CLASSIFICATION OF HIGGS JETS AS DECAY PRODUCTS OF A RANDALL-SUNDRUM GRAVITON AT THE ATLAS EXPERIMENT

AVIV CUKIERMAN, ZIHAO JIANG

1. Introduction

In 2012, physicists at the Large Hadron Collider announced the discovery of the Higgs boson, which earned the moniker "God Particle" in the popular media due to its far-reaching theoretical implications. The Standard Model, which is by far the most successful model we have describing matter and its interactions, predicts that the Higgs boson exists and gives mass to quarks. The Higgs boson was the final particle predicted by the Standard Model that had not yet been discovered and so the discovery was a major triumph for the Standard Model. However, there are problems with the Standard Model. For instance, there is a question of why the mass of the Higgs itself is so small, when naive theoretical extensions of the Standard Model predict it should be much higher.

Among all the more sophisticated models which account for this problem, one model that yields detectable quantities is the Randall-Sundrum Model, which predicts there is a new particle, a Graviton, which can decay to two Higgs bosons . A search for such a particle requires distinguishing the Higgs bosons from background processes, mostly quark and gluon showering from quantum chromodynamic (QCD) effects. In this project, we use machine learning techniques to distinguish the signal of a Randall-Sundrum Graviton (RSG) decay to two Higgs from the QCD background, using kinematic variables that would be observed in the ATLAS detector at CERN. In particular, we look at the case where each of the Higgs bosons decay to a pair of bottom-antibottom ($b\bar{b}$) quarks.

2. Data & Features

The study uses Monte Carlo simulated data with full reconstruction of ATLAS detector response. For the signal, we assume the mass of the RSM Graviton to be 1000 GeV/c^2, consisting of 24779 events. For the background, we use ATLAS tuned QCD simulations consisting of 25944 unweighted events.

In a signal event we have two Higgs bosons, each of which decay to a $b\bar{b}$ pair of quarks. These quarks are are observed as collimated energy deposits in the detector, which are called jets. We know from physics that in fact the quark-antiquark pair will be detected with small angular separation in the detector, and so the quark jets form the sub-structure of a larger Higgs jet. From these energy deposits we can reconstruct the space-time four momenta of the two quarks and the Higgs bosons ($\rightarrow 3 * 4 = 12$ features). In addition, we have b-tagging variables constructed from other observables that describe the probability that the quarks are bottoms ($\rightarrow$ 2 features). So in total we have (12+2=14) features for each potential Higgs jet.

3. Models

3.1. **Naive Bayes.**

The Naive Bayes model holds the assumption that the probability of a set of features $\{F_n\}$ conditioned on a variable X is $p(F_1, ..., F_n|X) = \prod_i^n p(F_i|X)$ In our case, $X = 0, 1$ denoting whether the a jet is or is not a Higgs boson. The features are the continuous kinematic variables of the jets and the sub-jets. We model the conditional probabilities as a Gaussian $p(F_i|X) = \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp -\frac{(x-\mu_i)^2}{2\sigma_i^2}$, where the parameters μ and σ are calculated directly from the mean and RMS of the kinematic variables.

Heuristically, the features that are most relevant to this classification problem are the invariant mass and transverse momenta (pT) of the jets. Higgs jets have invariant mass around 125 GeV, while the QCD jets have smaller mass because the showering of gluon and quarks usually produces light mesons. The mass and energy of sub-jets (decay products) are predictive. The Higgs tend to decay to heavy-flavor quarks like Bottom and Charm, while the sub-jets in QCD come from radiation of light particles.

For the Naive Bayes algorithm, we consider the parameters in the following order: mass of the jet, pT of the jet, mass of the two sub-jets, pT of the two sub-jets, and b-tagging score of the two sub-jets. We add these variables to our feature set one by one, and each time we re-run the learning algorithm and testing for 100 iterations. For each iteration, we use 70% of the signal and background to train the Gaussian parameters and test the prediction on the remaining 30% of the data set. We take the training error and testing error as the mean of these 100 iterations and RMS as the uncertainties.

The results of this algorithm can be seen in Figure 1. The tagger training and testing errors quickly get down to percent level as we increase the size of the feature set. For the signal, the errors are getting significantly smaller as we consider more features. However including the sub-jets information doesn't improve the performance of tagging background. We also note that that after including the first two features (mass and pT of the Higgs jet), almost all of the discriminative power is achieved.

