Class-Boundary Alignment for Imbalanced Dataset Learning

Gang Wu GWU@ENGINEERING.UCSB.EDU
Edward Y. Chang ECHANG@ECE.UCSB.EDU
Department of Electrical & Computer Engineering, University of California, Santa Barbara

Abstract

In this paper, we propose the class-boundary-alignment algorithm to augment SVMs to deal with imbalanced training-data problems posed by many emerging applications (e.g., image retrieval, video surveillance, and gene profiling). Through a simple example, we first show that SVMs can be ineffective in determining the class boundary when the training instances of the target class are heavily outnumbered by the non-target training instances. To remedy this problem, we propose to adjust the class boundary either by transforming the kernel function when the training data can be represented in a vector space, or by modifying the kernel matrix when the data do not have a vector-space representation (e.g., sequence data). Through theoretical analysis and empirical study, we show that the class-boundary-alignment algorithm works effectively with images (data that have a vector-space representation) and video sequences (data that do not have a vector-space representation).

1. Introduction

Support Vector Machines (SVMs) are a core machine learning technology. They have strong theoretical foundations and excellent empirical successes in many pattern recognition applications such as handwriting recognition (Vapnik, 1995), image retrieval (Tong & Chang, 2001), and text classification (Joachims, 1998). However, for many emerging applications, such as image understanding, security surveillance, and gene profiling, where the training instances of the target class are significantly outnumbered by the other training instances, the class-boundary learned by SVMs can be severely skewed towards the target class. As a result, the false-negative rate can be excessively high in identifying important target objects (e.g., a suspicious event or a gene disease), and hence can render the classifier ineffective. (We will present and discuss the details of this problem in Section 3.)

Several attempts have been made to improve class-prediction accuracy of SVMs (Amari & Wu, 1999; Veropoulos et al., 1999; Lin et al., 2002; Crammer et al., 2003; Ong et al., 2003). Given the class prediction function of SVMs,

$$sgn\left(f(\mathbf{x}) = \sum_{i=1}^{n} y_i \alpha_i K(\mathbf{x}, \mathbf{x}_i) + b\right), \quad (1)$$

three parameters can affect the decision outcome: b, α_i, and K. Our empirical study shows that the only effective method for improving SVMs, however, is through adaptively modifying K based on the training data distribution. As indicated by (Amari & Wu, 1999), by conformally spreading the area around the class-boundary outward on the Riemannian manifold S where all mapped data are located in feature space F, we can adapt K locally to data distribution to improve class-prediction accuracy. In this paper, we propose the class-boundary-alignment algorithm, which improves upon Amari and Wu's method for tackling the imbalanced training-data problem in three respects.

1. We conduct the transformation based on the spatial distribution of the support vectors in feature space F, instead of in input space I (Wu & Amari, 2002). Using feature-space distance to conduct conformation transformation takes advantage of the new information learned by SVMs in every iteration, whereas input-space distance remains unchanged.

2. We adaptively control the transformation based on the skew of the class-boundary. This transformation gives the neighborhood of minority support vectors a higher spatial resolution, and hence achieves better separation between the classes.

3. We show that in cases where the input space may not physically exist (e.g., sequence data may not have a vector space representation), the class-boundary-alignment algorithm can be applied directly to adjust the pair-wise object-distance in the *kernel matrix*[1] $\mathbf{K}$.

Our experimental results on both UCI and real-world image/video datasets show the class-boundary-alignment algorithm to be effective in correcting the skewed boundary.

[1] Given a kernel function K and a set of instances $X_{train} = \{\mathbf{x}_i, y_i\}_{i=1}^n$, the kernel matrix (Gram matrix) is the matrix of all possible inner-products of pairs from X_{train}, $\mathbf{K} = (k_{ij}) = K(\mathbf{x}_i, \mathbf{x}_j)$.

Workshop on Learning from Imbalanced Datasets II, ICML, Washington DC, 2003.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 explains the problem of imbalanced training data using a 2-D checkerboard example. In Section 4, we describe the class-boundary-alignment algorithm and its competing methods for addressing the imbalanced training-data problem. Section 5 presents the setup and the results of our empirical studies. We offer our concluding remarks in Section 6.

2. Related Work

Approaches for addressing the imbalanced training-data problem can be categorized into two main divisions: the data processing approach and the algorithmic approach. The data processing approach can be further divided into two methods: under-sample the majority class, and over-sample the minority class. The one-sided selection proposed by Kubat (Kubat & Matwin, 1997) is a representative under-sampling approach which removes noisy, borderline, and redundant majority training instances. However, these steps typically can remove only a small fraction of the majority instances, so they might not be very helpful in a scenario with a majority-to-minority ratio of more than $100:1$ (which is becoming common in many emerging applications). Multi-classifier training (Chan & Stolfo, 1998) and Bagging (Breiman, 1996) are two other under-sampling methods. These methods do not deal with noisy and borderline data directly, but use a large ensemble of sub-classifiers to reduce prediction variance.

Over-sampling (Chawla et al., 2000; Weiss & Provost, 2001) is the opposite of the under-sampling approach. It duplicates or interpolates minority instances in the hope of reducing the imbalance. The over-sampling approach can be considered as a "phantom-transduction" method. It assumes the neighborhood of a positive instance to be still positive, and the instances between two positive instances positive. Assumptions like these, however, can be data-dependent.

