Low Bias Bagged Support Vector Machines

Giorgio Valentini VALENTINI@DSI.UNIMI.IT
Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano, Italy
INFM, Istituto Nazionale per la Fisica della Materia, Italy.

Thomas G. Dietterich TGD@CS.ORST.EDU
Department of Computer Science, Oregon State University, Corvallis, OR 97331, USA

Abstract

Theoretical and experimental analyses of bagging indicate that it is primarily a variance reduction technique. This suggests that bagging should be applied to learning algorithms tuned to minimize bias, even at the cost of some increase in variance. We test this idea with Support Vector Machines (SVMs) by employing out-of-bag estimates of bias and variance to tune the SVMs. Experiments indicate that bagging of low-bias SVMs (the "lobag" algorithm) never hurts generalization performance and often improves it compared with well-tuned single SVMs and to bags of individually well-tuned SVMs.

1. Introduction

Bias–variance theory provides a way to analyze the behavior of learning algorithms and to explain the properties of ensembles of classifiers (Friedman, 1997; Domingos, 2000; James, 2003). Some ensemble methods increase expressive power of learning algorithms, thereby reducing bias (Freund & Schapire, 1996). Other ensemble methods, such as methods based on random selection of input examples and input features (Breiman, 2001; Buhlmann & Yu, 2002) reduce mainly the variance component of the error. The decomposition of the error into bias and variance can guide the design of ensemble methods by relating measurable properties of algorithms to the expected performance of ensembles (Valentini & Dietterich, 2002). In particular, bias–variance theory can tell us how to tune the individual classifiers in an ensemble so as to optimize the overall performance of the ensemble.

In this paper, we apply bias–variance analysis to direct the tuning of SVMs to optimize the performance of bagged ensembles. Specifically, since bagging is primarily a variance-reduction method, and since the overall error is (to a first approximation) the sum of bias and variance, this suggests that SVMs should be tuned to minimize bias before being combined by bagging. We propose a variant of bagging, that we call *Lobag* (*Lo*w bias *bag*ging), that estimates the bias of the SVM classifiers, selects low-bias classifiers, and then combines them by bootstrap aggregating.

Previous work with other classifiers is consistent with this approach. For example, several studies have reported that bagged ensembles of decision trees often give better results when the trees are not pruned (Bauer & Kohavi, 1999; Dietterich, 2000). Unpruned trees have low bias and high variance. Similarly, studies with neural networks have found that they should be trained with lower weight decay and/or larger numbers of epochs before bagging to maximize accuracy of the bagged ensemble (Andersen et al., 2001).

Unlike most learning algorithms, support vector machines have a built-in mechanism for variance reduction: from among all possible linear separators, they seek the maximum margin classifier. Hence, one might expect that bagging would not be very effective with SVMs. Previous work has produced varying results. On several real-world problems, bagged SVM ensembles are reported to give improvements over single SVMs (Kim et al., 2002). But for face detection, Buciu et al. (2001) report negative results for bagged SVMs.

A few other authors have explored methods for tuning SVMs in ensembles. Collobert et al. (2002) proposed solving very large scale classification problems by using meta-learning techniques combined with bagging. Derbeko et al. (2002) applied an optimization technique from mathematical finance to reduce the variance of SVMs.

The paper is structured as follows. In the following section we discuss differences and commonalities between Lobag and bagging, then we describe how to compute the bias–variance decomposition of the error. In Sect. 4, we outline the main results of an extended

analysis we performed on bias–variance decomposition of the error in random aggregated and bagged ensembles of SVMs, showing the effect of base learner aggregation on bias and variance. Then we present the Lobag algorithm and some numerical experiments to compare lobag ensembles of SVMs versus single SVMs and bagged ensembles of SVMs. An outline of future developments of this work concludes the paper.

2. Random aggregating, Bagging and Lobag

Let D be a set of m points drawn identically and independently from U according to P, where U is a population of labeled training data points $(\mathbf{x}_j, t_j)$, and $P(\mathbf{x}, t)$ is the joint distribution of the data points in U, with $\mathbf{x} \in \mathbb{R}^d$.

Let $\mathcal{L}$ be a learning algorithm, and define $f_D = \mathcal{L}(D)$ to be the predictor produced when $\mathcal{L}$ is applied to training set D. f_D produces a prediction $f_D(\mathbf{x}) = y$.

2.1. Random Aggregating

Suppose that a sequence of learning sets $\{D_k\}$ is given, each drawn i.i.d. from the same underlying distribution P. According to (Breiman, 1996), we can aggregate the f_D trained with different samples drawn from U to get a better predictor $f_A(\mathbf{x}, P)$. For regression problems $t_j \in \mathbb{R}$ and $f_A(\mathbf{x}, P) = E_D[f_D(\mathbf{x})]$, where $E_D[\cdot]$ indicates the expected value with respect to the distribution of D. In classification problems, $t_j \in \mathcal{C} \subset \mathbb{N}$ and $f_A(\mathbf{x}, P) = \arg\max_j |\{k|f_{D_k}(\mathbf{x}) = j\}|$ is the plurality vote of the individual predictors. Because the training sets D are randomly drawn from U, we call the procedure for building f_A *random aggregating*.

