Extracting more from boosted decision trees: A high energy physics case study

Vidhi Lalchand
Department of Physics
University of Cambridge, UK
The Alan Turing Institute
`vr308@cam.ac.uk`

Abstract

Particle identification is one of the core tasks in the data analysis pipeline at the Large Hadron Collider (LHC). Statistically, this entails the identification of rare signal events buried in immense backgrounds which mimic the properties of the former. In machine learning parlance, particle identification represents a classification problem characterised by overlapping and imbalanced classes. Boosted decision trees (BDTs) have had tremendous success in the particle identification domain but more recently have been overshadowed by deep learning (DNNs) approaches. This work proposes an algorithm to extract more out of standard boosted decision decision trees by targeting their main weakness, susceptibility to overfitting. This novel construction harnesses the meta-learning techniques of boosting and bagging simultaneously and performs remarkably well on the ATLAS Higgs (H) to tau-tau data set [A$^+$14] which was the subject of the 2014 Higgs ML Challenge [ABCG$^+$15]. While the decay of Higgs to a pair of tau leptons was established in 2018 [c$^+$17] at the 4.9σ significance based on the 2016 data taking period, the 2014 public data set continues to serve as a benchmark data set to test the performance of supervised classification schemes. We show that the score achieved by the proposed algorithm is very close to the published winning score which leverages an ensemble of deep neural networks (DNNs). Although this paper focuses on a single application, it is expected that this simple and robust technique will find wider applications in high energy physics.

1 Introduction

Classifier learning is exacerbated in the presence of overlapping and imbalanced class distributions and specialized attention is needed to achieve reasonable performance. These pathologies are often witnessed in real world data sets such as those related to fraud detection, diagnosis of rare diseases or detecting previously unseen phenomena. In the binary classification context, overlapping classes are characterised by the presence of regions in the feature space with a high density of points belonging to both classes. Class imbalance on the other hand is characterised by significantly different prior class probabilities; there is a minority class with much fewer training samples than the majority class. A significant amount of literature targeting these challenges use ensemble classifiers to boost the performance of simple algorithms by using them iteratively in multiple stages or training multiple copies of the same algorithm on simulated re-sampled sets of the training data. The former is called *boosting* and the latter is called *bagging*. These meta-learning techniques are typically used independently as boosting targets reduction in bias and bagging targets reduction in variance. The algorithm proposed here is a variant of BDTs that jointly harness both boosting and bagging and perform remarkably well when the data exhibit class overlap and class imbalance.

2 Multivariate Techniques in HEP

The search for interesting signatures in the extremely high volumes of data generated by LHC experiments requires more than just veteran physics ability [RWR+18]. The use of machine learning techniques to classify particles and events has been mainstream for the past two decades.

Neural nets had a timid start in high energy physics in the late 80s but since the advent of deep learning, NNs have started to be used more broadly [BSW14]. This is probably both due to an increased complexity of the data to be analysed and the demand for non-linear techniques. Deep learning is capable of discovering intricate structure in data by learning abstract representations of data that amplify the aspects needed for discrimination [LBH15]. While state-of-the-art results have been reported on several classification tasks, deep learning methods typically require a large amount of data to train and there is a high computational overload. BDTs have had tremendous success as a multivariate analysis tool in HEP. Pioneering works like [RYZ+05] [YRZ05] have used carefully tuned BDTs for robust classification as a workhorse for particle identification.

This short paper is organized as follows, section 3 provides a brief summary on boosting and bagging, 4 presents a variant of the vanilla boosted decision tree and discusses its performance edge in classification tasks which exhibit overlap and imbalance. Section 5 summarises the Higgs to tau-tau channel dataset along with experimental results.

3 Meta-Learning techniques: Boosting and Bagging

One of the most fundamental expressions of boosting is the AdaBoost (Adaptive Boosting) algorithm [FS+96]; typically used with decision trees as the base learners. Boosting works by training base learners [1] to build a master learner which is significantly better at the task. The decision rule for the master learner is a weighted combination of the base learners outcomes and the weights are usually a function of the weak learners accuracy. A boosted classifier with J stages takes the form,

$$M_J(\mathbf{x}) = \sum_{j=1}^{J} \alpha_j h_j(\mathbf{x}) \tag{1}$$

where $h_j(\mathbf{x})$ are the base learners and the coefficients α_j denote the confidence in base learners. The training happens in stages and each training observation is associated with a weight. At the end of each stage the weight on misclassified samples is increased in a weight update step thereby allowing the classifier to focus on the hard to classify events in subsequent stages. At each iteration it improves accuracy by classifying few more points correctly without disrupting the correct classifications from previous stages. This provides some intuition about how boosting achieves low bias. However, this very property makes boosting prone to overfitting and makes it a high variance classifier. Boosting with several stages of tree learning can rapidly overfit as trees are low bias-high variance learners. Over-fitting control is one of the main challenges in using this technique.

