Note

Feng-Yang Hsieh

1 Bootstrapping

The bootstrapping method involves an iterative process of training a classifier on mixed datasets. We start by training a classifier on the initial mixed datasets. The trained classifier is then used to reclassify the data, creating a new mixed training set. This process is repeated multiple times to increase the sample fraction differences in mixed datasets. We want to explore whether this process can improve CWoLa's performance.

We implemented this method, considering the events sampled from the normal distribution for the testing. We found it unsuccessful. The model initially achieved the best performance, then worsened at subsequent iterations.

The reason is the reclassification step breaks the key assumption of the CWoLa approach: the signal and background events should have the same distributions in both mixed datasets. Figure 1 shows the initial signal and background distributions. Signal has the same distribution in mixed dataset M_1 and M_2, as does the background. Figure 2 shows signal and background distributions after the reclassification. We could observe the signal events have different distributions in M_1 and M_2. As a result, the assumption of the CWoLa approach is violated, leading to the failure of the bootstrapping method.

2 Multi-class CWoLa

The original CWoLa is only applied to the binary classification tasks. We want to extend the traditional CWoLa to more than two classes.

2.1 Dataset and model setup

We consider three pure samples, denoted by A, B, and C. The mixed datasets are denoted by M_1, M_2, and M_3. The training dataset is sampled from 5-dimensional normal

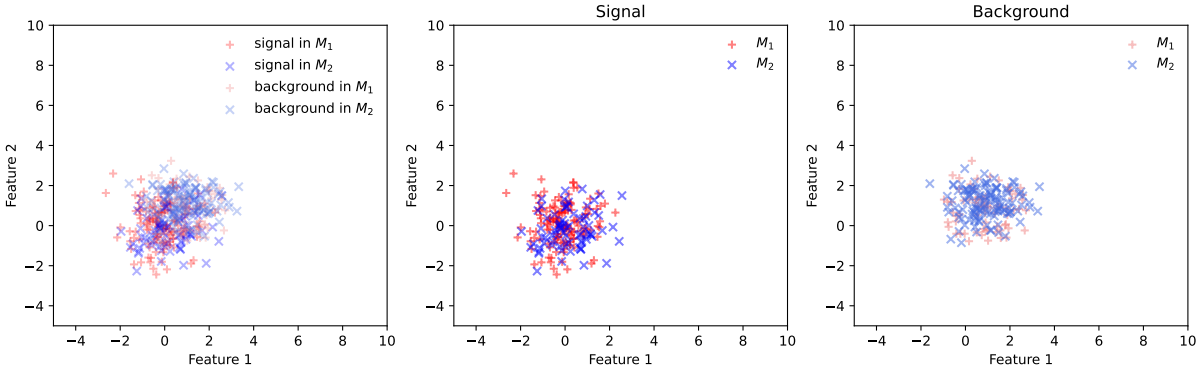


Figure 1: The signal and background samples distributions. The signal and background events are sampled from different two-dimensional normal distributions. They are randomly assigned to the mixed datasets M_1 or M_2.

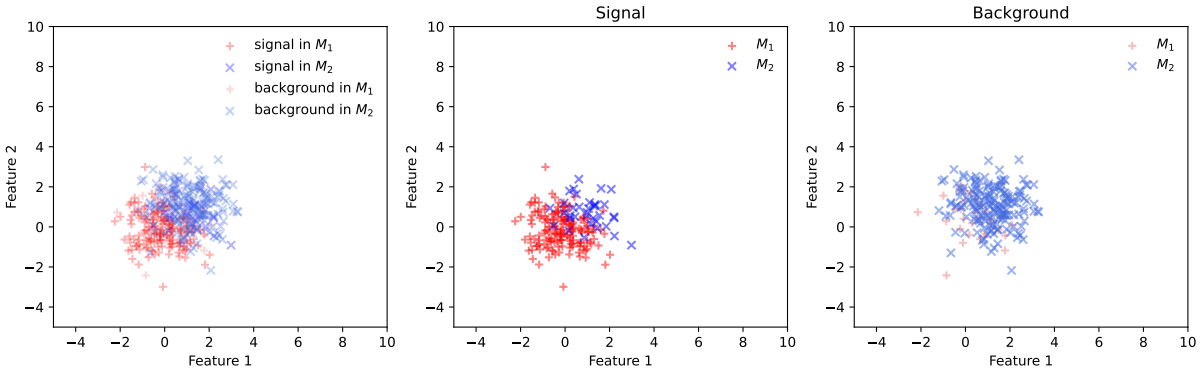


Figure 2: The signal and background samples distributions. The signal and background events are sampled from different two-dimensional normal distributions. The trained classifier assigns them to the M_1 or M_2.

distributions. Each mixed dataset has 10,000 events. The testing dataset consists of 1,000 pure samples for each class.

The neural network consists of two dense layers with eight hidden nodes. The loss function is categorical cross-entropy. To prevent over-training, we utilize the early stopping technique with patience 10. The output is a 3-dimensional vector, and each component can be interpreted as the probability belonging to each type.

2.2 Dominated case

To simplify the problem, we start by considering a case where each pure class dominates different mixed samples. This would help us understand the behavior of the multi-class classifier before going to more complex scenarios.

Table 1 lists the fractions of pure samples in mixed datasets. Each mixed dataset is dominated by one pure sample type. Figure 3 illustrates the distributions of the pure classes within these datasets.

Table 1: Fractions of pure samples in mixed datasets.

