BINARY JUNIPR: An Interpretable Probabilistic Model for Discrimination

Anders Andreassen,[1,*] Ilya Feige,[2,†] Christopher Frye,[2,‡] and Matthew D. Schwartz[3,§]
[1]*Department of Physics, University of California, Berkeley, California 94720, USA*
and Theoretical Physics Group, Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA
[2]*Faculty, 54 Welbeck Street, London W1G 9XS, United Kingdom*
[3]*Department of Physics, Harvard University, Cambridge, Massachusetts 02138, USA*

(Received 24 July 2019; published 31 October 2019)

JUNIPR is an approach to unsupervised learning in particle physics that scaffolds a probabilistic model for jets around their representation as binary trees. Separate JUNIPR models can be learned for different event or jet types, then compared and explored for physical insight. The relative probabilities can also be used for discrimination. In this Letter, we show how the training of the separate models can be refined in the context of classification to optimize discrimination power. We refer to this refined approach as BINARY JUNIPR. BINARY JUNIPR achieves state-of-the-art performance for quark-gluon discrimination and top tagging. The trained models can then be analyzed to provide physical insight into how the classification is achieved. As examples, we explore differences between quark and gluon jets and between gluon jets generated with two different simulations.

DOI: 10.1103/PhysRevLett.123.182001

Modern machine learning has already made impressive contributions to particle physics. Convolutional [1–6], recurrent and recursive networks [7–11], autoencoders [12–15], adversarial networks [16–18], and more have been shown effective in applications including quark-gluon jet discrimination, top tagging, and pileup removal. A key question that is beginning to be addressed is this: what is the optimal representation of the information in an event? Is it through analogy with images [1,2], natural-language processing [8,11], or set theory [19,20]? In many of these approaches, there is a competition between effectiveness in some task (e.g., pileup removal, jet classification) and interpretability of the neural network. An approach to machine learning for particle physics called JUNIPR [21] builds a separate network for each jet type using a physical representation of the information in the jet: the jet clustering tree. In [21] a method for construction and training of such a network was introduced. In this Letter, we show how the JUNIPR framework can be used in discrimination tasks, achieving state-of-the art classification power while maintaining physical interpretability.

JUNIPR begins by taking each jet in some sample and clustering it into a binary tree according to some deterministic algorithm. See Fig. 2 below for an example of such a tree. The algorithm can be physically motivated (like the k_T [22] or Cambridge-Aachen [23] algorithms) but does not have to be. In such a tree, the momenta of each mother branch is the sum of the momenta of her daughters. We denote the momenta of the particles in the jet by $\{p_1, \dots, p_n\}$ and the momenta in the clustering tree by $\{k_1^{(t)}, \dots, k_t^{(t)}\}$ at branching step t. To be concrete, at $t = 1$ we have $k_1^{(1)} = p_1 + \cdots + p_n$, at $t = n$ we have $\{k_1^{(n)}, \dots, k_n^{(n)}\} = \{p_1, \dots, p_n\}$, and at each branching in between, $\{k_1^{(t)}, \dots, k_t^{(t)}\} \to \{k_1^{(t+1)}, \dots, k_{t+1}^{(t+1)}\}$ involves a single $1 \to 2$ momentum splitting. JUNIPR learns to compute the probability $P_{\mathcal{J}}(\text{jet})$ of the jet, meaning the probability that the corresponding set of final state momenta $\{p_1, \dots, p_n\}$ would be found in the given sample. This probability can be factorized as a product over branching steps in the clustering tree:

$$P_{\mathcal{J}}(\text{jet}) = \left(\prod_{t=1}^{n-1} P^{(t)}(k_1^{(t+1)}, \dots, k_{t+1}^{(t+1)} | k_1^{(t)}, \dots, k_t^{(t)})\right) \times P^{(n)}(\text{end}|k_1^{(n)}, \dots, k_n^{(n)}) \tag{1}$$

