Multiboost: a multi-purpose boosting package

D. Benbouzid, Róbert Busa-Fekete, N. Casagrande, F.-D. Collin, Balázs Kégl

Journal of Machine Learning Research 13 (2012) 549-553 Submitted 8/11; Published 3/12

MultiBoost: A Multi-purpose Boosting Package

Djalel Benbouzid[1,2] DJALEL.BENBOUZID@GMAIL.COM
Róbert Busa-Fekete[1,3] BUSAROBI@GMAIL.COM
Norman Casagrande[4] NORMAN@WAVII.COM
François-David Collin[1] FRADAV@GMAIL.COM
Balázs Kégl[1,2] BALAZS.KEGL@GMAIL.COM

[1] *Linear Accelerator Laboratory, University of Paris-Sud, CNRS Orsay 91898, France*
[2] *Computer Science Laboratory, University of Paris-Sud, CNRS Orsay 91405, France*
[3] *Research Group on Artificial Intelligence of the Hungarian Academy of Sciences and University of Szeged, Aradi vértanúk tere 1., H-6720 Szeged, Hungary*
[4] *wavii.com*

Editor: Sören Sonnenburg

Abstract

The MultiBoost package provides a fast C++ implementation of multi-class/multi-label/multi-task boosting algorithms. It is based on AdaBoost.MH but it also implements popular cascade classifiers and FilterBoost. The package contains common multi-class base learners (stumps, trees, products, Haar filters). Further base learners and strong learners following the boosting paradigm can be easily implemented in a flexible framework.

Keywords: boosting, AdaBoost.MH, FilterBoost, cascade classifier

1. Introduction

AdaBoost (Freund and Schapire, 1997) is one of the best off-the-shelf learning methods developed in the last fifteen years. It constructs a classifier in an incremental fashion by adding simple classifiers to a pool, and uses their weighted "vote" to determine the final classification. AdaBoost was later extended to multi-class classification problems (Schapire and Singer, 1999). Although various other attempts have been made handle the multi-class setting, AdaBoost.MH has become the gold standard of multi-class boosting due to its simplicity and versatility.

Despite the simplicity and the practical success of the AdaBoost, there are relatively few off-the-shelf implementations available in the free software market. Whereas binary AdaBoost with decision stumps is easy to code, multi-class AdaBoost.MH and complex base learners are not straightforward to implement efficiently. The MultiBoost software package is intended to fill this gap. Its main boosting engine is based on the AdaBoost.MH algorithm of Schapire and Singer (1999), but popular cascade classifiers (VJCascade (Viola and Jones, 2004), SoftCascade (Bourdev and Brandt, 2005)) and FilterBoost (Bradley and Schapire, 2008) have also been implemented. The package includes common multi-class base learners (real and nominal valued decision stumps, trees, products (Kégl and Busa-

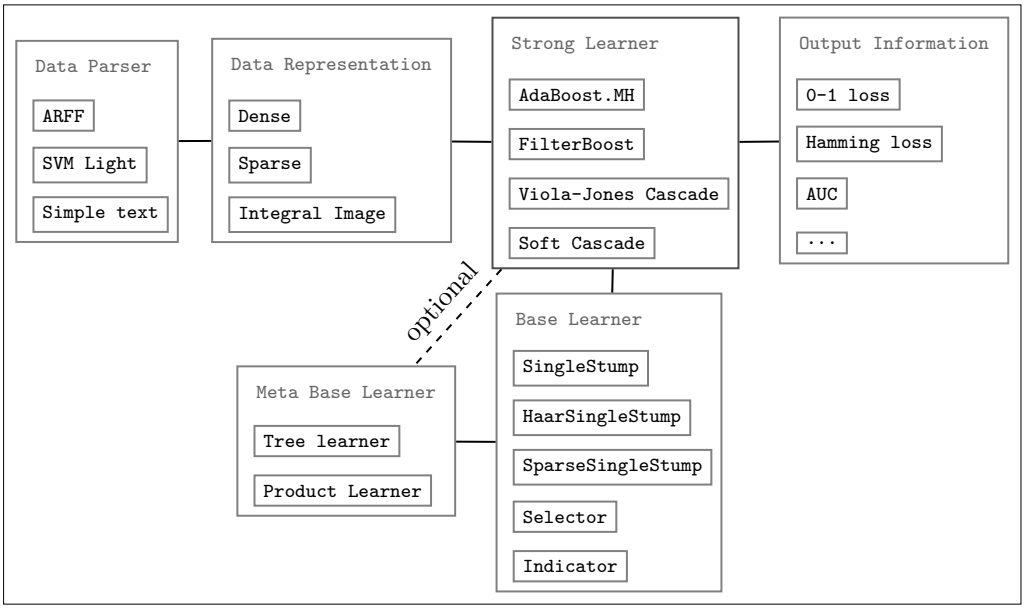

Figure 1: The architecture of MultiBoost

Fekete, 2009), and Haar filters), but the flexible architecture makes it simple to add new base learners without interfering with the main boosting engine. MultiBoost was designed in the object-oriented paradigm and coded in C++, so it is fast and it provides a flexible base for implementing further modules.

The rest of this paper is organized as follows. Section 2 describes the general architecture and the modules of MultiBoost. Section 3 deals with practical issues (website, documentation, licence), and Section 4 describes some of our results obtained on benchmark data sets and in data mining challenges.

2. The Architecture

MultiBoost was implemented within the object-oriented paradigm using some design patterns. It consists of several modules which can be changed or extended more or less independently (Figure 1). For instance, an advanced user can implement a data-type/base-learner pair without any need to modify the other modules.

