Bagging by Design (on the Suboptimality of Bagging)

Periklis A. Papakonstantinou and **Jia Xu** and **Zhu Cao**
IIIS, Tsinghua University

Abstract

Bagging (Breiman 1996) and its variants is one of the most popular methods in aggregating classifiers and regressors. Originally, its analysis assumed that the bootstraps are built from an unlimited, independent source of samples, therefore we call this form of bagging *ideal-bagging*. However in the real world, base predictors are trained on data subsampled from a limited number of training samples and thus they behave very differently. We analyze the effect of intersections between bootstraps, obtained by subsampling, to train different base predictors. Most importantly, we provide an alternative subsampling method called *design-bagging* based on a new construction of combinatorial designs, and prove it universally better than *bagging*. Methodologically, we succeed at this level of generality because we compare the prediction accuracy of bagging and design-bagging relative to the accuracy ideal-bagging. This finds potential applications in more involved bagging-based methods. Our analytical results are backed up by experiments on classification and regression settings.

Introduction

Bootstrapping (Efron 1979) is arguably one of the most significant developments in statistics. Bagging, the "machine learning" analog of bootstrapping, is one of the most influential methods in creating ensembles for classification and regression problems. A large number of variants and extensions of bagging have been devised since it was originally introduced in 1996 by Breiman (Breiman 1996), including the widely studied random forests (Breiman 2001). Despite this spade of works our mathematical understanding on how accurately bagging and its extensions predict is still limited. In this paper we develop new machinery towards this direction, and introduce a provably better alternative to bagging.

In theory, ideal-bagging works as follows. A learner φ given a training set L produces a predictor (for classification or regression), denoted by $\varphi(\cdot; L)$. The learner is provided with independently obtained training sets $L_1, L_2, \ldots$, constructs the classifiers, and given an instance x it decides by voting (or averaging for regression) with the outputs of $\varphi(x; L_1), \varphi(x; L_2), \ldots$. This ideal procedure was originally shown (Breiman 1996) to reduce (in fact, not to increase) the mean-square error compared to a single classifier. In practice, however, the training samples are limited and the training sets (aka bootstraps) $L_1, L_2, \ldots$ are obtained by subsampling from an originally obtained pool of training data $\hat{\mathcal{L}}$. This results in intersections between the L_i's. Many variants of bagging deal heuristically, either directly or indirectly, with the issue of reducing intersections and provide substantial empirical evidence in favor of reducing them, cf. (Ting and Witten 1997). To the best of our knowledge, prior to our work, the effect of intersection on the prediction accuracy was not mathematically understood.

Toy example Here is a very small, made-up example illustrating the effect of intersections – the reader should put this in proportion. Consider a uniform distribution $\mathcal{D}$ over 10 points $\{(0, 0.1), (-0.01, -0.11), (1, 1.1), (0.99, 0.89), (2, 2.1), (1.99, 1.89), (3, 3.1), (2.99, 2.89), (4, 4.1), (4.99, 4.89)\}$, which is hidden from us. Use $\mathcal{D}$ to sample uniformly a subset of 6 points in $\hat{\mathcal{L}}$. Then, sample from $\hat{\mathcal{L}}$ three bootstraps (3 subsets) L_1, L_2, L_3 each consisting of 3 elements. On each of the L_i's do linear regression to obtain $y_i(x) = a_i x + b_i$. The bagged predictor will be $y(x) = ax + b$, where $a = (a_1 + a_2 + a_3)/3$ and $b = (b_1 + b_2 + b_3)/3$. Such a small setting has non-negligible error but the advantage is that we can perform exact calculations. The mean square error $\mathbb{E}_{(X,Y)\sim\mathcal{D},\mathcal{L}_{\mathfrak{B}}}[(Y - y(X))^2] \approx 317$, where $\mathcal{L}_{\mathfrak{B}}$ is the distribution of bagging over the choices of $\hat{\mathcal{L}}$ and $\{L_1, L_2, L_3\}$, and the *average pairwise intersection size is 1.5*. Let us now consider an alternative way to sample the L_i's. Instead of sampling at random we consider a so-called "design", which in this case is a subset system of 3 sets each of which has 3 elements and every 2 of these sets *intersect at exactly 1 point* (it is impossible not to have an intersection since $|\mathcal{L}| = 6$). Note that this subset system is a pattern that we apply on $\hat{\mathcal{L}}$, but $\hat{\mathcal{L}}$ is still filled in by i.i.d. sampling from $\mathcal{D}$. With this being the only difference we have $\mathbb{E}_{(X,Y)\sim\mathcal{D},\mathcal{L}_{\mathfrak{D}}}[(Y - y(X))^2] \approx 170$ which is about 46% smaller mean square error.

