Provably Fast Algorithms for Anomaly Detection

Los Alamos
National Laboratory

Don Hush, Patrick Kelly, Clint Scovel and Ingo Steinwart
Modeling, Algorithms and Informatics Group, CCS-3
Los Alamos National Laboratory
{dhush,kelly,jcs,ingo}@lanl.gov

LANL Technical Report: LA–UR–05-4367

Report Date: June 8, 2005

Abstract

We describe a solution method for one of the most common anomaly detection formulations [23] that is proven to be computationally efficient, universally consistent, and to guarantee near optimal finite sample performance for a large class of (practical) distributions [23, 21]. We also describe an algorithm for this method that accepts the desired accuracy ϵ as an input and produces an approximate solution that is guaranteed to satisfy this accuracy in low order polynomial time. Experimental results are used to demonstrate the actual run times for a *typical* problem.

1 Introduction

In a recent paper we describe a new solution method for one of the most common anomaly detection formulations [23]. This method is unique in that it is proven to be computationally efficient, universally consistent, and to guarantee near optimal finite sample performance for a large class of (practical) distributions [23, 21]. Since this method solves a *density level detection* (DLD) problem using a *support vector machine* (SVM) approach (both described below) it is called the *density level detection support vector machine* (DLD–SVM). The DLD–SVM was recently compared with several popular methods [1] using real data from a cybersecurity problem and found to perform very well [23]. Indeed it gave the best overall performance and was far superior to some methods. In this paper we describe a provably fast algorithm for the DLD–SVM.

In practice most SVM algorithms produce *approximate* solutions and consequently they introduce a trade-off between computation and accuracy that is not well understood. The accuracy, as measured by the difference between the criterion value of the approximate solution and the optimal criterion value, is important for learning because it has a direct influence on the generalization error. The accuracy of the approximate solution produced by existing SVM algorithms is often unknown. In addition the computational requirements of existing SVM algorithms are largely unknown. However in this paper we describe a DLD–SVM algorithm that accepts the desired accuracy ϵ as an input and produces an approximate solution that is guaranteed to satisfy this accuracy in low order polynomial time. Our analysis reveals the effect of the accuracy on the run time, thereby allowing the user to make an informed decision regarding the trade–off between computation and accuracy. In addition this analysis provides a worst case bound on the number of iterations that is typically linear in the number of samples. We present experimental results which validate this linear relation, but also show that the actual number of iterations for a typical problem can be much smaller than the worst case bound.

2 Problem Formulation

Anomalies are often described as rare or unusual events. This notion can be represented mathematically by defining anomalies to be points with low probability density value. In particular the set of points with density value below a threshold ρ comprise the *anomalous set*, while the complement of this set is called the *normal set*. Our goal is to design a binary function (an anomaly detector) that assigns the value -1 to points in the anomalous set and $+1$ to points in the normal set.

To formalize these notions we first recall the basic concept of *density*. *Density* is a (local) valuation of the relative concentration of two measures. In particular, for two measures Q and μ on a space X where Q is absolutely continuous with respect to μ (i.e. every μ–negligible set is a Q–negligible set) the density h of Q with respect to μ is the Radon–Nikodym derivative

[1] These popular methods included schemes based on Parzen density estimates, Gaussian density estimates determined by maximum likelihood parameter estimates, Mixture of Gaussians density estimates determined by the EM algorithm, and the 1-CLASS SVM [19].

$h = dQ/d\mu$. In the anomaly detection problem Q is an (unknown) probability measure that describes the data and μ a (known) reference measure. For example when $X \subseteq \mathbb{R}^d$ the reference μ is usually taken to be the Lebesgue measure (i.e. the standard volume). In principle however the reference measure is chosen by the user in a way that establishes a definition of anomalies relevant to the application. Given a density level $\rho > 0$, the normal set $\{h > \rho\}$ is called the *ρ-level set.* The goal of the *density level detection* (DLD) problem is to find an estimate of the ρ-level set of h and therefore an estimate of the anomalous set (by taking the complement). To find this estimate we use information given to us by a training set $T = (x_1, \ldots, x_n) \in X^n$ that is i.i.d. drawn from Q. With the help of T a DLD algorithm constructs a function $\hat{f} : X \to \mathbb{R}$ for which the set $\{\hat{f} > 0\}$ is an estimate of the ρ-level set $\{h > \rho\}$. A standard performance measure that quantifies how well $\{\hat{f} > 0\}$ approximates the set $\{h > \rho\}$ is (see e.g. [1])

