The Machine Learning Landscape of Top Taggers

G. Kasieczka (ed)[1], T. Plehn (ed)[2], A. Butter[2], D. Debnath[3], M. Fairbairn[4], W. Fedorko[5],
C. Gay[5], L. Gouskos[6], P. T. Komiske[7], S. Leiss[1], A. Lister[5], S. Macaluso[3], E. M. Metodiev[7],
L. Moore[8], B. Nachman,[9,10] K. Nordström[11,12], J. Pearkes[5], H. Qu[6], Y. Rath[13],
M. Riegler[13], D. Shih[3], J. M. Thompson[2], and S. Varma[4]

1 Institut für Experimentalphysik, Universität Hamburg, Germany
2 Institut für Theoretische Physik, Universität Heidelberg, Germany
3 NHECT, Dept. of Physics and Astronomy, Rutgers, The State University of NJ, USA
4 Theoretical Particle Physics and Cosmology, King's College London, United Kingdom
5 Department of Physics and Astronomy, The University of British Columbia, Canada
6 Department of Physics, University of California, Santa Barbara, USA
7 Center for Theoretical Physics, MIT, Cambridge, USA
8 CP3, Université Catholique de Louvain, Louvain-la-Neuve, Belgium
9 Physics Division, Lawrence Berkeley National Laboratory, Berkeley, USA
10 Simons Inst. for the Theory of Computing, University of California, Berkeley, USA
11 National Institute for Subatomic Physics (NIKHEF), Amsterdam, Netherlands
12 LPTHE, CNRS & Sorbonne Université, Paris, France
13 III. Physics Institute A, RWTH Aachen University, Germany

gregor.kasieczka@uni-hamburg.de
plehn@uni-heidelberg.de

February 27, 2019

Abstract

**Based on the established task of identifying boosted, hadronically decaying top
quarks, we compare a wide range of modern machine learning approaches. We
find that they are extremely powerful and great fun.**

Content

1 Introduction

Top quarks are, from a theoretical perspective, particularly interesting because of their strong interaction with the Higgs boson and the corresponding structure of the renormalization group. Experimentally, they are unique in that they are the only quarks which decay before they hadronize. At the LHC many signal processes for the first time include phase space regimes where decaying tops are boosted enough to be analyzed through jet algorithms [1]. These subjet analyses have found their way into many LHC measurements and searches [2].

Top tagging based on an extension of standard jet algorithms has a long history [3]. Standard top taggers still used by ATLAS and CMS usually search for kinematic features induced by the top and W-boson masses [4–6]. This implies that top tagging is relatively straightforward, can be described in terms of perturbative quantum field theory, and hence makes an obvious candidate for a benchmark process. An alternative way to tag tops is based on the number of prongs, properly defined as N-subjettiness [7]. Combined with the SoftDrop mass variable [8], this defines a particularly economic tagger, but without a guarantee that the full top momentum gets reconstructed.

Based on simple deterministic taggers, the LHC collaborations have established that subjet analyses work and can be controlled in their systematic uncertainties [9]. The natural next step are advanced statistical methods [10], including multi-variate analyses [11]. In the same spirit, the second step has to be the question of why we apply highly complex tagging algorithms to a pre-processed set of kinematic observables rather than to actual data. This question becomes especially relevant when we consider the significant conceptual and performance progress in machine learning. Deep learning, or the use of neural networks with many hidden layers, is the tool which allows us to analyze low-level LHC data without constructing high-level observables. This directly leads us to standard classification tools in contemporary machine learning, for example in image or language recognition.

The goal of this study is to see how well different neutral network setups can classify jets based on calorimeter information. A straightforward way to apply standard machine learning tools to jets is with calorimeter images, so we use them for a comparison of the different available approaches on an equal footing. Considering calorimeter cells inside a fat jet as pixels defines a sparsely filled image which can be analyzed through standard convolutional networks [12–14]. A set of top taggers defined on the basis of image recognition will be part of our study and will be described in Sec. 3.1 [15,16]. A second set of taggers is based directly on the 4-momenta of the subjet constituents and will be introduced in Sec. 3.2 [17–19]; recurrent neural networks inspired by language recognition [20] can be grouped into this category. Finally, there are taggers which are motivated by theoretical considerations like soft and collinear radiation patterns or infrared safety [21–24] which we collect in Sec. 3.3.

While initially it was not clear if any of the machine learning methods applied to top tagging would be able to significantly exceed the performance of the multi-variate tools [15, 17], later studies have consistently showed that we can actually expect great performance improvement from cutting-edge tools. This turns around the question into which of the tagging approaches have the best performance (also relative to their training effort), and if the leading taggers make use of the same, hence complete set of information. Indeed, we will see that we can consider jet classification based on deep learning at the pure performance level an essentially solved problem. For a systematic experimental application of these tools our focus will be on a new set of questions related to training data, benchmarking, calibration,

systematics, etc.

2 Data set

The top signal and mixed quark-gluon background jets are produced with using Pythia8 [25] for a center-of-mass energy of 14 TeV and ignoring multiple interactions and pile-up. For a simplified detector simulation we use Delphes [26] with the default ATLAS detector card. The fat jet is defined through the anti-k_T algorithm [27] in FastJet [28] with $R = 0.8$. We only consider jets with

$$p_{T,j} = 550 \;.... \; 650 \text{ GeV} , \tag{1}$$

a matched parton-level top within $\Delta R = 0.8$, and all top decay partons within $\Delta R = 0.8$ of the jet axis as well. We also require $|\eta|_j < 2$. The constituents are extracted through the Delphes energy-flow algorithm, and the 4-momenta of the leading 200 constituents are stored. For jets with less than 200 constituents we simply add zero-vectors.

Particle information or additional tracking information is not included in this format. For instance, we do not record charge information or the expected displaced vertex from the b-decay. Therefore, the quoted performance should not be considered the last word for the LHC. On the other hand, limiting ourselves to essentially calorimeter information allows us to compare many different techniques and tools on an equal footing.

Our public data set consists of 1 million signal and 1 million background jets and can be obtained from the authors upon request [29]. They are divided into three samples: training with 600k signal and background jets each, validation with 200k signal and background jets each, and testing with 200k signal and 200k background jets. For proper comparison, all algorithms are optimized using the training and validation samples and all results reported are obtained using the test sample. For each algorithm, the classification result for each jet is made available, so we can not only measure the performance of the network, but also test which jets are correctly classified in each approach.
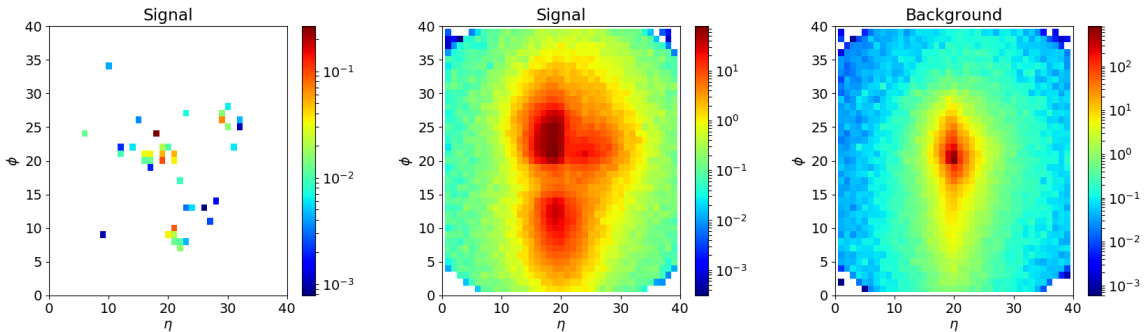

Figure 1: Left: jet image for the top signal after pre-processing. Center and right: signal and background images averaged over 10,000 individual images.