Thus, intersections do make a difference in the prediction accuracy of the bagged predictor. We aim to minimize intersection size among every pair of bootstraps. Set systems with such a property are called combinatorial designs, a vibrant area of combinatorics and statistics with diverse ap-

plications, cf. (Colbourn and Dinitz 2010). Known designs have much smaller size and much bigger number of blocks. Thus, they are not suitable for our purposes – our design is reminiscent to weak designs (Hartman and Raz 2003) for which we circumvent the lower bound using much less sets and relaxed intersection size.

We ask how much and what kind of theory can be developed for bagging and its extensions. Despite the wealth of successful empirical studies and heuristics, analytically much less has been done.

First, studies of more statistical nature (Bühlmann and Yu 2002; Friedman and Hall 2007) addressed the issue of prediction accuracy using assumptions about the predictors, and analyzing things at places either heuristically or asymptotically. For example, in (Bühlmann and Yu 2002) there are three explicit assumptions and some implicit assumptions, in places the analysis turns to the analysis of continuous analogs of Brownian motion, and at the end experiments are used to indicate rates of convergence. For the application to machine learning it is desirable to have only minimal statistical assumptions, and moreover we care about actual rates of convergence and not that much for asymptotics.

In more practical settings for bagging and more importantly for its extensions, such as random forests, it is surprising that the current state-of-the-art deals with arguments about the consistency of the aggregated predictors as outlined in (Biau, Devroye, and Lugosi 2008; Biau, Cérou, and Guyader 2010; Denil, Matheson, and de Freitas 2013). Thus, in addition to previous important works there is still much room in understanding bagging and its relatives.

In the realm of theory of machine learning boosting and in particular Adaboost (Freund and Schapire 1997) is shown to boost the accuracy of a weak classifier with PAC guarantees (Valiant 1984) to a strong classifier which also has PAC guarantees. Can bagging boost the performance of a weak learner in the PAC sense? Following the work of (Long and Servedio 2013) we show, for certain parameters, that nothing similar to Adaboost can be proved for bagging due to its inherent parallelism (this can be obtained easily as a corollary of Theorem 18 of their paper). In other words, *in the PAC sense* bagging, together with every parallel ensemble method, cannot achieve boosting. On the other hand it is not clear what PAC guarantees mean for the real-world bagging – e.g. in (Bauer and Kohavi 1999; 1999) there are certain settings where bagging appears to be more accurate than boosting. All these suggest that bagging in the real world should be theoretically studied further.

In this paper we take a novel, but modest first step in analyzing bagging-like algorithms by comparing them to its ideal analog (ideal-bagging) under very minimal assumptions. We feel that this is an appropriate level for developing a practically relevant theory. We analyze the effect of intersections assuming two properties that the base classifiers enjoy, and we show our design-based-bagging better than bagging independently each case. For us "better" means closer to the behavior of ideal-bagging. This is probably all we can hope at this level of generality. Our analysis is for regression settings and binary classification, but this is only for simplicity of exposition (it extends to multi-label classification).

Here is an informal statement of our first theorem.

Theorem 1 (informal statement). *If bagging is more stable to noise than a single predictor then design-bagging outperforms bagging.*

This is a perhaps surprising statement which assuming a good property for bagging we conclude that design-bagging is even better! We make precise what do we mean by tolerance to noise later on. For now we note that ideal-bagging unconditionally enjoys this noise-stability property. Note that the effectiveness of bagging to deal with noise has been verified before as in e.g. (Friedman and Hall 2007; Dietterich 2000; Bauer and Kohavi 1999; Melville et al. 2004). In this sense noise-stability is a very mild assumption for bagging.

Our second main theorem assumes that the base predictors enjoy the following property. If we independently train two predictors using two bootstraps L_1, L_2 then on (most) input instance(s) x the covariance of their outputs $\mathbb{E}_{L_1,L_2}[\varphi(x;L_1)\varphi(x,L_2)]$ increases the more L_1 and L_2 intersect. Again this is a very believable property.

Theorem 2 (informal statement). *If for the base classifiers the covariance* $\mathbb{E}_{L_1,L_2}[\varphi(x;L_1)\varphi(x,L_2) \mid |L_1 \cap L_2| = k]$ *increases as a function of* k*, then design-bagging outperforms bagging.*

By minimizing the covariance of pairs of predictors we increase the "diversity" of the base predictors. The issue of diversity in ensemble learning is a very well studied one cf. (Kuncheva and Whitaker 2003; Dietterich 2000; Tang, Suganthan, and Yao 2006), but it is not clear which notion of diversity is the right one (Tang, Suganthan, and Yao 2006). In our case though, this simple covariance (which accounts for "linear" correlations) is conclusive and mathematically sound. Our experiments show that more intersection on the bootstraps yields more covariance, and by only assuming this we analyze our design-bagging.

Besides these two main theorems we also obtain statements showing that design-bagging makes decisions which are well concentrated (not only in the mean square-loss sense), and we also bound its distance from ideal-bagging. This is regardless of any additional statistical dependencies among the classifiers. This concentration is on the rate of convergence and not in some asymptotic sense.