The algorithmic approach, which is orthogonal to the data-processing approach, is the focus of this paper. Nugroho (Nugroho et al., 2002) suggests combining a competitive learning network and a multilayer perceptron as a solution for the class imbalance problem. Kubat et al. (Kubat & Matwin, 1997; Drummond & Holte, 2000; Elkan, 2001; Ling & Li., 1998) modify the decision-tree generator to improve its learning performance on imbalanced datasets. For SVMs, few attempts (Karakoulas & Taylor, 1999; Lin et al., 2002; Veropoulos et al., 1999) have dealt with the imbalanced training-data problem. Veropoulos et al. (Lin et al., 2002; Veropoulos et al., 1999) use different penalty constants for different classes of data. We will explain in Section 4.2 why this method can be ineffective. Amari and Wu (Amari & Wu, 1999) propose using conformal transformation to change the spatial resolution around the class boundary. Their method does not deal with imbalanced datasets. It works only for data that have a vector-space representation since it has to calculate the input-space Euclidean distance. Our proposed class-boundary-alignment algorithm can work with both vector and non-vector data.

Recently, kernel target alignment (Cristianini et al., 2002) was proposed to adjust the kernel matrix to fit the training data. Subsequently, several novel methods based on the kernel-alignment idea have been proposed for clustering, kernel selection, and kernel-matrix modification (e.g., (Crammer et al., 2003; Ong et al., 2003)). Kandola et al. (Kandola & Shawe-Taylor, 2003) propose an extension of kernel target alignment with a simple transformation of the "ideal" target kernel, to adapt the kernel in the imbalanced training-data problem. Compared to (Kandola & Shawe-Taylor, 2003), our method deals with just the class-boundary data, not the entire training dataset. Furthermore, the solution we introduce here learns a discriminant kernel function by modifying a prior kernel function/matrix in the supervised learning setting, instead of learning a kernel matrix in the semi-supervised setting as in (Kandola & Shawe-Taylor, 2003). An interesting future work is to compare the difference and effectiveness of these two approaches.

3. Boundary Bias and SVMs

In this section, we use a checkerboard example to illustrate the class imbalance problem that SVMs face. We also use the example to explain the causes of the problem.

3.1. Checkerboard Example

A subtle but severe problem that an SVM classifier faces is the skewed class boundary caused by imbalanced training data. To illustrate this skew problem graphically, Figure 1 shows a 2D checkerboard example. The checkerboard divides a 200×200 square into four quadrants. The top-left and bottom-right quadrants are occupied by negative (majority) instances, but the top-right and bottom-left quadrants contain only positive (minority) instances. The lines between the classes are the "ideal" boundary that separates the two classes. In the rest of the paper, we will use *positive* when referring to minority instances, and *negative* when referring to majority instances.

Figure 2 exhibits the boundary distortion between the two left quadrants in the checkerboard under two different negative/positive training-data ratios, where a black dot with a circle represents a support vector, and its radius represents the weight value α_i of the support vector. The bigger the circle, the larger the α_i. The class boundaries are constructed by searching an $\mathbf{x}$ around the "ideal" boundary with $f(\mathbf{x}) = 0$ in Eq. 1. Figure 2(a) shows the SVM

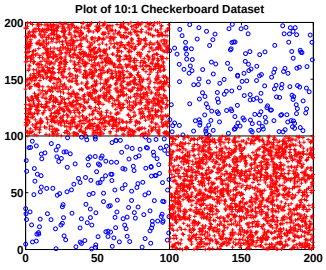
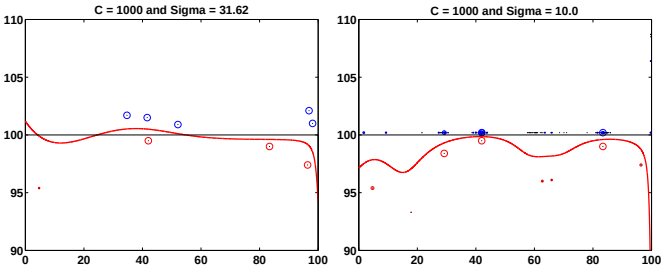


Figure 1. Checkerboard Experiment.

class boundary when the ratio of the number of negative instances (in the quadrant above) to the number of positive instances (in the quadrant below) is $10:1$. Figure 2(b) shows the boundary when the ratio increases to $10,000:1$. The boundary in Figure 2(b) is much more skewed towards the positive quadrant than the boundary in Figure 2(a), and hence causes a higher incidence of false negatives.



(a) 10:1 (b) 10,000:1

Figure 2. Boundaries of Different Ratios.

3.2. Causes of Skewed Boundary

To examine the causes, we present soft-margin SVMs (Vapnik, 1995) in the binary classification setting to set up a discussion context. Given a kernel function K and a set of labeled instances $X_{train} = \{\mathbf{x}_i, y_i\}_{i=1}^n$, SVMs find the optimal α_i for $\mathbf{x}_i$ to make the class prediction for $\mathbf{x}$ (a test instance) by using the following equation:

$$sgn(f(\mathbf{x}) = \sum_{i=1}^{n} y_i \alpha_i K(\mathbf{x}, \mathbf{x}_i) + b). \quad (2)$$

