

Lossless Online Bayesian Bagging

Herbert K. H. Lee
ISDS
Duke University
Box 90251
Durham, NC 27708
herbie@isds.duke.edu

Merlise A. Clyde
ISDS
Duke University
Box 90251
Durham, NC 27708
clyde@isds.duke.edu

July 1, 2002

Abstract

Bagging frequently improves the predictive performance of a model. An online version has recently been introduced, which attempts to gain the benefits of an online algorithm while approximating regular bagging. However, regular online bagging is an approximation to its batch counterpart and so is not lossless with respect to the bagging operation. By operating under the Bayesian paradigm, we introduce an online Bayesian version of bagging which is exactly equivalent to the batch Bayesian version, and thus when combined with a lossless learning algorithm gives a completely lossless online bagging algorithm. We also note that the Bayesian formulation resolves a theoretical problem with bagging.

1 Introduction

In a typical prediction problem, there is a trade-off between bias and variance, in that after a certain amount of fitting, any increase in the precision of the fit will cause an increase in the prediction variance on future observations. Similarly, any reduction in the prediction variance causes an increase in the expected bias for future predictions. Breiman (Breiman, 1996) introduced bagging as a method of reducing the prediction variance without affecting the prediction bias. As a result, predictive performance can be significantly improved.

Bagging, short for “Bootstrap AGGREGatING”, is an ensemble learning method. Instead of making predictions from a single model fit on the observed data, bootstrap samples are taken of the data, the model is

fit on each sample, and the predictions are averaged over all of the fitted models to get the bagged prediction. Breiman (Breiman, 1996) explains that bagging works well for unstable modeling procedures, i.e. those for which the conclusions are sensitive to small changes in the data. He also gives a theoretical explanation of how bagging works, demonstrating the reduction in mean-squared prediction error for unstable procedures. Breiman (Breiman, 1994) demonstrated that tree models, among others, are empirically unstable.

Online bagging (Oza and Russell, 2001) was developed to implement bagging sequentially, as the observations appear, rather than in batch once all of the observations have arrived. It uses an asymptotic approximation to mimic the results of regular batch bagging, and as such it is not a lossless algorithm. Online algorithms have many uses in modern computing. By updating sequentially, the update for a new observation is relatively quick compared to re-fitting the entire database, making real-time calculations more feasible. Such algorithms are also advantageous for extremely large datasets where reading through the data just once is time-consuming, so batch algorithms which would require multiple passes through the data would be infeasible.

In this paper, we consider a Bayesian version of bagging (Clyde and Lee, 2001) based on the Bayesian bootstrap (Rubin, 1981). This overcomes a technical difficulty with the usual bootstrap in bagging, and it leads to a reduction in variance over the bootstrap for certain classes of estimators. We present an online version which, when combined with a lossless online model-fitting algorithm, continues to be lossless with respect to the bagging operation, in contrast to ordinary online bagging.

In the next section we review the basics of the bagging algorithm, of online bagging, and of Bayesian bagging. Next we introduce our online Bayesian bagging algorithm. We then demonstrate its efficacy with classification trees on a variety of examples.

2 Bagging

In ordinary (batch) bagging, bootstrap re-sampling is used to reduce the variability of an unstable estimator. A particular model or algorithm, such as a classification tree, is specified for learning from a set of data and producing predictions. For a particular dataset \mathbf{X}_m , denote the vector of predictions (at the observed sites or at new locations) by $\mathbf{G}(\mathbf{X}_m)$. Denote the observed data by $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$. A bootstrap sample of the data is a sample with replacement, so that $\mathbf{X}_m = (\mathbf{x}_{m_1}, \dots, \mathbf{x}_{m_n})$, where $m_i \in \{1, \dots, n\}$ with repetitions allowed. \mathbf{X}_m can also be thought of as a re-weighted version of \mathbf{X} , where the weights, $\omega_i^{(m)}$ are drawn from the set $\{0, \frac{1}{n}, \frac{2}{n}, \dots, 1\}$ and correspond to the number of times that \mathbf{x}_i appears in the bootstrap

sample. We denote the weighted sample as $(\mathbf{X}, \boldsymbol{\omega}^{(m)})$. For each bootstrap sample, the model produces predictions $G(\mathbf{X}_m)_1, \dots, G(\mathbf{X}_m)_P$. M total bootstrap samples are used. The bagged predictor for the j th element is then

$$\frac{1}{M} \sum_{m=1}^M G(\mathbf{X}_m)_j = \frac{1}{M} \sum_{m=1}^M G(\mathbf{X}, \boldsymbol{\omega}^{(m)})_j$$

or in the case of classification, the j th element is predicted to belong to the most frequently predicted category by $G(\mathbf{X}_1)_j, \dots, G(\mathbf{X}_M)_j$.

