Comparing point cloud strategies for collider event classification

Peter Onyisi,[1,†] Delon Shen,[2,*] and Jesse Thaler[3,4,‡]

[1]*Department of Physics, University of Texas at Austin, Austin, Texas 78712, USA*
[2]*Department of Physics, Stanford University, Stanford, California 94305, USA*
[3]*Center for Theoretical Physics, Massachusetts Institute of Technology,*
Cambridge, Massachusetts 02139, USA
[4]*The NSF AI Institute for Artificial Intelligence and Fundamental Interactions*

(Received 9 January 2023; accepted 15 June 2023; published 5 July 2023)

In this paper, we compare several event classification architectures defined on the point cloud representation of collider events. These approaches, which are based on the frameworks of Deep Sets and edge convolutions, circumvent many of the difficulties associated with traditional feature engineering. To benchmark our architectures against more traditional event classification strategies, we perform a case study involving Higgs boson decays to tau leptons. We find a 2.5 times increase in performance compared to a baseline ATLAS analysis with engineered features. Our point cloud architectures can be viewed as simplified versions of graph neural networks, where each particle in the event corresponds to a graph node. In our case study, we find the best balance of performance and computational cost for simple Parwise architectures, which are based on learned edge features.

DOI: 10.1103/PhysRevD.108.012001

I. INTRODUCTION

When analyzing data collected at the Large Hadron Collider (LHC), the ability to distinguish between specific production and decay channels is vital for picking out signal events among overwhelming backgrounds. In the context of Higgs boson studies, the ATLAS and CMS Collaborations rely heavily on dense neural networks (dNNs) [1,2] and boosted decision trees (BDTs) [3–10] for event classification. These classifiers are typically trained on Monte Carlo simulated events to separate signal events from expected background processes. Both dNNs and BDTs expect collider events to be represented by fixed-size inputs. Creating a robust fixed-size representation of a collider event is challenging, however, often requiring hand engineering of a fixed number of features to distill relevant information from a variable number of particles.

In this paper, we compare event classification architectures defined on *point clouds*, which are a natural variable-size representation of collider events. Our architectures draw inspiration from two complementary approaches to point cloud processing. The first is *deep sets* [11,12], which compute global information about an event based on permutation symmetric functions. The second is *edge convolutions* (EdgeConvs) [13,14], which compute local information associated with each particle and its neighbors. We propose architectures which center around three different strategies for event classification:

(i) emphasizing local information with *multiple summations*;
(ii) improving latent representations of collider events from *iterated convolutions*; and
(iii) emphasizing global information through *nested concatenation* of global features.

The nested concatenation structure provides an interesting alternative to the equivariant layers of Refs. [11,15], with the same degree of expressivity.

To test our architectures, we perform a case study involving signal and background binary classification of Higgs boson decays to tau leptons. This channel has been studied by ATLAS [3–5], whose results we use as a baseline for comparison, and by CMS [16–19]. Based on our study, we recommend the *Parwise architecture* in Eq. (11) below, which can be interpreted as either

(i) a Deep Set acting on pairs of particles or
(ii) a symmetric pooling over EdgeConvs.

Compared to more complex graph neural networks, this pairwise structure balances classification performance, computational efficiency, and conceptual simplicity. We study

[*] Corresponding author.
delon@stanford.edu
[†] ponyisi@utexas.edu
[‡] jthaler@mit.edu

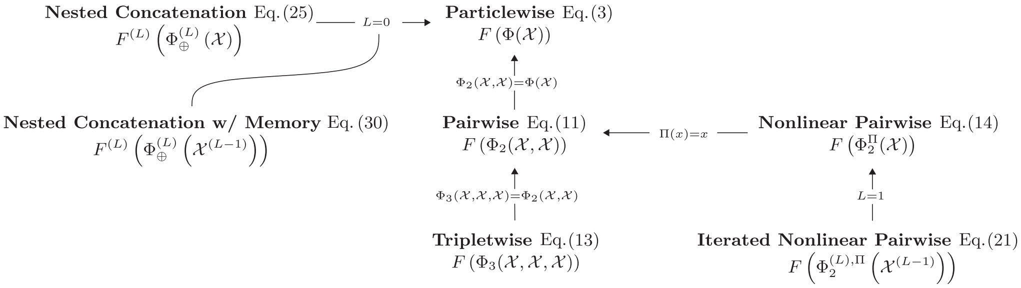

Nested Concatenation Eq. (25)
$F^{(L)}\left(\Phi_\oplus^{(L)}(\mathcal{X})\right)$
$L=0$
Particlewise Eq. (3)
$F(\Phi(\mathcal{X}))$
$\Phi_2(\mathcal{X},\mathcal{X})=\Phi(\mathcal{X})$
Nested Concatenation w/ Memory Eq. (30)
$F^{(L)}\left(\Phi_\oplus^{(L)}\left(\mathcal{X}^{(L-1)}\right)\right)$
Pairwise Eq. (11)
$F(\Phi_2(\mathcal{X},\mathcal{X}))$
$\Pi(x)=x$
Nonlinear Pairwise Eq. (14)
$F(\Phi_2^\Pi(\mathcal{X}))$
$\Phi_3(\mathcal{X},\mathcal{X},\mathcal{X})=\Phi_2(\mathcal{X},\mathcal{X})$
$L=1$
Tripletwise Eq. (13)
$F(\Phi_3(\mathcal{X},\mathcal{X},\mathcal{X}))$
Iterated Nonlinear Pairwise Eq. (21)
$F\left(\Phi_2^{(L),\Pi}\left(\mathcal{X}^{(L-1)}\right)\right)$

FIG. 1. Summary of the point cloud architectures studied in this paper, where the numbers refer to equations in the text. We also highlight limits where one architecture reduces to another. The particlewise, pairwise, and Tripletwise architectures are described in Sec. II D. The Nonlinear Pairwise and Iterated Nonlinear Parwise architectures are described in Sec. II E. The nested concatenation and Nested Concatenation with Memory architectures are described in Sec. II F.

the performance of the Parwise architecture as a function of the *latent dimension* size, finding that even a single latent dimension outperforms the baseline ATLAS strategy. Interestingly, we find that the discriminatory features identified by the Parwise architecture have some correlations with the traditionally selected hand-engineered features.

The point cloud representation and corresponding architectures have appeared before in the literature for various classification tasks [12,14,15,20–24]. We refer to Refs. [25,26] for a more thorough review of the use of point clouds in particle physics. Using point clouds, collider events are represented as an unordered set of (n)-dimensional vectors, where each vector corresponds to a measured particle in that collision event. We need a *set*, since there are a variable number of particles in each collision event. This set is *unordered*, since there is no inherent ordering to the particles.[1] As discussed below, a key benefit of using architectures defined on point clouds is that it bypasses the traditional feature engineering game and associated combinatorial problems.

The remainder of this paper is organized as follows. In Sec. II, we describe our architectures and review their motivation and inspirations. We make these architectures available along with example code on GitHub [27], and we describe the connection between nested concatenation and equivariant layers in Appendix A. In Sec. III, we perform a classification case study of the $H \rightarrow \tau\tau$ decay channel, comparing our proposed architectures with a baseline ATLAS strategy, and we show visualizations of the latent space for the Parwise architecture. Details of the neural network parameters are given in Appendix B. We conclude in Sec. IV with a summary of our recommendations and areas for future exploration.

II. POINT CLOUD ARCHITECTURES

In this section, we describe our point cloud architectures and their motivation. We start by describing why the point cloud representation is more natural for event classification compared to traditional fixed-size inputs. We then review Deep Sets and EdgeConvs, which are the main inspirations for our architectures. Following this, we describe our proposed architectures, as summarized in Fig. 1.

A. Why point clouds for event classification

The point cloud representation of collider events avoids two of the key challenges when trying to construct robust fixed-size inputs for event classification.

(i) *Combinatorial ambiguities.*—One way of creating a fixed-size representation of an event is to define high-level kinematic variables, but this can lead to combinatorial challenges. For example, if we find a good discriminatory feature that is derived assuming the final state has two b-tagged jets, but mistagging leads to three measured b-tagged jets, then one has to decide which of the three pairs should be used to compute the high-level feature. Similarly, if there is only one measured b-tagged jet, due to mistagging or kinematic acceptance, then the high-level feature is ill defined, even if, in principle, there is enough information available for event classification.

(ii) *Truncation ambiguities.*—Another way to create a fixed-size representation of an event is to input the kinematics of a fixed number of particles. This, however, introduces a dependence on how the particles are ordered and which ones are truncated away. While some orderings, like taking the most energetic particles, have a physical motivation, they might not yield the best discrimination power, especially if particle correlations are relevant. There is also a question about how to properly pad events when there are fewer particles than the desired fixed-size representation.

The point cloud representation avoids these combinatorial and truncation ambiguities. By enforcing permutation

[1] It is sometimes convenient to sort particles according to some measure of energy. When comparing our point cloud architectures to fixed-size input dNNs, we will sort over the particle transverse momenta (p_T).

invariance, all possible particle combinations are automatically considered for event classification. By allowing for variable-size inputs, there is no need for ordering or truncation. Of course, it is possible that cleverly engineered fixed-size features could outperform generic point cloud architectures, though this turns out not to hold for the case study in Sec. III. Graph neural networks are a popular approach to point cloud processing, but the simpler architectures studied in this paper also avoid these ambiguities with reduced computational complexity.

The architectures below take as input a set of M particles:

$$\mathcal{X} = \{x_1, \ldots, x_M\} \subset \mathbb{R}^n, \tag{1}$$

where each particle x_i is described by n features. These features could include particle characteristics like p_T and b-tag score. Let the classification of an event $\mathcal{X}$ be called

$$f(\mathcal{X}) \subset \mathbb{R}^m, \tag{2}$$

where f represents an event classification architecture for m possible channels. For binary classification as studied in this paper, $m = 2$.

B. Review of Deep Sets

Deep Sets are a way to parametrize permutation-symmetric functions with neural networks. They were defined in Ref. [11] and first introduced to the particle physics community in Ref. [12], under the name of *particle flow networks*. Deep Sets have achieved state-of-the-art performance on various collider physics tasks, such as quark and gluon jet discrimination and boosted top tagging [28].

As shown in Ref. [11], a function $f(\mathcal{X})$ operating on a set $\mathcal{X}$ is permutation invariant if (and, under certain assumptions, only if) it can be decomposed into the form

$$f(\mathcal{X}) = F\left(\frac{1}{M}\sum_{i=1}^{M}\Phi(x_i)\right). \tag{3}$$

Each particle $x_i \in \mathcal{X}$ is transformed into a latent representation of dimension ℓ by a function:

$$\Phi\colon\ \mathbb{R}^n \to \mathbb{R}^\ell. \tag{4}$$

The particlewise outputs $\Phi(x)$ are averaged over and processed by an eventwise function:

$$F\colon\ \mathbb{R}^\ell \to \mathbb{R}^m. \tag{5}$$

To approximate the optimal event classifier, the functions Φ and F are parametrized by neural networks.

The factor of $1/M$ in Eq. (3) differs from the presentation in Refs. [11,12], where sum pooling (instead of average pooling) was the default. Average pooling simplifies some of the later notation, and we use this pooling operation to define the action of Φ on the set $\mathcal{X}$:

$$\Phi(\mathcal{X}) \equiv \frac{1}{M}\sum_{i=1}^{M}\Phi(x_i). \tag{6}$$

This notation makes more clear that the function Φ effectively maps the entire set $\mathcal{X}$ into a latent representation of dimension ℓ.