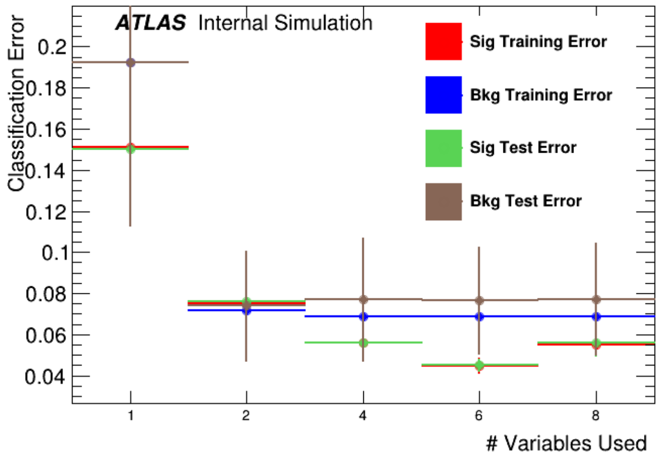

FIGURE 1. Naive Bayes Driven Higgs Tagger. With 8 features, the signal error was 5% and the background error was 8%.

3.2. **Logistic Regression.**

In logistic regression we have a hypothesis of the form

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1+\exp(-\theta^T x)}$$

Where x is a $(D+1)$-dimensional vector with $x_0 = 1$ and $x_{1,\dots,D}$ the features, and $\theta \in \mathbb{R}^{D+1}$ are the parameters of the model. The prediction for $y \in \{0,1\}$ is $1\{h_\theta(x) > 0.5\}$.

We split the data into 70% training set and 30% testing set. We optimize the parameters of the regression via batch gradient ascent on the log-likelihood function on the training set. I.e. we run the following algorithm:
Repeat for N iterations:

$$\theta_{j+1} := \theta_j + \alpha \sum_{i=1}^{m} (y^{(i)} - h_\theta(x^{(i)}))x_j^{(i)}$$

Where in the algorithm we set $\alpha = 1$, and $x^{(1)}, \dots, x^{(m)}$ is the training set.

We then calculate the error in the testing set (and training set) by calculating

$$\epsilon^{\text{Test/Train}}_{\text{Signal},N} = \frac{\sum_{i=1}^{m^{\text{Test/Train}}} 1\{y^{(i)} = 1\}1\{h_\theta(x^{(i)}) < 0.5\}\}}{m^{\text{Test/Train}}_{\text{Signal}}}$$

$$\epsilon^{\text{Test/Train}}_{\text{Background},N} = \frac{\sum_{i=1}^{m^{\text{Test/Train}}} 1\{y^{(i)} = 0\}1\{h_\theta(x^{(i)}) > 0.5\}\}}{m^{\text{Test/Train}}_{\text{Background}}}$$

$$\epsilon^{\text{Test/Train}}_{N} = \epsilon^{\text{Test/Train}}_{\text{Signal},N} + \epsilon^{\text{Test/Train}}_{\text{Background},N}$$

For each N.

Sometimes $\epsilon^{\text{Train}}_M > \epsilon^{\text{Train}}_N$ for $M > N$ as the algorithm moves out of local minima. So we look at

$$M^* = \text{argmin}_{i\in\{1,\dots,N\}}(\epsilon^{\text{Train}}_i)$$

Since in practice we can choose whatever θ_j minimizes the training error over the course of the algorithm, which isn't necessarily θ_N.

Running the algorithm with all the features, we get convergence of $\epsilon^{\text{Train}}_{M^*,\text{min}}$ at around $N = 1000$ iterations, which can be seen in Fig. 2. In order to compare across algorithms, we are concerned with the minimum total error when $\epsilon_{\text{Signal}} \sim 0.05$. So we define similarly

$$M^*_{5\%} = \text{argmin}_{i\in S}(\epsilon^{\text{Train}}_i)$$

$$S = \{i : |\epsilon^{\text{Train}}_{\text{Signal},i} - 0.05| < 0.01\}$$

The results after $N = 5000$ iterations are shown in Table 1.

