PHYS 139/239 Final Project Report (Group 2)

Arturo Sorensen, Danylo Drohobytsky, Fredy Ramirez, and Zihan Zhao

University of California, San Diego

I Introduction

A hadron is a subatomic particle composed of multiple quarks held together through the strong force by exchanging elementary particles known as gluons. Jets, on the other hand, are showers of hadrons initiated by a primary particle whose identity can be determined by examining the hadrons inside the jet. This is done in particle detectors by using algorithms that identify, or tag, the jets.

A common decay product of the Higgs boson tends to be the bottom quark, which has a unique jet signature since hadrons containing bottom quarks have a lifetime of approximately 1.5 ps, allowing there to be a detectable displacement from the point of proton-collision and their decay. Due to this b hadron property, the result is a secondary vertex (SV) displaced from the primary vertex (PV). Modern particle detectors being able to accurately calculate the SV position and displacement from the PV — despite dense environments such as jets with high transverse momentum — allows for tagging a Higgs boson decaying to a bottom quark and bottom antiquark since the jet components come from the two displaced vertices.

The main task of this paper is to achieve higher accuracy in jet tagging, i.e. to identify the nature of the primary particle that initiates a shower by studying the collective features of the hadrons inside the jet.

Deep learning (DL) algorithms have greatly improved the accuracy of jet tagging, due to their ability to automatically extract features from highly complex input data. However, traditional DL approaches are limited by the fact that particle jets involve multiple entities with complex interactions that are not easily encoded as images or lists. Such relational information naturally induces a graph representation, thus the application of Graph Neural Networks (GNN) to jet tagging.

More specifically, we plan to use similar methodology, as in [4], by using an interaction network (IN), a type of GNN, to identify $H \rightarrow b\bar{b}$ jets produced by a process where a Higgs boson decays into one bottom quark and one bottom antiquark. This is binary classification problem: we want to classify the input jets as either $H \rightarrow b\bar{b}$ jets (signal) or QCD jets (background).

II Dataset

The data set that is used to train and evaluate the model is a sample of fully simulated LHC collision events, released by the CMS Collaboration on the CERN Open Data portal [3]. Due to the limitation in available resources, we used 300k samples (jets) for training, 100k samples for validation, and 200k samples for testing. The features we used can be put into four categories: jet features, track features, Particle Flow (PF, a reconstruction algorithm) candidate features, and secondary vertex (sv) features [2]. We used the same features that were used in the original implementation since we are reproducing their results.

Our data processing steps are as follows: First we downloaded the root files from CMS Open Data portal [3]. Each root file contained 200k jets. We then extracted the necessary information

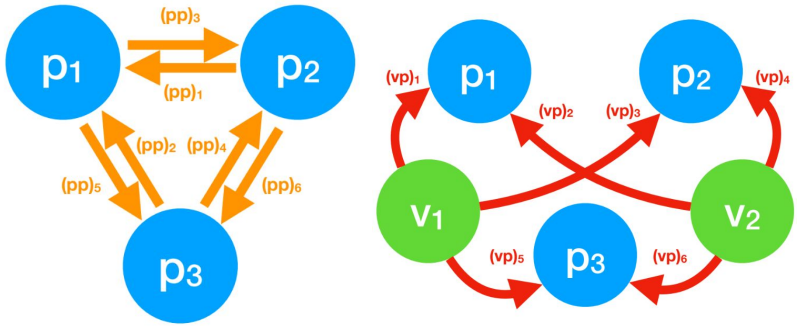

Figure 1: *Two example graphs with 3 particles and 2 vertices and the corresponding edges [4].*

(features, labels, and other variables), from which we created h5 files. Each h5 file contained 100k jets. During training and testing, we read the features and labels from the processed h5 files.

III Methods

We used a baseline/benchmark method to compare to the IN in order to evaluate it properly and see how the performance differs. Moreover, we plan to follow a similar method as in [1] when creating the benchmark and change it accordingly where it is needed. For reference, [1] uses a Keras model with batch normalization followed by three dense hidden layers of sizes (64,32,32) with a ReLU activation function after each, and a dense output layer the same size as the number of labels (in this case 2) with a softmax activation function. The model is trained using the Adam optimizer, a batch size of 1024 for up to 100 epochs, enforcing early stopping on the validation loss with a patience of 10 epochs, and the loss function is categorical cross-entropy. The 27 features used are mentioned in [4] as the high-level features (HFL) used by the DDB algorithm.