To learn these probability distributions, JUNIPR introduces a quantity $h^{(t)}$ as a representation of $\{k_1^{(t)}, \dots, k_t^{(t)}\}$, i.e., the "state" of the jet at branching step t. JUNIPR learns to compute $h^{(t)}$ in training. In machine-learning language, $h^{(t)}$ is the autoregressive latent variable, which in our implementations is taken to be the latent state of a recurrent neural network. Then we can write, e.g.,

$$P^{(n)}(\text{end}|k_1^{(n)}, \dots, k_n^{(n)}) = P_{\text{end}}(\text{true}|h^{(n)}), \tag{2}$$

where $P_{\text{end}}(\text{true}|h^{(n)})$ is the binary probability that the clustering tree ends at branching step n.

JUNIPR further factorizes the branching probabilities of Eq. (1) into more intuitive probability distributions

$$\begin{aligned} P^{(t)}(k_1^{(t+1)},\ldots|k_1^{(t)},\ldots) = {} & P_{\text{end}}(\text{false}|h^{(t)}) \\ & \times P_{\text{mother}}(m^{(t)}|h^{(t)}) \\ & \times P_{\text{branch}}(k_{d_1}^{(t+1)}k_{d_2}^{(t+1)}|k_m^{(t)}h^{(t)}). \end{aligned} \tag{3}$$

Here, $P_{\text{end}}(\text{false}|h^{(t)})$ is the binary probability that the clustering tree does not end at branching step t, $P_{\text{mother}}(m^{(t)}|h^{(t)})$ is the discrete probability that tree-momentum $k_m^{(t)}$ will participate in the $1 \to 2$ branching at step t, and $P_{\text{branch}}(k_{d_1}^{(t+1)}k_{d_2}^{(t+1)}|k_m^{(t)}h^{(t)})$ is the distribution over daughters of the branching. Structuring the probabilistic model in terms of the product of these parts is essential to the interpretability of the model's output, as each part has separate physical meaning.

The latent state $h^{(t)}$ has access to the global content of the jet at branching step t, i.e., to all the momenta $\{k_1^{(t)},\ldots,k_t^{(t)}\}$. The factorization over branching steps is powerful, and useful to the extent that the $1 \to 2$ branching dynamics encoded in $P_{\text{branch}}(k_{d_1}^{(t+1)}k_{d_2}^{(t+1)}|k_m^{(t)}h^{(t)})$ are local, depending only weakly on $h^{(t)}$. Even if there were no evidence for this factorization in the training data (as was explored with "printer jets" in [21]), JUNIPR would still learn the probability distributions, but physical interpretability would be lost.

In [21], JUNIPR was trained to model jet dynamics via unsupervised learning. In that approach, the probabilistic model is learned by maximizing the log likelihood of $P_{\mathcal{J}}$ over the training data:

$$\text{log likelihood} = \sum_{\text{jets}} \log P_{\mathcal{J}}(\text{jet}), \tag{4}$$

where the sum is over jets $\{p_1,\ldots,p_n\}$ in the training set. We call this the "unary objective function." Despite being unsupervised, this approach can be used to discriminate between two jet types, say a and b. To accomplish this, one trains two separate JUNIPR models: $P_{\mathcal{J}}(\text{jet}|a)$ on a data set containing predominantly type-a jets and $P_{\mathcal{J}}(\text{jet}|b)$ on predominantly type-b jets. Discrimination between a and b is then achieved by thresholding the likelihood ratio $P_{\mathcal{J}}(\text{jet}|a)/P_{\mathcal{J}}(\text{jet}|b)$.

