Creating simple, interpretable anomaly detectors for new physics in jet substructure

Layne Bradshaw[*] and Spencer Chang[†]

Department of Physics and Institute for Fundamental Science, University of Oregon, Eugene, Oregon 97403, USA

Bryan Ostdiek

Department of Physics, Harvard University, Cambridge, Massachusetts 02318, USA and The NSF AI Institute for Artificial Intelligence and Fundamental Interactions

(Received 16 March 2022; accepted 11 July 2022; published 12 August 2022)

Anomaly detection with convolutional autoencoders is a popular method to search for new physics in a model-agnostic manner. These techniques are powerful, but they are still a "black box," since we do not know what high-level physical observables determine how anomalous an event is. To address this, we adapt a recently proposed technique by Faucett *et al.* [Phys. Rev. D **103**, 036020 (2021).], which maps out the physical observables learned by a neural network classifier, to the case of anomaly detection. We propose two different strategies that use a small number of high-level observables to mimic the decisions made by the autoencoder on background events, one designed to directly learn the output of the autoencoder, and the other designed to learn the difference between the autoencoder's outputs on a pair of events. Despite the underlying differences in their approach, we find that both strategies have similar ordering performance as the autoencoder and independently use the same six high-level observables. From there, we compare the performance of these networks as anomaly detectors. We find that both strategies perform similarly to the autoencoder across a variety of signals, giving a nontrivial demonstration that learning to order background events transfers to ordering a variety of signal events.

DOI: 10.1103/PhysRevD.106.035014

I. INTRODUCTION

Many analyses have been carried out at the LHC to look for new physics beyond the Standard Model, but unfortunately these have yet to yield statistically significant deviations from the expected background. This may indicate that there is no new physics to be found in the data or, more optimistically, it may be a result of not looking for the right signals. There remain many well-motivated models to search for, but designing and carrying out dedicated analyses for each quickly becomes intractable. This motivates the need for broad, model-agnostic searches. The advent of modern machine learning has seen the creation of a variety of unsupervised anomaly detection techniques, all capable of searching for new physics with no reliance on a particular signal model. See Ref. [1] for a recent review of anomaly detection and unsupervised techniques.

Anomaly detection techniques rely on an ability to characterize the background in some way, with the hope that this characterization does not generalize to out-of-distribution events, thus making signal events appear "anomalous." Broadly speaking, anomaly detection can be split into two categories, depending on how similar one expects the signal and background to look. If they are expected to look similar, one has to work to exploit differences in the underlying probability distributions, and many techniques have been developed to highlight those differences [2–24]. However, one often expects there to be qualitative differences between signal and background. In that case, there are a variety of methods that can determine whether events are anomalous or not on an event-by-event basis [25–55].

Machine learning (ML) techniques, including unsupervised anomaly detection, typically make use of low-level, high-dimensional data. This is in contrast to human-engineered strategies, which tend to use high-level, low-dimensional data. When the two perform equally well on a given task, we tend to assume that the ML strategy must have used some combination of its low-level inputs to

[*]layneb@uoregon.edu
[†]chang2@uoregon.edu

create an approximation of the high-level variables used by humans. It could be, however, that the ML strategy has found an alternative that is just as efficient. Unfortunately, the "black box" nature of ML techniques make it difficult to understand what the machine is actually learning. This problem is only amplified when the ML strategy outperforms the human-engineered one. Has the machine learned a simple observable humans did not consider or has it perhaps found something new?

There have been efforts to understand a neural network by using existing high level observables [56–60], as well as "knowledge distillation" techniques to gain insights about complex networks by analyzing simpler ones [61–64]. In a recent paper (Ref. [65]), a promising iterative technique was introduced to build an interpretable classifier. This classifier mimics a "black box" deep neural network classifier, where the mimicker's inputs consists of a limited set of human-interpretable high-level variables (see also [66,67]). In this paper, we extend this technique to anomaly detectors by presenting two strategies for mapping the low-level information utilized by an anomaly detector into a handful of simple to understand high-level observables. As a concrete example, we attempt to mimic both the decisions and performance of an anomaly detector based on a convolutional autoencoder, which is trained on background jet images. The convolutional autoencoder then helps to iteratively select high-level observables that serve as the inputs to the mimicker networks. As our pool of high-level observables, we use the *energy flow polynomials* [68] because they form a basis for all infrared- and collinear-safe observables.

We introduce two strategies to mimic an autoencoder. The first strategy, the high-level network, uses a small number of high-level observables to match the autoencoder's anomaly score on an event-by-event basis. The other strategy, the paired neural network, is tasked with using a potentially different set of observables to learn to make the same ordering decisions as the autoencoder. Given a pair of events, the paired neural network learns which of the two was deemed to be less anomalous by the autoencoder. Note that like the convolutional autoencoder we want to mimic, both the paired and high-level neural networks are only trained on background events and so are unsupervised with respect to signal events. Despite their philosophical differences, we find that both strategies agree on which high-level observables are useful for ordering background events like the autoencoder. These two strategies also have comparable performance, where we find that they both make the same ordering decisions as the autoencoder ∼83% of the time.

Since these networks are unsupervised, applying these networks as anomaly detectors allows us to test whether the decision ordering on background events transfers to signal events. Interestingly, for seven of the eight different signals we consider, we find that the mimickers perform as well or better as anomaly detectors than the autoencoder. Thus, this shows that it is possible to create interpretable anomaly detectors that have a limited number of high-level inputs without compromising performance. This reduction of complexity is an obvious advantage for experimental applications of anomaly detection, reducing work needed for variable validation and determination of systematic uncertainties. Theoretically, this result gives insights into the features of a QCD jet image which are harder to compress into a lower-dimensional latent space.

This paper is outlined as follows. In Sec. II, we describe the Monte Carlo generated dataset, as well as the relevant selection criteria and preprocessing. Section III starts by describing the details of the convolutional autoencoder. We then review all of the pieces needed to mimic the autoencoder—the pool of high-level observables we use to explain the autoencoder, a metric to determine how similar the decisions of two networks are, the details of our two simplified anomaly detectors, and the iterative procedure we use to construct the mimickers from the pool of high-level observables. We present our results in Sec. IV, detailing the construction and performance of the mimickers. Finally we conclude in Sec. V. Details of the simulated events and network training hyperparameters appear in the appendices.

II. DATASETS

In this section, we briefly describe the simulated datasets we use in this study. In particular, our focus is on anomaly detection in boosted jets at the LHC. We utilize the publicly available datasets provided by Ref. [38], using QCD dijet events [69] as background and W, top, and Higgs jets [70] as the anomalous events. We consider four different W masses, $m_W = 59, 80, 120, 174$ GeV, two different top masses, $m_t = 80, 174$ GeV, and two different Higgs masses, $m_h = 20, 80$ GeV. Note that when $m_t = 80$ GeV, the mass of the decay product W is set to 20 GeV. The full simulation details are given in Appendix A. These signals give a broad range of signals with varying amounts of substructure (two to four prongs), which will prove useful when testing the ability of our anomaly detectors.

These datasets contain approximately 700,000 QCD dijet events and 100,000 events for each of the W, top, and Higgs signals. After applying a p_T cut (see Appendix A), we are left with $\sim 150,000$ QCD events and $\sim 30,0000$ events for each of the anomalous signals We use $2/3$ of the QCD dijet events for training the autoencoder, with the remaining $1/3$ being reserved for testing and validation. We are not considering training on real data at this point, so we do not include the possibility of contamination in the background set from signal samples when training the autoencoder. However, previous work has shown that autoencoders are robust to up to ∼10% signal contamination [28–30,71].

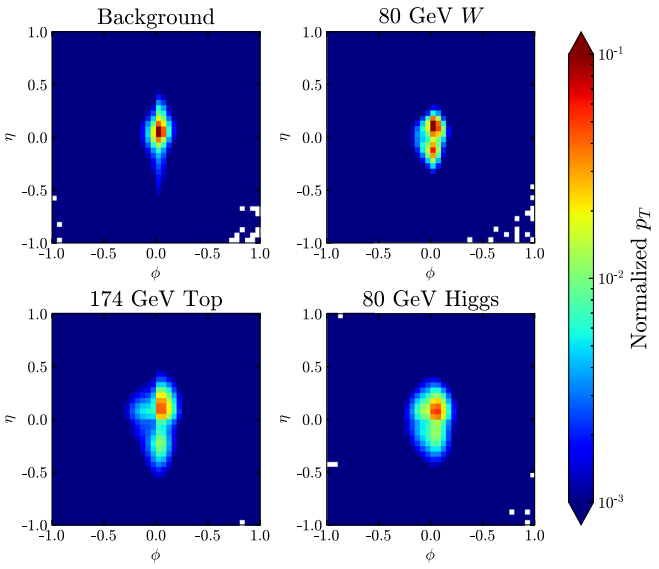


FIG. 1. The average jet image for the background, 80 GeV W, 174 GeV top, and 80 GeV Higgs. Note that the Higgs bosons are pair produced from the decay of a heavier Higgs, leading to potentially four prongs in the large-radius jet.

Our procedure for preprocessing the raw four vectors into images follows that outlined in Ref. [72] and is implemented with the EnergyFlow package [73]. For the leading jet in each event, we boost and rotate along the beam direction, such that the p_T weighted centroid lies at $(\eta, \phi) = (0, 0)$. The jet is then rotated about its centroid until its principal axis lies along the vertical. Finally, the jet is reflected about the horizontal and vertical axes so that the maximum intensity lies in the upper-right quadrant. Only after centering, rotating, and reflecting the jet do we pixelate the image. Our final pixelated images are 40×40, covering $\Delta\eta = \Delta\phi = 2.0$. The last step of our preprocessing procedure is to divide by the total p_T in the image. This final normalization step ensures that each image has the same scale, which helps with training. Figure 1 shows the average jet image for the background and three representative signals—the 80 GeV W, 174 GeV top, and 80 GeV Higgs.