Notation and terminology

By *design* we refer to a collection of sets $\mathcal{S} = \{L_1, \ldots, L_m\}$, each L_i is also called a *block* and $|L_i| = b$, where most pairs L_i, L_j $i \neq j$ do not intersect much (to be quantified in context). $\mathcal{D}$ commonly denotes distribution over instances $\mathbf{X}$. The pool of samples from which we obtain the bootstraps is denoted by $\hat{\mathcal{L}}$, $|\hat{\mathcal{L}}| = N$. A single learning algorithm $\varphi : \mathbf{X} \times \mathcal{L} \to \mathbf{Y}$ trains base models $\varphi(\cdot; L_1), \ldots, \varphi(\cdot; L_m)$, where $L_i \subseteq \hat{\mathcal{L}}$, $|L_i| = b$, and we write $\mathcal{L}$ for the collection of the L_i's. Technically, our study is for su-bagging, which is shown the same (but cleaner to analyze) as bagging (Bühlmann and Yu 2002) – all our statements carry through for bagging adding an extra conditioning. Three distributions are of interest: (i) $\mathcal{L}_\mathfrak{I}$ for

ideal-bagging where the L_i's are chosen i.i.d from $\mathcal{D}$; (ii) $\mathcal{L}_{\mathfrak{B}}$ for *bagging* where the probability space is induced by first sampling $\mathcal{L}$ i.i.d. and then subsampling from it the L_i's each independently from the set of all subsets of $\mathcal{L}$ of size b; (iii) $\mathcal{L}_{\mathfrak{D}}$ for *design-bagging* we choose a filter over $\mathcal{L}$ (this choice can also be probabilistic) which consists of the relative positions of the $L_1, L_2, \ldots$ inside $\hat{\mathcal{L}}$ and subsequently fill in i.i.d the elements of $\hat{\mathcal{L}}$ which also results the L_i's. We write $\varphi_{\mathfrak{I}}$, $\varphi_{\mathfrak{B}}$, $\varphi_{\mathfrak{D}}$ for the learners which aggregate by voting (or averaging for regression) for ideal-bagging, bagging, and design-bagging respectively. The generalization error of the learner is measured by the mean square error (MSE) – though our main results are stronger, they hold with high probability – where $\mathbb{S}(\varphi_{\mathfrak{I}}) = \mathbb{E}_{\mathcal{L}_{\mathfrak{I}}}(y - \varphi_{\mathfrak{D}}(x))^2$, $\mathbb{S}(\varphi_{\mathfrak{B}}) = \mathbb{E}_{\mathcal{D},\mathcal{L}_{\mathfrak{B}}}(y - \varphi_{\mathfrak{B}}(x))^2$ and $\mathbb{S}(\varphi_{\mathfrak{D}}) = \mathbb{E}_{\mathcal{D},\mathcal{L}_{\mathfrak{D}}}(y - \varphi_{\mathfrak{D}}(x))^2$.

The approximate design and its properties

The bulk of this paper are the theorems of the next section. Part of their lengthy proof is constructing an approximate design and obtaining relevant guarantees. We present our efficient approximate design here first, which can be of independent interest.

Efficient Approximate Design

Algorithm 1 Blocks Generating Algorithm (BGA)

Input: block size b, number of blocks m, universe size N
Initialize m empty blocks.
for $i = 1$ **to** $b \times m$ **do**
  choose L at random from the set of blocks with current min # of elements
  S : set of elements in the universe not in L that appear least frequently
  $L \leftarrow L \cup \{e\}$, where $e \in S$ chosen uniformly at random
end for
Output: m blocks each with b distinct elements.

This algorithm has a great advantage over choosing L's randomly from $\binom{\{1,\ldots,N\}}{b}$, especially when $m \times b \leq cN$ for $c \approx 5, \ldots, 20$. In particular, Algorithm 1 (i) achieves minimum average intersection size (Lemma 1) and (ii) it concentrates the distribution around this expectation. This (Lemma 2) is the main technical part of this paper; in particular, known Chernoff-Hoeffding or other concentration bounds are weaker than what our arguments ask for.

Blocks Generating Algorithm is optimal on the average Every greedy algorithm that iteratively minimizes the intersection size (e.g. BGA), has optimal average intersection size; i.e. the expected arithmetic mean of the $|L_i \cap L_j|$'s for every $i < j$. This simple fact is Lemma 1, and note that the probabilistic decisions made by BGA are not relevant for this lemma.