The IN, on the other hand, uses 30 features related to charged particles and 14 SV features. Each particle is represented with a feature vector length P and each vertex has feature vector length S. The interaction network has two input collections, N_p particles N_v vertices. A singular jet is designated with an an X matrix, size $P \times N_p$, this matrix contains columns of input features and rows of charged particles. In addition, a Y matrix, size $P \times N_v$ composed of the support vector input features is used to describe a singular jet. Two graphs are constructed, a particle graph, $\mathcal{G}_p$, and a particle vertex graph, $\mathcal{G}_{pv}$. Receiving adjacency matrix, R_R, and sending adjacency matrix R_S are defined. Only connections that are sent to particles are considered. The i, jth entry of the adjacency matrices is 0 unless the ith particle receives or sends to the jth edge. A particle-particle interaction matrix, B_{pp}, and a particle-vertex interaction matrix, B_{vp}, is defined. The effect matrix, E_{pp} and $\overline{E}_{pp}$, results from processing the interaction matrices into internal representations. Matrix C is defined as $(X, \overline{E}_{pp}, \overline{E}_{vp})^T$. The learned representation matrix O is computed by summing over particles to produce a feature vector them passed to the classifier.

When training, the input data is split into $8/10$ training, $1/10$ validation, and $1/10$ test. Implementation and training will be done using PyTorch with 8CPUs, 16GP RAM, and a GPU from UCSD's Data Science and Machine Learning Platform. The model will take up to 60 charged

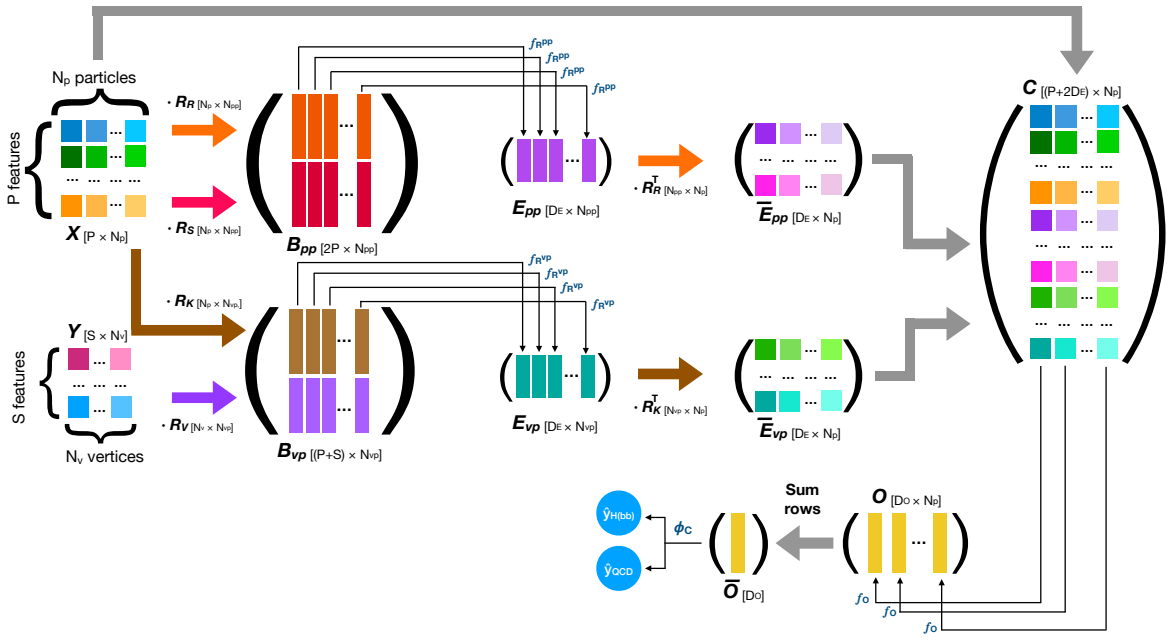

Figure 2: *Illustration of the IN classifier from [4].*

particles and 5 secondary vertices as input. The classifier is illustrated in Figure 2. In the figure f_R^{pp} and f_R^{vp} are expressed as a sequence of 3 dense layers of sizes $(60, 30, 20)$ with ReLU activation functions. Similarly, f_O dense layers are of size $(60, 30, 24)$. The model is trained with an initial learning rate of 10^{-4} and a batch size of 128 and is cut off at 200 epochs. Early stopping is used with a patience of 5 epochs.