To solve α_i's, the soft-margin SVMs maximize the primal Lagrangian

$$L_p = \frac{\|\mathbf{w}\|^2}{2} + C\sum_{i}^{n} \xi_i^k - \sum_{i=1}^{n} \alpha_i[y_i(w \cdot \mathbf{x}_i + b) - 1 + \xi_i] - \sum_{i=1}^{n} \mu_i \xi_i,$$

where $k = 1$ or 2, $\alpha_i \geq 0$, and $\mu_i \geq 0$. The penalty constant C represents the trade-off between the empirical error and the margin. According to the KKT conditions (Vapnik, 1995), the value of α_i satisfies

$$0 \leq \alpha_i \leq C \quad and \quad \sum_{i=1}^{n} \alpha_i y_i = 0, \quad (3)$$

To understand the causes of the boundary-skew phenomenon, Figure 3 plots the negative to positive support-vector ratios. The x-axis of the figure depicts the ratios of negative to positive training instances, and the y-axis the ratios of negative to positive support vectors. The y-axis shows that the negative to positive support-vector ratio grows as the imbalance between the classes grows. Using the SVM properties and Figure 3, we observe two potential causes for the boundary skew.

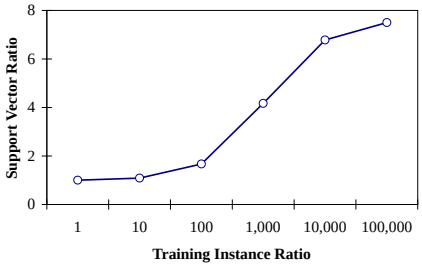


Figure 3. Support Vectors Ratios

1. The imbalanced training-data ratio. At the minority side of the boundary, the positive training data may not always reside as close to the "ideal boundary" as the negative training data do. This can be explained by a simple example. Suppose we randomly draw n numbers based on a uniform distribution between 1 and 100. The larger the value of n, the higher the chance that we draw a number close to 100, though the expected mean of the draws is invariant of the values of n. Thus, the low presence of the positive training instances makes them appear farther from the "ideal boundary" than the negative training instances.

2. The imbalanced support-vector ratio. When inspecting the boundary data in Figure 2, we find the α_i's values of the minority class (the positive class) tend to be much larger than those of the majority class, while the number of positive support vectors is substantially smaller. This phenomenon agrees with the constraint presented in Equation 3. As a consequence, the nearest neighborhood of a test point, especially when it is near the boundary, is likely to be dominated by negative support vectors, and hence the decision function (Eq. 2) is more likely to classify a boundary point negative.

4. Strategies for the Imbalanced Classification

In this section, we present three algorithmic approaches for adjusting the skewed boundary. We first present two competing approaches, then our class-boundary-alignment algorithm.

4.1. Boundary Movement (BM)

A naive method is to change b in the SVM decision function (Equation 2). One can set b so that the class-decision boundary can be adjusted as follows:

$$f(x) = sgn(\sum_{i=1}^{n} y_i \alpha_i K(\mathbf{x}, \mathbf{x}_i) + b + \triangle b), \tag{4}$$

where $\triangle b$ represents how much the boundary is moved. The boundary-movement method is a post-processing method. Intuitively, we can see that changing b trades a higher false positive count for a lower false negative one. We use *boundary movement* as the yardstick to measure how the other methods perform.

4.2. Biased Penalties (BP)

Veropoulos (Veropoulos et al., 1999) suggests using different penalty factors C^+ and C^- for positive and negative classes, reflecting their importance during training. Therefore, the L_p formulation has two loss functions for two types of errors.

$$\begin{aligned} L_p \quad = \quad & \frac{\|\mathbf{w}\|^2}{2} + C^+ \sum_{\{i|y_i=+1\}}^{n_+} \xi_i^k + C^- \sum_{\{j|y_j=-1\}}^{n_-} \xi_j^k \\ & - \sum_{i=1}^{n} \alpha_i [y_i(w \cdot \mathbf{x}_i + b) - 1 + \xi_i] - \sum_{i=1}^{p} \mu_i \xi_i. \end{aligned}$$

If the SVM algorithm uses an L_1 norm (k=1) for the losses, its dual formulation gives the same Lagrangian as in the original soft-margin SVMs, but with different constraints on α_i as follows:

$$0 \le \alpha_i \le C^+, \; if \; y_i \; = +1, \quad and \tag{5}$$

$$0 \le \alpha_i \le C^-, \; if \; y_i \; = -1. \tag{6}$$

It turns out that this biased-penalty method does not help SVMs as much as expected. From the KKT conditions (Eq. 3), we can see that C imposes only an upper bound on α_i, not a lower bound. Increasing C does not necessarily affect α_i. Moreover, the constraint in Equation 3 imposes equal total influence from the positive and negative support vectors. The increases in some α_i at the positive side will inadvertently increase some α_i at the negative side to satisfy the constraint. These constraints can make the increase of C^+ on minority instances ineffective.

4.3. Class-Boundary Alignment

Here, we present our Class-Boundary-Alignment algorithm in two parts. In the first part, the algorithm transforms the kernel function K when the training data can be represented in a vector space. We use ACT to denote such an adaptive conformal transformation algorithm. In the second part, the algorithm modifies the kernel matrix $\mathbf{K}$ when the data do not have a vector-space representation. We use KBA to denote this kernel matrix modification method.