While discrimination by likelihood ratio is theoretically optimal in the perfect-model limit, it has been shown that deep neural networks classify out-of-distribution data poorly [24,25]. That is, e.g., the $P_{\mathcal{J}}(\text{jet}|a)$ model is not expected to behave well on type-b jets. It is thus advantageous in practice to refine the training for discrimination. By training directly for discrimination, JUNIPR can also focus model capacity on learning the often-subtle differences between type-a and type-b jets. In fact, JUNIPR's probabilistic nature makes supervised discrimination learning very straightforward. Assuming a mixed sample of both jet types, the probability that a given jet drawn at random belongs to class a is, through Bayes's theorem, given by

$$P(a|\text{jet}) = \frac{P(\text{jet}|a)P(a)}{P(\text{jet})}. \tag{5}$$

For binary discrimination, $P(a|\text{jet}) + P(b|\text{jet}) = 1$, so

$$P(a|\text{jet}) = \frac{P(\text{jet}|a)P(a)}{P(\text{jet}|a)P(a) + P(\text{jet}|b)P(b)}. \tag{6}$$

Here $P(a)$ and $P(b)$ are simply the composition fractions f_a and f_b of the mixed sample, while $P(\text{jet}|a)$ and $P(\text{jet}|b)$ can be computed using two separate JUNIPR networks as laid out in the paragraphs above. This leads directly to the binary cross-entropy objective function one should use to train JUNIPR for discrimination:

$$\begin{aligned} \mathcal{L} = & \sum_{a\text{jets}} \log \frac{P_{\mathcal{J}}(\text{jet}|a)f_a}{P_{\mathcal{J}}(\text{jet}|a)f_a + P_{\mathcal{J}}(\text{jet}|b)f_b} \\ & + \sum_{b\text{jets}} \log \frac{P_{\mathcal{J}}(\text{jet}|b)f_b}{P_{\mathcal{J}}(\text{jet}|a)f_a + P_{\mathcal{J}}(\text{jet}|b)f_b}, \end{aligned} \tag{7}$$

where the sums extend over type-a and type-b jets in the training data, respectively. We call training with this objective function "BINARY JUNIPR." Note that BINARY JUNIPR still learns the probabilities for type-a and type-b jets and still trains the same neural-network functions; however, it uses a more effective objective function for discrimination applications. We also note that training can easily be generalized to multiclass classification.

As a test of the advantage that the binary objective function provides over its unary counterpart, we applied BINARY JUNIPR to the discrimination of quark and gluon jets. We used a mixed sample of 10^6 PYTHIA quark jets and 10^6 PYTHIA gluon jets from the data set at [26], see also [19,27]. We set aside 10^5 jets of each type into a test set, 10^5 for validation, and used the remaining 80% of the jets for training. For the JUNIPR models, $P_{\mathcal{J}}(\text{jet}|\text{quark})$ and $P_{\mathcal{J}}(\text{jet}|\text{gluon})$, we used an LSTM of dimension 30 to model $h^{(t)}$ and separate feed-forward networks, each with a single hidden layer of dimension 10, to model P_{end}, P_{mother}, and P_{branch}. (The BINARY JUNIPR architecture is available at [28] with example code.)

We began by pretraining the two JUNIPR models using the original unary objective function of Eq. (4). We followed the same training schedule as in [21], but scaled down the number of epochs by a factor of 5 because this data set is larger than the one used there. Pretraining took about five hours on a 16-core CPU server for each model. After pretraining, we optimized the binary objective function of Eq. (7) using Adam with standard settings [29] and the following batch-size schedule:

Schedule	1 epoch	5 epochs	10 epochs	10 epochs
batch size	10	100	1000	2000

This segment of training took 12 hours on a 16-core CPU server. BINARY JUNIPR parameters were decided upon by evaluating the AUC (area under the ROC curve) on the validation set 10 times per epoch and choosing the model that achieved the maximal AUC during the final 10 training epochs. Note that different hyperparameters might be appropriate for different applications.