3 Taggers

3.1 Imaged-based taggers

To evaluate calorimeter information efficiently we can use powerful methods from image recognition. We simply interpret the energy deposition in the pixelled calorimeter over the area of the fat jet as an image and apply convolutional networks to it. These convolutional networks encode the 2-dimensional information which the network would have to learn if gave it the energy deposition as a 1-dimensional chain of entries. Such networks are the drivers behind many advances in image recognition outside physics and allow us to benefit from active research in the machine learning community.

If we approximate the calorimeter resolution as $0.04 \times 2.25°$ in rapidity vs azimuthal angle a fat jet with radius parameter $R = 0.8$ can be covered with 40×40 pixels. Assuming a p_T threshold around 1 GeV, a typical QCD jet will feature around 40 constituents in this jet image [22]. In Fig. 1 we show calorimeter images averaged over large numbers of top jets and QCD jets, after some pre-processing. For both, signal and background jets the center of the image is defined by the hardest object. Next, we rotate the second-hardest object to 12 o'clock. Combined with a narrow p_T-bin for the jets this second jet develops a preferred distance from the center for the signal but not for the QCD background. Finally, we reflect the third-largest object to the right side of the image, where such a structure is really only visible for the 3-prong top signal. We will compare two cutting-edge networks analyzing calorimeter images. A comparison between the established multi-variate taggers and the modestly performing early DeepTop network can be found in Ref. [15].

3.1.1 CNN

This method applies a convolutional neural network (CNN) trained on jet images, generated from the list of per jet constituents of the reference sample, see Ref. [16] for more details. We perform all preprocessing before pixellating the image. First, the jet is centered and rotated according to its p_T-weighted centroid and principal axis. Then it is flipped horizontally and vertically so that the maximum intensity is in the upper right quadrant. Finally, the image is pixellated with p_T as the intensity, and normalized to unit total intensity. Although the original method includes color images, where the track p_T's and neutral p_T's are considered separately, for this dataset we restrict ourselves to gray-scale images.

The starting point for the CNN design is the architecture of the DeepTop tagger of Ref. [15], but augmented with more feature maps, hidden units, etc. See Fig. 2 for a full description.
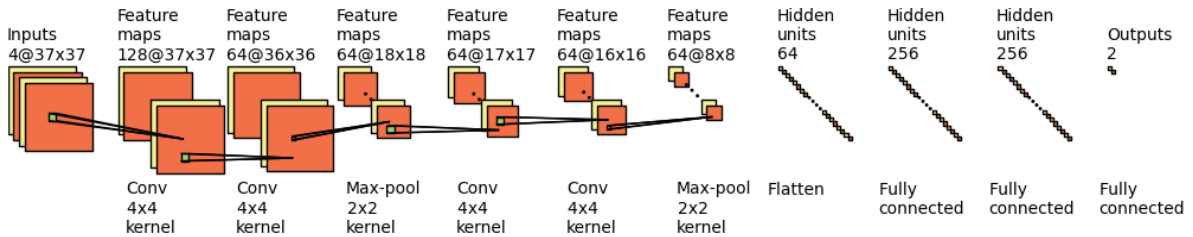

Figure 2: Architecture of the CNN top tagger. Figure from Ref. [16].

For training a cross entropy loss function and AdaDelta as the optimizer with a minibatch size of 128 is used. The weights are initialized with the Keras [30] default settings. We obtain further improvements with a learning rate of 0.3 and a learning rate schedule. These settings are taken directly from Ref. [16] and not re-optimized for this jet sample. The CNN is implemented on an NVidia Tesla P100 GPU using Keras with the TensorFlow [31] back-end.

3.1.2 ResNeXt

The ResNeXt-50 model is a deep 2D CNN using jet images as inputs. The images used in this model are 64×64 pixels in size centered on the jet axis, corresponding to a granularity of 0.025 radians in the $\eta-\phi$ space. The intensity of each pixel is the sum of p_T of all the particles within the pixel. The CNN architecture is based on the 50-layer ResNeXt architecture [32] proposed for image recognition. To adapt to the smaller size of the jet images, the number of channels in all the convolutional layers except for the first one is reduced by a factor of 4, and a dropout layer with a keep probability of 0.5 is added after the global pooling. The network in implemented in Apache MXNet [33], and trained from scratch on the top tagging dataset.

3.2 4-Vector-based taggers

A problem with the image recognition approaches discussed in Sec. 3.1 arises when we want to include additional information for example from tracking or particle identification. We can always combine different images in one analysis [34], but the significantly different resolution for example of calorimeter and tracker images becomes a serious challenge. For a way out we can follow the experimental approach developed by CMS and ATLAS and use particle flow or similar objects as input to neural network taggers. The challenge is to define an efficient network setup that either knows or learns the symmetry properties of 4-vectors and replace the notion of 2-dimensional geometric structure in the convolutional networks.
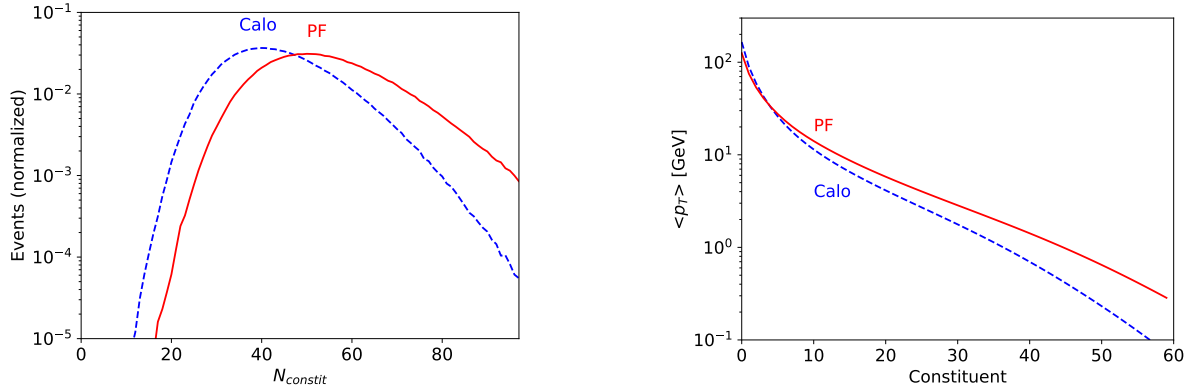

Figure 3: Number of constituents (left) and mean of the transverse momentum (right) of the ranked constituents of a signal top jet. We show calorimeter entries as well as particle flow constituents after Delphes.

3.2.1 TopoDNN

Given the 200 p_T-sorted 4-vectors per jet, arguably the simplest deep neural network architecture is a dense network taking all 800 floating point numbers as a fixed set [18]. Since the mass of individual particle-flow candidates cannot be reliably reconstructed, the TopoDNN tagger uses at most 600 inputs: (p_T, η, ϕ) for each jet constituent. In order to improve the training through physically-motivated pre-processing, a longitudinal boost and a rotation in the transverse plane are applied so that the constituent η and ϕ values of the highest p_T constituent is centered at $(0,0)$. The momenta are scaled by a constant $1/1700$, because a dynamic quantity such as the jet p_T can distort the jet mass [13]. A further rotation is applied so that the second highest jet constituent is aligned with the negative y-axis, to remove the rotational symmetry of the second prong about the first prong in the jet. This is a proper rotation (and not a simple rotation in the η-ϕ plane) and thus preserves the jet mass (but distorts quantities like N-subjettiness [7, 35]).

The network hyper-parameters are the number of constituents considered as inputs, the number of hidden layers, the number of nodes per layer, and the activation functions for each layer. For this simulated dataset, it was found that 30 constituents saturated the network performance. The TopoDNN tagger presented here has a similar architecture as a tagger with the same name used by the ATLAS collaboration [36]. The architecture of the TopoDNN tagger was optimized for a dataset of high p_T (450 to 2400 GeV) R=1.0 trimmed jets. The ATLAS TopoDNN architecture was not altered for this study, other than changing the number of inputs and retraining the network. The ATLAS tagger uses only 10 jet constituents, but the inputs are topoclusters [37] and not individual particles. Individual particle-flow candidates in reality are also not individual particles, but there is a closer correspondence. For this reason, the TopoDNN tagger performance presented here is not directly comparable to the results presented in Ref. [36].