If T and $\mathbf{X}$ are random variables having distribution P, the expected squared loss EL for the single predictor $f_D(\mathbf{X})$ is $EL = E_D[E_{T,\mathbf{X}}[(T - f_D(\mathbf{X}))^2]]$ while the expected squared loss EL_A for the aggregated predictor is $EL_A = E_{T,\mathbf{X}}[(T - f_A(\mathbf{X}))^2]]$. Developing the square it is easy to see that

$$EL = E_T[T^2] + E_{\mathbf{X}}[E_D[f_D^2(\mathbf{X})]] - 2E_T[T]E_{\mathbf{X}}[f_A(\mathbf{X})] \quad (1)$$

$$EL_A = E_T[T^2] + E_{\mathbf{X}}[(E_D[f_D(\mathbf{X})])^2] - 2E_T[T]E_{\mathbf{X}}[f_A(\mathbf{X})]$$

and hence $EL_A \leq EL$ as

$$E_{\mathbf{X}}[(E_D[f_D(\mathbf{X})])^2] \leq E_{\mathbf{X}}[E_D[f_D^2(\mathbf{X})]]. \quad (2)$$

The reduction of the error depends on the instability of the prediction (Breiman, 1998), that is, on how unequal the two sides of eq. 2 are. This in turn is related to the variance $V(\mathbf{X})$ of the base predictor:

$$\begin{aligned} V(\mathbf{X}) &= E_D[(f_D(\mathbf{X}) - E_D[f_D(\mathbf{X})])^2] \\ &= E_D[f_D^2(\mathbf{X})] - E_D[f_D(\mathbf{X})]^2 \geq 0 \end{aligned} \quad (3)$$

The degree to which eq. 3 is greater than 0 depends on how unequal the two sides of eq. 2 are. For classification problems with unstable predictors, random aggregating can lower the expected error, but with poor predictors, unlike regression, aggregation can worsen the accuracy (Breiman, 1996).

2.2. Bagging: Bootstrap Aggregating

Base learners of bagged ensembles f_B are trained on repeated bootstrap samples $\{D^b\}$ from D. Hence *Bagging* is an approximation of random aggregating, for at least two reasons. First, bootstrap samples are not independent data samples: they are drawn from a data set D that is in turn a sample from the population U. Second, bootstrap samples are drawn from D according to the uniform probability distribution, which is only an approximation of the unknown true distribution P. For these reasons, there is no guarantee that bagging provides a good enough approximation to f_A to produce variance reduction.

Random aggregating removes all variance, leaving only bias and noise. Hence, if bagging is a good approximation to random aggregating, it will also remove all of the variance. Therefore, to minimize the overall error, bagging should be applied to base learners with minimum bias.

2.3. Lobag

We propose to tune SVMs to minimize the bias and then apply bagging to reduce (if not eliminate) variance, resulting in an ensemble with very low error. The key challenge, then, is to find a reasonable way of tuning SVMs to minimize their bias. The bias of SVMs is typically controlled by two parameters. First, recall that the objective function for (soft margin) SVMs has the form: $\|\mathbf{w}\|^2 + C\sum_i \xi_i$, where $\mathbf{w}$ is the vector of weights computed by the SVM and the ξ_i are the margin slacks, which are non-zero for data points that are not sufficiently separated by the decision boundary. The parameter C controls the tradeoff between fitting the data (achieved by driving the ξ_i's to zero) and maximizing the margin (achieved by driving $\|\mathbf{w}\|$ to zero). Setting C large should tend to minimize bias.

The second parameter that controls bias arises only in SVMs that employ parameterized kernels such as the polynomial kernel (where the parameter is the degree d of the polynomial) and RBF kernels (where the

parameter is the width σ of the gaussian kernel). In previous work we showed that in gaussian and polynomial SVMs, bias depends critically on these parameters (Valentini & Dietterich, 2002).

The basic idea of Lobag is to perform a systematic search of values of C and either d or σ and experimentally estimate the bias of the resulting SVMs to find the parameter values to minimize the estimated bias. These parameter settings are then employed to create a bagged SVM ensemble.

3. Computation of the bias–variance decomposition of the error

3.1. Bias and Variance for Classification

We employ the definitions of bias and variance for general loss functions developed by Domingos (2000). Let $L(t, y)$ be the loss received when the true label is t and the hypothesis predicts y. Then the *optimal prediction*, y_*, for point $\mathbf{x}$ is defined to be the value of y that minimizes $E_t[L(t, y)]$:

$$y_*(\mathbf{x}) = \arg\min_y E_t[L(t, y)]$$

For the usual 0/1 loss function, $y_* = t$. The *main prediction* y_m at point $\mathbf{x}$ is defined as the class label that would give the minimum expected loss over all training sets D:

$$y_m = \arg\min_{y'} E_D[L(f_D(\mathbf{x}), y')].$$

It expresses the "central tendency" of a learner.