C. Review of edge convolutions

EdgeConvs are a way to incorporate local neighborhood information within point clouds for learning tasks. They were introduced in Ref. [13] and first used in particle physics by Ref. [14], under the name of ParticleNet. Architectures incorporating EdgeConvs have also achieved state-of-the-art performance for collider tasks [28].

The EdgeConv mechanism in Ref. [13] boils down to the following transformation of a particle $x_i \in \mathcal{X}$:

$$\Phi_2(x_i, \mathcal{X}) \equiv \frac{1}{M}\sum_{j=1}^{M}\Phi_2(x_i, x_j), \tag{7}$$

where the notation mirrors that in Eq. (6). Here, the particle x_i is transformed into a latent representation through a function Φ_2 based on pairwise information:

$$\Phi_2(x_i, x_j)\colon\ \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^\ell. \tag{8}$$

Like before, ℓ is the latent dimension, and Φ_2 is parametrized by a neural network. For later purposes, we can define Φ_2 acting on two sets via

$$\begin{aligned}\Phi_2(\mathcal{X}, \mathcal{X}) &\equiv \frac{1}{M}\sum_{i=1}^{M}\Phi_2(x_i, \mathcal{X})\\ &\equiv \frac{1}{M^2}\sum_{i=1}^{M}\sum_{j=1}^{M}\Phi_2(x_i, x_j).\end{aligned} \tag{9}$$

The EdgeConv operation can be iterated and combined with various nonlinearities and pooling operations to create a *Dynamic Graph Convolutional Neural Network* (DGCNN). These were used for various computer vision and graphics tasks in Ref. [13] and shown to achieve strong performance on several standard benchmark tasks. Furthermore, during these benchmarks, it was shown that DGCNNs achieved the best trade-off between the number of parameters and the run-time of the model.

D. Multiple summation

We now describe our first set of proposed architectures, based on simple summations. The Particlewisie architecture is a direct application of the Deep Set formalism in Eq. (3):

$$f(\mathcal{X}) = F\left(\frac{1}{M}\sum_{i=1}^{M}\Phi(x_i)\right) = F(\Phi(\mathcal{X})), \tag{10}$$

where, in the last line, we are using the notation from Eq. (6). The set of particles $\mathcal{X}$ are transformed into a latent ℓ-dimensional representation by Φ, which is further processed into the output by F. Because the particlewise function Φ can see only one particle at a time, this architecture is inefficient at capturing local information in the vicinity of each particle.

We can improve our ability to process local information using the Parwise architecture, which acts on all *pairs* of particles in an event[2]:

$$\begin{aligned} f(\mathcal{X}) &= F\left(\frac{1}{M^2}\sum_{i=1}^{M}\sum_{j=1}^{M}\Phi_2(x_i, x_j)\right) \\ &= F\left(\frac{1}{M}\sum_{i=1}^{M}\Phi_2(x_i, \mathcal{X})\right) \\ &= F(\Phi_2(\mathcal{X}, \mathcal{X})). \end{aligned} \tag{11}$$

Using Eq. (9), the three different notations above correspond to three different ways of thinking about this architecture. The first line corresponds to applying the Deep Set formalism in Eq. (3) to the set of (ordered) particle pairs.[3] The second line corresponds to applying the EdgeConvs in Eq. (7) to each particle and then performing average pooling and postprocessing. The last line emphasizes that the role of Φ_2 is to map particle pairs to an ℓ-dimensional latent space, which makes this local pairwise information directly accessible when constructing a latent representation. If we choose $\Phi_2(x_i, x_j) = \Phi(x_i)$, such that the x_j input is ignored, then the Parwise architecture reduces to the particlewise one.

The natural generalization of the above constructions is the Tripletwise architecture, which involves a function that maps triplets of particles into a latent space:

$$\Phi_3(x_i, x_j, x_k)\colon\ \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^\ell. \tag{12}$$

Mirroring the notation in Eq. (9), we can define this architecture in multiple ways:

$$\begin{aligned} f(\mathcal{X}) &= F\left(\frac{1}{M^3}\sum_{i=1}^{M}\sum_{j=1}^{M}\sum_{k=1}^{M}\Phi_3(x_i, x_j, x_k)\right) \\ &= F\left(\frac{1}{M^2}\sum_{i=1}^{M}\sum_{j=1}^{M}\Phi_3(x_i, x_j, \mathcal{X})\right) \\ &= F\left(\frac{1}{M}\sum_{i=1}^{M}\Phi_3(x_i, \mathcal{X}, \mathcal{X})\right) \\ &= F(\Phi_3(\mathcal{X}, \mathcal{X}, \mathcal{X})). \end{aligned} \tag{13}$$

This tripletwise structure makes available even more local information when constructing a latent representation of an event. If we choose $\Phi_3(x_i, x_j, x_k) = \Phi_2(x_i, x_j)$, such that the x_k input is ignored, then the Tripletwise architecture reduces to the pairwise one. We found it impractical to use architectures with more nested summations due to their heavy computational cost.

E. Iterated convolutions

One way to make neural networks more expressive is to introduce more nonlinearities. This is the motivation for our iterated convolution architectures, which can be viewed as a special case of DGCNNs [13].

A natural evolution of the Parwise architecture in Eq. (11) involves inserting an additional nonlinear function Π implemented with a neural network between the two sums, or, equivalently, after the EdgeConv layer, which yields the nonlinear Parwise architecture:

$$\begin{aligned} f(\mathcal{X}) &= F\left(\frac{1}{M}\sum_{i=1}^{M}\Pi\left(\frac{1}{M}\sum_{j=1}^{M}\Phi_2(x_i, x_j)\right)\right) \\ &= F\left(\frac{1}{M}\sum_{i=1}^{M}\Pi(\Phi_2(x_i, \mathcal{X}))\right) \\ &= F(\Phi_2^{\Pi}(\mathcal{X})). \end{aligned} \tag{14}$$

Here, we have introduced the notation

$$\Phi_2^{\Pi}(\mathcal{X}) \equiv \frac{1}{M}\sum_{i=1}^{M}\Pi(\Phi_2(x_i, \mathcal{X})), \tag{15}$$

which emphasizes that we are still transforming the point cloud into a latent representation of dimension ℓ. Note that the output of Φ_2 need not be ℓ dimensional as in Eq. (8), since now the latent representation is constructed through both Π and Φ_2:

$$\Phi_2(x_i, x_j)\colon\ \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^{\ell'}, \tag{16}$$

$$\Pi\colon\ \mathbb{R}^{\ell'} \to \mathbb{R}^{\ell}, \tag{17}$$

where ℓ' and ℓ could differ.

[2] We thank Patrick Komiske and Eric Metodiev for foundational discussions related to this architecture.

[3] With average pooling, we could choose to work with unordered pairs and make $\Phi_2(x_i, x_j)$ symmetric in its arguments, but using ordered pairs often yields simpler manipulations.

We can increase the expressivity of the latent representations by iteratively applying nonlinearities and EdgeConvs. Let

$$\mathcal{X}^{(0)} \equiv \mathcal{X} \tag{18}$$

be the original point cloud. We then define the point cloud at depth d as

$$\mathcal{X}^{(d)} = \left\{x_1^{(d)}, \ldots, x_M^{(d)}\right\}, \tag{19}$$

where the set size M matches the original point cloud. The ith element at depth d is the result of applying a nonlinearity $\Pi^{(d)}$ to an EdgeConv defined by $\Phi_2^{(d)}$:

$$x_i^{(d)} = \Pi^{(d)}\left(\Phi_2^{(d)}\left(x_i^{(d-1)}, \mathcal{X}^{(d-1)}\right)\right). \tag{20}$$

The Iterated Nonlinear Parwise architecture at total depth L is achieved by average pooling over these elements and then postprocessing:

$$\begin{aligned} f(\mathcal{X}) &= F\left(\frac{1}{M}\sum_{i=1}^{M} x_i^{(L)}\right) \\ &= F\left(\frac{1}{M}\sum_{i=1}^{M} \Pi^{(L)}\left(\Phi_2^{(L)}\left(x_i^{(L-1)}, \mathcal{X}^{(L-1)}\right)\right)\right) \\ &\equiv F\left(\Phi_2^{(L),\Pi}(\mathcal{X}^{(L-1)})\right). \end{aligned} \tag{21}$$

In the last line, we are using the same notation as Eq. (15) to emphasize that $\Phi_2^{(L),\Pi}$ transforms the point cloud into an ℓ-dimensional latent representation. When $L = 1$, this reduces to the nonlinear Parwise architecture from Eq. (14). A version of this architecture is also possible for the tripletwise case, but we shall not pursue it due to its heavy computational cost.

F. Nested concatenation of global features

Our final class of architectures combines global information about the whole event with local information about particles. Let $\hat{f}(\mathcal{X})$ be a permutation invariant function, which captures global information about the collider event $\mathcal{X}$. For example, $\hat{f}(\mathcal{X})$ could simply be a Deep Set applied to the set of particles:

$$\hat{f}(\mathcal{X}) = \hat{F}\left(\frac{1}{M}\sum_{i=1}^{M} \hat{\Phi}(x_i)\right). \tag{22}$$

The global information from $\hat{f}(\mathcal{X})$ can then be concatenated ($\oplus$) with local features associated with each particle:

$$f(\mathcal{X}) = F\left(\frac{1}{M}\sum_{i=1}^{M} \Phi(x_i \oplus \hat{f}(\mathcal{X}))\right). \tag{23}$$

This concatenation is a conceptually simple way to let individual particles see global information about the point cloud.

The Nested Concatenation architecture iterates the structure in Eq. (23) for L times. Let the base case be a standard Deep Set:

$$f^{(0)}(\mathcal{X}) = F^{(0)}\left(\frac{1}{M}\sum_{i=1}^{M} \Phi^{(0)}(x_i)\right). \tag{24}$$

At level d, we have

$$\begin{aligned} f^{(d)}(\mathcal{X}) &= F^{(d)}\left(\frac{1}{M}\sum_{i=1}^{M} \Phi^{(d)}\left(x_i \oplus f^{(d-1)}(\mathcal{X})\right)\right) \\ &= F^{(d)}\left(\Phi_\oplus^{(d)}(\mathcal{X})\right). \end{aligned} \tag{25}$$

In the last line, we have introduced the notation

$$\Phi_\oplus^{(d)}(\mathcal{X}) = \frac{1}{M}\sum_{i=1}^{M} \Phi^{(d)}\left(x_i \oplus f^{(d-1)}(\mathcal{X})\right), \tag{26}$$

which emphasizes that $\Phi_\oplus^{(d)}$ transforms the point cloud into a latent representation of dimension ℓ, just as in the previous architectures. The final architecture at total level L is

$$f(\mathcal{X}) \equiv f^{(L)}(\mathcal{X}), \tag{27}$$

such that Eq. (25) reduces to Eq. (3) when $L = 0$.

