Prepared for submission to JHEP

Explainable AI for ML jet taggers using expert variables and layerwise relevance propagation

Garvita Agarwal,[a] **Lauren Hay,**[a] **Ia Iashvili,**[a] **Benjamin Mannix,**[a,b] **Christine McLean,**[a] **Margaret Morris,**[a] **Salvatore Rappoccio,**[a] **Ulrich Schubert**[a,c]

[a] *Physics Department, and Institute for Computational and Data Sciences, University at Buffalo, State University of New York, Buffalo, New York, USA*

[b] *Department of Physics, University of Oregon, Eugene, OR, USA*

[c] *Google, Pittsburg, PA, USA*

E-mail: garvitaa@buffalo.edu, lmhay@buffalo.edu, iashvili@buffalo.edu, brmannix@buffalo.edu, ch.mclean@cern.ch, morris35@buffalo.edu, srrappoc@buffalo.edu, ulrichsc@buffalo.edu

Abstract: A framework is presented to extract and understand decision-making information from a deep neural network classifier of jet substructure tagging techniques. There are two methods studied. The first is using expert variables that augment the inputs ("expert-augmented" variables, or XAUGs). These XAUGs are concatenated to the classifier steps immediately before the final decision. The second is layerwise relevance propagation (LRP). The results show that XAUG variables can be used to interpret classifier behavior, increase discrimination ability when combined with low-level features, and in some cases capture the behavior of the classifier completely. The LRP technique can be used to find relevant information the network is using, and when combined with the XAUG variables, can be used to rank features, allowing one to find a reduced set of features that capture part of the network performance. These XAUGs can also be added to low-level networks as a guide to improve performance.

Contents

1 Introduction

Machine learning (ML) [1] has become an extremely prevalent tool in the classification of hadronically decaying highly Lorentz-boosted objects ("boosted jets") and to study the internal structure of hadronic jets ("jet substructure") [2–7]. In these areas, classification tasks are common, for which artificial neural networks (ANNs) are well suited. Recent work in "deep" neural networks (DNNs) has shown tremendous improvements in identification of boosted jets over selections based on expert variables (see Ref. [6] and references therein). These algorithms typically make use of classifiers based on convolutional neural networks (CNNs) and recurrent neural networks (RNNs).

However, these improvements come at a significant cost. The underlying understanding of particular decisions is lost (although it could be argued that this is not a drawback). This paper provides a method to elucidate classifier decisions in an explainable framework. This can assist in understanding of systematic uncertainties, as well as to develop better expert variables to capture the behavior of the classifier in a simpler way. We will also demonstrate that combining expert features with low-level information can improve network performance, similar (but not identical) to the approach in Ref. [8].

Collectively, explaining classifiers in artificial intelligence (AI) is referred to as "eXplainable AI" (XAI); see Refs. [9, 10] for reviews. There are many options to explain individual classifier decisions, mostly based on local approximations of the classifier function. Examples include LIME [11], SLISE [12], layerwise relevance propagation (LRP) [13–15], and explanation vectors [16]. We will utilize the LRP method in this paper due to simplicity of use and interpretation, although others could be used as well.

Particle physics has a unique property, in that many of the observables we are interested in have a full theoretical framework that can predict, or at least describe, their behavior. In particular, the expert variables provided can often be shown to fully capture available kinematic information, as is done in Refs. [17–20], or even have a theoretical description of the classifier itself [21]. However, there are other observables for which the theoretical completion is lower than others. In particular, information that can determine the original progenitor of the jet (such as flavor information) is sometimes less theoretically developed. Many new identifiers that show extremely high levels of performance rely on this type of information, such as track impact parameters, secondary vertex information, lepton content, and particle multiplicities, for instance as shown in Ref. [22]. Some of these observables are not easily calculable, and many are not even infrared/co-linear safe. As such, further tools are currently needed to extricate the various sources of discrimination aside from theoretical calculations.

Our approach is to augment low-level input information with high-level expert variables, referred to as "expert-augmented" (XAUG) variables, and utilize a local approximation system such as LRP to understand the importance of various pieces of information. The XAUG approach is similar to other contexts such as Ref. [8, 23, 24], however, our strategies and goals are different. Instead of using expert variables alongside low-level features in the inputs, we augment the entire low-level network with expert features by concatenating them to the post-processed decisions as a whole (either after the convolution or recurrent network decisions). This allows the a simple network to investigate and learn high-level details in the inputs. We also seek to provide these XAUG variables for dimensionality reduction in the function space of the DNN classifier, similar in strategy to the analytic tools proposed in [17], but with an eye toward extrication of the experimentally-available information that may not be easily captured by theory, such as flavor information. This can assist in explaining individual classifier decisions. In an ideal case, the DNN classifier can in fact be fully contained within the XAUG variables in a simple multilayer perceptron (MLP).

We develop a framework that can be applied to any classifier in an attempt to reduce the dimensionality of the problem by replacing or augmenting the lower-level/higher-

dimensional features by higher-level/lower-dimensional XAUG variables. We will show that the behavior of the DNN can be captured robustly by the addition of appropriate XAUG variables, and in some cases can entirely capture the behavior. This technique is not limited to information that can be theoretically described or predicted.

Firstly, we develop a trivial "toy" model with only a few features that are fully captured by XAUG variables. We show that the classifier decisions of such a DNN will be the same as a simpler classifier based only on XAUG variables themselves. We developed a 2D CNN based on "images" similar (but not identical) to the approaches in Refs. [24–26], as well as a 1D CNN and a 1D RNN based on particle lists inspired by the algorithm inputs in Ref. [8, 22].

Secondly, we develop several classifiers to distinguish boosted Z bosons decaying to closely separated $b\bar{b}$ pairs ($Z \to b\bar{b}$) from standard QCD jets based on simulations. Once again, we investigate several cases, including a 2D jet image-based CNN (using only jet kinematic and shape information), a 1D particle-list CNN, and a 1D particle-list RNN. The latter two contain information beyond the kinematics of the jet, such as particle content and decay impact parameters.

As mentioned above, it has been shown in [17–20] that the kinematic information of such classifiers is exhausted by angularity variables. One example "basis" is the set of N-subjettiness variables [27, 28]. As such, classifier decisions with only kinematic information should be almost entirely correlated with existing kinematic variables as XAUGS. Other information is not captured by these kinematic observables, however, such as the flavor and soft radiation in the jet such as the "jet pull" [29]. An advantage of the XAUG + LRP approach is that all of these types of information can be used to gain an understanding of the relevant features. This work is similar but complementary to Ref. [30], which extracts expert information from the network for kinematic and shape variables, whereas we also investigate flavor information.

It is known that in many cases, complete decorrelation with kinematic variables (such as transverse momentum and mass) is desirable for many analyses (see an overview in Ref. [6]). However, in the interest of simplicity, for this paper decorrelation is not addressed, although the general features of using XAUG variables and LRP extends to this case as will be demonstrated in future work.

In the following sections, we will introduce the layerwise relevance propagation technique in Sec. 2. We will then describe a toy model for demonstration in Sec. 3 to highlight how the technique works in a trivial but instructive case. Section 4 will describe the particle-level model. Explanations of both the toy model and the particle model will be discussed in Sec. 5. Finally, we will present conclusions in Sec. 6.

2 Layerwise Relevance Propagation

LRP is a linearized approximation to networks that can be thought of as a "Deep Taylor decomposition" [31]. To understand this method, take a neural network with a prediction $f(x)$, based on some inputs x; these inputs can be pixels in an image or input variables. LRP propagates the prediction backwards through the network, eventually assigning a relevance

score $R_j^{(1)}$ to each piece of input $x_j^{(1)}$. The relevance score indicates how much each input pixel or variable contributes to the final prediction.

In an ideal case, the LRP backwards propagation method has an overall relevance conservation:

$$f(x) = \cdots = \sum_{j \in l+1} R_j^{(l+1)} = \sum_{j \in l} R_j^{(l)} = \cdots = \sum_j R_j^{(1)} \tag{2.1}$$

here the superscript indicates the layer the relevances are being calculated at, and the subscript indicates the summation over the relevances within that layer. Relevance conservation means that at every layer of the network, the total relevance is conserved [15]. Therefore the backwards propagation process does not alter the prediction. Additionally LRP attributes the entirety of the network's decision to the inputs.

While there are many possible implementations of the LRP propagation rules [15], we focus on only a few of them in this paper. First, we consider **LRP-ϵ**:

$$\sum_j R_j^{(l)} = \sum_k \frac{x_j w_{jk}}{\sum_j x_j w_{jk} + \epsilon} R_k^{(l+1)} \tag{2.2}$$

Here, and in other LRP rules, x_j is the activation of the neurons at layer l, $R_k^{(l+1)}$ is the relevance scores assigned to the layer $l+1$ neurons, and w_{jk} is the weight connecting neurons j and k [14]. The ϵ term is included to prevent any division by zero. In LRP-ϵ, two criteria determine how relevance is propagated from layer $l+1$ to each layer l neuron. The first criterion is x_j, the neuron activation. Rather intuitively, more relevance goes to more activated neurons. Additionally, stronger connections (with larger w_{jk}) receive more relevance. The simple case where ϵ is set to zero is called **LRP-0**.

In particular, for LRP-ϵ, especially at large values, ϵ can absorb some of the relevance [13]. Therefore, we also consider **LRP-$\alpha\beta$**:

$$\sum_j R_j^{(l)} = \sum_k \left(\alpha \cdot \frac{(x_j w_{jk})^+}{\sum_j (x_j w_{jk})^+} - \beta \cdot \frac{(x_j w_{jk})^-}{\sum_j (x_j w_{jk})^-} \right) R_k^{(l+1)} \tag{2.3}$$

where the $+$ and $-$ superscripts indicate the positive and negative contributions to the relevance, respectively [14]. Therefore, choosing different values of α and β allows for control over the importance of positive and negative contributions to the network's decision [13]. Relevance conservation is enforced by requiring $\alpha - \beta = 1$.

In practice, ML models typically include a bias in each layer to improve accuracy. However, this violates the relevance conservation. In such cases, the summation of all relevance scores do not equal the total score, although this is not a significant disadvantage.

In this paper, we implement LRP using the *iNNvestigate* Python package [32]. To compute the LRP score for CNN models, we use the "Preset A" mode of *iNNvestigate*, which uses LRP-ϵ for dense layers, and LRP-$\alpha\beta$ for the convolution layers. To compute the LRP score for RNN models, we use LRP-ϵ throughout.

3 Toy model

We first create a toy model to explore the feature exhaustion of networks using XAUG variables. The toy model is designed to vaguely mimic the salient features of "jet images", to be processed by a 2D convolutional network, and a list of "particle level" information, to be processed by an RNN or 1D CNN. We assume there are two populations of inputs for each network, and design the networks to discriminate between the two populations.

For the toy images, the generated features are based on the substructure features of a jet with two "subjets" of radii $\boldsymbol{r_1}$ and $\boldsymbol{r_2}$, an angular distance of $\boldsymbol{\theta}$ apart, with momentum fraction of the leading "subjet" denoted as $\boldsymbol{z}$, as shown in Figure 1. These parameters are cartoon-level models for a jet with two subjets, where $\boldsymbol{\theta}$ corresponds to $\boldsymbol{\Delta}$ for a boosted jet with small angular separation as described in Ref. [33]. The "jet momentum" is normalized to one, and since we are limiting the model to events with 2 subjets, the subjet momenta are defined as $\boldsymbol{z}$ and $\boldsymbol{1-z}$. Using these parameters particles are randomly generated in each subjet. These are then pixelized into jet images. Each jet image is $\boldsymbol{16 \times 16}$ pixels, representing calorimeter cells in the detector, with 20 particles in each image or event.

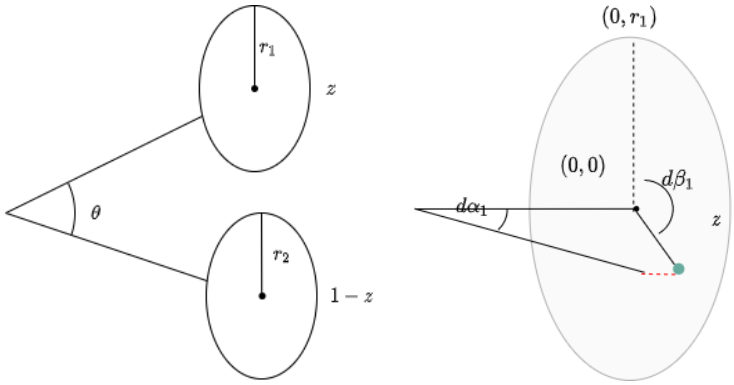


Figure 1: Diagram showing the toy "event" level parameters (left) and the toy "particle" level parameters (right).

There are two image populations, which we will refer to as **Signal** and **Background**. We generate 1M images, half Signal and half Background. The Signal is randomly sampled from normal distributions for $\boldsymbol{z}$ and $\boldsymbol{\theta}$, while the Background is sampled from exponential distributions that are mostly separated from signal in their combined phase space. The radii of the subjets for both cases are sampled from a uniform distribution within a range that keeps the particles within the 16x16 image. These radii enclose the sampling of the particles coordinates w.r.t the subjet axis, which are represented by variables $\boldsymbol{d\alpha}$ and $\mathbf{d}\boldsymbol{\beta}$. The angular distance from the subjet axis, $\boldsymbol{d\alpha}$ is sampled from the specified distributions (normal or exponential for Signal or Background, respectively), limited by the radius of the subjet. The azimuthal angle w.r.t the subjet axis, $\mathbf{d}\boldsymbol{\beta}$, is sampled from a uniform distribution of $\boldsymbol{\{0, 2\pi\}}$ for both populations. The momentum fraction of the subjets $\boldsymbol{z}$ is distributed among the 10 particles of each jet according to the distributions, partitioned to sum to $\boldsymbol{z}$. These are used as the $\boldsymbol{p_T}$ of the constituents, and are used as the pixel intensities

in the jet image. We obtain the coordinates of the pixels within the image by transforming $\boldsymbol{\alpha}$ and $\mathbf{d}\boldsymbol{\beta}$ to an abstracted $\boldsymbol{\eta}$-$\boldsymbol{\phi}$ plane where the leading-$\boldsymbol{p_T}$ subjet axis is at $\mathbf{(0, 0)}$ and the subleading-$\boldsymbol{p_T}$ subjet axis is at $\mathbf{(0, -1)}$. Finally, the image is flipped if the sum of pixel intensities on the left half of the image is greater than that on the right half. The input distributions are shown in Figure 2, and examples of summed input images are shown in Figure 3.

For use in networks that take 1-dimensional inputs, the data is structured as a list of the simulated "particle-level" information, i.e., the individual constituent $\boldsymbol{z}$ fractions, and $\boldsymbol{\alpha}$ and $\mathbf{d}\boldsymbol{\beta}$ values, as seen in the right diagram of Figure 1. Within each event, particles are sorted by $\boldsymbol{z_1}$, which is the leading jet's $\boldsymbol{z}$ value distributed among the leading jets particles. We refer to this as a "particle list", and just like the image data, we generate 1M events, half "signal" and half "background".