In Fig. 1 we show the quark-versus-gluon Significance Improvement Curve [30] (SIC), $(\varepsilon_S/\sqrt{\varepsilon_B})$, achieved by BINARY JUNIPR and compare it to recent results with previous state-of-the-art discriminants: a CNN approach based on jet images [3] (with architecture from [19]) and particle flow networks [19]. One can see that BINARY JUNIPR offers a small-but-significant advantage. Quantitatively, BINARY JUNIPR achieves an AUC of 0.8986 ± 0.0004, as compared to 0.8911 ± 0.0008 for particle flow networks, and 0.8799 ± 0.0008 for the CNN. (Each reported number is the mean and semi-interquartile range over 10 trainings.) Unary JUNIPR, trained with Eq. (4), performs significantly worse than the other methods, achieving an AUC of 0.6968 ± 0.0008. This demonstrates the importance of training JUNIPR with the binary objective function of Eq. (7) for classification.

As a second experiment, we trained and tested BINARY JUNIPR for boosted top-jet identification. We used the same architecture and training schedule that were optimized for quark-versus-gluon discrimination. In doing so, we obtain a sense of the performance one might expect from BINARY JUNIPR without specialized hyperparameter tuning. The training, validation, and test data for this experiment are taken from [7]. We found that untuned BINARY JUNIPR comes close to state-of-the-art top discrimination. Specifically, JUNIPR achieves an AUC of 0.9810 ± 0.0002 as compared to 0.9819 ± 0.0001 attained using particle flow networks [19], and 0.9848 reported for ParticleNet [20]; all significantly outperform traditional boosted top-tagging methods [31]. For a recent overview of machine learning in top tagging, see [32].

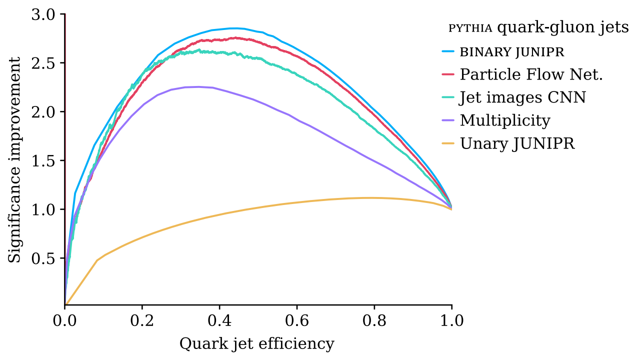

FIG. 1. Significance improvement $(\varepsilon_Q/\sqrt{\varepsilon_G})$ as a function of ε_Q for quark-gluon discrimination. BINARY JUNIPR is compared to a particle flow network [19], a CNN using jet images [19], constituent multiplicity, and unary JUNIPR.

Next we discuss the interpretability of JUNIPR models. As discussed below Eq. (3), each component of JUNIPR's output has a well-defined physical meaning. Moreover, the output is structured along a physically motivated binary tree, defined by clustering the momenta in a jet. One can thus decompose JUNIPR's prediction, say $P_{\mathcal{J}}(\text{top}|\text{jet})$ as in Eq. (6), visually along the clustering tree. In Fig. 2, we show the clustering tree for an easily classifiable top jet drawn from the mixed top-QCD test set. In the figure, we label the tth node with $P^{(t)}(\text{top}|\text{jet})$, i.e., the probability that the jet is top type, given only the information present at branching step t; this is computed with BINARY JUNIPR by substituting Eq. (3) into Eq. (6). One can see, for example, that the three-prong structure characteristic of $t \to W^+ b \to u\bar{d}b$ contributes to large $P_{\mathcal{J}}(\text{top}|\text{jet})$. Quantitatively, this results in the two hard branchings, with $P^{(t)}(\text{top}|\text{jet}) = 0.72$ and 0.71, dominating the prediction. By analyzing such trees, one can develop intuition for which branchings are most decisive in classifying different types of jets.