Bagging was a technique popularised by Leo Brieman who first published the idea in 1996 [Bre96]. The bagging technique helps stabilise classifiers by training a single classifier on multiple bootstrap samples (sampling with replacement) of the training data set. The size of the bootstrap sample usually is the same as the original data set. Due to sampling with replacement a bootstrap sample most likely has repeated samples, this increases the predictive force of the base learner on those samples. Brieman notes:

'*The vital element is the instability of the prediction method. If perturbing the learning set can cause significant changes in the predictor constructed, then bagging can improve accuracy.*'

Random Forests exemplify the bagging principle. A random forest is a collection of tree learners each trained on a bootstrap sample of the training data. There are two sources of randomness in a random forest, (1) the formation of bootstrap samples and (2) a random selection of features considered at each node for splitting [Ho98]. An extremely randomized tree (ET) is one of the newer incarnations of the random forest method [GEW06] which introduce further randomization in the tree construction process. While random forests attempt to create diversified trees by using a bootstrap samples, ETs work like random forests but take the randomization one step further by choosing a random split

---

[1]can classify samples better than random guessing

point at each node rather than searching for the best split. This ensures the creation of strongly **decorrelated** and **diverse** trees. From a computational view, ETs offer further advantage as they do not need to look for the most optimal split at each node. The algorithm proposed in section 4 uses a bagged ensemble of ETs as the base learners within a boosting framework. This nested construction is able to counteract overfitting which is a fundamental weakness of classical BDTs.

4 Boosting Extremely Randomized Trees (BXT)

At first the idea of using a randomized tree seems unintuitive as they are inferior to a tuned decision tree or a random forest which search for optimal split points, but it is precisely because of this that they serve as excellent weak learners. Bagging several extremely randomized trees alleviates overfitting and boosting ensures that their classification performance improves over stages. While it is true that BXT takes more stages of boosting to achieve the same performance of a classical BDT, it converges to a higher classification accuracy while the misclassification rate in a BDT saturates earlier. In the real world case study we show that this particular construction is superior to BDTs.

Algorithm 1 Boosting Extremely Randomized trees (BXT)

1: **Training**:
2: **Input**: Training set $\mathbf{D}$, primitive tree learner t, initial weights $\{w_i^1\}_{i=1}^N = \frac{1}{N}$
3: **Output**: Master learner $M(\mathbf{x}) = \sum_{j=1}^J \alpha_j T_j(\mathbf{x})$ where $T_j(\mathbf{x}) = agg\{t_1(\mathbf{x}) \ldots t_B(\mathbf{x})\}_j$ is a collection of extremely randomized trees trained on B bootstrap samples. $agg()$ refers to aggregation through majority vote.

4: **for all** $j = 1 \ldots J$ (stages) **do**
5:     **for all** $b = 1 \ldots B$ (bootstraps) **do**
6:         **Construct** a bootstrap sample $D_j^{(b)}$ (sample uniformly with replacement from $\mathbf{D}$)
7:         **Fit** tree learner t_b on sample $D_j^{(b)}$ using the Random Splitting Algorithm (see algorithm 2)
8:     **end for**
9:     **return** Set of extremely random fitted trees $\{t_1 \ldots t_B\}_j$ together serving as the base learner T_j for stage j.
        Compute coefficient $\alpha_j = \frac{1}{2} \ln \left(\frac{N_c}{N - N_c} \right)$ where N_c is the number of correctly classified samples for base learner T_j.
        Compute weight update step
        $w_i^{j+1} = w_i^j e^{\alpha_j \mathbf{1}(T_j(\mathbf{x}_i) \neq y_i)}$
10: **end for**

11: **Testing**:
12: **Input**: Test set $\mathbf{L} = \{\mathbf{x}_i^*\}_{i=1}^L$, full set of parameters for the master learner $\{\{\alpha_j\}_{j=1}^J, \{T\}_{j=1}^J\}$
13: **Output**: Predicted labels $\{\hat{y}_i\}_{i=1}^L$

