ GeV}, \quad (9)$$

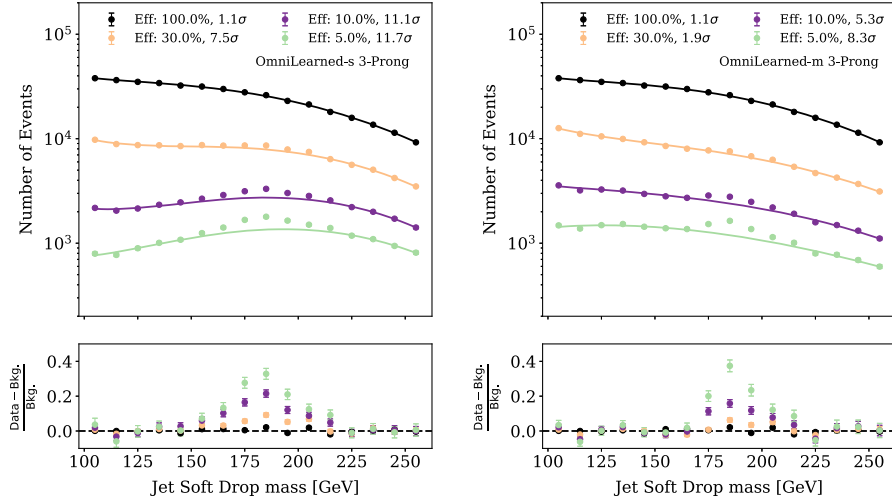


FIG. 5. Anomaly detection results using the CMS Open Data. The anomaly score is calculated directly from the classes used to pretrain OmniLearned using the ratio between the prediction for three-prong decays to QCD. Results for the small (left) and medium (right) models are shown.

with parameters p_i fixed from the fit to the sidebands. New jet mass values are then sampled in the signal region after proper normalization of the estimated probability density function. The Soft Drop mass values are used to condition a second diffusion model whose task is to generate three numbers: the jet p_T , jet invariant mass, and particle multiplicity. This second diffusion model is similar to the one used in Ref. [12], consisting of fully connected layers with nonlinear activations to encode the time, conditioning mass values, and the jet kinematic inputs. The full generation then proceeds as follows: first Soft Drop mass values are generated in the signal region following the distribution obtained by Eq. (9), then the diffusion model generates the jet level distributions, which in turn are used as conditions, together with the sampled Soft Drop masses, to generate background events in the signal region with the fine-tuned OmniLearned model. After the background estimation, we also use OmniLearned to fine-tune the classifier used to distinguish data from the background prediction and create the anomaly score. Anomaly detection results are shown in Fig. 4 for different model sizes, data efficiencies for the classifier selection, and the choice of using the pretrained model OmniLearned or training a model from scratch using the same datasets. The expected global significance is also listed, estimated as the ratio,

$$S = \frac{N - B}{\sqrt{N}}, \quad (10)$$

where the number of background events B is determined from the fit function [Eq. (9)] while N corresponds to the total number of events in the full mass range. If the number of expected background events matches or exceeds the number of data points, the significance is set to zero. We note that this significance is only approximate—it would be

interesting to extend it,⁵ possibly with strategies to remove cuts [87], in the future along with scanning the mass window to promote the benchmark dataset into a full search for new physics.

The models trained from scratch show only marginal sensitivity for different anomaly score thresholds, with the medium model not able to detect the signal (consistent with Ref. [97]). On the other hand, OmniLearned is able to successfully identify the signal above discovery threshold, with the medium model yielding higher significance.