III. METHODOLOGY

While neural networks have been used for classification and anomaly detection with great success, they are often viewed as black boxes, leading one to wonder what information they are using to match or outperform traditional techniques. With this in mind, the authors of Ref. [65] showed that modern classification networks are able to be mimicked by interpretable networks using a few high level physics variables as inputs. In this work, we adapt this method to the task of anomaly detection. In order to do this, we first need a good anomaly detector to mimic with physics variables.

A. Creating a target anomaly detector with a convolutional autoencoder

The anomaly detector we chose is a convolutional autoencoder (hereafter referred to as the AE). Given an input image, the AE is tasked with encoding the image down into a smaller latent space, then reconstructing the original image from its latent space representation. The idea behind compressing the data to a smaller representation is that it forces the network to learn what is important about the jet image, while ignoring noisy or less crucial aspects. The hope is that when the autoencoder is applied to anomalous data, the important characteristics will be different, and thus the image will be poorly encoded, leading to a decoded image that is quite different from the initial image. Thus, we can distinguish between the background data and the anomalous signal data by the size of the reconstruction error. AEs were first introduced to the high energy community as anomaly detectors in Refs. [27–29].[1]

The architecture of our AE is shown in Fig. 2 and is described below. The encoder consists of multiple layers. The first two layers are a set of five 3×3 pixel convolutional filters. We use a stride of one and pad the output to keep the same height and width as the original image. After each convolution we apply an exponential linear unit (ELU) activation [74]. Following these convolutions, the representation is down sampled with a 2×2 max pooling layer, leading to a height and width of 20 pixels. This reduced image is then passed through another two convolutional layers with five filters before being passed through a final convolutional layer with a single filter. This final 20×20 image is then flattened and connected to a dense layer with 100 nodes, which is in turn connected to our 32-dimensional latent space. We chose a 32-dimensional latent space, as that is where we found the performance of the AE as an anomaly detector began to saturate.

The decoder mirrors the encoder and consists of a dense layer with 100 nodes, followed by another dense layer with 400 nodes. Both of these dense layers use the ELU activation function. The output of this layer is then reshaped into a 20×20 image, and is then passed through two convolutional layers with five filters each. All of the convolutional layers in the decoder use a 3×3 convolutional kernel and the ELU activation function, with the exception of the last convolutional layer in the decoder, which uses the SOFTMAX activation function along the pixel dimension so that the sum of the pixel intensities is unity. These are then upsampled with a transposed convolutional layer to 40×40, passed through a convolutional layer with five filters, and finally passed through one last

[1]Often, AEs can be improved with variational autoencoders (VAEs), in which the latent space representation becomes a distribution, rather than a single point. As a proof of principle, we use the simpler AE, and leave the extension to VAEs for further study.

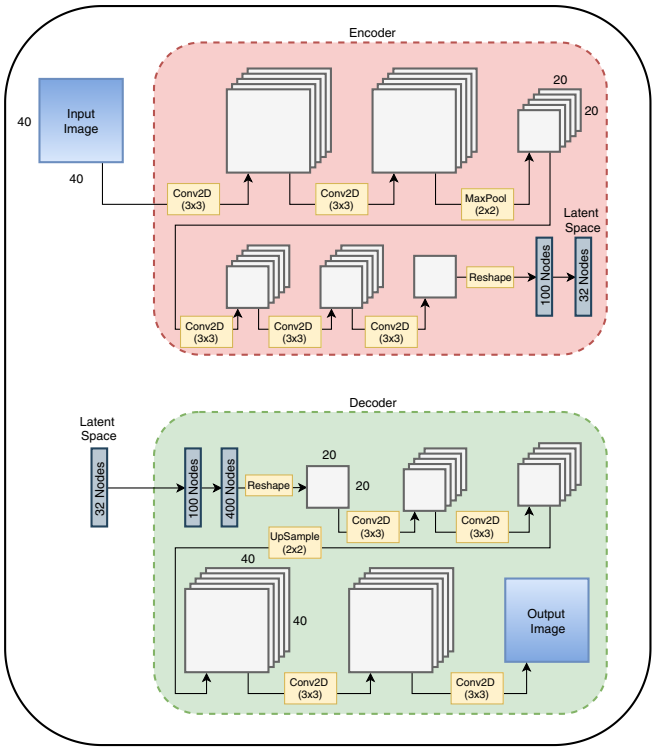


FIG. 2. The architecture of the convolutional AE. The AE consists of two separate networks, an encoder that compresses the original image down to a smaller latent space, and a decoder tasked with recreating the original image from the latent space representation.

convolutional layer to create the output image. We train the AE to reproduce QCD jet images, by minimizing the mean squared error of their reconstruction. Explicitly, this is given as

$$L_{\mathrm{AE}} = \frac{1}{N_i N_p} \sum_{k}^{N_i} \sum_{j}^{N_p} (f_A(I_k^j) - I_k^j)^2, \tag{1}$$

where N_i is the total number of images, N_p is the number of pixels in each image, I_k^j is the jth pixel of the kth input image, and $f_A(I_k^j)$ is the AE's reconstruction of that pixel for that input image. The training details for the AE are provided in Appendix B. Our AE, along with all of the other neural network architectures discussed in Sec. III are implemented with KERAS [75] using the TensorFlow [76] backend.

Figure 3 shows some examples of how the trained AE can act as an anomaly detector. The left panels display the distribution of the reconstruction errors as the anomaly score for the background training set as well as three different anomalous signals. At first glance, the reconstruction errors are very small, but this is explained by the normalization and the sparsity of our jet images. Because each image is normalized to sum to one, all pixels

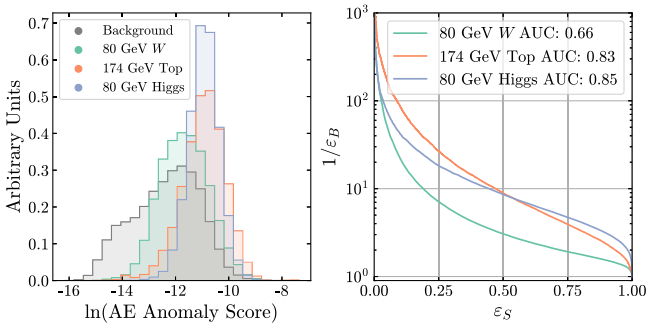


FIG. 3. The AE's performance as an anomaly detector on three of the anomalous signals, the 80 GeV W, the 174 GeV top, and the 80 GeV Higgs. Note that the Higgs bosons are pair produced from the decay of a heavier Higgs, leading to potentially four prongs in the large-radius jet. The left panel shows the normalized distribution of the log of the AE's anomaly score for the background and each of the signals. The right panel shows the ROC curves for each signal.

have a value of less than one. The images are also very sparse, so most pixels are identically 0, and the network is very good at predicting that. When we take the mean squared error over the pixels, we actually average over the number of pixels, so the number of pixels with no intensity leads to a very good average reconstruction. Importantly, we see that the background distribution is at lower scores than the signal distributions. The encoder has never seen jets with inherent substructure from the decay of a heavy resonance, so it does not recognize the important information to encode into the latent space, and the decoder therefore performs worse when reconstructing the images. The right panel displays the receiver operating characteristic (ROC) curves for these three signals. While the W is harder for the AE to distinguish from the background, the top and Higgs jets have decent area under the ROC curve (AUC) scores.

As we have seen, our constructed AE is capable of detecting jets which are different from the QCD background it was trained on. In the next section we build up a method to mimic the ordering decisions the AE makes using physics observables.

B. Mimicking the target anomaly detector

As shown in the previous section, the AE is able to tag various signals as being different from QCD. However, it is unclear what information in the event image is being used to do this. In order to mimic the behavior of the AE, we need a few ingredients. The first is a wide set of physics observables which could possibly explain the anomaly detector. For these, we use the energy flow polynomials, described in detail in Sec. III B 1. Next, we use the idea of decision ordering to select which observables are important as described in Sec. III B 2. Finally, we need a flexible function which can use the physics observables to produce

an anomaly score which mimics that of the AE. We describe two complementary methods that achieve this goal. The first method, a paired neural network, is a neural network which takes in the physics observables from two events at the same time and is trained to determine which event had the worse reconstruction error from the AE. We construct this in such a way that, at inference, we can feed in a single event and get an anomaly score. This technique is described in Sec. III B 3. The second method, a high-level neural network, instead takes in only a single event at a time and is trained to regress the reconstruction error of the AE for that event. This second method is described in Sec. III B 4.

1. High-level observables

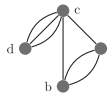
Since there is no way to know which human-constructed, high-level observables will be relevant *a priori*, we need to rely on using a basis of observables. To that end, we make use of the energy flow polynomials (EFPs) [68], a formally infinite set of jet substructure observables inspired by previous work on energy correlation functions [77–82]. The EFPs form a discrete linear basis for all infrared- and collinear-safe (IRC-safe) observables and are defined in terms of the momentum fraction, z_a, and pairwise angular distances, θ_{ab}. The EFPs are computed using the four-momentum of each particle in the jet, where z_a is the momentum fraction carried by particle a, and θ_{ab} is the pairwise angular distance between particles a and b. Each EFP is conveniently represented by a multigraph, using the following correspondences:

$$\text{each node a} \leftrightarrow \sum_{a=1}^{N} z_a \tag{2}$$

and

$$\text{each k-fold edge between nodes a and b} \leftrightarrow (\theta_{ab})^k. \tag{3}$$

As an example, we have



$$= \sum_{a=1}^{N}\sum_{b=1}^{N}\sum_{c=1}^{N}\sum_{d=1}^{N} z_a z_b z_c z_d \theta_{ab}^2 \theta_{ac}\theta_{bc}\theta_{cd}^3. \tag{4}$$

In this example, we have labeled the nodes for clarity, but we will not do so for future graphs. To build some intuition for this framework, we note that the fully connected graphs with N vertices correspond to the N-point energy correlation functions.