$$\mathcal{S}(f) \;:=\; \mu\Big(\{f > 0\}\Delta\{h > \rho\}\Big)\,,$$

where Δ denotes the symmetric difference. The goal of the DLD problem is to find $\hat{f}$ such that $\mathcal{S}(\hat{f})$ is close to zero.

Now let μ be a probability measure and define the risk

$$\mathcal{R}(f) \;:=\; \frac{1}{1+\rho}\; Q(f \leq 0) + \frac{\rho}{1+\rho}\; \mu(f > 0).$$

Steinwart et al. [23] show that any function that minimizes $\mathcal{R}$ also minimizes $\mathcal{S}$. Furthermore they prove a very tight relation between $\mathcal{R}$ and $\mathcal{S}$ for all functions f. This establishes $\mathcal{R}$ as a bona fide risk function for the DLD problem. Therefore $\mathcal{R}$ *is a legitimate performance measure for anomaly detection.* Consequently our goal of choosing $\hat{f}$ to (approximately) minimize $\mathcal{S}$ can be revised to choosing $\hat{f}$ to (approximately) minimize $\mathcal{R}$.

It turns out that $\mathcal{R}$ is also a performance measure for a supervised classification problem. Indeed let $Y := \{1, -1\}$ be the label set and let $x \in X$ and $y \in Y$ denote values of the random variables $\mathbf{x}$ and $\mathbf{y}$. The supervised classification problem is formed by identifying Q and μ with the conditional distributions $P_{\mathbf{x}|\mathbf{y}=1}$ and $P_{\mathbf{x}|\mathbf{y}=-1}$ respectively and defining the class marginals $P(\mathbf{y} = 1) := 1/(1+\rho)$ and $P(\mathbf{y} = -1) := \rho/(1+\rho)$. To form a data set for this classification problem we *collect* n_1 i.i.d. samples $(x_1, \ldots, x_{n_1})$ from Q and assign each of them the label $y = +1$, and we *synthesize* n_{-1} i.i.d. samples $(x_{n_1+1}, \ldots, x_{n_1+n_{-1}})$ from μ and assign each of them the label $y = -1$. This gives a training set $\mathcal{T} = ((x_1, y_1), ..., (x_n, y_n))$ of size $n = n_1 + n_{-1}$. The goal is to use $\mathcal{T}$ to choose a function $\hat{f}$ so that $\mathcal{R}(\hat{f})$ is as small as possible. The only difference between this problem and a standard classification problem is that the class marginal probabilities are *known.*

We now describe the DLD–SVM solution method. Let $k : X \times X \to \mathbb{R}$ be a kernel function, i.e. there exists a Hilbert space H and a map $\phi : X \to H$ such that $k(x_1, x_2) = \phi(x_1) \cdot \phi(x_2), \forall x_1, x_2 \in X$. SVM functions f take the form

$$f_{\psi,b}(x) = \psi \cdot \phi(x) + b.$$

The DLD–SVM determines the parameters $\hat{\psi}$ and $\hat{b}$ by (approximately) solving the *primal* QP problem

$$\begin{array}{ll} \min_{\psi,b,\xi} & \lambda\|\psi\|^2 + \sum_{i=1}^n u_i\xi_i \\ \text{s.t.} & y_i(\phi(x_i)\cdot\psi + b) \geq 1 - \xi_i \\ & \xi_i \geq 0, \quad i = 1, 2, ..., n \end{array} \tag{1}$$

where $\lambda > 0$ and

$$u_i = \begin{cases} \frac{1}{(1+\rho)n_1}, & y_i = 1 \\ \frac{\rho}{(1+\rho)n_{-1}}, & y_i = -1 \end{cases}.$$

