Microbagging Estimators: An Ensemble Approach to Distance-weighted Classifiers

Blaine Nelson BLAINE.NELSON@WSII.UNI-TUEBINGEN.DE
Wilhelm Schickard Institute for Computer Science
University of Tübingen

Battista Biggio BATTISTA.BIGGIO@DIEE.UNICA.IT
Department of Electrical and Electronic Engineering
University of Cagliari

Pavel Laskov PAVEL.LASKOV@UNI-TUEBINGEN.DE
Wilhelm Schickard Institute for Computer Science
University of Tübingen

Editor: Chun-Nan Hsu and Wee Sun Lee

Abstract

Support vector machines (SVMs) have been the predominate approach to kernel-based classification. While SVMs have demonstrated excellent performance in many application domains, they are known to be sensitive to noise in their training dataset. Motivated by the equalizing effect of bagging classifiers, we present a novel approach to kernel-based classification that we call microbagging. This method bags all possible maximal-margin estimators between pairs of training points to create a novel linear kernel classifier with weights defined directly as functions of the pairwise distance matrix induced by the kernel function. We derive relationships between linear and distance-based classifiers and empirically compare microbagging to the SVMs and robust SVMs on several datasets.

1. Introduction

The concept of large-margin learning (Vapnik, 1998; Scholkopf and Smola, 2001) has become one of the cornerstones of modern data analysis. Its intuitive geometric interpretation as well as a clear connection to statistical theory are the key contributions to its success in numerous learning problems. Conceived for the two-class classification problem, it has been extended to regression (Vapnik, 1998), anomaly detection (Tax and Duin, 2004), clustering (Xu et al., 2004), ranking (Shashua and Levin, 2002) and other learning problems.

One of the less understood properties of large-margin learning algorithms is their robustness to distribution noise, *i.e.*, a systematic difference between the distributions of the training and test data. The crucial stationarity assumption underlying the theory of large-margin learning is that these distributions—albeit unknown—are the same. If this assumption is violated, the learning algorithm's performance on test data may significantly degrade unless additional assumptions can be made about the form of the non-stationarity.

Robustness to contamination of the distribution has been also well studied in the field of robust statistics (Hampel et al., 1986; Huber, 1981). Several theoretical approaches have been developed to characterize the behavior of empirical estimators under the presence of distribution noise. Of special interest is the notion of *influence function* which measures the sensitivity of an estimator to an infinitesimal contamination of a distribution. Although the methods of robust statistics are not directly applicable to large-margin learning, some ideas are closely related to successful learning algorithms. For example, the popular statistical technique of bagging (Breiman, 1996) using aggregation of bootstrapped estimators is reminiscent of boosting algorithms (Schapire, 1990) in which simple classifiers are combined to improve the classification performance.

An interesting relation between bagging and robustness of empirical estimation has been discovered by Grandvalet (2004). It was observed that bagging equalizes the influence of individual training points, which is a very desirable property from the robustness standpoint. Based on this observation, we propose a new method for a combination of bagging with large margin classification aimed at improving robustness of large-margin methods to distribution noise. Our method, which we call *microbagging*, is based on the idea of aggregating large-margin classifiers based on *smallest possible bootstrap samples* of the training data: pairs of points with the opposite labels. Due to the simplicity of the large margin *learning* on two data points, the resulting aggregate classifier can be easily analyzed and exhibits some interesting properties. First, the combination of decisions made by microbagging classifiers corresponds to the classical voting schemes used in the field of data fusion. Second, the combination of decision functions has a closed-form expression as a linear combination of the training points in feature space, as with other kernel methods. Finally, it can be shown that microbagging is related to distance-based learning and can be used by only defining an appropriate distance over the feature space.

To empirically validate the robustness of microbagging, we examine label noise in two-class classification. Label noise occurs when some labels of training points are flipped, hence the joint distribution of features and labels changes between the training and test data. We observe that there is a tradeoff between robustness and accuracy of classification exhibited by microbagged classifiers. Our experiments on UCI datasets demonstrate an increased robustness of the resulting aggregated classifier at a cost of a marginal deterioration of detection accuracy on uncontaminated data as compared to SVMs.

1.1. Notation

In this paper, we use the usual classification framework. The input space is denoted by $\mathcal{X}$ (D-dimensional) and the prediction space is $\mathcal{Y} = \{\text{'}-\text{'}, \text{'}+\text{'}\}$; where mathematically convenient, we use -1 and $+1$ in place of the labels '$-$' and '$+$', respectively. The observed training data points are paired: $\mathbb{D} = \{(\mathbf{x}_i, y_i)\}$ where $\mathbf{x}_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$. The size of the dataset is N. The *kernel function*, $\kappa(\cdot,\cdot) : \mathcal{X} \times \mathcal{X} \to \Re$, is the inner product function for the feature space defined by the feature map $\phi(\cdot)$; *i.e.*, $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$. The positive semi-definite kernel matrix $\mathbf{K}$ contains all pairwise kernel evaluations between data points; *i.e.*, $K_{i,j} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$. Alternatively, one can use the label-annotated matrix, $Q_{i,j} = y_i y_j \kappa(\mathbf{x}_i, \mathbf{x}_j)$. The kernel function also defines a norm function based on the usual relationships between inner products and norms. This kernel norm is given by $\|\phi(\mathbf{x})\|_k^2 = \kappa(\mathbf{x}, \mathbf{x})$ and induces the pairwise distance, $\|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|_k^2 = \kappa(\mathbf{x}_i, \mathbf{x}_i) + \kappa(\mathbf{x}_j, \mathbf{x}_j) - 2\kappa(\mathbf{x}_i, \mathbf{x}_j)$.

1.2. Kernel-based Classifiers

We consider the family of linear classifiers in a feature space, for which the decision function is expressed in terms of a hyperplane norm $\mathbf{w} = \sum_i \alpha_i y_i \phi(\mathbf{x}_i)$ and a bias b as

$$f(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x}) - b \quad = \quad \sum_i \alpha_i y_i \kappa(\mathbf{x}_i, \mathbf{x}) - b \tag{1}$$

where the coefficients $\boldsymbol{\alpha}$ and the bias b are learned from training data. The resulting classifier predicts '$-$' if $f(\mathbf{x}) < 0$ and '+' otherwise. Learning algorithms that produce such decision functions include the soft-margin SVM and kernelized perceptron.

