Proximal Knowledge-Based Classification

O. L. Mangasarian* E. W. Wild† G. M. Fung ‡

June 26, 2008

Abstract

Prior knowledge over general nonlinear sets is incorporated into proximal nonlinear kernel classification problems as linear equalities. The key tool in this incorporation is the conversion of general nonlinear prior knowledge implications into linear equalities in the classification variables without the need to kernelize these implications. These equalities are then included into a proximal nonlinear kernel classification formulation [1] that is solvable as a system of linear equations. Effectiveness of the proposed formulation is demonstrated on a number of publicly available classification datasets. Nonlinear kernel classifiers for these datasets exhibit marked improvements upon the introduction of nonlinear prior knowledge compared to nonlinear kernel classifiers that do not utilize such knowledge.

Keywords: prior knowledge, kernel classification, proximal support vector machines

1 Introduction

Prior knowledge has been used effectively in improving classification for both linear [2] and nonlinear [3] kernel classifiers, as well as for nonlinear kernel approximation [4, 5]. In these applications, prior knowledge was specified over polyhedral regions, but when a nonlinear kernel was used, the kernelized prior knowledge could not be easily related to the original polyhedral regions. To address this concern, prior knowledge over *nonlinear* regions was added to nonlinear kernel approximation [6] and classification [7] *without* kernelizing the prior knowledge. All of these approaches convert the prior knowledge into linear inequalities which are added to the linear programming formulation of a support vector machine for classification or approximation [8]. In contrast, proximal nonlinear classifiers [1, 9, 10] require only the solution of a system of linear equations. Solvers for such systems are both computationally cheaper and more available than linear programming solvers. Here, we consider adding nonlinear prior knowledge as linear *equalities* to proximal nonlinear classification. Converting prior knowledge implications to linear equalities is motivated by a theorem of the alternative for convex functions [7, Theorem 2.1] employed as described in Proposition 2.1 in Section 2. Other interesting recent approaches to knowledge-based support vector machines include modifying the hypothesis space rather than the optimization problem [11], and adding additional points labeled based on the prior knowledge to the dataset [12].

In Section 3 we describe our proximal nonlinear kernel classification formulation that incorporates nonlinear prior knowledge as linear equalities, leading to a symmetric positive definite system of linear equations. Section 4 shows that prior knowledge can improve a nonlinear kernel classification significantly on public datasets. Section 5 concludes the paper.

We describe our notation now. All vectors will be column vectors unless transposed to a row vector by a prime $'$. The scalar (inner) product of two vectors x and y in the n-dimensional real space R^n will be denoted by $x'y$. For $x \in R^n$, $\|x\|_1$ denotes the 1-norm while $\|x\|$ denotes the 2-norm and x_+ denotes the vector $\max\{x, 0\}$, that is the vector the i-th component of which is the maximum of x_i and 0. The notation $A \in R^{m\times n}$ will signify a real $m \times n$ matrix. For such a matrix, A' will denote the transpose of A, A_i will denote the i-th row of A and $A_{\cdot j}$ the j-th column of A. A vector of all ones or all zeros of arbitrary dimension will be denoted by e and 0, respectively. Thus for $y \in R^m$ the notation $e'y$ will denote the sum of the components of y. For $A \in R^{m\times n}$ and $B \in R^{n\times k}$, a *kernel* $K(A, B)$ maps $R^{m\times n} \times R^{n\times k}$ into $R^{m\times k}$. In particular, if x and y are column vectors in R^n then, $K(x', y)$ is a real number, $K(x', A')$ is a row vector in R^m and $K(A, A')$ is an $m \times m$ matrix. We shall make no assumptions

*Computer Sciences Department, University of Wisconsin, Madison, WI 53706 and Department of Mathematics, University of California at San Diego, La Jolla, CA 92093. *olvi@cs.wisc.edu*.

†Computer Sciences Department, University of Wisconsin, Madison, WI 53706. *wildt@cs.wisc.edu*.