Since this QP problem can be prohibitively large (e.g. the dimension of ψ may be infinite) and its dual QP problem is considerably smaller we employ a two–stage process where the first stage produces an approximate solution to the dual QP problem and the second stage maps this approximate dual solution to an approximate primal solution. The *canonical dual* QP problem is

$$\begin{array}{ll} \max_\alpha & -\frac{1}{2}\alpha\cdot Q\alpha + \alpha\cdot w + w_0 \\ \text{s.t.} & 1\cdot\alpha = c \\ & 0 \leq \alpha_i \leq u_i \quad i = 1, 2, ..., n \end{array} \tag{2}$$

where

$$Q_{ij} = k(x_i, x_j)/2\lambda, \quad c = l\cdot 1, \quad w = Ql + y, \quad w_0 = -l\cdot y - \frac{1}{2}l\cdot Ql. \tag{3}$$

and

$$l_i = \begin{cases} 0 & y_i = 1 \\ u_i & y_i = -1 \end{cases}. \tag{4}$$

We denote the canonical dual criterion by

$$R(\alpha) \ := \ -\frac{1}{2}\alpha\cdot Q\alpha + \alpha\cdot w + w_0.$$

We define the set of ϵ–optimal solutions to the canonical dual QP problem to be $\{\alpha \in \mathcal{A} : |R^* - R(\alpha)| \leq \epsilon\}$ where $\mathcal{A}$ is the set of feasible points and R^* is the optimal criterion value. We use a similar definition for the set of ϵ_p–optimal solutions to the primal QP problem.

Our approach is to compute an ϵ–optimal canonical dual solution $\hat{\alpha}$ and then map it to an ϵ_p–optimal primal solution $(\hat{\psi}, \hat{b}, \hat{\xi})$. Let $K \geq \max_i k(x_i, x_i)$. For a dual solution $\hat{\alpha}$ with accuracy $\epsilon = (2\sqrt{2K} + 8)^{-2}\lambda\epsilon_p^2$ the map

$$\hat{\psi} = \frac{1}{2\lambda}\sum_{i=1}^n (\hat{\alpha}_i - l_i)\phi(x_i)$$

$$\hat{b} \in \arg\min_b \sum_{i=1}^n u_i \max(0, 1 - y_i(\hat{\psi}\cdot\phi(x_i) + b)) \tag{5}$$

and

$$\hat{\xi}_i = \max\left(0, 1 - y_i(\hat{\psi} \cdot \phi(x_i) + \hat{b})\right),\ i = 1, .., n$$

has been shown to produce a primal solution with accuracy ϵ_p [20]. Thus if we let $\hat{\gamma}_i = \frac{\hat{\alpha}_i - l_i}{2\lambda}$ the corresponding SVM anomaly detector takes the form

$$f_{\hat{\psi},\hat{b}}(x) = \sum_{i=1}^{n} \hat{\gamma}_i k(x_i, x) + \hat{b}.$$

Pseudocode for the main routine which produces the values $\hat{\gamma}$ and $\hat{b}$ corresponding to an ϵ_p–optimal primal solution is shown in Procedure 1. This routine forms an instance of the canonical dual QP according to (3), sets the desired accuracy of the canonical dual solution $\epsilon = (2\sqrt{2K} + 8)^{-2}\lambda\epsilon_p^2$, uses the routine `Composite` to compute an ϵ–approximate canonical dual solution, determines the expansion coefficients $\hat{\gamma}$, and uses the routine `Offset` to compute the offset parameter $\hat{b}$ according to (5). A simple $O(n \log n)$ algorithm for the routine `Offset` is described in Hush et al. [6]. Our focus here is on efficient algorithms for the routine `Composite`.

Procedure 1 The main algorithm for the DLD–SVM.