Dataset	A	B	C
M_1	0.70	0.20	0.10
M_2	0.10	0.70	0.20
M_3	0.20	0.10	0.70

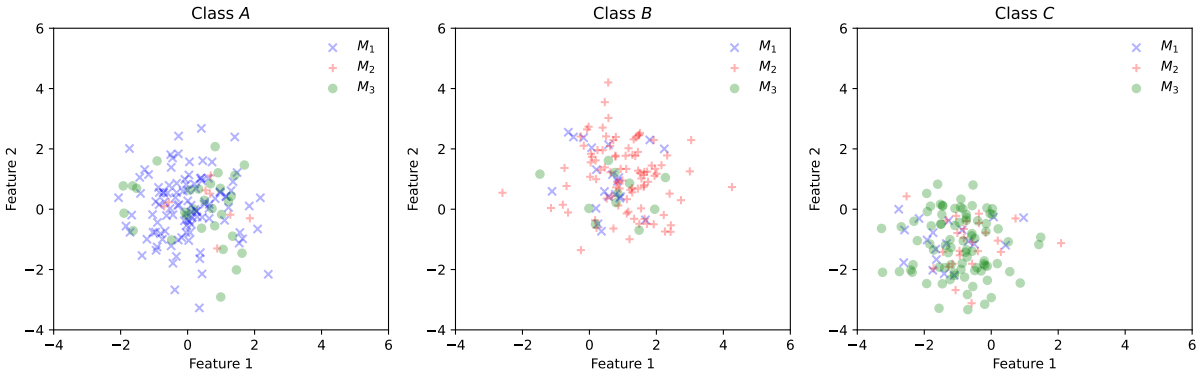
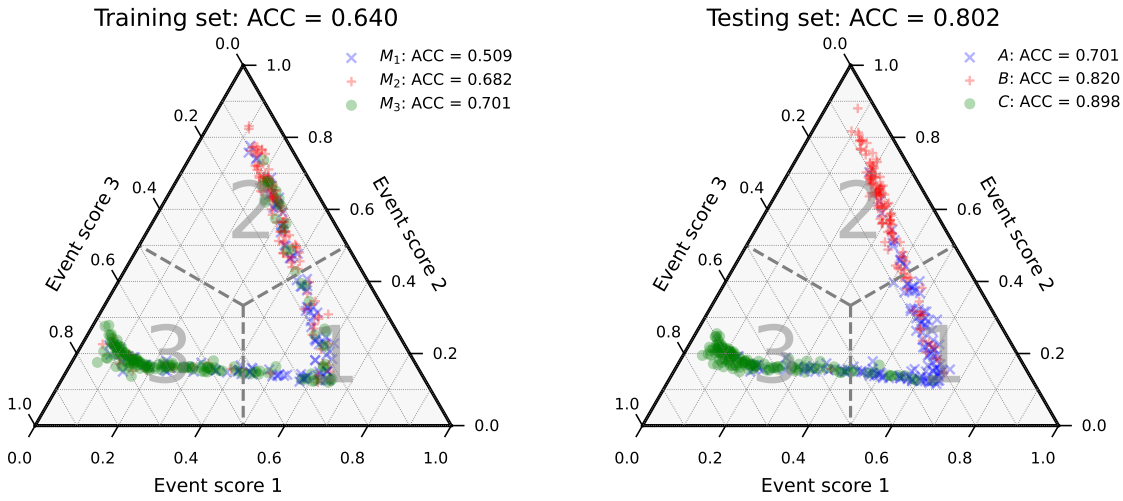


Figure 3: Distributions of pure samples in the mixed datasets. The classes A, B, and C are sampled from the normal distribution with mean values $0, 1, -1$, respectively.

If the output vector suggests that the first type has the highest probability for dominated cases, it can be directly interpreted as belonging to the class A. The same logic applies to classes B and C.

The event scores are the components of the output vector. Since the sum of event scores is always 1, we use the ternary plots to represent the event score distributions. In these plots, the coordinate of a point should be read following the direction of the ticks on each axis.

Figure 4 shows the scatter plots of output vectors. The event score distributions are well-separated for each class, indicating successful training. The connections from regions 2 to 1 to 3, due to the mean value of class B, A, and C, are $1, 0$, and -1, respectively.



(a) Training data

(b) Testing data

Figure 4: Ternary plots of event score distributions. The grey dashed lines separate the regions corresponding to different classification results. The total accuracy (ACC) and the accuracy for each type are displayed.

Table 2 presents another type of the dominated case. Although the pure sample A has the largest fraction in each mixed dataset, the pure class is still dominant in different mixed datasets. Figure 5 illustrates the distributions of the pure classes in these datasets.

Table 2: Fractions of pure samples in mixed datasets.

Dataset	A	B	C
M_1	0.80	0.10	0.10
M_2	0.70	0.25	0.05
M_3	0.60	0.20	0.20

Figure 6 shows scatter plots of output vectors. Although the training accuracy is not good, the testing results demonstrate excellent performance.

From these results, we can conclude the ternary CWoLa works for dominated cases.

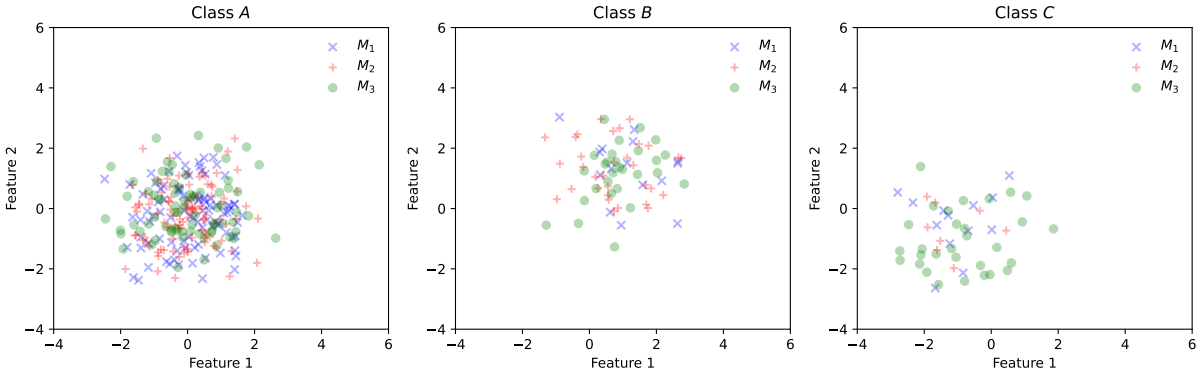
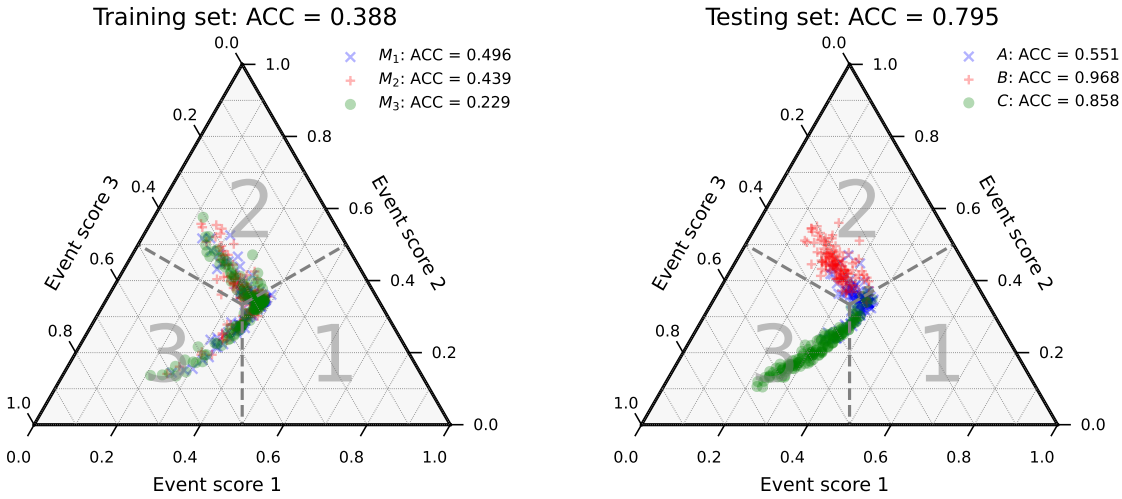


