LHCb Topological Trigger: Approaches and Optimization

Tatiana Likhomanenko[1,2,3] tatiana.likhomanenko@cern.ch and Michael Williams[4] mwill@mit.edu

[1] National Research University Higher School of Economics (HSE), RU
[2] Yandex School of Data Analysis (YSDA), RU
[3] NRC "Kurchatov Institute", RU
[4] Massachusetts Institute of Technology, US

Abstract. The LHCb trigger system plays a key role in selecting events (proton-proton collisions) to store them into memory for further processing. It should choose several thousands events per second among all detected by LHCb. The main b-physics trigger algorithm (should select events with B-meson decays) used by the LHCb experiment is a so-called topological trigger. In the LHC Run 1, this trigger, which utilized a simple boosted decision tree algorithm, selected a nearly 100% pure sample of b-hadrons with a typical efficiency of 60-70%; its output was used in about 60% of LHCb papers. In the paper [1] we presented studies carried out to optimize the topological trigger for LHC Run 2. Trigger data have a specific structure and that is why necessary quality measure was proposed. In this paper we investigate the presense of noisy samples in data and propose several ways to reduce this noise. Anoter side of the trigger system is a real-time prediction operation: it should be very fast. In the paper [1] two approaches how to speedup a prediction operation and to preserve the quality as possible were considered: bonsai boosted decision tree format (used in Run 1), BBDT, and decision trees post prunning. In this paper we analyse behaviour of the cleaned up samples for these two approaches. As a result, we demonstrate that removing of noise can still improve reoptimized topological trigger on the Run 1 performance for a wide range of b-hadron decays.

Keywords: high energy physics, machine learning, LHCb trigger system

1 Introduction

The LHCb detector [2] is designed for studying beauty and charm (heavy flavour) hadrons produced in proton-proton collisions at the Large Hadron Collider (LHC) at CERN. At LHCb data were collected with the rate 40 MHz in Run 1 ($40 * 10^6$ proton-proton collisions, called events, per second). This amount cannot be store into memory. A trigger system hardware part, called Level-0 (L0), reduced the visible bunch crossing rate to 1 MHz at which the detector could be read out. Further, a flexible software High-Level Trigger (HLT) applied a range of more advanced selections that reduce the rate to about 5 kHz ($5 * 10^3$ events per second) for offline storage and processing. This configuration allowed LHCb to record the largest beauty and charm hadron samples at a very high signal purity. The performance of the Run 1 trigger is described in detail in [3], [4].

In Run 1 it was shown that complex, multivariate trigger selections were possible. The HLT processes few enough events that it is possible to perform the reconstruction of a collision (to reconstruct tracks, vertices, a topology, physical characteristics). There are many HLT lines dedicated to triggering on various types of events. The majority of analyses using b-hadrons at LHCb made use of the topological n-body trigger [5]. Most n-body hadronic B decays ($n \geq 3$) are only triggered on efficiently in LHCb by these lines. The topological n-body trigger is an inclusive trigger which combines successive 2,3 and 4-body track combinations. A novel boosted decision trees (BDT) multivariate selection [6] was used in Run 1.

In the [1] we studied upgrading of the topological trigger: a new scheme of the trigger processing, which includes a more widespread usage of machine learning, was proposed; the appropriate quality measure was considered (receiver operating characteristic, ROC, computed for events for all B decays). However, obtained algorithms cannot be used in real-time trigger data processing. That is why, in the paper [1] we compared two approaches how to speedup a prediction operation: bonsai boosted decision tree format [6] and decision trees post prunning.

In the previous analysis [1] simulated data for various B decays were used as signal-like samples and real data from the collider without any physics were used as background-like samples. In this paper we present studies connected with the fact that simulated samples contain noisy data. We propose several ways how to clean up the samples and analyse the quality of the two speedup approaches in this case. As a result, we demonstrate that removing of noise can still improve the reoptimized topological trigger on the Run 1 performance for a wide range of b-hadron decays.