IV Results

The expected outcome of this project is to develop an IN algorithm which can identify high-transverse-momentum $H \to b\bar{b}$ and distinguish them from ordinary jets that reflect the configurations of quarks and gluons at short distances. We expect, that the ability of INs to learn complex relationships aids in identifying the patterns present in Higgs bosons decaying to bottom quark-antiquark pairs.

In this project, we will demonstrate that an IN with an extended feature representation outperforms other methods for $H \to b\bar{b}$ tagging, while relying on fewer parameters. Furthermore, we will investigate the use of INs on tracking, vertexing, and substructure properties of the jet and employ this optimized representation to enhance tagging. The network (i.e., charged particles and secondary vertices on a graph) can learn a characterization of each particle-to-particle and particle-to-vertex interactions. An illustration of this is shown in Fig. 1. Therefore, allowing us to exploit this information to categorize a given jet as a signal or background. We will compare performance to different algorithms that we trained with open simulation for $H \to b\bar{b}$ tagging.

The IN model we implemented achieved a significant increase in both AUC and accuracy, compared with the baseline DNN model, as reflected in the two tables below.2.

Model	AUC	Accuracy
Full IN (trained with entire dataset)	99.0%	95.5%
Small IN (trained with small dataset)	98.2%	90.4%

Table 1: *Full vs small.*

Model	AUC	Accuracy
Baseline DNN	90.5%	82.1%
Interaction Network	98.2%	90.4%

Table 2: *Model vs Baseline.*

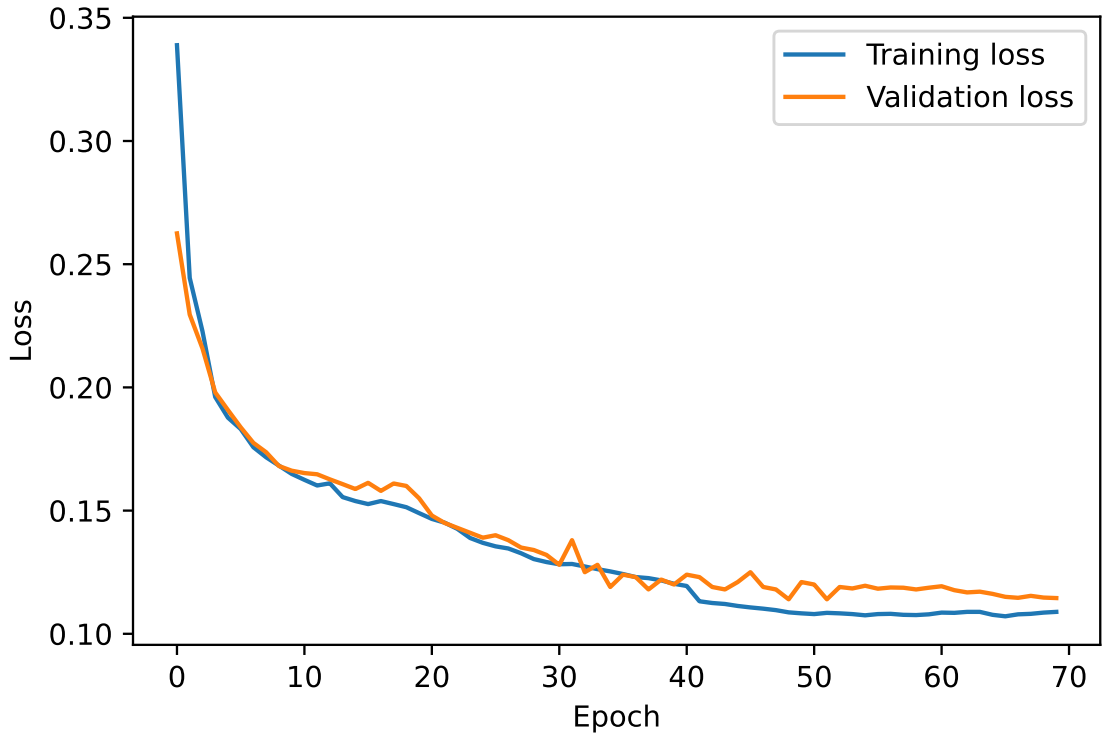

Figure 3: *Loss vs Epochs.*

IV References

[1] Javier Duarte. https://github.com/cernopendata-datascience/higgstobbmachinelearning, 2019.

[2] Javier Duarte. https://github.com/javierzhao/reproduction$_of_in$/blob/main/src/data/definitions.yml, 2019

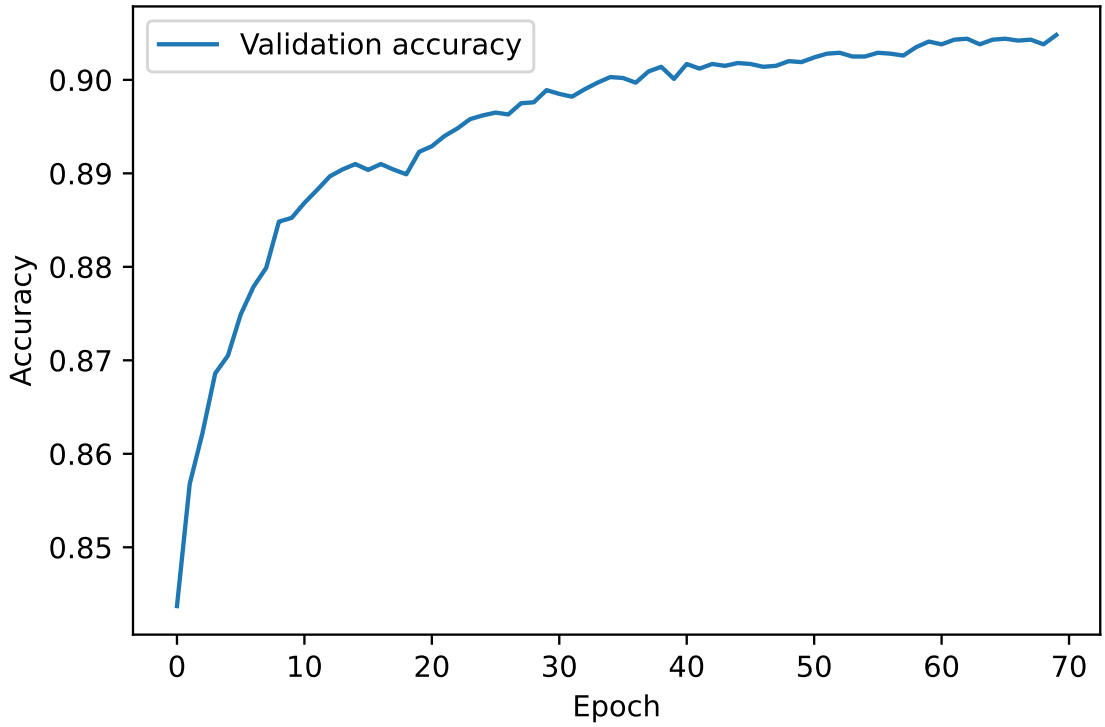

Figure 4: *accuracy vs Epochs.*

[3] Javier Duarte. Sample with jet, track and secondary vertex properties for hbb tagging ml studies higgstobbntuple$_h iggstobb_q cd_r unii_1 3tev_m c, July 2019$.

[4] Eric A. Moreno, Thong Q. Nguyen, Jean-Roch Vlimant, Olmo Cerri, Harvey B. Newman, Avikar Periwal, Maria Spiropulu, Javier M. Duarte, and Maurizio Pierini. Interaction networks for the identification of boosted $H \to b\bar{b}$ decays. *Phys. Rev. D*, 102:012010, Jul 2020.