	$\epsilon^{\text{Train}}_{\text{Signal},M}$	$\epsilon^{\text{Train}}_{\text{Background},M}$	$\epsilon^{\text{Train}}_{M}$	$\epsilon^{\text{Test}}_{\text{Signal},M}$	$\epsilon^{\text{Test}}_{\text{Background},M}$	$\epsilon^{\text{Test}}_{M}$
$M = M^*$	0.0291	0.1135	0.1426	0.0258	0.1129	0.1387
$M = M^*_{5\%}$	0.0501	0.0966	0.1467	0.0471	0.0940	0.1411

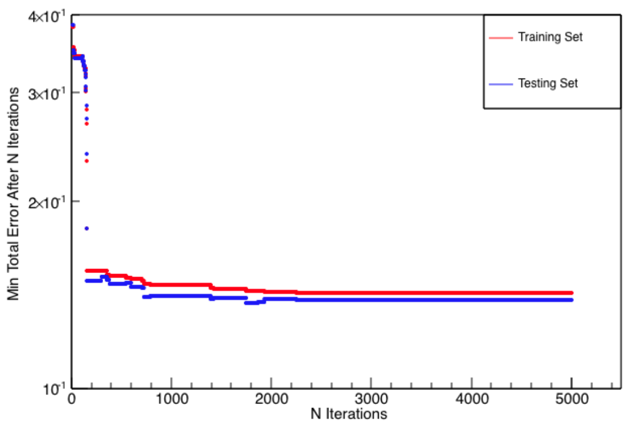

FIGURE 2. $\epsilon^{\text{Train/Test}}_{N,\text{min}}$ as a function of N using all the features. By about $N = 1000$, the algorithm has converged to within a few percent level of the $N \to \infty$ limit.

Table 1. Results of logistic regression algorithm.

3.3. **k-clustering.** We examine a significantly modified k-means clustering algorithm designed by us. First, we run an ordinary unsupervised clustering algorithm on the training data:
1. Choose k centroids $\mu_1, ..., \mu_k$ randomly from the data
2. For every i, set $c(i) = \text{argmin}_j ||x^{(i)} - \mu_j||^2$
3. For every j, set $\mu_j = \sum_{i=1}^m 1\{c(i) = j\} * x^{(i)}$
4. Repeat the last two steps until convergence
5. Repeat steps 1-4 a few times to globally minimize the cost function $J = \sum_{i=1}^m ||x^{(i)} - \mu_{c(i)}||^2$.

Then we supervise the unsupervised algorithm. For each cluster j we look at what fraction of the data classified in that cluster is signal:

$$f_j = \frac{\sum_{i=1}^m 1\{y^{(i)} = 1\}1\{c(i) = j\}}{\sum_{i=1}^m 1\{c(i) = j\}}$$

We then set a signal threshold s and background threshold b to label signal ($f_j > s$) and background ($f_j < b$) centroids, respectively. Then using only the centroids that passed the signal or background threshold cuts (i.e. the number of clusters used in the testing phase $k_{\text{Test}} \le k$), we cluster and label the testing data to get signal and background identification efficiencies:

$$e^{\text{Test/Train}}_{\text{Signal}}(s) = \frac{1}{m^{\text{Test/Train}}_{\text{Signal}}} \sum_{i=1}^{m^{\text{Test/Train}}_{\text{Signal}}} y^{(i)} 1\{f_{c^{(i)}} > s\}$$

$$e^{\text{Test/Train}}_{\text{Background}}(b) = \frac{1}{m^{\text{Test/Train}}_{\text{Background}}} \sum_{i=1}^{m^{\text{Test/Train}}_{\text{Background}}} (1 - y^{(i)}) 1\{f_{c^{(i)}} < b\}$$

And the error $\epsilon = 1 - e$.

By varying s and b, we get a ROC curve, which is shown in Figure 3.

3.4. **Multivariate Analysis.** There are correlations between different kinematics variables in the dataset we consider. For instance, both the Higgs jets and QCD background jets will split to two sub-jets of the same amount of energy. Therefore, the pT of the sub-jets are roughly the same. To account for such correlations we implement more advanced multivariate analysis tools BDT (Boosted Decision Tree) and k-NN (kth nearest neighborhood) which are widely used in ATLAS Collaboration.