1.2.1. Support Vector Machines

The primal objective for the soft-margin SVM is $W_{primal}(\mathbf{w}, b, \boldsymbol{\xi}) \triangleq \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_i \xi_i$ where C is the regularization parameter and ξ_i is the i^{th} slack variable (allowing for margin violations). This function is minimized with respect to $\mathbf{w}$ (and b) subject to the constraints that for every i, $y_i\left(\mathbf{w}^\top \phi(\mathbf{x}_i) - b\right) \geq 1 - \xi_i$. The corresponding dual objective is $W_{dual}(\boldsymbol{\alpha}) = \mathbf{1}_N{}^\top \boldsymbol{\alpha} - \frac{1}{2}\boldsymbol{\alpha}^\top \mathbf{Q}\boldsymbol{\alpha}$ where the dual variables $\boldsymbol{\alpha}$ must satisfy $\alpha_i \in [0, C]$ for all i and $\boldsymbol{\alpha}^\top \mathbf{y} = 0$.

The complementary-slackness conditions (*i.e.*, KKT conditions) for the dual objective above can be expressed as $\alpha_i\left(y_i\left(\mathbf{w}^\top\phi(\mathbf{x}_i) - b\right) - 1 + \xi_i\right) = 0$. These optimality conditions for the SVM along with the condition $\sum_i^N \alpha_i y_i = 0$ can be expressed as:

$$\begin{bmatrix} \mathbf{Q} & \mathbf{y} \\ \mathbf{y}^\top & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ b \end{bmatrix} = \begin{bmatrix} \mathbf{1}_N \\ 0 \end{bmatrix}$$

along with constraints on $\boldsymbol{\alpha}$ (*e.g.*, in the soft-margin SVM $\alpha_i \in [0, C]$). For the least-squares SVM (with a primal given by $W_{primal}(\mathbf{w}, b, \boldsymbol{\xi}) \triangleq \frac{1}{2}\|\mathbf{w}\|_2^2 + \frac{\gamma}{2}\sum_i \xi_i^2$), the complementary-slackness conditions also yield a similar equation with $\mathbf{Q}$ replaced by $\mathbf{Q} + \frac{1}{\gamma}\mathbf{I}^{(N\times N)}$.

2. Microbagging

Motivated by the idea that bagging equalizes the influence of data points (Grandvalet, 2004), we develop a techique we call *microbagging*. Bagging (bootstrap aggregating) is a process that creates K datasets from the original dataset by resampling $M \leq N$ data points *with replacement*.[1] As is discussed by Friedman and Hall (2000), bootstrapping can improve for $M < N$ which Grandvalet (2004) explains in terms of equalizing the influence of samples. Further, with more samples K, one can reduce the chance that a single point is overrepresented. By taking the limit of these trends, we develop a bagging-based approach with $M = 2$ and $K \to \infty$; this is equivalent to constructing a classification function from every pair of data points since the probability of each pair occuring is equal. However, this includes homogeneous pairs from the same class (and duplicates) for which a class-separating hyperplane is ill-defined. Instead, we reject these pairs and construct our bagged classifier by averaging only over the class-separating hyperplanes learned between every heterogeneous pair of data points—we call the resulting estimator a *microbagged classifier*.[2]

1. The original technique was introduced by Breiman (1996) using $M = N$ but subsequent work has also discussed sampling with $M < N$ (*e.g.*, Wehrens et al., 2000).
2. Microbagging can alternatively be defined as a limit of the *jackknife* or *cross-validation* subsampling (*e.g.*, Wehrens et al., 2000) which sample *without replacement* (rejecting homogenous pairs).

2.1. Pairwise Maximum-Margin Learning

Any maximum-margin hyperplane that separates a pair of heterogeneous training examples, $\mathbf{x}_i \in \mathbb{D}_X^{(+)}$ and $\mathbf{x}_j \in \mathbb{D}_X^{(-)}$, is specified by a normal vector proportional to the vector $\mathbf{x}_i - \mathbf{x}_j$ and a displacement that places their midpoint $\frac{\mathbf{x}_i+\mathbf{x}_j}{2}$ on the hyperplane. In feature space, these parameters $\mathbf{w}_{i,j}$ and $b_{i,j}$ are given by

$$\begin{aligned}\mathbf{w}_{i,j} &= \sigma_{i,j} \cdot [\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)] && (2)\\ b_{i,j} &= \frac{\sigma_{i,j}}{2}\left[\|\phi(\mathbf{x}_i)\|_k^2 - \|\phi(\mathbf{x}_j)\|_k^2\right] && (3)\end{aligned}$$

where $\sigma_{i,j} \geq 0$ is a scaling parameter, which we will discuss later. The parameters $\mathbf{w}_{i,j}$ and $b_{i,j}$ specify a decision function $f_{(i,j)}(\mathbf{x}) = \mathbf{w}_{i,j}{}^\top \phi(\mathbf{x}) - b_{i,j}$ whose sign yields our desired classifier. It has the following properties (in feature space): (1) it is linear (2) it separates $\mathbf{x}_i$ and $\mathbf{x}_j$, (3) it is correctly oriented to give the correct sign to both and (4) their midpoint (in feature space) lies on the hyperplane.

2.2. Microbagging the Classifiers

Using a strict bagging approach, all pairwise classifiers are averaged together to form the following aggregated decision function:

$$F(\mathbf{x}) = -1 + \frac{2}{N^+N^-} \sum_{i \in '+', j \in '-'} \mathrm{I}\left[\mathbf{w}_{i,j}{}^\top \mathbf{x} \geq b_{i,j}\right] .$$

By expanding the pairwise parameters from Eqs. (2) and (3), this decision function for the bagged classifier can be expressed as[3]

$$\begin{aligned}F(\mathbf{x}) &= -1 + \frac{2}{N^+N^-} \sum_{i \in '+', j \in '-'} \mathrm{I}\left[\kappa(\mathbf{x}_i, \mathbf{x}) - \kappa(\mathbf{x}_j, \mathbf{x}) \geq \tfrac{1}{2}[K_{i,i} - K_{j,j}]\right]\\ &= -1 + \frac{2}{N^+N^-} \sum_{i \in '+', j \in '-'} \mathrm{I}\left[\|\phi(\mathbf{x}_i) - \phi(\mathbf{x})\|_k \leq \|\phi(\mathbf{x}_j) - \phi(\mathbf{x})\|_k\right] .\end{aligned}$$

Since classifying based on an average of indicator functions is equivalent to taking their majority vote, this classifier is equivalent to a majority vote between all heterogeneous pairs of points. Thus, this bagged classifier has an intuitive robust quality since every outlier in the dataset can effect at most $\max\left[\frac{1}{N^+}, \frac{1}{N^-}\right]$ fraction of these decision functions. Moreover, because each bagged classifier has a range $\{-1, +1\}$, the impact of outliers is bounded.