Lemma 1 (Average intersection size optimality). *Among all constructions of m blocks, each of size b, from a universe of N elements, Algorithm 1 achieves minimum average intersection size with value $\frac{\binom{r}{2} \times N}{\binom{m}{2}}$, where $r = mb/N$.*

Proof. We count the sum of intersection sizes instead of average (same for a fixed number of blocks). We sum over all elements on the number of times each of them is covered by distinct blocks (much simpler to interchange the $\sum$'s this way). Denote the times that element i has been covered be r_i, $1 \leq i \leq N$. Then, since each element is in the intersection if it belongs to a block pair, the total intersection is $\sum_{i=1}^{N} \binom{r_i}{2}$, where $\sum r_i = bm$. It is easy to see that the sum of intersections achieves the minimum when the r_i's are equal, which is indeed achieved by our new algorithm (if N does not divide bm let $\max_i r_i = \min_i r_i + 1$). Therefore, in this algorithm, the total intersection is $\binom{r}{2} \times N$ and thus the average intersection size is $\frac{\binom{r}{2} \times N}{\binom{m}{2}}$. □

Corollary 1 (Design vs Random sampling). *Denote the average intersection size of design-bagging by $\mu_{\mathfrak{B}}$ and that of bagging by $\mu_{\mathfrak{D}}$, then $\mu_{\mathfrak{D}} < \mu_{\mathfrak{B}}$ and furthermore $\mu_{\mathfrak{D}} \leq \frac{bm-N}{bm-b}\mu_{\mathfrak{B}}$.*

Blocks Generating Algorithm outperforms bagging-derived subsets Lemma 2 is the most involved technical part – its proof is in the full version online. We show that the curve of the cumulative distribution (CDF) of the block intersection size k, $F_{\mathfrak{D}}(k; n, m, b)$ resulted-in by BGA strictly dominates $F_{\mathfrak{B}}(k; n, m, b)$, the CDF of the original bagging. $F_{\mathfrak{D}}$ is defined for an arbitrary pair of blocks, which is well-defined since BGA treats blocks symmetrically.

Lemma 2. *Let $F_{\mathfrak{D}}(k; N, m, b)$ and be $F_{\mathfrak{B}}(k; N, m, b)$ the cumulative distribution functions as above. Then for all k where $F_{\mathfrak{D}}(k; N, m, b) < 1$ it holds that $F_{\mathfrak{D}}(k; N, m, b) > F_{\mathfrak{B}}(k; N, m, b)$.*

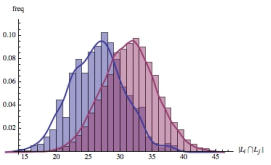

Figure 1: blue curve: PDF of the execution of our randomized algorithm for $N = 500, m = 20, b = 125$, red curve: random sampling.

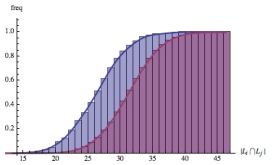

Figure 2: CDF for the same experiment. (mathematically shown in Corollary 1, Lemma 2).

A corollary of the above lemma is that for ideal-bagging, and for distributions over $\mathcal{D}$'s with sufficiently large support we obtain the following.

Corollary 2. *Let $F_{\mathfrak{I}}(k)$, $F_{\mathfrak{D}}(k; N, m, b)$, and $F_{\mathfrak{B}}(k; N, m, b)$ be the cumulative distribution functions of the intersection sizes of ideal-bagging, design-bagging, and bagging. Then, for all k where $F_{\mathfrak{D}}(k; N, m, b) < 1$ holds*

$$F_{\mathfrak{I}}(k; m, b) > F_{\mathfrak{D}}(k; N, m, b) > F_{\mathfrak{B}}(k; N, m, b)$$

Covariance-biased Chebyshev Sampling

Although the outputs of the predictors (classifiers or regressors) are otherwise arbitrarily correlated the following polynomial rate of convergence can be shown. This is used later on in the analysis and comparison of design-bagging.

Definition 1. (Approximate Pairwise Independence) *An ensemble of variables $\{X_i\}_{i\in\mathbb{I}}$ is said to be α-pairwise independent if for each pair $i \neq j$ of variables X_i and X_j,*

$$\mathbb{E}[X_i]\mathbb{E}[X_j] - \alpha \leq \mathbb{E}[X_iX_j] \leq \mathbb{E}[X_i]\mathbb{E}[X_j] + \alpha$$

We call α the covariance parameter *of $\{X_i\}_{i\in\mathbb{I}}$.*

The following simple theorem can be also of independent interest.