To allow for more dynamic manipulation of the point cloud information, we define the Nested Concatenation with Memory architecture. In this architecture, the intermediate latent representations of particles generated by $(d-1)$th nested layer are also utilized in the dth nested layer. We define the same base case function as Eq. (24), with the base case set as $\mathcal{X}^{(0)} \equiv \mathcal{X}$. After d nested layers, the set takes the form

$$\mathcal{X}^{(d)} = \left\{x_1^{(d)}, \ldots, x_M^{(d)}\right\}, \tag{28}$$

where the ith element is determined via

$$x_i^{(d)} = \Phi^{(d)}\left(x_i^{(d-1)} \oplus f^{(d-1)}(\mathcal{X}^{(d-1)})\right). \tag{29}$$

The function $f^{(d)}(\mathcal{X}^{(d)})$ sums and processes the latent representation $\mathcal{X}^{(d)}$ with a function $F^{(d)}$:

$$\begin{aligned} f^{(d)}(\mathcal{X}^{(d)}) &= F^{(d)}\left(\frac{1}{M}\sum_{i=1}^{M} x_i^{(d)}\right) \\ &= F^{(d)}(\Phi_{\oplus}^{(d)}(\mathcal{X}^{(d-1)})). \end{aligned} \tag{30}$$

The function $\Phi_{\oplus}^{(d)}$ is the same as in Eq. (26) but now applied to the set $\mathcal{X}^{(d-1)}$. Like before, we let

$$f(\mathcal{X}) \equiv f^{(L)}(\mathcal{X}^{(L)}) \tag{31}$$

be the classification of collider event $\mathcal{X}$ at level L.

It is worth mentioning that Ref. [11] defined an alternative method to incorporate global information based on permutation equivariance; this structure was used for jet tagging in Ref. [15]. In Appendix A, we show that permutation equivariant Deep Sets are a special case of our Nested Concatenation architecture. We prefer to use the more general concatenation structure, though, due to its flexibility.

III. EVENT CLASSIFICATION CASE STUDY

In this section, we describe the setup and results of our event classification case study to benchmark our proposed architectures against more traditional methods. We start with a description of the signal and background processes that will be the context for our case study. We then describe how we generate synthetic datasets and preprocess the inputs for both traditional architectures and our proposed architectures. Following this, we present several performance metrics of the tested architectures and advocate for the Parwise architecture as the best balance between computational cost and performance. We then perform a latent dimension study of the Parwise architecture and visualize the separation of signal and background events in the latent space. Finally, we examine correlations between the features found to be useful to represent collider events by our Parwise architecture and the hand-engineered features chosen by the ATLAS Collaboration.

A. Signal and background processes

Our case study is based on a problem relevant to analyzing the $H \to \tau\tau$ decay channel [3–5]. For leptonic Higgs boson decays, the $H \to \tau^+\tau^-$ channel has the largest branching ratio of 6.3% [29,30], which makes this channel a prime candidate to study the Yukawa-Higgs mechanism for mass generation. The presence of neutrinos in the final state of this process, however, degrades the resolution of the measured Higgs boson four-momentum. This degraded resolution makes the signal process much more difficult to distinguish from background processes, thereby motivating a machine learning approach.

To tackle the challenge of identifying the $H \to \tau^+\tau^-$ final state, one typically isolates different event topologies and studies them individually. One $H \to \tau\tau$ topology of interest—which will serve as the signal process in our classification case study—is the production of a Higgs boson associated with a pair of top quarks where both top quarks and both τ leptons decay hadronically. We denote this signal process as

$$t\bar{t}(H \to \tau\tau) \quad \text{or} \quad t\bar{t}H \text{ for short.} \tag{32}$$

A schematic of this process is shown in Fig. 2(a). Ideally, the final state for this process would result in two τ-tagged jets, two b-tagged jets, and four additional jets from $W^{\pm}$ decay. This channel was considered by ATLAS in Ref. [3] and by CMS in Ref. [31].

The main background process that mimics this $t\bar{t}H$ signature—and significantly hinders the analysis of this channel [3]—is the production of a top-antitop quark pair where each top quark decays as $t \to \tau\nu b$ and both τ decay hadronically. We denote this background process as

$$t\bar{t}(\to \tau\nu b) \quad \text{or} \quad t\bar{t} \text{ for short.} \tag{33}$$

A schematic of this process is shown in Fig. 2(b). The $t\bar{t}$ channel can mimic the signature of the ideal $t\bar{t}H$ process if there are four additional jets from gluons radiated before the hard scattering. In our case study, we focus on distinguishing between $t\bar{t}H$ events and $t\bar{t}$ events. A full analysis, of course, would consider multiple Higgs production topologies.

These two channels both exhibit high multiplicity final states with a diverse range of final state objects. This leads to many potential combinatorial reconstructions and a high probability of detecting extra or losing relevant objects. These characteristics make manually constructing features difficult and motivates the need for flexibility in the number of input objects. Our case study is, therefore, representative of situations where we hope to make gains from using point-cloud-based architectures, which naturally account for combinatorial ambiguities and incomplete reconstruction. The particular processes we study are among the highest multiplicity channels currently analyzed at the LHC.

B. Data generation

Following the $t\bar{t}(H \to \tau\tau)$ analysis strategy in Ref. [3], we select events that satisfy the following properties:

(1) Two visible τ-tagged jets, with kinematic conditions
  (a) $\max(\{p_T^{\tau}\}) > 40$ GeV,
  (b) $\min(\{p_T^{\tau}\}) > 30$ GeV,
  (c) $0.6 < \Delta R_{\tau\tau} < 2.5$,
  (d) $|\eta| \leq 2.5$,
  (e) $|\Delta\eta_{\tau\tau}| < 1.5$,
  (f) $0.1 < x_1, x_2 < 1.4$ (defined below);

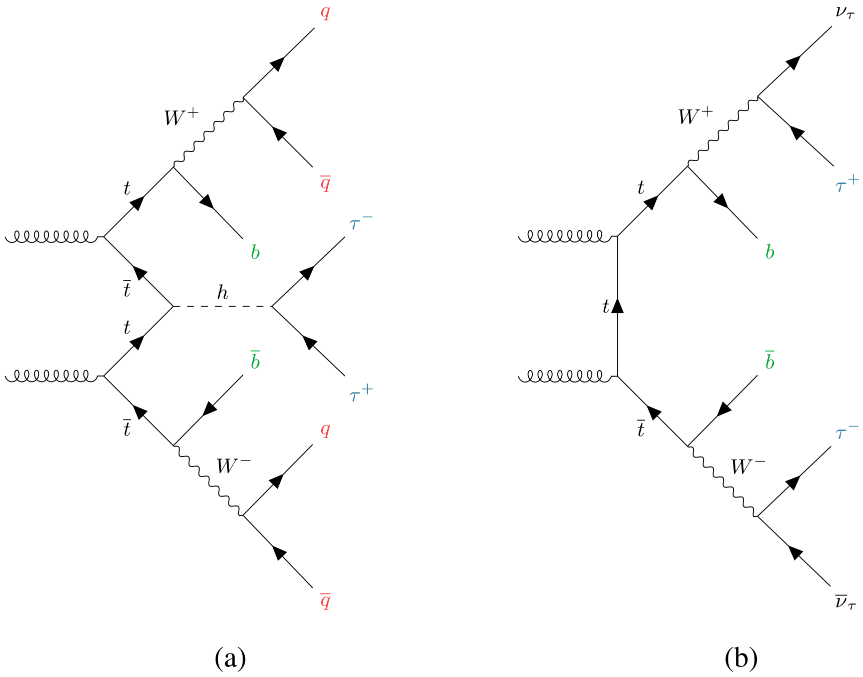


FIG. 2. A schematic of the (a) signal and (b) background process considered in our event classification case study. The signal process in (a) involves the production of a Higgs boson decaying to $\tau\tau$ associated with the production of a pair of top quarks where both top quarks and both τ leptons decay hadronically. The background process in (b) involves the production of a pair of top quarks where both tops decay to $\tau\nu b$ and both τ's decay hadronically. The background process can mimic the signal process if there are additional jets from initial-state radiation, which was found to be a significant effect in a recent ATLAS analysis of $H \to \tau\tau$ [3].

(2) (≥ 5 jets and $\geq 2b$ tags) *or* (≥ 6 jets and $\geq 1b$ tags), with kinematic conditions for
   (a) the leading (non-τ) jet:
   (i) $p_T > 70$ GeV,
   (ii) $|\eta| < 3.2$; and
   (b) other jets:
   (i) $p_T \geq 20$ GeV,
   (ii) $|\eta| \leq 5$;
(3) ≤ 15 jets total.[4]

Here, the transverse momenta (p_T) and pseudorapidities (η) are defined with respect to the beam line, $\Delta R^2 = \Delta\eta^2 + \Delta\phi^2$ is the distance between objects in the pseudorapidity-azimuth (η-ϕ) plane, and x_1 and x_2 are the momentum fractions carried away by visible τ decay products as computed by the collinear approximation [32–34].

For both the signal and background channels, we generate events with MadGraph 5 v3.1.1 [35] and PYTHIA 8.245 [36]. These events are passed through the DELPHES 3.5.0 [37] detector simulation with the ATLAS card.[5] Jets are then clustered with the $R = 0.4$ anti-k_T algorithm [38] using the FastJet 3.3.4 [39] package. From all of the generated events, we extract 80 000 events from each channel that satisfy the event selection criteria, such that we have balanced datasets. For the machine learning study, we split each dataset into 70% for training and 30% for testing.

C. Data processing

The ATLAS analysis in Ref. [3] is based on the following engineered features:

(1) Σp_T^{jets}.—scalar sum of all jets p_T;
(2) $M_{\hat{W}}$.—invariant mass of the dijet with invariant mass closest to W-boson mass;
(3) $\Delta R_{\min}$.—smallest ΔR between any two jets;
(4) $M_{\hat{t}}$.—invariant mass of the trijet with invariant mass closest to top quark mass;
(5) $\Delta R_{\tau\tau}$.—ΔR between two τ-tagged jets;
(6) $|\Delta\eta_{\tau\tau}$—$|\Delta\eta|$ between two τ-tagged jets;
(7) $p_T^{\tau\tau}$.—the p_T of the $\tau\tau$ dijet; and
(8) E_T^{miss}.—missing transverse energy E_T^{miss}.

These ATLAS features are used as inputs for a BDT to mimic the ATLAS analysis. We also use these ATLAS features to train a dNN, which yields comparable performance to the BDT. For discrimination between $t\bar{t}H$ and other background processes such as $Z+$jets, alternative features are chosen by ATLAS.

[4] This condition was not present in the original analysis of Ref. [3]. The analysis team reported, however, that there are no events in the dataset that contained more than 15 jets.

[5] Following the working point of Ref. [3], this ATLAS card is modified so that τ tagging is at 100% efficiency. This is because we apply a two-τ preselection, so there is no reason to simulate tau finding inefficiency.

For the point cloud architectures, we perform the following preprocessing of the inputs. Let K_{tot} be the scalar sum of the kinematic quantity K for all particles in an event and $\tilde{K}_{\mathrm{tot}}$ be the kinematic quantity K derived from the sum of the four-momenta of all particles in the event. We also define K_i as the kinematic feature for object i in the event. Each event is represented as a set of particles with the following kinematic features representing each particle:

$$\mathrm{Event} = \{\mathrm{particle}_i\} \equiv \left\{ \begin{bmatrix} \log(E_i/\tilde{E}_{\mathrm{tot}}) \\ \log(M_i/M_{\mathrm{tot}}) \\ \eta_i \\ \phi_i \\ \log(p_T^i/p_T^{\mathrm{tot}}) \\ (b_{\mathrm{tag}})_i \\ (\tau_{\mathrm{tag}})_i \end{bmatrix} \right\}. \tag{34}$$

Here, E is energy in the lab frame, M is invariant mass, and

$$b_{\mathrm{tag}}: \ 1 \text{ if particle is } b \text{ tagged and } 0 \text{ if not,}$$
$$\tau_{\mathrm{tag}}: \ 1 \text{ if a particle is } \tau \text{ tagged and } 0 \text{ if not.}$$

We found that taking the logarithm of the dimensionful features improved the training across architectures.