To be concrete, let us return to the BINARY JUNIPR model used to create Fig. 1, which learned to discriminate quark and gluon jets from PYTHIA. Much is already known about the difference between quark and gluon jets: gluon jets are known to be bigger, with larger multiplicity and larger shape parameters such as mass and width [33,34]. Although many methods exist for quark-gluon discrimination, including other machine-learning approaches [3,11], it is not clear how well these methods will work on actual data. In particular, it is known that real gluon jets are more similar to real quark jets than PYTHIA leads us to believe [35]. In particular, it is the modeling of gluon jets that seems most inaccurate. An alternative generator, HERWIG, produces and gluon jets that are more similar to its quark jets [36]. Thus, we also considered a secondary challenge: determine how PYTHIA and HERWIG gluon jets differ. To explore their differences, we trained a second BINARY JUNIPR model to discriminate PYTHIA8.226 and HERWIG7.1.4 gluon jets using 10^6 samples of each from [27,37].

Figure 3 shows another visualization, complementary to Fig. 2, of exploring how JUNIPR discriminates. The top row of Fig. 3 shows how JUNIPR separates PYTHIA quarks from PYTHIA gluons, and the bottom row shows how JUNIPR separates PYTHIA gluons from HERWIG gluons. In the middle column, the overall probability that JUNIPR uses for discrimination is decomposed into branching steps t, averaged over all jets of the given class. From this, we see

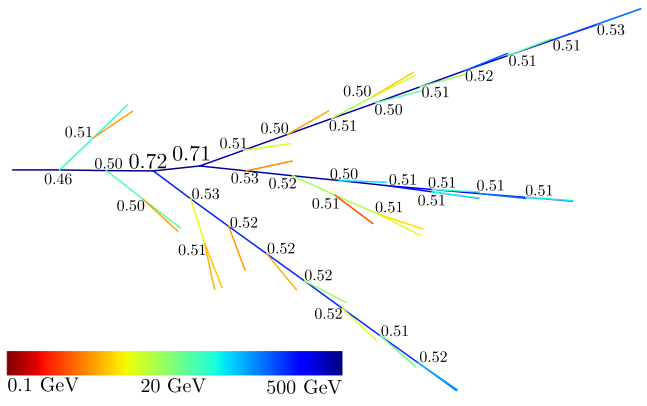

FIG. 2. BINARY JUNIPR tree for jet drawn from mixed top-QCD test set. BINARY JUNIPR predicts "top" with high probability: $P_{\mathcal{J}}(\text{top}|\text{jet}) = 0.99$. Each node is labeled with the probability that the jet is top type, given only the information at that branching. Planar angles correspond to 3D opening angles between clustered momenta, and color corresponds to energy. The final factor corresponding to the tree's true end is not shown: $P^{(n)} = 0.52$; see Eq. (2).

that near $t = 20$–50 there is roughly three times the quark-gluon discrimination power per branching step as for $t = 1$–10. This echoes the well-known fact that multiplicity allows one to separate quark and gluon jets better than perturbatively calculable observables sensitive to only the first few splittings [33]. The lower-middle plot shows that differences in PYTHIA and HERWIG gluon jets are more uniformly spread over branching steps.

Not only can JUNIPR break discrimination power down into branching steps; JUNIPR can further decompose classification probability into components at each branching. These components are displayed in the right column of Fig. 3; there are discrete components, such as whether branchings should end, as well as the energy z and angles θ, ϕ, δ of the branching itself. While multiplicity (P_{end}) is the main driver of performance for quark-gluon discrimination, the angle θ also contributes significantly over a wide range of branchings, echoing the importance of jet width in this context. For the PYTHIA-HERWIG task, both the angle θ and energy fraction z play a significant role in discrimination on early branchings, and multiplicity becomes important on later branchings.