A version of pseudocode for implementing bagging is:

1. For $m \in \{1, \dots, M\}$,
 - (a) Draw a bootstrap sample, \mathbf{X}_m , from \mathbf{X} .
 - (b) Find predicted values $\mathbf{G}(\mathbf{X}_m)$.
2. The bagging predictor is $\frac{1}{M} \sum_{m=1}^M \mathbf{G}(\mathbf{X}_m)$.

or equivalently:

1. For $m \in \{1, \dots, M\}$,
 - (a) Draw random weights $\boldsymbol{\omega}^{(m)}$ from $\{0, \frac{1}{n}, \frac{2}{n}, \dots, 1\}$ that sum to 1.
 - (b) Find predicted values $\mathbf{G}(\mathbf{X}, \boldsymbol{\omega}^{(m)})$.
2. The bagging predictor is $\frac{1}{M} \sum_{m=1}^M \mathbf{G}(\mathbf{X}_m, \boldsymbol{\omega}^{(m)})$.

For classification, the bagging predictor is a plurality vote.

2.1 Online bagging

Online bagging (Oza and Russell, 2001) was recently introduced as a sequential approximation to batch bagging. In batch bagging, the entire dataset is collected, and then bootstrap samples are taken from the whole database. An online algorithm must process observations as they arrive, and thus each observation must be resampled a random number of times when it arrives. The algorithm proposed by Oza and Russell resamples each observation according to a Poisson random variable with mean 1, i.e., $P(K_m = k) = \exp(-1)/k!$, where K_m is the number of resamples in “bootstrap sample” m , $K_m \in \{0, 1, \dots\}$. Thus as each observation arrives, it is added K_m times to \mathbf{X}_m , and then $\mathbf{G}(\mathbf{X}_m)$ is updated, and this is done for $m \in \{1, \dots, M\}$.

Pseudocode for online bagging is:

For $i \in \{1, \dots, n\}$,

1. For $m \in \{1, \dots, M\}$,
 - (a) Draw a weight K_m from a *Poisson*(1) random variable and add K_m copies of x_i to \mathbf{X}_m .
 - (b) Find predicted values $\mathbf{G}(\mathbf{X}_m)$.
2. The current bagging predictor is $\frac{1}{M} \sum_{m=1}^M \mathbf{G}(\mathbf{X}_m)$.

Ideally, step 1(b) is accomplished with a lossless online update that incorporates the K_m new points without refitting the entire model. We note that n may not be known ahead of time, but the bagging predictor is a valid approximation at each step.

Online bagging is not guaranteed to produce the same results as batch bagging. In particular, it is easy to see that after n points have been observed, there is no guarantee that \mathbf{X}_m will contain exactly n points, as the Poisson weights are not constrained to add up to n like a regular bootstrap sample. While it has been shown (Oza and Russell, 2001) that these samples converge asymptotically to the appropriate bootstrap samples, there may be some discrepancy in practice. Thus while it can be combined with a lossless online learning algorithm (such as for a classification tree), the bagging part of the online ensemble procedure is not lossless.

2.2 Bayesian bagging

Ordinary bagging is based on the ordinary bootstrap, which can be thought of as replacing the original weights of $\frac{1}{n}$ on each point with weights from the set $\{0, \frac{1}{n}, \frac{2}{n}, \dots, 1\}$, with the total of all weights summing to 1. A variation is to replace the ordinary bootstrap with the Bayesian bootstrap (Rubin, 1981). The Bayesian approach treats the vector of weights $\boldsymbol{\omega}$ as unknown parameters and derives a posterior distribution for $\boldsymbol{\omega}$, and hence $\mathbf{G}(\mathbf{X}, \boldsymbol{\omega})$. The non-informative prior $\prod_{i=1}^n \omega_i^{-1}$, when combined with the multinomial likelihood, leads to a *Dirichlet*(1, ..., 1) distribution for the posterior distribution of $\boldsymbol{\omega}$. The full posterior distribution of $\mathbf{G}(\mathbf{X}, \boldsymbol{\omega})$ can be estimated by Monte Carlo methods: generate $\boldsymbol{\omega}^{(m)}$ from a *Dirichlet*(1, ..., 1) distribution and then calculate $\mathbf{G}(\mathbf{X}, \boldsymbol{\omega}^{(m)})$ for each sample. The average of $\mathbf{G}(\mathbf{X}, \boldsymbol{\omega}^{(m)})$ over the M samples corresponds to the Monte Carlo estimate of the posterior mean of $\mathbf{G}(\mathbf{X}, \boldsymbol{\omega})$ and can be viewed as a Bayesian analog of bagging. (Clyde and Lee, 2001)