2.1 Strong Learners

The strong learner[1] calls the base learners iteratively, stores the learned base classifiers and their coefficients, and manages the weights of the training instances. The resulting classifier is serialized in a human-readable XML format that allows one to resume a run after it was stopped or crashed. MultiBoost implements the following strong learners:

1. The name originally comes from the boosting (PAC learning) literature. Here, we use it in a broader sense to mean the "outer" loop of the boosting iteration.

- AdaBoost.MH (Schapire and Singer, 1999): a multi-class/multi-label/multi-task version of AdaBoost that learns a "flat" linear combination of vector-valued base classifiers.
- FilterBoost (Bradley and Schapire, 2008): an online "filtering" booster.
- VJCascade (Viola and Jones, 2004): an algorithm that learns a cascade classifier tree by running AdaBoost at each node.
- SoftCascade (Bourdev and Brandt, 2005): another cascade learner that starts with a set of base classifiers, reorders them, and augments them with rejection thresholds.

2.2 Base Learners

MultiBoost implements the following base learners.

- The Stump learner is a one-decision two-leaf tree learned on real-valued features. It is indexed by the feature it cuts and the threshold where it cuts the feature.
- Selector is a one-decision two-leaf tree learned on nominal features. It is indexed by the feature it cuts and the value of the feature it selects.
- Indicator is similar to Selector but it can select several values for a given feature (that is, it can *indicate* a subset of the values).
- HaarStump is a Stump learned over a feature space generated using rectangular filters on images.
- Tree is a *meta* base learner that takes any base learner as input and learns a vector-valued multi-class decision tree using the input base learner as the basic cut.
- Product is another meta base learner that takes any base learner as input and learns a vector-valued multi-class decision product (Kégl and Busa-Fekete, 2009) using the input base learner as terms of the product.

2.3 The Data Representation

The multi-class data structure is a set of observation-label pairs, where each observation is a vector of feature values, and each label is a vector of binary class indicators. In binary classification, we also allow one single label that indicates the class dichotomy. In single-label multi-class classification, only one of the K labels is 1 and the others are -1, but the framework also allows multi-label classification with several positive classes per instance. In addition, multi-task classification can be encoded by letting each label column represent a different task. We implement a sparse data representation for both the observation matrix and the label matrix. In general, base learners were implemented to work with their own data representation. For example, SparseStump works on sparse observation matrices and HaarStump works on an integral image data representation.

2.4 The Data Parser and the Output Information

The training and test sets can be input in the attribute-relation file format (ARFF), in the SVMLight format, or using a comma separated text file. We augmented the first

two formats with initial label weighting, which is an important feature in the boosting framework (especially in the multi-class/multi-label setup).

In each iteration, MultiBoost can output several metrics (specified by the user), such as the 0-1 error, the Hamming loss, or the area under the ROC curve. New metrics can also be implemented without modifying other parts of the code.

2.5 LazyBoost and BanditBoost

When the number of features is large, *featurewise* learners (Stump, Selector, and Indicator) can be accelerated by searching only a subset of the features in each iteration. MultiBoost implements two options, namely, LazyBoost (Escudero et al., 2000), where features are sampled randomly, and BanditBoost (Busa-Fekete and Kégl, 2010), where the sampling is biased towards "good" features learned using a multi-armed bandit algorithm.

3. Documentation and License

The code of MultiBoost has been fully documented in Doxygen. It is available under the GPL licence at `multiboost.org`. The website also provides documentation that contains detailed instructions and examples for using the package along with tutorials explaining how to implement new features. The documentation also contains the pseudo-code of the multi-class base learners implemented in MultiBoost.

4. Challenges and Benchmarks

We make available reproducible test results (validated test errors, learning curves) of MultiBoost on the web site as we produce them. Among other results, the boosted decision product is one of the best reported no-domain-knowledge algorithms on MNIST. An early version of the program (Bergstra et al., 2006) was the best genre classifier out of 15 submissions and the second-best out of 10 submissions at recognizing artists in the MIREX 2005 international contest on music information extraction. More recently, we participated in the Yahoo! Learning to Rank Challenge using a pointwise ranking approach based on hundreds of MultiBoost classifiers. We finished 6th in Track 1 and 11th in Track 2 out of several hundred participating teams (Busa-Fekete et al., 2011).

Acknowledgments

This work was supported by the ANR-2010-COSI-002 grant of the French National Research Agency.

References

J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kégl. Aggregate features and AdaBoost for music classification. *Machine Learning Journal*, 65(2/3):473–484, 2006.

L. Bourdev and J. Brandt. Robust object detection via soft cascade. In *Conference on Computer Vision and Pattern Recognition*, volume 2, pages 236–243. IEEE Computer Society, 2005.

J.K. Bradley and R.E. Schapire. FilterBoost: Regression and classification on large datasets. In *Advances in Neural Information Processing Systems*, volume 20. The MIT Press, 2008.

R. Busa-Fekete and B. Kégl. Fast boosting using adversarial bandits. In *International Conference on Machine Learning*, volume 27, pages 143–150, 2010.

R. Busa-Fekete, B. Kégl, Éltető T., and Gy. Szarvas. Ranking by calibrated AdaBoost. In *(JMLR W&CP)*, volume 14, pages 37–48, 2011.

G. Escudero, L. Màrquez, and G. Rigau. Boosting applied to word sense disambiguation. In *Proceedings of the 11th European Conference on Machine Learning*, pages 129–141, 2000.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55:119–139, 1997.

B. Kégl and R. Busa-Fekete. Boosting products of base classifiers. In *International Conference on Machine Learning*, volume 26, pages 497–504, Montreal, Canada, 2009.

R.E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

P. Viola and M. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57:137–154, 2004.