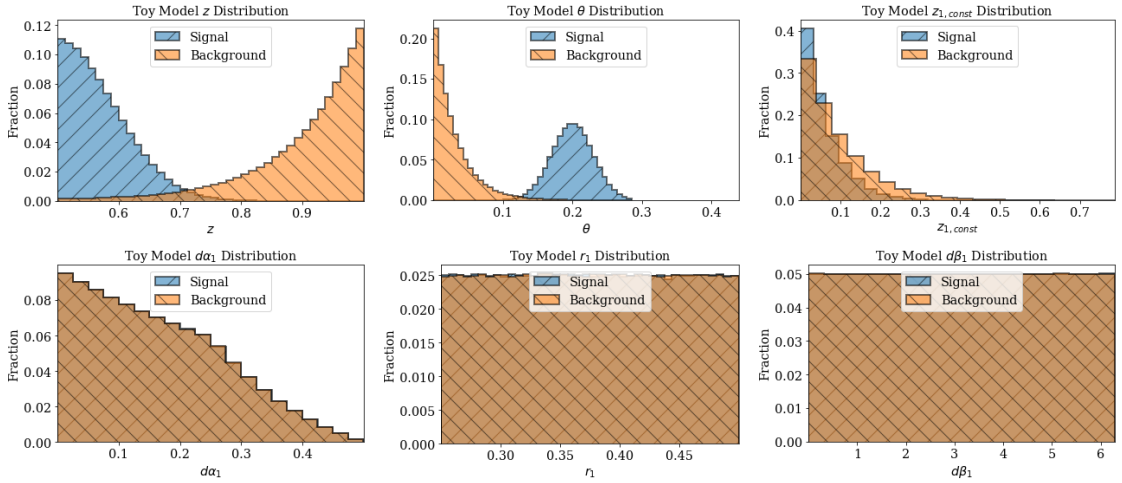


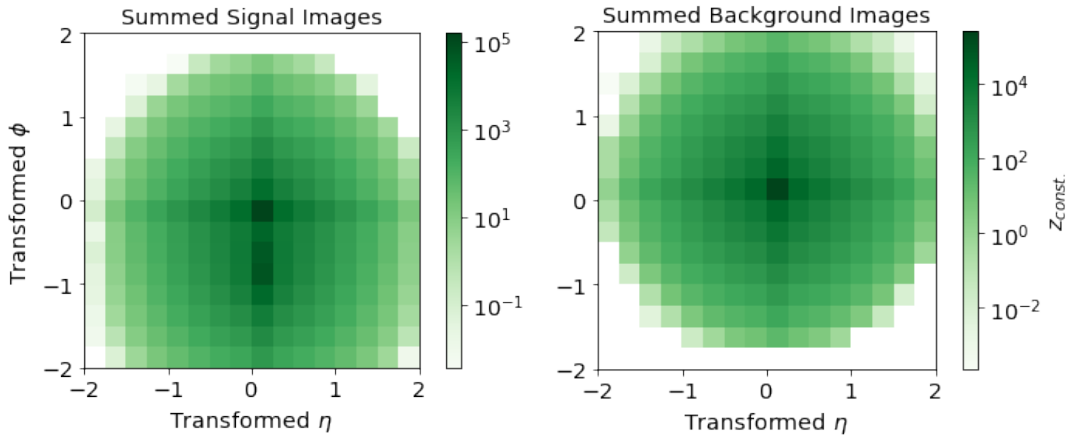
Figure 2: The normalized $\boldsymbol{z}$, $\boldsymbol{\theta}$ distributions for Signal and Background events in the Toy Model, along with the $\boldsymbol{z_1}$, $\mathrm{d}\boldsymbol{\theta_1}$, $\mathrm{d}\boldsymbol{\beta_1}$, and $\boldsymbol{r_1}$ values (w.r.t the 1st subjet axis) for those events.

3.1 Toy Model Networks

We investigate three network structures: a 2D CNN for jet images, a 1D CNN for particle lists, and a 1D RNN for particle lists. The datasets were shuffled and split into testing, training, and validation subsets. In all cases, when XAUGs are used in conjunction with low-level inputs, they are concatenated to the post-processed outputs of the CNN or RNN. This augmented list is then combined in a flattened layer for the final decision.

3.1.1 CNN for toy images

For the 2D CNN, we build a network consisting of 3 convolutional layers, followed by a dropout layer that randomly drops nodes, and a max-pooling layer that calculates the maximum value of the feature map, then 2 dense rectified linear unit (ReLU) layers, with a sigmoid dense layer making the final classification between the Signal and Background



(a) (b)

Figure 3: Toy model summed input images for Signal (a) and Background (b) jets.

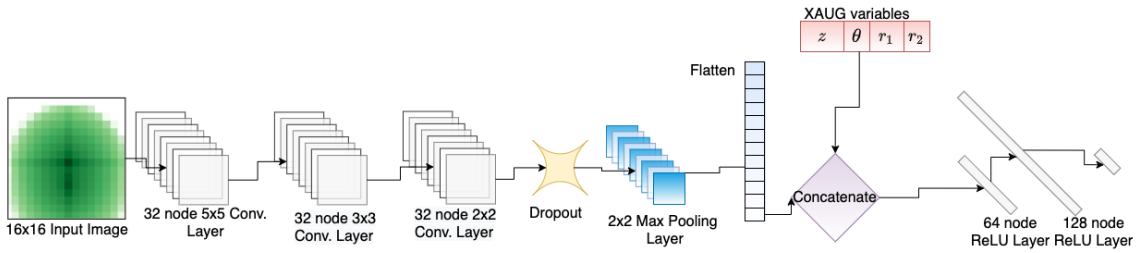


Figure 4: Diagram of 2D CNN for classification of toy model images. When used, XAUGs are concatenated to the processed outputs immediately before the final two ReLu layers.

populations. This network is based on networks made to analyze 2D jet images, like those in Refs. [25, 26]. The network that operates on the toy images has 154,178 trainable parameters. When we add in the "event" level XAUGs, we add a layer to flatten the 2D convolved output so that we can concatenate the 1D expert variables onto the end. Additionally an extra dense layer is included to find further relations in the concatenated datasets. The final network has 43,073 trainable parameters, the details of which can be seen in Fig. 4.

3.1.2 CNN for toy list

For the 1D CNN, we build a network consisting of a set of layers that goes over each set of information within the particle list separately ($\boldsymbol{p_{frac}}$, $\boldsymbol{\alpha}$, d$\boldsymbol{\beta}$) before the outputs of these layers are concatenated together and passed through a dense layer before a final decision is made. Each set of inputs goes through two 1-dimensional convolutional layers, a max pooling layer, and two additional convolutional layers before being flattened and concatenated together. In the case where XAUGs are added, they are concatenated with the particle level information after they've gone through their respective convolutional layers. This model has 270,082 trainable parameters, and the details can be seen in Fig. 5.

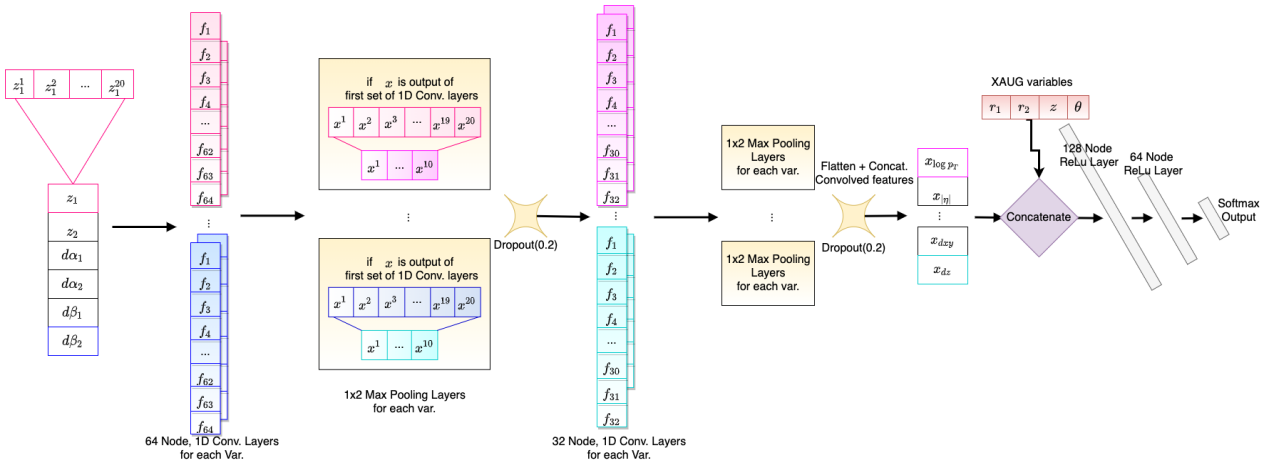


Figure 5: Diagram of 1D CNN for classification of toy model particle lists. When used, XAUGs are concatenated to the processed outputs immediately before the final two ReLu layers.

3.1.3 RNN for toy list

For the toy model RNN, we build a network consisting of recurrent layers that process the particle level information before being flattened and passed through a dense layer before the final decision is made. The inputs are divided into subsets based on the chosen timestep of the recurrent layers. In our network there are 10 timesteps per event; this equates to processing information from 2 particles at a time, with a memory of the previous 2 particles being used to find patterns within the event. The particle list information passes through 2 gated recurrent unit layers, then a batch normalization, one more gated recurrent unit, before being flattened and passed to the dense layers for decision-making. In the case where XAUGs are added, they are concatenated with the particle level information after they've gone through their respective recurrent layers. This model has 248,938 trainable parameters, and the details can be seen in Fig. 6.

4 Particle-level model

The more realistic simulation is a set of `PYTHIA` 8.2.35 [34] events.[1] To simulate 2-prong structure of boosted jets, we produce SM $\boldsymbol{ZZ}$ production with both $\boldsymbol{Z}$ bosons decaying to $\boldsymbol{b\bar{b}}$ [2]. Both $\boldsymbol{b}$ quarks from the $\boldsymbol{Z}$ decay are required to be within $\boldsymbol{\Delta R < 0.8}$ of the $\boldsymbol{Z}$. We use generic QCD multijet events as background[3]. We only consider the leading jet in the simulations, so we refer to these two samples as $\boldsymbol{Zbb}$ and $\boldsymbol{QCD}$, respectively.

It has been shown in Ref. [35] that independence of kinematic variables is a desirable feature for boosted object tagging, and has been applied in several "decorrelated" versions of DNNs, with examples in Refs. [22, 36, 37]. The approaches taken in other work has been

[1] Our software is outlined here
[2] The configuration for SM $\boldsymbol{ZZ}$ production is here
[3] The configuration for QCD multijet event simulation is here

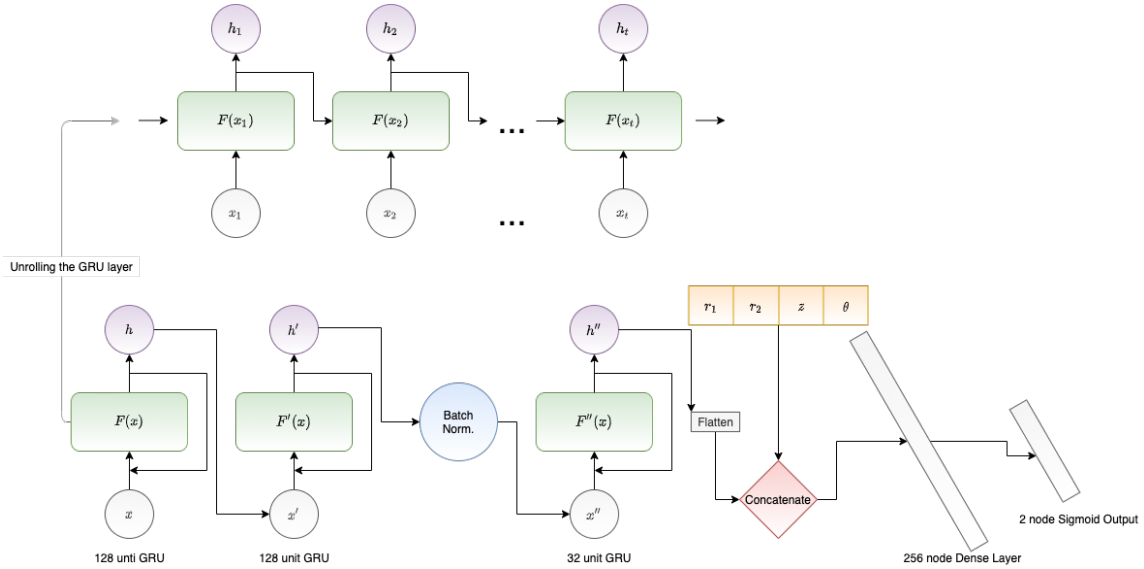


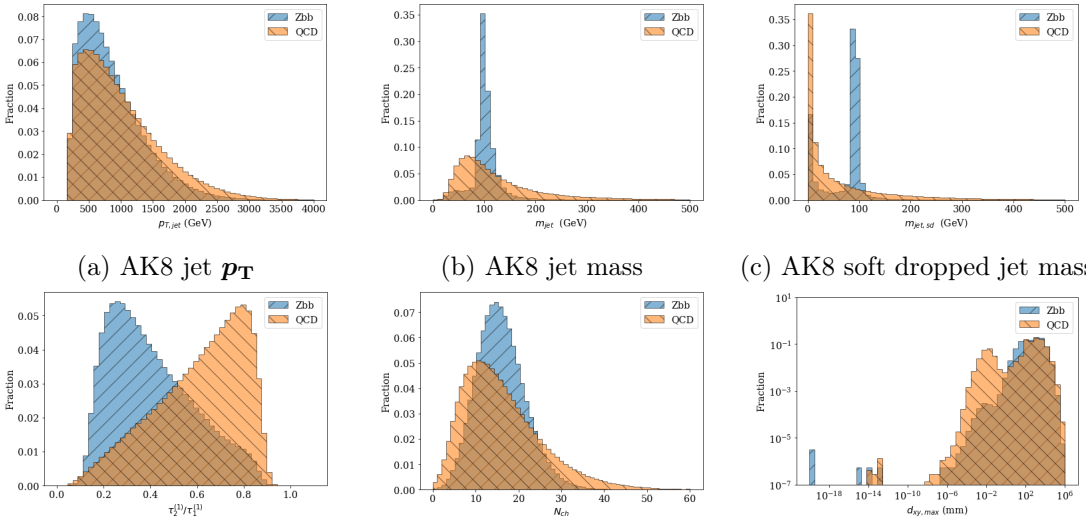
Figure 6: Diagram of RNN for classification of toy model particle lists. When used, XAUGs are concatenated to the processed outputs immediately before the final dense layer.

to either train an adversarial network to achieve the desired independence from kinematic variables, or to decorrelate the behavior in a brute force approach. Our simulated samples have different $\boldsymbol{p_T}$ spectra in reality, however we wish to eliminate such kinematic differences to some extent in our training. As such, we artificially weight the `PYTHIA` generation with the `bias2Selection` parameter in order to have similar jet $\boldsymbol{p_T}$ spectra, which is shown in Fig. 7a. We therefore partially remove the $\boldsymbol{p_T}$ and mass dependencies of the classifier. The goal of this paper is to explain decisions, so we take this simpler approach for ease of demonstration. It does not achieve complete decorrelation. The principles we develop are also applicable to more thoroughly decorrelated taggers, but become more complicated due to the increased dimensionality of having adversarial networks and are thus left to future work for investigation.