One additional hand-engineered feature we must consider is the ditau invariant mass $M_{\tau\tau}$. This feature is left out of standard ATLAS training but discoverable by the point cloud architectures. Because the ditau mass would peak at the Higgs mass for signal events but not for background events, it is expected to be a good discriminant. In the context of the ATLAS analysis, this feature is intentionally left out so that it can be used as a sanity check on the classification. To get a more complete performance comparison for our case study, we concatenate $M_{\tau\tau}$ with the previously described ATLAS features to be used as input to a BDT. Specifically, we include

(9) $M_{\tau\tau}^{\mathrm{Coll}}$.—the reconstructed ditau invariant mass using the collinear approximation [32–34] to account for the energy carried off by neutrinos.

The ATLAS analysis in Ref. [3] uses the missing mass calculator [33], but we chose to test against the collinear approximation instead for its relative simplicity.

As a cross-check, we train a dNN that takes as input a p_T-sorted and flattened version of the event representation in Eq. (34) with padding to make each event have 15 objects. We call this a *flattened point cloud*. By training a dNN on the flattened point cloud, we can assess whether the improvement in performance we find from the point cloud architectures is truly due to how we structure the architectures rather than just because of an increase in available information.

D. Performance of point cloud architectures

We now compare the performance of our proposed point cloud architectures on the $t\bar{t}H$ versus $t\bar{t}$ event classification problem. The point cloud architectures from Sec. II that we test are

(i) Particlewise, Eq. (10);
(ii) Pairwise, Eq. (11);
(iii) Tripletwise, Eq. (13);
(iv) Nonlinear Pairwise, Eq. (14);
(v) Iterated Nonlinear Pairwise, Eq. (21);
(vi) Nested Concatenation, Eq. (25); and
(vii) Nested Concatenation with Memory, Eq. (30).

The parameters for all architectures are summarized in Appendix B. We chose hyperparameters such that each architecture has approximately 100 000 trainable parameters, to make sure we were comparing architectures based on their structure and not just on model size. For comparison, we test four more traditional architectures:

(i) BDT trained with the ATLAS features;
(ii) BDT trained with the ATLAS features and $M_{\tau\tau}^{\mathrm{Coll}}$;
(iii) dNN trained with the ATLAS features; and
(iv) dNN trained with the flattened point cloud,

where again the parameters are specified in Appendix B.

To assess the classification performance of each architecture, we plot their receiver operator characteristic (ROC) curves in Fig. 3. These curves show the inverse background false-positive rate ($1/\epsilon_b$) as a function of the signal efficiency (ϵ_s), as the cut on the architecture output is varied. The best performing architectures are based on tripletwise or pairwise information. The next-best architectures use the nested concatenation structure from Sec. II F. The point cloud architecture with the weakest performance is the Particlewisie architecture. The dNN acting on the flattened point cloud has significantly worse performance, which indicates the importance of linking the point cloud data inputs to a suitable architecture for processing.

In Table I, we tabulate the number of trainable parameters in each architecture, the area under the ROC curve (AUC), and various performance metrics at the operating points of $\epsilon_s = \{0.3, 0, 7\}$. The choice of $\epsilon_s = 0.7$ mimics the choice made in Ref. [3]. At this operating point, our best performing models (nonlinear pairwise, iterated nonlinear pairwise, tripletwise, and pairwise) achieve a 2.5 times increase in ϵ_s/ϵ_b over the more traditional dNN and BDT models. This jump in performance would be even more pronounced if we choose an operating point of $\epsilon_s = 0.3$, which leads to a nearly fourfold increase in discrimination power.[6] Furthermore, the poor performance of the dNN on the flattened point cloud implies to us that the increase in performance in our architectures comes from their

[6] This operating point is also closer to the one that would maximize $\epsilon_s/\sqrt{\epsilon_b}$, i.e., the one that would yield the largest *significance improvement* [40].

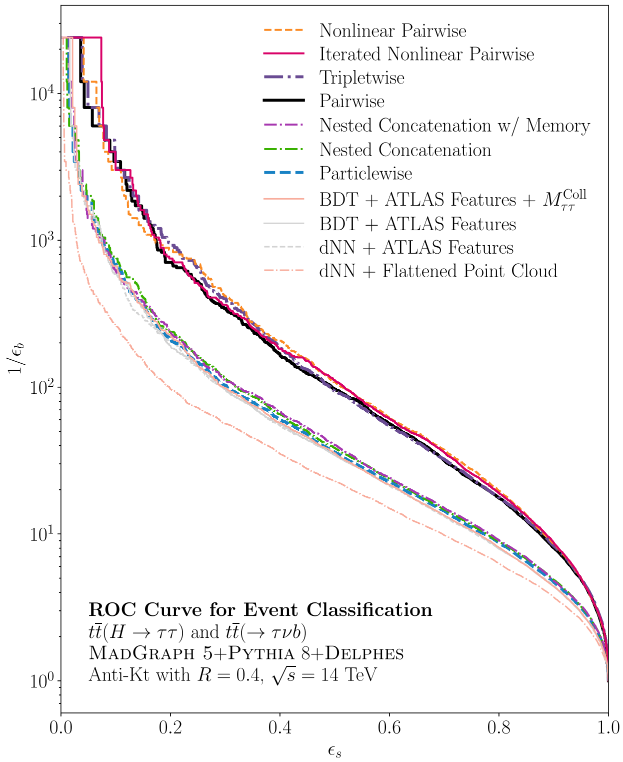


FIG. 3. ROC curves for event classification between $t\bar{t}H$ and $t\bar{t}$, comparing our proposed point cloud architectures to more traditional strategies. The signal efficiency ϵ_s is on the x axis, and the inverse background false positive rate $1/\epsilon_b$ is on the y axis, such that better performance corresponds to curves that are more up and to the right. The Parwise architecture, which is our recommended strategy, is one of the best performing methods for this task.

improved methods of processing information from a collider event and not from simply increasing the amount of information given to an architecture.

The processing of local information via tripletwise or pairwise features yields the most powerful classifiers, but this local information does not need to be processed in overly complex ways. Notably, the architecture which processes information in nearly the simplest manner, the Parwise architecture, is comparable to the performance of architectures with (iterated) nonlinear structures. This implies that more complex strategies to process local information do not necessarily lead to better performance, at least in this context of this event classification problem.

Finally, we note that machine learning architectures trained on Monte Carlo event samples are sensitive to simulation-specific behavior that may affect model performance on real data. For example, the analysis in Ref. [3] did not utilize some input features because they were imperfectly modeled in simulation. In our case study, though, some of these imperfectly modeled variables were used as inputs, so our architectures could be learning unphysical correlations. Thus, the calibrated performance of our architectures on real data will likely be inferior compared to the performance on simulated data, unless some method is used to minimize the dependence on simulation-specific behavior. Despite this caveat, we expect the *relative* performance of the different point cloud strategies to be similar.

E. Computational cost of point cloud architectures

Of the best performing architectures, the Parwise architecture is the most computationally efficient. To give an idea of the computational cost of our architectures,

TABLE I. Performance summary of the studied architectures. Here we tabulate (i) the area under the ROC curve, (ii) the number of trainable parameters, (iii) the inverse false-positive rate $1/\epsilon_b$ at fixed signal efficiency $\epsilon_s = \{0.3, 0.7\}$, which roughly corresponds to how many background events we see until a background event is misclassified as a signal event, and (iv) the ratio of signal efficiency to background false-positive rate ϵ_s/ϵ_b at fixed signal efficiency $\epsilon_s = \{0.3, 0.7\}$, which measures the discriminatory power of the architecture. By comparing the average of the values in the $\epsilon_s = 0.7$ columns of our best performing architectures (nonlinear pairwise, iterated nonlinear pairwise, tripletwise, and pairwise) to the best performing traditional architecture (BDT + ATLAS features + $M_\tau^{\rm Coll}$), we see that we gain about a 2.5 times increase in performance. Boldfaced entries are the best performing in each column.

			($\epsilon_s = 0.3$)		($\epsilon_s = 0.7$)	
Architecture	AUC	# Params	$1/\epsilon_b$	ϵ_s/ϵ_b	$1/\epsilon_b$	ϵ_s/ϵ_b
Nonlinear pairwise, Eq. (14)	**0.9590**	**96 K**	**414.4**	**124.1**	**37.4**	**26.2**
Iterated nonlinear pairwise, Eq. (21)	**0.9590**	100 K	364.2	109.2	35.9	25.2
Tripletwise, Eq. (13)	0.9578	105 K	400.6	120.1	32.3	22.6
Pairwise, Eq. (11)	0.9568	104 K	343.4	102.3	31.7	22.2
Nested Concatenation with Memory, Eq. (30)	0.9285	100 K	111.8	33.5	15.1	10.6
Nested concatenation, Eq. (25)	0.9277	105 K	116.1	34.7	14.6	10.2
Particlewise, Eq. (10)	0.9253	100 K	102.7	30.8	14.4	10.1
BDT + ATLAS features + $M_{\tau\tau}^{\rm Coll}$	0.9206	⋯	98.5	29.5	13.4	9.4
BDT + ATLAS features	0.9201	⋯	96.9	28.9	13.3	9.3
dNN + ATLAS features	0.9198	134 K	99.3	29.6	13.2	9.2
dNN + flattened point cloud	0.9015	160 K	56.3	16.9	9.8	6.9

TABLE II. Computational cost summary of the studied architectures. Here, we tabulate the time per epoch and total number of epochs it took to train each model on a server equipped with two NVIDIA Tesla K80s. While training, we reserve 30% of the training data as validation data and monitored validation loss. If validation loss has not improved in 32 consecutive epochs, we stop the training and restore the weights of the model to the point where validation loss was lowest. Boldfaced entries are the most computationally efficient in each column.

Architecture	Time/epoch (seconds)	Total # of epochs
Tripletwise	240	100
Pairwise	25	93
Nonlinear (NL) pairwise	27	88
Iterated NL pairwise	50	149
Nested concatenation	20	64
Nested Concatenation with Memory	20	**54**
Particlewise	**10**	61
dNN + ATLAS features	10	110
dNN + flattened point cloud	**5**	**38**

we tabulate the approximate time or epoch and total number of epochs to train each architecture in Table II. These training times are obtained on a server equipped with two NVIDIA Tesla K80s. The Parwise architecture and nonlinear Parwise architectures are close in both performance and run-time efficiency. The other two architectures that achieve similar performance, the Iterated Nonlinear and Tripletwise architectures, are significantly more computationally expensive.

Since the Parwise architecture is nearly the best performing architecture while still being one of the simplest conceptually and most efficient computationally, we recommend the use of the Parwise architecture for event classification. For other point cloud tasks that have seen gains from using graph neural networks, we recommend their performance be benchmarked against the Parwise architecture.

F. Latent dimension studies of the Parwise architecture

Having convinced ourselves that the Parwise architecture is a prime candidate for event classification, we now study the ℓ-dimensional latent representation of events generated by this architecture. Specifically, we study what happens if we restrict the latent dimension ℓ of the Parwise architecture. This gives us a way to study how powerful this architecture is at identifying useful discriminatory features from the event kinematics. Furthermore, we can visualize the latent representations to get some picture of what the architecture is physically learning.