3.2.2 Multi-Body N-Subjettiness

The multi-body phase space tagger is based on the proposal in Ref. [38] to use a basis of N-subjettiness variables [7] spanning an m-body phase space to teach a dense neural network to separate background and signal using a minimal set of input variables. A setup for the specific purpose of top tagging was introduced in Ref. [24]. To generate the input for our network we first convert the event to HEPMC and use Rivet [39] to evaluate n-subjettiness variables by using the FastJet [28] contrib plug-in library [7, 40]. The input variables spanning the m-body phase space are then given by:

$$\left\{ \tau_1^{(0.5)}, \tau_1^{(1)}, \tau_1^{(2)}, \tau_2^{(0.5)}, \tau_2^{(1)}, \tau_2^{(2)}, \ldots, \tau_{m-2}^{(0.5)}, \tau_{m-2}^{(1)}, \tau_{m-2}^{(2)}, \tau_{m-1}^{(1)}, \tau_{m-1}^{(2)} \right\}, \tag{2}$$

where

$$\tau_N^{(\beta)} = \frac{1}{p_{T,J}} \sum_{i \in J} p_{T,i} \min \left\{ R_{1i}^{\beta}, R_{2i}^{\beta}, \ldots, R_{Ni}^{\beta} \right\}, \tag{3}$$

and R_{ni} is the distance in the $\eta - \phi$ plane of the jet constituent i to the axis n. We choose the N axes using the k_T algorithm [41] with E-scheme recombination. This input has the advantage that it is theoretically sound and IR-safe, while it can be easily understood as a set of correlators between 4-momenta combining the number of prongs in the jet with additional momentum information.

The machine learning setup is a dense neural network implemented in TensorFlow [31] using these input variables. We also add the jet mass and jet p_T as input variables to allow the network to learn physical scales. The network consists of four fully connected hidden layers, the first two with 200 nodes and a dropout regularization of 0.2, and the last two with 50 nodes and a dropout regularization of 0.1. The output layer consists of two nodes. We use a ReLu activation function throughout and minimize the cross-entropy using Adam optimization [42].

3.2.3   Recurrent Networks

In this method, a recursive neural network (RecNN) is trained on jet trees. The RecNN provides a "jet embedding", which maps a set of 4-momenta into a vector of fixed size and can be trained together with a successive network used for classification or regression [20]. Jet constituents of the reference sample are reclustered to form binary trees, and the topology is determined by the clustering algorithm (e.g. k_t, anti-k_t or Cambridge/Aachen). For this paper, we chose the k_t clustering algorithm, and 7 features for the nodes: $|p|$, η, ϕ, E, E/E_{jet}, p_{T} and θ. We removed the median and scaled each feature according to the range between the first and the third quartiles (this scaling is robust to outliers).

To make training fast enough, a special batching was implemented in Ref. [20]. Jets are reorganized by levels (e.g. the root node of each tree in the batch is added at level zero, their children at level one, etc). Each level is restructured so that all inner nodes come first, followed by outer nodes (leaves), and zero padding is applied when necessary.

Results are obtained from a PyTorch implementation of the simple RecNN introduced in Ref. [20] to do batch training on an NVidia Tesla P100 GPU. Training is performed over 40 epochs with a minibatch of 128 and a learning rate of 5×10^{-3} (decayed by a factor of 0.9 after every epoch), using the cross entropy loss function and Adam [42] as the optimizer. Testing is done with the set of weights that give the best accuracy on the validation set. Most settings (e.g. hyperparameter values, topology, training methods, features, etc) were taken directly from [20] and not re-optimized for this jet sample.

3.2.4   P-CNN

The particle-level convolutional neural network (P-CNN), used in the CMS particle-based DNN tagger [43], is a customized 1D CNN for boosted jet tagging. Each input jet is represented as a sequence of particles with a fixed length of 100, organized in descending order of p_T. The sequence is padded with zeros if a jet has less than 100 particles. Occasionally, if a jet contains more than 100 particles, the extra particles are simply discarded. For each particle, seven input features, computed from the four-momenta of the particle, are used as inputs to the network: $\log p_T^i$, $\log E^i$, $\log(p_T^i/p_T^{\text{jet}})$, $\log(E^i/E^{\text{jet}})$, $\Delta\eta^i$, $\Delta\phi^i$ and ΔR^i. Angular distances are measured with respect to the jet axis. The use of these transformed features instead of the raw four-momenta was found to lead to slightly improved performance.

The P-CNN used in this paper follows the same architecture as the CMS particle-based DNN tagger. However, the top tagging dataset in this paper contains only kinematic information of the particles, while the CMS particle-based DNN tagger uses particle tracks and secondary vertices in addition, therefore the network components related to them are removed here. The P-CNN is similar to the ResNet model [44] for image recognition, but uses only 1D convolution instead of 2D convolution. A total of 14 convolutional layers are used in the

P-CNN, all with a kernel size of 3. The number of channels for the 1D convolutional layers ranges from 32 to 128. The outputs of the convolutional layers undergo a global pooling, followed by a fully-connected layer of 512 units and a dropout layer with a keep rate of 0.5, before yielding the final prediction. The network is implemented in Apache MXNet [33] and trained with the Adam optimizer [42].

3.2.5   ParticleNet

Similar to the Particle Flow Network, the ParticleNet [45] is also built upon the point cloud representation of jets where each jet is treated as an unordered set of particles. The input features for each particle are the same as that in the P-CNN model. In ParticleNet, a graph is first constructed for each jet, with the constituent particles as the vertices of the graph. The edges of the graph are then initialized by connecting each particle to its k nearest neighboring particles based on the distance in the $\eta - \phi$ space. The EdgeConv operation proposed in [46] is applied on the graph to transform and aggregate information from the nearby particles at each vertex, analogous to how regular convolution operates on square patches of images. The weights of the EdgeConv operator are shared among all particles in the graph, therefore preserving the permutation invariance property of the particles in a jet. The EdgeConv operations can be stacked to form a deep graph convolutional network.

The ParticleNet is built upon the dynamic graph convolution approach in [46] and further extends it. The ParticleNet consists of three stages of EdgeConv operations, with three EdgeConv layers and a shortcut connection [47] in each stage. Between the stages, the jet graph is dynamically updated by redefining the edges based on the distances in the new feature space generated by the EdgeConv operations. The number of nearest neighbors, k, is also varied in each stage. The three stages of EdgeConv are followed by a global pooling over all particles, and then two fully connected layers. The details of the ParticleNet architecture can be found in [45]. It is implemented in Apache MXNet [33] and trained with the Adam optimizer [42].

3.3   Theory-inspired taggers

Going beyond a relatively straightforward analysis of 4-vectors we can build networks specifically for subjet analyses and include as much of our physics knowledge as possible. The motivation for this is two-fold: building this information into the network should save training time, and it should allow us to test what kind of physics information the network relies on.

At the level of these 4-vectors the main difference between top jets and massless QCD jets is a mass drop [2] which appears after we combine 4-vectors based on soft and collinear proximity. In addition, any jet analysis tool should give stable results in the presence of additional soft of collinear splittings. From theory we know that smooth limits from very soft or collinear splittings to no splitting have to exist, a property usually referred to infrared safety. If we go beyond calorimeter images with their coarse resolution it is not clear how IR-safe top taggers are [48]. This is not only a theoretical problem which arises when we for example want to compare rate measurements with QCD predictions, a lack of IR-safety will also make it hard to train or benchmark on Monte Carlo simulations, and to extraction tagging efficiencies [49].