2.3. Microbagging the Decision Functions

While the above strict bagging approach gives a direct sense of robustness, summations of indicator functions are difficult to analyze, yield a non-linear bagged classifier (even in the

3. For this aggregation approach, scale parameters $\sigma_{i,j} > 0$ do not impact on the resulting bagged classifier and thus are removed.

feature space), and are computationally expensive. We instead consider a classifier that bags the pairwise decision functions instead of classifiers given by :

$$\begin{aligned}\hat{F}(\mathbf{x}) &\triangleq \frac{1}{N^+N^-}\sum_{i\in'+',j\in'-'} f_{(i,j)}(\mathbf{x}) = \frac{1}{N^+N^-}\sum_{i\in'+',j\in'-'} \mathbf{w}_{i,j}{}^\top \phi(\mathbf{x}) - \frac{1}{N^+N^-}\sum_{i\in'+',j\in'-'} b_{i,j} \\ &= \mathbf{w}^\top\phi(\mathbf{x}) - b \ .\end{aligned}$$

The parameters $\mathbf{w}$ and b are the average normal and bias from all pairwise decision functions and can be expressed as follows in terms of parameters $\boldsymbol{\alpha}$ and b:

$$\begin{aligned}\mathbf{w} &\triangleq \frac{1}{N^+N^-}\sum_{i\in'+',j\in'-'} \mathbf{w}_{i,j} = \frac{1}{N^+N^-}\sum_{i\in'+',j\in'-'} \sigma_{i,j}\left(\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\right) \\ &= \sum_{i\in'+'}\underbrace{\sum_{j\in'-'}\frac{\sigma_{i,j}}{N^+N^-}}_{\alpha_i}\phi(\mathbf{x}_i) - \sum_{j\in'-'}\underbrace{\sum_{i\in'+'}\frac{\sigma_{i,j}}{N^+N^-}}_{\alpha_j}\phi(\mathbf{x}_j) = \sum_i \alpha_i y_i \phi(\mathbf{x}_i) \\ b &\triangleq \frac{1}{N^+N^-}\sum_{i\in'+',j\in'-'} b_{i,j} = \frac{1}{N^+N^-}\sum_{i\in'+',j\in'-'} \frac{\sigma_{i,j}}{2}\left[\|\phi(\mathbf{x}_i)\|_k^2 - \|\phi(\mathbf{x}_j)\|_k^2\right] \\ &= \frac{1}{2}\sum_i \alpha_i y_i \|\phi(\mathbf{x}_i)\|_k^2 = \frac{1}{2}\sum_i \alpha_i y_i K_{i,i} \\ \alpha_i &\triangleq \frac{1}{N^+N^-}\sum_j \mathrm{I}\left[y_i \neq y_j\right]\sigma_{i,j} \qquad (4)\end{aligned}$$

Thus we see that this decision function is expressible as a linear combination of kernel evaluations for each training data point weighted $\boldsymbol{\alpha}$:

$$\hat{F}(\mathbf{x}) = \sum_i \alpha_i y_i \kappa(\mathbf{x}_i, \mathbf{x}) - b = \sum_i \alpha_i y_i \left[\kappa(\mathbf{x}_i, \mathbf{x}) - \tfrac{1}{2}K_{i,i}\right] \qquad (5)$$

This form for the decision function is computationally convenient since all weights $\boldsymbol{\alpha}$ can be computed (in the training phase) with complexity $\mathcal{O}\left(N^2 \cdot \Sigma\right)$, where Σ is the complexity of computing each scaling, and the cost of a prediction is N kernel evaluations. Thus, training of microbagged classifiers is quadratic in the size of the training set as it requires all pairwise scalings; this is comparable to the training complexity of an SVM. However, unlike the computationally intensive optimization algorithm for SVM learning, microbagging only performs a simple sum over the scaling matrix $\boldsymbol{\sigma}$. Indeed, preliminary studies demonstrated that the running time of the SVM can be an order of magnitude worse than microbagging. For prediction, the lack of sparsity in the weights requires all training data to be compared to each query which generally makes microbagging several times slower than SVMs; both of these comparisons will appear in a long version of this paper.

2.4. Pairwise Hyperplane Scaling

Here we discuss strategies for selecting the scalings $\boldsymbol{\sigma}$ for the pairwise hyperplanes. Notably, the strategies we discuss are symmetric: $\sigma_{i,j} = \sigma_{j,i}$ as to be discussed in Section 3.2.

2.4.1. Uniform-weighted Microbagging

Here we consider the simplest microbagging estimator obtained by equally weighting each hyperplane; *i.e.*, $\sigma_{i,j} = 1$. We call such an estimator a *uniform-weighted microbagging estimator*. For this simple case, the weights $\boldsymbol{\alpha}$ can be simplified to

$$\alpha_i \;=\; \frac{1}{N^+N^-}\sum_j \mathrm{I}\left[y_i \neq y_j\right] = \begin{cases} \frac{1}{N^+} & \text{if } y_i = \text{'+'} \\ \frac{1}{N^-} & \text{if } y_i = \text{'}-\text{'} \end{cases} ;$$

that is, each data point is weighted inversely to the number of training points in its class, thus, equally distributing the weight among all data points of each class. These weights are insensitive to the location of the training data and thus can only be influenced by changing the class of training data; *e.g.*, due to label flipping.

2.4.2. Distance-weighted Microbagging

In this section, we consider a family of microbagging estimators for which the pairwise learners are weighted by a power $m \in \Re$ of the kernel distance between the two points; *i.e.*, $\sigma_{i,j} = \left\|\phi\left(\mathbf{x}_i\right) - \phi\left(\mathbf{x}_j\right)\right\|_k^m$. This family of scalings is motivated by the fact that the norm of the pairwise hyperplane $\mathbf{w}_{i,j}$ is simply $\left\|\mathbf{w}_{i,j}\right\|_k = \sqrt{\sigma_{i,j}}\left\|\phi\left(\mathbf{x}_i\right) - \phi\left(\mathbf{x}_j\right)\right\|_k$ and thus this scaling family allows us to control the norms of pairwise hyperplanes and their corresponding margins. For this family, the weights can be expressed as

$$\hat{\alpha}_i \;=\; \frac{1}{N^+N^-}\sum_{j:y_i\neq y_j} \left\|\phi\left(\mathbf{x}_i\right) - \phi\left(\mathbf{x}_j\right)\right\|_k^m \tag{6}$$

(defining $0^0 = 1$ for cases when $\mathbf{x}_i = \mathbf{x}_j$).

We now discuss the properties of this family. For $m = 0$, they yield uniform weights—hence, the uniform-weighted scalings are contained within this family. Further, when $m = -2$, all of the pairwise hyperplanes have unitary norm; *i.e.*, $\left\|\mathbf{w}_{i,j}\right\|_k = 1$. For this case, all pairwise hyperplanes contribute equally to the bagged estimator whereas for uniform-weighted scalings, the hyperplanes of more distant pairs have more influence on the resulting bagged normal. We thus call the case when $m = -2$ *equi-influence microbagging.*

When $m < 0$, we say that the corresponding classifier is *SVM-like*, because it distributes more weight on data points that lie closer to points from the other class; *e.g.*, closer to the boundary between the classes. However, unlike an SVM, this approach does not yield sparsity; *i.e.*, training points with $\alpha_i = 0$. This allocation strategy gives the estimator resilience to outliers, although it is vulnerable to inliers (*i.e.*, heterogeneous points that are very close or identical). This vulnerability is demonstrated empirically in Section 4.2.