14: **for all** $\mathbf{x}_i \in \mathbf{L}$ **do**
15:     **Predict** $\hat{y}_i = M(\mathbf{x_i}) = \sum_{j=1}^J \alpha_j T_j(\mathbf{x})$
16: **end for**
17: **return** $\{\hat{y}_i\}_{i=1}^L$

---

5 Results on ATLAS Higgs to tau-tau data

The ATLAS Higgs dataset from the CERN Open data portal [A$^+$14] has a total of 800K collision events (labelled s or b) which have been simulated by the high energy ATLAS simulator. The simulator encompasses the current best understanding of the physics underlying the signal (events which generated a Higgs particle) and background decays. The signal events mimic properties of the background events as is known to happen in real events. The taxonomy of the dataset is given in appendix section 7.2. For a more detailed description of the data set please refer to [ABCG$^+$15]. This data set is an example that encapsulates both class overlap and class imbalance.

A typical binary classifier $h : \mathbb{R}^d \rightarrow \{b, s\}$ calculates a discriminant function $f(\mathbf{x}) \in \mathbb{R}, \mathbf{x} \in \mathbb{R}^d$ which is a score giving small values to the negative class (background) and large values to the positive class (signal). One puts a threshold of choice θ (for this data set we choose a consistent cut-off

threshold at the 85th percentile, a threshold chosen by physics experts) on the discriminant score and classifies all samples below the threshold as belonging to the negative class (b or -1) and all samples with a score above the threshold as belonging to the positive class (s or $+1$) or the selection region.

The goal of the classification exercise is to maximise a physics inspired objective called the *Approximate Median Significance* (AMS) metric. One can think of the AMS as denoting a significance in terms of (σ). Given a binary classifier $h : \mathbb{R}^d \rightarrow \{b, s\}$, the AMS is given by, $\sqrt{2((\hat{s} + \hat{b})\ln(1 + \frac{\hat{s}}{\hat{b}}) - \hat{s})}$ where $\hat{s}$ and $\hat{b}$ are the number of signal and background events in the selection region.

Learning Algorithm	Selection Region	False Positives	AMS (σ)	No. of learners	Training time (sec.)
Boosted Decision Trees (BDT)	66789	5501	3.73311σ	20×100	729.28 (CPU)
Boosted Extremely Random Trees (BXT)	66876	5413	3.79361σ	20×100	983.26 (CPU)
Ensemble of Neural Nets (DNN) (winning)	N/A	N/A	3.8058σ	70	600 per NN (GPU)

Table 1: Performance metrics of tree ensembles along with winning solution. Our score is only marginally lower than the winning score [Mel15] at the 85th percentile, and BXT is much computationally cheaper to train versus a DNN (several hours on a CPU).
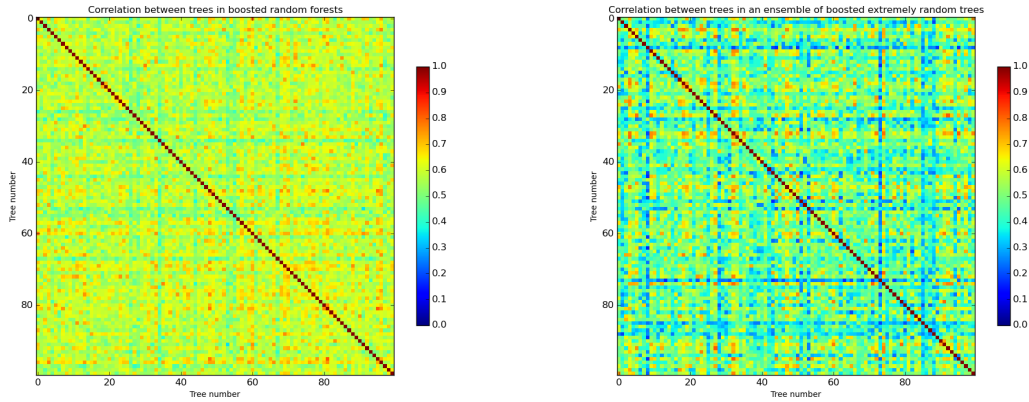