3.3.1 Lorentz Boost Network

The Lorentz Boost Network (LBN) is designed to autonomously extract a comprehensive set of physics-motivated features given only low-level variables in the form of particle four-vectors [19]. These engineered features can be utilized in a subsequent neural network to solve a specific physics task. The resulting two-stage architecture is trained jointly so that extracted feature characteristics are adjusted during training to serve the minimization of the objective function by means of back-propagation.

The general approach of the LBN is to reconstruct parent particles from four-vectors of their decay products and to exploit their properties in appropriate rest frames. Its architecture is comprised of three layers. First, the N provided input vectors are composed to $2M$ intermediate vectors through linear combinations. Corresponding linear coefficients are trainable and constrained to positive numbers to prevent constructing vectors with unphysical implications, such as $E < 0$ or $E < m$. In the subsequent layer, M of these intermediate vectors are treated as particles, whereas the other M vectors are considered rest frames. Via Lorentz transformation, the particles are boosted into the rest frames in a pairwise approach, i.e., the mth particle is boosted into the mth rest frame. In the last layer, F features are extracted from the obtained M boosted particles by employing a set of generic feature mappings. The autonomy of the LBN lies in its freedom to construct arbitrary boosted particles through trainable particle and rest frame combinations, and to consequently access and provide underlying characteristics which are otherwise distorted by relativistic kinematics.

For this study, best training results are obtained for $M = 50$ and six generic feature mappings: E, m, p_T, ϕ, and η of all boosted particles, and the cosine of the angle between momentum vectors of all pairs of boosted particles. Thus, $F = 5M + (M^2 - M)/2$ features are extracted in total. Batch normalization with floating averages during training is employed after the feature extraction layer [50]. This subsequent NN incorporates a common, fully-connected architecture with four hidden layers, involving 1024, 512, 256, and 128 exponential linear (ELU) units, respectively. Generalization and overtraining suppression are enforced via L2 regularization with a factor of 10^{-4}. The Adam optimizer is utilized for minimizing the binary cross-entropy loss [42], configured with an initial learning rate of 10^{-3} for a batch size of 1024. The training procedure converges after around 5000 batch iterations.

The order of input vectors is adjusted on a per-jet basis before being fed to the LBN. The method utilizes linearized clustering histories as preferred by the anti-k_T algorithm [27], implemented in the Fastjet package [28]. First, the jet constituents are reclustered with $\Delta R = 0.2$, and the resulting subjets are ordered by p_T. Per subjet, another clustering with $\Delta R = 0.2$ is performed and the order of constituents is inferred from the anti-k_T clustering history. In combination, this approach yields a consistent order of the initial jet constituents.

3.3.2 Lorentz Layer

Switching from image recognition to a setup based on 4-momenta we can take inspiration from graph convolutional networks, where sparsely filled images are analyzed in terms of objects and a free distance measure. While often the most appropriate distance measure needs to be determined from data, fundamental physics assumptions tell us that the relevant distance for jet physics is the scalar product of two 4-vectors. This scalar product will be especially effective in searching for heavy masses in a jet clustering history when we evaluate it for combinations of final-state objects.

The input to the Lorentz Layer (LoLa) network [22] are sets of 4-vectors $k_{\mu,i}$ with $i = 1, ..., N$. As a first step we apply a combination layer to define linear combinations of the 4-vectors,

$$k_{\mu,i} \xrightarrow{\text{CoLa}} \widetilde{k}_{\mu,j} = k_{\mu,i} \, C_{ij} \quad \text{with} \quad C = \begin{pmatrix} 1 & \cdots & 0 & C_{1,N+1} & \cdots & C_{1,M} \\ \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & \cdots & 1 & C_{N,N+1} & \cdots & C_{N,M} \end{pmatrix} . \quad (4)$$

Here we use $N = 60$ and $M = 90$. In a second step we transform all 4-vectors into measurement-motivated objects,

$$\tilde{k}_j \xrightarrow{\text{LoLa}} \hat{k}_j = \begin{pmatrix} m^2(\tilde{k}_j) \\ p_T(\tilde{k}_j) \\ w_{jm}^{(E)} \, E(\tilde{k}_m) \\ w_{jm}^{(p_T)} \, p_T(\tilde{k}_m) \\ w_{jm}^{(m^2)} \, m^2(\tilde{k}_m) \\ w_{jm}^{(d)} \, d_{jm}^2 \end{pmatrix} , \quad (5)$$

where d_{jm}^2 is the Minkowski distance between two four-momenta $\tilde{k}_j$ and $\tilde{k}_m$, and we either sum or minimize over the internal indices. One copy of the sum and five copies of the minimum term are used. Just for amusement we have checked that the network also learns the Minkowski metric from top vs QCD jet data. The LoLa network setup is then straightforward, three fully connected hidden layers with 100, 50, and 2 nodes.

3.3.3 Energy Flow Polynomials

Energy Flow Polynomials (EFPs) [21] are a collection of observables designed to form a linear basis of all infrared- and collinear- (IRC) safe observables, building upon a rich literature of energy correlators [51–53]. The EFPs naturally enable linear methods to be applied to collider physics problems, where the simplicity and convexity of linear models is highly desirable.

EFPs are energy correlators whose angular structures are in correspondence with non-isomorphic multigraphs. Specifically, they are defined using the transverse momentum fractions, $z_i = p_{T,i}/\sum_j p_{T,j}$, of the M particles as well as their pairwise rapidity-azimuth distances, $\theta_{ij} = ((y_i - y_j)^2 + (\phi_i - \phi_j)^2)^{\beta/2}$, without any additional preprocessing:

$$\text{EFP}_G = \sum_{i_1=1}^{M} \cdots \sum_{i_N=1}^{M} z_{i_1} \cdots z_{i_N} \prod_{(k,\ell) \in G} \theta_{i_k i_\ell}, \quad (6)$$

where G is a given multigraph, N is the number of vertices in G, and (k, ℓ) is an edge connecting vertices k and ℓ in G. The EFP-multigraph correspondence yields simple visual rules for translating multigraphs to observables: vertices contribute an energy factor and edges contribute an angular factor. Beyond this, the multigraphs provide a natural organization of the EFP basis when truncating in the number of edges (or angular factors) d, with exactly 1000 EFPs with $d \leq 7$.

For the top tagging EFP model, all $d \leq 7$ EFPs computable in $\mathcal{O}(M^3)$ or faster were used (995 observables total) with angular exponent $\beta = 1/2$. Linear classification was performed with Fisher's Linear Discriminant [54] using `scikit-learn` [55]. Implementations of the EFPs are available in the EnergyFlow package [56].

3.3.4 Energy Flow Networks

An Energy Flow Network (EFN) [23] is an architecture built around a general decomposition of IRC-safe observables that manifestly respects the variable-length and permutation-invariance symmetries of observables. Encoding the proper symmetries of particle collisions in an architecture results in an extremely natural way of processing the collider data.

Any IRC-safe observable can be approximated arbitrarily well as:

$$\text{EFN} = F\left(\sum_{i=1}^{M} z_i \Phi(y_i, \phi_i)\right),\tag{7}$$

where $\Phi : \mathbb{R}^2 \to \mathbb{R}^\ell$ is a per-particle mapping that embeds the locations of the particles in a latent space of dimension ℓ. Particles are summed over in this latent space to obtain an event representation, which is then mapped by F to the target space of interest.