4.3.1. Conformally Transforming K (ACT)

Kernel-based methods, such as SVMs, introduce a mapping function Φ which embeds the the input space I into a high-dimensional feature space F as a curved Remannian manifold S where the mapped data reside (Amari & Wu, 1999; Burges, 1999). A Riemannian metric $g_{ij}(\mathbf{x})$ is then defined for S, which is associated with the kernel function $K(\mathbf{x}, \mathbf{x}')$ by

$$g_{ij}(\mathbf{x}) = \left(\frac{\partial^2 K(\mathbf{x}, \mathbf{x}')}{\partial x_i \partial x'_j} \right)_{\mathbf{x}'=\mathbf{x}}. \tag{7}$$

The metric g_{ij} shows how a local area around $\mathbf{x}$ in I is magnified in F under the mapping of Φ. The idea of conformal transformation in SVMs is to enlarge the margin by increasing the magnification factor $g_{ij}(\mathbf{x})$ around the boundary (represented by support vectors) and to decrease it around the other points. This could be implemented by a conformal transformation of the related kernel $K(\mathbf{x}, \mathbf{x}')$ according to Eq. 7, so that the spatial relationship between the data would not be affected too much (Amari & Wu, 1999). Such a conformal transformation can be depicted as

$$\tilde{K}(\mathbf{x}, \mathbf{x}') = D(\mathbf{x}) D(\mathbf{x}') K(\mathbf{x}, \mathbf{x}'). \tag{8}$$

In the above equation, $D(\mathbf{x})$ is a properly defined positive conformal function. $D(\mathbf{x})$ should be chosen in a way such that the new Remannian metric $\tilde{g}_{ij}(\mathbf{x})$, associated with the new kernel function $\tilde{K}(\mathbf{x}, \mathbf{x}')$, has larger values near the decision boundary. Furthermore, to deal with the skew of the class boundary caused by imbalanced classes, we magnify $\tilde{g}_{ij}(\mathbf{x})$ more in the boundary area close to the minority class. In (Wu & Chang, 2003), we demonstrate that an RBF distance function such as

$$D(\mathbf{x}) = \sum_{k \in \mathbf{SV}} \exp\left(-\frac{|\mathbf{x} - \mathbf{x}_k|}{\tau_k^2} \right) \tag{9}$$

is a good choice for $D(\mathbf{x})$.

In Eq. 9, we can see that if τ_k^2's are fixed for all support vectors $\mathbf{x}_k$'s, $D(\mathbf{x})$ would be very dependent on the density of support vectors in the neighborhood of $\Phi(\mathbf{x})$. To alleviate this problem, we adaptively tune τ_k^2 according to the spatial distribution of support vectors in F (Wu & Chang, 2003). This goal can be achieved by the following equations:

$$\tau_k^2 = \mathrm{AVG}_{i \in \{\|\Phi(\mathbf{x}_i) - \Phi(\mathbf{x}_k)\|^2 < M, \; y_i \neq y_k\}} \left(\|\Phi(\mathbf{x}_i) - \Phi(\mathbf{x}_k)\|^2 \right). \tag{10}$$

In this equation, the average on the right-hand side comprises all support vectors in $\Phi(\mathbf{x}_k)$'s neighborhood within the radius of M but having a different class label. Here, M is the average distance of the nearest and the farthest support vectors from $\Phi(\mathbf{x}_k)$. Setting τ_k^2 in this way takes into consideration the spatial distribution of the support vectors in F. Although the mapping Φ is unknown, we can play the kernel trick to calculate the distance in F:

$$\|\Phi(\mathbf{x}_i) - \Phi(\mathbf{x}_k)\|^2 = K(\mathbf{x}_i, \mathbf{x}_i) + K(\mathbf{x}_k, \mathbf{x}_k) - 2 * K(\mathbf{x}_i, \mathbf{x}_k). \quad (11)$$

Substituting Eq. 11 into Eq. 10, we can then calculate the τ_k^2 for each support vector, which can adaptively reflect the spatial distribution of the support vector in F, not in I.

When the training dataset is very imbalanced, the class boundary would be skewed towards the minority class in the input space I. We hope that the new metric $\tilde{g}_{ij}(\mathbf{x})$ would further magnify the area far away from a minority support vector $\mathbf{x}_i$ so that the boundary imbalance could be alleviated. Our algorithm thus assigns a multiplier for the τ_k^2 in Eq. 10 to reflect the boundary skew in $D(\mathbf{x})$. We tune $\tilde{\tau}_k^2$ as $\eta_p \tau_k^2$ if $\mathbf{x}_k$ is a minority support vector; otherwise, we tune it as $\eta_n \tau_k^2$. Examining Eq. 9, we can see that $D(\mathbf{x})$ is a monotonously increasing function of τ_k^2. To increase the metric $\tilde{g}_{ij}(\mathbf{x})$ in an area which is not very close to the support vector $\mathbf{x}_k$, it would be better to choose a larger η_p for the τ_k^2 of a minority support vector. For a majority support vector, we can choose a smaller η_n, so as to minimize influence on the class-boundary. We empirically demonstrate that η_p and η_n are proportional to the skew of support vectors, or η_p as $O(\frac{|\mathbf{SV}^-|}{|\mathbf{SV}^+|})$, and η_n as $O(\frac{|\mathbf{SV}^+|}{|\mathbf{SV}^-|})$, where $|\mathbf{SV}^+|$ and $|\mathbf{SV}^-|$ denote the number of minority and majority support vectors, respectively. (Please see (Wu & Chang, 2003) for the details of ACT.)