We then cluster the particles with `fastjet` 3.3.1 [38, 39] to create anti-$\boldsymbol{k_T}$ jets [40] with a distance parameter of $\boldsymbol{R = 0.8}$. We then use the `RecursiveTools` package from `fastjet-contrib` 1.036, which implements the "modified mass drop tagger" (mMDT) [33]. This is the same algorithm as the "soft drop" (SD) algorithm [41] with $\boldsymbol{\beta = 0}$. We use the mMDT with $\boldsymbol{z_{cut} = 0.1}$ and $\boldsymbol{\beta = 0}$ to find exactly two subjets. We define the "groomed" or "soft dropped" jet mass as $\boldsymbol{m_{jet,sd}}$.

We select the leading jet in the event with $\boldsymbol{p_T > 200}$ GeV, $\boldsymbol{|\eta| < 2.4}$, and $\boldsymbol{50 < m_{jet,sd} < 150}$ GeV. After these selections, there are 552k events in the signal and background categories (half in each). The distributions for the jet $\boldsymbol{p_T}$, mass, and soft-dropped mass are shown in Fig. 7.

The information used for each particle in our classifiers are the four-vector information ($\boldsymbol{p_T}$, $\boldsymbol{\eta}$, $\boldsymbol{\phi}$, mass), the production vertex spatial location ($\boldsymbol{d_x}$, $\boldsymbol{d_y}$, $\boldsymbol{d_z}$, with transverse distance denoted by $\boldsymbol{d_{xy}}$), and the particle PDG ID. In addition, the distance to the center



(a) AK8 jet p_T (b) AK8 jet mass (c) AK8 soft dropped jet mass

(d) τ_2/τ_1 (e) Number of charged hadrons (f) Max d_{xy} per event

Figure 7: Input distributions to training for Zbb and QCD events. Shown are the jet p_T (upper left), ungroomed jet mass (upper middle), groomed jet mass with the soft drop algorithm (upper right), n-subjettiness ratio τ_2/τ_1 (lower left), charged hadron multiplicity (lower middle), and $d_{xy,\,max}$ (lower right).

of the jet, and the distances to each subjet are also stored. The distribution for d_{xy} is shown in Fig. 7.

4.1 XAUGs for particle-level model

To encapsulate the kinematic information of jets and their substructure, we will select the N-subjettiness variables τ_m^β as one set of XAUGs. We also input the jet four-vector, and the groomed jet mass.[4] The angular distance between the two subjets $\Delta R_{subjets}$ and momentum fraction of the leading subjet z are also added.

To add simple flavor identification, we add jet composition variables such as charged hadron multiplicity (N_{ch}), neutral hadron multiplicity(N_{neut}), photon multiplicity (N_γ), muon multiplicity (N_μ), and electron multiplicity (N_e).

A measure of the soft radiation in the jet is represented by the jet pull angle (ϕ_{pull}) [29].

Finally, heavy flavor identification information is encapsulated by an (over-)simplified metric of the maximum of the spatial locations of the production vertex for each particle in the $x-y$ (d_{xy}) and z (d_z) directions.

Distributions for τ_2/τ_1 and N_{ch} are shown in Fig. 7.

4.2 Variable normalization for particle-level model

To ensure that all of the inputs to our networks have comparable numerical magnitude, we perform pre-processing on each input variable based on the overall distribution in the

[4]Most taggers have historically avoided adding the mass to the classifier inputs, however in our case we have decided to add it since it can be deduced from the N-subjettiness variables in any case.

signal and background. An equal number of signal and background events are chosen in each case, and then the distribution of each input variable for the ensemble are calculated. The mean and standard deviations are computed, and the distribution is truncated between $\pm\mathbf{3}$ standard deviations of the mean. We then define the minimum to be zero, and the maximum to be one. These are referred to as "normalized" variables. To clarify, the underflows and overflows appear at 0 and 1, respectively.

4.3 Classifiers for particle-level model

We consider three types of classifiers. The first is a 2D image-based CNN, with preprocessing inspired by Refs. [25, 26]. The second is a 1d CNN that inputs measurements from the first $\boldsymbol{N}$ particles in a transverse momentum ($\boldsymbol{p_T}$)-ordered list, similar to the `DeepAK8` algorithm in Ref. [22]. The third is a recurrent neural network (RNN) that inputs an arbitrarily large list of the same information as the 1d CNN.

We discuss each classifier in turn.

4.4 2D CNN for particle-level model

The first classifier we considered for particle-level simulations is a CNN that is based on 2D jet images, with preprocessing inspired by Refs. [25, 26]. We then use a CNN with the same architecture shown in Fig. 4, but with different XAUGs and the full list of particle inputs. At this point, if only looking at image data, the network will pass the flattened images to 2 dense layers before being output. If the network was also passed XAUG variables, these will be concatenated with the flattened images before going through the dense and output layers.

The pre-processing is as follows. First, we find exactly two subjets (events with fewer than two subjets are discarded). We then examine the leading jet in the event, and calculate the subjet directions. The subjet with the higher $\boldsymbol{p_T}$ is used as the center of the jet image $\boldsymbol{(0,0)}$, and the image is then rotated in local pseudo-rapidity/azimuth space ($\boldsymbol{\eta-\phi}$) such that the lower-$\boldsymbol{p_T}$ subjet is pointing downward. The radial distances are scaled in units of the $\boldsymbol{\Delta R}$ between the two subjets, such that the lower-$\boldsymbol{p_T}$ subjet is placed at $\boldsymbol{(0,-1)}$. The $\boldsymbol{p_T}$ of all of the constituents are scaled by the total jet $\boldsymbol{p_T}$, and are then pixelated into a grid of size **16x16**. The image is then parity flipped so that the largest sum of the pixel intensities is on the right-hand side of the image. Plots of the aggregated Signal ($\boldsymbol{Zbb}$) and background (QCD) are shown in Fig. 8.

4.5 1D CNN for particle-level model

The second classifier we considered for particle-level simulations is a CNN that is based on observables from the first $\boldsymbol{N}$ particles in a $\boldsymbol{p_T}$-ordered list of inputs, similar to the `DeepAK8` algorithm in Ref. [22]. One of the main advantages of `DeepAK8` is the use of particle-level information to have sensitivity to particle content such as hadron flavor, and quark-gluon discrimination. For this reason, we have two separate particle list taggers, one with only the four-momentum information of the constituents, and another that adds other observables. We are interested in general characteristics of a procedure in this paper, so we do not

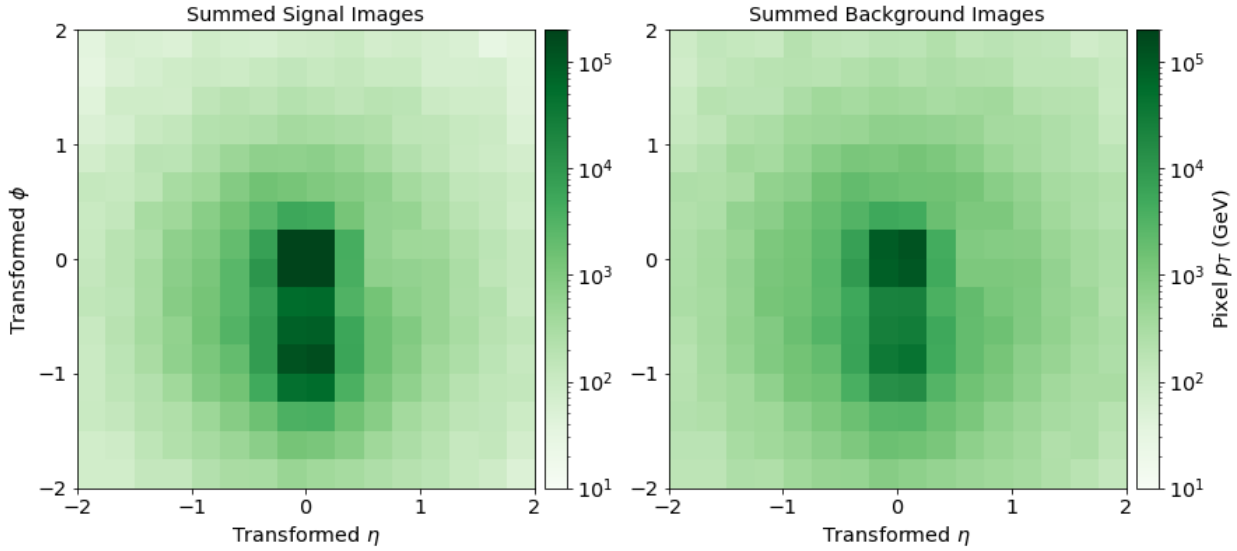


Figure 8: Aggregated Signal (***Zbb***) and Background (QCD) Jet Images after pre-processing.

attempt a realistic flavor definition for our simulated sample. Instead, we directly input the particle ID of the stable particle, and the $\boldsymbol{x, y, z}$ position of its production vertex. These are overly simplistic assumptions for a realistic tagger, however it will demonstrate the procedure of how flavor information could be investigated with our method.

In particular, we have found that $\boldsymbol{N = 20}$ provides good discrimination between 2-prong jets and QCD jets. We use a network architecture of 2 convolutional layers, one with shape (64, 3) and one with shape (64, 1) followed by a Max Pooling with pool size 2, followed by a dropout of **20%** of the network's nodes. The 2 convolutional layers and Max Pooling layer are repeated with shapes (32, 3), (32, 1) and 2 respectively with another **20%** dropout following. The convolved values are then flattened and concatenated with the XAUG input features, which are then fed through a dense layer with a RELU activation. The network's structure is the same as in Fig. 5 with different XAUGs.

For inputs to the 1D CNN that are similar to those in DeepAK8, 17 variables were created. The variables are listed in Table 1.

4.6 1D RNN for particle-level model

The final classifier is an RNN that uses similar inputs as the CNN from the particle list; however, we limit the number of particle and expert variables due to the increased training

Variable
$\boldsymbol{log(p_T)}$
$\boldsymbol{log(p_T/p_{T_{jet}})}$
$\boldsymbol{log(E)}$
$\boldsymbol{\lvert\eta\rvert}$
$\boldsymbol{\Delta\phi(jet)}$
$\boldsymbol{\Delta\eta(jet)}$
$\boldsymbol{\Delta R(jet)}$
$\boldsymbol{\Delta R(subjet1)}$
$\boldsymbol{\Delta R(subjet2)}$
Charge $\boldsymbol{q}$
isMuon
isElectron
isPhoton
isChargedHadron
isNeutralHadron
$\boldsymbol{d_{xy}}$
$\boldsymbol{d_z}$

Table 1: Particle list input variables of 1D CNN, properties of the constituents of the leading jet.

and processing time of the network. The network architecture we apply is a first input layer, followed by 3 recurrent layers, 2 additional input layers, and 2 dense layers at the end, shown in Fig. 6.

For the first input, the network takes in the four-vector ($\boldsymbol{p_T}$, $\boldsymbol{\eta}$, and $\boldsymbol{\phi}$) data of the first 20 constituents of the leading jet of each event sorted by $\boldsymbol{p_T}$. This is given to the recurrent layers. Then, the XAUG variables $\boldsymbol{\tau_{21}}$ and charged-hadron multiplicity are added before the final dense layers.

5 Explanations

We now explain the performance of the models described in the previous sections. In the following section, we will discuss explanations of both the toy and particle models.

5.1 Toy Model Explanations

The classification of the toy Signal and Background is trivial, even without the XAUG variables. The Area Under Curve (AUC) of the ROC curve is close to unity. However, it is still instructive to investigate the explanations of these decisions to demonstrate the features of LRP and XAUGs. The understanding gained can also be applied to the more realistic particle simulation.

Examples of the classification in the 2D CNN are shown in Fig. 9. There are four examples of toy Signal and Background events in green. Their LRP scores are also shown as heatmaps in red and blue. The blue LRP scores indicate pixels that contributed positively to the identification, whereas the red LRP scores indicate pixels that contributed negatively to the identification. It is clear that the Signal classifications prefer information where there is a disjoint subjet near **(0,-1)**, whereas Background classifications discourage that area. Indeed, the input Signal images in Fig. 3 have a well-separated second subjet near **(0,-1)**, while the Background images do not. As such, the network is clearly learning the appropriate information that was used to create the Toy Model, including the position (i.e. angle) and intensity (i.e. energy) of the "subjets".

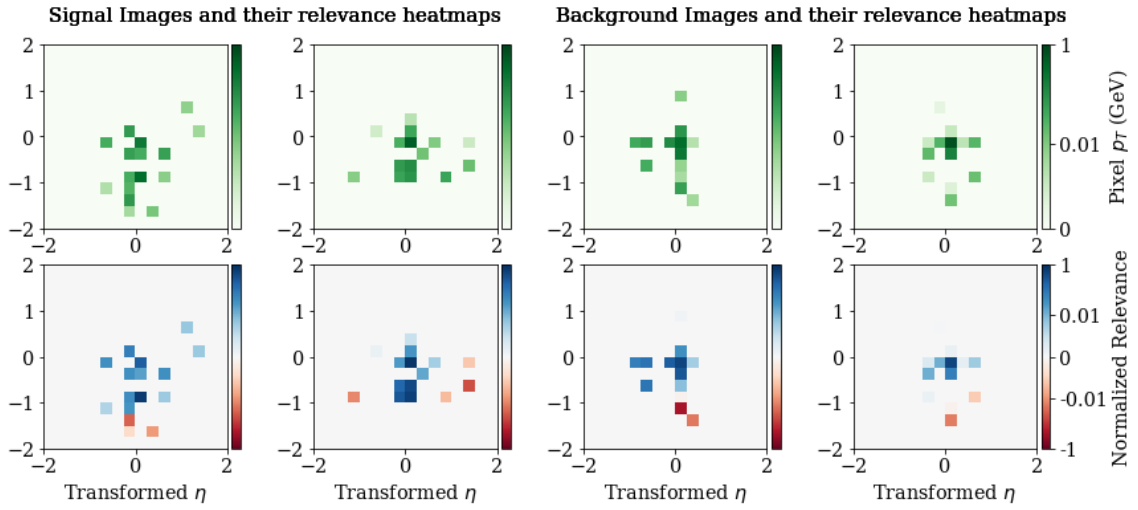


Figure 9: Individual Toy Model Signal (left two columns) and Background (right two columns) images that were correctly predicted (top) and their corresponding LRP score heatmaps (bottom).

We then trained a network with the XAUG variables $\boldsymbol{r_1}$, $\boldsymbol{r_2}$, $\boldsymbol{z}$, and $\boldsymbol{\theta}$ included among the network input features. (These variables are defined and plotted in Figs. 1 and 2, respectively.) This is an exhaustive list of the entire information content of the toy model. As such, we expect that the network will be able to (almost) entirely ignore the information

contained in the jet images, because its decision is completely determined by the XAUGs. Any residual relevance is an artifact of the optimization, not a distinguishing feature.