Figure 5: Distributions of pure samples in the mixed datasets. The classes A, B, and C are sampled from normal distributions with mean values 0, 1, and -1, respectively.



(a) Training data

(b) Testing data

Figure 6: Ternary plots of event score distributions. The grey dashed lines separate the regions corresponding to different classification results. The total accuracy (ACC) and the accuracy for each type are displayed.

2.3 Ambiguous case

Table 3 lists the fractions of pure samples in mixed datasets. The ambiguous cases mean that the pure sample is not dominant in different mixed datasets. For instance, in table 3, pure samples B and C both dominate in M_3. Figure 7 illustrates the distributions of the pure classes within these datasets.

Table 3: Fractions of pure samples in mixed datasets.

Dataset	A	B	C
M_1	0.80	0.10	0.10
M_2	0.70	0.20	0.10
M_3	0.60	0.25	0.15

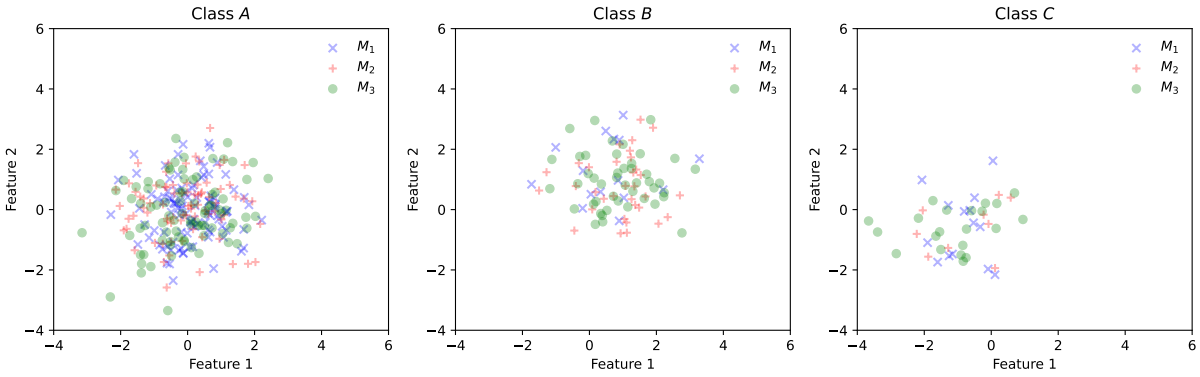
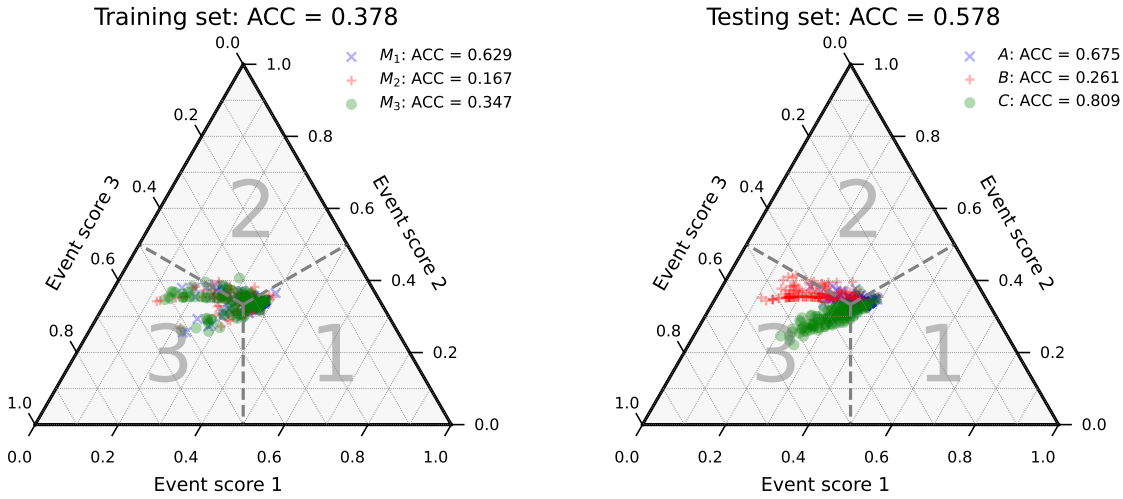


Figure 7: Distributions of pure samples in the mixed datasets. The classes A, B, and C are sampled from the normal distribution with mean values $0, 1$, and -1, respectively.

For ambiguous cases, we apply the same logic to interpret the output vector as in dominated cases. However, the results could be problematic.

Figure 8 shows scatter plots of the output vectors. While the testing accuracy is very low, different types of pure samples exhibit distinct distributions. This suggests that we need another method to interpret the output vector.

Note that the accuracy for class B is very low, while the accuracy for class C is significantly higher. This suggests that when the classifier receives an event from either class B or C, it tends to assign a high event score of type 3. This behavior is consistent with the fact that the pure samples B and C both are dominated in M_3. We need to refine our classification method or better interpret the output vectors to improve performance.



(a) Training data

(b) Testing data

Figure 8: Ternary plots of event score distributions. The grey dashed lines separate the regions corresponding to different classification results. The total accuracy (ACC) and the accuracy for each type are displayed.

