Empirical Comparisons of Online Boosting Algorithms with Running Time

Xiaowei Sun[1, a*]

[1]Software College,Shenyang Normal University, Shenyang, 110034, China

[a]email: junyaomail@163.com

Keywords: boosting, ensemble learning,online learning, running time

Abstract. Boosting is an effective classifier combination method, which can improve classification performance of an unstable learning algorithm due to its theoretical performance guarantees and strong experimental results. However, the algorithm has been used mainly in batch mode, i.e., it requires the entire training set to be available at once and, in some cases, require random access to the data. Recently, Nikunj C.oza(2001) proved that some preliminary theoretical results and some empirical comparisons of the classification accuracies of online algorithms with their corresponding batch algorithms on many datasets. In this paper, we present online versions of some boosting methods that require only one pass through the training data. Specifically, we present theoretical and experimental evidence that our online algorithms succeed in this mirroring, often obtaining classification performance comparable to their batch counterparts in less time. We compare the online and batch algorithms experimentally in terms of running time.

Introduction

Traditional supervised learning algorithms generate a single model such as a Naïve Bayes classifier or TAN[1] classifier or BAN[2] classifier and use it to classify examples. Ensemble learning algorithms combine the predictions of multiple base models, each of which is learned using a traditional algorithm. Boosting [3] is a well-known ensemble learning algorithm that has been shown to improve generalization performance compared to the individual base models. Theoretical analysis of Boosting's performance supports these results [4-7].

Nikunj C.oza(2001)[8] developed online versions of bagging and boosting. Online learning algorithms process each training example once "on arrival" without the need for storage and reprocessing, and maintain a current model that reflects all the training examples seen so far. Such algorithms run faster than typical batch algorithms in situations where data arrive continuously. They are also faster with large training sets for which the multiple passes through the training set required by most batch algorithms are prohibitively expensive.

In this paper, we expand upon these results by comparing their running times. We test our online boosting algorithms with three different base models: Naïve Bayes classifiers, TAN classifiers and BAN classifiers. We chose Naïve Bayes classifiers because a lossless online learning algorithm is available for them. For a given training set, a loss-less online learning algorithm returns a model identical to that returned by the corresponding batch algorithm.

Online Boosting Algorithm

Our online boosting algorithm is designed to be an online version of AdaBoost.M1[9]. AdaBoost generates a sequence of base Models h1,h2,⋯⋯,hM using weighted training sets (weighted by D1,D2,⋯⋯,DM) such that the training examples misclassified by model hm-1 are given half the total weight when generating model hm and the correctly classified examples are given the remaining half of the weight.

```
  Online Boosting Algorithm:
Initial conditions : For all m ∈ {1,2,⋯⋯,M}, $\lambda$ msc=0， $\lambda$ msw=0.
Online Boosting(h, L0, (x, y))
        Set  the example's "weighted" $\lambda$ =1.
```

```
For each base model hm,   (m $\in$ {1,2,……,M}) in h,
            Set k according to Poisson( $\lambda$ ).
            Do k times
                      hm = L0(hm, (x, y)).
            If y = hm (x)
            then
                      $\lambda$ msc$\leftarrow$ $\lambda$ msc+ $\lambda$
                      $\varepsilon$ m$\leftarrow$ $\lambda$ msw/( $\lambda$ msc+ $\lambda$ msw)
                         $\lambda$  $\leftarrow$ $\lambda$ (1/(2(1- $\varepsilon$ m)))
            else
                      $\lambda$ msw$\leftarrow$ $\lambda$ msw+ $\lambda$
                      $\varepsilon$ m$\leftarrow$ $\lambda$ msw/( $\lambda$ msc+ $\lambda$ msw)
                         $\lambda$  $\leftarrow$ $\lambda$ (1/(2 $\varepsilon$ m))
To classify new examples:
            Return h(x)=arg maxc$\in$Y$\sum$m:hm(x)=clog((1-$\varepsilon$m)/$\varepsilon$m).
```

Comparsion and results

In this section, we discuss results on several datasets, whose names and numbers of training examples, test examples, classes, attributes and missing values are given in Table 1. The Census Income dataset comes with fixed training and test sets, which we use in our experiments. For the remaining datasets, we used 5-fold cross-validation. We tested with some small datasets to show that the online algorithms can often achieve performance comparable to batch algorithms even when given a small number of data points. Of course, our results with larger datasets are more important. All but three of the datasets are from the UCI KDD repository [10]. The remaining three are synthetic datasets that were chosen because the performance of a single Naïve Bayes classifier varies significantly across these three datasets. These datasets allow us to compare the performances of the online and batch ensemble algorithms on datasets of varying difficulty.

Table 1. Datasets used in the experiments

No.	Dataset	Training Set	Test Set	Classes	Attributes	Missing values
1	Promoters	84	22	2	57	×
2	Breast-cancer-w	559	140	2	10	√
3	German	800	200	2	20	×
4	Car Evaluation	1382	346	4	6	×
5	Mushroom	6499	1625	2	22	×
6	Synthetic-1	80000	20000	2	20	×
7	Synthetic-2	80000	20000	2	20	×
8	Synthetic-3	80000	20000	2	20	×
9	Census Income	199523	99762	2	40	√
10	Forest Covertype	464809	116203	7	54	√

We report and analyze the running times of the batch and online algorithms that we experimented with. There are several factors that affect the difference between the running times of an online learning algorithm and its batch counterpart. Online learning algorithms' main advantage over batch learning algorithms is the ability to incrementally update their models with new training examples---batch algorithms often have to throw away the previously learned model and learn a new model after adding the new examples to the training set. This is clearly very wasteful computationally and is impossible when there are more data than can be stored. Additionally, batch bagging must cycle through the dataset at least MT times, where M is the number of base models and T is the number of times the base model learning algorithm must cycle through the training set to construct one model. Therefore, each training example is examined MT times. On the other hand, Online algorithms do not require storing the entire training set. However, for a fixed training set (i.e., one to which new training examples are not continually added), batch algorithms sometimes run faster than the corresponding online algorithms. This is because batch algorithms can often set their model parameters once and for all by examining the entire training set at once while online algorithms have to update their parameters once per training example.