The latent dimension ℓ corresponds to the number of discriminatory features the architecture can extract from the point cloud. Concretely, if we restrict the latent dimension of the Parwise architecture to $\ell = 2$, we are essentially asking the architecture to extract two features from the point cloud that, when processed by the F function, can robustly distinguish between signal and background events. In the Parwise architecture from Eq. (11), the latent representation of an event $\mathcal{X}$ is the result of a double summation over pairs of particles: $\sum_{i,j} \Phi_2(x_i, x_j) \in \mathbb{R}^{\ell}$. In Fig. 3 and Table I above, we used $\ell = 2^6$, as described in Appendix B. We can think of the BDTs and dNNs trained on the ATLAS variables [3] as having $\ell = 2^3$, because they take as input eight hand-engineered features, as described in Sec. III C.

As shown in Table III, we can significantly restrict the size of the latent dimension of our Parwise architecture and still outperform traditional methods of event classification. This table shows the performance of the Parwise architecture as a function of the size of the latent dimension. The strong performance even for $\ell = 2^0 = 1$ implies that deep-learning-driven feature engineering is extremely powerful for finding robust discriminatory features for event classification. These discriminatory features could, in principle, be found from the traditional feature engineering game, but we expect they would be extremely difficult to find in practice without the use of machine learning.

TABLE III. Impact of latent dimension size on the performance of the Parwise architecture. We show the same performance metrics as Table I, and the uncertainties correspond to the mean and variance from eight random initializations. Boldfaced entries are the best performing in each column. Even for a latent dimension size of $2^0 = 1$, the Parwise architecture outperforms the baseline ATLAS method.

			$(\epsilon_s = 0.3)$		$(\epsilon_s = 0.7)$	
Latent dimension	AUC	# Params	$1/\epsilon_b$	ϵ_s/ϵ_b	$1/\epsilon_b$	ϵ_s/ϵ_b
2^0	0.9428 ± 0.0025	92 K	170.1 ± 10.2	50.8 ± 3.0	21.5 ± 1.4	15.1 ± 1.0
2^1	0.9452 ± 0.0013	92 K	205.1 ± 15.7	61.3 ± 4.7	22.7 ± 0.9	15.9 ± 0.7
2^2	0.9495 ± 0.0022	93 K	270.2 ± 38.6	80.8 ± 11.5	26.0 ± 1.8	18.2 ± 1.2
2^3	0.9526 ± 0.0020	94 K	326.8 ± 45.6	97.6 ± 13.2	29.2 ± 2.0	20.4 ± 1.4
2^4	0.9550 ± 0.0010	95 K	334.9 ± 23.2	100.1 ± 6.9	31.2 ± 1.6	21.8 ± 1.1
2^5	0.9565 ± 0.0011	98 K	358.0 ± 42.6	106.5 ± 12.6	32.8 ± 1.5	23.0 ± 1.0
2^6	0.9581 ± 0.0006	104 K	382.9 ± 37.8	114.4 ± 11.4	34.5 ± 1.4	24.2 ± 1.0
2^7	$\mathbf{0.9584 \pm 0.0010}$	117 K	$\mathbf{402.8 \pm 43.5}$	$\mathbf{120.7 \pm 13.0}$	$\mathbf{34.9 \pm 1.4}$	$\mathbf{24.4 \pm 1.0}$

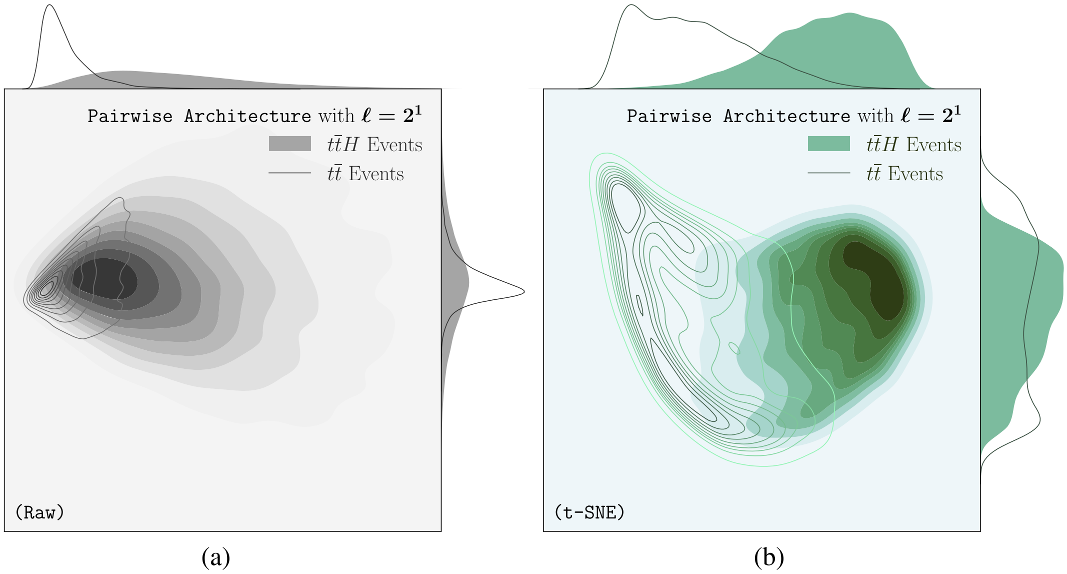


FIG. 4. Visualization of the latent representations of events in the $\ell = 2^1$ Parwise architecture, using KDE for density estimation. In (a), we plot the latent representations corresponding to the $t\bar{t}H$ and $t\bar{t}$ events. In (b), we apply t-SNE to disentangle the two distributions and see a clearer separation. Each plot also shows the marginalized distributions along each axis.

For $\ell = 2$, we can directly visualize the latent space of the Parwise architecture, as shown in Fig. 4. Here we plot two versions of the latent dimension, where densities are approximated with kernel density estimation (KDE) [41,42].[7] In Fig. 4(a), we plot the densities of the raw latent representations. We see that one of the latent space variables yields a clean separation between signal and background events, while the other one yields an approximately Gaussian distributed feature with different widths for signal and background. In Fig. 4(b), we apply t-distributed stochastic neighbor embedding (t-SNE) [43–46] to the latent representation, which shows more clearly the separation between signal and background events. This t-SNE visualization serves as a reference for later plots with higher ℓ.

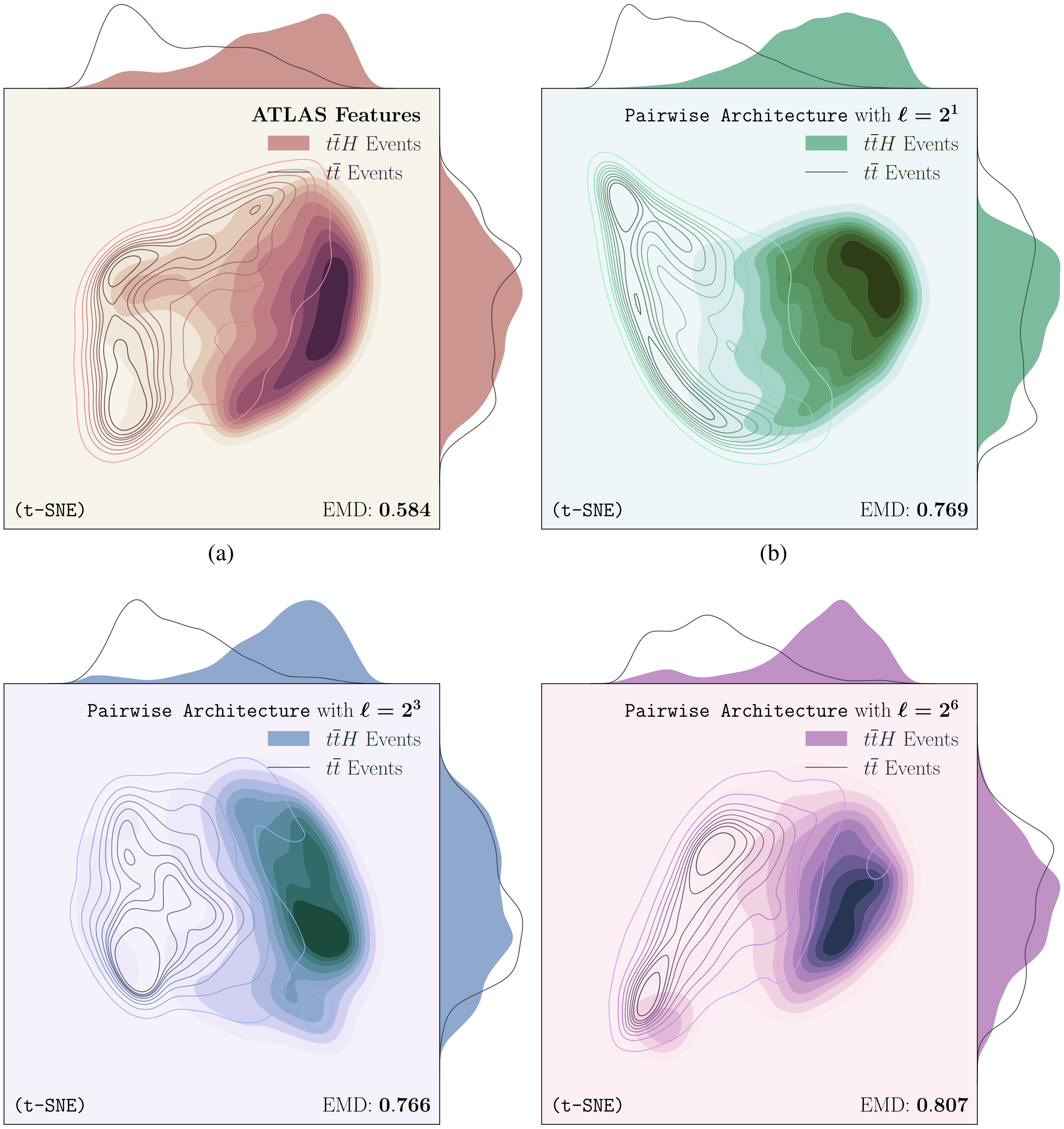
As we increase the latent dimension ℓ, the trained architectures identify more discriminatory features, leading to better separation between signal and background events in the latent space. In Fig. 5, we plot the two-dimensional t-SNE embeddings of the eight-dimensional ATLAS feature space and compare it to our $\ell = \{2^1, 2^3, 2^6\}$ Parwise architectures. To quantify the separation between the two (t-SNE projected) distributions, we approximate the Earth mover's distance (EMD) [47–49] using the Euclidean distance as the ground metric between the $t\bar{t}H$ and $t\bar{t}$ distributions. Since the length scales within a t-SNE embedding are not physical and we wish for the EMD to be a meaningful metric of comparison, we standardize the whole distribution of events in each plot so that along each dimension we have zero mean and unit variance. Qualitatively, we see that, for our proposed Parwise architecture, the joint and marginalized distributions of $t\bar{t}H$ and $t\bar{t}$ are more clearly separated than for the ATLAS features. This observation is reinforced quantitatively by seeing that the EMD between the embedded distribution of signal and background events is smallest for the ATLAS features, implying the largest degree of overlap.