`INPUTS:` A data set $\mathcal{T} = ((x_1, y_1), ..., (x_n, y_n))$, a density level ρ, a kernel function k, and parameter values λ and ϵ_p

`OUTPUTS:` Parameter values $\hat{\gamma}$ and $\hat{b}$

Form canonical dual parameters: $Q_{ij} = \frac{k(x_i,x_j)}{2\lambda}$, $l_i = \frac{1-y_i}{2}$, $w = Ql + y$, $c = l \cdot 1$, $\epsilon = \frac{\lambda\epsilon_p^2}{2\sqrt{2K}+8}$

and $u_i = \begin{cases} \frac{1}{(1+\rho)n_1}, & y_i = 1 \\ \frac{\rho}{(1+\rho)n_{-1}}, & y_i = -1 \end{cases}$

$\hat{\alpha} \leftarrow$ `Composite`(Q, w, c, u, ϵ)

Compute expansion coefficients: $\hat{\gamma}_i = (\hat{\alpha}_i - l_i)/2\lambda$

$\hat{b} \leftarrow$ `Offset`$(\hat{\gamma}, \mathcal{T})$

Return$(\hat{\gamma}, \hat{b})$

The routine `Composite` solves the canonical dual QP problem by solving a sequence of smaller QP problems where each of the smaller QP problems is obtained by fixing a subset of the variables and optimizing with respect to the remaining variables. A number of these so–called *decomposition* algorithms have been developed for SVMs [3, 4, 5, 7, 8, 10, 11, 14, 15, 16, 17, 18, 22]. The key to developing a successful decomposition algorithm is in the method used to determine the *working sets*, which are the subsets of variables to be optimized at each iteration. To guarantee stepwise improvement each working set must contain a *certifying pair* [7]. Stronger conditions are required to guarantee convergence [2, 3, 7, 9, 12, 13, 14] and even stronger conditions appear necessary to guarantee rates of convergence [7, 12]. Indeed, although numerous decomposition algorithms have been proposed few are known to possess polynomial run time bounds. However by restricting to working sets of size 2 and augmenting the working set selection algorithm introduced by Simon [22] we have constructed a decomposition algorithm called `Composite` whose worst case run time is a low order polynomial given by the following theorem. The proof of this theorem is obtained by Hush et al. [6] through a slight modification of the analysis of List and Simon [15].

Theorem 1. *Consider the DLD–SVM canonical dual QP problem in (2) with criterion function R. Let $K \geq \max_i k(x_i, x_i)$, $r = (n_1 + n_{-1} - 1)/n_1$, and assume that the number of synthetic samples n_{-1} is chosen large enough so that $\frac{1}{(1+\rho)n_1} \geq \frac{\rho}{(1+\rho)n_{-1}}$. Define*

$$\beta := \frac{2Kr}{\lambda(1+\rho)^2 n_1}.$$

Then the `Composite` *decomposition algorithm in [6] achieves $R^* - R(\alpha^m) \leq \epsilon$ after $m = \acute{m}$ iterations of the main loop where*

$$\acute{m} = \begin{cases} 2rn_1 \ln\left(\dfrac{1}{\epsilon}\right), & \epsilon \geq \beta \\ \\ 2rn_1\left(\dfrac{\beta}{\epsilon} - 1 + \ln\dfrac{1}{\beta}\right), & \epsilon < \beta \end{cases} \tag{6}$$

Furthermore the overall run time of this algorithm is $O\left(r^2 n_1^2 \log(1/\epsilon)\right)$ for $\epsilon \geq \beta$ and $O\left(\frac{Kr^3 n_1}{\lambda\epsilon(1+\rho)^2}\right)$ for $\epsilon < \beta$.

For typical parameter values these run time bounds are on the order of n_1^2. This result is significant for two reasons. First, if we applied the fastest known algorithm for the general convex QP problem the run time bound would be $O(n_1^3/\log n_1)$. Thus by developing an algorithm for a specific class of QP problems we have obtained a significant improvement. Second, non–asymptotic run time guarantees with this type of efficiency are extremely rare for anomaly detection algorithms, especially for algorithms which also guarantee near optimal performance for such a large class of practical distributions.

To achieve the run time guarantees described by this theorem the `Composite` algorithm must be terminated properly. The simplest stopping rule that guarantees an ϵ–optimal solution is to stop after $\acute{m}$ iterations. However for a typical problem instance the algorithm may reach the accuracy ϵ in far fewer iterations. For this reason Hush et al. [6] introduce a rule that computes an upper bound on $R^* - R(\alpha)$ *adaptively* and then stops the algorithm when this upper bound falls below ϵ. With this rule we are able to achieve run times for *typical* problem instances that are much faster than the worst case bound.