It is interesting that a significant fraction of the difference between PYTHIA and HERWIG results from the way energy and angles are distributed early on in the clustering trees. Early branchings are controlled primarily by perturbative elements of the simulated parton showers. This suggests that a substantial portion of the difference between PYTHIA and HERWIG gluon jets may be driven by the parton-shower implementations, rather than exclusively by the modeling of nonperturbative effects. To gain further insight into the importance of nonperturbative effects like hadronization in discrimination, JUNIPR could be upgraded to include quantum numbers of final state particles—a straightforward next step.

In [21], JUNIPR was introduced as a new framework for unsupervised machine learning in particle physics that prioritizes interpretability. Given a jet, i.e., a set of momenta, JUNIPR learns to compute the probability of that jet, i.e., how consistent the distribution of momenta is with the training data. In this Letter, we used the same probabilistic framework as in [21], but we augmented the training to learn subtle differences between two

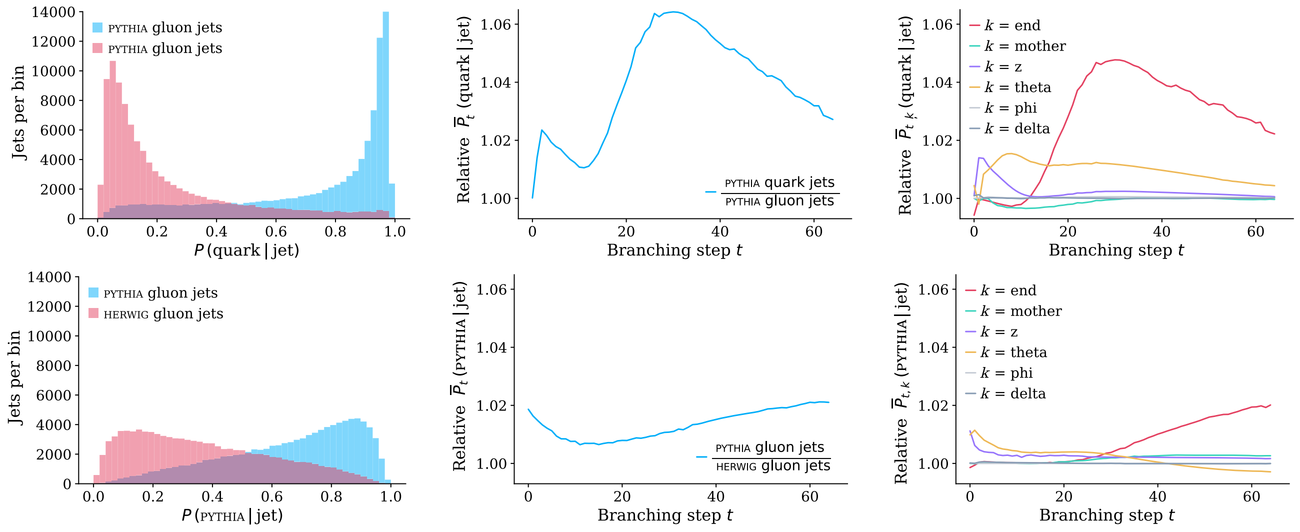

FIG. 3. Quark-gluon (top row) and PYTHIA-HERWIG discrimination (bottom row) with BINARY JUNIPR. Here we will refer to quark jets and PYTHIA jets as "signal," and to gluon jets and HERWIG jets as "background." The left column shows the binary probability with which JUNIPR predicts each jet is a signal jet. The middle column breaks these probabilities down by branching step in the clustering tree. Specifically, the plots show the ratio of $P_t(\text{signal}|\text{jet})$, averaged over signal jets in the numerator and background jets in the denominator. The right column breaks these ratios down further by branching component.

samples, an enhancement we call BINARY JUNIPR. We demonstrated both its effectiveness and interpretability, using quark-gluon jets, boosted top jets, and Monte Carlo generator dependence as examples. It is satisfying that demanding interpretability does not lead to a loss in effectiveness: BINARY JUNIPR discriminates at levels competitive with the best machine-learning methods available.