Theorem 3 (cov-biased Chebychev Sampling). *For α-pairwise independent variables $X_1, \ldots, X_n$ and $k > 0$*

$$\mathbb{P}[|\sum_i X_i - \mathbb{E}[\sum_i X_i]| \geq k] \leq \frac{\sum_i \mathbf{Var}[X_i]}{k^2} + \alpha\left(\frac{n}{k}\right)^2$$

Proof. Recall that Chebyshev's inequality states for random variable X with finite expectation μ and finite variance σ and any positive real number k, $\mathbb{P}(|X-\mu| \geq k\sigma) \leq \frac{1}{k^2}$.

$$\begin{aligned}
&\mathbb{P}[|\sum_i X_i - \mathbb{E}[\sum_i X_i]| \geq k] \leq \frac{\mathbf{Var}[\sum_i X_i]}{k^2}\\
=&\frac{\mathbb{E}[(\sum_i X_i - \mathbb{E}[\sum_i X_i])^2]}{k^2}\\
=&\frac{\sum_i \mathbf{Var}[X_i] + 2\sum_{i<j}(\mathbb{E}[X_iX_j] - \mathbb{E}[X_i]\mathbb{E}[X_j])}{k^2}\\
\leq&\frac{\sum_i \mathbf{Var}[X_i] + n(n-1)\alpha}{k^2}
\end{aligned}$$

□

Better than bagging

The presentation of our proofs is modular; i.e. the statements of the previous section are used in the proofs here.

The goodness of Design-Bagging

Fix any bagging-like aggregation method (e.g. bagging or design-bagging) and consider the covariance parameter α of any two predictors. Theorem 3 suggests that significantly bounded covariance parameter α for specific classifiers (e.g. k-NN, SVM) yields very strong concentration guarantees in the rates of convergence. If we do not care about so precise performance guarantees, e.g. if we only care about the MSE then things become straightforward. In particular, if α is an upper bound on the covariance parameter in the outputs of any two predictors for some bagging-like method then

$$\text{MSE of this aggregation method} \leq \mathbb{S}(\varphi_{\mathfrak{I}}) + \alpha$$

The above follows immediately by expanding the MSE for this aggregation method. In the full version we compute the parameter α for certain restricted cases of classifiers.

Noise-stability for Bagging $\Longrightarrow$ Design-Bagging is even better

The previous developments enable us to compare design-bagging and bagging. We first deduce this assuming bagging slightly noise stable (see Introduction for references on empirical evidence). We define the noise η generally as the statistical distance induced on the output of the classifiers aggregating with and without noise.

Assumption 1 (noise stability). *Given φ, there is $\rho \in (0,1]$, such that for every x and noise η, if $|\mathbf{Var}[\varphi(x,D)] - \mathbf{Var}[\varphi(x,D+\eta)]| \leq \epsilon$, then $|\mathbf{Var}[\varphi_{\mathfrak{B}}(x,D)] - \mathbf{Var}[\varphi_{\mathfrak{B}}(x,D+\eta)]| \leq \frac{\epsilon}{m^\rho}$.*

We remark that the requirement about all x's can be replaced by "most x's" (in fact, just for Theorem 1 if the assumption holds for the expected x would be sufficient). This assumption is not to be confused with the fact that bagging works better when the base classifier is unstable, which means $(\mathbb{E}_{\mathcal{L}}[\varphi(x,L)])^2$ is far away from $\mathbb{E}_{\mathcal{L}}[\varphi(x,L)^2]$. Finally, we remark that ideal-bagging unconditionally satisfies it and moreover with the strongest $\rho = 1$.

Proposition 1. *For ideal-bagging, if $|\mathbf{Var}[\varphi(x,D)] - \mathbf{Var}[\varphi(x,D+\eta)]| \leq \epsilon$, then $|\mathbf{Var}[\varphi_{\mathfrak{I}}(x,D)] - \mathbf{Var}[\varphi_{\mathfrak{I}}(x,D+\eta)]| \leq \frac{\epsilon}{m}$.*

Now, we are ready to restate Theorem 1 formally.

Theorem 1 (formally stated). Suppose that Assumption 1 holds for a base learner φ. If $\mathbb{S}(\varphi_{\mathfrak{B}}) - \mathbb{S}(\varphi_{\mathfrak{I}}) \geq \delta$, then there is $\lambda < 1$, such that $\mathbb{S}(\varphi_{\mathfrak{D}}) - \mathbb{S}(\varphi_{\mathfrak{I}}) \leq \lambda(\mathbb{S}(\varphi_{\mathfrak{B}}) - \mathbb{S}(\varphi_{\mathfrak{I}}))$. Furthermore, $\lambda(\delta)$ is a decreasing function of δ.

Proof outline: Here is a high-level sketch of the proof of this theorem. The argument relies on the fact that typically there is huge statistical distance between ideal-bagging and bagging, and that the distance from ideal-bagging to design-bagging is smaller. The details are somewhat technical but the high-level idea is simple (see full version for the details). We proceed using a probabilistic hybrid argument. Intuitively, we can visualize a point in a metric space (each point is a distribution) which corresponds to ideal-bagging, and another point which corresponds to bagging. The argument shows that we can move ("slowly reducing distance/interpolating") between these two points and on this specific path we will meet the point that corresponds to design-bagging. Formally, we proceed by introducing the notion of *pointwise noise stable CDF graph*, which is then formally related to Assumption 1. Then, we invoke Lemma 2 to show that if both ideal-bagging and bagging enjoy this property then design-bagging is also point-wise noise stable and in some sense between the two.