The EFN architecture parametrizes the functions Φ and F with neural networks, with specific implementation details of the top tagging EFN architecture given in the Particle Flow Network section. For both the top tagging EFNs and PFNs, input particles are translated to the origin of the rapidity-azimuth plane according to the p_T-weighted centroid, rotated in this plane to consistently align the principal axis of the energy flow, and reflected to locate the highest p_T quadrant in a consistent location. A fascinating aspect of the decomposition in Eq. 7 is that the learned latent space can be examined both quantitatively and visually, as the rapidity-azimuth plane is two dimensional and can be viewed as an image. Figure 4 shows a visualization of the top tagging EFN, which learns a dynamic pixelization of the space.
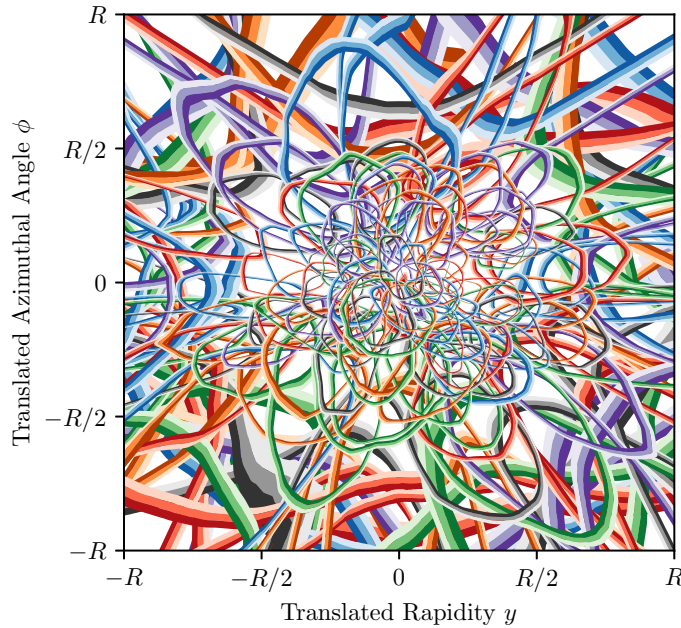

Figure 4: Visualization of the trained top tagging EFN. Figure from Ref. [23].

3.3.5 Particle Flow Networks

Particle Flow Networks (PFNs) [23] generalize the EFNs beyond IRC safety. In doing so, they make direct contact with machine learning literature on learning from point clouds, in particular the Deep Sets framework [57]. This identification of point clouds as the machine learning data structure with intrinsic properties most similar to collider data provides a new avenue of exploration.

In the collider physics language, the key idea is that any observable can be approximated arbitrarily well as:

$$\text{PFN} = F\left(\sum_{i=1}^{M} \Phi(p_i)\right),\tag{8}$$

where p_i contains per-particle information, such as momentum, charge, or particle type. Similar to the EFN case, Φ maps from the particle feature space into a latent space and F maps from the latent space to the target space. The per-particle features provided to the network can be varied to study their information content. Only particle four-momentum information was used here, exploring the importance of particle-type information for top tagging is an interesting direction for future studies.

The PFN architecture parameterizes the functions Φ and F with neural networks. For the top tagging EFN and PFN models, Φ and F were parameterized with simple three-layer neural networks of $(100, 100, 256)$ and $(100, 100, 100)$ nodes in each layer, respectively, corresponding to a latent space dimension of $\ell = 256$. A ReLU activation with He-uniform parameter initialization [58], and a two-node classifier output is used with SoftMax activation, trained with Keras [30] and TensorFlow [31]. Implementations of the EFNs and PFNs are available in the `EnergyFlow` package [56].

4 Comparison

To assess the performance of different algorithms we first look at the individual ROC curves over the full range of top jet signal efficiencies. They are shown in Fig. 5, compared to a simple tagger based on N-subjettiness [7] and jet mass. We see how, with few exceptions, the different taggers define similar shapes in the signal efficiency vs background rejection plane.

Given that observation we can instead analyze three single-number performance metrics for classification tasks. First, we compute the *area under the ROC curve* shown in Fig. 5. It is bounded to be between 0 and 1, and stronger classification corresponds to values larger than 0.5 at a chosen working point. Next, the *accuracy* is defined as the fraction of correctly classified jets. Finally, for a typical analysis application the *rejection power* at a realistic working point is most relevant. We choose the background rejection at a signal efficiency of 30%.

All three figures of merit for each of the algorithms are shown in Tab. 1. Most approaches achieve an AUC of approximately 0.98 with the strongest performance from the image-based ResNeXt50 tagger and the 4-vector-based ParticleNet tagger, followed by the theory-inspired Particle Flow Network. These approaches also reach the highest accuracy and background rejection at fixed signal efficiency. A typical accuracy is 93%, and the quoted differences between the taggers are unlikely to define a clear experimental preference. The most intuitive performance measure is the background rejection at given signal efficiency, even though
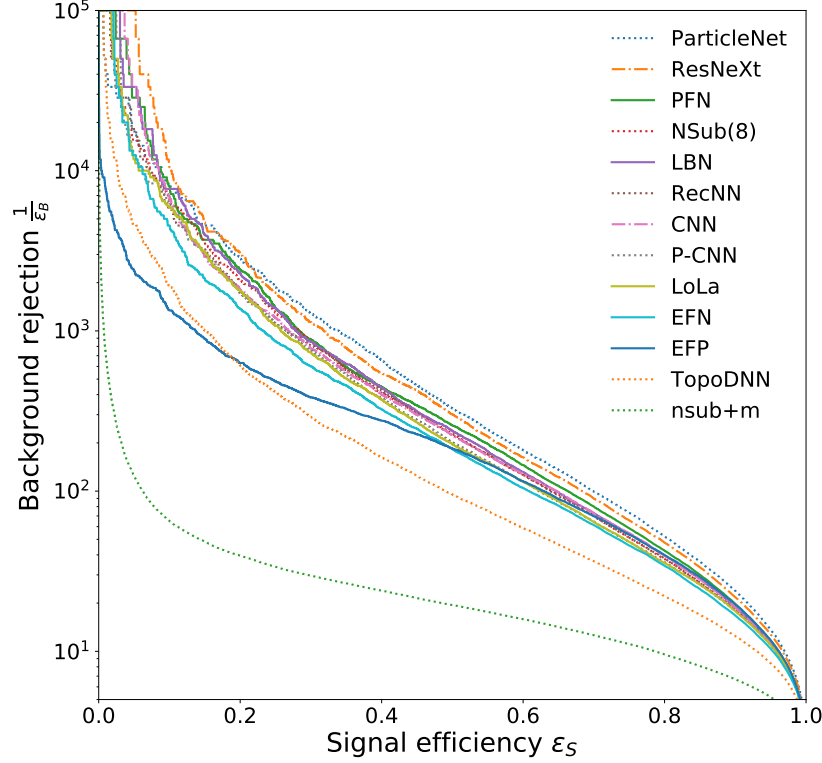
Figure 5: ROC curves for all algorithms evaluated on the same test sample, shown as the median of multiple trainings.

similar information should be included in the ROC curve. For the background rejection we see a sizeable variation from around 1/600 to better than 1/1000. Again, the cutting edge ResNeXt50 and ParticleNet approaches lead to the best results, corresponding to an improvement of the signal-to-background ratio by a factor $\epsilon_S/\epsilon_B > 300$, and vastly exceeding the current top tagging performance in ATLAS and CMS.

On the other hand, in Fig. 5 and Tab. 1 we see that some of the physics-motivated setups remain competitive with the technically much more advanced ResNeXt50 and ParticleNet networks. This suggests that even for a straightforward task like top tagging in fat jets we can develop competitive and efficient physics-specific tools. While their performance does not quite match the state of the art standard networks, it is close enough to test both approaches on key requirements in particle physics, like treatment of uncertainties, stability with respect to detector effects, etc.

The obvious question in any deep-learning analysis is if the tagger captures all relevant information. As a starting point we can test how correlated the classifier output of the different taggers are. We show the pair-wise correlations for a subset of classifier outputs in Fig. 6, with the correlation matrix given in Tab. 2. As expected from strong classifier performances, most jets are clustered in the bottom left and top right corners, corresponding to identification as background and signal, respectively. The largest spread is observed for correlations with the EFP. The two strongest individual classifier outputs — ResNeXt50 and ParticleNet — are not perfectly correlated.