While these case studies were all simulation based, there is a straightforward path to repeating these exercises on collider data. Although real data do not come with truth labels, there are established methods for working with mixed samples [38,39] that can be adapted to JUNIPR without much modification. Then one could use a data or simulation BINARY JUNIPR model to understand deficiencies in simulations. One could also use insights derived from BINARY JUNIPR trees to judge whether predictions should be trusted experimentally (was information below experimental resolution deemed important?) or to design new calculable observables (sensitive to previously overlooked decisive branchings). Having interpretable methods opens the door to whole new approaches to understanding data from particle colliders.

We thank P. Komiske and A. Parthak for assistance with the samples. This work is supported in part by the U.S. Department of Energy under Contract No. DE-SC0013607. Computations were performed on the CORI supercomputing resources at NERSC, Harvard's Odyssey cluster, and the Faculty Platform for machine learning.

*andersja@berkeley.edu
†ilya@faculty.ai
‡chris.f@faculty.ai
§schwartz@physics.harvard.edu

[1] L. de Oliveira, M. Kagan, L. Mackey, B. Nachman, and A. Schwartzman, Jet-images deep learning edition, J. High Energy Phys. 07 (2016) 069.
[2] P. Baldi, K. Bauer, C. Eng, P. Sadowski, and D. Whiteson, Jet substructure classification in high-energy physics with deep neural networks, Phys. Rev. D **93**, 094034 (2016).
[3] P. T. Komiske, E. M. Metodiev, and M. D. Schwartz, Deep learning in color: Towards automated quark/gluon jet discrimination, J. High Energy Phys. 01 (2017) 110.
[4] P. T. Komiske, E. M. Metodiev, B. Nachman, and M. D. Schwartz, Pileup mitigation with machine learning (PUMML), J. High Energy Phys. 12 (2017) 051.
[5] G. Kasieczka, T. Plehn, M. Russell, and T. Schell, Deep-learning top taggers or the end of QCD, J. High Energy Phys. 05 (2017) 006.
[6] S. Macaluso and D. Shih, Pulling out all the tops with computer vision and deep learning, J. High Energy Phys. 10 (2018) 121.
[7] A. Butter, G. Kasieczka, T. Plehn, and M. Russell, Deep-learned top tagging with a Lorentz layer, SciPost Phys. **5**, 028 (2018).
[8] G. Louppe, K. Cho, C. Becot, and K. Cranmer, QCD-Aware recursive neural networks for jet physics, J. High Energy Phys. 01 (2019) 057.
[9] T. Cheng, Recursive neural networks in quark/gluon tagging, Comput. Software Big Sci. **2**, 3 (2018).
[10] S. Egan, W. Fedorko, A. Lister, J. Pearkes, and C. Gay, Long Short-Term Memory (LSTM) networks with jet constituents for boosted top tagging at the LHC, arXiv:1711.09059.
[11] K. Fraser and M. D. Schwartz, Jet charge and machine learning, J. High Energy Phys. 10 (2018) 093.
[12] J. H. Collins, K. Howe, and B. Nachman, Anomaly Detection for Resonant New Physics with Machine Learning, Phys. Rev. Lett. **121**, 241803 (2018).
[13] M. Farina, Y. Nakai, and D. Shih, Searching for new physics with deep autoencoders, arXiv:1808.08992.
[14] A. Blance, M. Spannowsky, and P. Waite, Adversarially-trained autoencoders for robust unsupervised new physics searches, arXiv:1905.10384.
[15] T. S. Roy and A. H. Vijay, A robust anomaly finder based on autoencoder, arXiv:1903.02032.
[16] M. Paganini, L. de Oliveira, and B. Nachman, CaloGAN: Simulating 3D high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks, Phys. Rev. D **97**, 014021 (2018).
[17] L. de Oliveira, M. Paganini, and B. Nachman, Learning particle physics by example: Location-aware generative adversarial networks for physics synthesis, Comput. Software Big Sci. **1**, 4 (2017).
[18] M. Paganini, L. de Oliveira, and B. Nachman, Accelerating Science with Generative Adversarial Networks: An Application to 3D Particle Showers in Multilayer Calorimeters, Phys. Rev. Lett. **120**, 042003 (2018).
[19] P. T. Komiske, E. M. Metodiev, and J. Thaler, Energy flow networks: Deep sets for particle jets, J. High Energy Phys. 01 (2019) 121.
[20] H. Qu and L. Gouskos, ParticleNet: Jet Tagging via Particle Clouds, arXiv:1902.08570.
[21] A. Andreassen, I. Feige, C. Frye, and M. D. Schwartz, Junipr: A framework for unsupervised machine learning in particle physics, Eur. Phys. J. C **79**, 102 (2019).
[22] S. D. Ellis and D. E. Soper, Successive combination jet algorithm for hadron collisions, Phys. Rev. D **48**, 3160 (1993).
[23] Y. L. Dokshitzer, G. D. Leder, S. Moretti, and B. R. Webber, Better jet clustering algorithms, J. High Energy Phys. 08 (1997) 001.
[24] A. Nguyen, J. Yosinski, and J. Clune, Deep neural networks are easily fooled: High confidence predictions for unrecognizable images, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, New York, 2015), https://doi.org/10.1109/CVPR.2015.7298640.
[25] A. Bendale and T. E. Boult, Towards open set deep networks, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, New York, 2016), https://doi.org/10.1109/CVPR.2016.173.
[26] See energyflow.network for more information.
[27] P. Komiske, E. Metodiev, and J. Thaler, Pythia8 quark and gluon jets for energy flow, 2019, https://doi.org/10.5281/zenodo.3164691.