The EFPs corresponding to each multigraph can be modified with a pair of parameters, (κ,β), which determine the precise meaning of z_a and θ_{ab}. More specifically,

$$z_a^{(\kappa)} = \left(\frac{p_{T_a}}{\sum_b p_{T_b}}\right)^{\kappa}, \tag{5}$$

$$\theta_{ab}^{(\beta)} = (\Delta\eta_{ab}^2 + \Delta\phi_{ab}^2)^{\beta/2}, \tag{6}$$

where p_{T_a} is the transverse momentum of particle a, $\Delta\eta_{ab}$ is the difference in pseudorapidity between particles a and b, and $\Delta\phi_{ab}$ is the difference in azimuthal angle between particles a and b. The original IRC-safe EFPs require $\kappa = 1$. While there are well-motivated reasons to explore a broader space of observables at the cost of IR and/or C safety [83–85], we restrict ourselves to only IRC-safe observables in this work. For our iterative procedure to mimic the AE, we choose $\kappa = 1$, $\beta = 1$, and consider all EFPs with degree (i.e., the number of edges) $d \leq 5$. With these parameters, we have a total of 102 EFPs to explore.

2. Decision ordering

To create an interpretable alternative to the AE, we will iteratively add EFP observables as inputs to the mimicking networks. To compare how well a network (or EFP input) orders events relative to the AE, we use a series of metrics implemented in Ref. [65]. Here we briefly summarize these metrics. Given two decision functions, $f(x)$ and $g(x)$, the *decision ordering* (DO) for a pair of events x_1 and x_2 is defined as

$$\mathrm{DO}[f,g](x_1,x_2) = \Theta([f(x_1)-f(x_2)][g(x_1)-g(x_2)]), \tag{7}$$

where $\Theta(x)$ is the Heaviside theta function, and we choose $\Theta(0) = 1$. Here, we can think of $f(x)$ as being the anomaly score/reconstruction error for the AE and $g(x)$ being the output of one of our methods. Later, we will also use $f(x) = \mathrm{AE}(x)$ and $g(x) = \mathrm{EFP}(x)$ to determine which EFP observables to include for our mimickers. A DO of 1 means that f and g agree that one event is more anomalous than another; a DO of 0 indicates the two methods disagree on which event is more anomalous. If two decision functions have $\mathrm{DO} = 1$ for all possible pairs x_1 and x_2, then the two are effectively identical decision functions on the domain tested.

To create a summary statistic, we then average the DO over all possible pairs, weighted by the underlying distributions that x_1 and x_2 are drawn from. The resulting statistic, the *average decision ordering* (ADO), is given by

$$\mathrm{ADO}[f,g] = \int \mathrm{d}x_1 \mathrm{d}x_2 p_1(x_1) p_2(x_2) \mathrm{DO}[f,g](x_1,x_2). \tag{8}$$

This evaluates to 1 if both decision functions order every possible pair of events in the same manner (making them equivalent decision functions), 0 if they order the pairs in the opposite manner, and $\frac{1}{2}$ if there is no consistency to the way the decision functions order the events. Due to

computing constraints, we could not compute the ADO on the entirety of the background training set. Instead, when computing the ADO, we choose 10,000 events at random, and then evaluate on the $\binom{10,000}{2} \sim 5 \times 10^7$ pairs of events.

We now follow the black-box guided search strategy from Ref. [65] to iteratively construct neural networks whose decision functions should become better and better approximations of the AEs. We start by training a neural network, NN_0, on some initial set of observables, $X_0 = (m_J, p_T)$. We will later describe the two possible architectures for NN_0, but for now it is enough to say it aims to produce decision functions that mimic the AE on background events. We then compute the ADO between NN_0 and the AE, and isolate all of the pairs of events misordered by NN_0. From our set of high-level observables, O, we then want to find the observable $O_1 \in O$ with the highest ADO on the pairs misordered by NN_0.[2] We then train a new neural network, NN_1, whose input observables are $X_1 = X_0 \cup O_1$. Given its inputs, we would expect NN_1 to have a decision function that more closely resembles that of the AE—and consequently, a higher ADO compared to NN_0—since it has access to the same information NN_0 had, as well as information that can help order the pairs misordered by NN_0.

From here, we continue to iterate using the remaining observables in O. On the nth iteration, we start by finding the observable $O_n \in O$ with the highest ADO on the pairs misordered by $\mathrm{NN}_{n-1}(X_{n-1})$ that is not already part of X_{n-1}. We then build a new set of inputs, $X_n = X_{n-1} \cup O_n$, and train a new neural network, NN_n on X_n. At each iteration, we expect the ADO between the neural network and AE to increase, since the neural network we construct on the nth iteration has access to all of the same information available to the previous network, as well as a new observable O_n that helps order the events misordered by the $(n-1)$th neural network.

Now that we have described both the physics observables and the general method for choosing which observables to give the networks, we describe the two network architectures in more detail.

3. Paired neural network

Our first attempt to mimic the AE is an approach we call the paired neural network (PNN). The aim of the paired neural network is to mimic the AE by learning to predict the relative anomaly score between two events. To do this, the PNN takes pairs of events as its input and classifies which has a larger anomaly score. This is in contrast to other methods such as trying to match the AE's output or anomaly score on an event-by-event basis. In general, classifiers are easier to train, so this seems like a promising method.

[2]If the ADO of an observable is less than 0.5, then we take 1-ADO, since a highly anticorrelated variable is also useful.

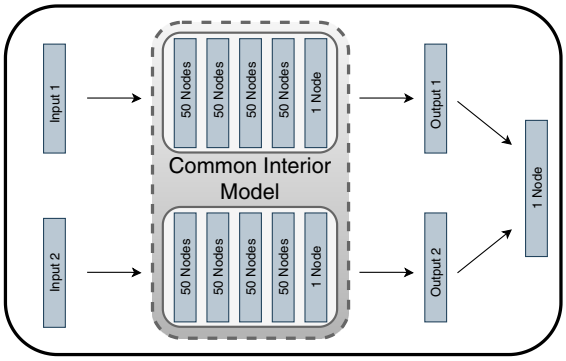


FIG. 4. The architecture of the paired neural network. The interior model consists of four hidden layers each with 50 nodes and using the ELU activation function. The interior model outputs a single node for each input and uses the RELU activation function. The final output of the model is a single node which is the difference between the two interior model outputs and uses a sigmoid activation function. Our input data are the jet's mass, p_T, and up to 14 EFPs.

Figure 4 shows the PNN architecture. Both events are fed through the same interior model in parallel. This is shown in the image as the "common interior model." The interior model consists of four hidden layers with 50 nodes each, and the ELU activation function is used for all layers. The interior model produces a single output for each input event, and this single output node uses the RELU activation. The motivation for this is to think of the output for each event as its own anomaly score. Within the larger PNN, we then subtract these two output anomaly scores from each other. If the first event is more anomalous, then the result should be negative, and if the second is more anomalous, then the result will be positive. The larger the difference in scores should tell us about the networks confidence in the relative ordering. Finally, to turn this into a classification problem, we apply the sigmoid function to the interior model difference, mapping large negative numbers to 0 and large positive values to 1. If the anomaly scores are the same (the difference is 0) the sigmoid gives a value of 0.5.

To train the network, we continue the idea of classification and minimize the binary cross entropy given by

$$L_{\mathrm{PNN}} = -\frac{1}{N}\sum_{k}^{N}[y_k \ln(f_P(X_k)) + (1-y_k)\ln(1-f_P(X_k))], \tag{9}$$

where k represents a specific pair of events, where the order matters. The value of y_k is the truth "label" for the pair of events as determined by the AE, i.e., $y_k = 0(1)$ if the AE determines the event in input 1 to be more (less) anomalous than the event in input 2, and $f_P(X_k)$ is the PNN's output for the pair of events. Appendix B provides the training details for the PNN.

After training the PNN on ∼250,000 pairs of events, we extract the interior model for use on single events. Thus, even though the training procedure requires pairs of events and was trained as a classifier, the interior model provides a function which takes in observables from a single event and outputs an anomaly score.

4. High-level neural network

The PNN described in the last section does not attempt to learn the actual anomaly score of the AE, but only the relative difference in the anomaly score between pairs of events. We also introduce a method that specifically aims to mimic the actual anomaly score of the AE. We call this network the high-level neural network (HLN). In practice, the anomaly score (reconstruction error) from the AE spans many orders of magnitude, so we found better results when the HLN is trained to predict the log of the anomaly score rather than the score itself.

We find that a relatively simple neural network is able to achieve the task of reproducing the loss of the AE. Figure 5 shows the architecture we use for the HLN. The HLN consists of four hidden layers, with each hidden layer having 50 nodes. The final output of the network is a single node. All of the nodes in the hidden layers use the ELU activation function.

To train the HLN, we minimize the mean squared error between the (log of the) anomaly score of the AE and the output of the HLN. Specifically, we use a loss function of

$$L_{\mathrm{HLN}} = \frac{1}{N}\sum_{k}^{N}\left[f_H(X_k) - \ln\left(\frac{1}{N_p}\sum_{j}^{N_p}(f_A(I_k^j) - I_k^j)^2\right)\right]^2, \tag{10}$$

where $f_H(X_k)$ is the HLN's output given some input data X_k and $f_A(I_k^j)$ is the AE's output given a pixel j in an image I_k^j for the kth event. When using the HLN as an anomaly detector, we use the model's output as the anomaly score. See Appendix B for the HLN training details.

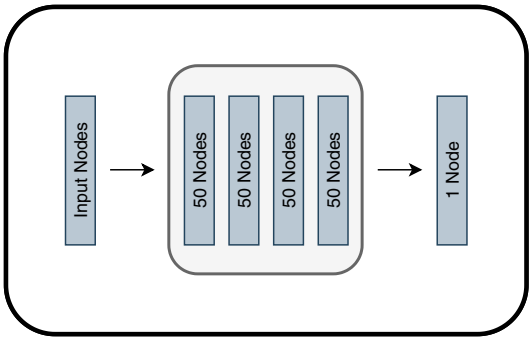


FIG. 5. The architecture of the high-level neural network. This network consists of four hidden layers, with each having 50 nodes and using the ELU activation function. The network output is a single node. Like the PNN, our input data are the jet's mass, p_T, and up to 14 EFPs.