3 Output vector interpretation

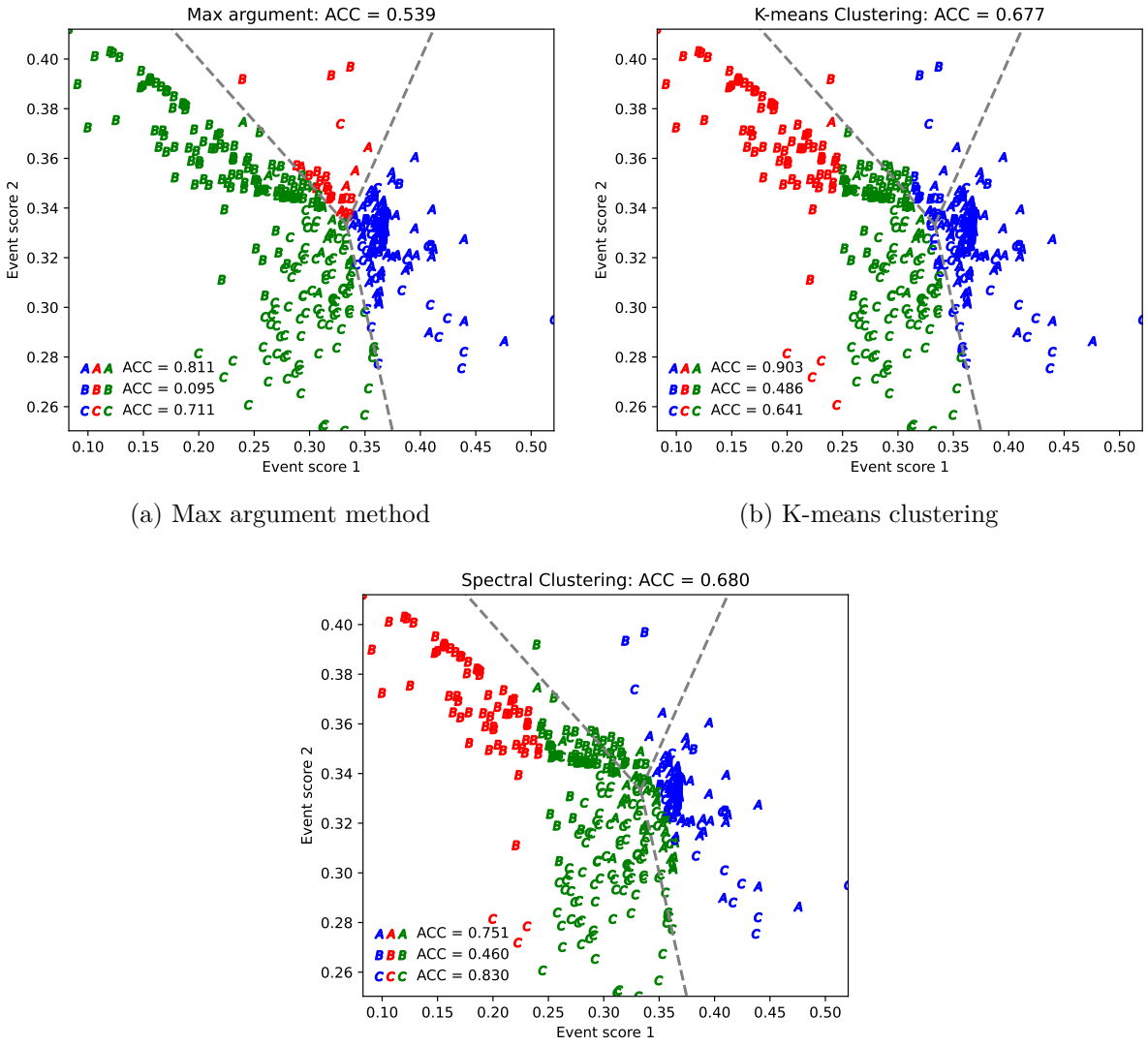
In section 2, the predicted label is determined from the output vector using the following method: if the first class has the highest probability, the sample is assigned to class A. The same logic applies to classes B and C. We refer to this approach as the "max argument" method.

While this interpretation works well for dominant cases, it performs poorly for ambiguous cases. However, for such cases, the output vector distributions still differ between pure classes, suggesting that we need a better prediction method. The clustering algorithms might provide an alternative way to determine the type of a given sample.

3.1 Ambiguous case

We consider the ambiguous cases described in section 3.1 and test different clustering methods as alternative prediction strategies.

Figure 9 presents the prediction results for various methods, where colors represent the predicted labels. Both clustering approaches show improvements in overall accuracy. Table 4 summarizes the mean and standard deviation of accuracies for each prediction method. Notably, both clustering methods significantly improve the accuracy for pure class B, contributing to the observed increase in total accuracy.



(a) Max argument method

(b) K-means clustering

(c) Spectral clustering

Figure 9: Event score distribution with prediction results. The grey dashed lines indicate regions corresponding to different predicted labels using the max argument method. The total accuracy and per-class accuracies are displayed.

Table 4: Accuracy of different prediction methods. The means and standard deviations are computed over ten training runs.

Prediction Method	A	B	C	Total
Max argument	0.742 ± 0.035	0.164 ± 0.100	0.707 ± 0.091	0.538 ± 0.027
K-means clustering	0.868 ± 0.028	0.534 ± 0.077	0.595 ± 0.050	0.666 ± 0.041
Spectral clustering	0.823 ± 0.114	0.625 ± 0.096	0.656 ± 0.173	0.702 ± 0.061

3.2 Dominant Case

We consider the dominant case described in table 1 using different clustering methods as alternative prediction strategies.

Figure 10 shows the prediction results for various methods. The three prediction methods exhibit similar performance. Since the max argument method already performs well, there is limited room for improvement.

Table 5 summarizes the mean and standard deviation of accuracies. While spectral clustering achieves better accuracy for class A, it performs worse for class B, resulting in a total accuracy that is approximately 2% lower compared to the other prediction methods.

Table 5: Accuracy of different prediction methods. Means and standard deviations are computed over ten training runs.