The bias $B(\mathbf{x})$ of the learning algorithm at point $\mathbf{x}$ is the loss of the main prediction relative to the optimal prediction: $L(y_*, y_m)$. For 0/1 loss, this will either be zero if the main prediction is correct or one if the main prediction is wrong. Hence, for a given learning algorithm $\mathcal{L}$ and any point $\mathbf{x}$, the bias will either be 0 or 1. We will refer to the points $\mathbf{x}$ where the bias is zero as the "unbiased points". We will call the other points the "biased points".

The variance $V(\mathbf{x})$ is the expected loss of the individual predictions relative to the main prediction: $V(\mathbf{x}) = E_D[L(f_D(\mathbf{x}), y_m)]$.

These definitions of bias and variance do not give an additive decomposition of the error. Instead, for biased points, the expected 0/1 loss can be written as $B(\mathbf{x})-V(\mathbf{x})$, whereas for unbiased points, the expected 0/1 loss can be written as $B(x)+V(\mathbf{x})$. The intuition here is that if the learning algorithm is biased at point $\mathbf{x}$, then increased variation (away from the main prediction) will increase the chance of classifying $\mathbf{x}$ correctly. But if the learning algorithm is unbiased at $\mathbf{x}$, then increased variation (away from y_m) will increase the probability of misclassifying $\mathbf{x}$.

We can aggregate the bias and variance over a test data set D of m points as follows. The average bias B is

$$B = \frac{1}{m}\sum_{i=1}^{m} B(\mathbf{x}_i).$$

The unbiased variance, V_u is the average variance of the unbiased points,

$$V_u = \frac{1}{m_u}\sum_{\{i|B(\mathbf{x}_i)=0\}} V(\mathbf{x}_i),$$

where m_u denotes the number of unbiased points. The biased variance V_b is the average variance of the biased points:

$$V_b = \frac{1}{m_b}\sum_{\{i|B(\mathbf{x}_i)=1\}} V(\mathbf{x}_i),$$

where $m_b = m - m_u$ is the number of biased points. Define the net variance $V_n = V_u - V_b$. Then we obtain the overall decomposition that the expected 0/1 error on the test data set as $B + V_n$.

3.2. Measuring Bias and Variance

We propose to apply out-of-bag procedures (Breiman, 2001) to estimate the bias and variance of SVMs trained with various parameter settings. The procedure works as follows. First, we construct B bootstrap replicates of the available training data set $\mathcal{D}$ (e. g., $B = 200$): $D_1, \ldots, D_B$. Then we apply a learning algorithm $\mathcal{L}$ to each replicate D_b to obtain an hypothesis $f_b = \mathcal{L}(D_b)$. For each bootstrap replicate D_b, let $T_b = \mathcal{D}\backslash D_b$ be the ("out-of-bag") data points that do not appear in D_b. We apply hypothesis f_b to the examples in T_b and collect the results.

Consider a particular training example $(\mathbf{x}, t)$. On the average, this point will be in 63.2% of the bootstrap replicates D_b and hence in about 36.8% of the out-of-bag sets T_b. Let K be the number of times that $(\mathbf{x}, t)$ was out-of-bag; K will be approximately $0.368B$. The optimal prediction at $\mathbf{x}$ is just t. The main prediction y_m is the class that is most frequently predicted among the K predictions for $\mathbf{x}$. Hence, the bias is 0 if $y_m = t$ and 1 otherwise. The variance $V(\mathbf{x})$ is the fraction of times that $f_b(\mathbf{x}) \neq y_m$. Once the bias and variance have been computed for each individual point $\mathbf{x}$, they can be aggregated to give B, V_u, V_b, and V_n for the entire data set $\mathcal{D}$.

