
Metropolis-Hastings Sampling in a FilterBoost Music Classifier

Balázs Kégl

Laboratoire de l'Accélérateur Linéaire, Université Paris-Sud 11, FRANCE

BALAZS.KEGL@GMAIL.COM

Thierry Bertin-Mahieux

DIRO, University of Montreal, Montreal, QC, CANADA

BERTINMT@IRO.UMONTREAL.CA

Douglas Eck

DIRO, University of Montreal, Montreal, QC, CANADA

DOUGLAS.ECK@UMONTREAL.CA

Abstract

Rejection sampling is a useful technique for performing supervised learning on training sets too large to be learned in their entirety. FilterBoost is a recent extension to AdaBoost which uses rejection sampling in an online learning framework and has been shown to work for automatic tagging of music. In this paper we improve on FilterBoost by adding Metropolis-Hastings sampling, thus allowing the algorithm to focus on hard-to-classify examples. We describe how our knowledge of artist-level similarity can be used effectively in a Metropolis-Hastings framework and demonstrate a significant increase in classification accuracy over standard FilterBoost.

1. Introduction and Previous Work

Rejection sampling is a useful technique for performing supervised learning on training sets too large to be learned in their entirety. In particular, FilterBoost is a recent extension to AdaBoost based on that concept: an oracle gathers data that is later accepted by a filter by a probability related to classification error, thus forcing the model to focus more attention to hard-to-classify data points. This method has been shown to work for automatic tagging of music (Bertin-Mahieux et al., 2008). However, this approach does not take advantage of any structure in our data. For example, if we have a similarity measure for data points we can improve performance by increasing our probability to

sample near-neighbors to a difficult point. We possess several similarity measures for working with recorded music. Perhaps the simplest is to treat songs coming from the same album or same artist as being similar. This is a strong assumption, but more refined measures of similarity can also be used.

1.1. FilterBoost and Metropolis-Hasting Sampling

Much attention has been paid to the automatic description of audio using a fixed vocabulary of words (Eck et al., 2008; Turnbull et al., 2008). In previous work we demonstrated that the AdaBoost large-margin ensemble learner performs well on this task (Bergstra et al., 2006). The challenge therefore is the scaling problem: our current music collection contains more than 120K songs, and a commercial system has millions of them. How can we apply such a classifier to what is, for practical purposes, an infinitely large data set?

A recent extension of AdaBoost was presented at NIPS 2007 (Bradley & Schapire, 2008). FilterBoost samples $\langle \text{train}, \text{target} \rangle$ pairs from a data store (or “oracle”) assumed to be infinite, but only learns from a candidate pair if it cannot already classify it. Otherwise it rejects that point and samples another one. Bradley et. al showed tremendous speed increases and memory savings with no loss in performance. By default, FilterBoost considers the training data to be i.i.d. However this is not the case for recorded music where there is strong inter-album and inter-artist similarity. One standard way to take advantage of such structure is via the Metropolis-Hastings algorithm (Bishop, 2006) which is used to sample from a complex probability distribution, and is a special case of the wide framework of Markov Chain Monte Carlo. From a point $z^{(\tau)}$

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

we draw a sample z^* with probability

$$A(z^*, z^{(\tau)}) = \min \left(1, \frac{\tilde{p}(z^*)q_k(z^{(\tau)}|z^*)}{\tilde{p}(z^{(\tau)})q_k(z^*|z^{(\tau)})} \right) \quad (1)$$

If the sample z^* is refused, we keep $z^{(\tau)}$.

In our standard FilterBoost model, the oracle samples equally from positive and negative examples and rejects based on how easily the current strong learner classifies. Then the filter looks how easily the current strong learner classifies it. The filter accepts the example at iteration t with probability

$$q_t(x, l) = 1/(1 + e^{lg_t(x)}) \quad (2)$$

where $lg_t(x)$ is positive if x well-classified, negative otherwise. We train 3000 weak learners using FilterBoost, weak learners being single stumps. Each iteration of FilterBoost, we gather a mini batch of size 3000, and we select 50 single stump using classical AdaBoost. Then we gather another 3000 examples to evaluate the weight of those 50 new weak learners.