Prediction Method	A	B	C	Total
Max argument	0.755 ± 0.040	0.824 ± 0.025	0.884 ± 0.021	0.821 ± 0.003
K-means clustering	0.784 ± 0.020	0.828 ± 0.017	0.857 ± 0.017	0.823 ± 0.003
Spectral clustering	0.813 ± 0.029	0.746 ± 0.019	0.864 ± 0.019	0.807 ± 0.005

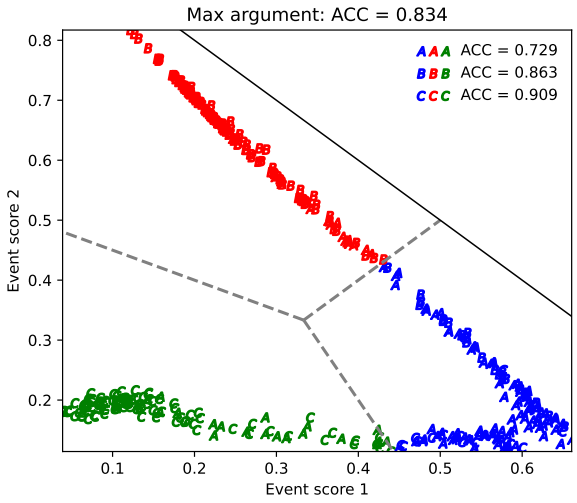
4 Differences between training and testing sets

In real applications, we only have training datasets and cannot obtain the true labels of the samples. Therefore, it is crucial to ensure that the performance of the training and testing datasets is similar, allowing prediction workflows to be effectively applied back to the training sets.

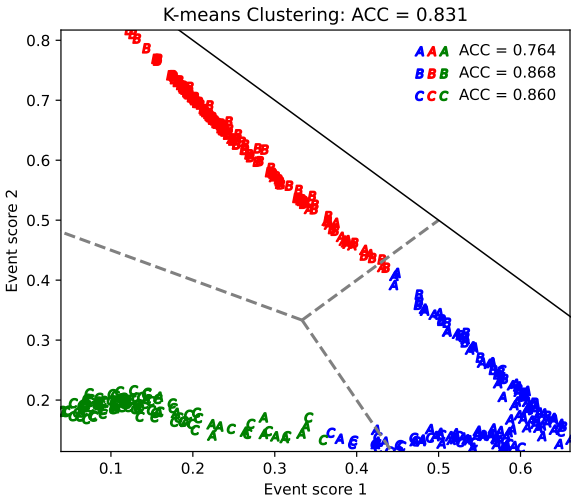
4.1 Dominant Case

For the dominant case described in table 1, we evaluate different clustering methods as alternative prediction strategies.

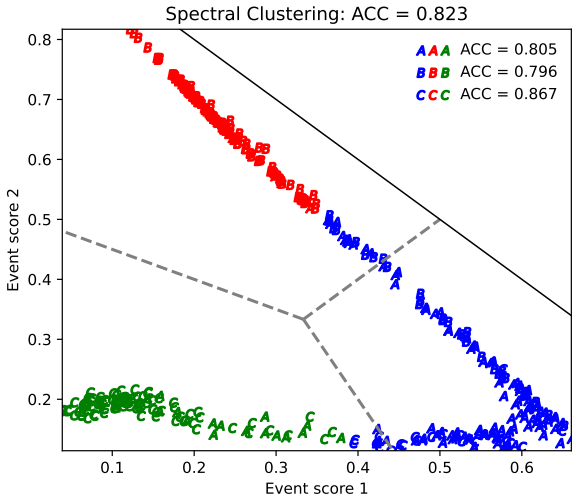
Figure 11 presents the prediction results for both training and testing datasets. The performance is consistent across the two datasets, with similar results for all three prediction methods. These findings are also consistent with the results discussed in section 3.2 and demonstrate that the clustering prediction strategy can be effectively applied to training datasets.



(a) Max argument method

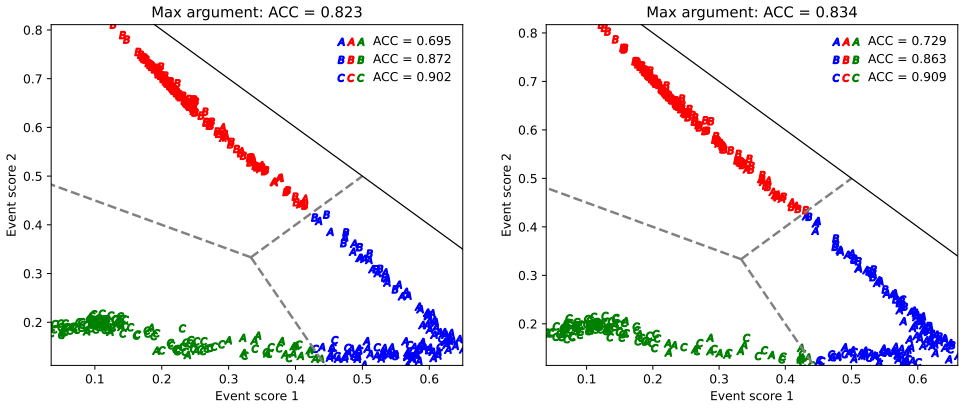


(b) K-means clustering



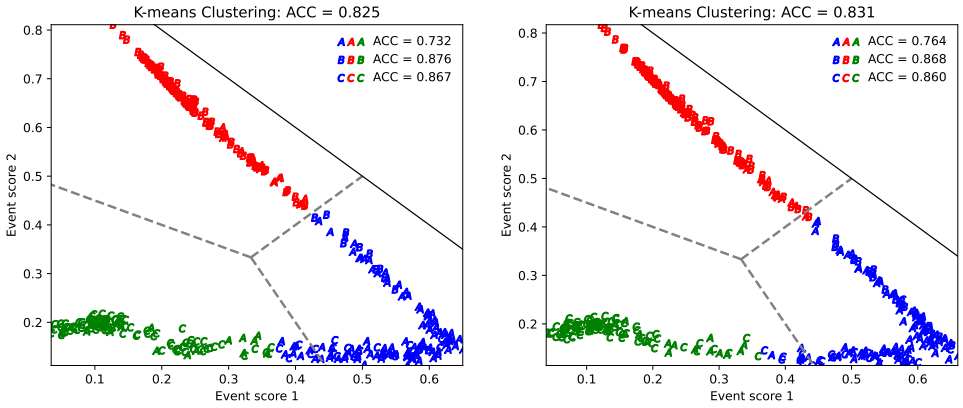
(c) Spectral clustering

Figure 10: Event score distribution with prediction results. The grey dashed lines indicate regions corresponding to different predicted labels using the max argument method. The black solid line represents the event score boundary. Total accuracy and per-class accuracies are displayed.