[a]0.2

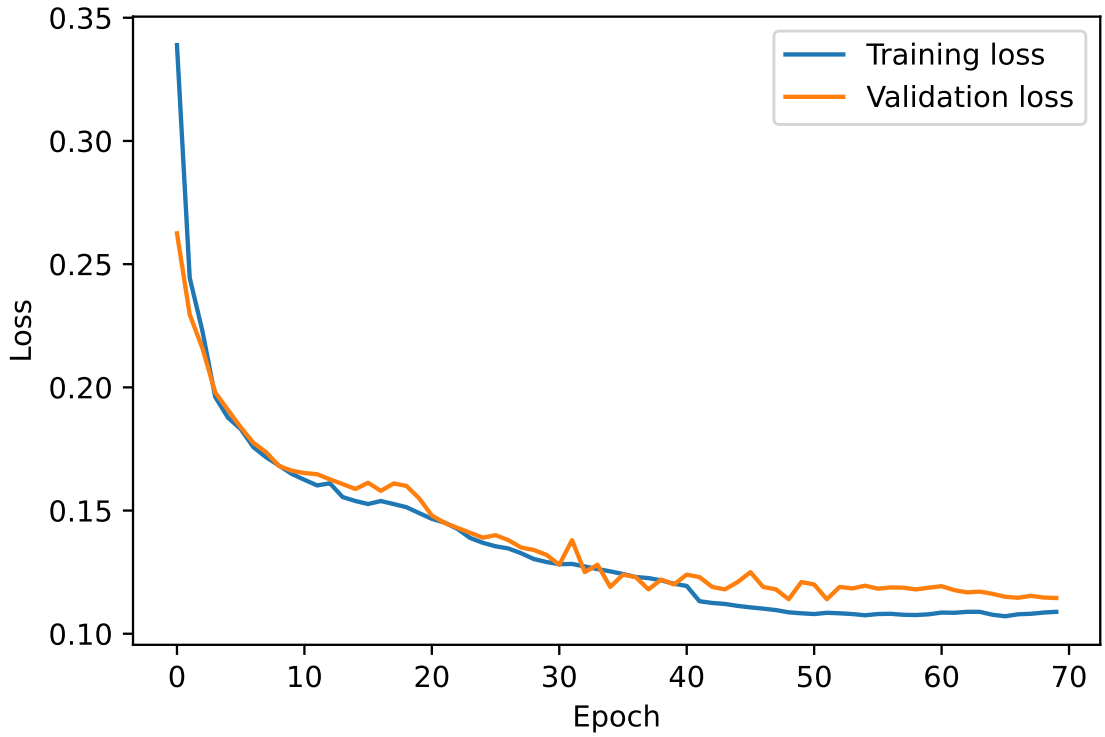

Figure 5:

[b]0.2

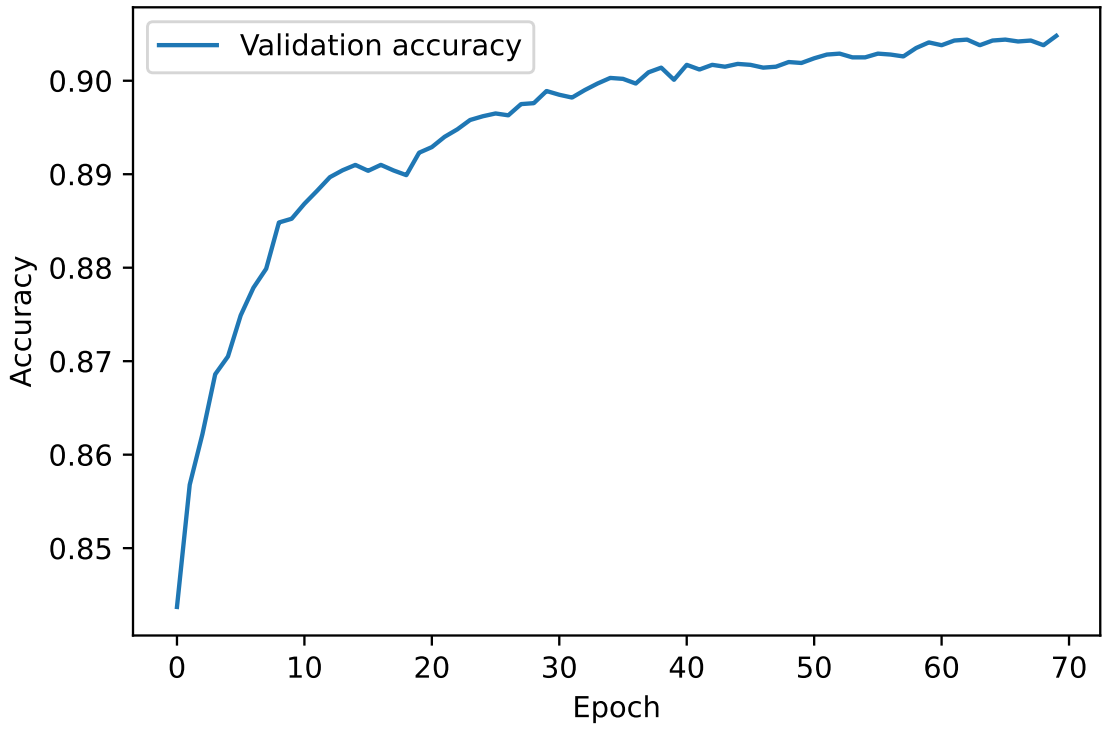

Figure 6:

Figure 7: (a). (b).

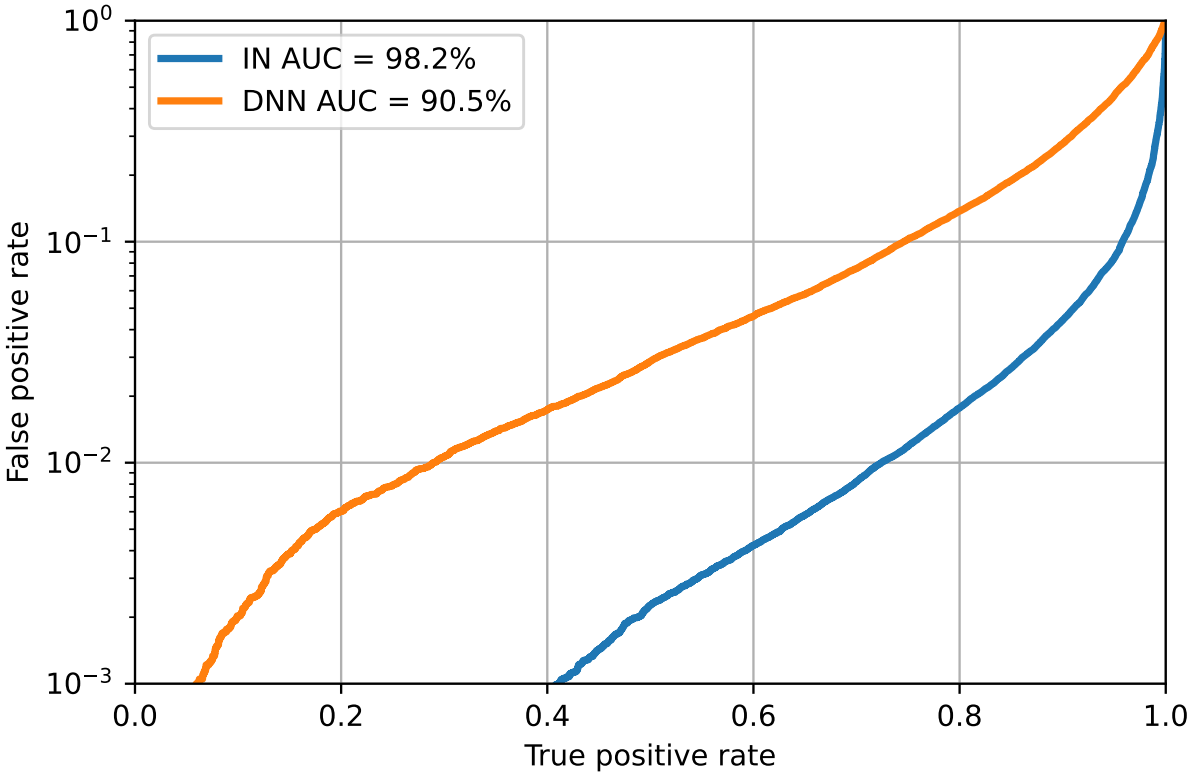

Figure 8: *Log ROC.*

TABLE III. Charged particle features. The IN and DDB+ models use all of the features, while DDB algorithm uses the subset of features indicated in bold.