The structure of OmniLearned allows for an alternative to Cathode-style methods. In particular, we can leverage the multiple classes used during the training of OmniLearned to directly search for anomalies. For example, we can use the classes associated to generic three-prong (the top quark decays into three quarks via an intermediate W boson) decay modes from the pretraining dataset, divided by the QCD prediction nodes as the anomaly score. Notice that even though OmniLearned has dedicated output nodes for top quarks, we use only generic three-prong decays without top quarks for this anomaly detection exercise. This strategy is similar to the one proposed in [33]. Even though this approach does not converge to the supervised classifier with increasing data (asymptotically optimal [98]), it may still be effective for a broad set of anomalies and the theory prior may help its performance at low signal fraction [82]. The sidebands are again used to determine the background distribution and evaluate the anomaly detection performance of the three-prong classifier in the signal region. Results for this approach are shown in Fig. 5.

⁵This would include a study of the false positive rate and the calibration of p values. One could also use simulations to compare with the known top quark production cross section.

For all model sizes the signal component is clearly visible. However, bigger model sizes lead to a small degradation of the model sensitivity. This is likely due to imperfect simulation details used during the pretraining. This application directly uses OmniLearned without additional fine-tuning, resulting in faster deployment.

VI. CONCLUSION AND OUTLOOK

In this paper, we have introduced the follow up to the OmniLearn model named OmniLearned, a general model architecture capable of learning a useful representation of jets that is easily adaptable to many downstream tasks and datasets. With the new model, we also introduce a machine-learning ready dataset consisting of 1×10^6 jets that are accessible from the software package. Before pretraining, the PET v2 model already shows improved performance compared to the previous iteration, achieving competitive results for jet tagging. The pretraining of PET v2 using the new combined dataset leads to the OmniLearned foundational model, capable of achieving superior performance in traditional benchmarks for jet classification, such as the community top tagging, while also pushing the performance in flavor tagging with realistic datasets from the ATLAS Collaboration. We also show that task-specific heads of OmniLearned can be successfully repurposed for different tasks, such as the use of the generative head to perform track origin prediction in the ATLAS flavor tagging dataset.

The same model is used for anomaly detection, this time applied to experimental data collected by the CMS Collaboration—the first foundation model applied to real collider data to the best of our knowledge. This application displays generalization power across detectors even when all experimental conditions are considered. In this case, we investigate two strategies to identify anomalies. First, we perform an extended, full-phase space bump hunt, where the background prediction is made using the generative capability of OmniLearned and an anomaly score is defined using the classifier capability of OmniLearned. We also explore an alternative option to leverage the multiple classes used to pretrain OmniLearned to create the anomaly score directly. This strategy has the benefit of directly using the foundational model without any additional fine-tuning. Both strategies are able to successfully identify the anomaly in the dataset, due to Lorentz-boosted top quarks, expected to be about 0.1% of the total events.

All model implementations and pretrained checkpoints for all model sizes are provided in the software package. Bigger models are shown to yield the best performance, both in terms of pretraining representation, evidenced by the anomaly detection results, as well as in terms of fine-tuning performance, seen by the different classification

tasks. However, fine-tuning the bigger models requires significant computing resources, making the smaller models more efficient and still more performant compared to models trained from scratch.

While we introduce OmniLearned as a foundational model for jet physics, we strongly believe that the methodology, and possibly even the pretrained weights, can be useful beyond jet physics. This includes the use of full event topologies and different collision systems, leading to exciting possibilities for future investigations.

ACKNOWLEDGMENTS

We thank Pradyun Hebbar, Joschka Birk, Runze Li, and Ibrahim Elsharkawy for helpful comments and testing the software environment during development. We also thank Kevin Grief for discussion on the use of the ATLAS flavor tagging dataset. We thank our colleagues from the H1 Collaboration for allowing us to use the simulated MC event samples. We also thank DESY-IT and the MPI für Physik for providing computing infrastructure and supporting the data preservation project of the HERA experiments. V.M. is supported by JST EXPERT-J, Japan Grant No. JPMJEX2509. B.N. is supported by the U.S. Department of Energy (DOE), Office of Science under Contract No. DE-AC02-76SF00515. This research used resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231 using NERSC Awards No. ERCAP0034229, No. HEP-ERCAP0021099, and No. HEP-ERCAP0028249.