Given that we find the outputs of the different algorithms not to be fully correlated, we can

	AUC	Accuracy	$1/\epsilon_B$ ($\epsilon_S = 0.3$)	#Parameters
CNN [16]	0.981	0.930	780	610k
ResNeXt [32]	0.984	0.936	1140	1.46M
TopoDNN [18]	0.972	0.916	290	59k
Multi-body N-subjettiness 6 [24]	0.979	0.922	856	57k
Multi-body N-subjettiness 8 [24]	0.981	0.929	860	58k
RecNN	0.981	0.929	810	13k
P-CNN	0.980	0.930	760	348k
ParticleNet [45]	0.985	0.938	1280	498k
LBN [19]	0.981	0.931	860	705k
LoLa [22]	0.980	0.929	730	127k
Energy Flow Polynomials [21]	0.980	0.932	380	1k
Energy Flow Network [23]	0.979	0.927	600	82k
Particle Flow Network [23]	0.982	0.932	880	82k
GoaT (see text)	0.985	0.939	1440	25k

Table 1: Single-number performance metrics for all algorithms evaluated on the test sample. We quote the area under the ROC curve (AUC), the accuracy, and the background rejection at a signal efficiency of 30%. The number of trainable parameters of the model is given as well. Performance metrics for the GoaT meta-tagger are based on a subset of events.

investigate whether their combination into a meta-tagger might improve performance. Note that this GoaT (Greatest of all Taggers) meta-tagger should not be viewed as a potential analysis tool, but rather as a benchmark of how much unused information is available in correlations that could be captured by a future approach. It is implemented as a fully connected network with 5 layers containing 100-100-100-20-2 nodes. All activation functions are ReLu, apart from the final layer where we use SoftMax. Training is performed with the Adam [42] optimizer with an initial learning rate of 0.001 and binary cross-entropy loss. We train for up to 50 epochs, but terminate if there is no improvement in the validation loss for two consecutive epochs, so a typical training ends after 5 epochs. The training data is provided by individual tagger output on the previous test sample and split intro three subsets: GoaT-training (160k events), GoaT-testing (160k events) and GoaT-validation (80k events). Training/testing is repeated ten times and for each repetition the events are randomly re-shuffled between the three different subsets. In Tab. 1 we see that this combination of algorithms improves the best individual tagger by more than 10% in the background rejection. We consider this number a realistic estimate of the kind of improvement we can still expect for deep-learning top quark identification.

In spite of the fact that our study gives some definite answers concerning deep learning for simple jet classification at the LHC, a few questions remain open: first, we use jets in a relatively narrow and specific p_T-slice. Future efforts could explore softer jets, where the decay products are not necessarily inside one fat jet; higher p_T, where detector resolution effects become crucial; and wider p_T windows, where stability of taggers becomes relevant. The samples also use a simple detector simulation and do not contain effects from underlying event and pile-up.
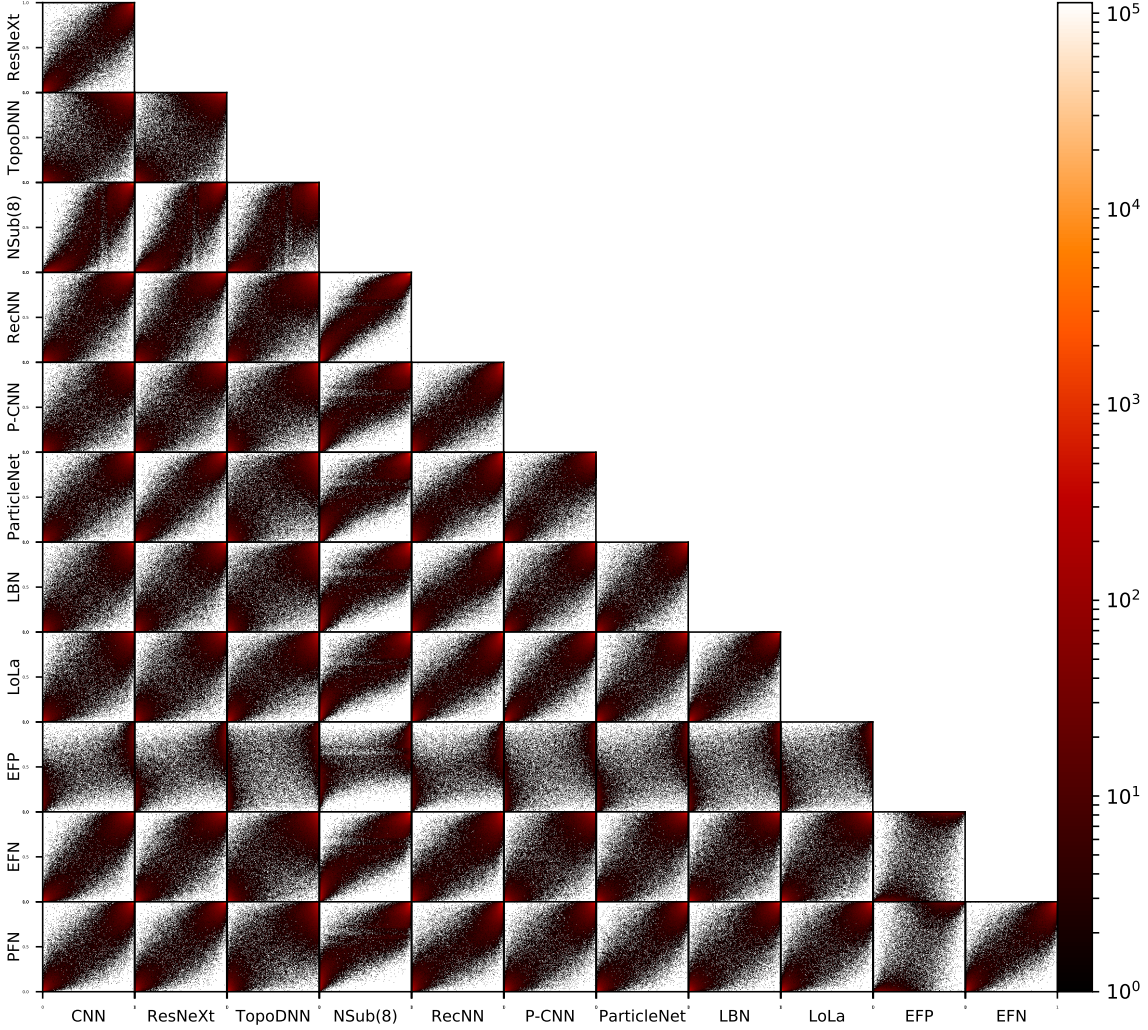
Figure 6: Pairwise distributions of classifier outputs, each in the range 0 ... 1 from pure QCD to pure top.

Second, our analysis essentially only includes calorimeter information as input. Additional information exists in the distribution of tracks of charged particles and especially the displaced secondary vertices from decays of B-hadrons. How easily this information can be included might well depend on details of the network architecture.

Third, when training machine learning classifiers on simulated events and evaluating them on data, there exists a number of systematic uncertainties that need to be considered. Typical examples are jet calibration, MC generator modeling, and IR-safety when using theory predictions. Understanding these issues will be a crucial next step. Possibilities are to include uncertainties in the training or to train on data in a weakly supervised or unsupervised fashion [59].

Finally, we have neglected questions such as computational complexity, evaluation time and memory footprint. These will be important considerations, especially once we want to include deep networks in the trigger. A related questions will be how many events we need

| | CNN | | TopoDNN | | RecNN | | ParticleNet | | LoLa | | EFN | |
		ResNeXt		NSub(8)		P-CNN		LBN		EFP		PFN
CNN	1	.983	.953	.977	.974	.972	.979	.973	.973	.972	.981	.983
ResNeXt	.983	1	.953	.977	.975	.975	.986	.976	.974	.975	.977	.984
TopoDNN	.953	.953	1	.963	.961	.962	.955	.959	.970	.946	.954	.961
NSub(8)	.977	.977	.963	1	.983	.975	.977	.976	.977	.964	.978	.977
RecNN	.974	.975	.961	.983	1	.977	.977	.979	.979	.962	.973	.978
P-CNN	.972	.975	.962	.975	.977	1	.979	.979	.984	.964	.968	.979
ParticleNet	.979	.986	.955	.977	.977	.979	1	.979	.978	.971	.973	.982
LBN	.973	.976	.959	.976	.979	.979	.979	1	.982	.966	.971	.979
LoLa	.973	.974	.970	.977	.979	.984	.978	.982	1	.967	.970	.980
EFP	.972	.975	.946	.964	.962	.964	.971	.966	.967	1	.969	.978
EFN	.981	.977	.954	.978	.973	.968	.973	.971	.970	.969	1	.982
PFN	.983	.984	.961	.977	.978	.979	.982	.979	.980	.978	.982	1

Table 2: Correlation coefficients from the combined GoaT analyses

to saturate the performance of a given algorithm.