IV. RESULTS

In the previous section, we outlined two different architectures we could use to iteratively build neural networks whose decision functions would more closely resemble the AE's decision function. Here, we provide the results of the iterative procedure and analyze the specific EFPs that are selected to mimic the anomaly detector. We will find that the EFPs selected are composite observables built out of only six prime EFP factors. We show that using only the prime components gives very similar results. Finally, we demonstrate that using the EFPs with a traditional anomaly detection technique, the isolation forest, gives very poor results. The failure of the isolation forest when provided with the same basic physics observables highlights the benefits of using our mimicker networks.

A. Background decision ordering

We start our iterative process by training both a PNN and HLN on jet mass and p_T for QCD events in the training set and then compute the ADO for each model. Of the $\sim 5 \times 10^7$ pairs of events we use to compute the ADO, both the initial PNN and HLN correctly order ∼72% of the events relative to the reconstruction error of the AE. Next, we take all of the pairs which are misordered and compute the ADO between all 102 EFPs and the AE. On this first iteration, we find that the observable with the highest ADO for both networks is EFP 2, given by

 $= \sum_{a,b=1}^{N} z_a z_b \theta_{ab}^2.$ (11)

This observable is then added to the list of inputs. So in the next iteration the input for each event is given by $(m_J, p_T, \mathrm{EFP}\,2)$. We then repeat this process 14 more times, recording both the ADO of each network, as well as which EFP has the largest ADO for the pairs of events which are misordered by the respective networks.

Figure 6 shows the result of this iterative process. The solid lines show the ADO of the models we used to determine the next best observable to add; the shaded band shows the maximum and minimum value of the ADO for each model after recalculating the ADO an additional 50 times at each iteration using a different set of $\sim 5 \times 10^7$ pairs of events. We also created PNN and HLN models trained on m, p_T, and all $d \leq 5$ EFPs. The ADOs of these two models agree to three significant digits and thus is plotted as the single dashed line in the panel. Since they use all of the EFPs, this line gives a sense of the highest ADO each model is capable of achieving, given our set of observables. The blue "+" and orange "×" will be discussed in Sec. IV C. There are a few key takeaways

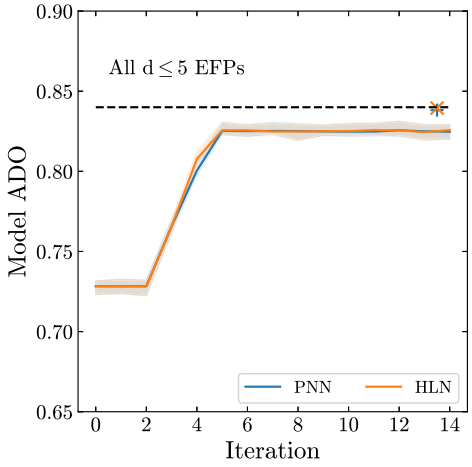


FIG. 6. The ADOs for each PNN and HLN. The center line shows the ADO of the model that was used to select the EFPs. The shaded bands show the maximum and minimum ADO values obtained when recalculating the ADO an additional 50 times, using a different set of pairs of events each time. The x axis denotes the iteration step of the iterative process. See Table I for the multigraph and mathematical representations of the selected EFPs and the iteration step at which they were added. The blue "+" (orange "×") shows the ADO of a PNN (HLN) trained on only the five prime EFPs picked out by each method [see Eq. (12)]. The ADO of each model trained on m, p_T, and all of the $d \leq 5$ EFPs is the same to three significant digits, and is plotted as a single dashed line.

from these plots. By the time the ADOs start to plateau, both the HLN and PNN are correctly ordering 83% of the pairs of events in the QCD sample relative to the AE. For the first two iterations, the model ADOs do not change. Looking at Table I, we see that the first two EFPs are EFP 2 and $[\mathrm{EFP}\,2]^2$, which are proportional to m^2/p_T^2 and m^4/p_T^4. Since the initial inputs to both the PNN and HLN are mass and p_T, these observables contain no new information, and thus it makes sense that the model ADO does not improve. This redundancy of information follows since the EFPs are a linear basis of substructure observables, whereas our neural networks can utilize nonlinear combinations of its inputs. Despite their underlying philosophical differences—the HLNs are trying to match the AE's anomaly score, while the PNN is trying the match the DO of the AE—both methods select the same set of 14 EFPs in the same order. In Table I, we list the multigraph and mathematical expression corresponding to each of these EFPs as well as the iteration step in which they were added. The agreement of the PNN and HLN approaches gives us confidence that these observables are important to detect jets which do not look like typical QCD jets. Also, since by the last iteration, the PNN and HLN have nearly reached the ADO of the dashed line, it suggests that the decision ordering of our mimickers has almost converged to what is possible with our set of EFPs.

B. Anomaly detection

While both the HLN and PNN have demonstrated the ability to mimic the AE's anomaly score on QCD events, it is unclear if matching the decision ordering on in-distribution events will generalize to out-of-distribution events. In other words, having mimicked the AE on QCD background events with HLNs and PNNs, we must test if this decision ordering transfers to boosted jet signals by comparing the AE, PNN, and HLN as anomaly detectors. To determine how well each network performs as an anomaly detector we use a popular metric, the AUC.

Figure 7 shows how the HLN and PNN on their final iteration compare to the autoencoder on the same three signals as Fig. 3. The left panels show the normalized distributions of each network's anomaly scores for the background and three of the signals. The right panel then shows all of the ROC curves for each model on each signal. We can see that both the HLN and PNN do a good job of mimicking the anomaly detector on events with higher anomaly scores. But the long tails in each of the background distributions indicate that the HLN and PNN struggle to match the AE on less anomalous events, explaining their poorer background rejection at low signal efficiency.

Figure 8 shows how the mimickers perform on all eight signals described in Sec. II at each step of the iterative progress. The dashed black line in each panel shows the AUC when using the reconstruction error of the AE as the anomaly score. The blue and orange curves show the results of the PNN and HLN, respectively, as a function of the number of iterations for selecting extra observables. The solid center lines denote the AUC of the model used to select observables in the iterative process. The shaded bands show the maximum and minimum AUCs when retraining each network ten additional times, to give us a sense of how stable the training is. The bands are quite narrow, indicating that the results are robust to training uncertainties.

Like we saw with the ADOs in Fig. 6, the HLN and PNN perform similarly, despite their different approaches. For both the decision ordering and the AUCs, the results start to plateau around the fifth iteration. When the HLN and PNN AUC scores begin to plateau, we see that the value is similar to the AUC of the AE. This indicates that the HLN and PNN are performing comparably to the AE when all three are acting as anomaly detectors. It is surprising that mimicking the decision ordering on the in-distribution (QCD) events seems to also generalize to the relative differences between the signals and the background. Some of the mimicking networks even exceed the anomaly detection capability of the AE they are trying to mimic for certain signals.

For some signals—specifically the 20 GeV Higgs, 80 GeV W, 120 GeV W, and 174 GeV W—we see a drop in AUC around the third iteration for both the PNN

TABLE I. The EFP multigraphs and corresponding expressions for each of the EFPs selected by both the HLN and PNN. In the last two columns, we list the iteration step where the PNN or HLN selects the corresponding EFP.

EFP no.	EFP multigraph	EFP expression	PNN iteration	HLN iteration
1		$\sum_{a,b=1}^{N} z_a z_b \theta_{ab}$	5	5
2		$\sum_{a,b=1}^{N} z_a z_b \theta_{ab}^2$	1	1
54		$\sum_{a,b,c,d=1}^{N} z_a z_b z_c z_d \theta_{ab} \theta_{cd}$	6	6
57		$\sum_{a,b,c,d=1}^{N} z_a z_b z_c z_d \theta_{ab}^2 \theta_{cd}^2$	2	2
65		$\sum_{a,b,c,d,e=1}^{N} z_a z_b z_c z_d z_e \theta_{ab}^2 \theta_{cd} \theta_{de}^2$	3	3
70		$\sum_{a,b,c,d,e,f=1}^{N} z_a z_b z_c z_d z_e z_f \theta_{ab} \theta_{cd} \theta_{ef}$	7	7
85		$\sum_{a,b,c,d,e,f=1}^{N} z_a z_b z_c z_d z_e z_f \theta_{ab} \theta_{cd}^2 \theta_{ef}^2$	4	4
86		$\sum_{a,b,c,e,d,f,g=1}^{N} z_a z_b z_c z_d z_e z_f z_g \theta_{ab} \theta_{ac} \theta_{de} \theta_{fg}$	13	13
94		$\sum_{a,b,c,e,d,f,g=1}^{N} z_a z_b z_c z_d z_e z_f z_g \theta_{ab} \theta_{ac} \theta_{bc} \theta_{de} \theta_{fg}$	11	11
95		$\sum_{a,b,c,d,e,f,g,h=1}^{N} z_a z_b z_c z_d z_e z_f z_g z_h \theta_{ab} \theta_{cd} \theta_{ef} \theta_{gh}$	8	8
97		$\sum_{a,b,c,d,e,f,g,h=1}^{N} z_a z_b z_c z_d z_e z_f z_g z_h \theta_{ab} \theta_{bc} \theta_{cd} \theta_{ef} \theta_{gh}$	12	12
99		$\sum_{a,b,c,d,e,f,g,h=1}^{N} z_a z_b z_c z_d z_e z_f z_g z_h \theta_{ab}^2 \theta_{cd} \theta_{ef} \theta_{gh}$	14	14
100		$\sum_{a,b,c,d,e,f,g,h,i=1}^{N} z_a z_b z_c z_d z_e z_f z_g z_h z_i \theta_{ab} \theta_{ac} \theta_{de} \theta_{fg} \theta_{hi}$	10	10
101		$\sum_{a,b,c,d,e,f,g,h,i,j=1}^{N} z_a z_b z_c z_d z_e z_f z_g z_h z_i z_j \theta_{ab} \theta_{cd} \theta_{ef} \theta_{gh} \theta_{ij}$	9	9