Table 2 shows the running times for Naïve Bayes as well as all the ensemble learning algorithms using Naïve Bayes classifiers as base models. Online boosting has a clear running time advantage over batch boosting. Online boosting's fewer sweeps through the dataset clearly outweigh any reduction in the number of base models returned by batch boosting, especially for larger datasets. Tables 3 and Table 4 give the running times for TAN/BAN and the batch and online ensemble algorithms. This time, online boosting are faster than their batch counterparts especially on small datasets. The batch algorithms are slowed down because each TAN/BAN requires ten cycles through the dataset.

Table 2. Running times (sec.) for Native Bayes and Ensembles

No.	Dataset	Native Bayes	Boosting	Online Boosting
1	Promoters	0.02	0.44	0.72
2	Breast-cancer-w	0.02	1.32	0.66
3	German	0	0.7	1.5
4	Car Evaluation	0.04	0.88	1.72
5	Mushroom	0.38	114	11.08
6	Synthetic-1	7.48	1352	394
7	Synthetic-2	5.94	5333	343
8	Synthetic-3	4.58	3762	284
9	Census Income	56.6	25605	1200
10	Forest Covertype	106	67611	15638

Table 3. Running times (sec.) for TANs and Boosting

No.	Dataset	TAN	Boosting	Online Boosting
1	Promoters	0.22	0.28	1.89
2	Breast-cancer-w	0.19	0.22	4.3
3	German	0.37	1.44	12.18
4	Car Evaluation	0.11	0.56	13.66
5	Mushroom	0.7	28.36	225.52
6	Synthetic-1	3.9	1528.3	552.6
7	Synthetic-2	3.92	6075.2	490.8
8	Synthetic-3	4.26	8000.9	320.69
9	Census Income	51.4	25611.7	4785.4
10	Forest Covertype	176.64	27568.9	42107.04

Table 4. Running times (sec.) for BANs and Boosting

No.	Dataset	BAN	Boosting	Online Boosting
1	Promoters	0.16	3.56	3.5
2	Breast-cancer-w	0.18	6.12	7.24
3	German	0.22	17.86	312.52
4	Car Evaluation	0.3	14.9	53.56
5	Mushroom	1.12	162.42	30.12
6	Synthetic-1	3.8	1634.68	552.68
7	Synthetic-2	3.82	4759.2	475.6
8	Synthetic-3	4.06	7778.9	350.6
9	Census Income	62.9	23768.4	3785.1
10	Forest Covertype	152.4	25366	34321.6

Conclusions

In this paper, we discussed online versions of boosting and gave both theoretical and experimental evidence that they can perform comparably to their batch counterparts in terms of accuracy while running much faster. In this paper, we experimented only with batch datasets, i.e., one is not concerned with concept drift. Online algorithms are useful for batch datasets that cannot be loaded into memory in their entirety. This paper provides a stepping stone to using ensemble learning algorithms on large datasets. We hope we can come up with new ideas to make ensemble learning algorithms more practical for modern data mining problems in the future.

References

[1] Cheng Jie, Greiner Russell, “Comparing Bayesian network classifiers”, In: Kathryn Blackmond Laskey, Henri Prade eds. Proc of the 15th Conf on Uncertainty in Artificial Intelligence. San Francisco: Morgan Kaufmann, pp.101-108, 1999.

[2] Friedman Nir, Geiger Dan, Goldszmidt Moises, “Bayesian network classifiers”, Machine Learning, 29 (2/3), pp. 131-163, 1999.

[3] Bauer Eric, Kohavi Ron, “An empirical comparison of voting classification algorithms: Bagging, boosting, and variants”, Machine Learning, 36 (1/2), pp. 105-139, 1999.

[4] Dietterich, Thomas, “Ensemble methods in machine learning”, In Kittler, J., Roli, F., eds.: Multiple Classifier Systems. Lecture Notes Computer Sciences, Vol. 1857, pp. 1-15,2001.

[5] Freund Yoav, Robert Schapire, “A decision-theoretic generalization of on-line learning and an application to boosting.Unpublished manuscript available electronically (on our web pages, or by email request)”, An extended abstract appeared in Computational Learning Theory: Second European Conf, EuroCOLT, pp.23-37,1995.

[6] Xiaowei Sun, Hongbo Zhou, “An Empirical Comparison of Two Boosting Algorithms on Real Data Sets based on Analysis of Scientific Materials”, Springer, Advances in Intelligent and Soft Computing, vol.105, pp.324-327, 2011.

[7] Hongbo Shi, Houkuan Huang, Zhihai Wang, “Boosting-Based TAN Combination Classifier”, Journal of Computer research and development, 41(2), pp. 340-345,2004.

[8] Nikunj C. Oza, “Online Ensemble Learning,” Ph.D. thesis, Department of Electrical Engineering and Computer Science, University of California, Berkeley, 2001.

[9] Yoav Freund and Robert Schapire, “A Decision-Theoretic Generalization of On-line Learning and an Application to Boosting,” Journal of Computer System Sciences, Vol. 55, No. 1, pp. 119-139, 1997.

[10] Stephen D. Bay, “The UCI KDD Archive UCI Machine Learning Repository.” http://archive.ics.uci.edu/ml/.