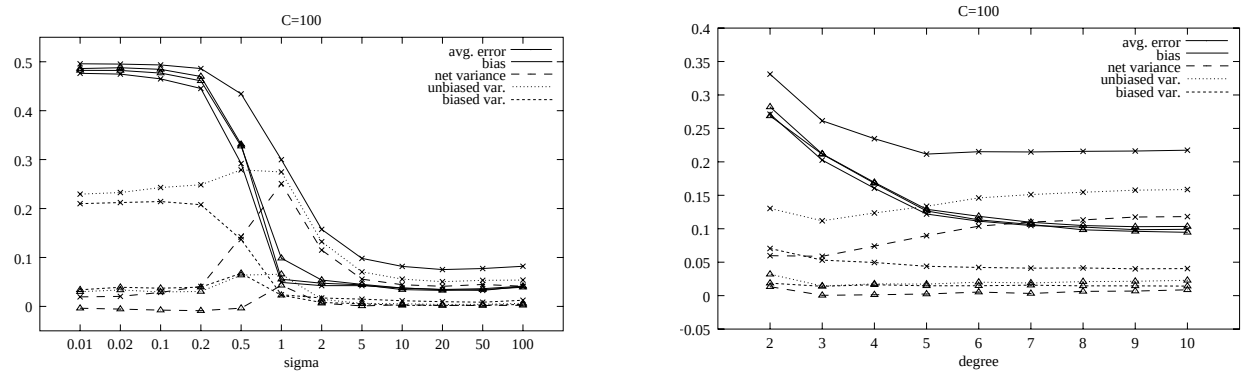

Figure 1. Comparison of bias-variance decomposition between single SVMs (lines labeled with crosses) and random aggregated SVM ensembles (lines labeled with triangles). (a) Gaussian kernels, varying σ with $C = 100$, Letter-Two data set (b) Polynomial kernels, varying the degree with $C = 100$, P2 data set.

4. Bias–variance decomposition of the error in single SVMs, random aggregated and bagged ensembles of SVMs

Before presenting our experimental tests of Lobag, we first summarize the main results of an extensive experimental analysis of the bias and variance of random aggregated and bagged ensembles of SVMs. The experiments involved the training and testing of more than 10 million SVMs. We employed several two-class data sets, both synthetic and "real". Most of them are from the UCI repository (Merz & Murphy, 1998). In all cases, we used relatively small data sets (100 examples) bootstrapped from a relatively large data set and reasonably large test sets to perform a reliable evaluation of bias and variance.

Fig. 1 shows results typical of those found in our experiments. The first surprise was that SVMs have high bias when the RBF width σ is set very small. This is counter-intuitive, since one would expect very low bias in these cases. In order to understand how the parameter settings affect the bias, we analyzed the relationships between σ, generalization error, training error, bias and the ratio of support vectors with respect to the total number of input examples (Fig. 2). For very small values of σ, overfitting problems arise: the training error is very small (about 0), while the number of support vectors is very high, and the error and bias are also high (Fig. 2). Indeed with very small σ values, the discriminant function learned by the SVMs is essentially flat everywhere except in the immediate neighborhood of each of the training data points (Fig. 3 a). Hence, the learned decision surface is very inflexible, and this gives high bias. By enlarging σ, we obtain a wider response on the input domain. The discriminant function computed by the SVM becomes smoother (Fig. 3 b), as the "bumps" around the support vectors become wider. This permits SVM to make better predictions on new examples. At the same time, the number of support vectors decreases, as do the bias and the generalization error (Fig. 2). Although σ is the parameter that most affects the RBF-SVMs, the regularization parameter C also plays an important role: it can counter-balance the increased bias due to large values of σ (Fig. 2). However, if its value is too small, the bias can increase independently of the other learning parameters.

Another important observation is that for single SVMs, the minimum of the error and the minimum of the bias are often achieved for different values of the tuning parameters C, d, and σ. This is what spurred us to consider the Lobag approach, which seeks the latter rather than the former.

The main purpose of the experiments was to analyze the effect of random aggregation (the theoretical ideal) and bagging. We found that random aggregation of SVMs gives relative error reductions of between 10 and 70% (depending on the data set). This reduction is slightly larger for high values of the C parameter and is due primarily to the reduction of the unbiased variance. Indeed in all data sets, the relative reduction of the unbiased variance amounts to about 90%, while bias remains substantially unchanged (Fig. 1). Note that the *error* of the ensemble is reduced to the *bias* of the single SVM, while net and unbiased variance are almost entirely eliminated.

Fig. 4 shows typical results of our experiments with bagging. In this case we also observed a reduction of

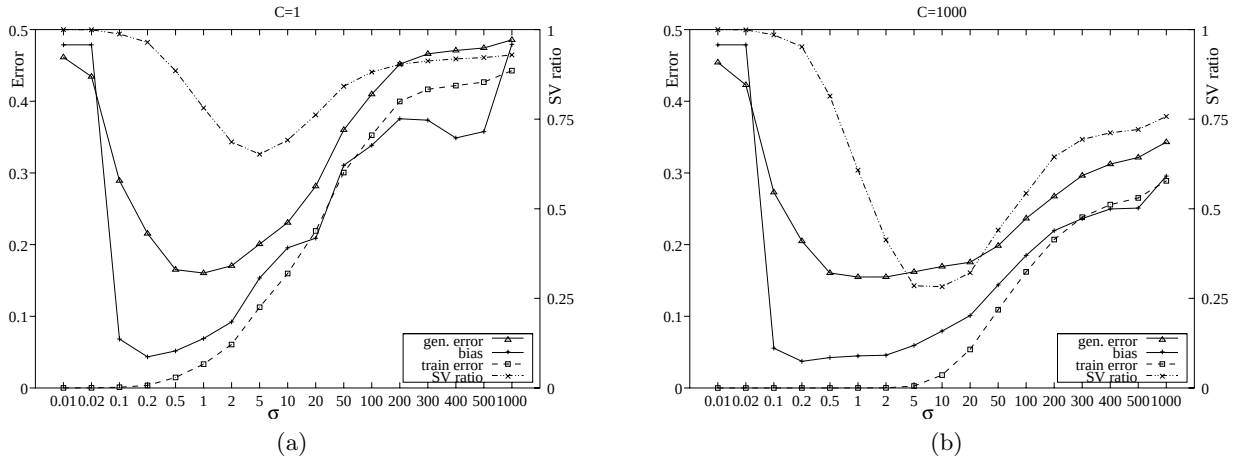

Figure 2. Estimated generalization error, training error, bias and support vector ratio while varying the σ parameter and for $C = 1$ (a) and $C = 1000$ (b) in the *P2* data set.