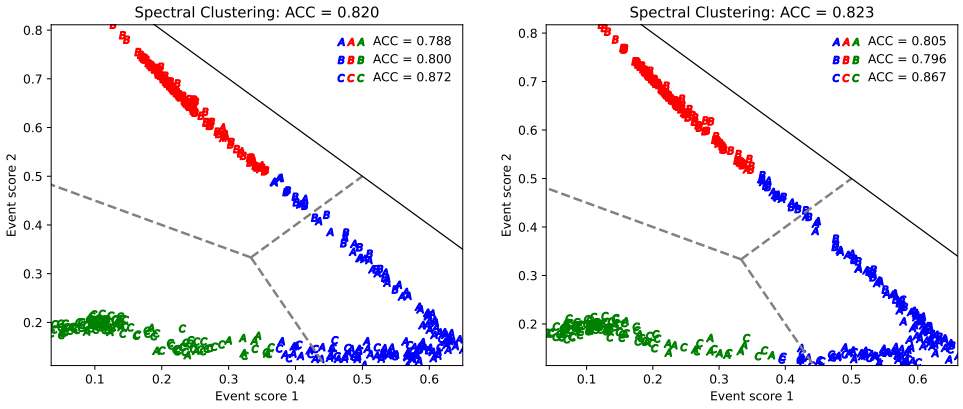
(a) Training sets

(b) Testing sets



(c) Training sets

(d) Testing sets



(e) Training sets

(f) Testing sets

Figure 11: Event score distribution with prediction results. The grey dashed lines indicate regions corresponding to different predicted labels using the max argument method. The black solid line represents the event score boundary. Total accuracy and per-class accuracies are displayed.

4.2 Ambiguous Case

We consider the ambiguous case described in table 3 using different clustering methods as alternative prediction strategies.

Figure 12 compares the prediction results for training and testing datasets. The training sets exhibit better performance for the Max-argument and K-means methods, whereas spectral clustering performs worse. The performance depends on the prediction method used.

Since the classifiers have been exposed to the training datasets, the training sample distribution should be more well-separated compared with the testing set. This could be the reason why the Max-argument and K-means methods have greater ACC on the training set. However, the poorer performance of spectral clustering on training sets is less clear, and we should explore it.

5 Prediction Dependency Issue

When using clustering methods as alternative prediction strategies, the prediction results depend on the entire dataset, as the clustering process requires all samples to make predictions. To predict an event, we need to access the entire dataset.

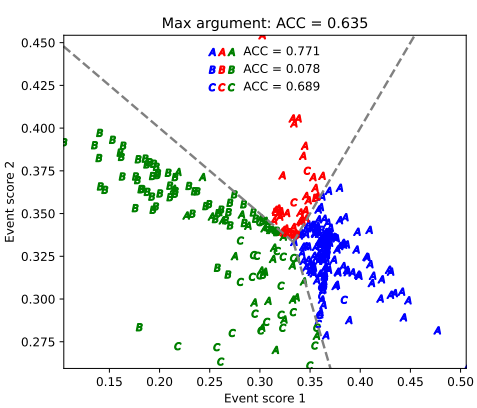
To address this prediction dependency issue, we propose training an additional neural network using the sample labels obtained from clustering-based predictions. This new neural network combines the functionalities of the CWoLa classifier (assigning event scores) and clustering-based prediction (final classification).

The workflow consists of three steps:

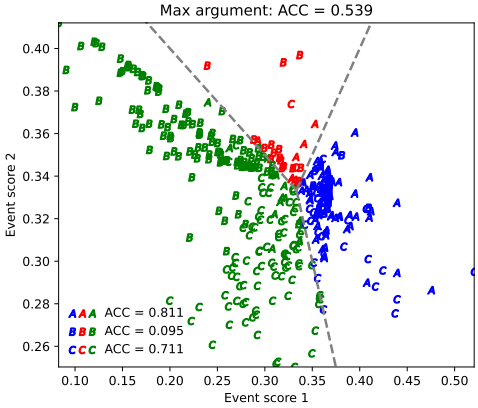
1. Train the initial neural network following the setup described in section 2.1.
2. Apply a clustering prediction method to the training dataset to generate new labels.
3. Train a second neural network using the relabeled training dataset.

This second neural network should reproduce the performance of the original neural network combined with clustering-based prediction.

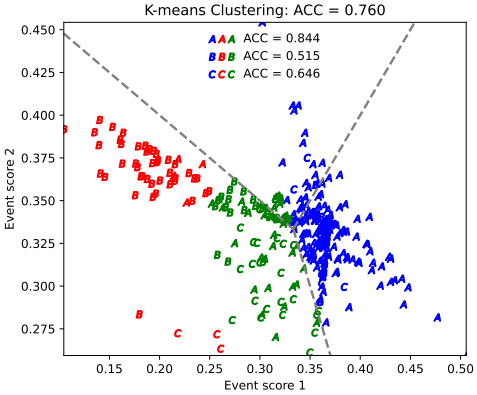
Table 6 summarizes the results for various prediction methods. "K-means NN" refers to the second neural network trained on K-means clustering labels. The results demonstrate that this approach performs similarly to directly applying clustering-based prediction on the original neural network. This finding suggests that a secondary neural network can effectively integrate the functionalities of a neural network and clustering-based prediction, removing the dependency on the entire dataset for predictions.