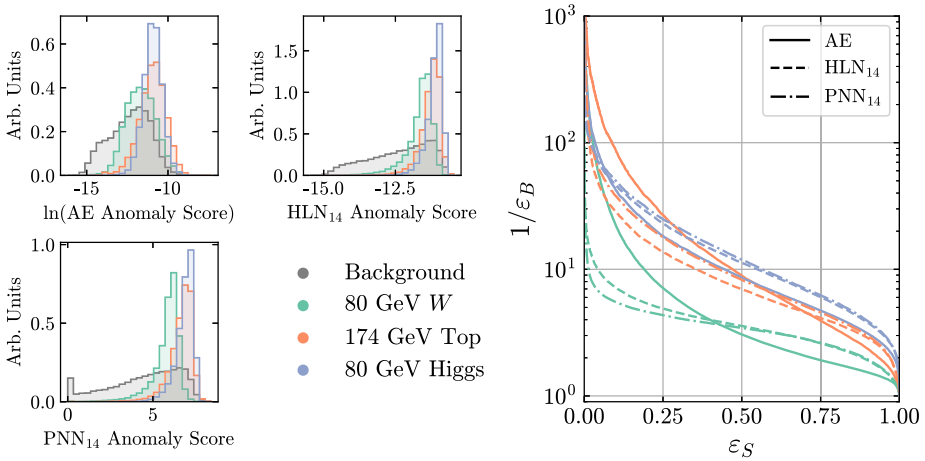


FIG. 7. The performance of the AE, PNN_{14}, and HLN_{14} as anomaly detectors on the 80 GeV W, 174 GeV top, and 80 GeV Higgs. Note that the Higgs bosons are pair produced from the decay of a heavier Higgs, leading to potentially four prongs in the large-radius jet. The left panels show the normalized distribution of each method's respective anomaly score for the background and each signal. The right panel shows the ROC curves for each signal, with the solid lines being the ROC curves for the AE, the dashed lines for HLN_{14}, and the dashed-dot lines for PNN_{14}.

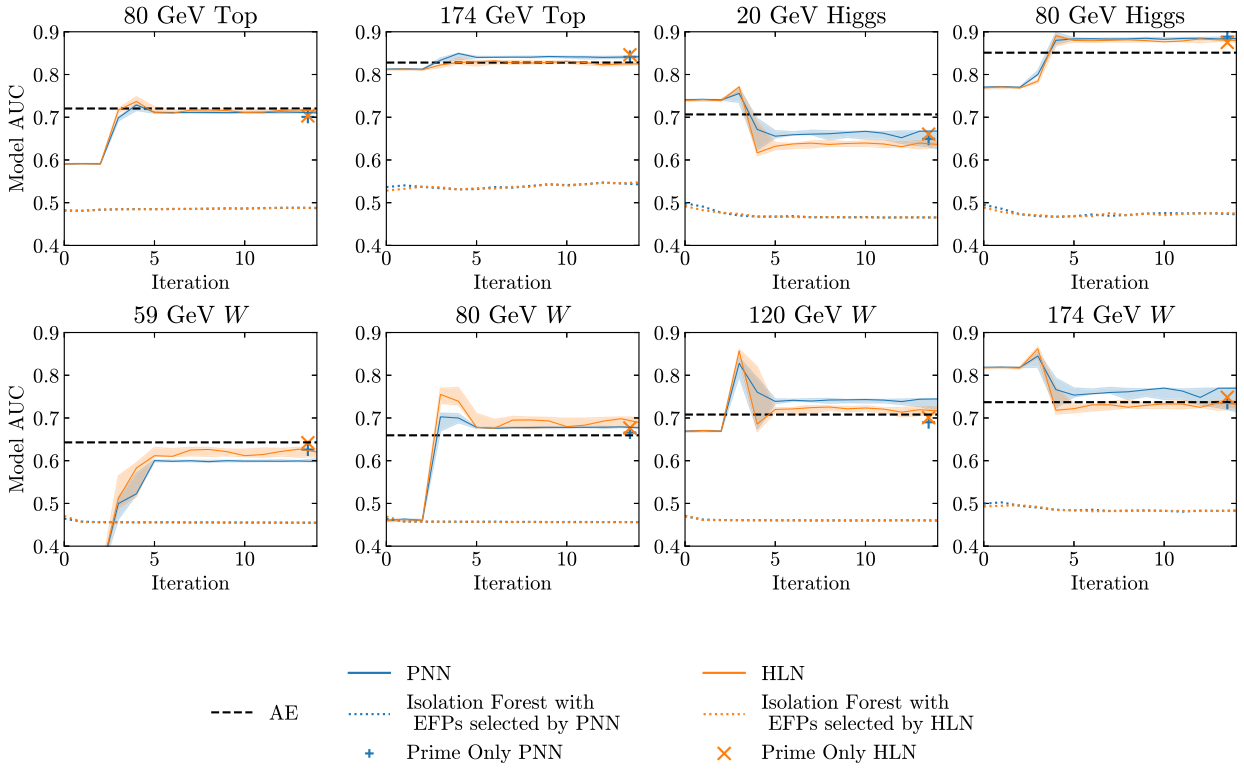


FIG. 8. AUCs for the PNN and HLN at each iteration for each of the eight signals reserved for testing. Note that the Higgs bosons are pair produced from the decay of a heavier Higgs, leading to potentially four prongs in the large-radius jet. The solid center lines are the AUC of the model used in the iterative process, the shaded bands show the maximum and minimum AUCs from retraining each network an additional 10 times. The dashed black line corresponds to the AE's AUC. The dotted lines correspond to the isolation forest anomaly detectors and the blue "+" (orange "×") is the PNN (HLN) trained using mass, p_T, and the five prime factors in Eq. (12).

and HLN. While such dips are not ideal, they are not completely unexpected. Our iterative process is trying to pick out the observables that help to best order the background events, with no attention paid to how effective they may or may not be to picking out signal events. So, for those three signals, it appears that the EFP added at the iteration where the AUC dips improves the ADO relative to the AE, but at the same time makes it more difficult for the HLNs and PNNs to distinguish those signal events from the background.

However, AUC is an inclusive figure of merit and, consequently, does not tell the whole story. As Fig. 7 highlights, networks with similar AUCs are not necessarily making the exact same decisions when used as anomaly detectors. Some more physically interpretable metrics are the background rejection ($1/\varepsilon_B$) at fixed signal efficiency (ε_S) and the signal efficiency at fixed background rejection. Table II shows the background rejection at two different fixed signal efficiencies—0.5 and 0.1—and the signal efficiency at two different fixed values of the background rejection—10 and 100—for all eight signals and five different networks—the AE, HLN_0, PNN_0, HLN_{14}, and PNN_{14}.

There are a few key takeaways from this table. Looking at the signal efficiency at a fixed value of the background rejection, we can see that, in general, our mimicker networks need to operate at lower signal efficiencies to achieve the same background rejection as the AE. The exceptions here are the final iteration of the mimicker networks when used as anomaly detectors for the 174 GeV Top and 80 GeV Higgs. These networks, when applied to these signals operate at comparable signal efficiencies to the AE for lower fixed values of the background rejection. Shifting now to the background rejection at fixed signal efficiency, we see that our mimicker networks compare favorably to the AE at higher signal efficiencies across all of the anomalous signals we consider, but fall behind the AE at lower signal efficiencies. Again, the exception here are the mimicker networks applied to the 80 GeV Higgs. As was observed earlier in Fig. 7, as we make tighter cuts on our mimicker networks, forcing them to operate at lower signal efficiencies, they begin to deem the background as being more anomalous than the signal when compared to the autoencoder. While this type of behavior would be difficult to deal with in a real analysis, it is not unique to our mimicker networks and is a challenge with anomaly detection in general. The cuts that result in $\varepsilon_B > \varepsilon_S$ are highlighted in bold in Table II. Taken together, these indicate that most of the performance of our mimicker networks is coming at higher signal efficiencies, and the long tails in their anomaly scores for the background distribution holds them back from exactly matching the AE.

TABLE II. The background rejection ($1/\varepsilon_B$) at two different fixed signal efficiencies (ε_S)—0.5 and 0.1—and the signal efficiency at two different fixed values of the background rejection—10 and 100—for all eight anomalous signals. We present these metrics for five different networks, the AE, PNN_0, HLN_0, PNN_{14}, and HLN_{14}. The values shown in bold are those where $\varepsilon_B > \varepsilon_S$.