Figure 1: Correlation among trees in a boosted ensemble of decision trees (BDTs, left) and extremely randomized trees (BXT, right). The figures depicts the correlation between 100 primitive trees picked at random from the 2000 trees that comprise the BDT and BXT model (20 stages of boosting $\times$ 100 trees in each tree ensemble). The higher level of de-correlation between the outputs of BXT versus the BDT indicates that diversity in ensembles enhances classification performance.
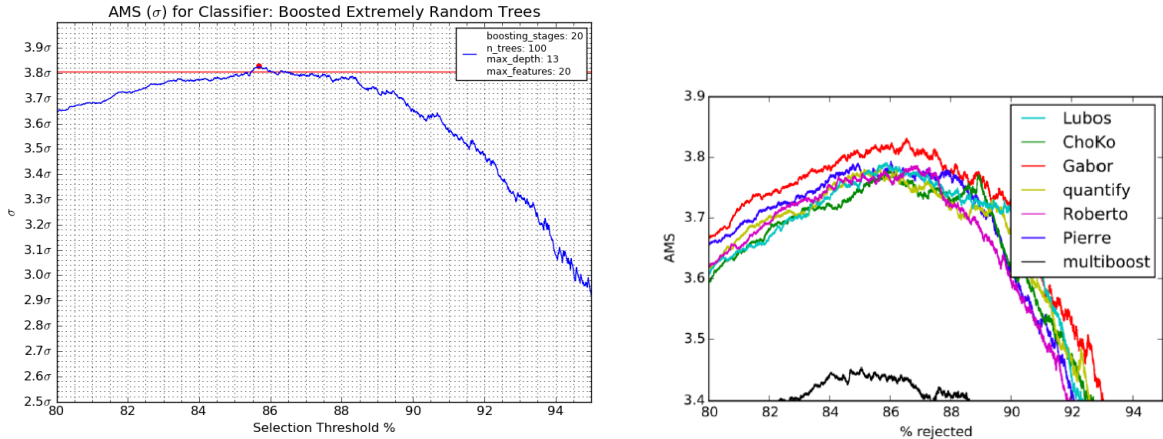

Figure 2: AMS curves as a function of cut-off threshold for the selection region for our algorithm (left) and other dominant algorithms (winning one in red)

4

6 Discussion

The compelling performance of a relatively simple algorithm based on boosting and extremely randomized trees hints that we can extract a lot more mileage from primitive learning algorithms by embedding them in novel architectures. The BXT algorithm shines a light on the strength of diversity in ensemble learning; essentially by making predictors less deterministic and more diverse we can jointly improve classification performance. The general idea of using bagged learners within a boosting framework can be utilized with any component learner (not just trees). While deep learning approaches are powerful non-linear pattern extractors, their use might be overkill in many classification settings including high energy physics domains. Further, models based on primitive learners like trees are *white-box* models; they are far more interpretable than DNNs.

In replacing BDTs with deep learning (DNNs), have we thrown the baby out with the bathwater?

Acknowledgments

References

[A$^+$14] ATLAS et al. ATLAS Higgs Challenge 2014. *CERN Open Data Portal*, 2014.

[ABCG$^+$15] Claire Adam-Bourdarios, Glen Cowan, Cécile Germain, Isabelle Guyon, Balázs Kégl, and David Rousseau. The Higgs Boson Machine learning challenge. In *JMLR : Journal of Machine Learning Research*, volume 42, pages 19–55, 2015.

[Bre96] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[BSW14] Pierre Baldi, Peter Sadowski, and Daniel Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature communications*, 5:4308, 2014.

[c$^+$17] CMS collaboration et al. Observation of the higgs boson decay to a pair of tau leptons with the cms detector. *arXiv preprint arXiv:1708.00373*, 2017.

[FS$^+$96] Yoav Freund, Robert E Schapire, et al. Experiments with a new boosting algorithm. In *ICML*, volume 96, pages 148–156, 1996.

[GEW06] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006.

[Ho98] Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE transactions on pattern analysis and machine intelligence*, 20(8):832–844, 1998.

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[Mel15] Gábor Melis. Dissecting the winning solution of the higgsml challenge. In *Cowan et al., editor, JMLR: Workshop and Conference Proceedings*, number 42, pages 57–67, 2015.

[RWR$^+$18] Alexander Radovic, Mike Williams, David Rousseau, Michael Kagan, Daniele Bonacorsi, Alexander Himmel, Adam Aurisano, Kazuhiro Terao, and Taritree Wongjirad. Machine learning at the energy and intensity frontiers of particle physics. *Nature*, 560(7716):41, 2018.

[RYZ$^+$05] Byron P Roe, Hai-Jun Yang, Ji Zhu, Yong Liu, Ion Stancu, and Gordon McGregor. Boosted decision trees as an alternative to artificial neural networks for particle identification. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 543(2-3):577–584, 2005.