To investigate the optimization dependence, we trained the same 2D CNN with images and XAUGs four times, labeled Models 1-4. This allows us to have four separate optimizations and judge their consistency. The mean normalized LRP scores of the aggregated events are shown in Fig. 10. To produce these bar plots, for each event we first found the feature (XAUG or image pixel) with the maximum absolute LRP score. Then we normalized the event by dividing all LRP scores by this maximum value, in order to eliminate the dependence of the relevance on the events with large deviations in total relevance. Then, for each event, we summed the absolute values of the normalized pixel LRP scores to get an image LRP score. After this, we averaged the absolute values of the XAUG and image LRP scores across all events to produced the mean normalized relevances shown. This procedure is used for all bar plots.

It is clear from these bar plots that the two most important pieces of information are the XAUG variables $\boldsymbol{z}$ and $\boldsymbol{\theta}$. There is very little information that the network learns from the "subjet" radii (which is expected, since they are drawn from the same uniform distributions). Furthermore, the network does not extract many features from the image itself after the addition of the XAUG variables. The image relevance varies between 0.0 to 0.1, but is always much lower than that of $\boldsymbol{z}$ and $\boldsymbol{\theta}$. This indicates an extremely flat optimization in the network training in the space of the input image. The network is unable to discern the subleading feature with much accuracy.

To study the impact of various features on the decision making, we plot the relative rank of each feature in Fig. 11. The annotation on each bubble is the percent of events for which the variable had a certain rank. The $\boldsymbol{z}$ variable is ordinarily the highest ranked in all of the four models we used, while the image is ordinarily last. The variables $\boldsymbol{r_1}$ and $\boldsymbol{r_2}$ do not carry any discriminatory powers, as can be seen from Fig. 10.

It is interesting to note that there is some variation within the optimizations, especially in the subleading domain of the optimization space. Differences in relevance attributed to features between trainings point toward differences in local optimization of the network. This leads one to conclude that there should be numerical uncertainties applied to the derivation of DNN-based identification algorithms.

We plot the predicted score for these 4 models for $\boldsymbol{\theta}$ versus $\boldsymbol{z}$ in Fig. 12. The orange distribution is the "background" while the blue distribution is the "signal". The intensity of the color indicates the predicted score. The plot in Fig. 13 shows the same events, but this time shading the histograms by the $\boldsymbol{z}$ LRP score.

In Fig. 12, a clear gradient is shown across the decision boundary between the two populations. This is very consistent across different trainings of the network. Figure 13 shows high LRP scores along certain subspaces that correspond to details of decision boundaries, and while the various trainings are qualitatively similar in their "broad" features, the individual details vary among them. This indicates that there are sub-spaces within the classification that are not identical between different trainings.

Another way to display this information is shown in Fig. 14. Here, the XAUG variables $\boldsymbol{z}$ and $\boldsymbol{\theta}$ are shown as 1D histograms, with their LRP relevance shown in the panel below.

The Signal is shown in blue and the Background is shown in orange. The network is placing a high magnitude of relevance on the values that are far from the overlap region ($\boldsymbol{z > 0.75}$, and $\boldsymbol{\theta > 0.10}$).

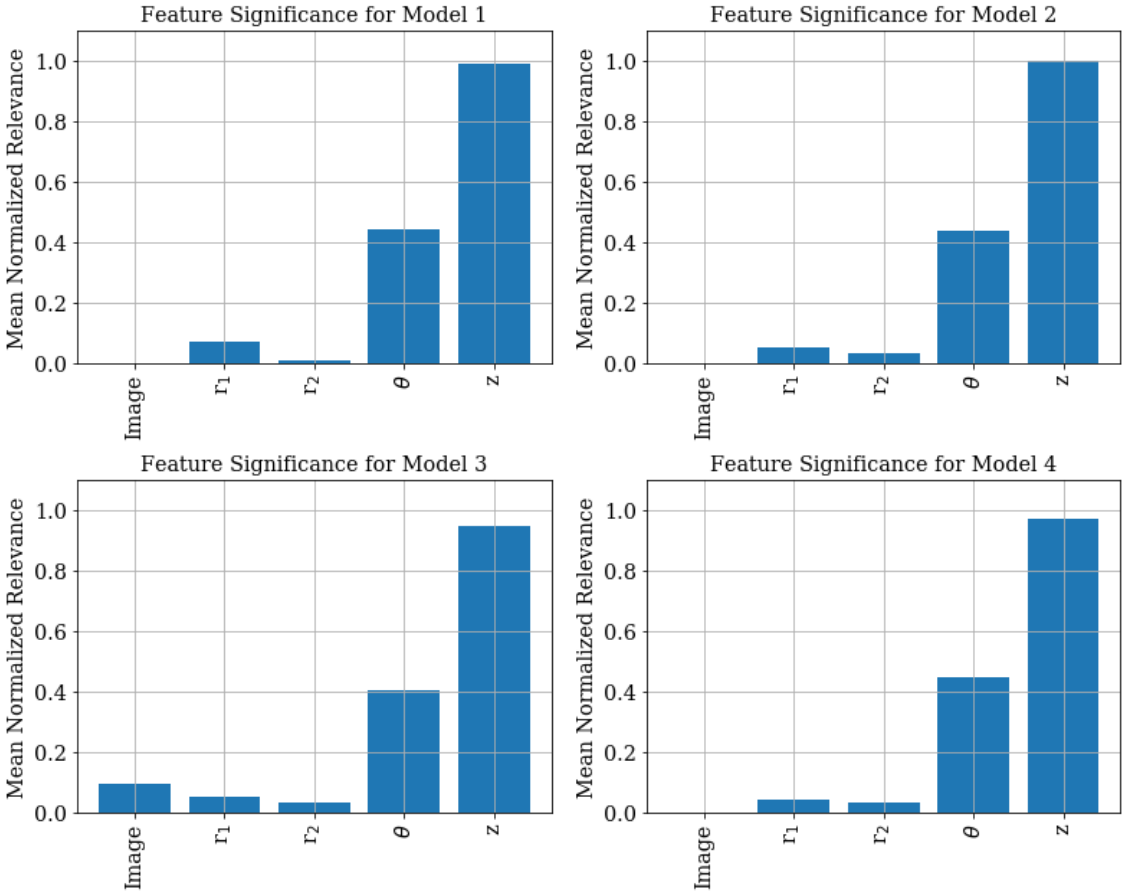


Figure 10: LRP scores for all variables in the Toy 2D CNN Model, with 4 separate trained models.

We performed the same studies shown above on the 1D CNN model. However, for brevity we show an average over 4 different trainings in our plots. The RMS of the 4 trainings are shown as uncertainties. Similar to the 2D CNN model, we can see from Fig.15 that the variables that were given the most relevance by LRP were the expert variables that had the greatest separation in combined phase space, $\boldsymbol{z}$ and $\boldsymbol{\theta}$. We can see a clear decision boundary in their combined phase space in Fig. 16. The left image again shows a clear gradient in the relevance of the variables, indicating that the network is unsure how to categorize the event in the overlap region.

The final model that we investigated was an RNN. This model showed much different relevance results from the other two, despite having comparable accuracy to the 1D and 2D CNN's. The RNN gives the greatest relevances to the constituent list variables, driven by the relevances of $\boldsymbol{z_1}$ (the values of $\boldsymbol{z}$ broken up among the leading jet constituents) and $\boldsymbol{z_2}$ (the values of $\boldsymbol{(1 - z)}$ broken up among the sub-leading jet constituents), as can be

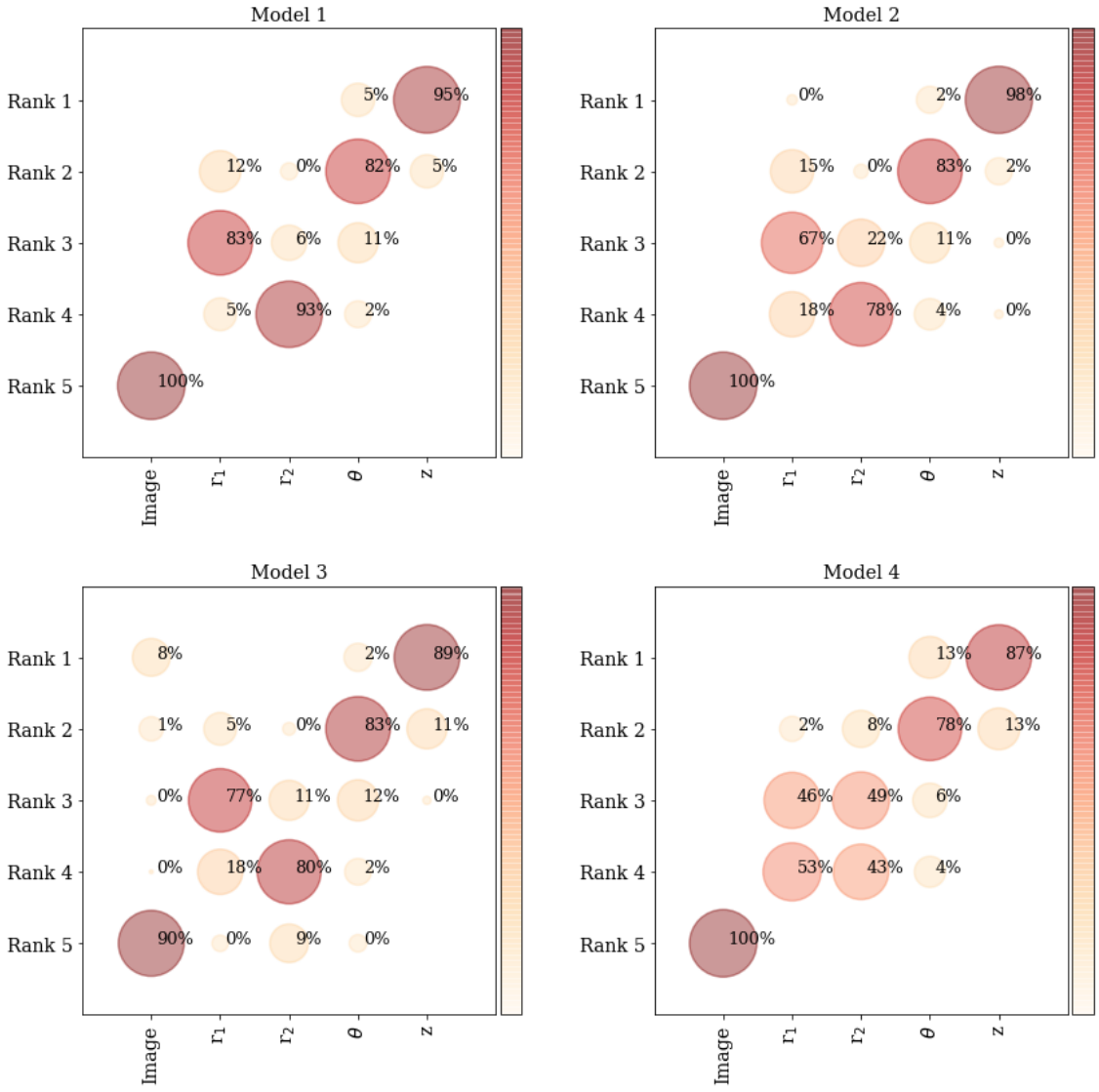


Figure 11: Ranked LRP scores for all variables in the Toy 2D CNN Model shown here for four separately trained models.

seen in the right plot of Fig. 15. $\boldsymbol{z_1}$ and $\boldsymbol{z_2}$ in encode the same information as $\boldsymbol{z}$, but in a different format. We still have the same leading variable, however the RNN gives the information more relevance in its constituent format than the event-level or XAUG format that the CNNs gave more relevance.

For the 2D CNN (Fig. 13) and the 1D CNN (left image in Fig. 16), the combined $\boldsymbol{z - \theta}$ phase space shows a clear decision boundary. The RNN, on the other hand, has a less apparent gradient (right image in Fig. 16), with most of the relevance in background. Again, this is because the network's recurrent layers are extracting information in a different way, looking at the list information much more than the 1D CNN.

The plots in this section provide two important pieces of information. Firstly, in the case

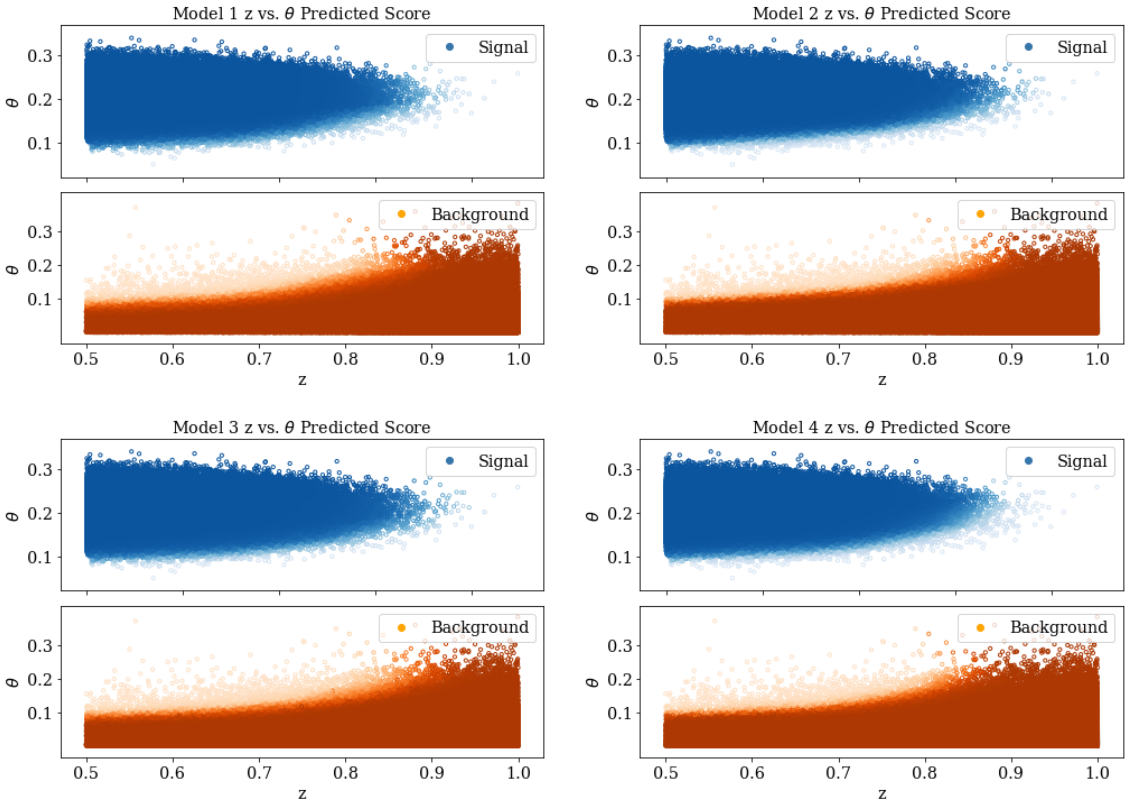


Figure 12: Predicted score for Toy Model for four separately trained models. The momentum fraction $\boldsymbol{z}$ is shown on the $\boldsymbol{x}-$axis while the angle $\boldsymbol{\theta}$ is shown on the $\boldsymbol{y}-$axis. The orange distribution is the "Background" and the blue distribution is the "Signal". The intensity of the color indicates the predicted scores.

where there are XAUG variables that dominantly capture the information in the system, the LRP accurately selects the most important variables, rendering the image redundant, as seen in Fig. 10. Secondly, the network can arrive at different optimizations, each with different sub-leading relevance, which can impact the confidence of the prediction in regions of confusion. However, these separately trained models have nearly identical performance (see Fig. 12), despite clearly arriving at different minima (see Fig. 13).