5 Conclusion

Because it is experimentally and theoretically well defined, top tagging is a prime benchmark candidate to determine what we can expect from modern machine learning methods in classification tasks relevant for the LHC. We have shown how different neural network architectures and different representations of the data distinguish hadronically decaying top quarks from a background of light quark or gluon jets.

We have compared three different classes of deep learning taggers: image-based networks, 4-vector-based networks, and taggers relying on additional theory considerations. We find that each of these approaches provide competitive taggers with comparable performance, making it clear that there is no golden deep network architecture. Simple tagging performance will not allow us to identify the kind of network architectures we want to use for jet classification tasks at the LHC. Instead, we need to investigate open questions like versatility and stability in the specific LHC environment with the specific requirements of the particle physics community for example related to calibration and uncertainty estimates.

This result is a crucial first step in establishing deep learning at the LHC. Clearly, jet classification using deep networks working on low-level observables is the logical next step in subjet analysis and ready to be applied by ATLAS and CMS. On the other hand, there exist a range of open questions related to training, calibration, and uncertainties. They are specific to LHC physics and might well require particle physics to move beyond applying standard tagging approaches and in return contribute to the field of deep learning. Given that we have now understood that different network setups can achieve very similar performance for mostly calorimeter information, it is time to tackle these open questions.

Acknowledgments

References

[1] M. H. Seymour, Z. Phys. C **62**, 127 (1994); doi:10.1007/BF01559532

[2] J. M. Butterworth, A. R. Davison, M. Rubin and G. P. Salam, Phys. Rev. Lett. **100**, 242001 (2008) doi:10.1103/PhysRevLett.100.242001 [arXiv:0802.2470 [hep-ph]].

[3] W. Skiba and D. Tucker-Smith, Phys. Rev. D **75**, 115010 (2007) doi:10.1103/PhysRevD.75.115010 [arXiv:hep-ph/0701247]; B. Holdom, JHEP **0703**, 063 (2007) doi:10.1103/PhysRevD.75.115010 [arXiv:hep-ph/0702037]; M. Gerbush, T. J. Khoo, D. J. Phalen, A. Pierce and D. Tucker-Smith, Phys. Rev. D **77**, 095003 (2008) doi:10.1103/PhysRevD.77.095003 [arXiv:0710.3133 [hep-ph]]. L. G. Almeida, S. J. Lee, G. Perez, G. F. Sterman, I. Sung and J. Virzi, Phys. Rev. D **79**, 074017 (2009) doi:10.1103/PhysRevD.79.074017 [arXiv:0807.0234 [hep-ph]]; L. G. Almeida, S. J. Lee, G. Perez, I. Sung and J. Virzi, Phys. Rev. D **79**, 074012 (2009) doi:10.1103/PhysRevD.79.074012 [arXiv:0810.0934 [hep-ph]]; T. Plehn and M. Spannowsky, J. Phys. G **39**, 083001 (2012) doi:10.1088/0954-3899/39/8/083001 [arXiv:1112.4441 [hep-ph]].

[4] D. E. Kaplan, K. Rehermann, M. D. Schwartz and B. Tweedie, doi:10.1103/PhysRevLett.101.142001 Phys. Rev. Lett. **101**, 142001 (2008) [arXiv:0806.0848 [hep-ph]].

[5] T. Plehn, G. P. Salam and M. Spannowsky, doi:10.1103/PhysRevLett.104.111801 Phys. Rev. Lett. **104**, 111801 (2010) [arXiv:0910.5472 [hep-ph]].

[6] T. Plehn, M. Spannowsky, M. Takeuchi, and D. Zerwas, JHEP **1010**, 078 (2010) doi:10.1007/JHEP10(2010)078 [arXiv:1006.2833 [hep-ph]].

[7] J. Thaler and K. Van Tilburg, JHEP **1103**, 015 (2011) doi:10.1007/JHEP03(2011)015 [arXiv:1011.2268 [hep-ph]].

[8] A. J. Larkoski, S. Marzani, G. Soyez and J. Thaler, JHEP **1405**, 146 (2014) doi:10.1007/JHEP05(2014)146 [arXiv:1402.2657 [hep-ph]].

[9] A. Abdesselam *et al.*, Eur. Phys. J. C **71**, 1661 (2011) doi:10.1140/epjc/s10052-011-1661-y [arXiv:1012.5412 [hep-ph]]; A. Altheimer *et al.*, J. Phys. G **39**, 063001 (2012) doi:10.1088/0954-3899/39/6/063001 [arXiv:1201.0008 [hep-ph]]; A. Altheimer *et al.*, Eur. Phys. J. C **74**, no. 3, 2792 (2014) doi:10.1140/epjc/s10052-014-2792-8 [arXiv:1311.2708 [hep-ex]]; D. Adams *et al.*, Eur. Phys. J. C **75**, no. 9, 409 (2015) doi:10.1140/epjc/s10052-015-3587-2 [arXiv:1504.00679 [hep-ph]].

[10] D. E. Soper and M. Spannowsky, Phys. Rev. D **87**, 054012 (2013) doi:10.1103/PhysRevD.87.054012 [arXiv:1211.3140 [hep-ph]]; D. E. Soper and M. Spannowsky, Phys. Rev. D **89**, no. 9, 094005 (2014) doi:10.1103/PhysRevD.89.094005 [arXiv:1402.1189 [hep-ph]];

[11] G. Kasieczka, T. Plehn, T. Schell, T. Strebler and G. P. Salam, JHEP **1506**, 203 (2015)

doi:10.1007/JHEP06(2015)203 [arXiv:1503.05921 [hep-ph]].

[12] J. Cogan, M. Kagan, E. Strauss and A. Schwarztman, JHEP **1502**, 118 (2015) doi:10.1007/JHEP02(2015)118 [arXiv:1407.5675 [hep-ph]].

[13] L. de Oliveira, M. Kagan, L. Mackey, B. Nachman and A. Schwartzman, JHEP **1607**, 069 (2016) doi:10.1007/JHEP07(2016)069 [arXiv:1511.05190 [hep-ph]].

[14] P. Baldi, K. Bauer, C. Eng, P. Sadowski and D. Whiteson, Phys. Rev. D **93**, no. 9, 094034 (2016) doi:10.1103/PhysRevD.93.094034 [arXiv:1603.09349 [hep-ex]].

[15] G. Kasieczka, T. Plehn, M. Russell and T. Schell, JHEP **1705**, 006 (2017) doi:10.1007/JHEP05(2017)006 [arXiv:1701.08784 [hep-ph]].

[16] S. Macaluso and D. Shih, JHEP **1810**, 121 (2018) doi:10.1007/JHEP10(2018)121 [arXiv:1803.00107 [hep-ph]].

[17] L. G. Almeida, M. Backovic, M. Cliche, S. J. Lee and M. Perelstein, JHEP **1507**, 086 (2015) doi:10.1007/JHEP07(2015)086 [arXiv:1501.05968 [hep-ph]].

[18] J. Pearkes, W. Fedorko, A. Lister and C. Gay, arXiv:1704.02124 [hep-ex].