DATA AVAILABILITY

The data that support the findings of this article are openly available [99].

APPENDIX: MODEL SIZES

In the main text we introduce the OmniLearned models using different model sizes. The main differences between each model size are the number of transformer layers, the size of the internal representation of the model, and the number of heads considered in the transformer blocks. The values used in this work are listed in Table V.

TABLE V. Hyperparameters used in OmniLearned.

Parameter	Small	Medium	Large
Transformer blocks	8	12	28
Transformer heads	8	16	32
Latent dimension	128	512	1024
Trainable weights	3M	58M	423M

- [1] R. Kogler *et al.*, *Rev. Mod. Phys.* **91**, 045003 (2019).
- [2] A. J. Larkoski, I. Moutl, and B. Nachman, *Phys. Rep.* **841**, 1 (2020).
- [3] S. Marzani, G. Soyez, and M. Spannowsky, *Looking Inside Jets: An Introduction to Jet Substructure and Boosted-Object Phenomenology*, Vol. 958 (Springer, New York, 2019).
- [4] M. Feickert and B. Nachman, [arXiv:2102.02770](#).
- [5] J. Birk, A. Hallin, and G. Kasieczka, *Mach. Learn. Sci. Technol.* **5**, 035031 (2024).
- [6] A. Hallin, in *2nd European AI for Fundamental Physics Conference* (2025).
- [7] P. Harris, M. Kagan, J. Krupa, B. Maier, and N. Woodward, *Phys. Rev. D* **111**, 032010 (2025).
- [8] T. Golling, L. Heinrich, M. Kagan, S. Klein, M. Leigh, M. Osadchy, and J. A. Raine, *Mach. Learn. Sci. Tech.* **5**, 035074 (2024).
- [9] M. Leigh, S. Klein, F. Charton, T. Golling, L. Heinrich, M. Kagan, I. Ochoa, and M. Osadchy, [arXiv:2409.12589](#).
- [10] J. Bardhan, R. Agrawal, A. Tilak, C. Neeraj, and S. Mitra, [arXiv:2502.03933](#).
- [11] V. Mikuni and B. Nachman, *Phys. Rev. D* **111**, 054015 (2025).
- [12] V. Mikuni and B. Nachman, *Phys. Rev. D* **111**, L051504 (2025).
- [13] J. de Favereau, C. Delaere, P. Demin, A. Giammanco, V. Lemaître, A. Mertens, and M. Selvaggi (DELPHES 3 Collaboration), *J. High Energy Phys.* **02** (2014) 057.
- [14] H. Qu, C. Li, and S. Qian, [arXiv:2202.03772](#).
- [15] S. Agostinelli *et al.* (GEANT4 Collaboration), *Nucl. Instrum. Methods Phys. Res., Sect. A* **506**, 250 (2003).
- [16] P. T. Komiske, E. M. Metodiev, and J. Thaler, *J. High Energy Phys.* **01** (2019) 121.
- [17] ATLAS Collaboration, Deep Sets based Neural Networks for Impact Parameter Flavour Tagging in ATLAS, Tech. Rep. ATL-PHYS-PUB-2020-014, CERN, Geneva, 2020.
- [18] E. A. Moreno, O. Cerri, J. M. Duarte, H. B. Newman, T. Q. Nguyen, A. Periwai, M. Pierini, A. Serikova, M. Spiropulu, and J.-R. Vlimant, *Eur. Phys. J. C* **80**, 58 (2020).
- [19] H. Qu and L. Gouskos, *Phys. Rev. D* **101**, 056019 (2020).
- [20] C. Shimmmin, [arXiv:2107.02908](#).
- [21] V. Mikuni and F. Canelli, *Eur. Phys. J. Plus* **135**, 463 (2020).
- [22] V. Mikuni and F. Canelli, *Mach. Learn. Sci. Technol.* **2**, 035027 (2021).
- [23] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, *J. Mach. Learn. Res.* **15**, 1929 (2014), <https://jmlr.org/papers/v15/srivastava14a.html>.
- [24] S. Goriely, A. Choplin, W. Ryssens, and I. Kullmann, in *Proceedings of the 28th International Nuclear Physics Conference*, Vol. 2586 (AIP Publishing, Melville, NY, USA, 2022), p. 012104.