4.3.2. MODIFYING $\mathbf{K}$ (KBA)

For data that do not have a vector-space representation (e.g., sequence data), it may not be applicable to conformally transform K. In this situation, KBA modifies kernel matrix $\mathbf{K}$ based on training-data distribution. Kernel matrix $\mathbf{K}$ contains the pairwise similarity information between all pairs of instances in a training dataset. Hence, in kernel-based methods, all we need is a kernel matrix to learn the classifier, even the data do not reside in a vector space.

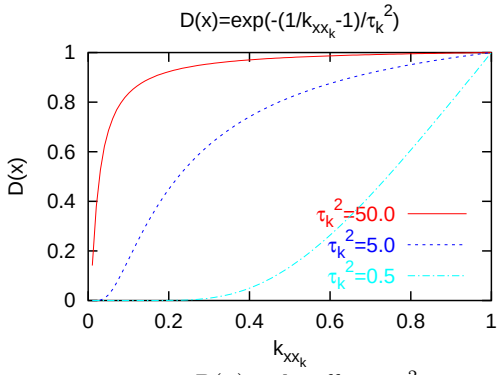


Figure 4. $D(\mathbf{x})$ with Different τ_k^2.

Now, since a training instance $\mathbf{x}$ might not be a vector, in this paper, we introduce a term, *support instance*, to denote $\mathbf{x}$ if its embedded point via $\mathbf{K}$ is a support vector[2]. In this situation, we cannot choose $D(\mathbf{x})$ as in Eq. 9. (It is impossible to calculate the Euclidean distance $|\mathbf{x} - \mathbf{x}_i|$ for non-vector data.) In Section 4.3.1, we show that $D(\mathbf{x})$ should be chosen in such a way that the spatial resolution of the manifold S would be magnified around the support instances. In other words, if $\mathbf{x}$ is close to a support instance $\mathbf{x}_k$ in F (or in its neighborhood), we hope that $D(\mathbf{x})$ would be larger so as to achieve a greater magnification. In KBA, we use the pairwise-similarity $k_{\mathbf{x}\mathbf{x}_k}$ to measure the distance of $\mathbf{x}$ from $\mathbf{x}_k$ in F. Therefore, we choose $D(\mathbf{x})$ as

$$D(\mathbf{x}) = \sum_{k \in \mathbf{SI}} \exp\left(-\frac{\frac{1}{k_{\mathbf{x}\mathbf{x}_k}} - 1}{\tau_k^2} \right), \quad (12)$$

where **SI** denotes the support-instance set, and τ_k^2 controls the magnitude of $D(\mathbf{x})$.

Figure 4 illustrates a $D(\mathbf{x})$ for a given support instance $\mathbf{x}_k$, where we can see that $D(\mathbf{x})$ (y-axis) becomes larger when an instance $\mathbf{x}$ is more similar to $\mathbf{x}_k$ (a larger $k_{\mathbf{x}\mathbf{x}_k}$ in the x-axis), so that there would be more magnification on the spatial resolution around the support vector embedded by $\mathbf{x}_k$ in F. Notice in the figure that $D(\mathbf{x})$ can be shaped very differently with different τ_k^2. We thus need to adaptively choose τ_k^2 as

$$\tau_k^2 = \mathrm{AVG}_{i \in \{Dist^2(\mathbf{x}_i, \mathbf{x}_k) < M,\ y_i \neq y_k\}} \left(Dist^2(\mathbf{x}_i, \mathbf{x}_k) \right), \quad (13)$$

where the distance $Dist^2(\mathbf{x}_i, \mathbf{x}_k)$ between two support instances $\mathbf{x}_i$ and $\mathbf{x}_k$ is calculated via the kernel trick as

$$Dist^2(\mathbf{x}_i, \mathbf{x}_k) = k_{\mathbf{x}_i\mathbf{x}_i} + k_{\mathbf{x}_k\mathbf{x}_k} - 2 * k_{\mathbf{x}_i\mathbf{x}_k}. \quad (14)$$

The neighborhood range M in Eq. 13 is chosen as the average of the minimal distance $Dist_{min}^2$ and the maximal distance $Dist_{max}^2$ from $\mathbf{x}_k$. In addition, τ_k^2 is scaled in the same way as we did in Section 4.3.1 for dealing with the imbalanced training-data problem.

Figure 5 summarizes the KBA algorithm. We apply KBA on the training dataset X_{train} until the testing accuracy on X_{test} cannot be further improved. In each iteration, KBA adaptively calculates τ_k^2 for each support instance (step 10), based on the distribution of support instances in feature space F. KBA scales the τ_k^2 according to the negative-to-positive support-instance ratio (steps 11 to 14). Finally, KBA updates the kernel matrix and performs retraining on X_{train} (steps 15 to 18).

5. Experimental Results

Our empirical study examined the effect of the class-boundary-alignment algorithm (i.e., ACT and KBA) in two aspects.

1. *Vector-space evaluation*. We compared ACT with other algorithms for imbalanced-data learning. We used

[2] In KBA algorithm, if $\mathbf{x}$ is a support instance, we call both $\mathbf{x}$ and its embedded support vector via $\mathbf{K}$ in F *support instance*.