In the case that $m > 0$, the microbagging classifier allocates more weight to points that are furthest from heterogeneous points. Again, the estimator does not yield SVM-like sparsity. This weighting strategy is insensitive to inliers but can be heavily influenced by outliers; *i.e.*, data that lies far away from all other training data.

3. Relationship to Distance-Weighted Voting Classifiers

We now explore a connection between kernel-based classifiers (such as microbagged classifiers and SVMs) and distance-weighted voting algorithms. Using algebra, the relationship between inner products and norms, and the condition $\boldsymbol{\alpha}^\top \mathbf{y} = 0$, the decision function in Eq. (1) can be rewritten in terms of kernel distances as $f(\mathbf{x}) = \frac{1}{2}\sum_i \alpha_i y_i \|\phi(\mathbf{x}_i)\|_k^2 - \frac{1}{2}\sum_i \alpha_i y_i \|\phi(\mathbf{x}_i) - \phi(\mathbf{x})\|_k^2 - b$. Since only the first term depends on the query data point $\mathbf{x}$, the classifier predicts '+' when

$$\frac{1}{N^+}\sum_{i\in'+'} N^+\alpha_i \|\phi(\mathbf{x}_i) - \phi(\mathbf{x})\|_k^2 - \frac{1}{N^-}\sum_{i\in'-'} N^-\alpha_i \|\phi(\mathbf{x}_i) - \phi(\mathbf{x})\|_k^2 \le 2b - \sum_i \alpha_i y_i \|\phi(\mathbf{x}_i)\|_k^2.$$

Hence the prediction depends on the difference between average squared distances of the query point from both classes and uses a constant threshold $\beta \triangleq 2b - \sum_i \alpha_i y_i \|\phi(\mathbf{x}_i)\|_k^2$ depending only on training data points; if also $\beta = 0$, then the above decision is equivalent to distance-weighted classification.

The relationship between linear classifiers and distance-weighted voting classifiers depends on the following two conditions for any linear classifier:

$$\sum_i \alpha_i y_i = 0 \qquad\qquad b = \frac{1}{2}\sum_i \alpha_i y_i \|\phi(\mathbf{x}_i)\|_k^2 \ . \tag{7}$$

When the first condition is met, we say that the values of $\boldsymbol{\alpha}$ are *class-balanced.* When the second condition is met (*i.e.*, $\beta = 0$), we say that the bias b is *class-balanced.* If both $\boldsymbol{\alpha}$ and b are class-balanced, the prediction of the linear classifier can be expressed as

$$\operatorname*{argmin}_{c\in\{'+','-'\}} \frac{1}{N^c}\sum_{i\in c} N^c\alpha_i \|\phi(\mathbf{x}_i) - \phi(\mathbf{x})\|_k^2 \ ; \tag{8}$$

that is, the prediction is the class, c, that minimizes the average squared distance from the query to the class' set of training points re-weighted by α_i times the size of the class N^c.

3.1. SVMs as Distance-Weighted Voting

The formulation for SVMs in Section 1.2 shows that both soft-margin and least-squares SVMs satisfy class balance for $\boldsymbol{\alpha}$, but do not generally achieve class balance for b. Instead, the bias b of SVMs is selected to satisfy the margin constraints for its support vectors. These criteria for bias-selection do not generally coincide. However, as we discuss at the end of this section, these differing strategies result in two-different balancing strategies.

3.2. Microbagging as Distance-Weighted Voting

Unlike the SVM, the microbagged classifier achieves both class-balance conditions. Firstly, assuming the scalings $\boldsymbol{\sigma}$ are symmetric, the first condition from Eq. (7) is met. To see this, we expand the balance condition using the definition of positive and negative weights:

$$\begin{aligned}\sum_{i\in'+'} y_i\alpha_i &= \sum_{i\in'+'} \frac{1}{N^+N^-}\sum_j \mathrm{I}\left[y_i \neq y_j\right]\sigma_{i,j} - \sum_{i\in'-'} \frac{1}{N^+N^-}\sum_j \mathrm{I}\left[y_i \neq y_j\right]\sigma_{i,j} \\ &= \frac{1}{N^+N^-}\sum_{i,j} \mathrm{I}\left[y_i \neq y_j\right](\sigma_{i,j} - \sigma_{i,j}) \ .\end{aligned}$$

For this sum to be zero, it is sufficient that $\boldsymbol{\sigma}$ is symmetric, which holds for the scalings considered in Section 2.4. Moreover, the second balance condition in Eq. (7) requires that the bias b is exactly the bias we derived for the microbagging estimator above.

Based on class-balance, the decision function of the microbagged estimator is equivalent to a classifier that predicts the class that minimizes the weighted average distance between the query $\mathbf{x}$ and its training points as in Eq. (8). Notably, for uniform scalings (for which $\alpha_i = \frac{1}{N^+}$ for $i \in$ '+' and $\alpha_i = \frac{1}{N^-}$ for $i \in$ '$-$'), the class predicted by the microbagged classifier minimizes the average distance from the query to the class' training points.

3.3. Class Balance versus Variance Equalization

As noted above, the biases for the SVM and microbagging classifiers are based on different strategies. The SVM bias is selected to achieve margin constraints for the support vectors but does not generally achieve the class balance condition, $\beta = 0$. In failing to achieve class-balance, when trained on an unbalanced dataset, it will favor the majority class and thus requires re-weighting (*e.g.*, Vapnik, 1998). In contrast, the microbagged bias does satisfy class-balance (data points are implicitly re-weighted since their weights are multiplied by the class size in Eq. (8)), but has poor behavior when the intraclass variance of the two classes is dramatically different. That is, as is evidenced by Eq. (8), distance-weighted classifiers bias toward the class with smaller intraclass variance. SVMs do not exhibit this bias. Thus, we see that the strategy for choosing b results in different potential biases by the classifier—either a class bias or a class-variance bias.

4. Experiments

We use a toy dataset to assess the basic properties of microbagging, then evaluate its performance on real-world datasets using a random label-noise model. We compare SVMs classifiers to microbagging estimators. We evaluate both soft margin SVMs and least-squares SVMs as discussed in Section 1.2. For both, we use cross-validation to select their parameters; $C \in \{0.01, 0.1, 0.5, 1, 5, 10, 20, 50, 100, 200, 500, 1000\}$ and $\gamma \in \{0.0001, 0.001, 0.005, 0.01, 0.05, 0.1, 1, 5, 10, 100, 1000\}$. Once a value is selected, the SVM is trained on the full training dataset using that value. In all plots, SVMs appear as dashed lines. For microbagging, we consider both uniform-weighting ($m = 0$) and distance-weightings ($m = +2$ and $m = -2$). The microbagging results are plotted with solid lines.