More intersection yields more correlation $\Longrightarrow$ Design-Bagging better than Bagging

Consider the situation where we independently train base predictor on two bootstraps L_1, L_2. For these two predictors and any input instance x we denote the correlation of their outputs by $g(k;x) \stackrel{\text{def}}{=} \mathbb{E}_{L_1,L_2}[\varphi(x;L_1)\varphi(x,L_2)||L_1 \cap L_2| = k]$. Then, it is plausible to assume that

Assumption 2 (correlation). *For all input instances x, $g(k;x)$ is an increasing function of k.*

Again, the requirement "for all" can be relaxed to "for most". Our empirical results overwhelmingly show that natural classifiers and regressors satisfy this property. This assumption is necessary since there exist (unrealistic) learners where the covariance oscillates with the intersection size.

Counter-example Here is an example of a constant classifier which works as follows: ignore any label assigned to the input instance (just the value of the instance matters) and consider the parity of the numerical sum of the integer instances. This parity determines the output of the constant classifier. Consider $\{1,2,3,4,5,6\}$, and subsets of size 3 with distinct elements a_i, a_j, a_k. If $a_i + a_j + a_k$ is odd we let the $y_z(x) = 1$ where z is the subset, else $y_z(x) = 0$. Below z_1, z_2 denote two subsets of size 3. Then, $|z_1 \cap z_2| = 0 \implies \mathbb{E}[y_{z_1}(x)y_{z_2}(x)] = 0$, $|z_1 \cap z_2| = 1 \implies \mathbb{E}[y_{z_1}(x)y_{z_2}(x)] = 3/10$, $|z_1 \cap z_2| = 2 \implies \mathbb{E}[y_{z_1}(x)y_{z_2}(x)] = 2/10$, $|z_1 \cap z_2| = 3 \implies \mathbb{E}[y_{z_1}(x)y_{z_2}(x)] = 1/2$, and so on.

Theorem 2 (formally stated). If Assumption 2 holds then

$$\mathbb{S}(\varphi_{\mathfrak{D}}) < \mathbb{S}(\varphi_{\mathfrak{B}})$$

Proof. First, let us expand $\mathbb{S}(\varphi_{\mathfrak{D}})$:

$$\mathbb{S}(\varphi_{\mathfrak{D}}) = \mathbb{E}_{\mathcal{D},\mathcal{L}_{\mathfrak{D}}}\left(y - \frac{1}{m}\sum_i^m \varphi(x,L_i)\right)^2$$
$$= y^2 + \frac{1}{m^2}\underbrace{\mathbb{E}_{\mathcal{D},\mathcal{L}_{\mathfrak{D}}}\left(\sum_i^m \varphi(x,L_i)\right)^2} - \frac{2y\sum_i \mathbb{E}\varphi(x,L_i)}{m}$$

We deliberately did not subscript the expectation $\mathbb{E}\varphi(x,L_i)$ to emphasize that although the underlying distribution is indeed $\mathcal{L}_{\mathfrak{D}}$ this is not important — this is the expectation over a single isolated predictor and it is the same for any sampling method that uses a training set of size b chosen i.i.d. from $\mathcal{D}$. Now, let us expand on the underbraced part:

$$\mathbb{E}_{\mathcal{L}_{\mathfrak{D}}}\left(\sum_i^m \varphi(x,L_i)\right)^2 = 2\sum_{i<j}\underbrace{\mathbb{E}_{\mathcal{L}_{\mathfrak{D}}}[\varphi(x,L_i)\varphi(x,L_j)]}_{(*)} + \sum_i^m \mathbb{E}\varphi^2(x,L_i)$$

We focus on (*), since $\mathbb{S}(\varphi_{\mathfrak{B}})$ and $\mathbb{S}(\varphi_{\mathfrak{D}})$ differ only there. By conditioning and decomposing the expectations:

$$\mathbb{E}_{\mathcal{L}_{\mathfrak{D}}}[\varphi(x,L_i)\varphi(x,L_j)]$$
$$= \sum_k \big(F_{\mathfrak{D}}(k;N,m,b) - F_{\mathfrak{D}}(k-1;N,m,b)\big)g(k;x)$$
$$= \sum_k F_{\mathfrak{D}}(k;N,m,b)\big(g(k;x) - g(k+1;x)\big)$$

with the last equality being a very handy observation. Doing the same for $\mathbb{S}(\varphi_{\mathfrak{B}})$:

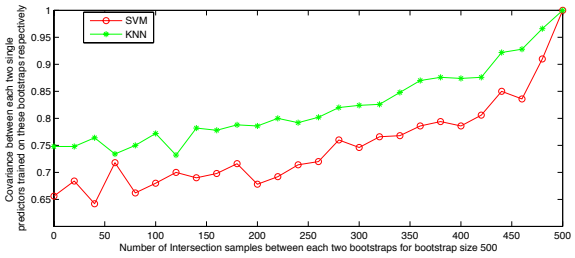

Figure 3: The mean covariance between single predictors trained on each bootstrap pair is a function of the intersection portion between these two bootstraps. The x-axis denotes the intersection size of two bootstraps (each 500 elements large). The y-axis denotes the mean covariance between two predictor outputs trained on these two bootstraps. This classification task is performed on the real dataset MNIST with single predictors trained using k-NN and SVM.

$$\mathbb{S}(\varphi_{\mathfrak{D}}) - \mathbb{S}(\varphi_{\mathfrak{B}}) = \frac{m-1}{m}\sum_k \big((F_{\mathfrak{D}}(k;N,m,b) - F_{\mathfrak{B}}(k;N,m,b))(g(k;x) - g(k+1;x))\big) < 0$$

where the inequality is by Lemma 2 and Assumption 2. □

This discussion is for regression. Something similar holds for classification (see full version).

Empirical results

We provide empirical evidence (i) supporting the covariance assumption and (ii) comparing bagging to design-bagging in classification and regression settings. Our study is on various base learners and data sets from the UCI repository and on a real data set MNIST. In the full version we *significantly* expand this experimental section including: (i) more real-world datasets (not limited to the UCI repository), (ii) experimental results for MNIST (not only for the assumption verification), (iii) statistical significance tests if necessary, and (iv) in a later version (but not the current full version) we may include extensions to random forests.

Assumption verification

In the main part we prove that under the covariance assumption design-bagging outperforms bagging (Theorem 2). We will analyze the correlation of classification and regression outputs on both real data set MNIST and data sets from the UCI repository, as well as on simulated data generated by adding multivariable low and medium energy Gaussian noise around hyperplanes in $\mathbb{R}^n$.

For MNIST, we uniformly at random choose 100 from the test data and 1000 from the training data considering 100 features. Figure 3 shows the correlation between the intersection portion of each two bootstraps in average and the covariance of each two single predictors trained on these

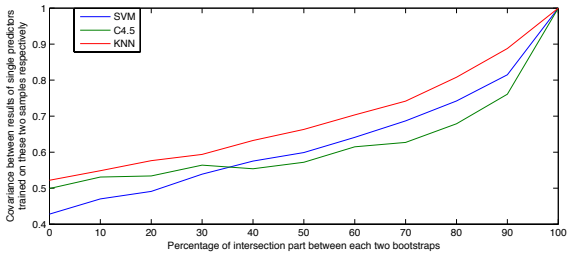

Figure 4: The mean covariance of the outputs of two predictors trained on a bootstrap pair as a function of the intersection portion between these two bootstraps. The x-axis denotes the percentage of the intersection of the two bootstraps. The y-axis denotes the mean covariance between the predictor outputs trained on these two bootstraps. This classification task is performed on an artificial dataset of a mixture of Gaussian distributed samples with single predictors trained using C4.5, k-NN and SVM.

Data set	Base	Bagging	Design	Algorithm
Fisher's Iris	70.67	73.33	74.00	SVM
Wine	92.50	94.81	95.37	C4.5
Ionosphere	87.91	90.24	91.00	C4.5
Wine Quality (MSE)	0.6386	0.4075	0.4048	CART

Table 1: Classification accuracy[%] and regression squared *error* rate using single base predictor, bagging (Breiman 1996) and design-bagging on different data set.

bootstraps respectively. We repeated the same experiment 1000 times for all classification tasks to remove the random noise. As we can see, from Figure 4, the functions for algorithm SVM and k nearest neighbors, i.e. k-NN ($k = 3$) are close to monotone increasing functions. In fact, the more repetition of experiments we perform, the function tends to be more strictly monotone.[1] Figure 4 and Figure 5 also indicates that this function is close to a monotone increasing one on a set of uniform randomly generated Gaussian distributed samples. To reduce sporadic effects, we repeated the same experiment 1000 times for all classification tasks and linear regression. For polynomial regression we repeated for 450K times.

Recognition results

We applied two algorithms (SVM and decision tree C4.5) to evaluate the recognition and regression performance boosted by the design bagging we introduced. Four well-known data sets are used Iris, Ionosphre (converted) for binary classification tasks, Wine as a multi-class (3 classes) classification task, and the Wine Quality as a regression task. The details of the data sets can be obtained from the UCI repository: number of samples, features as well as classes can be found in the full version. Bagging and design bagging are performed on 30 bootstraps ($m = 30$) and combined based on voting, and we set the number of samples in each bootstrap to $N/2$ (cf. (Bühlmann and Yu 2002; Friedman and Hall 2007) justifying this choice), where N is the number of training samples. To reduce the effect of outliers we run the same experiments 30 times and the results in average are shown in Table 1.

[1]Theoretically, measure concentration (as derived by an immediate calculation on the Chernoff bound) shows up later on ≈0.5M iterations/sample.