These are not related to insufficient training data, as we have checked this with both low-and high-statistics samples and the features persist. This leads to a recommendation about numerical uncertainty in the classifier.

5.2 Numerical uncertainty in the classifier

In the above studies, the LRP distributions of the various XAUGs show that the main decision boundaries made by the classifier are relatively stable. However, there are details in the local approximation (LRP) boundaries that are caused by differences in relevance among sub-leading features.

The LRP score is a reliable way to quantify the most relevant variables. However,

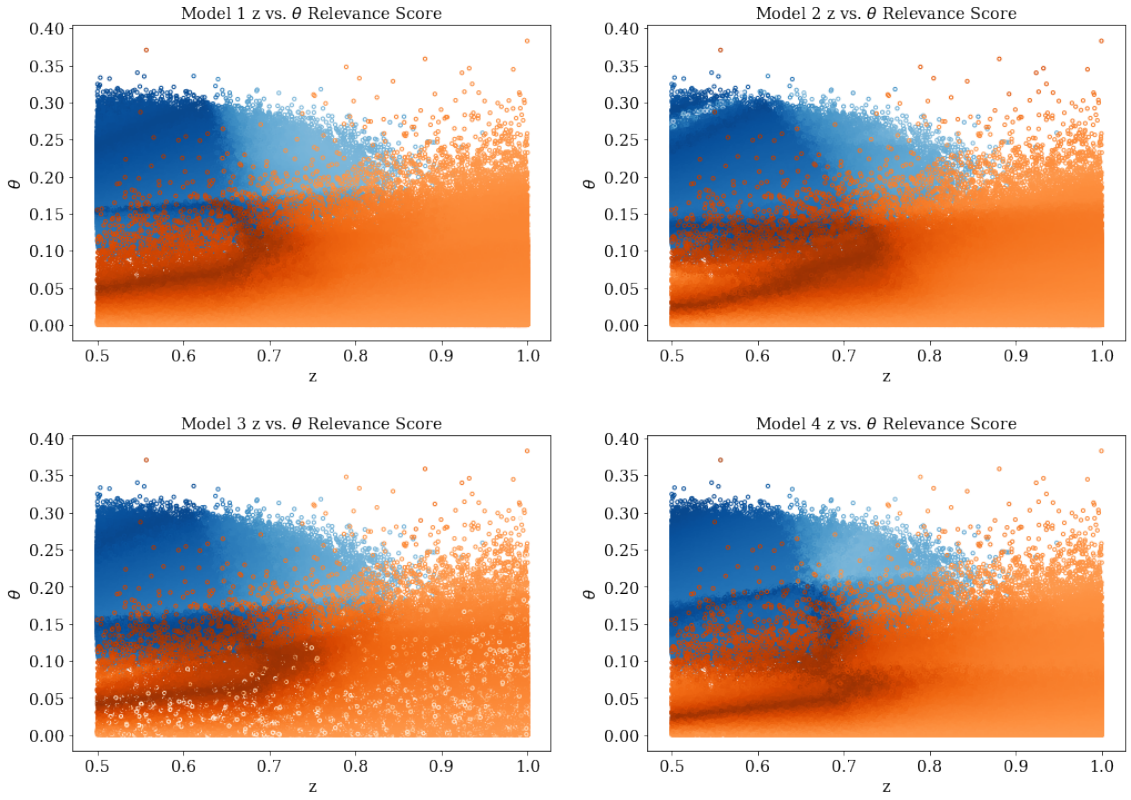
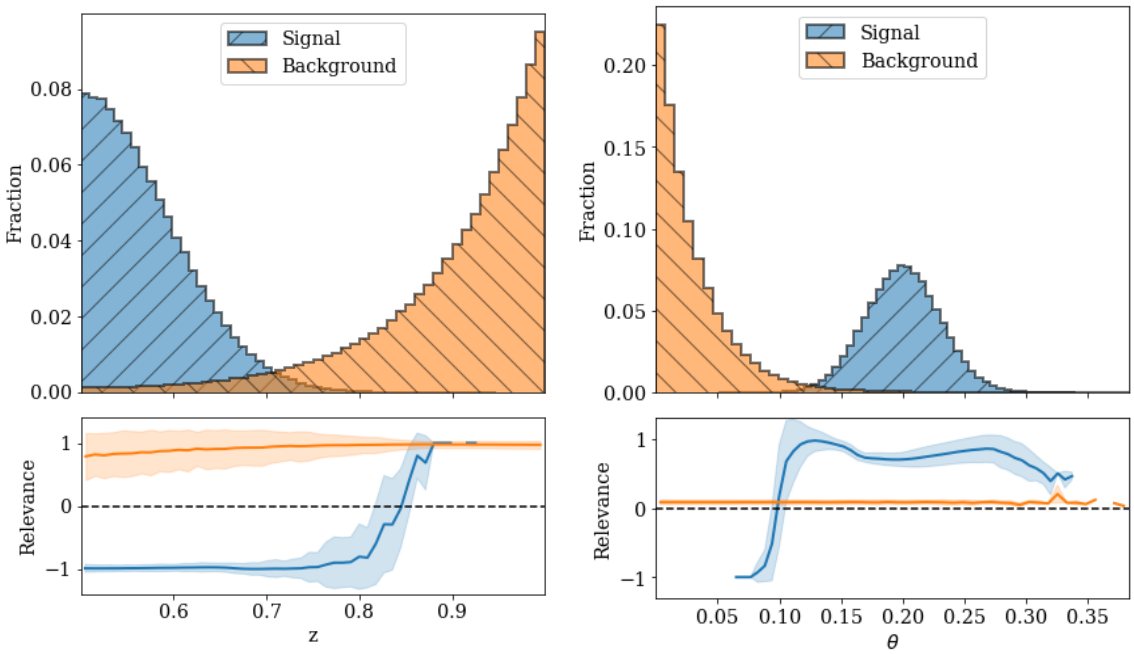


Figure 13: Relevance (LRP) scores for Toy Model 1D CNN for four separately trained models. The momentum fraction $\boldsymbol{z}$ is shown along the horizontal axis while the angle $\boldsymbol{\theta}$ is shown on the vertical axis. The orange distribution is the "background" while the blue distribution is the "signal". The intensity of the color indicates the summed LRP score, with darker colors representing more confident predictions of the model.

this also means that the optimization with respect to less relevant variables is not easily determined, with very different local behavior for different trainings. This points to the need for a numerical uncertainty of classifiers that is not easily visible in the final classifier output, since the latter is primarily sensitive to the most relevant variables. Decisions based on less relevant variables can vary depending on the training. Optimizers like ADAM have a difficult time optimizing along these axes because they are so shallow, and the optimizer loss functions are dominated by the most relevant variables.

A consequence of this study is that it is very important to perform multiple different trainings for networks, especially if the decisions are made on sub-dominant features. The trustworthiness of individual classifier decisions is very high in the case of differences in features with high relevance, but is considerably lower for features with low relevance.

In the remaining study, we will therefore train the network 4 times and take the average response. The RMS of the predictions can then be used also for an uncertainty quantification, which we add for the relevance scores. These have similar interpretations to other systematic uncertainties, inasmuch as differences that exist, but are covered within the



(a) z profile

(b) theta profile

Figure 14: Profiles for Toy 2D CNN Model with relevance and predictions averaged over the four models previously shown.

uncertainty, are not trustworthy differences.

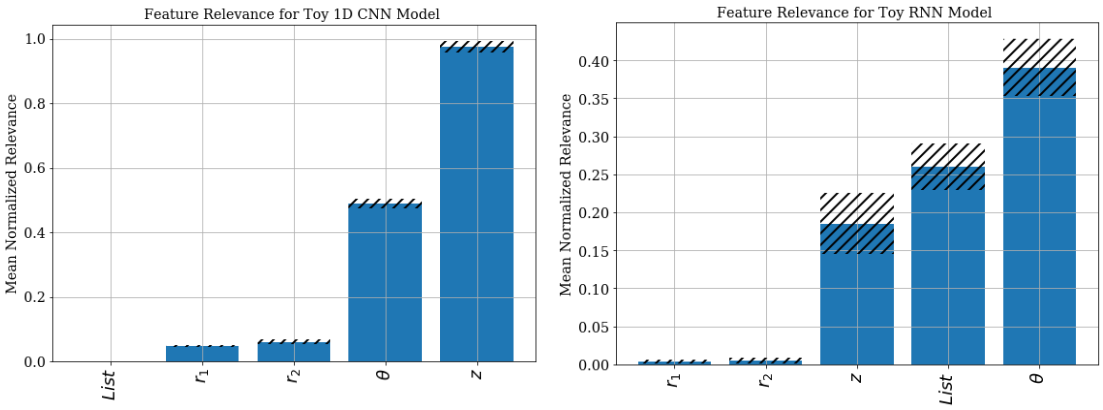


Figure 15: Mean normalized relevance score for 4 trainings of the toy 1D CNN(toy 1D RNN) on the left(right), where the relevance scores of constituent variables List ($d\boldsymbol{\alpha_1}$, $d\boldsymbol{\beta_1}$, $d\boldsymbol{\alpha_1}$, $d\boldsymbol{\alpha_2}$, $\boldsymbol{z_{const,1}}$, $\boldsymbol{z_{const,2}}$) are summed.

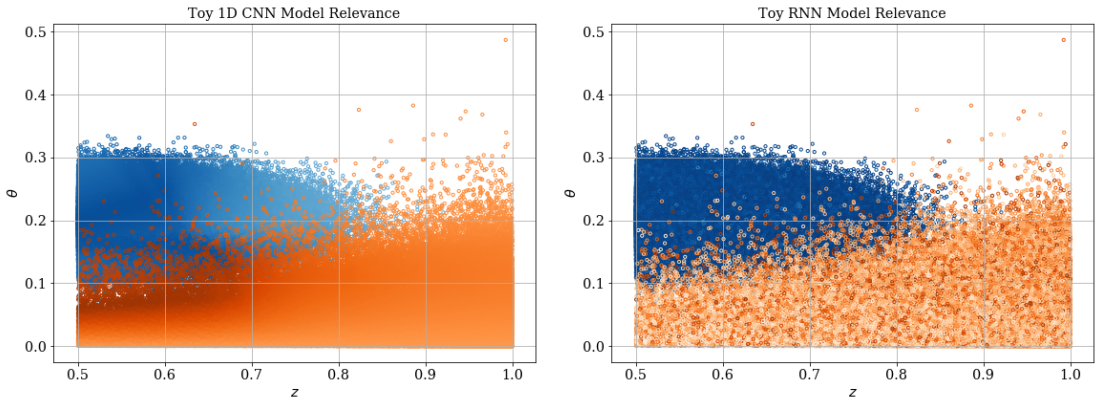


Figure 16: Mean LRP scores for z and $\boldsymbol{\theta}$ variables in the Toy 1D CNN Model (left), and RNN (right).

5.3 Particle Model Explanations

The Toy Model introduced previously is intended as a highly simplified "cartoon" of jet substructure. The angle $\boldsymbol{\theta}$ and momentum fraction $\boldsymbol{z}$ of the "subjets" in the Toy Model are analogous to the $\boldsymbol{\Delta R}$ and momentum fraction $\boldsymbol{z}$ of the actual subjets from the soft drop algorithm. The drastic separation observed in the Toy Model will be reduced, although we can use the insight and techniques developed to apply to the particle-level simulation.

Distributions of the normalized variables are shown in the upper portions of some variables in Figs. 21 and in the Appendix for all variables in 25-30. The jet images from a few individual events, along with their LRP heatmaps, are shown in Fig.17. The background events are more spread out than the signal events, which tend to be clustered among one or two subjets. This is exploited by the network, as can be observed in the bottom panes. The network identifies signal by weighting toward the central two subjets, while the network identifies background from pixels away from the two subjets. These images provide similar information to those in Ref. [26], which show the Pearson Correlation Coefficient.

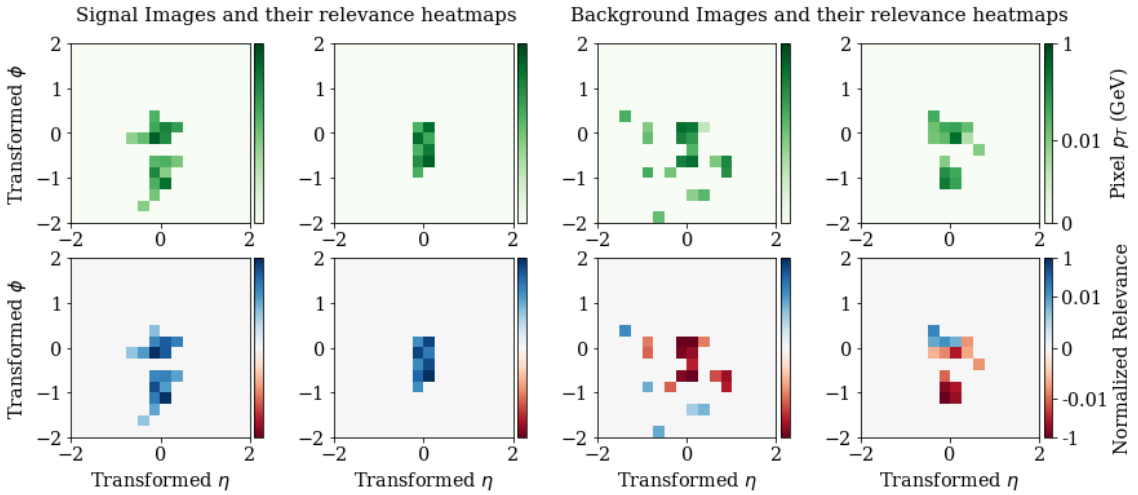


Figure 17: Individual Particle Model Signal (left two columns) and Background (right two columns) images (top) and their corresponding LRP score heatmaps (bottom).

The predicted scores and LRP scores for the 2D CNN trained on the Particle Model are shown in Fig. 18 for the rescaled angular separation $\boldsymbol{\Delta R_{subjets}}$ versus the rescaled momentum fraction of the subjets $\boldsymbol{z}$. These are the analogous plots to our cartoon toy model, shown in Fig. 18. It is clear that the separation between signal and background is reduced and the separation in these geometric variables is not as clear to the eye.