- [25] M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. Barron, and R. Ng, in *Advances in Neural Information Processing Systems*, Vol. 33, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin (Curran Associates, Inc., Red Hook, NY, USA, 2020), pp. 7537–7547.
- [26] Y. Wu, K. Wang, C. Li, H. Qu, and J. Zhu, *Chin. Phys. C* **49**, 013110 (2025).
- [27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, [arXiv:1706.03762](#).
- [28] J. Zhu, X. Chen, K. He, Y. LeCun, and Z. Liu, in *Proceedings of the Computer Vision and Pattern Recognition Conference* (IEEE, Piscataway, NJ, USA, 2025), pp. 14901–14911.
- [29] T. Salimans and J. Ho, in *International Conference on Learning Representations* (OpenReview (ICLR proceedings are published electronically), Amherst, MA, USA, 2022).
- [30] V. Mikuni, B. Nachman, and M. Pettee, *Phys. Rev. D* **108**, 036025 (2023).
- [31] J. Y. Araz, V. Mikuni, F. Ringer, N. Sato, F. T. Acosta, and R. Whitehill, *Phys. Lett. B* **868**, 139694 (2025).
- [32] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le, [arXiv:2210.02747](#).
- [33] C. Li *et al.*, [arXiv:2405.12972](#).
- [34] O. Amram, L. Anzalone, J. Birk, D. A. Faroughy, A. Hallin, G. Kasieczka, M. Krämer, I. Pang, H. Reyes-Gonzalez, and D. Shih, *Mach. Learn. Sci. Technol.* **6**, 030601 (2025).
- [35] G. Aad *et al.* (ATLAS Collaboration), *J. Instrum.* **19**, P08018 (2024).
- [36] J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, O. Mattelaer, H. S. Shao, T. Stelzer, P. Torrielli, and M. Zaro, *J. High Energy Phys.* **07** (2014) 079.
- [37] T. Sjöstrand, S. Mrenna, and P. Z. Skands, *J. High Energy Phys.* **05** (2006) 026.
- [38] T. Sjöstrand, S. Ask, J. R. Christiansen, R. Corke, N. Desai, P. Ilten, S. Mrenna, S. Prestel, C. O. Rasmussen, and P. Z. Skands, *Comput. Phys. Commun.* **191**, 159 (2015).
- [39] A. Mertens, *J. Phys. Conf. Ser.* **608**, 012045 (2015).
- [40] M. Selvaggi, *J. Phys. Conf. Ser.* **523**, 012033 (2014).
- [41] M. Cacciari and G. P. Salam, *Phys. Lett. B* **641**, 57 (2006).
- [42] M. Cacciari, G. P. Salam, and G. Soyez, *Eur. Phys. J. C* **72**, 1896 (2012).
- [43] M. Cacciari, G. P. Salam, and G. Soyez, *J. High Energy Phys.* **04** (2008) 063.
- [44] G. Aad *et al.* (ATLAS Collaboration), *Eur. Phys. J. C* **70**, 823 (2010).
- [45] R. D. Ball *et al.*, *Nucl. Phys.* **B867**, 244 (2013).
- [46] A. Buckley, in *Proceedings of the 6th International Workshop on Multiple Partonic Interactions at the LHC* (World Scientific Publishing, Singapore, 2014), p. 29.
- [47] G. Aad *et al.* (ATLAS Collaboration), *Eur. Phys. J. C* **81**, 334 (2021).
- [48] A. J. Larkoski, S. Marzani, G. Soyez, and J. Thaler, *J. High Energy Phys.* **05** (2014) 146.
- [49] P. Berta, M. Spousta, D. W. Miller, and R. Leitner, *J. High Energy Phys.* **06** (2014) 092.
- [50] P. Berta, L. Masetti, D. W. Miller, and M. Spousta, *J. High Energy Phys.* **08** (2019) 175.
- [51] M. Cacciari, G. P. Salam, and G. Soyez, *Eur. Phys. J. C* **75**, 59 (2015).
- [52] H. Jung, *Comput. Phys. Commun.* **86**, 147 (1995).
- [53] K. Charchula, G. A. Schuler, and H. Spiesberger, *Comput. Phys. Commun.* **81**, 381 (1994).
- [54] R. Brun, F. Bruyant, M. Maire, A. C. McPherson, and P. Zancarini, CERN, (1987).
- [55] D. Britzger, S. Levonian, S. Schmitt, and D. South (H1 Collaboration), *EPJ Web Conf.* **251**, 03004 (2021).