Input:
X_{train}, X_{test}, $\mathbf{K}$;
θ; /* *stopping threshold* */
T; /* *maximum running iterations* */
Output:
$\mathcal{C}$; /* *output classifier* */
Variables:
$\mathbf{SI}$; /* *support-instance set* */
M; /* *neighborhood range* */
$\mathbf{s}$; /* *a support instance* */
$\mathbf{s}.\tau$; /* *parameter of* $\mathbf{s}$ */
$\mathbf{s}.y$; /* *class label of* $\mathbf{s}$ */
Function Calls:
SVMTrain(X_{train}, $\mathbf{K}$); /* *train classifier* $\mathcal{C}$ */
SVMClassify($X_{test}, \mathcal{C}$); /* *classify* X_{test} *by* $\mathcal{C}$ */
ExtractSI($\mathcal{C}$); /* *obtain* $\mathbf{SI}$ *from* $\mathcal{C}$ */
ComputeM($\mathbf{s}, \mathbf{SI}$); /* *compute M* */
Begin
1) $\mathcal{C} \leftarrow$ SVMTrain($X_{train}, \mathbf{K}$);
2) $\varepsilon_{old} \leftarrow \infty$;
3) $\varepsilon_{new} \leftarrow$ SVMClassify($X_{test}, \mathcal{C}$);
4) $t \leftarrow 0$;
5) **while** $((\varepsilon_{old} - \varepsilon_{new} > \theta) \&\& (t < T))$ {
6) $\mathbf{SI} \leftarrow$ExtractSI($\mathcal{C}$);
7) $\eta_p \leftarrow O(\frac{|\mathbf{SI}^-|}{|\mathbf{SI}^+|})$, $\eta_n \leftarrow O(\frac{|\mathbf{SI}^+|}{|\mathbf{SI}^-|})$;
8) **for** each $\mathbf{s} \in \mathbf{SI}${
9) $M \leftarrow$ComputeM($\mathbf{s}$, $\mathbf{SI}$);
10) $\mathbf{s}.\tau \leftarrow \sqrt{\text{AVG}_{i \in \{Dist^2(\mathbf{s}_i, \mathbf{s}) < M,\ \mathbf{s}_i.y \neq \mathbf{s}.y\}} \left(Dist^2(\mathbf{s}_i, \mathbf{s})\right)}$;
11) **if** $\mathbf{s} \in \mathbf{SI}^+$ **then** /* *a minority* */
12) $\mathbf{s}.\tau \leftarrow \sqrt{\eta_p} \times \mathbf{s}.\tau$;
13) **else** /* *a majority* */
14) $\mathbf{s}.\tau \leftarrow \sqrt{\eta_n} \times \mathbf{s}.\tau$;}
15) $D(\mathbf{x}) = \sum_{\mathbf{s} \in SI} \exp\left(-\frac{\frac{1}{k_{\mathbf{xs}}} - 1}{\mathbf{s}.\tau^2}\right)$
16) **for** each k_{ij} in $\mathbf{K}${
17) $k_{ij} \leftarrow D(\mathbf{x}_i) \times D(\mathbf{x}_j) \times k_{ij}$;}
18) $\mathcal{C} \leftarrow$ SVMTrain($X_{train}, \mathbf{K}$);
19) $\varepsilon_{old} \leftarrow \varepsilon_{new}$;
20) $\varepsilon_{new} \leftarrow$ SVMClassify($X_{test}, \mathcal{C}$);
21) $t \leftarrow t + 1$;}
22) **return** $\mathcal{C}$;
End

Figure 5. The KBA Algorithm.

six UCI datasets and an image dataset to conduct this evaluation. (We present the datasets shortly.)

2. *Non-vector-space evaluation*. We evaluated the effect of KBA on a set of video surveillance data, which are represented as spatio-temporal sequences and do not have a vector-space representation.

In our experiments, we employed Laplacian kernels of the form $\exp(-\gamma|\mathbf{x} - \mathbf{x}'|)$ as $K(\mathbf{x}, \mathbf{x}')$. Then we used the following procedure. The dataset was randomly split into training and test subsets generated in an optimal ratio, which was empirically chosen for each dataset. Hyper-parameters (C and γ) of $K(\mathbf{x}, \mathbf{x}')$ were obtained for each run using 7-fold cross-validation. All training, validation, and test subsets were sampled in a stratified manner that ensured each of them had the same negative/positive ratio (Kubat & Matwin, 1997). We repeated this procedure ten times, computed average class-predication accuracy, and compared the results. For ACT and KBA, we chose the stopping threshold θ as 0.001 and maximum running iteration T as 10.

Dataset	# Attrib	# Pos	# Neg	SVMs	SMOTE	ACT
seg1	19	30	180	**98.1**	**98.1**	**98.1**
g7	10	29	185	89.9	91.8	**93.7**
euth1	24	238	1762	92.8	92.4	**94.5**
car3	6	69	1659	99.0	99.0	**99.9**
yeast5	8	51	1433	59.1	69.9	**78.5**
ab19	8	32	4145	0.0	0.0	**51.9**

Table 1. UCI-Dataset Prediction Accuracy.

5.1. Vector-space Evaluation

For this evaluation, we used six UCI datasets and a 116-category image dataset. The six UCI datasets we experimented with are *abalone* (abalone19), *car* (car3), *segmentation* (seg1), *yeast* (yeast5), *glass* (g7), and *euthyroid* (euthy1). The number in the parentheses indicates the target class we chose. Table 1 shows the characteristics of these six datasets organized according to their negative-to-positive training-instance ratios. The top three datasets (seg1, g7, and euth1) are not-too-imbalanced. The middle two (car3 and yeast5) are mildly imbalanced. The bottom dataset (ab19) is the most imbalanced (the ratio is about $130:1$).