However, the simple intuition from using XAUGs can be translated to more appropriate variables. In particular, Fig. 19 shows the analogous distribution for two XAUGs, the rescaled groomed jet mass $\boldsymbol{m_{jet,SD}}$ and the rescaled $\boldsymbol{\tau^1_{3,SD}}$. It is clear that there are separations between the signal and background populations. The region for the rescaled $\boldsymbol{m_{jet,SD}}$ between 0.35 and 0.4 (corresponding to the Z mass window) tends to be preferred as a signal region, and just outside that region is preferred as background. In addition, there is a clear correlation between $\boldsymbol{m_{jet,SD}}$ and $\boldsymbol{\tau^1_{3,SD}}$ that the network utilizes to identify

background in the upper left and lower right portions of the plot (low $\boldsymbol{\tau}$ at high mass, or high $\boldsymbol{\tau}$ at low mass). Furthermore, Fig. 20 shows the groomed versus ungroomed jet mass. This gives information about the fraction of the jet that is groomed away compared to the original, which can disentangle the hard and soft parts of the jet. The network clearly makes use of this information. In both of these plots, the network has regions of extremely high confidence that offer clear distinction between signal and background.

Another way to see the same information is to look at the individual variables from the 2D CNN trained on the Particle Model in Figs. 21 for highly relevant variables, and in Figs. 25- 30 in the appendix for all variables. These plots show the distribution of the XAUGs in the top pane, as well as a profile distribution of their relevance scores in the bottom pane. The relevance score shows the contribution to the correct classification of either signal or background.

It is possible to use the information gathered by the LRP procedure to quantify the impact of each individual variable on the decision of the network. Figures 22 and 23 show the mean relevance score over four models for each input in the 2D CNN and 1D CNN, respectively. In these cases, the relevance scores from the image and list are summed and displayed as a single bar in the histogram. The uncertainties on the scores are drawn from the RMS of four separate training iterations of the CNN. It is clear that there is a hierarchy of variables, with 5-6 very important ones with relevances above 0.3, around a dozen with moderate importance between 0.1 to 0.3, and around 5-6 with low relevances below 0.1.

In order to understand how much information is being used for the decision making, we can investigate the performance of our networks in several cases, shown in Fig. 24. Due to the fact that we are using a highly simplified model at the particle level only (for demonstration), the ROC curves are (over-)optimistic about performance, although it is still instructive to investigate differences to understand relevant features.

First, as a baseline, we can look at the simplest case of MLPs using only the XAUGs as inputs. We will look at two such cases, one where we do not use flavor information (particle content, $\boldsymbol{d_{xy}}$, and $\boldsymbol{d_z}$), and then when this information is added. These are labeled "XAUGs only (no flavor)" and "XAUGs only").

Next, we can investigate the 2D CNN with only the jet images as input (labeled as "Images only" in the figure). We then combine the information together from the image and the 5 XAUGs with highest relevance ("Images + 5 XAUGs") or with all XAUGs ("Images + XAUGS").

We can repeat the above cases with the particle list for the 1d CNN ("Particle List"), and then add the top 5 XAUGS ("Particle List + 5 XAUGS") and all XAUGS ("Particle List + XAUGS"). The RNN can also be similarly investigated with and without XAUGs, shown in "RNN - Particle List" and "RNN - Particle List + XAUGs".

We see that the image-only NN performs the worst, with an AUC of 0.932. The XAUG variables perform better, achieving AUC of 0.957 (0.976) without (with) flavor information. Combining the image with the XAUGs achieves an AUC of 0.984. Combining the particle list with the XAUGs achieves AUCs of 0.994 and 0.995 for the top 5, and all XAUGs, respectively. Similarly, the RNN achieves an AUC=0.975 regardless of whether the XAUGs are used.

These results demonstrate that adding XAUG variables can considerably improve network performance. In fact, the XAUGs-only performance is already fairly good at distinguishing between $Z \to b\bar{b}$ and QCD. It may be quite useful for some simplistic cases to consider these simple networks, guided by what DNNs find important. In cases where more discrimination is necessary, combining XAUGs with the network can act as a sort of guide for the optimization, achieving better performance together than either achieves individually. It also gives a robust method for investigation of subspaces of relevance that can be understood easily by analyzers.

Furthermore, the cases where we add only the 5 XAUGs with the highest relevance scores are almost identical to using all XAUGs. We can therefore conclude that the XAUGs with the highest relevance scores are the primary decision factors that the network is making, and could be used by analysts to audit and understand network behavior. In addition, only these few XAUGs would be needed to greatly improve performance.

In addition to the above observations, Fig. 22 shows considerable variation from one training to another in the local behavior of networks. The local network optimization depends weakly on features with low relevance, so decisions based on these features should be trusted to a somewhat lesser degree. These findings suggest that networks should be trained multiple times, with the average taken and variations counted as uncertainties.

In our case, the networks tend to be strongly affected by dominant smaller subspaces related to a few shape-based variables, jet mass with and without grooming, and the lifetime information. Such an analysis combining LRP and XAUGs could also be applied in other use cases to rank salient features.

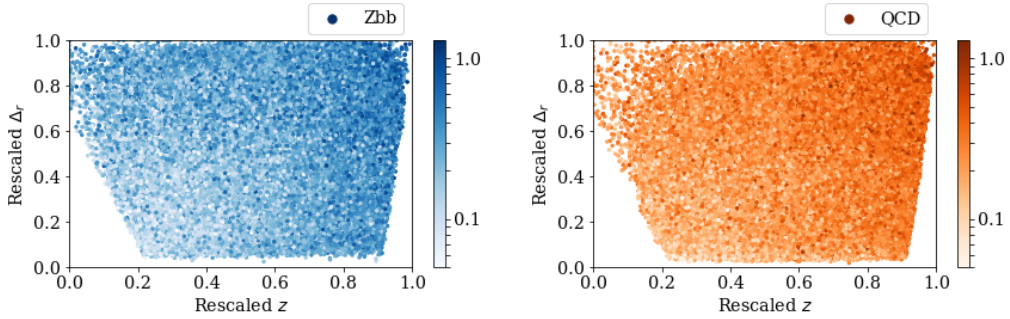


Figure 18: Averaged relevance score of 5 models for the particle list CNN for $\boldsymbol{\Delta R_{subjets}}$ vs $\boldsymbol{z}$ for the Zbb and QCD samples.

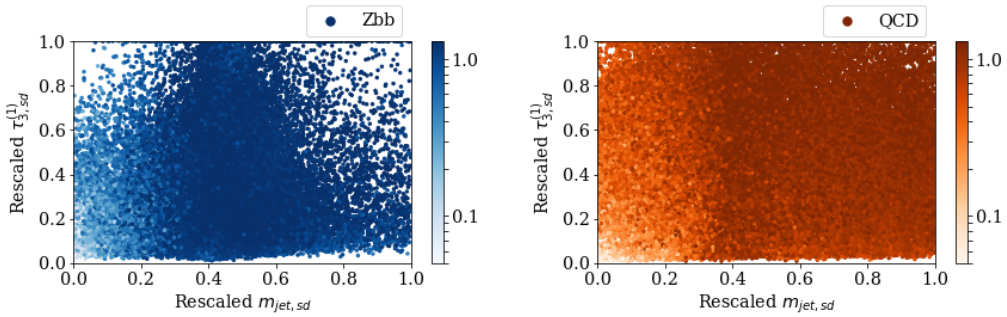


Figure 19: Averaged relevance score of 5 models for the particle list CNN for $\boldsymbol{\tau_{3,SD}^{1.0}}$ vs $\boldsymbol{m_{jet,SD}}$ for the Zbb and QCD samples. The features to compare are selected from Figure 23.

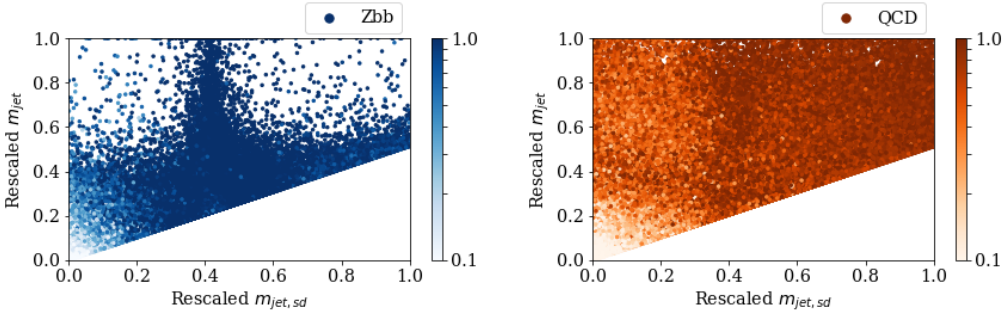


Figure 20: Averaged relevance score over 5 models for particle list CNN for ungroomed mass $\boldsymbol{m_{jet}}$ vs groomed mass $\boldsymbol{m_{jet,SD}}$ for the Zbb and QCD samples.

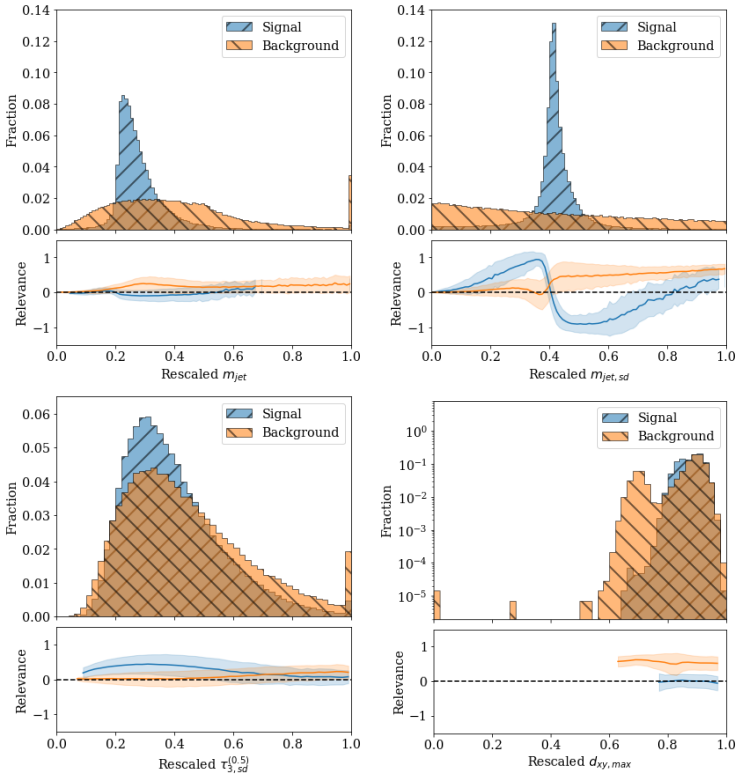


Figure 21: Histograms with profiles of LRP relevances normalized per event and averaged over four models. Signal and background are the Zbb and QCD samples, respectively.

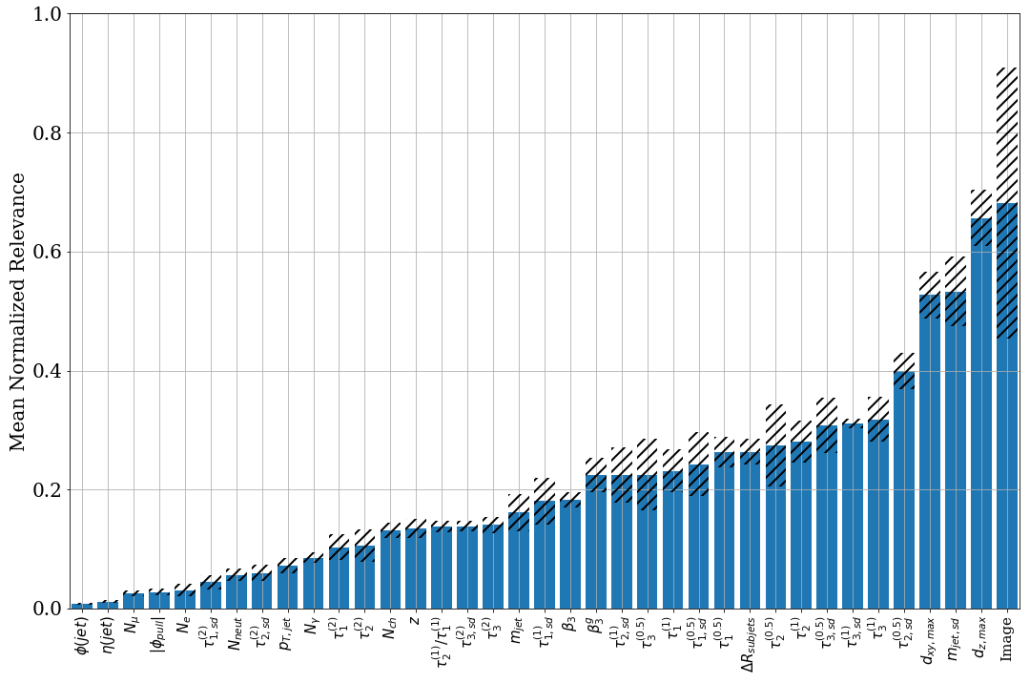


Figure 22: Input features with the greatest mean normalized relevance after averaging the relevance scores of four models for the 2D CNN. These models use ***Zbb*** and QCD simulation as Signal and Background, respectively. The relevance of the image is the summed relevance of all the pixels in the image.

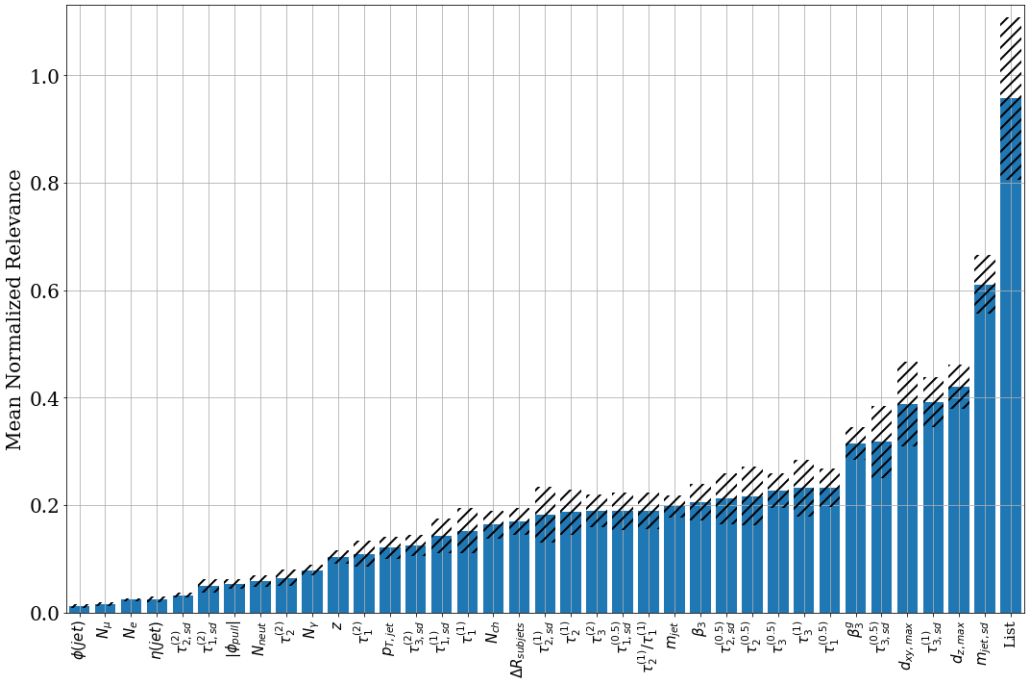


Figure 23: Input features with the greatest mean normalized relevance after averaging the relevance scores of four models for the particle list CNN. 'List' is the summed relevance of the 20 particle inputs per event and the 17 particle list inputs.