G. Correlations between Parwise architecture features and hand-engineered features

The features found to be useful to represent collider events by our Parwise architecture are correlated to several of the features chosen by ATLAS to classify events. In Fig. 6, we tabulate the Spearman's rank correlation coefficients (Spearman's ρ) between the ATLAS features used in Ref. [3] (see Sec. III C) and the learned latent features of our Parwise architecture with $\ell = 2$ and $\ell = 8$. Spearman's ρ quantifies the degree to which two variables are monotonically related, with $\rho = 0$ meaning no correlation and $\rho = 1$ meaning perfect correlation. Unlike the more common Pearson correlation coefficient, Spearman's ρ does not care if the relationship is linear or not, which makes it a more robust notion of correlation in the context of nonlinear neural networks.

Because Spearman's ρ quantifies monotonic relations, we have to be mindful of features whose classification performance is related to how close they are to the true W-boson mass ($m_W = 80.4$ GeV), top quark mass ($m_t = 172.5$ GeV), and Higgs boson mass ($m_H = 125$ GeV). Thus, instead of tabulating correlations for $M_{\hat{W}}$, $M_{\hat{t}}$, and $M_{\tau\tau}^{\mathrm{Coll}}$, we look at $|M_{\hat{W}} - m_W|$, $|M_{\hat{t}} - m_t|$, and $|M_{\tau\tau}^{\mathrm{Coll}} - m_H|$. We see that,

[7] Because of computational restrictions, we generate the figures in this subsection using roughly a third of the events from the testing set.



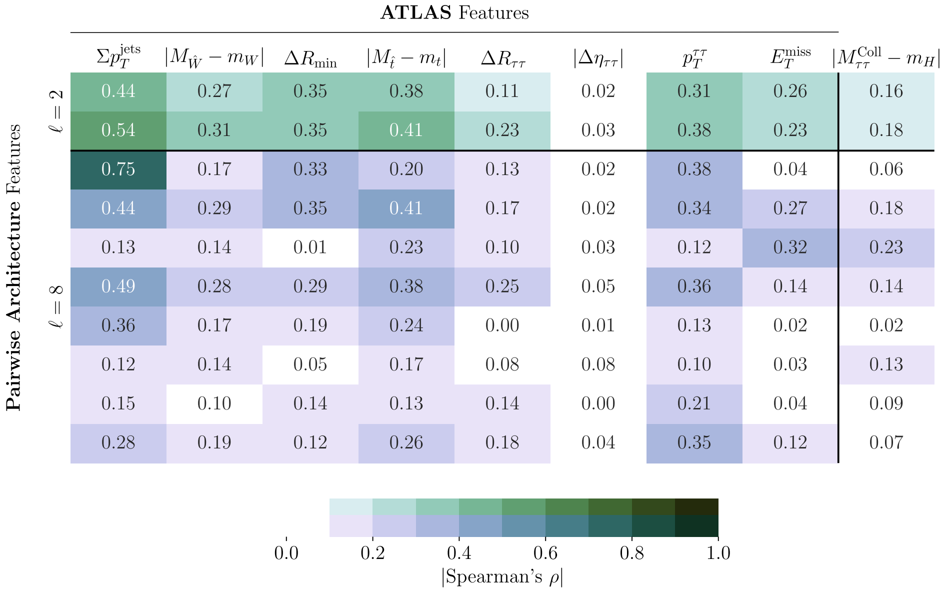
(a) (b)

(c) (d)

FIG. 5. Visualization of the latent representations using t-SNE embedding. Shown are (a) the ATLAS features which effectively have $\ell = 2^3$, (b) the Parwise architecture with $\ell = 2^1$, which has the same information as Fig. 4(b), (c) the Parwise architecture with $\ell = 2^3$, and (d) the Parwise architecture with $\ell = 2^6$. The t-SNE embeddings have been standardized such that the distributions have mean 0 and standard deviation 1 along both dimensions. The standardized embedding is then rotated such that the $t\bar{t}H$ events are centered on the right side of the figure. For each plot, we report the EMD between the distribution of $t\bar{t}H$ and $t\bar{t}$ events, which roughly measures the separation of the two distributions, with larger EMD corresponding to better separation. We also plot marginalized densities along each axis.

of all the ATLAS features, the scalar sum of all jet p^T is most closely related to the Parwise architecture features. Other ATLAS features that have some correlation with the Parwise architecture features are the smallest ΔR between two jets, the p^T of the $\tau\tau$ dijet, the ΔR between the two τ-tagged jets, and the invariant mass of the dijet or trijet with invariant mass closest to the W-boson or top quark mass.

From Fig. 6, we can see that there exists a correlation between $M_{\tau\tau}^{\mathrm{Coll}}$ and the Parwise architecture features, as anticipated in Sec. III C. This correlation is subtle, though, and not captured by a single latent space feature. In Fig. 7, we consider the Parwise architecture with $\ell = 2^6$ and plot the $M_{\tau\tau}^{\mathrm{Coll}}$ distribution for $t\bar{t}H$ events correctly and incorrectly classified by this architecture at fixed signal efficiency $\epsilon_s = 0.7$. We see that correctly labeled events have a sharper peak in $M_{\tau\tau}^{\mathrm{Coll}}$, albeit shifted a bit to the right of m_H. This behavior suggests that the Parwise architecture with $\ell = 2^6$, which was used for our comparison comparison in Sec. III D, has learned features with some correlation to $M_{\tau\tau}^{\mathrm{Coll}}$.



	$\Sigma p_T^{\rm jets}$	$\lvert M_{\hat{W}} - m_W \rvert$	$\Delta R_{\rm min}$	$\lvert M_{\hat{t}} - m_t \rvert$	$\Delta R_{\tau\tau}$	$\lvert \Delta\eta_{\tau\tau} \rvert$	$p_T^{\tau\tau}$	$E_T^{\rm miss}$	$\lvert M_{\tau\tau}^{\rm Coll} - m_H \rvert$
$\ell = 2$	0.44	0.27	0.35	0.38	0.11	0.02	0.31	0.26	0.16
	0.54	0.31	0.35	0.41	0.23	0.03	0.38	0.23	0.18
$\ell = 8$	0.75	0.17	0.33	0.20	0.13	0.02	0.38	0.04	0.06
	0.44	0.29	0.35	0.41	0.17	0.02	0.34	0.27	0.18
	0.13	0.14	0.01	0.23	0.10	0.03	0.12	0.32	0.23
	0.49	0.28	0.29	0.38	0.25	0.05	0.36	0.14	0.14
	0.36	0.17	0.19	0.24	0.00	0.01	0.13	0.02	0.02
	0.12	0.14	0.05	0.17	0.08	0.08	0.10	0.03	0.13
	0.15	0.10	0.14	0.13	0.14	0.00	0.21	0.04	0.09
	0.28	0.19	0.12	0.26	0.18	0.04	0.35	0.12	0.07

FIG. 6. The Spearman's rank correlation coefficients of the ATLAS features from Ref. [3] (left block) and ditau mass (right block) with the latent features learned by our Parwise architectures, using $\ell = 2$ (top block) and $\ell = 8$ (bottom block). Here, $m_W = 80.4$ GeV, $m_t = 172.5$ GeV, and $m_H = 125$ GeV. We compute the ρ between the Parwise architecture feature and the absolute difference of $M_{\hat{W}}$, $M_{\hat{t}}$, and $M_{\tau\tau}$ to canonical values, since the "goodness" of these mass variables is monotonic with how close these variables are to the true W-boson, top quark, and Higgs mass. Of all the chosen ATLAS features, the scalar sum of the jets' transverse momenta is particularly correlated with the features used by the Parwise architecture. Also noteworthy is the presence of a correlation between $M_{\tau\tau}^{\rm Coll}$ and the pairwise latent features.

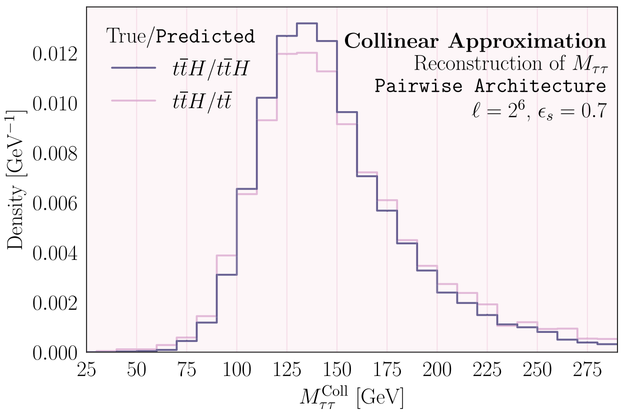


FIG. 7. Normalized distributions of $M_{\tau\tau}^{\rm Coll}$ for $t\bar{t}H$ signal events. Predicted labels are from the Parwise architecture with $\ell = 2^6$ at a fixed signal efficiency of $\epsilon_s = 0.7$, where the purple curves correspond to $t\bar{t}H$-labeled events and the pink curves correspond to $t\bar{t}$-labeled events. Events classified as $t\bar{t}H$ have a sharper $M_{\tau\tau}^{\rm Coll}$ feature, suggesting that the Parwise architecture has learned features with some correlation to $M_{\tau\tau}^{\rm Coll}$.

IV. CONCLUSIONS

In this paper, we compared neural network architectures defined on the point cloud representation of collider events with traditional approaches for event classification. The point cloud representation allows us to circumvent many difficulties arising from trying to robustly represent an event with a fixed-size input. Our architectures explore three complementary strategies to process information within a point cloud: (i) using multiple summations to improve local information processing; (ii) using iterated convolutions to increase an architecture's power to build latent representations; and (iii) using nested concatenation of global features to improve global information processing. These can be viewed as simplified versions of the strategies used to build graph neural networks.

To benchmark our architectures, we performed a case study of event classification in the $H \to \tau\tau$ channel and compared the results to an ATLAS study using hand-engineered features [3]. At a comparable signal efficiency operating point to the one used by ATLAS, we found a 2.5 times increase in background rejection. This gain in performance was not simply due to the increased size of the input space. Indeed, when the flattened point cloud representation was processed with a dNN, we found worse performance than for the ATLAS baseline. We therefore recommend further explorations of point cloud architectures for event classification problems.

Among the tested architectures, the Parwise architecture exhibited the best balance of classification performance, computationally efficiency, and conceptual simplicity. The Particlewisie architecture, based on a straightforward application of the Deep Set formalism [11], yielded performance similar to the ATLAS baseline. By considering the set of pairs of particles, we found a boost in performance without requiring more complex nonlinear or iterated structures as in EdgeConv architectures [13]. The

Parwise architecture continues to have good discriminatory power as the latent space dimension ℓ is decreased, and, by visualizing the learned latent representations, we conclude that the Parwise architecture is able to identify discriminatory features that are well suited for event classification. We found that these learned features are correlated with traditionally chosen handcrafted features. While more complex graph neural networks might provide better performance for certain eventwide tasks, we recommend that they be benchmarked against the simpler Parwise architecture.

A key open question regarding our proposed architectures is how they will perform as the final state multiplicity increases. In this paper, we considered a final state of $O(10)$ objects, but final states of interest with $O(100)$ or even $O(1000)$ objects also appear in particle physics applications such as jet substructure studies. We found that, for our case of $O(10)$ final state objects, the Parwise architecture achieved the best balance between performance and cost. As we scale up to more final state objects, however, it will be important to understand the performance-cost balance, as Parwise architectures scale quadratically with the number of objects being studied. To extend our architectures to more final state objects without suffering from quadratic scaling, one could follow the strategy of the original EdgeConv construction [13] and consider only k-nearest neighbors instead of all pairs. These trade-offs are an important area for future studies.