The image dataset contains 20K images in 116 categories collected from the Corel Image CDs[3]. Each image is represented by a vector of 144 dimensions including color, texture, and shape features (Tong & Chang, 2001). To perform class prediction, we employed the one-per-class (OPC) ensemble (Dietterich & Bakiri, 1995), which trains 116 classifiers, each of which predicts the class membership for one class. The class prediction on a testing instance is decided by voting among the 116 classifiers.

5.1.1. Results on UCI Benchmark Datasets

We first report the experimental results with the six UCI datasets in Table 1. In addition to conducting experiments with SVMs and ACT, we also implemented and tested one popular minority over-sampling strategy SMOTE (Chawla et al., 2000). We used the L_2-norm RBF function for $D(\mathbf{x})$.

[3]We exclude from our testbed categories that are not possible to classify automatically, such as "industry", "Rome", and "Boston". (E.g., the Boston category contains various subjects, e.g., architectures, landscapes, and people, of Boston.)

Category	Ratio	SVMs	BM	BP	ACT
Mountain	34 : 1	24.8	21.2	24.8	**33.3**
Snow	37 : 1	46.4	47.5	47.8	**54.6**
Desert	39 : 1	33.7	31.8	34.3	**39.1**
Dog	44 : 1	32.9	28.5	35.2	**41.5**
Woman	54 : 1	27.9	25.3	26.2	**35.3**
Church	66 : 1	**21.8**	19.4	**21.8**	20.0
Leaf	80 : 1	26.1	27.2	24.8	**32.6**
Lizard	101 : 1	13.9	11.8	15.1	**22.2**
Parrot	263 : 1	7.1	3.5	7.1	**14.3**
Horse	264 : 1	14.3	10.4	14.3	**28.6**
Leopard	283 : 1	7.7	5.6	7.7	**23.1**
Shark	1232 : 1	0.0	0.0	0.0	**16.6**

Table 2. Image-dataset Prediction Accuracy.

In each run, the training and test subsets were generated in the ratio $6 : 1$, which was empirically proven to be optimal. For SMOTE[4], the minority class was over-sampled at 200%, 400% and 1000% for each of three groups of UCI datasets in Table 1, respectively.

We report in Table 1 using the Kubat's g-means metric defined as $\sqrt{a^+ \cdot a^-}$, where a^+ and a^- are positive (the target class) and negative testing accuracy, respectively (Kubat & Matwin, 1997). The table shows that ACT achieves the highest accuracy in five of the six datasets (marked by bold font). When the data is very imbalanced (the last row of Table 1), ACT achieves 51.9% class-prediction accuracy, where SVMs and SMOTE fail completely.

5.1.2. Results on 20K Image Dataset

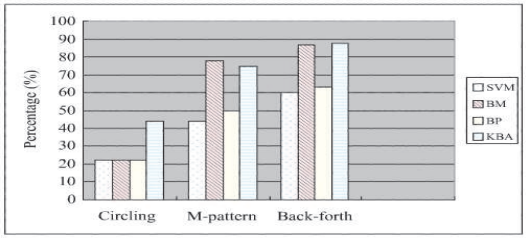
The image dataset is more imbalanced than the UCI datasets. We first set aside $4K$ images to be used as the test subset; the remaining $16K$ images were used for training and validation. We compared four schemes: SVMs, BM (boundary movement), BP (biased penalty), and ACT. Notice that in this experiment, we use the L_1-norm RBF function for $D(\mathbf{x})$, since the L_1-norm RBF works the best for the image dataset (Tong & Chang, 2001).

Table 2 presents the prediction accuracy for twelve representative categories out of 116 ones, sorted by their imbalance ratios. ACT improves the accuracy over SVMs by 7.6%, 4.9%, and 13.4% on the three subgroup datasets, respectively. ACT achieves the best prediction accuracy for eleven out of twelve categories among all schemes (marked by bold font). BM is inferior to SVMs for almost all categories. Finally, BP outperforms SVMs, but only slightly. (We have predicted BP's ineffectiveness, due to the KKT conditions, in Section 4.2.)

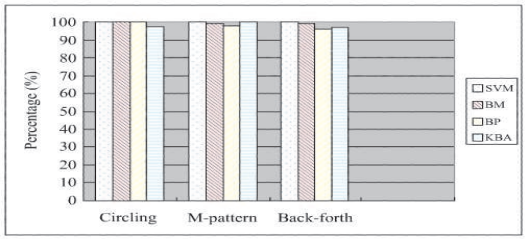
5.2. Non-vector-space Evaluation

For our multi-camera video-surveillance project (Wu et al., 2003), We recorded video at parking lot-20 on the UCSB campus. We collected trajectories depicting five motion patterns: *circling* (30 instances), *zigzag-pattern* or *EM-pattern* (22 instances), *back-and-forth* (40 instances), *go-straight* (200 instances), and *parking* ($3,161$ instances including additional synthetic data to simulate the skew effect). We divided these events into the benign and suspicious categories and aimed to detect suspicious events with high accuracy. The benign-event category consists of patterns *go-straight* and *parking*, and the suspicious-event category consists of the other three patterns.

[4]For the datasets in Table 1 from top to bottom, for SMOTE, the optimal γ was $0.002, 0.003, 0.085, 0.3, 0.5$, and 0.084 respectively. For SVMs and ACT, the optimal γ was $0.004, 0.003, 0.08, 0.3, 0.5$, and 0.086 respectively. All optimal C's were $1,000$.