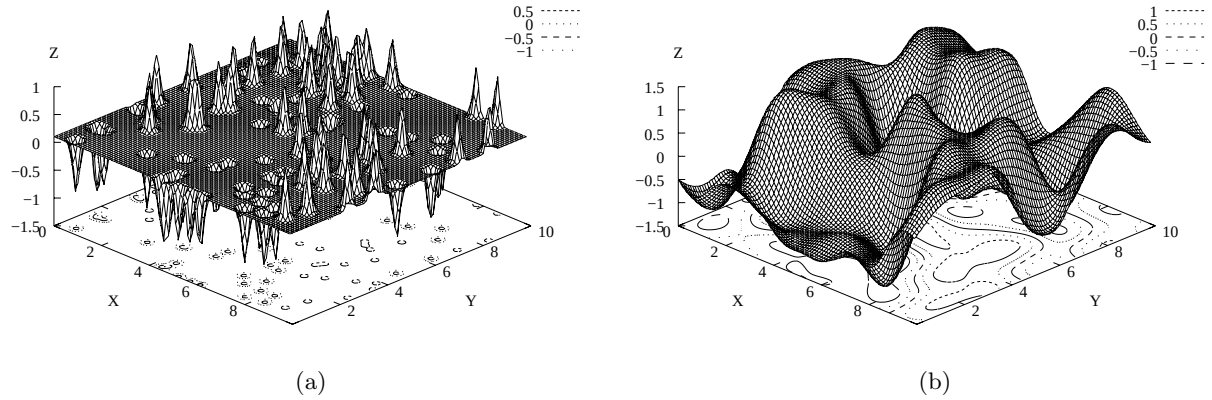

Figure 3. The discriminant function computed by the SVM on the *P2* data set: (a) $\sigma = 0.02, C = 1$, (b) $\sigma = 1, C = 1$

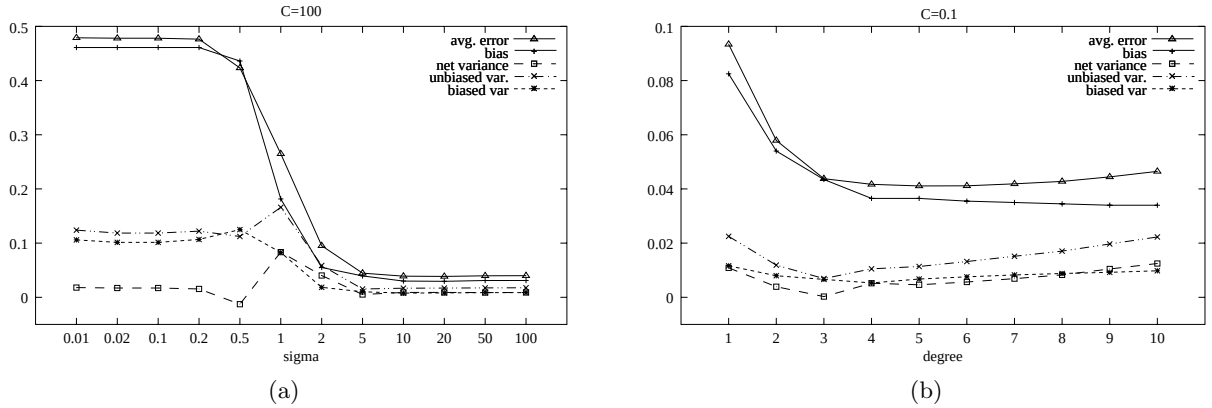

Figure 4. Bias-variance decomposition of error in bias, net variance, unbiased and biased variance in bagged SVMs (Grey-Landsat data set). (a) Gaussian kernel base learners, varying σ, $C = 100$ (b) Polynomial kernel base learners, varying the degree, $C = 0.1$.

the error, but it was not as large as with random aggregated ensembles. In particular, unlike random aggregating, net and unbiased variance are not reduced to 0: in our experiments, we obtained a smaller reduction

of the average error (from 0 to 20%) due to a lower decrease of the net-variance (about 35% on the average against a reduction of over 90% with random aggregated ensembles), while bias remained unchanged or increased slightly.