[YRZ05] Hai-Jun Yang, Byron P Roe, and Ji Zhu. Studies of boosted decision trees for MiniBooNE particle identification. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 555(1):370–385, 2005.

7 Appendices

7.1 Random Splitting Algorithm

The random split generating algorithm from [GEW06] is summarized in 2.

Algorithm 2 Random Splitting algorithm

1: **Input**: Training set $\mathbf{D}$
2: **Output**: A split c_j which is a scalar value from a single feature vector of the training set $\mathbf{D}$

3: **Select** K features $a_1, \ldots, a_K$ at random from $\mathbf{D}$
4: **Select** K splits $\{c_1, \ldots, c_K\}$, one per feature for the K features chosen in the previous step; each c_i is selected at random from the range of the feature values $\forall i = 1, \ldots K$
5: Rank the splits c_i by a criterion say Q which gives a score $Q(\mathbf{D}, c_i) \in \mathbb{R}$ for each split.
6: Select $c_* = max_{i=1\ldots K} Q(\mathbf{D}, c_i)$
7: **return** c_*

7.2 Taxonomy of the Higgs to tau-tau dataset

Dataset	Total events	Background $|\mathcal{B}|$	Signal $|\mathcal{S}|$
Training	250,000	164,333	85,667
Cross Validation	100,000	65,975	34,025
Testing	450,000	296,317	153,683

Table 2: ATLAS Breakdown of datasets for the learning task

7.3 Derivation of AdaBoost

Consider a binary classification problem with input vectors $\mathbf{D} = \{(\mathbf{x}_1, y_1, w_1) \ldots (\mathbf{x}_N, y_N, w_N)\}$ with binary class labels $y_i \in \{-1, +1\}, \forall i = 1 \ldots N$.

At the start of the algorithm, the training data weights $\{w_i\}$ are initialized to $w_i^{(1)} = 1/N, \forall i = 1 \ldots N$. A base classifier $h_1(\mathbf{x}) : \mathbf{D} \to \{-1, +1\}$ that misclassifies the least number of training samples is chosen. Formally, $h_1(\mathbf{x})$ minimizes the weighted error function given by,

$$R(h_1) = \sum_{i=1}^{N} w_i \mathbf{1}(h_1(\mathbf{x}) \neq y_i) \tag{2}$$

where $\mathbf{1}$ is the indicator function.

After the first round of classification, the coefficient α_1 is computed that indicates the confidence in the classifier. It is chosen to minimize an exponential error metric given by,

$$E = \sum_{i=1}^{N} e^{y_i \alpha_1 h_1(\mathbf{x}_i)}$$
$$= \sum_{y_i \neq h_1(\mathbf{x}_i)} e^{\alpha_1} + \sum_{y_i = h_1(\mathbf{x}_i)} e^{-\alpha_1}$$

6

$$\frac{dE}{d\alpha_1} = \sum_{y_i \neq h_1(\mathbf{x}_i)} e^{\alpha_1} - \sum_{y_i = h_1(\mathbf{x}_i)} e^{-\alpha_1}$$

$$\Rightarrow \sum_{y_i \neq h_1(\mathbf{x}_i)} e^{\alpha_1} = \sum_{y_i = h_1(\mathbf{x}_i)} e^{-\alpha_i}$$

$$\Rightarrow (N - N_c)e^{\alpha_1} = N_c \frac{1}{e^{\alpha_1}}$$

where N_c is number of correctly classified samples by h_1

$$\Rightarrow e^{2\alpha_1} = \frac{N_c}{N - N_c}$$

$$\Rightarrow \alpha_1 = \frac{1}{2} \ln \left(\frac{N_c}{N - N_c} \right)$$

Denoting $\epsilon_1 = \dfrac{N - N_c}{N}$ as the error rate for h_1,

$$\alpha_1 = \frac{1}{2} \ln \left(\frac{1 - \epsilon_1}{\epsilon_1} \right) \tag{3}$$

The weight update equation at each stage is given by,

$$w_i^{(j+1)} = w_i^{(j)} e^{\alpha_j \mathbf{1}(h_j(\mathbf{x}_i) \neq y_i)}$$

The master learner $M_J(\mathbf{x})$ for a J stage classifier is given by,

$$M_J(\mathbf{x}) = \sum_{j=1}^{J} \alpha_j h_j(\mathbf{x}) \tag{4}$$