In practice, we may only be interested in a point estimate, rather than the full posterior distribution. In this case, the Bayesian bootstrap can be seen as a continuous version of the regular bootstrap. Thus Bayesian bagging can be achieved by generating M Bayesian bootstrap samples, and taking the average or majority

vote of the $G(\mathbf{X}, \omega^{(m)})$. This is identical to regular bagging except that the weights are continuous-valued on $(0, 1)$, instead of being restricted to the discrete set $\{0, \frac{1}{n}, \frac{2}{n}, \dots, 1\}$. In both cases, the weights must sum to 1. In both cases, the expected value of a particular weight is $\frac{1}{n}$ for all weights, and the expected correlation between weights is the same (Rubin, 1981). Thus the Bayesian bagging will generally have the same expected point estimates as ordinary bagging. The variability of the estimate is slightly smaller under Bayesian bagging, as the variability of the weights is $\frac{n}{n+1}$ times that of ordinary bagging. As the sample size grows large, this factor becomes arbitrarily close to one, but we do note that it is strictly less than one, so the Bayesian approach does give a further reduction in variance compared to the standard approach.

Pseudocode for Bayesian bagging is:

1. For $m \in \{1, \dots, M\}$,
 - (a) Draw random weights $\omega^{(m)}$ from a *Dirichlet*($1, \dots, 1$) to produce the Bayesian bootstrap sample $(\mathbf{X}, \omega^{(m)})$.
 - (b) Find predicted values $G(\mathbf{X}, \omega^{(m)})$.
2. The bagging predictor is $\frac{1}{M} \sum_{m=1}^M G(\mathbf{X}, \omega^{(m)})$.

Use of the Bayesian bootstrap does have a major theoretical advantage, in that for some problems, bagging with the regular bootstrap is actually estimating an undefined quantity. To take a simple example, suppose one is bagging the fitted predictions for a point y from a least-squares regression problem. Technically, the full bagging estimate is $\frac{1}{M_0} \sum_m \hat{y}_m$ where m ranges over all possible bootstrap samples, M_0 is the total number of possible bootstrap samples, and \hat{y}_m is the predicted value from the model fit using the m th bootstrap sample. The issue is that one of the possible bootstrap samples contains the first data point replicated n times, and no other data points. For this bootstrap sample, the regression model is undefined (since at least two different points are required), and so \hat{y} and thus the bagging estimator are undefined. In practice, only a small sample of the possible bootstrap samples is used, so the probability of drawing a bootstrap sample with an undefined prediction is very small. Yet it is disturbing that in some problems, the bagging estimator is technically not well-defined. In contrast, the use of the Bayesian bootstrap completely avoids this problem. Since the weights are continuous-valued, the probability that any weight is exactly equal to zero is zero. Thus with probability one, all weights are strictly positive, and the Bayesian bagging estimator will be well-defined (assuming the ordinary estimator on the original data is well-defined).

3 Online Bayesian bagging

Regular online bagging cannot be exactly equivalent to the batch version because the Poisson counts cannot be guaranteed to sum to the number of actual observations. Gamma random variables can be thought of as continuous analogs of Poisson counts, which motivates our derivation of Bayesian online bagging. The key is to recall a fact from basic probability — a set of independent gamma random variables divided by its sum has a Dirichlet distribution, i.e.,

$$\text{If } w_i \sim \Gamma(\alpha_i, 1), \text{ then } \left(\frac{w_1}{\sum w_i}, \frac{w_2}{\sum w_i}, \dots, \frac{w_k}{\sum w_i} \right) \sim \text{Dirichlet}(\alpha_1, \alpha_2, \dots, \alpha_k).$$

(See for example (Hogg and Craig, 1995) pp. 187–188.) This relationship is a common method for generating random draws from a Dirichlet distribution, and so is also used in the implementation of batch Bayesian bagging in practice.

Thus in the online version of Bayesian bagging, as each observation arrives, it has a realization of a *Gamma*(1) random variable associated with it for each bootstrap sample. Within each (Bayesian) bootstrap sample, all weights are then re-normalized with the new sum of gammas and the model is updated. At any point in time, the current predictions are those aggregated across all bootstrap samples, just as with batch bagging. If the model is fit with an ordinary lossless online algorithm, as exists for trees (Utgoff et al., 1997), then the entire online Bayesian bagging procedure is completely lossless relative to batch Bayesian bagging. Furthermore, since batch Bayesian bagging gives the same mean results as ordinary batch bagging, online Bayesian bagging also has the same expected results as ordinary batch bagging.

Pseudocode for online Bayesian bagging is:

For $i \in \{1, \dots, n\}$,

1. For $m \in \{1, \dots, M\}$,

(a) Draw a weight $\omega_i^{(m)}$ from a *Gamma*(1, 1) random variable, associate with weight with x_i , and add x_i to \mathbf{X} .

(b) Find predicted values $\mathbf{G}(\mathbf{X}, \boldsymbol{\omega}^{(m)})$ (renormalizing weights if necessary).