80 GeV top	AE	HLN_{14}	PNN_{14}	HLN_0	PNN_0
$\varepsilon_S(1/\varepsilon_B = 10)$	0.252	**0.012**	0.114	**0.071**	**0.071**
$\varepsilon_S(1/\varepsilon_B = 100)$	0.022	**0.007**	**0.008**	**0.007**	**0.008**
$1/\varepsilon_B(\varepsilon_S = 0.5)$	4.24	4.03	3.95	2.29	2.29
$1/\varepsilon_B(\varepsilon_S = 0.1)$	26.5	12.0	11.3	**7.33**	**7.39**
174 GeV top	AE	HLN_{14}	PNN_{14}	HLN_0	PNN_0
$\varepsilon_S(1/\varepsilon_B = 10)$	0.470	0.357	0.428	0.146	0.148
$\varepsilon_S(1/\varepsilon_B = 100)$	0.088	0.016	0.022	0.013	0.013
$1/\varepsilon_B(\varepsilon_S = 0.5)$	8.93	6.94	8.26	5.96	6.00
$1/\varepsilon_B(\varepsilon_S = 0.1)$	87.6	28.6	38.0	12.8	12.9
20 GeV Higgs	AE	HLN_{14}	PNN_{14}	HLN_0	PNN_0
$\varepsilon_S(1/\varepsilon_B = 10)$	0.240	**0.027**	**0.086**	**0.032**	**0.033**
$\varepsilon_S(1/\varepsilon_B = 100)$	0.025	**0.001**	**0.001**	**0.001**	**0.001**
$1/\varepsilon_B(\varepsilon_S = 0.5)$	4.06	3.39	4.14	4.87	4.91
$1/\varepsilon_B(\varepsilon_S = 0.1)$	25.7	**6.82**	**9.67**	**6.68**	**6.72**
80 GeV Higgs	AE	HLN_{14}	PNN_{14}	HLN_0	PNN_0
$\varepsilon_S(1/\varepsilon_B = 10)$	0.446	0.549	0.565	**0.030**	**0.031**
$\varepsilon_S(1/\varepsilon_B = 100)$	0.036	0.022	0.020	**0.002**	**0.002**
$1/\varepsilon_B(\varepsilon_S = 0.5)$	8.58	11.3	11.9	4.67	4.70
$1/\varepsilon_B(\varepsilon_S = 0.1)$	42.4	46.1	50.1	**6.41**	**6.44**
59 GeV W	AE	HLN_{14}	PNN_{14}	HLN_0	PNN_0
$\varepsilon_S(1/\varepsilon_B = 10)$	0.155	**0.017**	**0.007**	**0.011**	**0.012**
$\varepsilon_S(1/\varepsilon_B = 100)$	0.015	**0.0003**	**0.0003**	**0.0007**	**0.0007**
$1/\varepsilon_B(\varepsilon_S = 0.5)$	2.86	2.76	2.62	**1.40**	**1.40**
$1/\varepsilon_B(\varepsilon_S = 0.1)$	16.1	**5.08**	**3.91**	**2.36**	**2.35**
80 GeV W	AE	HLN_{14}	PNN_{14}	HLN_0	PNN_0
$\varepsilon_S(1/\varepsilon_B = 10)$	0.190	**0.043**	**0.013**	**0.014**	**0.014**
$\varepsilon_S(1/\varepsilon_B = 100)$	0.028	**0.0005**	**0.0004**	**0.0009**	**0.0009**
$1/\varepsilon_B(\varepsilon_S = 0.5)$	3.06	3.57	3.44	**1.77**	**1.77**
$1/\varepsilon_B(\varepsilon_S = 0.1)$	22.4	**7.17**	**5.52**	**2.83**	**2.84**
120 GeV W	AE	HLN_{14}	PNN_{14}	HLN_0	PNN_0
$\varepsilon_S(1/\varepsilon_B = 10)$	0.244	**0.070**	**0.089**	**0.021**	**0.022**
$\varepsilon_S(1/\varepsilon_B = 100)$	0.040	**0.001**	**0.001**	**0.001**	**0.001**
$1/\varepsilon_B(\varepsilon_S = 0.5)$	3.71	4.01	4.76	2.97	2.97
$1/\varepsilon_B(\varepsilon_S = 0.1)$	32.9	**8.52**	**9.58**	**4.30**	**4.31**
174 GeV W	AE	HLN_{14}	PNN_{14}	HLN_0	PNN_0
$\varepsilon_S(1/\varepsilon_B = 10)$	0.289	0.124	0.190	**0.064**	**0.064**
$\varepsilon_S(1/\varepsilon_B = 100)$	0.052	**0.003**	**0.003**	**0.003**	**0.003**
$1/\varepsilon_B(\varepsilon_S = 0.5)$	4.40	4.21	5.53	6.05	6.10
$1/\varepsilon_B(\varepsilon_S = 0.1)$	42.4	11.4	14.7	**8.61**	**8.57**

Finally, by the end point of the iterative process, we had found that the PNN and HLN agreed on ordering of background events at about 83% when compared to the AE. Here, we see that in terms of the AUC metric, 83% mimicking transferred quite well to the use of these mimickers as simpler anomaly detectors with comparable performance. We expect the tendency for the mimicker networks to tag the background as being more anomalous than the signal at low signal efficiencies to subside as the ADO of the mimickers approaches 1.

C. Using only prime EFPs

In examining the EFPs selected to improve the decision ordering, we note that even though we use up to 14 EFPs, they only depend on six prime EFP factors:

(12)

Notably in these primes, the first and fifth prime factors are the energy correlation functions for two and three prong structures [77]. It is also interesting to note that these prime factors are nonzero only for $\geq 2, 3$ prong structures. As the AE is learning to encode the predominantly one-prong QCD events, it seems that it is losing information contained in these higher prong observables. With this loss of information, networks with direct access to these observables are able to explain the reconstruction error of the network.

The observation that the anomaly scores can be explained by composite operators which only have a few prime operators leads one to wonder if the prime EFPs are good enough. To test this, we trained both the PNN and the HLN using mass, p_T, and the six prime EFPs. The results are denoted in Figs. 6 and 8 by the blue "+" and orange "×," respectively. Not only do these "prime-only" networks perform comparably to each other, which matches the behavior we saw from the networks trained on the composite EFPs, but the prime-only and composite networks also perform comparably across all of the signals. The results in Fig. 6 show the ADO of the prime-only networks computed on the same pairs of events as the center line for the composite models. The ADO of the prime-only models has a similar spread as the composite models, and thus the two do indeed perform comparably. Taken together, this seems to indicate that the prime EFPs

alone contain all of the necessary information to construct simple anomaly detectors capable of matching much more complex ones. While each of the prime EFPs on their own would have been selected eventually, these results also suggests a more efficient iterative procedure for creating HLN and PNN mimickers, where one uses the redundancy in the full space of EFPs to their advantage and allows the algorithm to explore the full space of composite EFPs, but only selects those containing new prime factors.

D. Comparison with isolation forests

Through this iterative process, we have constructed two different types of dense neural networks that approximately match the AE not only in how their decision functions order background events, but also as anomaly detectors for classifying a variety of signals. It is clear then that the observables picked out by this procedure contain the information needed to match the AE on both fronts. One then wonders if an even simpler anomaly detector than the ones presented in Sec. III would give similar results. To investigate this possibility, we consider isolation forests as implemented by ISOLATION FOREST in SCIKIT-LEARN [86].

Isolation forests work by randomly selecting a feature from a given set of inputs, and then randomly selecting a split value for that feature. This splitting process is repeated until each event the model is trained on has been isolated from the rest, resulting in a treelike structure. We then build an ensemble, or "forest" of these classifiers. The anomaly score is the number of splittings needed to isolate each event, averaged over the entire ensemble. This kind of random partitioning tends to take fewer splittings to isolate anomalous events, so if the average number of splittings across a large ensemble is low, the event is likely to be anomalous. We wanted to see if the performance of the isolation forests saturate in the same way the HLNs and PNNs did, so we trained a series of them and added the new observable picked out by either the HLN or PNN each time. The details of our specific implementation is given in Appendix B. Since the HLNs and PNNs selected EFPs in a slightly different order, we trained two different sets of isolation forests. One set added observables in the order selected by the HLN, while the other added them in the order selected by the PNN.

Figure 8 shows how the isolation forests compare to the HLNs, CNNs, and AE when used as a classifier on the 8 signals considered in this work. The blue dotted line shows the AUC of the isolation forests trained on the EFPs selected by the PNN, the orange dotted line corresponds to isolation forests trained on the EFPs selected by the HLN. For most of the signals, both isolation forests have an AUC of $\sim$0.5, and are unable to match the performance of the HLN, PNN, or AE. This is a very interesting observation. The same small set of observables are able to lead to good anomaly detection when trying to match the decisions of the AE. However, as discussed above, these observables in some sense tell us what the AE is choosing to ignore when learning to reconstruct QCD images. Since these observables are not very descriptive for QCD events, the isolation forest does not have much to learn from. We expect the results would hold for other anomaly detection techniques trained on the same observables. Thus, we suspect it is the mimicking aspect of our procedure which allows for good anomaly detection with the simple set of observables.

V. CONCLUSION

In this paper, we have extended the results of Ref. [65] to build simpler, more interpretable anomaly detectors. Starting with a convolutional autoencoder, we iteratively built a network that mimics the autoencoder's ordering of background events, where the network's inputs are high-level variables taken from a set of energy flow polynomials. We presented two network architectures for the mimickers, the high-level network and the paired neural network. The high-level network aims to reproduce the reconstruction error of the autoencoder, while the paired neural network takes in two events and is trained to order them like the autoencoder. Note that both the PNN and HLN are trained to order anomalous events from the physics observables, which is an inherently different task than the autoencoder, which was only trained to compress and decompress background data. This highlights the difference with Ref. [65], in which the black-box network and mimicking network have the same task of binary classification. Given this fundamental difference between our AE and mimicking networks, it is not obvious that employing the same strategy will work when trying to mimic the autoencoder's ordering. However, we find that these two complementary approaches give similar performance, $\sim$83% agreement, when ordering background events and also pick out the same list of EFPs, suggesting the commonality of the information that is needed to order events like the autoencoder.

After mimicking the autoencoder on ordering of background events, we take these networks and apply them as anomaly detectors on eight different signals. Even though the mimickers and autoencoder have never seen these events, we find that the similarity in ordering transfers to these events, making the mimickers as good (or better) than the autoencoder as an anomaly detector for seven of the eight signals. It is worth emphasizing how such results were not guaranteed to occur. The autoencoder, having been trained only on background events, has no concept of what is anomalous. So it is not obvious that mimicking the ordering of events for the background will generalize to anomalous events, especially given a large set of signal classes.

Since the high-level observables picked out by these mimickers rely only on six prime energy flow polynomials, it indicates that the information required to order events like the autoencoder is reasonably small. However, since the

isolation forests based on these high-level inputs did not perform as well, it shows that mimicking the autoencoder's background ordering is crucial in creating a simpler anomaly detector.