There are several more directions for further explorations. First, our case study focused on binary classification, but, as described in Eq. (2), these point cloud architectures could be applied to multicategory classification or regression. Second, the Parwise architecture is based on learning a generic function Φ_2, but it may be possible to improve performance and interpretability by restricting its functional form. Finally, there has been a rising interest in incorporating physical symmetries into neural networks. While our point cloud architectures already exhibit manifest permutation invariance among the particles, event classification could benefit from directly incorporating Lorentz symmetry [50–55] or infrared and collinear safety [12,15,22,56–59].

ACKNOWLEDGMENTS

J. T. thanks Patrick Komiske, Eric Metodiev, Nishat Protyasha, and Raymond Wynne for collaboration on unpublished work related to alternative point cloud architectures. D. S. and P. O. were supported by the U.S. Department of Energy, Office of Science, Office of High Energy Physics, under Grant No. DE-SC0007890. J. T. is supported by the U.S. Department of Energy (DOE) Office of High Energy Physics under Grant No. DE-SC0012567 and by the National Science Foundation under Cooperative Agreement No. PHY-2019786 (The NSF AI Institute for Artificial Intelligence and Fundamental Interactions).

APPENDIX A: PERMUTATION EQUIVARIANT DEEP SETS

In Sec. II F, we argued that concatenation (with or without memory) is a powerful way to incorporate global information into local particle processing. An alternative additive approach to incorporating global information was presented in Ref. [11] and applied in Ref. [15]. In this appendix, we show that the additive approach can be viewed as a special case of the concatenation approach with memory.

A permutation equivariant Deep Set that maps n-dimensional point clouds to ℓ-dimensional point clouds can be written as follows [11]:

$$\Omega(x_i, \mathcal{X}) = \sigma(x_i \boldsymbol{\Lambda} - \hat{g}(\mathcal{X})\boldsymbol{\Gamma}). \tag{A1}$$

Here, σ is some activation function, $\hat{g}(\mathcal{X}) \in \mathbb{R}^{1\times n}$ is the result of a symmetric aggregation operation over the point cloud, $\boldsymbol{\Lambda}$ is a transformation of the features of particle $x_i \in \mathbb{R}^{1\times n}$, and $\boldsymbol{\Gamma} \in \mathbb{R}^{n\times \ell}$ is a transformation of the pooled features. One can iteratively apply these equivariant Deep Set layers to create a permutation equivariant architecture, which was shown to be universal for permutation equivariant functions in Ref. [60].

We now show that a single permutation equivariant Deep Set layer from Eq. (A1) is a special case of the concatenation approach from Eq. (29). First, since we are looking at one layer, we can take

$$\mathcal{X}^{(d-1)} \equiv \mathcal{X}. \tag{A2}$$

Since Deep Sets are universal for permutation invariant functions and $\hat{g}$ is by definition permutation invariant, we can assume that

$$f^{(d-1)}(\mathcal{X}^{(d-1)}) \equiv \hat{g}(\mathcal{X}). \tag{A3}$$

Now consider implementing $\Phi^{(d)}$ with a one-layer neural network with the same activation function σ as in Eq. (A1) and weights $\mathbf{w}^{(d)} \in \mathbb{R}^{2n\times \ell}$. With these assumptions, Eq. (29) reduces to

$$\begin{aligned} x_i^{(d)} &= \Phi^{(d)}\left(x_i^{(d-1)} \oplus f^{(d-1)}\left(\mathcal{X}^{(d-1)}\right)\right) \\ &= \sigma\left(\left(x_i^{(d-1)} \oplus \hat{g}(\mathcal{X})\right)\mathbf{w}^{(d)}\right). \end{aligned} \tag{A4}$$

Since we are multiplying the concatenation of two vectors with the weight matrix $\mathbf{w}^{(d)}$, we can decompose the weight matrix as

$$\mathbf{w}^{(d)} = \begin{bmatrix} \boldsymbol{\lambda}^{(d)} \\ \boldsymbol{\gamma}^{(d)} \end{bmatrix}, \qquad \boldsymbol{\lambda}^{(d)}, \boldsymbol{\gamma}^{(d)} \in \mathbb{R}^{n,\ell}. \tag{A5}$$

This reduces Eq. (A4) to

$$x_i^{(d)} = \sigma\left(x_i^{(d-1)}\boldsymbol{\lambda}^{(d)} + \hat{g}(\mathcal{X})\boldsymbol{\gamma}^{(d)}\right). \tag{A6}$$

Finally, if we choose $\boldsymbol{\lambda}^{(d)} = \boldsymbol{\Lambda}$ and $\boldsymbol{\gamma}^{(d)} = -\boldsymbol{\Gamma}$, then Eq. (A6) is equivalent to Eq. (A1).

We therefore conclude that a single permutation equivariant Deep Set layer is a special case of the concatenation approach. This, in turn, means that the composition of L-equivariant Deep Set layers from Eq. (A1) is equivalent to the resulting point cloud after L-nested layers of Eq. (29). Thus, the architecture in Ref. [11] based on iteratively applied equivariant Deep Set layers is a special case of our Nested Concatenation with Memory architecture.

APPENDIX B: MODEL PARAMETERS

In this appendix, we specify the model parameters used for the event classification study in Sec. III D. All architectures were implemented with Keras [61] using the TensorFlow [62] back end. We chose model parameters so that the total number of trainable parameters in each model was roughly the same across all architectures.

(1) Particlewisie architecture, Eq. (3):
   (a) F.—four layers, each 128 nodes wide;
   (b) Φ.—four layers with widths (128, 128, 128, 64).
(2) Parwise architecture, Eq. (11):
   (a) F.—five layers, each 64 nodes wide;
   (b) Φ.—five layers with widths (64, 128, 256, 128, 64).
(3) Tripletwise architecture, Eq. (13)):
   (a) F.—five layers, each 64 nodes wide;
   (b) Φ.—five layers with widths (64, 128, 256, 128, 64).
(4) Nonlinear Parwise architecture, Eq. (14)):
   (a) F.—five layers, each 32 nodes wide;
   (b) Φ.—five layers with widths (64, 128, 256, 128, 64);
   (c) Π.—three layers, each 32 nodes wide.
(5) Iterated nonlinear Parwise architecture, Eq. (21):
   (a) L.—Since we found that performance does not improve for $L > 3$, we use $L = 3$;
   (b) F.—five layers, each 32 nodes wide;
   (c) $\Pi^{(i)}$.—All have three layers, each 32 nodes wide;
   (d) $\Phi^{(i)}$.—All have five layers with widths (64, 64, 116, 64, 64).
(6) Nested Concatenation architecture, Eq. (25):
   (a) L.—Since we found that performance does not improve for $L > 2$, we use $L = 2$;
   (b) $F^{(i)}$.—All have four layers, each 70 nodes wide;
   (c) $\Phi^{(i)}$.—For $i \neq L$, the $\Phi^{(i)}$ have three layers, each 70 nodes wide; for $i = L$, $\Phi^{(L)}$ has four layers with widths (70, 70, 70, 64).
(7) Nested Concatenation with Memory architecture, Eq. (30):
   (a) L.—Since we found that performance does not improve for $L > 2$, we use $L = 2$;
   (b) $F^{(i)}$.—All have three layers, each 68 nodes wide;
   (c) $\Phi^{(i)}$.—For $i \neq L$, the $\Phi^{(i)}$ have three layers, each 68 nodes wide; for $i = L$, $\Phi^{(L)}$ has four layers with widths (68, 68, 68, 64).
(8) Dense neural network (ATLAS or naive features):
   (a) We implement dNNs with a batch normalization layer [63] followed by three layers, each 256 nodes wide.

We use leaky_relu activation functions between each layer in all of our neural networks. A two-unit layer followed by a softmax activation function is used as the output layer in all models. To train, we minimize categorical cross entropy using the Adam optimization algorithm [64] with the AMSGrad enhancement [65]. When training, we reserve 30% of the training data as validation data and monitor the validation loss. To avoid overfitting, we stop training if validation loss has not improved in 32 epochs and restore the weights of the model to the point when the validation loss was lowest.

The following BDT parameters were used in our model implemented with XGBoost [66]. These parameters were chosen using the hyperparameter tuning library Hyperopt [67] with the tree of Parzen estimators algorithm [68].

(i) colsample_bytree: 0.703;
(ii) eta: 0.35;
(iii) gamma: 1.02;
(iv) max_depth: 10;
(v) min_child_weight: 10;
(vi) n_estimators: 297;
(vii) reg_alpha: 40.0; and
(viii) reg_lambda: 0.987.

Again, we softmax the output and optimize categorical cross entropy.

[1] CMS Collaboration, Measurement of $\mathrm{t\bar{t}H}$ production in the $\mathrm{H} \rightarrow \mathrm{b\bar{b}}$ decay channel in 41.5 fb^{-1} of proton-proton collision data at $\sqrt{s} = 13$ TeV, https://cds.cern.ch/record/2675023/files/HIG-18-030-pas.pdf.

[2] Albert M. Sirunyan *et al.* (CMS Collaboration), Search for $\mathrm{t\bar{t}H}$ production in the $\mathrm{H} \rightarrow \mathrm{b\bar{b}}$ decay channel with leptonic $\mathrm{t\bar{t}}$ decays in proton-proton collisions at $\sqrt{s} = 13$ TeV, J. High Energy Phys. 03 (2019) 026.

[3] Georges Aad *et al.* (ATLAS Collaboration), Measurements of Higgs boson production cross-sections in the $H \to \tau^+\tau^-$ decay channel in pp collisions at $\sqrt{s} = 13$ TeV with the ATLAS detector, J. High Energy Phys. 08 (2022) 175.

[4] Georges Aad *et al.* (ATLAS Collaboration), Test of CP invariance in vector-boson fusion production of the Higgs boson in the H → ττ channel in proton–proton collisions at s = 13 TeV with the ATLAS detector, Phys. Lett. B **805**, 135426 (2020).

[5] Morad Aaboud *et al.* (ATLAS Collaboration), Cross-section measurements of the Higgs boson decaying into a pair of τ-leptons in proton-proton collisions at $\sqrt{s} = 13$ TeV with the ATLAS detector, Phys. Rev. D **99**, 072001 (2019).

[6] M. Aaboud *et al.* (ATLAS Collaboration), Evidence for the $H \to b\bar{b}$ decay with the ATLAS detector, J. High Energy Phys. 12 (2017) 024.

[7] Morad Aaboud *et al.* (ATLAS Collaboration), Observation of $H \to b\bar{b}$ decays and VH production with the ATLAS detector, Phys. Lett. B **786**, 59 (2018).

[8] Morad Aaboud *et al.* (ATLAS Collaboration), Search for the standard model Higgs boson produced in association with top quarks and decaying into a $b\bar{b}$ pair in pp collisions at $\sqrt{s} = 13$ TeV with the ATLAS detector, Phys. Rev. D **97**, 072016 (2018).

[9] Georges Aad *et al.* (ATLAS Collaboration), Measurements of WH and ZH production in the $H \to b\bar{b}$ decay channel in pp collisions at 13 TeV with the ATLAS detector, Eur. Phys. J. C **81**, 178 (2021).

[10] Georges Aad *et al.* (ATLAS Collaboration), Search for the $b\bar{b}$ decay of the standard model Higgs boson in associated $(W/Z)H$ production with the ATLAS detector, J. High Energy Phys. 01 (2015) 069.