3 Experimental Results

To illustrate the run-time performance of the `Composite` training algorithm, we made use of the cybersecurity data set introduced in [23]. This data was derived from network traffic collected from a single computer over a 16-month period. Each vector in the data set contains 12 feature values, each representing some measurement of network activity over a one-hour window (e.g. "average number of bytes per session"). All values are normalized to fall in the interval $[0, 1]$. This collected data was used to represent samples from our unknown data distribution Q, and our goal was to build a detector that would recognize anomalous behavior from the machine. We used a uniform distribution over $[0,1]^{12}$ for the background distribution μ. The kernel function used in our DLD-SVM problem was the Gaussian RBF kernel

$$k(x,x') = e^{-\sigma^2\|x-x'\|^2}.$$

A grid search over λ and σ^2 was used to determine values that provided the best performance on a set of hold-out data [23]. This resulted in parameter values $\lambda = 10^{-7}$ and $\sigma^2 = 0.1$ and a solution that separates the training data. The associated hold-out value of $\mathcal{R}$ is 0.00025. The corresponding *alarm rate* (i.e. the rate at which anomalies are predicted by the classifier once it is placed in operation) is 0.0005. This corresponds to approximately one alarm every three months. This rate can be adjusted by training with a different value of ρ. It was also noted during the grid search that some parameter value selections (e.g. $\lambda = 0.05$ and $\sigma^2 = 0.05$) provided partial separation of the training data while exhibiting markedly different run-time behavior. We decided to repeat our experimental analysis using these parameter values as well.

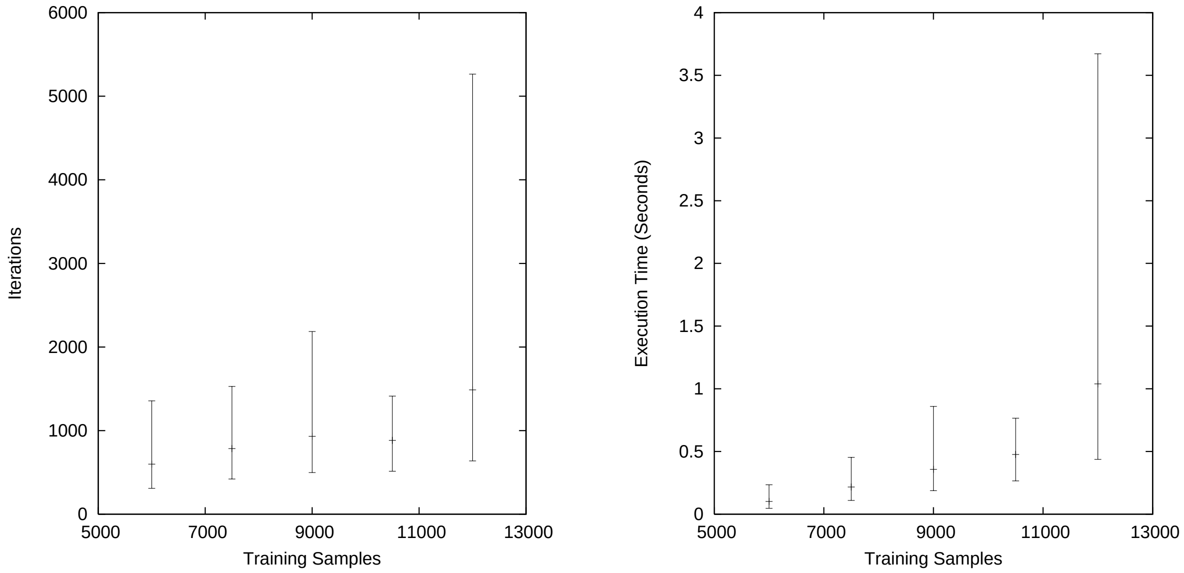

Figure 1: Training with cybersecurity data set ($\lambda = 10^{-7}$, $\sigma^2 = 0.1$).

In our experiments, we focus only on the run-time characteristics of the main loop in the `Composite` algorithm. We have purposefully omitted the setup time for each experiment from our plots. This non-trivial amount of work included: (A) drawing random samples from the base data sets; (B) setting up internal variables for the canonical dual formulation of the DVD-SVM problem; (C) initialization of the algorithm to a feasible solution; and (D) pre-computing all kernel values $k(x_j, x_k)$.