[28] See github.com/andersjohanandreassen/JUNIPR for more information.

[29] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv:1412.6980.

[30] J. Gallicchio, J. Huth, M. Kagan, M. D. Schwartz, K. Black, and B. Tweedie, Multivariate discrimination and the Higgs $+ W/Z$ search, J. High Energy Phys. 04 (2011) 069.

[31] D. E. Kaplan, K. Rehermann, M. D. Schwartz, and B. Tweedie, Top Tagging: A Method for Identifying Boosted Hadronically Decaying Top Quarks, Phys. Rev. Lett. **101**, 142001 (2008).

[32] A. Butter *et al.*, The machine learning landscape of top taggers, SciPost Phys. **7**, 014 (2019).

[33] J. Gallicchio and M. D. Schwartz, Quark and Gluon jet substructure, J. High Energy Phys. 04 (2013) 090.

[34] J. Gallicchio and M. D. Schwartz, Quark and Gluon Tagging at the LHC, Phys. Rev. Lett. **107**, 172001 (2011).

[35] G. Aad *et al.* (ATLAS Collaboration), Light-quark and gluon jet discrimination in pp collisions at $\sqrt{s} = 7$ TeV with the ATLAS detector, Eur. Phys. J. C **74**, 3023 (2014).

[36] P. Gras, S. Höche, D. Kar, A. Larkoski, L. Lönnblad, S. Plätzer, A. Siódmok, P. Skands, G. Soyez, and J. Thaler, Systematics of quark/gluon tagging, J. High Energy Phys. 07 (2017) 091.

[37] A. Pathak, P. Komiske, E. Metodiev, and M. Schwartz, Herwig7.1 quark and gluon jets, 2019, https://doi.org/10.5281/zenodo.3066475.

[38] P. T. Komiske, E. M. Metodiev, B. Nachman, and M. D. Schwartz, Learning to classify from impure samples with high-dimensional data, Phys. Rev. D **98**, 011502 (2018).

[39] E. M. Metodiev, B. Nachman, and J. Thaler, Classification without labels: Learning from mixed samples in high energy physics, J. High Energy Phys. 10 (2017) 174.