5. The lobag algorithm

The *Lobag* algorithm accepts the following inputs: (a) a data set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, with $\mathbf{x}_i \in \mathbb{R}$ and $y_i \in \{-1, 1\}$, (b) a learning algorithm $\mathcal{L}(\cdot, \alpha)$, with tuning parameters α, and (c) a set $\mathcal{A}$ of possible settings of the α parameters to try. *Lobag* estimates the bias of each parameter setting $\alpha \in \mathcal{A}$, chooses the setting α^* that minimizes the estimated bias, and applies the standard bagging algorithm to construct a bag of classifiers using $\mathcal{L}(\cdot, \alpha^*)$. The remainder of this section provides a high-level pseudo-code for *Lobag*.

5.1. The *Bias–variance decomposition* procedure

This procedure estimates the bias–variance decomposition of the error for a given learning algorithm $\mathcal{L}$ and learning parameters α.

The learning algorithm $\mathcal{L}$ returns a hypothesis $f_\alpha = \mathcal{L}(\mathcal{D}, \alpha)$ using a learning set $\mathcal{D}$, and it is applied to multiple bootstrap replicates D_b of the learning set $\mathcal{D}$ in order to generate a set $F_\alpha = \{f_\alpha^b\}_{b=1}^B$ of hypotheses f_α^b. The procedure returns the models F_α and the estimate of their loss and bias. For each set of learning parameters $\alpha \in \mathcal{A}$ it calls `Evaluate_BV`, a procedure that provides an out-of-bag estimate of the bias–variance decomposition.

Procedure $[V, \mathcal{F}]$ **BV_decomposition** $(\mathcal{L}, \mathcal{A}, \mathcal{D}, B)$
`Input`:
- Learning algorithm $\mathcal{L}$
- Set of algorithm parameters $\mathcal{A}$
- Data set $\mathcal{D}$
- Number of bootstrap samples B

`Output`:
- Set V of triplets $(\alpha, loss, bias)$, where $loss$ and $bias$ are the estimated loss and bias of the model trained through the learning algorithm $\mathcal{L}$ with algorithm parameters α.
- Set of ensembles $\mathcal{F} = \{F_\alpha\}_{\alpha \in \mathcal{A}}$ with $F_\alpha = \{f_\alpha^b\}_{b=1}^B$

```
begin procedure
  V = ∅
  F = ∅
  for each α ∈ A
  begin
    F_α = ∅
    T_α = ∅
    for each b from 1 to B
    begin
      D_b = Bootstrap_replicate(D)
      f_α^b = L(D_b, α)
      T_b = D \ D_b
      F_α = F_α ∪ f_α^b
      T_α = T_α ∪ T_b
    end
    F = F ∪ F_α
    [loss, bias, variance] = Evaluate_BV (F_α, T_α)
    V = V ∪ (α, loss, bias)
  end
end procedure.
```

5.2. The overall *Lobag* algorithm

Using the procedure `BV_decomposition`, we can implement a version of the Lobag algorithm that exhaustively explores a given set of learning parameters in order to build a low bias bagged ensemble.

Algorithm Lobag exhaustive
`Input:`
- Learning algorithm $\mathcal{L}$
- Set of algorithm parameters $\mathcal{A}$
- Data set $\mathcal{D}$
- Number of bootstrap samples B

`Output:`
- Selected Lobag ensemble : $F_{Lob} = \{f_{\alpha_B}^b\}_{b=1}^B$
- Selected bagged ensemble : $F_{Bag} = \{f_{\alpha_L}^b\}_{b=1}^B$
- OOB error of the Lobag ensemble : B_{min}
- OOB error of the bagged ensemble : $B_{L_{min}}$
- OOB error of the single model : L_{min}

```
begin algorithm
  V = ∅
  F = ∅
  [V, F] = BV_decomposition (L, A, D, B)
  [α_B, α_L, B_min, L_min, B_{L_min}] = Select_model (V)
  F_Lob = {f_{α_L}^b}_{b=1}^B = Select_ensemble (F, α_B)
  F_Bag = {f_{α_B}^b}_{b=1}^B = Select_ensemble (F, α_L)
end algorithm.
```

The procedure `Select_model` selects the model with the minimum bias and/or minimum loss and returns the parameter values α_B and α_L that correspond respectively to the model with minimum bias and minimum loss. Then the Lobag and bagged ensembles are chosen through the procedure `Select_ensemble`: the Lobag ensemble has base learners with the minimum estimated bias, while the bagged ensemble has base learners with the minimum estimated loss. In order to speed up the computation, we could design variants of the exhaustive Lobag algorithm. For example, we could apply multidimensional search methods, such as

the Powell's method, to select the tuning values that minimize bias.

6. Numerical experiments

We performed numerical experiments on different data sets to test the Lobag ensemble method using SVMs as base learners. We compared the results with single SVMs and classical bagged SVM ensembles.