We incorporate Metropolis-Hastings into FilterBoost as a direct replacement for the simple rejection filter. The oracle samples from the same artist with probability q_t from equation 2, and randomly from the database with probability $1 - q_t$. The probability of an example is also computed using equation 2.

2. Experiment and Results

Our data come from social tags obtained from the Last.fm website using their AudioScrobbler service. Matched audio is taken from a lab collection of more than 120K audio files spanning more than 5600 artists. Audio features include MFCCs, autocorrelation coeffs. and Constant-Q coeffs. summarized (mean and standard deviation) every five seconds. For each tag, artists are ordered by the (normalized) tag frequency. For each tag, the songs from the top 10 artists are considered positive examples, following artists (up to a thousand) are ignored because we are unsure about them. Nonetheless, we can reasonably expect these examples to be usually positive. The rest of the artists in the database are used as negative examples. See (Eck et al., 2008) for details.

Figure 1 gives results for the tag *Grunge*. We see a significant improvement for the new approach on negative examples, and only a small decrease in accuracy for the positive and ignore examples. We conclude that the Metropolis-Hastings algorithm has helped FilterBoost focus on hard (generally negative) examples, thus improving performance significantly.

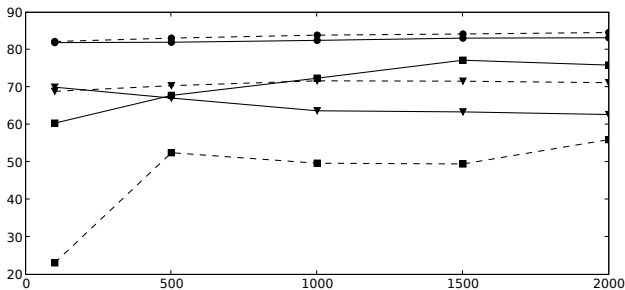


Figure 1. Accuracy by the number of weak learners for tag *Grunge*. The thick line uses Metropolis-Hastings, the dotted line our older method. The held out positive examples are circles, the ignore examples are triangles, and the held out negative examples are squares.

3. Conclusion and Ongoing Work

Previously (Bertin-Mahieux et al., 2008) we demonstrated that FilterBoost works significantly better than AdaBoost at this task. Here we demonstrate that a Metropolis-Hastings rejection policy outperforms a simple error-driven rejection policy. Our current measure of similarity is crude: songs by the same artist are similar. In future work (probably before the workshop) we will incorporate more sophisticated similarity measures which take into account inter-artist similarity. Finally we note that these results are preliminary; more complete results will be available at the time of the workshop.

References

- Bergstra, J., Casagrande, N., Erhan, D., Eck, D., & Kégl, B. (2006). Aggregate features and AdaBoost for music classification. *Machine Learning*, 65, 473–484.
- Bertin-Mahieux, T., Eck, D., Maillet, F., & Lamere, P. (2008). Autotagger: a model for predicting social tags from acoustic features on large music databases. *Journal of New Music Research*. (to appear).
- Bishop, C. (2006). *Pattern recognition and machine learning*. Springer Verlag.
- Bradley, J. K., & Schapire, R. (2008). Filterboost: Regression and classification on large datasets. In *Nips 2007*.
- Eck, D., Lamere, P., Bertin-Mahieux, T., & Green, S. (2008). Automatic generation of social tags for music recommendation. In *Nips 2007*.
- Turnbull, D., Barrington, L., Torres, D., & Lanckriet, G. (2008). Semantic annotation and retrieval of music and sound effects. *IEEE Trans. on Audio, Speech & Lang. Proc.*