3.4.1. *Boosted Decision Tree.* The Boosted Decision Tree (BDT) is an ensemble of weak classifiers called decision trees. The algorithm involves two steps: (1) building a forest of decision trees, and (2) adaptive boosting. A decision tree is a multi-layer binary classifier. The training starts with the root node and splits to two subsets, each of which searches for the best variable and its corresponding cut based on calculation of the Gini Index, defined as $p(1 - p)$, where p is the purity of the sample. If in a sample, signal and background are mixed equally then $p = 0.5$ and the Gini Index is maximized, while it falls to zero if the sample is entirely signal or background. In this search step, we partition each variable into 20 equal segments and decide the cut based on the maximum change between the parent node Gini Index

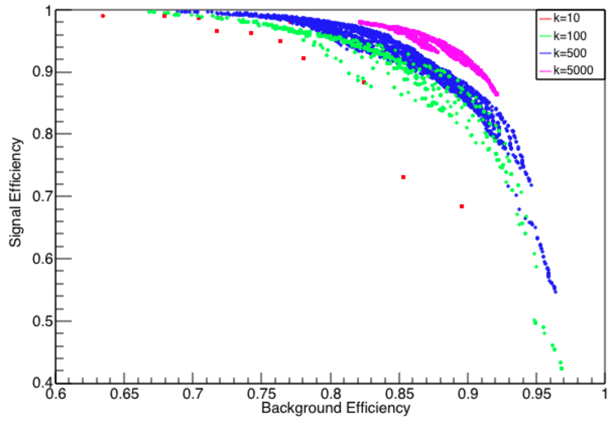

FIGURE 3. The ROC curves for the the modified k-clustering algorithm. The model improved with increasing numbers of clusters, but low data statistics prevented making more clusters.

and the sum of the daughter nodes Gini Indices. The tree keeps splitting until the resulting split subsets have fewer than 200 data points. Once we obtain 200 such weak classifiers h_i, we pass them to the boosting step.

The goal of the boosting is to construct a strong classifier as a linear combination of the 200 weak classifiers we obtained, $f(x) = \sum_{t=1}^{T=200} \alpha_t h_t(x)$. For $b \in \{-1, 1\}$, we come up with a prediction model $H_b(x) = 1\{f(x) > b\}$. Given m training samples, the α's are chosen such that $\epsilon_b = \frac{1}{m}\sum_i 1\{H_b(x_i) \neq y_i\}$ is minimized. The minimization scheme we use is adaptive boosting, which estimates and minimizes an upper bound of ϵ. By varying b, we get a ROC curve, which can be seen in Figure 4.

The BDT algorithm is also useful in determining which variables are most useful for separating signal and background. We calculated the discriminating power of each feature as the number of splits occurred for that variable weighted by the gain-squared and number of events in it. The gain is calculated as the change in information entropy H from a previous stage to a new stage with more information. As seen in Table 2, we find that the Higgs jet transverse momentum and mass have the most discriminating power.

Variable	Importance	Variable	Importance
jet pT	0.19	leading sub-jet b-tagging score	0.13
jet mass	0.15	sub-leading sub-jet b-tagging score	0.12
sub-leading sub-jet mass	0.14	leading sub-jet mass	0.09
sub-leading sub-jet pT	0.13	leading sub-jet pT	0.07

Table 2. Importance calculated by BDT algorithm. Most indicative variables are the jet pT and jet mass.

3.4.2. *k-NN.* The philosophy of k-NN is neighborhood majority vote. The label of a point is decided by the labels of the majority of its neighborhood points. To formalize the problem, we need a metric to decide for two points to be close or neighbors. For this problem, we use the conventional Euclidean metric. Given a test point p and a set of features $\{f_1, f_2, ..., f_n\}$, we calculate the metric $R = \left(\sum_i^n \frac{1}{w_i^2}|a_i - b_{ij}|\right)^{\frac{1}{2}}$, where a_i and b_{ij} are the i^{th} feature value for p and j^{th} training point. Then we select the closest k training points (in our case, we use k = 20) and assign a probability of p being signal $P_s = \frac{k_s}{k}$, where k_s is the number of signal points of the k-nearest points. The weight factor w_i in the metric is the width of feature i for all the data. By varying the cut on the signal probability, we come up with a ROC curve, which can be seen in Figure 4.