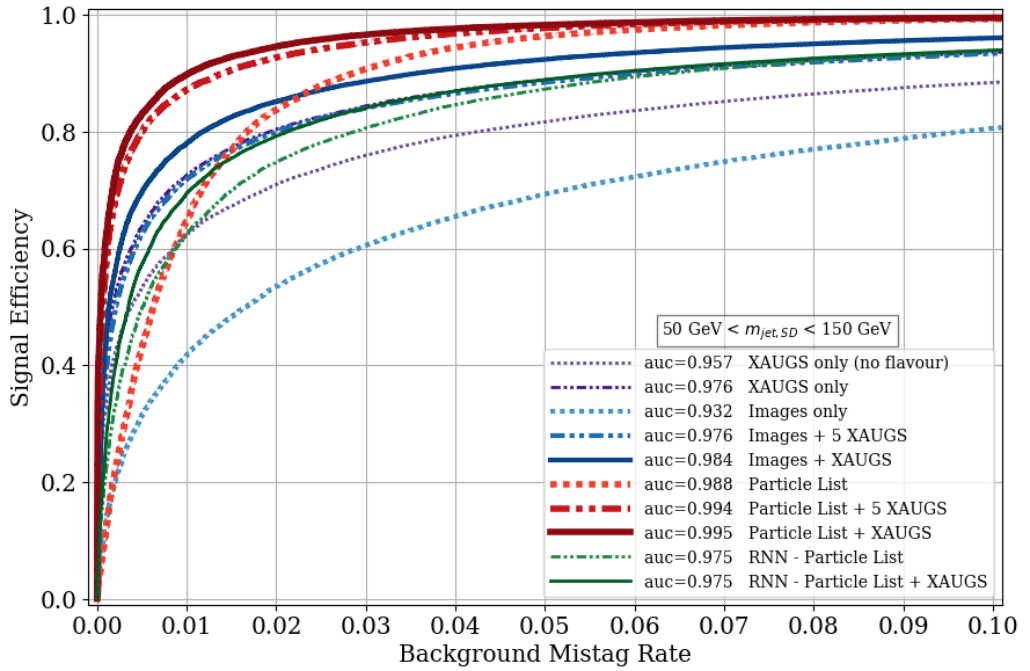


Figure 24: ROC curves from Image CNN, Particle List CNN, DNN with XAUGs only, and RNN, with mass window cut of $\mathbf{50 < m_{jet,SD} < 150}$ GeV.

6 Conclusion

We have presented a scheme to systematically explain jet identification decisions from a deep neural network (DNN) using eXpert AUGmented variables (XAUG) and layerwise relevance propagation (LRP). The combination of these two techniques can shed light on network decisions that would otherwise be difficult to ascertain. The XAUG variables can also be used alone for classification, or can be used in conjunction with lower-level / higher-dimensional classifiers to "guide" the network decisions, resulting in better performance. In some cases, XAUG variables can even capture the information of the lower-level networks entirely if there are subspaces of the network optimization that are correlated with the XAUGs themselves.

An additional benefit to using LRP and XAUGs is the ability to investigate relevant subspaces of the training. This can highlight when there are very shallow optimizations, where training multiple times could lead to slightly different subspaces. In such cases, it is recommended to train multiple times and take the mean of the predictions.

Acknowledgments

This work was supported under NSF Grants PHY-1806573, PHY-1719690 and PHY-1652066. Computations were performed at the Center for Computational Research at the University at Buffalo.

We would like to acknowledge Kyle Cranmer, Marat Freytsis, Loukas Gouskos, Gregor Kasieczka, Martin Kwok, Simone Marzani, David Miller, Ben Nachman, Juska Pekkanen, Jesse Thaler, Daniel Whiteson, and David Yu for helpful conversations during the BOOST 2020 poster session and manuscript review.

A Software Versions

We used `PYTHIA` v8.2.35 [34] with the default parameters, with events stored in the `ROOT` [42] format. Both the toy model and particle-level simulations were processed with `coffea` v0.6 [43] using `uproot` v3.11 [44] and `AwkwardArray` v0.12.0rc1 [45]. We used `tensorflow` v1.13 [46] and `inNvestigate` [32] for the DNNs and LRP, respectively.

Our software can be found here on github.

B Relevance Scores for All XAUGs

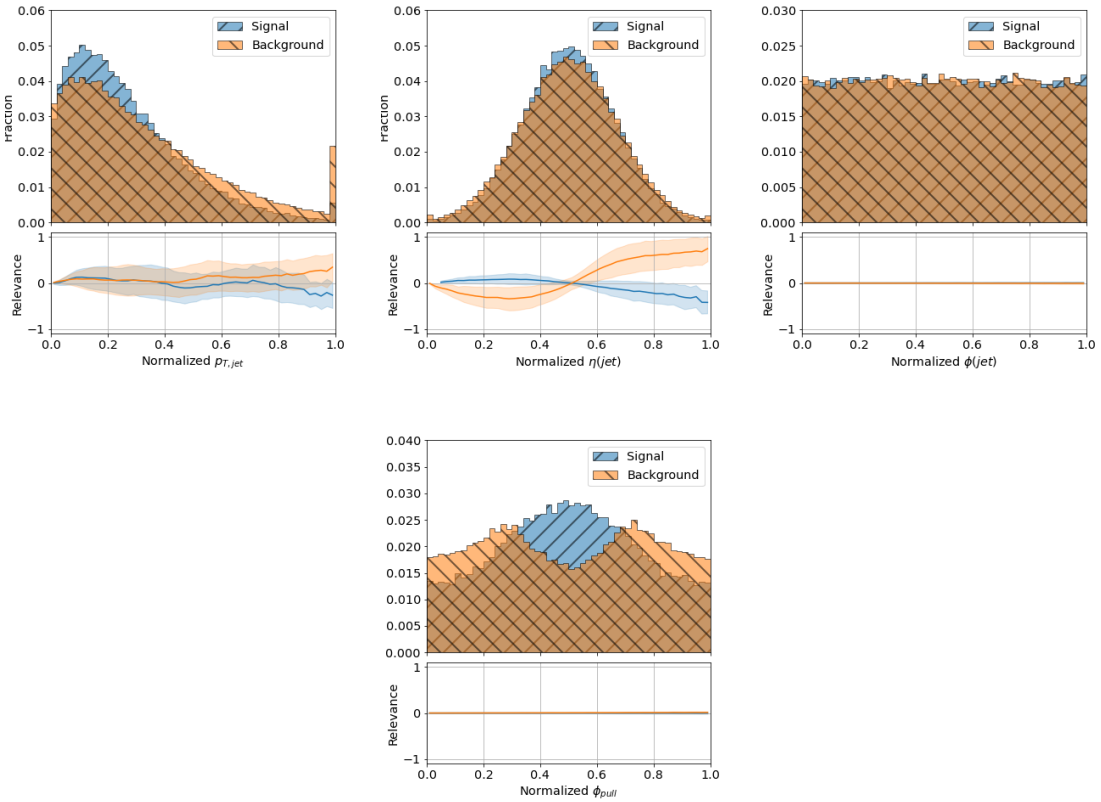


Figure 25: Histograms with profiles of normalized LRP relevances.

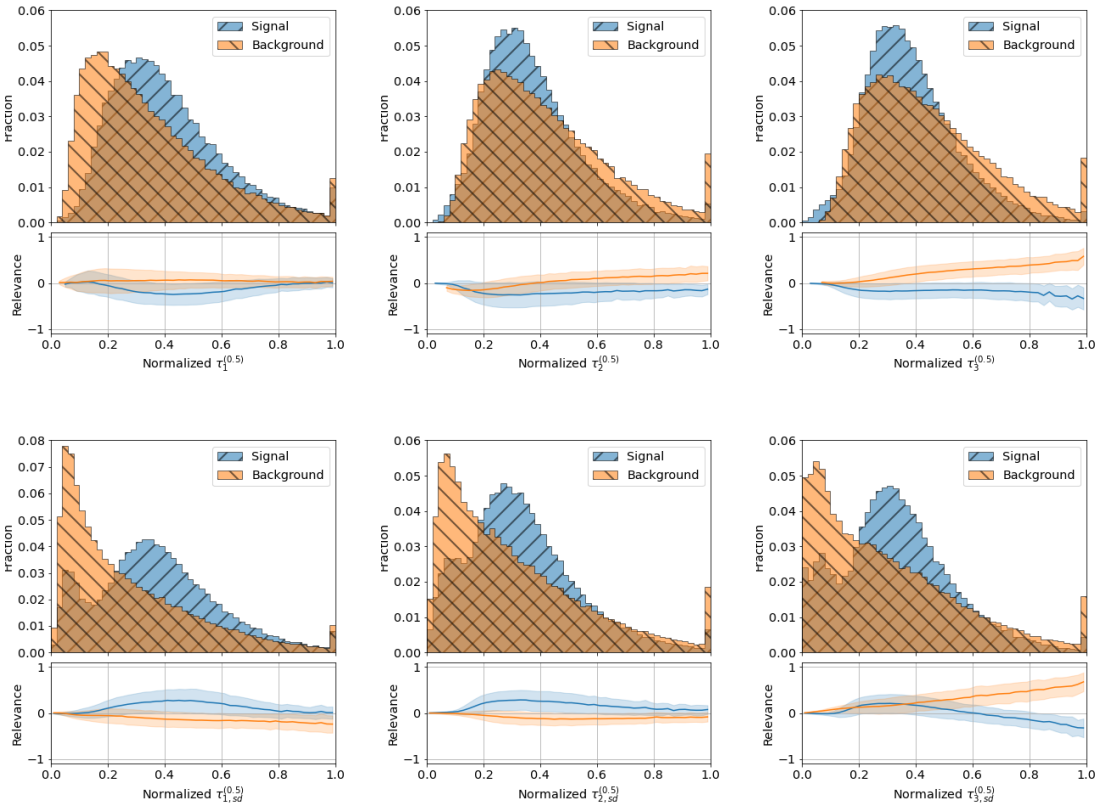


Figure 26: Histograms with profiles of normalized LRP relevances.

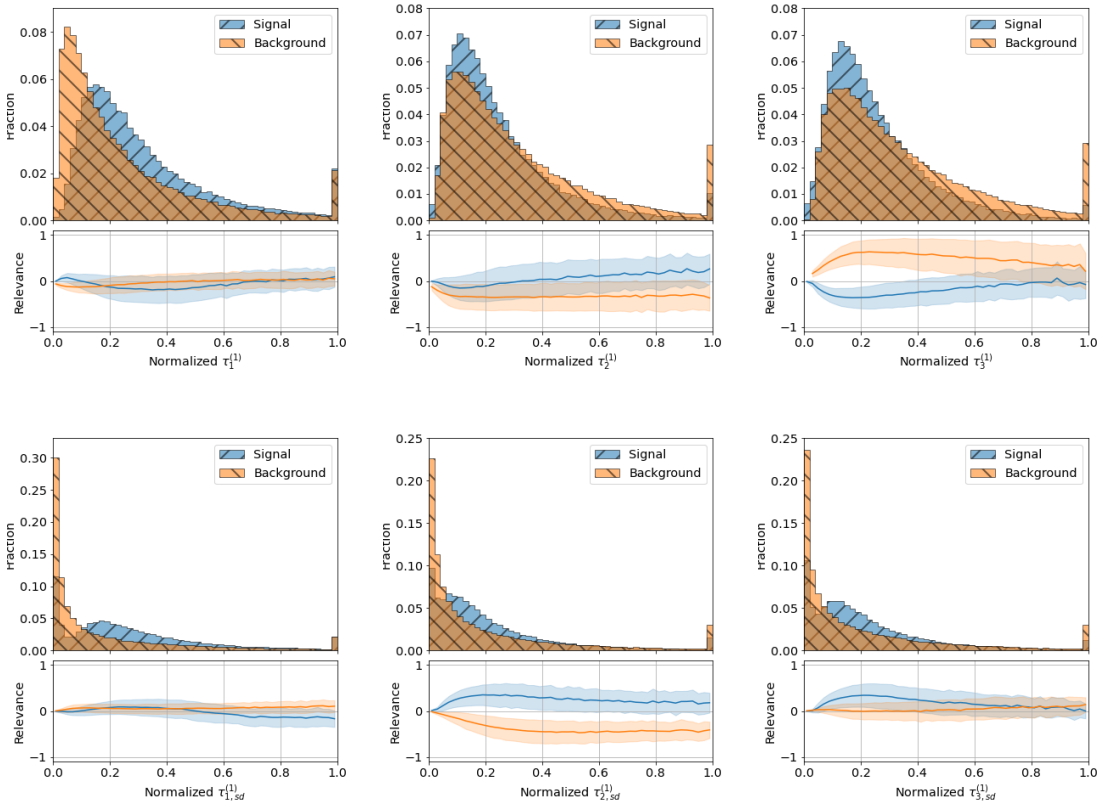


Figure 27: Histograms with profiles of normalized LRP relevances.

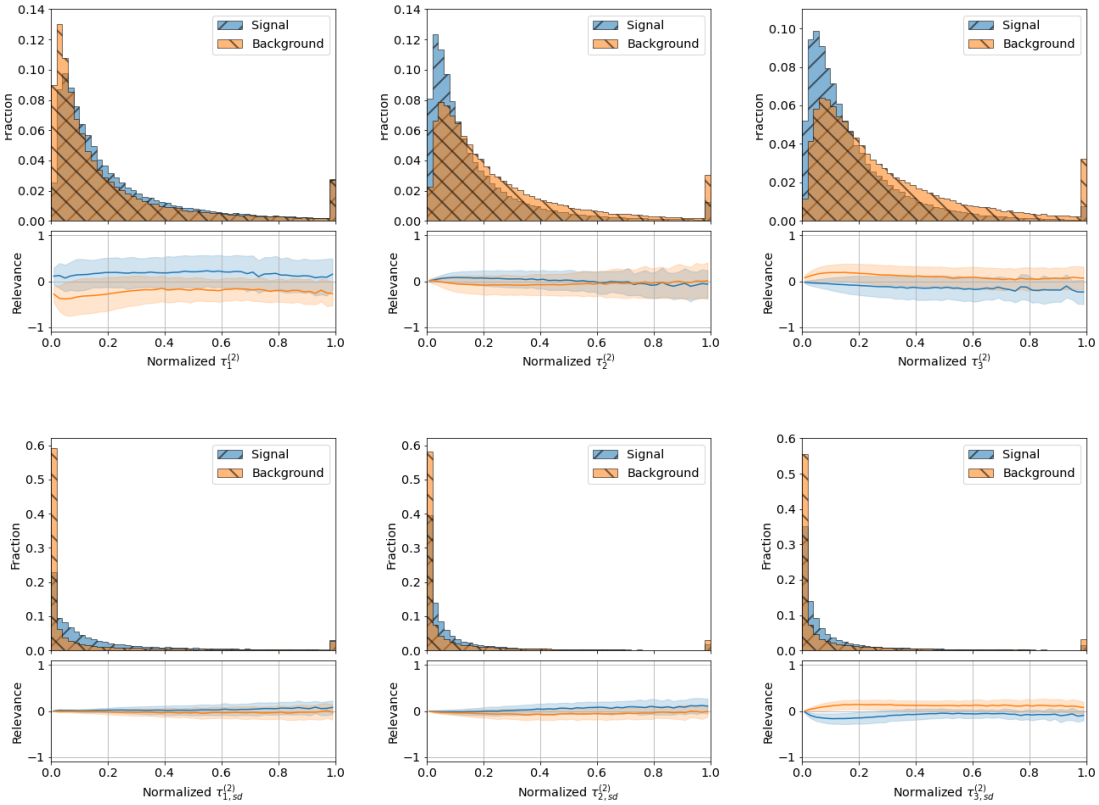


Figure 28: Histograms with profiles of normalized LRP relevances.