2 Motivation

Data for the trigger system have a specific structure: each event consists of the several reconstructed secondary vertices; a set of all secondary vertices for all B decays are the training sample. After training an event will pass the trigger if at least one its secondary vertex will pass the trigger (passing the trigger means that a prediction for a sample is greater than some fixed threshold; a threshold is chosen in a way that the false positive rate value provides the trigger rate). And we are interested in the greatest true positive rate for each B-decay mode.

As background-like data we use real data from the collider without any physics (that is why chosing the trigger rate will be equivalent to chosing the false posive rate). It means that all reconstructed secondary vertices for each event in the background-like data don't contain any interesting decays. As signal-like data we use the proton-proton collisions simulated for each mode of B-decays.

For these data we really know that at least in one secondary vertex necessary decay happened. But it doesn't guaratee that all secondary vertices contain interesting B-decays.

Each event can contain up to several hundreds secondary vertices. Most of them are removed from the training data by several physical selections wchich are applyed before. In the training data each event contains up to 124 secondary vertices and the mean equals to 6. For the signal-like data one of these is a truly signal-like sample, but others can be noise.

3 Noise clean up approaches

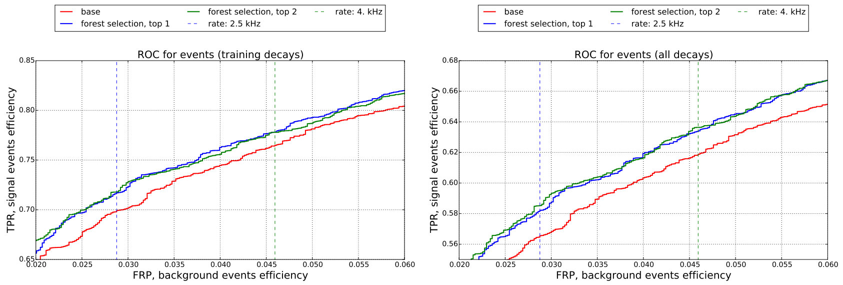
Specific data structure and physical properties lead to the necessity to clean up only signal-like samples. It is well-known [7] that random forest algorithm is stable with respect to noise and outliers. That is why random forest probability-predictions for signal-like data will be greater for truly signal-like samples and lower for output noise samples (samples which should be marked as background-like). The first idea of the random forest application is the following:

1. train random forest on the full training data;
2. choose for each signal-like event one secondary vertex which has the greatest random forest prediction (in the training data);
3. train original algorithm on the selected training signal-like samples and all training background-like samples.

This approach was applied to the topological trigger HLT2 (n-body, $n \geq 2$). Training data for this line contain only six B-decays and test data contain 20 B-decays including these training modes. In figures 1 this approach is presented in comparison with the original algorithm without any preselections of the secondary vertices. Here also another possible way, in which for each event two secondary vertices with the greatest predictions are taken on the second step, is shown.

Because data are the mixture of several B-decays and these modes have different recognition quality there are several possible approaches to clean up the data:

- Do preselection only for well recognized B-decays (in our case there are 4 well determined B-decay), Figure 2:
  1. train random forest on the full training data;
  2. in each well determined mode choose for each signal-like event one-two secondary vertices which have the greatest random forest predictions (in the training data); for each non-well determined modes in the trainig data take all secondary vertices;
  3. train original algorithm on the selected training signal-like samples and all training background-like samples.
- Use different random forests for preselections depending on the mode, Figure 3:



(a) (b)

Fig. 1: Comparison between the original algorithm (`base`) and the random forest preselection approach: **`forest selection, top 1`** — preselect for each event a secondary vertex with the greatest random forest prediction; **`forest selection, top 2`** — preselect for each event two secondary vertices with the greatest random forest predictions. ROC curves is produced only for six training modes on (a) and for all available B-decays on (b).

1. for each mode train random forest on the training B-decay mode samples and full background-like samples;
2. choose for each signal-like event one-two secondary vertices which have the greatest random forest (corresponding to the event's mode type) predictions (in the training data);
3. train original algorithm on the selected training signal-like samples and all training background-like samples.