In terms of future directions, it would be interesting to extend the list of energy flow polynomials to check that one can saturate the decision ordering of the autoencoder and to determine what prime energy flow polynomials are needed for that. Applying this technique to other anomaly detection methods on the same dataset would help uncover what high-level variables are being used by these methods and could help in designing more powerful anomaly detectors. Finally, it would be interesting to see if one can extend this technique to cases where there is no known high-level variable basis (like the energy flow polynomials) and to see to what extent decision ordering transfers to different signals. For instance, the methods which performed best on the Dark Machines anomaly score challenge [25,51,54] used variational autoencoder structures which only aimed to make a Gaussian latent space and did not try to reconstruct events. It would be very interesting to see what physics these methods are using, but there is no obvious basis of observables to use.

ACKNOWLEDGMENTS

The work of L. B. and S. C. was supported in part by the U.S. Department of Energy under Grant No. DE-SC0011640. B. O. is supported by the National Science Foundation under Cooperative Agreement No. PHY-2019786 (The NSF AI Institute for Artificial Intelligence and Fundamental Interactions, [87]). This work also benefited from access to the University of Oregon high performance computing cluster, Talapas.

APPENDIX A: SIMULATION DETAILS

In this appendix, we provide further details of the simulated public datasets we use in this work [38,69,70]. All of the QCD dijet, W, top, and Higgs samples are subject to the same selection criteria, showering, and detection simulation parameters. The background and anomalous events are generated using MadGraph [88] and PYTHIA 8 [89], with detector effects being simulated by DELPHES [90]. The jets are then clustered with FastJet [91,92] using the anti-k_T algorithm [93] with a cone size of $R = 1.0$. All events are required to have two hard jets, with the leading jet having $p_T > 450$ GeV and the subleading jet having $p_T > 200$ GeV. We then take only the leading jet in each event.

The QCD jets are created via $pp \to jj$. The W jets are created using $pp \to W' \to W(\to jj)Z(\to \nu\bar{\nu})$ with $m_{W'} = 1.2$ TeV. The top jets are produced via $pp \to Z' \to t\bar{t}$ with $m_{Z'} = 1.3$ TeV. Finally, the Higgs jets are produced with $pp \to HH$, $H \to hh$, $h \to jj$ with $m_H = 174$ GeV. For each of these signals, we only consider jets with $p_T \in [550, 650]$ GeV. This same p_T cut is applied to the background training and testing sets.

APPENDIX B: NETWORK TRAINING HYPERPARAMETERS

Here, we provide the details of the training hyperparameters of the AE, PNN, HLN, and isolation forests. For all three deep neural network architectures, we use the ReduceLROnPlateau and EarlyStopping callbacks from KERAS to dynamically reduce the learning rate and stop training early, respectively. All three neural networks are trained with the Adam optimizer [94].

For the AE, our training hyperparameters are

(i) Train for 100 epochs with EarlyStopping on the `validation_loss` with a patience of ten epochs.
(ii) Initial learning rate of 10^{-3} with ReduceLROnPlateau on the `validation_loss` with a patience of five epochs.
(iii) Batch size of 256.

For the HLN and PNN, our training hyperparameters are

(i) Train for 200 epochs with EarlyStopping on the `validation_loss` with a patience of ten epochs.
(ii) Initial learning rate of 10^{-3} with ReduceLROnPlateau on the `validation_loss` with a patience of five epochs.
(iii) Batch size of 256.

With the early stopping conditions, the AE trains in ∼30 epochs, the PNN trains in ∼50 epochs, and the HLN trains in ∼60 epochs.

For the isolation forests, our training hyperparameters are

(i) 250 estimators in the ensemble.
(ii) The `max_features` used to train each estimator is set to the number of inputs for each event.
(iii) `contamination` is set to "`auto`" since there is no way to determine what fraction of events can reliably be called outliers *a priori*.
(iv) `bootstrap` is set to "`False`," so individual trees are trained on random subsets of the data without replacement.

[1] B. Nachman, Anomaly detection for physics analysis and less than supervised learning, arXiv:2010.14554.

[2] G. Kasieczka *et al.*, The LHC Olympics 2020 a community challenge for anomaly detection in high energy physics, Rep. Prog. Phys. **84**, 124201 (2021).

[3] J. H. Collins, K. Howe, and B. Nachman, Anomaly Detection for Resonant New Physics with Machine Learning, Phys. Rev. Lett. **121**, 241803 (2018).

[4] R. T. D'Agnolo and A. Wulzer, Learning new physics from a machine, Phys. Rev. D **99**, 015014 (2019).

[5] A. De Simone and T. Jacques, Guiding new physics searches with unsupervised learning, Eur. Phys. J. C **79**, 289 (2019).

[6] A. Casa and G. Menardi, Nonparametric semisupervised classification for signal detection in high energy physics, arXiv:1809.02977.

[7] B. M. Dillon, D. A. Faroughy, and J. F. Kamenik, Uncovering latent jet substructure, Phys. Rev. D **100**, 056002 (2019).

[8] A. Mullin, S. Nicholls, H. Pacey, M. Parker, M. White, and S. Williams, Does SUSY have friends? A new approach for LHC event analysis, J. High Energy Phys. 02 (2021) 160.

[9] R. T. D'Agnolo, G. Grosso, M. Pierini, A. Wulzer, and M. Zanetti, Learning multivariate new physics, Eur. Phys. J. C **81**, 89 (2021).

[10] B. Nachman and D. Shih, Anomaly detection with density estimation, Phys. Rev. D **101**, 075042 (2020).

[11] A. Andreassen, B. Nachman, and D. Shih, Simulation assisted likelihood-free anomaly detection, Phys. Rev. D **101**, 095004 (2020).

[12] G. Aad *et al.* (ATLAS Collaboration), Dijet Resonance Search with Weak Supervision Using $\sqrt{s} = 13$ TeV pp Collisions in the ATLAS Detector, Phys. Rev. Lett. **125**, 131801 (2020).

[13] B. M. Dillon, D. A. Faroughy, J. F. Kamenik, and M. Szewc, Learning the latent structure of collider events, J. High Energy Phys. 10 (2020) 206.

[14] K. Benkendorfer, L. L. Pottier, and B. Nachman, Simulation-assisted decorrelation for resonant anomaly detection, Phys. Rev. D **104**, 035003 (2021).

[15] V. Mikuni and F. Canelli, Unsupervised clustering for collider physics, Phys. Rev. D **103**, 092007 (2021).

[16] G. Stein, U. Seljak, and B. Dai, Unsupervised in-distribution anomaly detection of new physics through conditional density estimation, in *Proceedings of the 34th Conference on Neural Information Processing Systems* (2020), arXiv:2012.11638.

[17] J. Batson, C. G. Haaf, Y. Kahn, and D. A. Roberts, Topological obstructions to autoencoding, J. High Energy Phys. 04 (2021) 280.

[18] A. Blance and M. Spannowsky, Unsupervised event classification with graphs on classical and photonic quantum computers, J. High Energy Phys. 08 (2021) 170.

[19] B. Bortolato, B. M. Dillon, J. F. Kamenik, and A. Smolkovič, Bump hunting in latent space, Phys. Rev. D **105**, 115009 (2022).

[20] J. H. Collins, P. Martín-Ramiro, B. Nachman, and D. Shih, Comparing weak- and unsupervised methods for resonant anomaly detection, Eur. Phys. J. C **81**, 617 (2021).

[21] T. Dorigo, M. Fumanelli, C. Maccani, M. Mojsovska, G. C. Strong, and B. Scarpa, RanBox: Anomaly detection in the copula space, arXiv:2106.05747.

[22] S. Volkovich, F. De Vito Halevy, and S. Bressler, A data-directed paradigm for BSM searches, Eur. Phys. J. C **82**, 265 (2022).

[23] A. Hallin, J. Isaacson, G. Kasieczka, C. Krause, B. Nachman, T. Quadfasel, M. Schlaffer, D. Shih, and M. Sommerhalder, Classifying Anomalies THrough Outer Density Estimation (CATHODE), arXiv:2109.00546.

[24] T. Buss, B. M. Dillon, T. Finke, M. Krämer, A. Morandini, A. Mück, I. Oleksiyuk, and T. Plehn, What's anomalous in LHC jets?, arXiv:2202.00686.

[25] T. Aarrestad *et al.*, The dark machines anomaly score challenge: Benchmark data and model independent event classification for the large hadron collider, SciPost Phys. **12**, 043 (2022).

[26] J. A. Aguilar-Saavedra, J. H. Collins, and R. K. Mishra, A generic anti-QCD jet tagger, J. High Energy Phys. 11 (2017) 163.

[27] J. Hajer, Y.-Y. Li, T. Liu, and H. Wang, Novelty detection meets collider physics, Phys. Rev. D **101**, 076015 (2020).

[28] T. Heimel, G. Kasieczka, T. Plehn, and J. M. Thompson, QCD or what?, SciPost Phys. **6**, 030 (2019).

[29] M. Farina, Y. Nakai, and D. Shih, Searching for new physics with deep autoencoders, Phys. Rev. D **101**, 075021 (2020).

[30] O. Cerri, T. Q. Nguyen, M. Pierini, M. Spiropulu, and J.-R. Vlimant, Variational autoencoders for new physics mining at the large hadron collider, J. High Energy Phys. 05 (2019) 036.

[31] T. S. Roy and A. H. Vijay, A robust anomaly finder based on autoencoders, arXiv:1903.02032.

[32] A. Blance, M. Spannowsky, and P. Waite, Adversarially-trained autoencoders for robust unsupervised new physics searches, J. High Energy Phys. 10 (2019) 047.

[33] M. Romão Crispim, N. F. Castro, R. Pedro, and T. Vale, Transferability of deep learning models in searches for new physics at colliders, Phys. Rev. D **101**, 035042 (2020).

[34] O. Amram and C. M. Suarez, Tag N' Train: A technique to train improved classifiers on unlabeled data, J. High Energy Phys. 01 (2021) 153.

[35] M. Crispim Romão, N. F. Castro, J. G. Milhano, R. Pedro, and T. Vale, Use of a generalized energy Mover's distance in the search for rare phenomena at colliders, Eur. Phys. J. C **81**, 192 (2021).