We assess our algorithms using two standard metrics: classification accuracy and area under the ROC curve (AUC). Both are estimated using a test set of data drawn from the same distribution as the training data. The classification accuracy is the total number of correct predictions divided by the total number of predictions. The AUC is computed using the Mann-Whitney U-statistic divided by the total number of predictions in each class.

4.1. Toy Dataset: Uniform Boxes

In this experiment, we use a toy dataset, *uniform boxes*, to understand how microbagging estimators behave compared to SVMs. In the uniform boxes dataset, the positive and negative dataset are uniformly sampled from the D-dimensional unit hypercubes $[T, 1+T] \times [0,1]^{D-1}$ and $[-(1+T), -T] \times [0,1]^{D-1}$, respectively. This dataset is characterized by its dimensionality, D, its overlap, T, the size of the training set, S, and the proportion, P, of positive training points. We vary each of these parameters individually while fixing the others. The table below contains the parameters' ranges with their default value in bold.

D	1, 2, 3, 4, 5, **10**, 15, 20, 25, 30, 35, 40, 45, 50, 60, 70, 80, 90, 100, 120, 140, 160, 180, 200
T	−0.5, −0.4, −0.3, −0.2, −0.1, 0.0, **0.1**, 0.2, 0.3, 0.4, 0.5
S	20, 40, 60, 80, 100, 120, 140, 160, 180, **200**, 240, 280, 320, 360, 400
P	0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, **0.5**

In Figure 1, we compare microbagging estimators and SVMs across these ranges of parameters with a linear kernel.[4] We compare a soft-margin SVM (red dashed line), a least-squares SVM (blue-dashed line), and microbagged classifiers using distance-weighted scalings with $m = 0$ (black solid line), $m = -2$ (magenta solid line), and $m = +2$ (cyan solid line). The paired graphs show the corresponding accuracy and AUC for the algorithms averaged over 200 trials. The plots include error bars although they are often too small to be visible.

The results of Figure 1 demonstrate that microbagging is a competitive classifier on this toy dataset. We note that the soft-margin SVM is actually less resilient to increasing data dimensionality (with fixed training set size) than either the least-squares SVM or the microbagging estimators and the soft-margin SVM also performs slightly worse on very small datasets. We also note that both the soft-margin and least-squares SVM have unusual performance characteristics on class-skewed data—in fact, these effects are due to class bias in the SVM and can be corrected by using class-re-weighting (*e.g.*, Vapnik, 1998). Notably, re-weighting is not necessary for the microbagging estimators as we discussed above.

4.1.1. Weight Distributions

Here we evaluate how the different kernel classifiers allocate their weights over their training data on the uniform boxes dataset. In Figure 2, we show how the three learning algorithms distribute their $\boldsymbol{\alpha}$ values. For the i^{th} training point, we plot the absolute value of its α_i value (renormalized to one) against its distance to the learner's hyperplane. The microbagging estimators (leftmost plot) behave as predicted: when $m = 0$ the distribution is flat, when $m > 0$ the weight increases with increasing distance from hyperplane, and when $m < 0$ the weight concentrates on points nearest to the hyperplane (and thus nearby the opposing class). The soft-margin SVM (center plot) concentrates all its weight nearly equally on the points nearest to the hyperplane (*i.e.*, on the support vectors) and as C increases the number of points in that support decreases. Finally for the least-squares SVM (rightmost plot) for finite values of the parameter γ, the weight concentrates mostly in the neighborhood of the hyperplane but it decreases in a smooth fashion with greater distance. However, when $\gamma = \infty$ (hard-margin), the weight increases with the point's distance from the hyperplane.

4.2. UCI Datasets

Here we compare the performance of microbagging estimators and SVM estimators on five datasets from the UCI repository (Frank and Asuncion, 2010).[5] For these datasets, we use

4. We also obtained similar results using an rbf kernel with $\omega = 1.0$.
5. We subsequently studied microbagging on an additional five UCI datasets but did not find significant new results. These additional studies have been omitted due to space but will appear in a longer paper.