To study the effect of training set size on the run-time properties of `Composite`, we used five different problem sizes. For simplicity, we always chose the number of background data samples (drawn from μ) to be twice the number of actual network data samples (drawn from Q). The five problem sizes of data drawn from $Q:\mu$ were 2000:4000, 2500:5000, 3000:6000, 3500:7000, and 4000:8000. For each of these problem sizes, we performed ten different random samplings of our base data sets, and trained our DLD-SVM classifier on each. The density level ρ was always fixed at 1, and accuracy ϵ was fixed at 10^{-6}. Tabulated results include the min, max, and average number of main loop iterations. Results when using parameter values of $\lambda = 10^{-7}$ and $\sigma^2 = 0.1$ are given in Figure 1. For all of these experiments, our adaptive stopping criterion was able to terminate the main processing loop after a total number of iterations that was about 9 orders of magnitude smaller than the theoretical worst-case given by Equation 6. Wallclock execution times for the main processing loop were also tabulated, as shown in Figure 1. It is noteworthy that there was a significant variance in number of iterations across different random samplings of the same problem size (3x-8x).

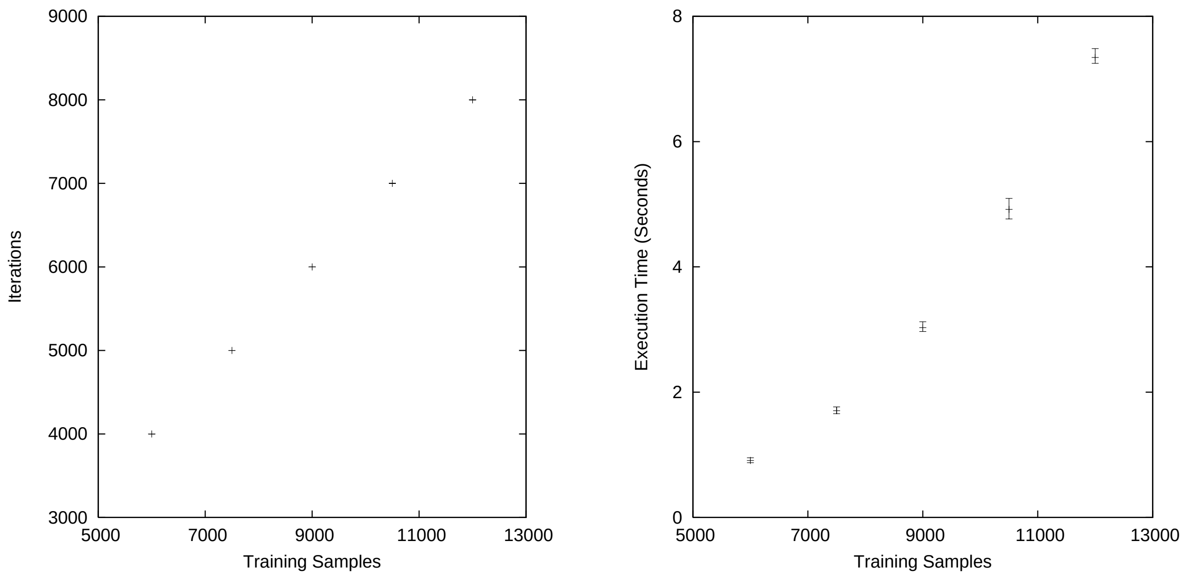

Figure 2: Training with cybersecurity data set ($\lambda = 0.05$, $\sigma^2 = 0.05$).

Results from our second set of experiments, using the same randomly sampled data sets that we used before, are given in Figure 2. This time we selected algorithm parameters ($\lambda = 0.05$ and $\sigma^2 = 0.05$) that gave solutions that did not separate the training data. This larger value of λ, which corresponds to strong regularization, caused our DLD-SVM to always produce a simple solution that discriminates based on a difference in means. All variables were forced to one of the extreme values defined by the canonical dual's inequality constraints (see Equation 3), and every randomly sampled subset of a given size required the same number of iterations for convergence. The relationship between the number of main loop iterations and the training set size is demonstrably linear in this case, and the corresponding wallclock processing time indeed appears quadratic. Training always required more effort when using these parameter values than with those used in our first set of experiments.