Table 1. Results of the experiments using pairs of train $\mathcal{D}$ and test $\mathcal{T}$ sets. E_{lobag}, E_{bag}, and E_{SVM} stand respectively for estimated mean error of lobag, bagged, and single SVMs on the test set $\mathcal{T}$. The three last columns show the number of wins-ties-losses according to the McNemar's test, using 5 different training sets $\mathcal{D}$. **L/B**, **L/S**, and **B/S** stand respectively for the comparison Lobag/Bag, Lobag/Single SVM, and Bag/Single SVM.

Kernel type	$\mathbf{E_{lobag}}$	$\mathbf{E_{bag}}$	$\mathbf{E_{single}}$	**Win-tie-loss**		
				L/B	**L/S**	**B/S**
	Data set *P2*					
Polyn.	0.1687	0.1863	0.1892	4-1-0	4-1-0	1-4-0
Gauss.	0.1429	0.1534	0.1605	4-1-0	5-0-0	3-2-0
	Data set *Waveform*					
Linear	0.0811	0.0821	0.0955	2-3-0	5-0-0	5-0-0
Polyn.	0.0625	0.0677	0.0698	2-3-0	2-3-0	3-2-0
Gauss.	0.0574	0.0653	0.0666	4-1-0	4-1-0	2-3-0
	Data set *Grey-Landsat*					
Linear	0.0508	0.0510	0.0601	0-5-0	3-2-0	3-2-0
Polyn.	0.0432	0.0493	0.0535	1-4-0	2-3-0	1-4-0
Gauss.	0.0475	0.0486	0.0483	1-3-1	1-3-1	0-5-0
	Data set *Letter-Two*					
Linear	0.0832	0.0864	0.1011	0-5-0	4-1-0	4-1-0
Polyn.	0.0574	0.0584	0.0636	0-5-0	1-4-0	0-5-0
Gauss.	0.0668	0.0659	0.0701	0-5-0	0-5-0	0-5-0
	Data set *Letter-Two with added noise*					
Linear	0.3617	0.3662	0.3705	1-4-0	1-4-0	0-5-0
Polyn.	0.3587	0.3679	0.3812	0-5-0	2-3-0	0-5-0
Gauss.	0.3665	0.3813	0.3908	1-4-0	3-2-0	1-4-0
	Data set *Spam*					
Linear	0.1356	0.1340	0.1627	0-4-1	5-0-0	5-0-0
Polyn.	0.1309	0.1338	0.1388	1-4-0	2-3-0	2-2-1
Gauss.	0.1239	0.1349	0.1407	3-2-0	3-2-0	2-3-0
	Data set *Musk*					
Linear	0.1244	0.1247	0.1415	0-5-0	4-1-0	4-1-0
Polyn.	0.1039	0.1193	0.1192	4-1-0	4-0-1	2-2-1
Gauss.	0.0872	0.0972	0.0920	4-1-0	2-2-1	1-0-4

6.1. Experimental setup

We employed two synthetic data sets (*P2* and a two-class version of *Waveform*) and 5 "real" data sets (*Grey-Landsat*, *Letter* (reduced to the two-class problem of discriminating between the difficult letters B and R), *Letter* with 20% noise added, *Spam*, and *Musk*). Most of them are from the UCI repository (Merz & Murphy, 1998).

We employed small $\mathcal{D}$ training sets and large $\mathcal{T}$ test sets in order to obtain a reliable estimate of the generalization error: the number of examples for $\mathcal{D}$ was set to 100, while the size of $\mathcal{T}$ ranged from a few thousand for the "real" data sets to tens of thousands for synthetic data sets. Then we applied the Lobag algorithm described in Sect. 5, setting the number of samples bootstrapped from $\mathcal{D}$ to 100, and computing an out-of-bag estimate of the bias–variance decomposition of the error. The selected lobag, bagged, and single SVMs were finally tested on the separated test set $\mathcal{T}$. The experiments were repeated 5 times using 5 different training sets randomly drawn from each data set.

The C++ classes and applications we used to perform all the experiments are included in the *NEURObjects* library (Valentini & Masulli, 2002).

6.2. Results

Table 1 shows the results of the experiments. We compared lobag, bagging, and single SVMs with respect to 20 classification tasks: 7 data sets and 3 kernels (gaussian, polynomial, and dot-product) have been tested considering all the possible combinations, except for *P2*, for which we did not apply the dot-product kernel (because it was obviously inappropriate). We repeated each classification task 5 times, using randomly-drawn training sets of size 100 for each data set, obtaining in this way 100 (20×5) outcomes for each method. For each pair of methods, we applied McNemar's test (Dietterich, 1998) to determine whether there was a significant difference in predictive accuracy on the test set.

On nearly all the data sets, both bagging and Lobag outperform the single SVMs independently of the kernel used. The null hypothesis that Lobag has the same error rate as a single SVM is rejected at or below the 0.05 significance level in 58 of the 100 cases. Similarly, the null hypothesis that bagging has the same error rate as a single SVM is rejected at or below the 0.05 level in 39 of the 100 cases. Most importantly, Lobag generally outperforms standard bagging: it is significantly better than bagging in 32 of the 100 cases, and significantly inferior only twice (Tab. 2).