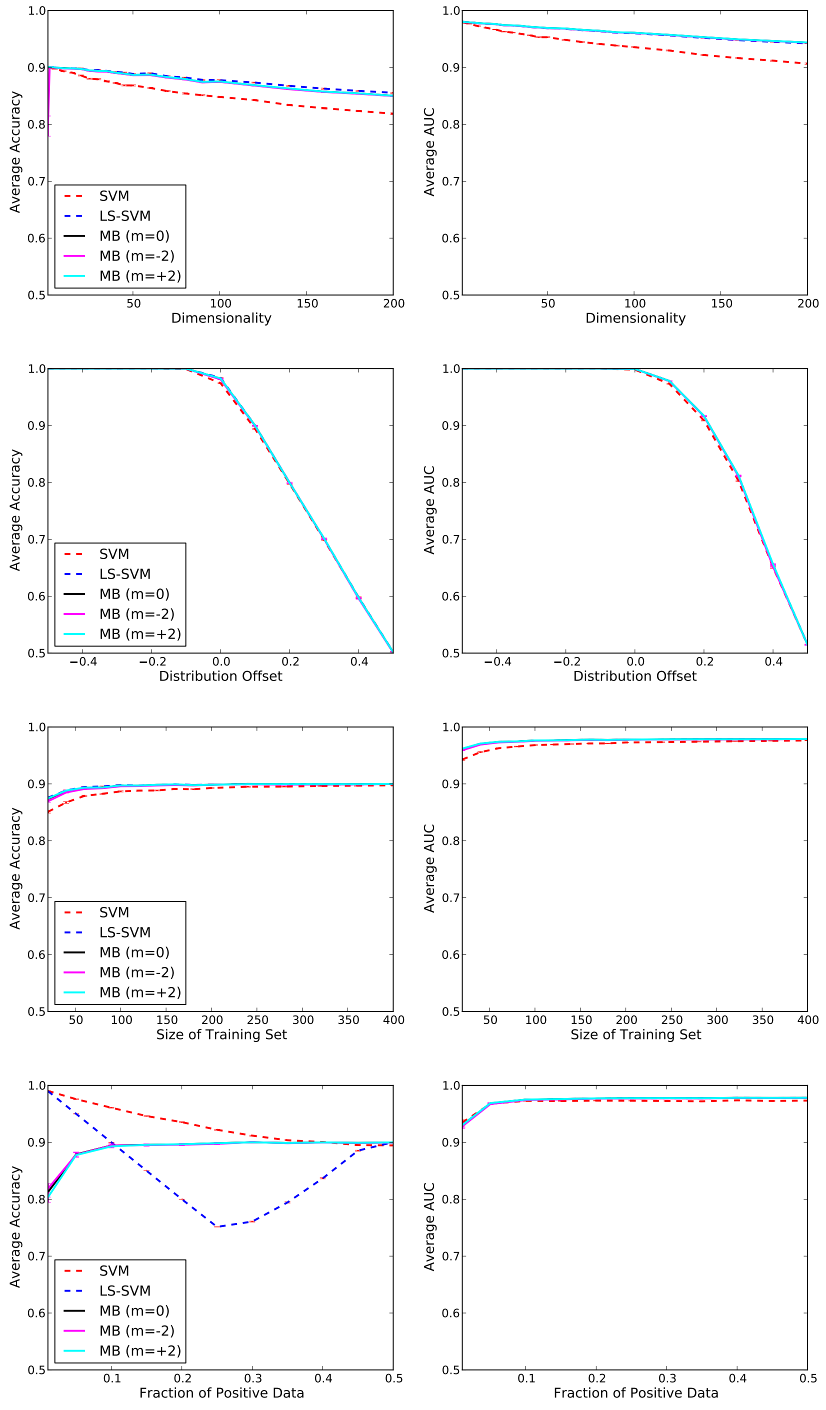


Figure 1: Plots of average model accuracy (left column) and AUC (right column) on the uniform boxes dataset. In the first row, dimensionality varies, in the second distribution overlap varies, in the third training set size varies, and in the fourth fraction of positive points varies.

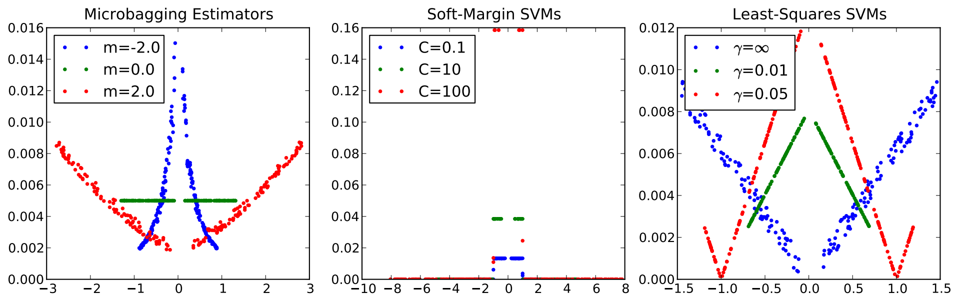


Figure 2: Distribution of $\boldsymbol{\alpha}$ for three different learning strategies: microbagging (left), soft-margin SVMs (center), and least-squares SVMs (right). Each shows the absolute value of the parameters α_i against the distance of the i^{th} training point to the hyperplane for different parameterizations of the learning algorithm. The values $\boldsymbol{\alpha}$ are renormalized to sum to one to show their relative distributions.

the rbf kernel with values of ω selected based on preliminary analysis of the data. The table below lists the properties of the five datasets we present in this paper: the dataset's name, its dimensionality, the label used as the positive and negative class (and the number of examples), and the size of the samples used for training and evaluation (training and evaluation samples do not overlap). For the Wine dataset, the data was standardized as recommended by its publishers and for the Iris dataset, three duplicates were removed.

Name	D	'+' Class (size)	'−' Class (size)	Sample Size	ω
Iris	4	setosa (48)	versicolor(50)	40 (50% '+')	5.0E−3
Ecoli	8	cp (143)	im (77)	50 (50% '+')	1.0E−3
Wine	13	1 (59)	2 (71)	50 (50% '+')	1.0E−4
Image Segmentation	19	1 (330)	2 (330)	200 (50% '+')	1.0E−6
MiniBooNE	50	0 (36499)	1 (93565)	200 (50% '+')	1.0E−5

For each dataset, we compare microbagging classifiers to SVMs with random label flipping. For each trial, we randomly select non-overlapping data to serve as the training and test data. We then select a percentage $P \in \{$ 0%, 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, 50%$\}$ of random labels to flip and record the accuracy and AUC for each classifier. We repeat these trials 200 times and present the average performance in Figures 3 and 4.

There are two prominent trends in these results. First, microbagging classifiers demonstrate equal or lower performance to SVMs under low label noise but better maintain their performance under high noise. On the Iris and Image Segmentation datasets, the microbagging estimators begin with roughly equal performance to the SVMs, on the Ecoli dataset, their initial performance is slightly worse, and they are significantly worse on the Wine and MiniBoonE datasets. However, the performance of the microbagging estimators ($m = 0$ and $m = +2$) surpasses the SVMs' performance at 25–30% label noise. Second, the *SVM-like* microbagging classifier ($m = -2$) performs worse under label noise than the other microbagging estimators and on the Image Segmentation dataset, it performs significantly worse than all classifiers even with low noise. As discussed in Section 2.4, *SVM-like* mi-