[36] O. Knapp, O. Cerri, G. Dissertori, T. Q. Nguyen, M. Pierini, and J.-R. Vlimant, Adversarially learned anomaly detection on CMS open data: Re-discovering the top quark, Eur. Phys. J. Plus **136**, 236 (2021).

[37] M. Crispim Romão, N. F. Castro, and R. Pedro, Finding new physics without learning about it: Anomaly detection as a tool for searches at colliders, Eur. Phys. J. C **81**, 27 (2021); **81**, 1020(E) (2021).

[38] T. Cheng, J.-F. Arguin, J. Leissner-Martin, J. Pilette, and T. Golling, Variational autoencoders for anomalous jet tagging, arXiv:2007.01850.

[39] C. K. Khosa and V. Sanz, Anomaly awareness, arXiv:2007.14462.

[40] P. Thaprasop, K. Zhou, J. Steinheimer, and C. Herold, Unsupervised outlier detection in heavy-ion collisions, Phys. Scr. **96**, 064003 (2021).
[41] J. A. Aguilar-Saavedra, F. R. Joaquim, and J. F. Seabra, Mass Unspecific Supervised Tagging (MUST) for boosted jets, J. High Energy Phys. 03 (2021) 012; Erratum, J. High Energy Phys. 04 (2021) 133.
[42] A. A. Pol, V. Berger, G. Cerminara, C. Germain, and M. Pierini, Anomaly detection with conditional variational autoencoders, in *Eighteenth International Conference on Machine Learning and Applications* (2020), arXiv:2010.05531.
[43] M. van Beekveld, S. Caron, L. Hendriks, P. Jackson, A. Leinweber, S. Otten, R. Patrick, R. Ruiz De Austri, M. Santoni, and M. White, Combining outlier analysis algorithms to identify new physics at the LHC, J. High Energy Phys. 09 (2021) 024.
[44] S. E. Park, D. Rankin, S.-M. Udrescu, M. Yunus, and P. Harris, Quasi anomalous knowledge: Searching for new physics with embedded knowledge, J. High Energy Phys. 06 (2021) 030.
[45] P. Chakravarti, M. Kuusela, J. Lei, and L. Wasserman, Model-independent detection of new physics signals using interpretable semi-supervised classifier tests, arXiv:2102.07679.
[46] D. A. Faroughy, Uncovering hidden new physics patterns in collider events using Bayesian probabilistic models, *Proc. Sci.*, ICHEP2020 (2021) 238 [arXiv:2012.08579].
[47] T. Finke, M. Krämer, A. Morandini, A. Mück, and I. Oleksiyuk, Autoencoders for unsupervised anomaly detection in high energy physics, J. High Energy Phys. 06 (2021) 161.
[48] O. Atkinson, A. Bhardwaj, C. Englert, V. S. Ngairangbam, and M. Spannowsky, Anomaly detection with convolutional Graph Neural Networks, J. High Energy Phys. 08 (2021) 080.
[49] B. M. Dillon, T. Plehn, C. Sauer, and P. Sorrenson, Better latent spaces for better autoencoders, SciPost Phys. **11**, 061 (2021).
[50] A. Kahn, J. Gonski, I. Ochoa, D. Williams, and G. Brooijmans, Anomalous jet identification via sequence modeling, J. Instrum. **16**, P08012 (2021).
[51] S. Caron, L. Hendriks, and R. Verheyen, Rare and different: Anomaly Scores from a combination of likelihood and out-of-distribution models to detect new physics at the LHC, SciPost Phys. **12**, 077 (2022).
[52] E. Govorkova *et al.*, Autoencoders on FPGAs for real-time, unsupervised new physics detection at 40 MHz at the Large Hadron Collider, Nat. Mach. Intell. **4**, 154 (2022).
[53] J. Gonski, J. Lai, B. Nachman, and I. Ochoa, High-dimensional anomaly detection with radiative return in e^+e^- collisions, J. High Energy Phys. 04 (2022) 156.
[54] B. Ostdiek, Deep set auto encoders for anomaly detection in particle physics, SciPost Phys. **12**, 045 (2022).
[55] K. Fraser, S. Homiller, R. K. Mishra, B. Ostdiek, and M. D. Schwartz, Challenges for unsupervised anomaly detection in particle physics, J. High Energy Phys. 03 (2022) 066.
[56] P. Baldi, P. Sadowski, and D. Whiteson, Searching for exotic particles in high-energy physics with deep learning, Nat. Commun. **5**, 4308 (2014).
[57] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy, Deep variational information bottleneck, arXiv:1612.00410.
[58] S. Chang, T. Cohen, and B. Ostdiek, What is the machine learning?, Phys. Rev. D **97**, 056009 (2018).
[59] S. Wunsch, R. Friese, R. Wolf, and G. Quast, Identifying the relevant dependencies of the neural network response on characteristics of the input space, Comput. Softw. Big Sci. **2**, 5 (2018).
[60] T. Roxlo and M. Reece, Opening the black box of neural nets: Case studies in stop/top discrimination, arXiv:1804.09278.
[61] J. Gou, B. Yu, S. J. Maybank, and D. Tao, Knowledge distillation: A survey, Int. J. Comput. Vis. **129**, 1789 (2021).
[62] G. Agarwal, L. Hay, I. Iashvili, B. Mannix, C. McLean, M. Morris, S. Rappoccio, and U. Schubert, Explainable AI for ML jet taggers using expert variables and layerwise relevance propagation, J. High Energy Phys. 05 (2021) 208.
[63] F. Mokhtar, R. Kansal, D. Diaz, J. Duarte, J. Pata, M. Pierini, and J.-R. Vlimant, Explaining machine-learned particle-flow reconstruction, in *Proceedings of the 35th Conference on Neural Information Processing Systems* (2021), arXiv:2111.12840.
[64] J. Craven, V. Jejjala, and A. Kar, Disentangling a deep learned volume formula, J. High Energy Phys. 06 (2021) 040.
[65] T. Faucett, J. Thaler, and D. Whiteson, Mapping machine-learned physics into a human-readable space, Phys. Rev. D **103**, 036020 (2021).
[66] J. Collado, J. N. Howard, T. Faucett, T. Tong, P. Baldi, and D. Whiteson, Learning to identify electrons, Phys. Rev. D **103**, 116028 (2021).
[67] J. Collado, K. Bauer, E. Witkowski, T. Faucett, D. Whiteson, and P. Baldi, Learning to isolate muons, J. High Energy Phys. 10 (2021) 200.
[68] P. T. Komiske, E. M. Metodiev, and J. Thaler, Energy flow polynomials: A complete linear basis for jet substructure, J. High Energy Phys. 04 (2018) 013.
[69] J. Leissner-Martin, T. Cheng, and J.-F. Arguin, QCD jet samples with particle flow constituents, 10.5281/zenodo.4641460 (2020).
[70] T. Cheng, Test sets for jet anomaly detection at the LHC, 10.5281/zenodo.4614656 (2021).
[71] V. Mikuni, B. Nachman, and D. Shih, Online-compatible unsupervised non-resonant anomaly detection, Phys. Rev. D **105**, 055006 (2022).
[72] S. Macaluso and D. Shih, Pulling out all the tops with computer vision and deep learning, J. High Energy Phys. 10 (2018) 121.
[73] https://energyflow.network.
[74] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, Fast and accurate deep network learning by exponential linear units (elus), arXiv:1511.07289.
[75] F. Chollet *et al.*, KERAS (2015), https://keras.io.
[76] M. Abadi *et al.*, TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. https://www.tensorflow.org/. Software available from tensorflow.org.
[77] A. J. Larkoski, G. P. Salam, and J. Thaler, Energy correlation functions for jet substructure, J. High Energy Phys. 06 (2013) 108.

[78] A. J. Larkoski, I. Moult, and D. Neill, Power counting to better jet observables, J. High Energy Phys. 12 (2014) 009.
[79] A. Banfi, G. P. Salam, and G. Zanderighi, Principles of general final-state resummation and automated implementation, J. High Energy Phys. 03 (2005) 073.
[80] G. Gur-Ari, M. Papucci, and G. Perez, Classification of energy flow observables in narrow jets, arXiv:1101.2905.
[81] M. Jankowiak and A. J. Larkoski, Jet substructure without trees, J. High Energy Phys. 06 (2011) 057.
[82] I. Moult, L. Necib, and J. Thaler, New angles on energy correlation functions, J. High Energy Phys. 12 (2016) 153.
[83] S. Chatrchyan *et al.* (CMS Collaboration), Search for a Higgs boson in the decay channel H to ZZ(*) to q qbar ℓ^- l+ in pp collisions at $\sqrt{s} = 7$ TeV, J. High Energy Phys. 04 (2012) 036.
[84] A. J. Larkoski, J. Thaler, and W. J. Waalewijn, Gaining (mutual) information about quark/gluon discrimination, J. High Energy Phys. 11 (2014) 129.
[85] 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.
[86] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, SCIKIT-LEARN: Machine learning in PYTHON, J. Mach. Learn. Res. **12**, 2825 (2011).
[87] http://iaifi.org/.
[88] J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, O. Mattelaer, H. S. Shao, T. Stelzer, P. Torrielli, and M. Zaro, The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations, J. High Energy Phys. 07 (2014) 079.
[89] 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, Comput. Phys. Commun. **191**, 159 (2015).
[90] J. de Favereau, C. Delaere, P. Demin, A. Giammanco, V. Lemaître, A. Mertens, and M. Selvaggi (DELPHES 3 Collaboration), DELPHES 3, A modular framework for fast simulation of a generic collider experiment, J. High Energy Phys. 02 (2014) 057.
[91] M. Cacciari, G. P. Salam, and G. Soyez, FastJet user manual, Eur. Phys. J. C **72**, 1896 (2012).
[92] M. Cacciari and G. P. Salam, Dispelling the N^3 myth for the k_t jet-finder, Phys. Lett. B **641**, 57 (2006).
[93] M. Cacciari, G. P. Salam, and G. Soyez, The anti-k_t jet clustering algorithm, J. High Energy Phys. 04 (2008) 063.
[94] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv:1412.6980.