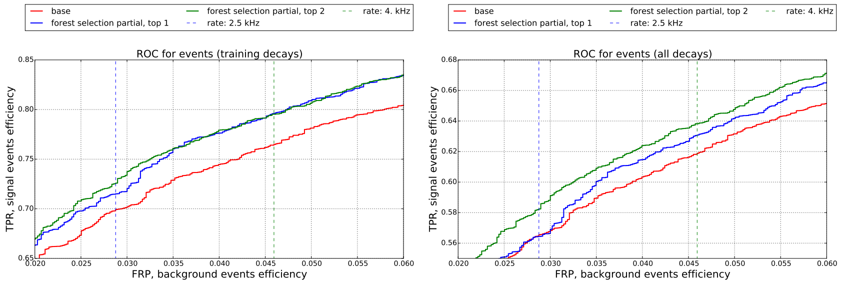
- Use different algorithms apart from random forests for preselections depending on the mode. We used XGBoost, Figure 4.

In all of these approaches it is seen that samples cleaning up allows to improve the signal efficiency on 1.5-2% for the considered output rates. And the ROC curve is strictly higher for these approaches. All of these models are presented together in Figure 5.

4 Speedup approaches with cleaned up samples

The trigger system is an online system and that is why speedup approaches were investigated in [1]. For these approaches we obtained some quality reduction. What is happened with the models, in which we use the noise cleaning up preprocessing, if speedup approaches are applyed?

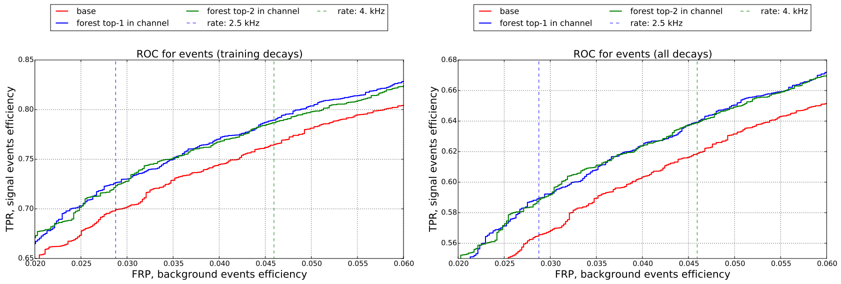
For both approaches, BBDT format and post prunning, noise cleaning up preprocessing also provides 1.5-2% signal efficiency improvement for the considered output rates. This result is shown in Figure **??**.



(a)

(b)

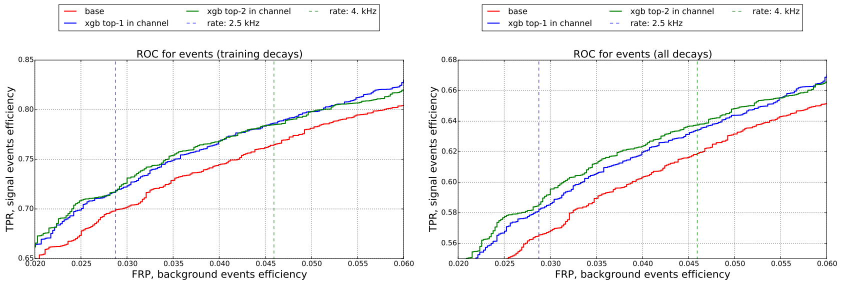
Fig. 2: Comparison between the original algorithm (`base`) and the random forest preselection approach: `forest selection partial, top 1` , `forest selection partial, top 2`. ROC curves is produced only for six training modes on (a) and for all available B-decays on (b).



(a)

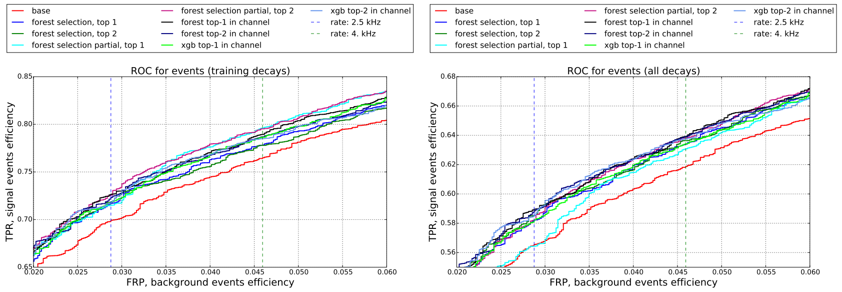
(b)

Fig. 3: Comparison between the original algorithm (`base`) and the random forest preselection approach: `forest top-1 in channel`, `forest top-2 in channel`. ROC curves is produced only for six training modes on (a) and for all available B-decays on (b).