[11] Manzil Zaheer, Satwik Kottur, Siamak Ravanbhakhsh, Barnabás Póczos, Ruslan Salakhutdinov, and Alexander J. Smola, Deep sets, arXiv:1703.06114.

[12] Patrick T. Komiske, Eric M. Metodiev, and Jesse Thaler, Energy flow networks: Deep sets for particle jets, J. High Energy Phys. 01 (2019) 121.

[13] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon, Dynamic graph CNN for learning on point clouds, arXiv:1801.07829.

[14] Huilin Qu and Loukas Gouskos, ParticleNet: Jet tagging via particle clouds, Phys. Rev. D **101**, 056019 (2020).

[15] Matthew J. Dolan and Ayodele Ore, Equivariant energy flow networks for jet tagging, Phys. Rev. D **103**, 074022 (2021).

[16] Serguei Chatrchyan *et al.* (CMS Collaboration), Evidence for the 125 GeV Higgs boson decaying to a pair of τ leptons, J. High Energy Phys. 05 (2014) 104.

[17] Albert M. Sirunyan *et al.* (CMS Collaboration), Observation of the Higgs boson decay to a pair of τ leptons with the CMS detector, Phys. Lett. B **779**, 283 (2018).

[18] Armen Tumasyan *et al.* (CMS Collaboration), Measurement of the Inclusive and Differential Higgs Boson Production Cross Sections in the Decay Mode to a Pair of τ Leptons in pp Collisions at $\sqrt{s} = 13$ TeV, Phys. Rev. Lett. **128**, 081805 (2022).

[19] CMS Collaboration, Measurements of Higgs boson production in the decay channel with a pair of τ leptons in proton-proton collisions at $\sqrt{s} = 13$ TeV, arXiv:2204.12957.

[20] Eric A. Moreno, Olmo Cerri, Javier M. Duarte, Harvey B. Newman, Thong Q. Nguyen, Avikar Periwal, Maurizio Pierini, Aidana Serikova, Maria Spiropulu, and Jean-Roch Vlimant, JEDI-net: A jet identification algorithm based on interaction networks, Eur. Phys. J. C **80**, 58 (2020).

[21] Vinicius Mikuni and Florencia Canelli, ABCNet: An attention-based method for particle tagging, Eur. Phys. J. Plus **135**, 463 (2020).

[22] Amit Chakraborty, Sung Hak Lim, Mihoko M. Nojiri, and Michihisa Takeuchi, Neural network-based top tagger with two-point energy correlations and geometry of soft emissions, J. High Energy Phys. 07 (2020) 111.

[23] Jonathan Shlomi, Sanmay Ganguly, Eilam Gross, Kyle Cranmer, Yaron Lipman, Hadar Serviansky, Haggai Maron, and Nimrod Segol, Secondary vertex finding in jets with neural networks, Eur. Phys. J. C **81**, 540 (2021).

[24] Jonathan Shlomi, Peter Battaglia, and Jean-Roch Vlimant, Graph neural networks in particle physics, Mach. Learn. **2**, 021001 (2021).

[25] Javier Duarte and Jean-Roch Vlimant, Graph neural networks for particle tracking and reconstruction, in *Artificial Intelligence for High Energy Physics* (World Scientific, Singapore, 2022).

[26] Savannah Thais, Paolo Calafiura, Grigorios Chachamis, Gage DeZoort, Javier Duarte, Sanmay Ganguly, Michael Kagan, Daniel Murnane, Mark S. Neubauer, and Kazuhiro Terao, Graph neural networks in particle physics: Implementations, innovations, and challenges, in *Proceedings of the 2022 Snowmass Summer Study* (2022), arXiv:2203.12852.

[27] Delon Shen, Classifying collider events with point clouds, https://github.com/DelonShen/classifying-collider-events-with-point-clouds (2022).

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

[29] A. Djouadi, J. Kalinowski, and M. Spira, HDECAY: A program for Higgs boson decays in the standard model and its supersymmetric extension, Comput. Phys. Commun. **108**, 56 (1998).

[30] D. de Florian *et al.* (LHC Higgs Cross Section Working Group), *Handbook of LHC Higgs Cross Sections: 4. Deciphering the Nature of the Higgs Sector* (CERN, Geneva, 2017), Vol. 2.

[31] Albert M. Sirunyan *et al.* (CMS Collaboration), Measurement of the Higgs boson production rate in association with top quarks in final states with electrons, muons, and hadronically decaying tau leptons at $\sqrt{s} = 13$ TeV, Eur. Phys. J. C **81**, 378 (2021).

[32] R. Keith Ellis, I. Hinchliffe, M. Soldate, and J. J. van der Bij, Higgs decay to tau+ tau-: A possible signature of intermediate mass Higgs bosons at the SSC, Nucl. Phys. **B297**, 221 (1988).

[33] A. Elagin, P. Murat, A. Pranko, and A. Safonov, A new mass reconstruction technique for resonances decaying to di-tau, Nucl. Instrum. Methods Phys. Res., Sect. A **654**, 481 (2011).

[34] Partha Konar and Abhaya Kumar Swain, Reconstructing semi-invisible events in resonant tau pair production from Higgs, Phys. Lett. B **757**, 211 (2016).

[35] Johan Alwall, Michel Herquet, Fabio Maltoni, Olivier Mattelaer, and Tim Stelzer, MadGraph 5: Going beyond, J. High Energy Phys. 06 (2011) 128.

[36] Torbjörn Sjöstrand, Stefan Ask, Jesper R. Christiansen, Richard Corke, Nishita Desai, Philip Ilten, Stephen Mrenna, Stefan Prestel, Christine O. Rasmussen, and Peter Z. Skands, An introduction to PYTHIA 8.2, Comput. Phys. Commun. **191**, 159 (2015).

[37] J. de Favereau, C. Delaere, P. Demin, A. Giammanco, V. Lemaître, A. Mertens, and M. Selvaggi (DELPHES 3 Collaboration), DELPHES 3, A modular framework for fast simulation of a generic collider experiment, J. High Energy Phys. 02 (2014) 057.

[38] Matteo Cacciari, Gavin P. Salam, and Gregory Soyez, The anti-k_t jet clustering algorithm, J. High Energy Phys. 04 (2008) 063.

[39] Matteo Cacciari, Gavin P. Salam, and Gregory Soyez, FastJet user manual, Eur. Phys. J. C **72**, 1896 (2012).

[40] Jason Gallicchio and Matthew D. Schwartz, Quark and gluon jet substructure, J. High Energy Phys. 04 (2013) 090.

[41] Murray Rosenblatt, Remarks on some nonparametric estimates of a density function, Ann. Math. Stat. **27**, 832 (1956).

[42] Emanuel Parzen, On estimation of a probability density function and mode, Ann. Math. Stat. **33**, 1065 (1962).

[43] Laurens Van Der Maaten, Learning a parametric embedding by preserving local structure, J. Mach. Learn. Res. **5**, 384 (2009).

[44] Laurens Van Der Maaten and Geoffrey Hinton, Visualizing non-metric similarities in multiple maps, Mach. Learn. **87**, 33 (2012).

[45] Carlos R. García-Alonso, Leonor M. Pérez-Naranjo, and Juan C. Fernández-Caballero, Multiobjective evolutionary algorithms to identify highly autocorrelated areas: The case of spatial distribution in financially compromised farms, Ann. Oper. Res. **219**, 187 (2014).

[46] Laurens Van Der Maaten, Accelerating t-SNE using tree-based algorithms, J. Mach. Learn. Res. **15**, 3221 (2015).

[47] S. Peleg, M. Werman, and H. Rom, A unified approach to the change of resolution: Space and gray-level, IEEE Trans. Pattern Anal. Mach. Intell. **11**, 739 (1989).

[48] Y. Rubner, C. Tomasi, and L. J. Guibas, A metric for distributions with applications to image databases, in *Proceedings of the Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)* (IEEE, New York, 1998), pp. 59–66.

[49] Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas, The Earth mover's distance as a metric for image retrieval, Int. J. Comput. Vis. **40**, 99 (2000).

[50] Anja Butter, Gregor Kasieczka, Tilman Plehn, and Michael Russell, Deep-learned top tagging with a Lorentz layer, SciPost Phys. **5**, 028 (2018).

[51] M. Erdmann, E. Geiser, Y. Rath, and M. Rieger, Lorentz boost networks: Autonomous physics-inspired feature engineering, J. Instrum. **14**, P06006 (2019).

[52] Alexander Bogatskiy, Brandon Anderson, Jan T. Offermann, Marwah Roussi, David W. Miller, and Risi Kondor, Lorentz group equivariant neural network for particle physics, arXiv:2006.04780.

[53] Shiqi Gong, Qi Meng, Jue Zhang, Huilin Qu, Congqiao Li, Sitian Qian, Weitao Du, Zhi-Ming Ma, and Tie-Yan Liu, An efficient Lorentz equivariant graph neural network for jet tagging, J. High Energy Phys. 07 (2022) 030.

[54] Alexander Bogatskiy *et al.*, Symmetry group equivariant architectures for physics, in *Proceedings of the 2022 Snowmass Summer Study* (2022), arXiv:2203.06153.

[55] Shikai Qiu, Shuo Han, Xiangyang Ju, Benjamin Nachman, and Haichen Wang, A holistic approach to predicting top quark kinematic properties with the covariant particle transformer, arXiv:2203.05687.

[56] Amit Chakraborty, Sung Hak Lim, and Mihoko M. Nojiri, Interpretable deep learning for two-prong jet classification with jet spectra, J. High Energy Phys. 07 (2019) 135.

[57] Partha Konar, Vishal S. Ngairangbam, and Michael Spannowsky, Energy-weighted message passing: An infra-red and collinear safe graph neural network algorithm, J. High Energy Phys. 02 (2022) 060.

[58] Alexis Romero, Daniel Whiteson, Michael Fenton, Julian Collado, and Pierre Baldi, Safety of quark/gluon jet classification, arXiv:2103.09103.

[59] Oliver Atkinson, Akanksha Bhardwaj, Christoph Englert, Partha Konar, Vishal S. Ngairangbam, and Michael Spannowsky, IRC-safe graph autoencoder for unsupervised anomaly detection, Front. Artif. Intell. **5**, 943135 (2022).

[60] Nimrod Segol and Yaron Lipman, On universal equivariant set networks, arXiv:1910.02421.

[61] François Chollet *et al.*, Keras, https://github.com/fchollet/keras (2015).

[62] Martín Abadi *et al.*, TensorFlow: Large-scale machine learning on heterogeneous distributed systems, arXiv:1603.04467.

[63] Sergey Ioffe and Christian Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, arXiv:1502.03167.

[64] Diederik P. Kingma and Jimmy Ba, Adam: A method for stochastic optimization, arXiv:1412.6980.

[65] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar, On the convergence of Adam and beyond, arXiv:1904.09237.

[66] Tianqi Chen and Carlos Guestrin, XGBoost, in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16* (ACM, New York, NY, USA, 2016), pp. 785–794.

[67] J. Bergstra, D. Yamins, and D. D. Cox, Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures, in *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Part 1* (JMLR, 2013), pp. 115–123.

[68] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl, Algorithms for hyper-parameter optimization, in *Advances in Neural Information Processing Systems 24: Proceedings of the 25th Annual Conference on Neural Information Processing Systems 2011, NIPS 2011* (Curran Associates Inc, Red Hook, New York, USA, 2011).