[19] M. Erdmann, E. Geiser, Y. Rath and M. Rieger, arXiv:1812.09722 [hep-ex].

[20] G. Louppe, K. Cho, C. Becot and K. Cranmer, JHEP **1901**, 057 (2019) doi:10.1007/JHEP01(2019)057 [arXiv:1702.00748 [hep-ph]].

[21] P. T. Komiske, E. M. Metodiev and J. Thaler, JHEP **1804**, 013 (2018) doi:10.1007/JHEP04(2018)013 [arXiv:1712.07124 [hep-ph]].

[22] A. Butter, G. Kasieczka, T. Plehn and M. Russell, SciPost Phys. **5**, no. 3, 028 (2018) doi:10.21468/SciPostPhys.5.3.028 [arXiv:1707.08966 [hep-ph]].

[23] P. T. Komiske, E. M. Metodiev and J. Thaler, JHEP **1901**, 121 (2019) doi:10.1007/JHEP01(2019)121 [arXiv:1810.05165 [hep-ph]].

[24] L. Moore, K. Nordström, S. Varma and M. Fairbairn, arXiv:1807.04769 [hep-ph].

[25] T. Sjöstrand *et al.*, Comput. Phys. Commun. **191** (2015) 159 doi:10.1016/j.cpc.2015.01.024 [arXiv:1410.3012 [hep-ph]].

[26] J. de Favereau *et al.* [DELPHES 3 Collaboration], JHEP **1402**, 057 (2014) doi:10.1007/JHEP02(2014)057 [arXiv:1307.6346 [hep-ex]].

[27] M. Cacciari, G. P. Salam and G. Soyez, JHEP **0804**, 063 (2008) doi:10.1088/1126-6708/2008/04/063 [arXiv:0802.1189 [hep-ph]].

[28] M. Cacciari, G. P. Salam and G. Soyez, Eur. Phys. J. C **72**, 1896 (2012) doi:10.1140/epjc/s10052-012-1896-2 [arXiv:1111.6097 [hep-ph]].

[29] link to top tagging sample, for more information and citation please use Ref. [22].

[30] F. Chollet, Keras (2015).

[31] M. Abadi *et al.*, OSDI **16**, 265 (2016).

[32] S. Xie, R. B. Girshick, P. Dollár, Z. Tu, and K. He, CoRR (2016) [arXiv:1611.05431 [cs.CV]].

[33] T. Chen *et al.*, CoRR (2015) [arXiv:1512.01274 [cs.CV].

[34] J. Gallicchio and M. D. Schwartz, Phys. Rev. Lett. **105**, 022001 (2010) doi:10.1103/PhysRevLett.105.022001 [arXiv:1001.5027 [hep-ph]].

[35] L. de Oliveira, M. Paganini and B. Nachman, Comput. Softw. Big Sci. **1**, no. 1, 4 (2017) doi:10.1007/s41781-017-0004-6 [arXiv:1701.0592 [stat.ML]].

[36] M. Aaboud *et al.* [ATLAS Collaboration], arXiv:1808.07858 [hep-ex].

[37] G. Aad *et al.* [ATLAS Collaboration], Eur. Phys. J. C **77**, 490 (2017) doi:10.1140/epjc/s10052-017-5004-5 [arXiv:1603.02934 [hep-ex]].

[38] K. Datta and A. Larkoski, JHEP **1706**, 073 (2017) doi:10.1007/JHEP06(2017)073 arXiv:1704.08249 [hep-ph].

[39] A. Buckley, J. Butterworth, L. Lonnblad, D. Grellscheid, H. Hoeth, J. Monk, H. Schulz and F. Siegert, Comput. Phys. Commun. **184**, 2803 (2013) doi:10.1016/j.cpc.2013.05.021 [arXiv:1003.0694 [hep-ph]].

[40] J. Thaler and K. Van Tilburg, JHEP **1202**, 093 (2012) doi:10.1007/JHEP02(2012)093 [arXiv:1108.2701 [hep-ph]].

[41] S. D. Ellis and D. E. Soper, Phys. Rev. D **48**, 3160 (1993) doi:10.1103/PhysRevD.48.3160 [arXiv:hep-ph/9305266].

[42] D. P. Kingma and J. Ba, CoRR (2014) [arXiv:1412.6980 [cs.LG]].

[43] CMS Collaboration, CMS-DP-2017-049

[44] K. He, X. Zhang, S. Ren, and J. Sun, CoRR (2016) [arXiv:1603.05027 [cs.CV]].

[45] H. Qu and L. Gouskos, arXiv:1902.08570 [hep-ph].

[46] Y. Wang *et al.* CoRR (2018) [arXiv:1801.07829 [cs.CV]].

[47] K. He, X. Zhang, S. Ren, and J. Sun, CoRR (2015) [arXiv:1512.03385 [cs.CV]].

[48] S. Choi, S. J. Lee and M. Perelstein, arXiv:1806.01263 [hep-ph].

[49] J. Barnard, E. N. Dawe, M. J. Dolan and N. Rajcic, Phys. Rev. D **95**, no. 1, 014018 (2017) doi:10.1103/PhysRevD.95.014018 [arXiv:1609.00607 [hep-ph]].

[50] S. Ioffe and C. Szegedy, Proc. 32nd International Conference on Machine Learning (ICML) (2015), [arXiv:1502.03167 [cs.LG]].

[51] F. V. Tkachov, Int. J. Mod. Phys. A **12**, 5411 (1997) doi:10.1142/S0217751X97002899 [arXiv:hep-ph/9601308].

[52] A. J. Larkoski, G. P. Salam and J. Thaler, JHEP **1306**, 108 (2013) doi:10.1007/JHEP06(2013)108 [arXiv:1305.0007 [hep-ph]].

[53] I. Moult, L. Necib and J. Thaler, JHEP **1612**, 153 (2016) doi:10.1007/JHEP12(2016)153 [arXiv:1609.07483 [hep-ph]].

[54] R. A. Fisher, Annals of human genetics **7**, 179 (1936).

[55] F. Pedregosa *et al.* Journal of Machine Learning Research **12**, 2825 (2011).

[56] P. T. Komiske, E. M. Metodiev and J. Thaler, EnergyFlow

[57] M. Zaheer, S, Kottur, S. Ravanbakhsh, B. Póczos, R. R. Salakhutdinov, and A. J. Smola, Annual Conference on Neural Information Processing Systems (2017), [arXiv:1703.06114 [cs.LG]].

[58] K. He, X. Zhang, S. Ren, and J. Sun, Jian, IEEE international conference on computer vision, 1026 (2015).

[59] L. M. Dery, B. Nachman, F. Rubbo and A. Schwartzman, JHEP **1705**, 145 (2017) doi:10.1007/JHEP05(2017)145 [arXiv:1702.00414 [hep-ph]]; T. Cohen, M. Freytsis and B. Ostdiek, JHEP **1802**, 034 (2018) doi:10.1007/JHEP02(2018)034 [arXiv:1706.09451 [hep-ph]]; E. M. Metodiev, B. Nachman and J. Thaler, JHEP **1710**, 174 (2017) doi:10.1007/JHEP10(2017)174 [arXiv:1708.02949 [hep-ph]]. P. T. Komiske, E. M. Metodiev, B. Nachman and M. D. Schwartz, Phys. Rev. D **98**, no. 1, 011502 (2018) doi:10.1103/PhysRevD.98.011502 [arXiv:1801.10158 [hep-ph]]; A. Andreassen, I. Feige, C. Frye and M. D. Schwartz, arXiv:1804.09720 [hep-ph]; T. Heimel, G. Kasieczka, T. Plehn and J. M. Thompson, arXiv:1808.08979 [hep-ph]; M. Farina, Y. Nakai and D. Shih, arXiv:1808.08992 [hep-ph]. O. Cerri, T. Q. Nguyen, M. Pierini, M. Spiropulu and J. R. Vlimant, arXiv:1811.10276 [hep-ex].