- [56] M. Peez, Search for deviations from the standard model in high transverse energy processes at the electron proton collider HERA, Other thesis, Université de Lyon, 2003.
- [57] S. Hellwig, Untersuchung der $D^* - \pi_{\text{slow}}$ Double Tagging Methode in Charmanalysen, Master's thesis, Universität Hamburg, 2004.
- [58] B. Porthault, First measurement of charged and neutral current cross sections with the polarized positron beam at HERA II and QCD-electroweak analyses, Other thesis, Université Paris-Sud, 2005.
- [59] “Perlmutter system”, https://docs.nersc.gov/systems/perlmutter/system_details/, (accessed: 2022-05-04).
- [60] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, PyTorch: An imperative style, high-performance deep learning library, in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., Red Hook, NY, USA, 2019).
- [61] I. Loshchilov and F. Hutter, [arXiv:1608.03983](https://arxiv.org/abs/1608.03983).
- [62] X. Chen, C. Liang, D. Huang, E. Real, K. Wang, H. Pham, X. Dong, T. Luong, C.-J. Hsieh, Y. Lu *et al.*, Advances in Neural Information Processing Systems (Curran Associates, Inc., Red Hook, NY, USA, 2024).
- [63] A. Butter *et al.*, *SciPost Phys.* **7**, 014 (2019).
- [64] A. Bogatskiy, B. Anderson, J. T. Offermann, M. Roussi, D. W. Miller, and R. Kondor, [arXiv:2006.04780](https://arxiv.org/abs/2006.04780).
- [65] S. Gong, Q. Meng, J. Zhang, H. Qu, C. Li, S. Qian, W. Du, Z.-M. Ma, and T.-Y. Liu, *J. High Energy Phys.* **07** (2022) 030.
- [66] A. Bogatskiy, T. Hoffman, D. W. Miller, and J. T. Offermann, [arXiv:2211.00454](https://arxiv.org/abs/2211.00454).
- [67] A. Hammad and M. Nojiri, *J. High Energy Phys.* **06** (2024) 176.
- [68] J. Brehmer, V. Bresó, P. de Haan, T. Plehn, H. Qu, J. Spinner, and J. Thaler, *SciPost Phys.* **19**, 108 (2025).
- [69] G. Aad *et al.* (ATLAS Collaboration), *Nat. Commun.* **17**, 541 (2026).
- [70] Atlas Collaboration, “Atlas $t\bar{t}$ simulation for ml-based jet flavour tagging (jetset)”, CERN Open Data Portal, (2025) [10.7483/OPENDATA.ATLAS.QG8W.TO8P](https://opendata.cern.ch/record/10.7483/OPENDATA.ATLAS.QG8W.TO8P).
- [71] S. Alioli, P. Nason, C. Oleari, and E. Re, *J. High Energy Phys.* **06** (2010) 043.
- [72] S. Frixione, P. Nason, and C. Oleari, *J. High Energy Phys.* **11** (2007) 070.
- [73] P. Nason, *J. High Energy Phys.* **11** (2004) 040.
- [74] S. Frixione, P. Nason, and G. Ridolfi, *J. High Energy Phys.* **09** (2007) 126.
- [75] E. M. Metodiev, B. Nachman, and J. Thaler, *J. High Energy Phys.* **10** (2017) 174.