‡Siemens Medical Solutions, Inc., 51 Valley Stream Parkway, Malvern, PA 19355. *glenn.fung@siemens.com*.

whatsoever on our kernels other than symmetry, that is $K(x',y)' = K(y',x)$, and in particular we shall not assume or make use of Mercer's positive definiteness condition [13]. The base of the natural logarithm will be denoted by ε. In this paper, we will use the frequently used Gaussian kernel [8] whose ij-th element, $i = 1,\ldots,m$, $j = 1,\ldots,k$, is given by: $(K(A,B))_{ij} = \varepsilon^{-\mu\|A_i{}' - B_{\cdot j}\|^2}$, where $A \in R^{m\times n}$, $B \in R^{n\times k}$ and μ is a positive constant.

2 Conversion of Nonlinear Prior Knowledge into a Linear Equality

We wish to impart prior knowledge to the problem of classifying a dataset in R^n represented by the m rows of the matrix $A \in R^{m\times n}$ that are labeled as belonging to the class $+1$ or -1 by a corresponding diagonal matrix $D \in R^{m\times m}$ of ± 1's. The nonlinear kernel classifier to be generated based on this data as well as prior knowledge will be:

$$K(x',B')u - \gamma = 0, \tag{2.1}$$

where $B \in R^{k\times n}$ and $K(x',B') : R^{1\times n} \times R^{n\times k} \longrightarrow R^{1\times k}$ is an arbitrary kernel function. The values of the variables $u \in R^k$ and $\gamma \in R$ will be determined by an optimization problem such that the labeled data A satisfy, to the extent possible, the separation condition:

$$D(K(A,B')u - e\gamma) \geq 0. \tag{2.2}$$

Condition (2.2) places the $+1$ and -1 points represented by A on opposite sides of the nonlinear separating surface (2.1). In general the matrix B is set equal to A [14]. However, in reduced support vector machines [15] $B = \bar{A}$, where $\bar{A}$ is a submatrix of A whose rows are a small subset of the rows of A. In fact, B can be an arbitrary matrix in $R^{k\times n}$. We now impose prior knowledge on the construction of our classifier function $K(x',B')u - \gamma$ to ensure that a certain set of points lies on the $+1$ side of the classifier (2.1). We achieve this through the following implication:

$$g(x) \leq 0 \implies K(x',B')u - \gamma = 1, \ \ \forall x \in \Gamma_1. \tag{2.3}$$

Here, $g(x) : \Gamma_1 \subset R^n \longrightarrow R^r$ is an r-dimensional function defined on a subset Γ_1 of R^n that determines the region $\{x|g(x) \leq 0\}$. The prior knowledge requires that for any x in this region, the classifier function $K(x',B')u - \gamma$ return a value $+1$. Such points would thus be classified in class $+1$. Prior knowledge about the class -1 can be added through a similar implication.

It is interesting to compare the implication (2.3) to the similar implication treated in [7]:

$$g(x) \leq 0 \implies K(x',B')u - \gamma \geq 1, \ \ \forall x \in \Gamma_1. \tag{2.4}$$

This implication has the same interpretation as (2.3), but uses an inequality constraint to impose the condition that any x for which $g(x) \leq 0$ is classified in class $+1$. While implication (2.4) can be added as a set of linear inequality constraints to the linear programming formulation of an SVM, here we shall preserve the computational advantages of the proximal SVM by adding implication (2.3) as a set of linear *equality* constraints as follows.

The implication (2.3) can be written in the following equivalent form:

$$g(x)_+ = 0 \implies K(x',B')u - \gamma = 1, \ \ \forall x \in \Gamma_1, \tag{2.5}$$

where $g(x)_+ = \max\{g(x),0\}$. The use of $g(x)_+ = 0$ in (2.5) in place of $g(x) \leq 0$ is a key observation which allows us to convert the implication to a linear equality via Proposition 2.1 with a multiplier that is not required to be nonnegative. We can then add this linear equality to a proximal nonlinear classifier *without* adding nonnegativity constraints. Motivated by a fundamental theorem of the alternative for convex functions [7, Theorem 2.1],[16, Corollary 4.2.2], Proposition 2.1 ensures implication (2.5) is satisfied once a certain linear equality is satisfied.