4 Conclusions

We have proposed a solution method for training a DLD–SVM. This method is guaranteed to satisfy a user-provided accuracy ϵ in low order polynomial time. Experimental results suggest that actual run times can be many orders of magnitude smaller than our theoretical worst-case bounds.

References

[1] S. Ben-David and M. Lindenbaum. Learning distributions by their density levels: a paradigm for learning without a teacher. *J. Comput. System Sci.*, 55:171–182, 1997.

[2] C.C. Chang, C.W. Hsu, and C.J. Lin. The analysis of decomposition methods for support vector machines. *IEEE Transactions on Neural Networks*, 11(4):1003–1008, 2000.

[3] P.-H. Chen, R.-E. Fan, and C.-J. Lin. A study on SMO-type decomposition methods for support vector machines. Technical report, 2005. http://www.csie.ntu.edu.tw/ cjlin/papers.html.

[4] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods.* Cambridge University Press, Canbridge ; United Kingdom, 1st edition, 2000.

[5] C.-W. Hsu and C.-J. Lin. A simple decomposition algorithm for support vector machines. *Machine Learning*, 46:291–314, 2002.

[6] D. Hush, P. Kelly, C. Scovel, and I. Steinwart. QP algorithms with guaranteed accuracy and run time for support vector machines. Technical report 05-5165, Los Alamos National Laboratory, 2005. submitted for publication.

[7] D. Hush and C. Scovel. Polynomial-time decomposition algorithms for support vector machines. *Machine Learning*, 51:51–71, 2003.

[8] T. Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning.* MIT Press, Cambridge, MA, 1998.

[9] S.S. Keerthi and E.G. Gilbert. Convergence of a generalized SMO algorithm for SVM classifier design. *Machine Learning*, 46:351–360, 2002.

[10] S.S. Keerthi, S.K. Shevade, C. Bhattacharyya, and K.R.K. Murthy. Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Computation*, 13:637–649, 2001.

[11] P. Laskov. Feasible direction decomposition algorithms for training support vector machines. *Machine Learning*, 46(1–3):315–349, 2002.

[12] C.-J. Lin. Linear convergence of a decomposition method for support vector machines. Report, 2001. http://www.csie.ntu.edu.tw/ cjlin/papers.html.

[13] C.-J. Lin. On the convergence of the decomposition method for support vector machines. *IEEE Transactions on Neural Networks*, 12:1288–1298, 2001.

[14] N. List and H.U. Simon. A general convergence theorem for the desomposition method. In J. Shawe-Taylor and Y. Singer, editors, *17th Annual Conference on Learning Theory, COLT 2004, volume 3120 of Lecture Notes in Computer Science*, pages 363–377, 2004.

[15] N. List and H.U. Simon. General polynomial time decomposition algorithms. Report, 2005. submitted for publication.

[16] O.L. Mangasarian and D.R. Musicant. Lagrangian support vector machines. *Journal of Machine Learning Research*, 1:161–177, 2001.

[17] E.E. Osuna, R. Freund, and F. Girosi. Support vector machines: training and applications. Technical Report AIM-1602, MIT, 1997.

[18] J.C. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 41–64. MIT Press, Cambridge, MA, 1998.

[19] B. Schölkopf, J.C. Platt, J. Shawe-Taylor, and A.J. Smola. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13:1443–1471, 2001.

[20] C. Scovel, D. Hush, and I. Steinwart. A computation–performance bound for SVM classifiers. *in preparation*, 2005.

[21] C. Scovel, D. Hush, and I. Steinwart. Learning rates for density level detection. *Analysis and Applications*, 2005. to appear.

[22] H.U. Simon. On the complexity of working set selection. In *Proceedings of the 15th International Conference on Algorithmic Learning Theory*, 2004.

[23] I. Steinwart, D. Hush, and C. Scovel. A classification framework for anomaly detection. *Journal of Machine Learning Research*, 6:211–232, 2005.