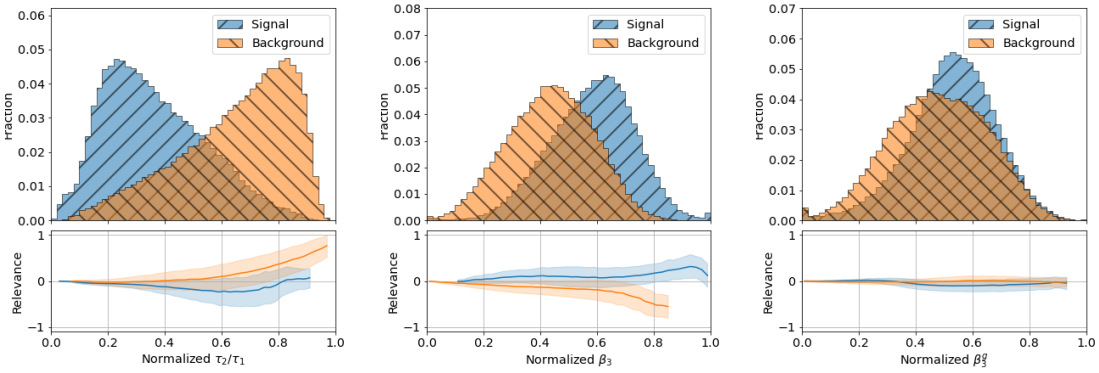


Figure 29: Histograms with profiles of normalized LRP relevances.

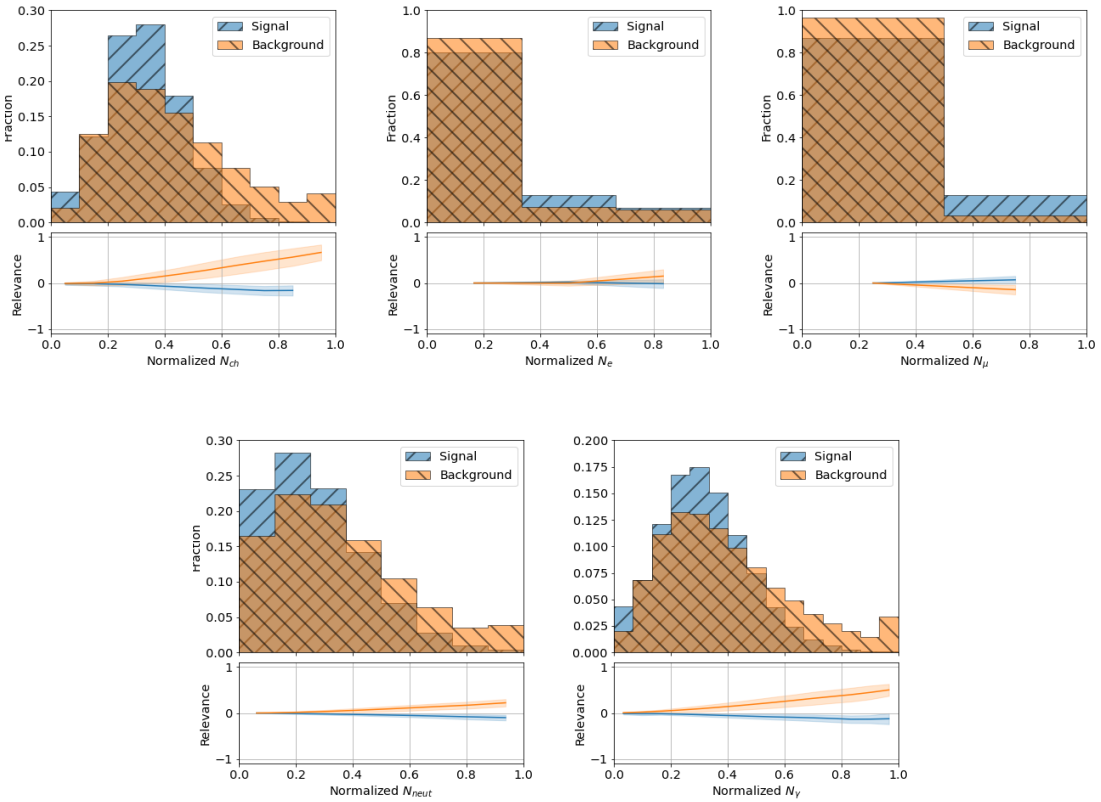


Figure 30: Histograms with profiles of normalized LRP relevances.

References

[1] J. Zhang, R. Zhang, Y. Huang, and Q. Zou, *Unsupervised part mining for fine-grained image classification*, *CoRR* **abs/1902.09941** (2019) [arXiv:1902.09941].

[2] A. Abdesselam et al., *Boosted Objects: A Probe of Beyond the Standard Model Physics*, *Eur. Phys. J.* **C71** (2011) 1661, [arXiv:1012.5412].

[3] A. Altheimer et al., *Jet Substructure at the Tevatron and LHC: New results, new tools, new benchmarks*, *J. Phys.* **G39** (2012) 063001, [arXiv:1201.0008].

[4] A. Altheimer et al., *Boosted Objects and Jet Substructure at the LHC. Report of BOOST2012, held at IFIC Valencia, 23rd-27th of July 2012*, *Eur. Phys. J.* **C74** (2014), no. 3 2792, [arXiv:1311.2708].

[5] D. Adams et al., *Towards an Understanding of the Correlations in Jet Substructure*, *Eur. Phys. J.* **C75** (2015), no. 9 409, [arXiv:1504.00679].

[6] A. J. Larkoski, I. Moult, and B. Nachman, *Jet Substructure at the Large Hadron Collider: A Review of Recent Advances in Theory and Machine Learning*, *Phys. Rept.* **841** (2020) 1–63, [arXiv:1709.04464].

[7] R. Kogler et al., *Jet Substructure at the Large Hadron Collider: Experimental Review*, *Rev. Mod. Phys.* **91** (2019), no. 4 045003, [arXiv:1803.06991].

[8] D. Guest, J. Collado, P. Baldi, S.-C. Hsu, G. Urban, and D. Whiteson, *Jet Flavor Classification in High-Energy Physics with Deep Neural Networks*, *Phys. Rev.* **D94** (2016), no. 11 112002, [arXiv:1607.08633].

[9] W. J. Murdoch, C. Singh, K. Kumbier, R. Abbasi-Asl, and B. Yu, *Definitions, methods, and applications in interpretable machine learning*, *Proceedings of the National Academy of Sciences* **116** (Oct, 2019) 22071–22080.

[10] Z. C. Lipton, *The mythos of model interpretability*, *CoRR* **abs/1606.03490** (2016) [arXiv:1606.03490].

[11] M. T. Ribeiro, S. Singh, and C. Guestrin, *"why should I trust you?": Explaining the predictions of any classifier*, *CoRR* **abs/1602.04938** (2016) [arXiv:1602.04938].

[12] A. Björklund, A. Henelius, E. Oikarinen, K. Kallonen, and K. Puolamäki, *Sparse robust regression for explaining classifiers*, in *Discovery Science* (P. Kralj Novak, T. Šmuc, and S. Džeroski, eds.), (Cham), pp. 351–366, Springer International Publishing, 2019.

[13] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. MÃŒller, and W. Samek, *On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation*, *PLOS ONE* **10** (07, 2015) 1–46.

[14] W. Samek, T. Wiegand, and K. Müller, *Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models*, *CoRR* **abs/1708.08296** (2017) [arXiv:1708.08296].

[15] G. Montavon, A. Binder, S. Lapuschkin, W. Samek, and K.-R. Müller, *Layer-Wise Relevance Propagation: An Overview*, pp. 193–209. Springer International Publishing, Cham, 2019.

[16] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K. Muller, *How to explain individual classification decisions*, *Journal of Machine Learning Research* **11** (6, 2010) 1803–1831.

[17] K. Datta and A. Larkoski, *How Much Information is in a Jet?*, *JHEP* **06** (2017) 073, [arXiv:1704.08249].

[18] S. H. Lim and M. M. Nojiri, *Spectral Analysis of Jet Substructure with Neural Networks: Boosted Higgs Case*, *JHEP* **10** (2018) 181, [arXiv:1807.03312].

[19] A. Chakraborty, S. H. Lim, and M. M. Nojiri, *Interpretable deep learning for two-prong jet classification with jet spectra*, *JHEP* **07** (2019) 135, [arXiv:1904.02092].

[20] K.-F. Chen and Y.-T. Chien, *Deep learning jet substructure from two-particle correlations*, *Physical Review D* **101** (Jun, 2020).

[21] G. Kasieczka, S. Marzani, G. Soyez, and G. Stagnitto, *Towards machine learning analytics for jet substructure*, *Journal of High Energy Physics* **2020** (Sep, 2020).

[22] **CMS** Collaboration, A. M. Sirunyan et al., *Identification of heavy, energetic, hadronically decaying particles using machine-learning techniques*, *JINST* **15** (2020), no. 06 P06005, [arXiv:2004.08262].

[23] V. Mikuni and F. Canelli, *Abcnet: an attention-based method for particle tagging*, *The European Physical Journal Plus* **135** (Jun, 2020).

[24] G. Kasieczka, T. Plehn, M. Russell, and T. Schell, *Deep-learning Top Taggers or The End of QCD?*, *JHEP* **05** (2017) 006, [arXiv:1701.08784].

[25] J. Cogan, M. Kagan, E. Strauss, and A. Schwartzman, *Jet-Images: Computer Vision Inspired Techniques for Jet Tagging*, *JHEP* **02** (2015) 118, [arXiv:1407.5675].

[26] L. de Oliveira, M. Kagan, L. Mackey, B. Nachman, and A. Schwartzman, *Jet-images — Deep Learning Edition*, *JHEP* **07** (2016) 069, [arXiv:1511.05190].

[27] J. Thaler and K. Van Tilburg, *Identifying Boosted Objects with N-subjettiness*, *JHEP* **03** (2011) 015, [arXiv:1011.2268].

[28] J. Thaler and K. Van Tilburg, *Maximizing Boosted Top Identification by Minimizing N-subjettiness*, *JHEP* **02** (2012) 093, [arXiv:1108.2701].

[29] J. Gallicchio and M. D. Schwartz, *Seeing in Color: Jet Superstructure*, *Phys. Rev. Lett.* **105** (2010) 022001, [arXiv:1001.5027].

[30] T. Faucett, J. Thaler, and D. Whiteson, *Mapping Machine-Learned Physics into a Human-Readable Space*, arXiv:2010.11998.

[31] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller, *Explaining nonlinear classification decisions with deep taylor decomposition*, *Pattern Recognition* **65** (May, 2017) 211–222.

[32] M. Alber, S. Lapuschkin, P. Seegerer, M. Hägele, K. T. Schütt, G. Montavon, W. Samek, K.-R. Müller, S. Dähne, and P.-J. Kindermans, *innvestigate neural networks!*, 2018.

[33] M. Dasgupta, A. Fregoso, S. Marzani, and G. P. Salam, *Towards an understanding of jet substructure*, *JHEP* **09** (2013) 029, [arXiv:1307.0007].

[34] 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** (2015) 159–177, [arXiv:1410.3012].

[35] J. Dolen, P. Harris, S. Marzani, S. Rappoccio, and N. Tran, *Thinking outside the ROCs: Designing Decorrelated Taggers (DDT) for jet substructure*, *JHEP* **05** (2016) 156, [arXiv:1603.00027].

[36] C. Shimmin, P. Sadowski, P. Baldi, E. Weik, D. Whiteson, E. Goul, and A. Søgaard, *Decorrelated Jet Substructure Tagging using Adversarial Neural Networks*, *Phys. Rev.* **D96** (2017), no. 7 074034, [arXiv:1703.03507].

[37] O. Kitouni, B. Nachman, C. Weisser, and M. Williams, *Enhancing searches for resonances with machine learning and moment decomposition*, arXiv:2010.09745.

[38] M. Cacciari and G. P. Salam, *Dispelling the N^3 myth for the k_t jet-finder*, *Phys. Lett.* **B641** (2006) 57–61, [hep-ph/0512210].

[39] M. Cacciari, G. P. Salam, and G. Soyez, *FastJet User Manual*, *Eur. Phys. J.* **C72** (2012) 1896, [arXiv:1111.6097].

[40] M. Cacciari, G. P. Salam, and G. Soyez, *The anti-k_T jet clustering algorithm*, *JHEP* **04** (2008) 063, [arXiv:0802.1189].

[41] A. J. Larkoski, S. Marzani, G. Soyez, and J. Thaler, *Soft Drop*, *JHEP* **05** (2014) 146, [arXiv:1402.2657].

[42] R. Brun, F. Rademakers, P. Canal, A. Naumann, O. Couet, L. Moneta, V. Vassilev, S. Linev, D. Piparo, G. GANIS, B. Bellenot, E. Guiraud, G. Amadio, wverkerke, P. Mato, TimurP, M. Tadel, wlav, E. Tejedor, J. Blomer, A. Gheata, S. Hageboeck, S. Roiser, marsupial, S. Wunsch, O. Shadura, A. Bose, CristinaCristescu, X. Valls, and R. Isemann, *root-project/root: v6.18/02*, Aug., 2019.

[43] L. Gray, N. Smith, A. Novak, D. Taylor, P. Fackeldey, C. Carballo, P. Gessinger, J. Pata, A. Woodard, Andreas, B. Fischer, Z. Surma, A. Perloff, D. Noonan, L. Heinrich, N. Amin, P. Das, I. Dutta, J. Duarte, J. Rübenach, and A. R. Hall, *Coffeateam/coffea: Release v0.6.46*, Nov., 2020.

[44] J. Pivarski, P. Das, C. Burr, D. Smirnov, M. Feickert, T. Gal, L. Kreczko, N. Smith, N. Biederbeck, O. Shadura, M. Proffitt, benkrikler, H. Dembinski, H. Schreiner, J. Rembser, M. R., C. Gu, J. Rübenach, M. Peresano, and R. Turra, *scikit-hep/uproot: 3.12.0*, July, 2020.

[45] J. Pivarski, C. Escott, M. Hedges, N. Smith, C. Escott, J. Rembser, J. Nandi, B. Fischer, H. Schreiner, P. Das, P. Fackeldey, Nollde, and B. Krikler, *scikit-hep/awkward-array: 0.12.0rc1*, July, 2019.

[46] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, *TensorFlow: Large-scale machine learning on heterogeneous systems*, 2015. Software available from tensorflow.org.