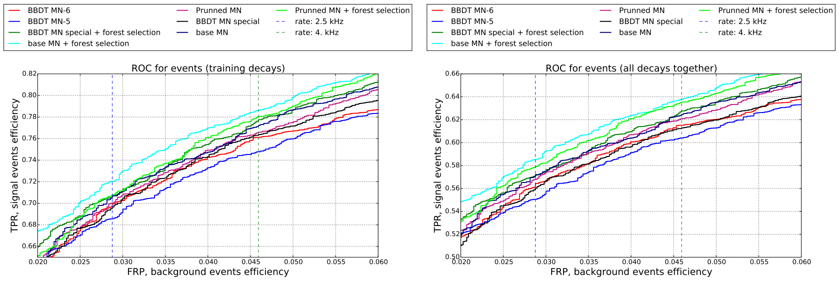
(a) (b)

Fig. 4: Comparison between the original algorithm (`base`) and the random forest preselection approach: `xgb top-1 in channel`, `xgb top-2 in channel`. ROC curves is produced only for six training modes on (a) and for all available B-decays on (b).



(a) (b)

Fig. 5: Comparison between the original algorithm (`base`) and all proposed noise cleaninig up approaches. ROC curves is produced only for six training modes on (a) and for all available B-decays on (b).



(a)

(b)

Fig. 6: Comparison between original algorithms, bbdt and prunning approaches with noise cleaning up and without. Here we used one of the cleaning up models: different random forests for preselections depending on the mode. ROC curves is produced only for six training modes on (a) and for all available B-decays on (b).

5 Conclusion

A simple idea about the simulated event structure allows to assume that simulated samples almost consist of noise. In the paper we demonstrated that this noise presence exists and proposed several ways how to clean up the samples and obtain the 1.5-2% signal efficiency improvement for the considered otput rates. Moreover, we obtained strictly higher the ROC curves for our approaches.

Notes and Comments. The part of the topological trigger analysis connected with the machine learning studies presented in this paper and [1] are open source and can be found on the github (`https://github.com/tata-antares/LHCb-topo-trigger`).

References

1. Likhomanenko T., Ilten P., Khairullin E., Rogozhnikov A., Ustyuzhanin A., Williams M.: LHCb Topological Trigger Reoptimization. Journal of Physics: Conference Series 664 (2015) 082025 [arXiv:1510.00572]
2. A. A. Alves Jr. et al.: The LHCb Detector at the LHC, JINST 3, S08005 (2008)
3. Aaij, R., *et al.* [LHCb Trigger Group]: The LHCb trigger and its performance. JINST **8** P04022 (2013) [arXiv:1211.3055]
4. Aaij R. et al.: Performance of the LHCb High Level Trigger in 2012, J. Phys., Conf. Ser. 513, 012001 (2014).
5. Gligorov V., Thomas C., Williams M.: The HLT inclusive B triggers. Technical Report LHCb-PUB-2011-016. CERN-LHCb-PUB-2011-016. LHCb-INT-2011-030, CERN, Geneva, Sep 2011. LHCb-INT-2011-030.

6. Gligorov V., Williams, M.: Efficient, reliable and fast high-level triggering using a bonsai boosted decision tree. JINST **8**, P02013 (2013) [arXiv:1210.6861]
7. Breiman, L.: Random forests. Machine learning, 45(1), 5-32 (2001).
8. Gulin, A., Kuralenok, I., Pavlov, D.: Winning the transfer learning track of Yahoo's Learning to Rank Challenge with YetiRank. JMLR: Workshop and Conference Proceedings 14 (2011) 63 (Yandex MatrixNet).