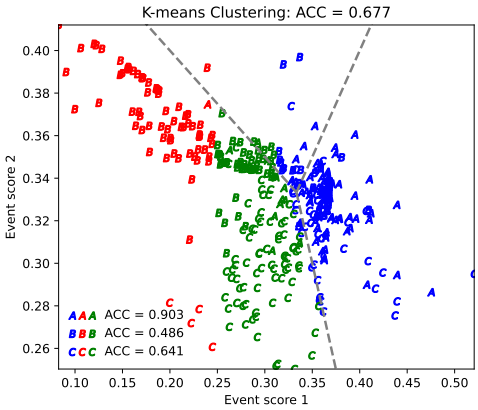
(a) Training sets



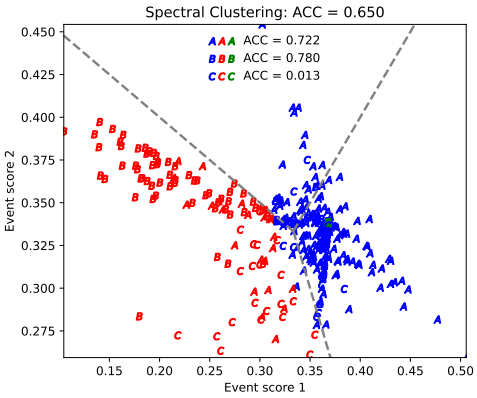
(b) Testing sets



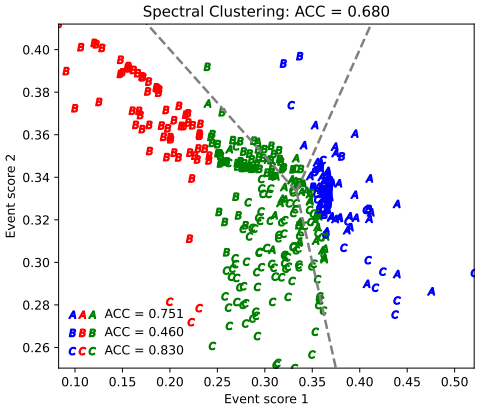
(c) Training sets



(d) Testing sets



(e) Training sets



(f) Testing sets

Figure 12: Event score distribution with prediction results. The grey dashed lines indicate regions corresponding to different predicted labels using the max argument method. The black solid line represents the event score boundary. Total accuracy and per-class accuracies are displayed.

Table 6: Accuracy of different prediction methods evaluated on the testing dataset. Means and standard deviations are calculated over ten training runs.

Prediction Method	A	B	C	Total
Max argument	0.751 ± 0.027	0.831 ± 0.022	0.884 ± 0.017	0.822 ± 0.003
K-means clustering	0.782 ± 0.015	0.830 ± 0.016	0.856 ± 0.018	0.823 ± 0.003
K-means NN	0.781 ± 0.014	0.831 ± 0.018	0.856 ± 0.020	0.823 ± 0.003
Spectral clustering	0.811 ± 0.029	0.768 ± 0.026	0.850 ± 0.045	0.810 ± 0.007
Spectral NN	0.804 ± 0.027	0.762 ± 0.025	0.863 ± 0.027	0.810 ± 0.005

6 Physics process

6.1 Monte Carlo sample generation

We consider the following physics processes:

1. The Z' model process:

$$pp \to Z' \to W^+W^- \to \ell^+\nu_\ell\ell^-\overline{\nu}_\ell$$

2. SM process, W^+W^- production:

$$pp \to W^+W^- \to \ell^+\nu_\ell\ell^-\overline{\nu}_\ell$$

3. SM process, ZZ production:

$$pp \to ZZ \to \ell^+\ell^-\nu_\ell\overline{\nu}_\ell$$

They all have the same final state. Thus, distinguishing these three production processes could be a real application of multi-class CWoLa.

We adopt the Z' model from this page [1, 2].

We use `MadGraph 3.3.1` [3] to generate these processes at a center-of-mass energy of $\sqrt{s} = 13$ TeV. The parton showering and hadronization are simulated using `Pythia 8.306` [4]. The detector simulation is conducted by `Delphes 3.4.2` [5]. Jet reconstruction is performed using `FastJet 3.3.2` [6] with the anti-k_t algorithm [7] and a jet radius of $R = 0.5$. These jets are required to have transverse momentum $p_\mathrm{T} > 20$ GeV. We keep all settings as their default values for now.

The following `MadGraph` scripts generate Monte Carlo samples for each process.

Z' production

```
import model HAHM_variableMW_v5_UFO
generate p p > zp > w+ w-, (w+ > l+ vl), (w- > l- vl~)
output ppZp
launch ppZp

shower=Pythia8
detector=Delphes
analysis=OFF
madspin=OFF
done

set run_card nevents 10000
set run_card ebeam1 6500.0
set run_card ebeam2 6500.0

done
```

W^+W^- production

```
generate p p > w+ w-, (w+ > l+ vl), (w- > l- vl~)
output ppWpWm
launch ppWpWm

shower=Pythia8
detector=Delphes
analysis=OFF
madspin=OFF
done

set run_card nevents 10000
set run_card ebeam1 6500.0
set run_card ebeam2 6500.0

done
```

ZZ production

```
generate p p > z z, (z > l+ l-), (z > vl vl~)
output ppZZ
launch ppZZ

shower=Pythia8
detector=Delphes
analysis=OFF
madspin=OFF
done

set run_card nevents 10000
set run_card ebeam1 6500.0
set run_card ebeam2 6500.0

done
```

6.2 Event selection

The selection cuts after the `Delphes` simulation:

- n_l cut: The number of leptons (electron + muon) should be at least 2.

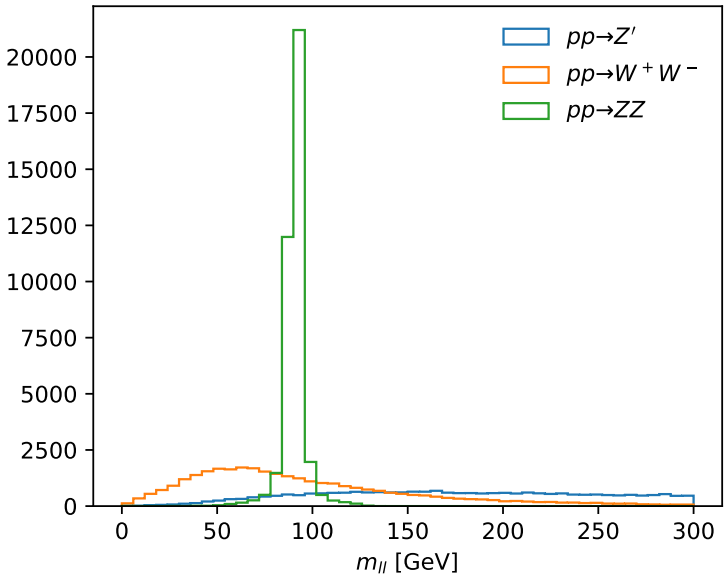
Figure 13 shows the distributions of m_{ll} (the invariant mass of the two leading leptons).

6.3 Event image

The inputs for the neural networks are event images [8, 9, 10]. These images are constructed from events that pass the kinematic selection criteria described in section 6.2. Each event image has three channels corresponding to calorimeter towers, track, and leptons. The following preprocessing steps are applied to all event constituents:

1. Translation: Compute the p_{T}-weighted center in the ϕ coordinates, then shift this point to the origin.
2. Flipping: Flip the highest p_{T} quadrant to the first quadrant.
3. Pixelation: Pixelate in a $\eta \in [-5, 5],\ \phi \in [-\pi, \pi]$ box, with 40×40 pixels

Figure 14 shows the event images for three different physics processes.