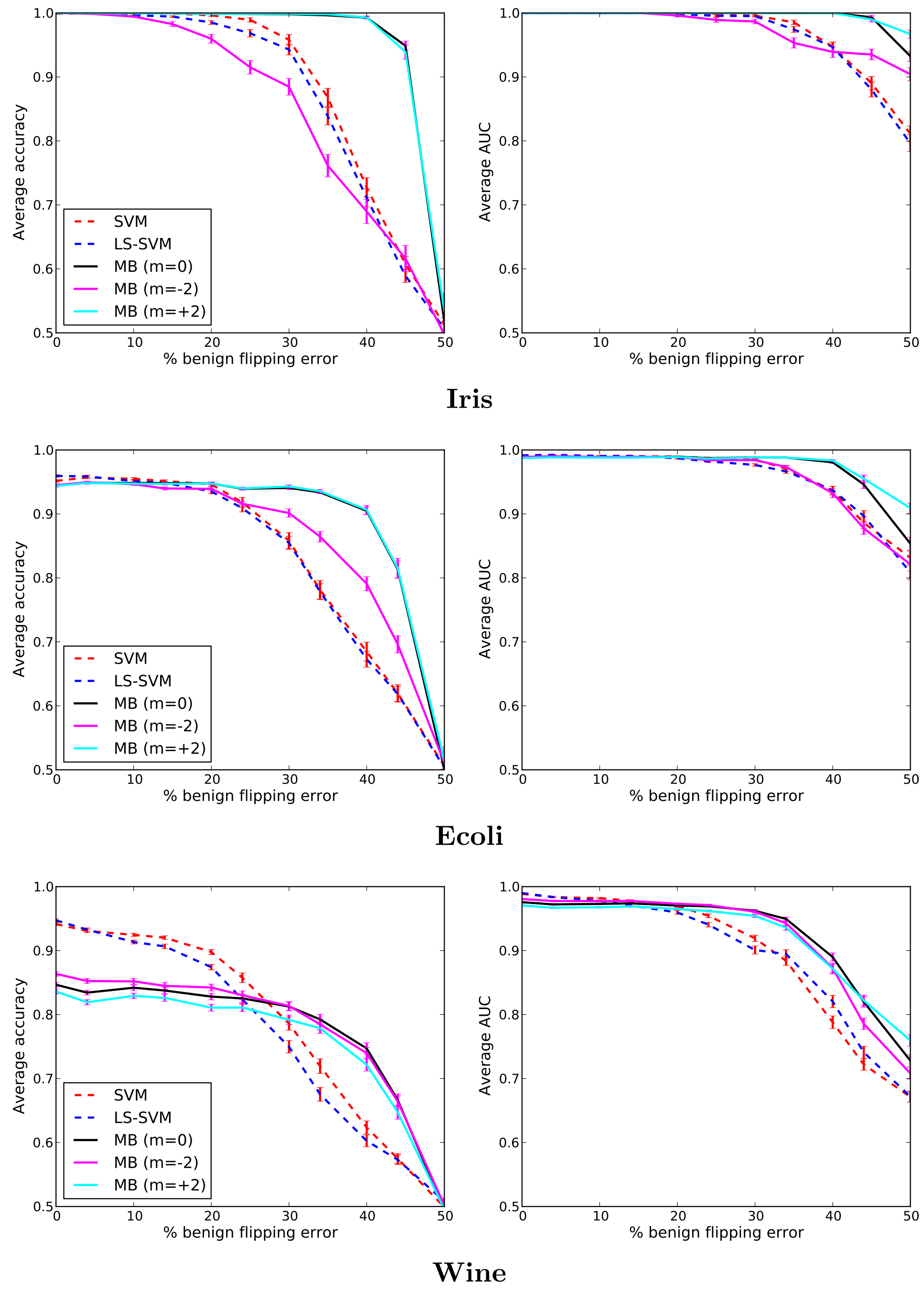
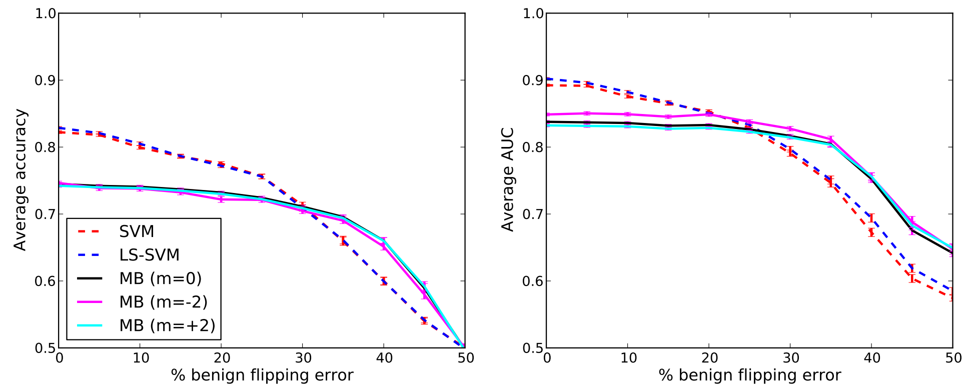


Figure 3: Performance of SVM and microbagging classifiers on the Iris (top), Ecoli (middle), and Wine (bottom) UCI datasets. In the left column we plot the average accuracy versus fraction of randomly flipped labels. In the right column, we plot average AUC versus fraction of randomly flipped labels. Note that all graphs range from 0.5 to 1.0 on the y-axis.



MiniBooNE

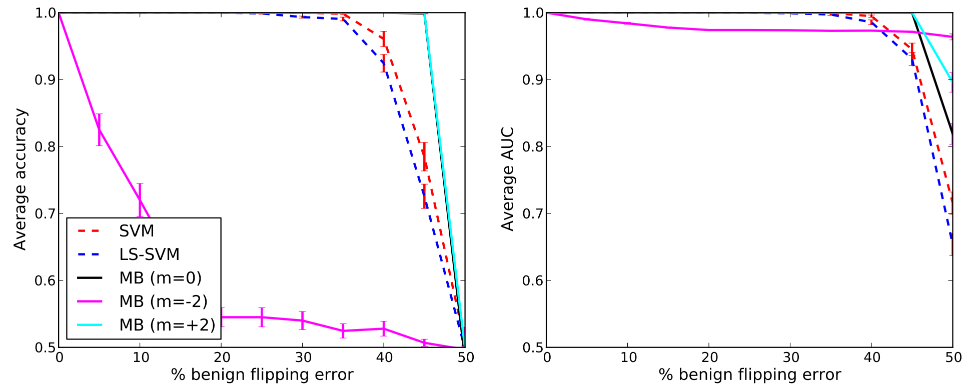


Image Segmentation

Figure 4: Performance of SVM and microbagging classifiers on the MiniBooNE (top), and Image Segmentation (bottom) UCI datasets. In the left column we plot the average accuracy versus fraction of randomly flipped labels. In the right column, we plot average AUC versus fraction of randomly flipped labels. Note that all graphs range from 0.5 to 1.0 on the y-axis.

crobagging classifiers are sensitive to inliers, which is the type of error that arises from label flipping. Hence, it is unsurprising that *SVM-like* microbagging was the least impressive.

5. Related Work

Robust learning has recently been identified as an important direction for a variety of applications (*e.g.*, Bagnell, 2005). Several researchers have identified robustness problems with soft-margin SVMs and proposed modifications that address these problems. Globerson and Roweis (2006) formulate a modified SVM designed to be robust in its choice of features against potential feature deletion. Alternative approaches have used modified SVM formulations to identify and exclude outliers while training (*e.g.*, Xu et al., 2006).

Rather than altering the SVM formulation, our approach relies on the equalizing effect of bagging to provide robustness. Naturally, there are many proposed frameworks for bagging SVMs (Kim et al., 2002), however to our knowledge, this is the first for microbagging.

As we show in Section 3, kernel-based linear classifiers can also be cast as distance-weighted voting algorithms (*e.g.* Mitchell, 1997, Chapter 8.2.1). There have been many modifications of this framework including inverse-distance weights (Dudani, 1976) and an adoption as a kernel-based method (Zuo et al., 2007). We show that the microbagged classifiers predict the class whose training data minimizes the average weighted distance to the query $\mathbf{x}$. Thus, microbagging generally is a form of Shepard's Method (Shepard, 1968) since the average is applied to all training data. However, our weights derive directly from our bagging methodology and, to our knowledge, have a form not previously used in the literature. In fact, as pointed out by Zuo et al. (2007), many of the previous weighting schemes were heuristics whereas our weights naturally arise from our microbagging methodology.