Variable	Description
`track_ptrel`	p_{T} of the charged particle divided by the p_{T} of the AK8 jet
`track_erel`	Energy of the charged particle divided by the energy of the AK8 jet
`track_phirel`	$\Delta\phi$ between the charged particle and the AK8 jet axis
`track_etarel`	$\Delta\eta$ between the charged particle and the AK8 jet axis
`track_deltaR`	ΔR between the charged particle and the AK8 jet axis
`track_drminsv`	ΔR between the associated SVs and the charged particle
`track_drsubjet1`	ΔR between the charged particle and the first soft drop subjet
`track_drsubjet2`	ΔR between the charged particle and the second soft drop subjet
`track_dz`	Longitudinal impact parameter of the track, defined as the distance of closest approach of the track trajectory to the PV projected on to the z direction
`track_dzsig`	Longitudinal impact parameter significance of the track
`track_dxy`	Transverse (2D) impact parameter of the track, defined as the distance of closest approach of the track trajectory to the beam line in the transverse plane to the beam
`track_dxysig`	Transverse (2D) impact parameter of the track
`track_normchi2`	Normalized χ^2 of the track fit
`track_quality`	Track quality: `undefQuality=-1, loose=0, tight=1, highPurity=2, confirmed=3, looseSetWithPV=5, highPuritySetWithPV=6, discarded=7, qualitySize=8`
`track_dptdpt`	Track covariance matrix entry (p_{T}, p_{T})
`track_detadeta`	Track covariance matrix entry (η, η)
`track_dphidphi`	Track covariance matrix entry (ϕ, ϕ)
`track_dxydxy`	Track covariance matrix entry (d_{xy}, d_{xy})
`track_dzdz`	Track covariance matrix entry (d_z, d_z)
`track_dxydz`	Track covariance matrix entry (d_{xy}, d_z)
`track_dphidz`	Track covariance matrix entry (d_ϕ, d_z)
`track_dlambdadz`	Track covariance matrix entry (λ, d_z)
`trackBTag_EtaRel`	$\Delta\eta$ between the track and the AK8 jet axis
`trackBTag_PtRatio`	Component of track momentum perpendicular to the AK8 jet axis, normalized to the track momentum
`trackBTag_PParRatio`	Component of track momentum parallel to the AK8 jet axis, normalized to the track momentum
`trackBTag_Sip2dVal`	Transverse (2D) signed impact parameter of the track
`trackBTag_Sip2dSig`	Transverse (2D) signed impact parameter significance of the track
`trackBTag_Sip3dVal`	3D signed impact parameter of the track
`trackBTag_Sip3dSig`	3D signed impact parameter significance of the track
`trackBTag_JetDistVal`	Minimum track approach distance to the AK8 jet axis

Variable	Description
`sv_ptrel`	p_{T} of the SV divided by the p_{T} of the AK8 jet
`sv_erel`	Energy of the SV divided by the energy of the AK8 jet
`sv_phirel`	$\Delta\phi$ between the SV and the AK8 jet axis
`sv_etarel`	$\Delta\eta$ between the SV and the AK8 jet axis
`sv_deltaR`	ΔR between the SV and the AK8 jet axis
`sv_pt`	p_{T} of the SV
`sv_mass`	Mass of the SV
`sv_ntracks`	Number of tracks associated with the SV
`sv_normchi2`	Normalized χ^2 of the SV fit
`sv_costhetasvpv`	$\cos\theta$ between the SV and the PV
`sv_dxy`	Transverse (2D) flight distance of the SV
`sv_dxysig`	Transverse (2D) flight distance significance of the SV
`sv_d3d`	3D flight distance of the SV
`sv_d3dsig`	3D flight distance significance of the SV

TABLE IV. Secondary vertex features. The IN and DDB+ models use all of the features, while the DDB algorithm uses the subset of features indicated in bold.