(a) m_{ll} distribution

Figure 13: Distributions of the invariant mass m_{ll} of the two leading leptons.

References

[1] D. Curtin *et al.*, "Exotic decays of the 125 GeV Higgs boson," *Phys. Rev. D*, vol. 90, no. 7, p. 075004, 2014.

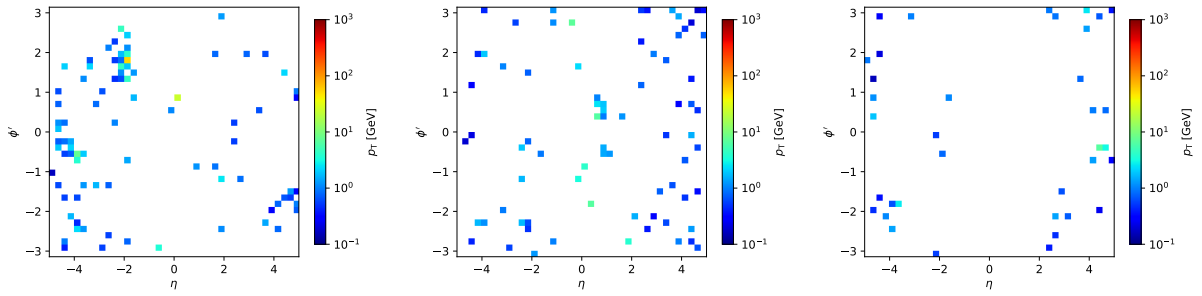
[2] D. Curtin, R. Essig, S. Gori, and J. Shelton, "Illuminating Dark Photons with High-Energy Colliders," *JHEP*, vol. 02, p. 157, 2015.

[3] J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, O. Mattelaer, H. S. Shao, T. Stelzer, P. Torrielli, and M. Zaro, "The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations," *JHEP*, vol. 07, p. 079, 2014.

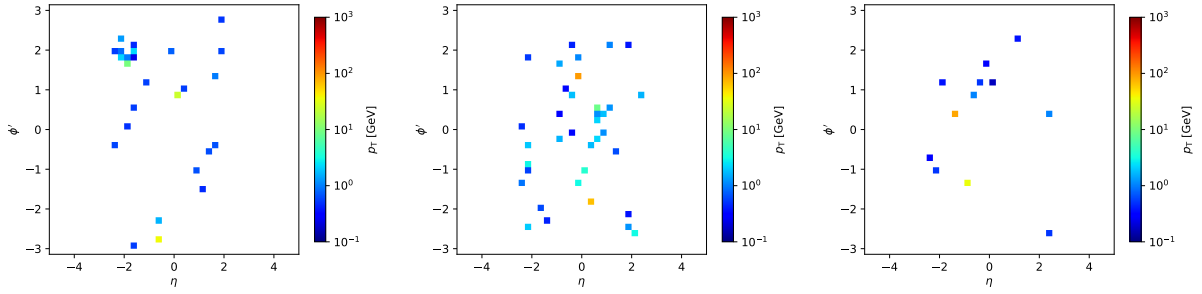
[4] T. Sjöstrand, S. Ask, J. R. Christiansen, R. Corke, N. Desai, P. Ilten, S. Mrenna, S. Prestel, C. O. Rasmussen, and P. Z. Skands, "An introduction to PYTHIA 8.2," *Comput. Phys. Commun.*, vol. 191, pp. 159–177, 2015.

[5] J. de Favereau, C. Delaere, P. Demin, A. Giammanco, V. Lemaître, A. Mertens, and M. Selvaggi, "DELPHES 3, A modular framework for fast simulation of a generic collider experiment," *JHEP*, vol. 02, p. 057, 2014.

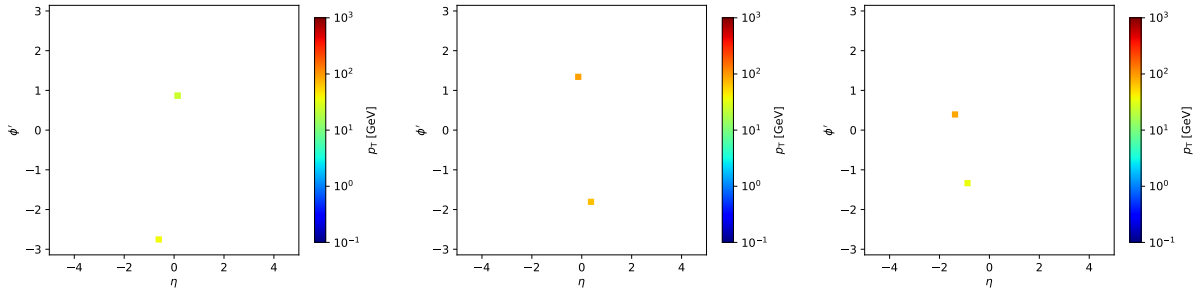
[6] M. Cacciari, G. P. Salam, and G. Soyez, "FastJet User Manual," *Eur. Phys. J. C*, vol. 72, p. 1896, 2012.



(a) Z': Calorimeter Tower (b) W^+W^-: Calorimeter Tower (c) ZZ: Calorimeter Tower



(d) Z': Track (e) W^+W^-: Track (f) ZZ: Track



(g) Z': Lepton (h) W^+W^-: Lepton (i) ZZ: Lepton

Figure 14: Event images for Z', W^+W^-, and ZZ channel, separately shown for calorimeter towers, tracks, and leptons.

[7] M. Cacciari, G. P. Salam, and G. Soyez, "The anti-k_t jet clustering algorithm," *JHEP*, vol. 04, p. 063, 2008.

[8] A. Butter *et al.*, "The Machine Learning landscape of top taggers," *SciPost Phys.*, vol. 7, p. 014, 2019.

[9] L. de Oliveira, M. Kagan, L. Mackey, B. Nachman, and A. Schwartzman, "Jet-images — deep learning edition," *JHEP*, vol. 07, p. 069, 2016.

[10] G. Kasieczka, T. Plehn, M. Russell, and T. Schell, "Deep-learning Top Taggers or The End of QCD?," *JHEP*, vol. 05, p. 006, 2017.