2. The current bagging predictor is $\frac{1}{M} \sum_{m=1}^M \mathbf{G}(\mathbf{X}, \boldsymbol{\omega}^{(m)})$.

In step 1(b), the weights may need to be renormalized (by dividing by the sum of all current weights) if the implementation requires weights that sum to one. We note that for many models, such as classification trees, this renormalization is not a major issue; for a tree, each split only depends on the relative weights of

the observations at that node, so nodes not involving the new observation will have the same ratio of weights before and after renormalization and the rest of the tree structure will be unaffected; in practice, in most implementations of trees, renormalization is not necessary.

4 Examples

We demonstrate the effectiveness of online Bayesian bagging using classification trees. Our implementation uses the lossless online tree learning algorithms (ITI) of Utgoff et al. (Utgoff et al., 1997) (available at <http://www.cs.umass.edu/~lrn/iti/>). We compared Bayesian bagging to a single tree, ordinary batch bagging, and ordinary online bagging, all three of which were done using the minimum description length criterion (MDL), as implemented in the ITI code, to determine the optimal size for each tree. To implement Bayesian bagging, the code was modified to account for weighted observations. We used a modified MDL to determine the optimal tree size at each stage, where our modified version replaces all counts of observations with the sum of the weights of the observations at that node or leaf with the same response category. Replacing the total count directly with the sum of the weights is justified by looking at the multinomial likelihood when written as an exponential family in canonical form; the weights enter through the dispersion parameter and it is easily seen that the unweighted counts are replaced by the sums of the weights of the observations that go into each count.

Table 1 displays the results of our comparison study. All of the datasets, except the final one, are available online from the UCI Machine Learning Repository (<http://www.ics.uci.edu/mlearn/MLRepository.html>). The last dataset is described in (Lee, 2001). We compare the results of training a single tree, ordinary batch bagging, online bagging, and Bayesian online bagging (or equivalently Bayesian batch). For each of the bagging techniques, 100 bootstrap samples were used, and the whole procedure was done 10 times and the error rates were averaged, to try to reduce the sampling variability due to the randomness of the bootstrap samples.

We note that in all cases, both online bagging techniques produce results similar to ordinary batch bagging. However, online Bayesian bagging generally matches more closely than the ordinary online version, as is to be expected, since the Bayesian version is lossless and differences are due only to randomness in choosing/weighting the bootstrap samples.

Dataset	Size of	Size of	Bayesian			
	Training	Test	Single	Batch	Online	Online/Batch
	Data Set	Data Set	Tree	Bagging	Bagging	Bagging
Breast cancer (WI)	299	400	0.040	0.035	0.034	0.035
German credit	200	800	0.350	0.277	0.279	0.253
House votes	185	250	0.048	0.045	0.043	0.044
Ionosphere	200	151	0.060	0.066	0.079	0.058
Pima diabetes	200	332	0.277	0.229	0.231	0.228
Spam	2000	2601	0.099	0.072	0.074	0.074
American credit	4000	4508	0.351	0.300	0.301	0.299

Table 1: Comparison of average error rates

5 Discussion

Bagging is a useful ensemble learning tool, particularly when models sensitive to small changes in the data are used. It is sometimes desirable to be able to use the data in an online fashion. By operating in the Bayesian paradigm, we can introduce an online algorithm that will exactly match its batch Bayesian counterpart. Unlike previous versions of online bagging, the Bayesian approach produces a completely lossless bagging algorithm.

Acknowledgments

This research was partially supported by NSF grants DMS 9873275 and DMS 9733013.

References

- Breiman, L. (1994). “Heuristics of Instability in Model Selection.” Tech. rep., University of California at Berkeley.
- (1996). “Bagging Predictors.” *Machine Learning*, 26, 2, 123–140.
- Clyde, M. A. and Lee, H. K. H. (2001). “Bagging and the Bayesian Bootstrap.” In *Artificial Intelligence and Statistics 2001*, eds. T. Richardson and T. Jaakkola, 169–174.

- Hogg, R. V. and Craig, A. T. (1995). *Introduction to Mathematical Statistics*. 5th ed. Upper Saddle River, NJ: Prentice-Hall.
- Lee, H. K. H. (2001). “Model Selection for Neural Network Classification.” *Journal of Classification*, 18, 227–243.
- Oza, N. C. and Russell, S. (2001). “Online Bagging and Boosting.” In *Artificial Intelligence and Statistics 2001*, eds. T. Richardson and T. Jaakkola, 105–112.
- Rubin, D. B. (1981). “The Bayesian Bootstrap.” *Annals of Statistics*, 9, 130–134.
- Utgoff, P. E., Berkman, N. C., and Clouse, J. A. (1997). “Decision Tree Induction Based on Efficient Tree Restructuring.” *Machine Learning*, 29, 5–44.