If we consider the win-tie-loss results per classification task (that is a method wins if it achieves significant better results according to McNemar's test in the majority of the corresponding experiments), lobag consistently outperforms single SVMs in 18 of the 20 tasks, and it never loses. Moreover, lobag wins with respect to bagging in 12 cases, and it loses only once (Tab. 2). While both lobag and bagging outperform single SVMs with dot-product kernels, lobag shows significantly better results than bagging with respect to single SVMs if gaussian or polynomial kernels are used. This is confirmed also by the direct comparison lobag and bagging: we can observe a larger difference in fa-

Table 2. Summary of the wins-ties-losses between the 3 proposed methods according to the McNemar's test, considering separately dot-product, polynomial, and gaussian kernels. **L/B**, **L/S**, and **B/S** stand respectively for the comparison Lobag/Bag, Lobag/Single SVM, and Bag/Single SVM.

Win-tie-loss on the overall data sets			
Kernel	**L/B**	**L/S**	**B/S**
Linear	3-26-1	23-7-0	21-9-0
Polyn.	12-23-0	17-17-1	9-24-2
Gauss.	17-17-1	18-15-2	9-22-4
Total	32-66-2	58-39-3	39-55-6
Win-tie-loss per classification task			
Kernel	**L/B**	**L/S**	**B/S**
Linear	2-3-1	6-0-0	5-1-0
Polyn.	5-2-0	7-0-0	5-2-0
Gauss.	5-2-0	5-2-0	4-2-1
Total	12-7-1	18-2-0	14-5-1

vor of lobag especially with gaussian and polynomial kernels (Tab. 2).

7. Conclusions and future work

This paper has shown that despite the ability of SVMs to manage the bias–variance tradeoff, SVM performance can generally be improved by bagging, at least for small training sets. Furthermore, the best way to tune the SVM parameters is to adjust them to minimize bias and then allow bagging to reduce variance.

In future work, we plan to apply Lobag to DNA microarray gene expression data which has very few training examples, but each example has very high dimension.

In our experiments, we did not consider noise. As a result, noise is folded into bias, and bias itself is overestimated. Even if the evaluation of noise in real data sets is an open problem (James, 2003), we plan to evaluate the role of the noise in synthetic and real data sets, in order to develop variants of Lobag specific for noisy data.

Acknowledgements

This work was partially funded by grant ITR 0085836 from the US National Science Foundation and by INFM, unità di Genova.

References

Andersen, T., Rimer, M., & Martinez, T. R. (2001). Optimal artificial neural network architecture selection for voting. *Proc. of IJCNN'01* (pp. 790–795). IEEE.

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

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*, 123–140. .

Breiman, L. (1998). Arcing classifiers. *The Annals of Statistics*, *26*, 801–849.

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*, 5–32.

Buciu, I., Kotropoulos, C., & Pitas, I. (2001). Combining Support Vector Machines for Accurate Face Detection. *Proc. of ICIP'01* (pp. 1054–1057).

Buhlmann, P., & Yu, B. (2002). Analyzing bagging. *Annals of Statistics*, *30*, 927–961.

Collobert, R., Bengio, S., & Bengio, Y. (2002). A Parallel Mixture of SVMs for Very Large Scale Problems. *Neural Computation*, *14*, 1105–1114.

Derbeko, P., El-Yaniv, R., & Meir, R. (2002). Variance Optimized Bagging. In *ECML 2002*, vol. 2430 of *LNCS*, pp. 60–71. Springer-Verlag.

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, *10*, 1895–1924.

Dietterich, T. G. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, *40*, 139–158.

Domingos, P. (2000). A Unified Bias-Variance Decomposition and its Applications. *Proc. of the* 17^{th} *ICML* (pp. 231–238).

Freund, Y., & Schapire, R. (1996). Experiments with a new boosting algorithm. *Proc. of the* 13^{th} *ICML* (pp. 148–156). Morgan Kauffman.

Friedman, J. (1997). On bias, variance, 0/1 loss and the curse of dimensionality. *Data Mining and Knowledge Discovery*, *1*, 55–77.

James, G. (2003). Variance and bias for general loss function. *Machine Learning*. (in press).

Kim, H., Pang, S., Je, H., Kim, D., & Bang, S. (2002). Pattern Classification Using Support Vector Machine Ensemble. *Proc. of ICPR'02* (pp. 20160–20163). IEEE.

Merz, C., & Murphy, P. (1998). UCI repository of machine learning databases.

Valentini, G., & Dietterich, T. (2002). Bias–variance analysis and ensembles of SVM. *MCS2002, Cagliari, Italy* (pp. 222–231). Springer-Verlag.

Valentini, G., & Masulli, F. (2002). NEURObjects: an object-oriented library for neural network development. *Neurocomputing*, *48*, 623–646.