6. Conclusions and Future Work

In this paper, we present a new family of kernel-based estimators. Motivated by the fact that bagging equalizes influence among the training data, we derived microbagging classifiers to be less sensitive to noise in the training data than SVMs. As we show, these microbagging estimators have closed-form expressions for their weights that can be computed from the pairwise distances between the training data points. Further, we show that the microbagging classifiers indeed equalize influence over the dataset, and are equivalent to a distance-weighted voting classifiers but with a novel weighting scheme that arises naturally from the bagging framework. Finally, we show that different weighting schemes and bias choices yield different vulnerabilities and different classes biases.

To assess the efficacy of microbagging, we compare its performance to that of soft-margin and least-squares SVMs on both toy and real-world datasets. On the toy dataset, we demonstrate that microbagging is a competitive classifier that scales well both with the dimensionality of the dataset, its size, and the degree of class overlap. We also demonstrate that it is indeed a class-balanced classifier. On the real-world UCI datasets, microbagging demonstrates a tradeoff between accuracy on clean data and robustness to label flipping noise. For all the datasets, SVMs perform as well or better than microbagging on clean data, but microbagging classifiers generally perform better at noise levels above 25%. However, particularly on the Image Segmentation dataset, we show that *SVM-like* microbagging ($m < 0$) can be adversely affected by inliers. As we discuss below, we are currently considering a trimming strategy to reduce the impact of inliers and outliers on microbagged classifiers.

Efficiency and Scalability As discussed above, training of microbagged classifiers with distance-weighted scalings is quadratic in the size of the training set for any kernel as it requires all pairwise distances; this is comparable to the training complexity of an SVM. Empirically, we observe that training SVMs is an order of magnitude slower than microbagging since microbagging only performs a sum over the distance matrix whereas SVMs require a sophisticated optimization algorithm. For prediction, however, the lack of sparsity in the weights generally makes microbagging several times slower than SVMs. Microbagging complexities could potentially be improved by using metric trees and approximating the distances that exceed a certain threshold; we leave these proposals for future work.

6.1. Hyperplane Pruning

As discussed in Section 2.4 and demonstrated empirically in Section 4, different hyperplane scaling strategies make the microbagging estimators sensitive to different types of noise in the data. When $m > 0$, the classifier is sensitive to gross outliers whereas when $m < 0$, the classifier is sensitive to inliers that lie near many points from the opposing class. In both cases, the problem lies in the fact that a small number of data points garner the bulk of the distribution of $\boldsymbol{\alpha}$ because the interclass pairwise distances for those points are themselves outliers. One solution to this problem is to down-weigh or remove pairwise hyperplanes corresponding to points that are unusually close or unusually distant—we call this proposal *trimmed-distance* microbagging. We conjecture that this trimming of the pairwise distance matrix can further stabilize the microbagging approach and gives an intuitive mechanism for down-weighing outliers and inliers; we will explore this in future work.

6.2. Generalizations of Distance-Weighted Voting

The distance-weighted voting estimator that arises from microbagging immediately generalizes the linear kernelized estimator from Section 1.2 in two ways.

1. **Non-kernelized Distances:** Because both the learning and prediction phases can be expressed solely in terms of the pairwise distances between data points without direct kernel evaluations, we can use any distance function defined on the space $\mathcal{X}$.

2. **Multi-class Learning:** Both the training and prediction phase of the distance-weighted voting estimator naturally generalize to a multi-class setting. The weights expressed in Eq. (4) can be generalized as a sum over the scalings $\sigma_{i,j}$ for every i and j with $y_i \neq y_j$. For prediction, based on the minimal weighted average interpretation of microbagging, we simply select the class according to Eq. (8) with $c \in \mathcal{Y}$.

Acknowledgements

The authors wish to acknowledge the Alexander von Humboldt Foundation and the Heisenberg Fellowship of the Deutsche Forschungsgemeinschaft (DFG) for providing financial support to carry out this research. This work was also partly supported by a grant awarded to B. Biggio by Regione Autonoma della Sardegna, PO Sardegna FSE 2007-2013, L.R. 7/2007 “Promotion of the scientific research and technological innovation in Sardinia”. The opinions expressed in this paper are solely those of the authors and do not necessarily reflect the opinions of any sponsor.

References

J. Andrew Bagnell. Robust supervised learning. In *AAAI*, pages 714–719, 2005.

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

Sahibsingh A. Dudani. The distance-weighted k-nearest neighbor rule. In *IEEE Transactions on Systems, Man, and Cybernetics*, volume SMC-6, pages 325–327, 1976.

Andrew Frank and Arthur Asuncion. UCI machine learning repository, 2010. URL http://archive.ics.uci.edu/ml.

Jerome H. Friedman and Peter Hall. On bagging and nonlinear estimation. Technical report, Stanford University, 2000.

Amir Globerson and Sam Roweis. Nightmare at test time: Robust learning by feature deletion. In *ICML*, pages 353–360, 2006.

Yves Grandvalet. Bagging equalizes influence. *Machine Learning*, 55:251–270, June 2004.

Frank R. Hampel, Elvezio M. Ronchetti, Peter J. Rousseeuw, and Werner A. Stahel. *Robust Statistics: The Approach Based on Influence Functions*. John Wiley and Sons, 1986.

Peter J. Huber. *Robust Statistics*. John Wiley & Sons, 1981.

Hyun Chul Kim, Shaoning Pang, Hong-Mo Je, Daijin Kim, and Sung Yang Bang. Support vector machine ensemble with bagging. In *Proceedings of the First International Workshop on Pattern Recognition with Support Vector Machines*, pages 397–407, 2002.

Tom Mitchell. *Machine Learning*. McGraw Hill, 1997. ISBN 0-07-042807-7.

Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 5:197–227, 1990.

Bernhard Scholkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001. ISBN 0262194759.

Amnon Shashua and Anat Levin. Ranking with large margin principle: Two approaches. In *NIPS*, pages 937–944, 2002.

Donald Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 23rd ACM National Conference*, pages 517–524, 1968.

David M. J. Tax and Robert P. W. Duin. Support vector data description. *Machine Learning*, 54(1):45–66, 2004.

Vladimir Vapnik. *Statistical learning theory*. Wiley, 1998. ISBN 978-0-471-03003-4.

Ron Wehrens, Hein Putter, and Lutgarde M.C. Buydens. The bootstrap: a tutorial. *Chemometrics and Intelligent Laboratory Systems*, 54(1):35–52, 2000.

Linli Xu, James Neufeld, Bryce Larson, and Dale Schuurmans. Maximum margin clustering. In *NIPS*, pages 1537–1544, 2004.

Linli Xu, Koby Crammer, and Dale Schuurmans. Robust support vector machine training via convex outlier ablation. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, pages 536–542, 2006.

Wangmeng Zuo, Kuanquan Wang, Hongzhi Zhang, and David Zhang. Kernel difference-weighted k-nearest neighbors classification. In *Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence*, volume 4682 of *LNCS*, pages 861–870. 2007.