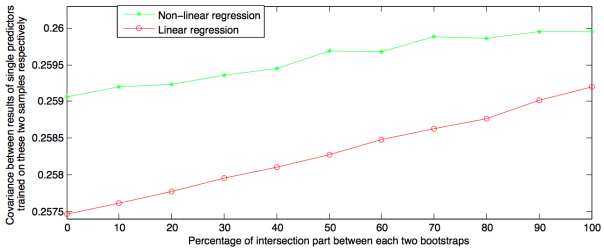

Figure 5: The mean covariance of the outputs of two predictors trained on a bootstrap pair as a function of the intersection portion between these two bootstraps. The x-axis denotes the percentage of the intersection part of two bootstraps. The y-axis denotes the difference of the covariance with the covariance at x=0 (0.2709 for linear regression and 0.2574 for non-linear regression); The covariance is measured between single predictor outputs trained on these two bootstraps. This regression task is performed on an artificial set of Gaussian noise added around hyperplane with single predictors trained using linear regression and low-degree polynomial regression.

On Fisher's Iris data, we applied a 10-fold cross-validation to evaluate a binary classification task, whether the class is the species 'versicolor'. We used the SVM in Matlab to train the single classifier. On the data set of Wine, Wine Quality and Ionosphere, we applied Decision Tree C4.5 to train the single classifier. The test set is 10% of the whole data set uniform randomly selected and the rest samples are taken as the training set for each task. Table 1 shows a consistent improvement in the classification accuracy and regression squared error rate using bootstraps selected with design bagging than that selected with the bagging of (Breiman 1996) and that of a single base predictor.

Conclusions and future work

This work is in its biggest part theoretical. Under two very general properties of classifiers we prove that bagging is not the optimal solution, where optimal refers to ideal-bagging. Implicit to our work is the idea that instead of proving directly that a certain ensemble method works, it removes complication and unnatural assumptions to compare with its ideal counterparts. One question we did not touch is what is the optimal *bagging-like* aggregation method. Finally, besides the foundational issues, we believe that our approach can find further applications in ensemble methods for which bagging is a building block. This latter part necessarily will involve large-scale experimentation.

Acknowledgements

We are most grateful to the anonymous reviewers who pointed a number of typos and inaccuracies and whose remarks significantly improved this paper and its full version.

This work was supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61033001, 61361136003, 61350110536.

References

Bauer, E., and Kohavi, R. 1999. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine learning* 36(1-2):105–139.

Biau, G.; Cérou, F.; and Guyader, A. 2010. On the rate of convergence of the bagged nearest neighbor estimate. *The Journal of Machine Learning Research* 11:687–712.

Biau, G.; Devroye, L.; and Lugosi, G. 2008. Consistency of random forests and other averaging classifiers. *The Journal of Machine Learning Research* 9:2015–2033.

Breiman, L. 1996. Bagging predictors. *Machine learning* 24(2):123–140.

Breiman, L. 2001. Random forests. *Machine learning* 45(1):5–32.

Bühlmann, P., and Yu, B. 2002. Analyzing bagging. *The Annals of Statistics* 30(4):927–961.

Colbourn, C. J., and Dinitz, J. H. 2010. *Handbook of combinatorial designs*. CRC press.

Denil, M.; Matheson, D.; and de Freitas, N. 2013. Narrowing the gap: Random forests in theory and in practice. *arXiv preprint arXiv:1310.1415*.

Dietterich, T. G. 2000. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine learning* 40(2):139–157.

Efron, B. 1979. Bootstrap methods: another look at the jackknife. *The annals of Statistics* 1–26.

Freund, Y., and Schapire, R. E. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* 55(1):119–139.

Friedman, J. H., and Hall, P. 2007. On bagging and nonlinear estimation. *Journal of statistical planning and inference* 137(3):669–683.

Hartman, T., and Raz, R. 2003. On the distribution of the number of roots of polynomials and explicit weak designs. *Random Structures & Algorithms* 23(3):235–263.

Kuncheva, L. I., and Whitaker, C. J. 2003. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning* 51(2):181–207.

Long, P., and Servedio, R. 2013. Algorithms and hardness results for parallel large margin learning. *Journal of Machine Learning Research* 14:3073–3096. (also NIPS'11).

Melville, P.; Shah, N.; Mihalkova, L.; and Mooney, R. J. 2004. Experiments on ensembles with missing and noisy data. In *Multiple Classifier Systems*. Springer. 293–302.

Tang, E. K.; Suganthan, P. N.; and Yao, X. 2006. An analysis of diversity measures. *Machine Learning* 65(1):247–271.

Ting, K. M., and Witten, I. H. 1997. Stacking bagged and dagged models. In *ICML*, 367–375.

Valiant, L. G. 1984. A theory of the learnable. *Communications of the ACM* 27(11):1134–1142.