(a) Sensitivity



(b) Specifi city

Figure 6. Boundaries of Different Ratios.

For each experiment, we chose 60% of the data as the training set, and the remaining 40% as our testing data. We employed a sequence-alignment kernel to compare similarity between two trajectories (see (Wu et al., 2003) for details). Figure 6(a) reports the sensitivities of using SVMs and three other improvement methods. All three methods, BM, BP, and KBA, improve sensitivity. Among the three, KBA achieves the largest magnitude of improvement over SVMs, around 30 percentile points. Figure 6(b) shows that all methods maintain high specificity. Notice that BM method performs well for detecting *M-pattern* and *back-forth*; however, it does not do well consistently over all patterns. The performance of the BM method can be highly dependent on the data distribution. BP does not work effectively, which bears out our prediction in Section 4.2.

6. Conclusion

We have proposed the class-boundary-alignment algorithm for tackling the imbalanced training-data challenge. Through theoretical justifications and empirical studies, we show this method to be effective. We believe that class-boundary alignment is attractive, not only because of its accuracy, but also because it can be applied for both vector-

data and sequence-data (e.g., DNA sequences and spatio-temporal patterns) learning through modifying the kernel matrix directly.

References

Amari, S., & Wu, S. (1999). Improving support vector machine classifiers by modifying kernel functions. *Neural Networks, 12*, 783–789.

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

Burges, C. (1999). Geometry and invariance in kernel based methods. in adv. in kernel methods: Support vector learning. 89–116.

Chan, P., & Stolfo, S. (1998). Learning with non-uniform class and cost distributions: Effects and a distributed multi-classifier approach. *Workshop Notes KDD-98 Workshop on Distributed Data Mining*, 1–9.

Chawla, N., Bowyer, K., Hall, L., & Kegelmeyer, W. P. (2000). Smote:synthetic minority over-sampling technique. *International Conference on Knowledge Based Computer Systems*.

Crammer, K., Keshet, J., & Singer, Y. (2003). Kernel design using boosting. In *Advances in Neural Information Processing Systems*.

Cristianini, N., Kandola, J., Elisseeff, A., & Shawe-Taylor, J. (2002). On kernel target alignment. *Journal Machine Learning Research, 1*.

Dietterich, T., & Bakiri, G. (1995). Solving multiclass learning problems via error-correcting output codes. *Journal of Artifi cal Intelligence Research, 2*, 263–286.

Drummond, C., & Holte, R. (2000). Exploiting the cost (in)sensitivity of decision tree splitting criteria. *In Proceedings of the Seventeenth International Conference on Machine Learning*, 239–246.

Elkan, C. (2001). The foundations of cost-sensitive learning. *In Proceedings of the Seventeenth International Joint Conference on Artifi cial Intelligence*.

Joachims, T. (1998). Text categorization with support vector machines: learning with many relevant features. *Proceedings of ECML-98, 10th European Conference on Machine Learning*, 137–142.

Kandola, J., & Shawe-Taylor, J. (2003). Refining kernels for regression and uneven classification problems. In *Proc. of the 9th International Workshop on Artifi cial Intelligence and Statistics*.

Karakoulas, G., & Taylor, J. S. (1999). Optimizing classifiers for imbalanced training sets. In *Advances in Neural Information Processing Systems*.

Kubat, M., & Matwin, S. (1997). Addressing the curse of imbalanced training sets: One-sided selection. *Proceddings of the Fourteenth International Conference on Machine Learning*, 179–186.

Lin, Y., Lee, Y., & Wahba, G. (2002). Support vector machines for classification in nonstandard situations. *Machine Learning, 46*, 191–202.

Ling, C., & Li., C. (1998). Data mining for direct marketing - specific problems and solutions. *Proceedings of Fourth International Conference on Knowledge Discovery and Data Mining*, 73–79.

Nugroho, A., Kuroyanagi, S., & Iwata, A. (2002). A solution for imbalanced training sets problem by combnet-ii and its application on fog forecasting. *IEICE Transaction on Information and Systems, E85-D*, 1165–1174.

Ong, C. S., Smola, A. J., & Williamson, R. C. (2003). Superkernels. In *Advances in Neural Information Processing Systems*.

Tong, S., & Chang, E. (2001). Support vector machine active learning for image retrieval. *Proceedings of ACM International Conference on Multimedia*, 107–118.

Vapnik, V. (1995). *The nature of statistical learning theory*. New York: Springer.

Veropoulos, K., Campbell, C., & Cristianini, N. (1999). Controlling the sensitivity of support vector machines. *Proceedings of the International Joint Conference on Artifi cial Intelligence*, 55–60.

Weiss, G. M., & Provost, F. (2001). The effect of class distribution on classifier learning: An empirical study. *Technical Report ML-TR-44, Department of Computer Science, Rutgers University*.

Wu, G., & Chang, E. (2003). Adaptive feature-space conformal transformation for imbalanced data learning. *To appear in Proc. of the 20th International Conference on Machine Learning*.

Wu, G., Wu, Y., Jiao, L., Wang, Y.-F., & Chang, E. (2003). Multi-camera spatio-temporal fusion and biased sequence-data learning for security surveillance. *Technical Report, ECE Department, University of California at Santa Barbara*.

Wu, S., & Amari, S. (2002). Conformal transformation of kernel functions: A data-dependent way to improve the performance of support vector machine classifiers. *Neural Processing Letter, 15*.