- [76] J. H. Collins, K. Howe, and B. Nachman, *Phys. Rev. Lett.* **121**, 241803 (2018).
- [77] J. H. Collins, K. Howe, and B. Nachman, *Phys. Rev. D* **99**, 014038 (2019).
- [78] G. Kasieczka *et al.*, *Rep. Prog. Phys.* **84**, 124201 (2021).
- [79] T. Aarrestad *et al.*, *SciPost Phys.* **12**, 043 (2021).
- [80] G. Karagiorgi, G. Kasieczka, S. Kravitz, B. Nachman, and D. Shih, [arXiv:2112.03769](https://arxiv.org/abs/2112.03769).
- [81] A. Hallin, J. Isaacson, G. Kasieczka, C. Krause, B. Nachman, T. Quadfasel, M. Schlaffer, D. Shih, and M. Sommerhalder, *Phys. Rev. D* **106**, 055006 (2021).
- [82] C. L. Cheng, G. Singh, and B. Nachman, *Phys. Rev. Lett.* **135**, 021801 (2025).
- [83] C. L. Cheng, R. Das, R. Li, R. Mastandrea, V. Mikuni, B. Nachman, D. Shih, and G. Singh, [arXiv:2506.00119](https://arxiv.org/abs/2506.00119).
- [84] G. Aad *et al.* (ATLAS Collaboration), *Phys. Rev. Lett.* **125**, 131801 (2020).
- [85] G. Aad *et al.* (ATLAS Collaboration), *Phys. Rev. D* **112**, 072009 (2025).
- [86] V. Chekhovsky *et al.* (CMS Collaboration), *Rep. Prog. Phys.* **88**, 067802 (2025).
- [87] R. Gambhir, R. Mastandrea, B. Nachman, and J. Thaler, *Phys. Rev. Lett.* **135**, 021902 (2025).
- [88] D. Shih, M. R. Buckley, L. Necib, and J. Tamanas, *Mon. Not. R. Astron. Soc.* **509**, 5992 (2021).
- [89] D. Shih, M. R. Buckley, and L. Necib, *Mon. Not. R. Astron. Soc.* **529**, 4745 (2024).
- [90] M. Pettee, S. Thanvantri, B. Nachman, D. Shih, M. R. Buckley, and J. H. Collins, *Mon. Not. R. Astron. Soc.* **527**, 8459 (2023).
- [91] D. Sengupta, S. Mulligan, D. Shih, J. A. Raine, and T. Golling, *Mon. Not. R. Astron. Soc.* **536**, 1104 (2024).
- [92] A. Hallin, D. Shih, C. Krause, and M. R. Buckley, [arXiv:2509.08064](https://arxiv.org/abs/2509.08064).
- [93] O. Knapp, O. Cerri, G. Dissertori, T. Q. Nguyen, M. Pierini, and J.-R. Vlimant, *Eur. Phys. J. Plus* **136**, 236 (2021).
- [94] T. Aaltonen *et al.* (CDF Collaboration), *Phys. Rev. D* **79**, 112002 (2009).
- [95] G. Aad *et al.* (ATLAS Collaboration), *Phys. Lett. B* **754**, 302 (2016).
- [96] V. Khachatryan *et al.* (CMS Collaboration), *Phys. Rev. Lett.* **117**, 031802 (2016).
- [97] E. Buhmann, C. Ewen, G. Kasieczka, V. Mikuni, B. Nachman, and D. Shih, *Phys. Rev. D* **109**, 055015 (2024).
- [98] B. Nachman and D. Shih, *Phys. Rev. D* **101**, 075042 (2020).
- [99] V. Mikuni and C. Harris, Viniciusmikuni/OmniLearned: 0.2.0, (2026) [10.5281/zenodo.18489564](https://zenodo.org/record/18489564).