4. RESULTS

The results are detailed in the Models section. But we wish to compare the models used. Table 3 shows the minimum test background error for each model with test signal error $\sim 5\%$.

Model	Background Error
Naive Bayes	0.08
Logistic Regression	0.0966
Modified k-clustering	0.1129
BDT	0.0265
k-NN	0.0473

Table 3. Results of the models used here for test signal error $\sim 5\%$.

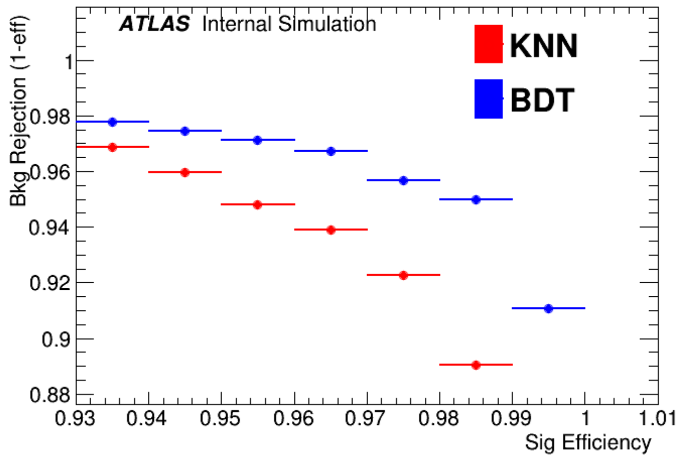

Figure 4. ROC curves for BDT and k-NN algorithms. The BDT algorithm outperforms k-NN at all signal efficiencies.

As we can see, the BDT algorithm is the best performing.

5. Discussion

We examine both linear and non-linear classification algorithms. In general we find the non-linear classifiers to perform better than the linear ones. The best performing linear classifier was Naive Bayes, and the best performing classifier overall was BDT. We note that the ATLAS collaboration has generally agreed on using BDT to classify signal versus background in a wide variety of problems, and our results support that decision.

We also designed a modified k-clustering algorithm. Our new algorithm was the worst performing of all the ones we examined. However it was interesting to see that an algorithm with unsupervised components could still do a decent job of classification.

We also analyzed and measured the importance of each feature used in our algorithms. Both our Naive Bayes algorithm and BDT find that the two most important features are the Higgs jet mass and transverse momentum.

6. Future Work

Most of the algorithms discussed here have seen years of research and development. The k-clustering algorithm developed here, on the other hand, shows some promise, even though it was the worst performing of all the algorithms. With more time it would be interesting to look at this algorithm from a theoretical standpoint to understand its limitations and under what model it performs best. If we were to work on this project for a few more months, we would also probably look at other algorithms like SVM.

7. Conclusion

We find that machine learning algorithms can be very powerful in distinguishing Higgs signal from background in simulated data from ATLAS. Its particularly interesting that the algorithms themselves had no knowledge of the physics, only of the data and training classifications. The prospect of discovering or ruling out a Randall-Sundrum graviton via decay to two Higgs bosons at ATLAS seems feasible given the power of the classification algorithms discovered here. From a wider perspective, machine learning algorithms can be used in many physics analyses other than the specific one studied here, such as searches for supersymmetric particles, dark matter, or exotic particles like axions. We believe that the future of physics done at experiments like ATLAS inevitably lies with the use of machine learning techniques like the ones presented here.

8. References

[1] ATLAS Collaboration (2012). "Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC. Phys. Lett. B 716 (39)
[2] M. Cacciari, G. Salam, G. Soyez (2008), "The Anti-Kt jet clustering algorithm", JHEP 0804:063
[3] A. Hoecker et al. (2007). "TMVA - Toolkit for Multivariate Data Analysis, PoS ACAT 040
[4] A. Ng. "CS229 Lecture Notes, Retrieved from CS 229 Autumn 2014 Course Website, http://cs229.stanford.edu/materials.
[5] L. Randall, R. Sundrum (1999). "Large Mass Hierarchy from a Small Extra Dimension. Physical Review Letters 83 (17)
[6] T. Sjostrand, S. Mrenna and P. Skands (2007). "A Brief introduction to PYTHIA 8.1, arXiv:0710.3820