PROPOSITION 2.1. **Prior Knowledge as a Linear Equality** *The implication (2.5), or equivalently the implication (2.3), is satisfied if $\exists v \in R^r$ such the following linear equality in v is satisfied:*

$$K(x',B')u - \gamma - 1 + v'g(x)_+ = 0, \forall x \in \Gamma_1. \tag{2.6}$$

Proof If the implication (2.5) does not hold then for some $x \in \Gamma_1$ such that $g(x)_+ = 0$ it follows that $K(x',B')u - \gamma \neq 1$. However this leads to the following contradiction for that x:

$$0 = K(x',B')u - \gamma - 1 + v'g(x)_+ = K(x',B')u - \gamma - 1 \neq 0, \tag{2.7}$$

where the first equality follows from (2.6), the second equality from $g(x)_+ = 0$ and the inequality follows from $K(x',B')u - \gamma \neq 1$. □

We now use Proposition 2.1 to formulate proximal knowledge-based nonlinear kernel classification.

3 Nonlinear Prior Knowledge via Proximal Support Vector Machines

We first formulate the classification problem (2.2) without knowledge using a proximal support vector

machine approach [1]. The error in a proximal formulation is measured by closeness to the two following bounding surfaces that are parallel to the classifier (2.1) in the u-space:

$$\begin{array}{l} K(x', B')u - \gamma = +1 \\ K(x', B')u - \gamma = -1 \end{array} \tag{3.8}$$

In contrast to an ordinary support vector machine, the two parallel surfaces in (3.8) are chosen so that the first surface is close to the training points in class $+1$ and the second surface is close to points in class -1, and the two surfaces are as far apart as possible. Thus, we measure the error in satisfying (2.2) by how close the points in class $+1$ and class -1 are to the first and second surface of (3.8), respectively. This error can be succinctly written as $\|D(K(A, B')u - e\gamma) - e\|$. Minimizing the square of this error with parameter weight $\nu/2$ and the square of the 2-norm of the variables (u, γ) for model simplicity, we obtain our proximal support vector machine classification formulation without prior knowledge:

$$\min_{(u,\gamma)} \ \frac{\nu}{2}\|D(K(A, B')u - e\gamma) - e\|^2 + \frac{1}{2}\|u\|^2 + \frac{1}{2}\gamma^2. \tag{3.9}$$

This unconstrained quadratic optimization problem can be solved by setting its gradient, a system of linear equations in (u, γ), equal to zero. Before we do that we shall introduce prior knowledge in the form of implication (2.5) by means of the linear equality (2.6) with weight $\sigma/2$ to be satisfied in a least square sense at ℓ discrete points x^i, $i = 1, \ldots, \ell$ in the set Γ_1 as follows:

$$\begin{array}{ll} \min\limits_{(u,\gamma,v)} & \frac{\nu}{2}\|D(K(A, B')u - e\gamma) - e\|^2 + \\ & \frac{\sigma}{2}\sum\limits_{i=1}^{\ell}(K(x^{i'}, B')u - \gamma - 1 + \\ & v'g(x^i)_+)^2 + \\ & \frac{1}{2}\|u\|^2 + \frac{1}{2}\gamma^2 + \frac{1}{2}\|v\|^2. \end{array} \tag{3.10}$$

In the above equation, we have also added an additional regularization term, $\|v\|^2$. This term ensures that the optimization problem is strongly convex, and trades off the size of v with the fit to the data, the extent to which (2.6) is satisfied, and the solution complexity. Figure 1 and its caption describe an example of the effect of v.

REMARK 3.1. *One important consequence of Proposition 2.1 is that the multiplier v causes the linear equality* (2.6) *to be sufficient for the implication* (2.5) *to hold* **even if Γ_1 does not closely match** $\{x|g(x)_+ = 0\}$. *In terms of (3.10), the discretization points x^i, $i = 1, \ldots, \ell$ need not be in the set $\{x|g(x)_+ = 0\}$. This sets our approach apart from those such as [12] where knowledge is imparted through discrete points in $\Gamma_1 = \{x|g(x)_+ = 0\}$. In the extreme example shown in Figure 1, our formulation* (3.10) *gives similar results whether Γ_1 is disjoint from $\{x|g(x)+ = 0\}$ or not. In Section 4.2 we do not ensure that every point in the discretization satisfies the left-hand side of the prior knowledge implication.*

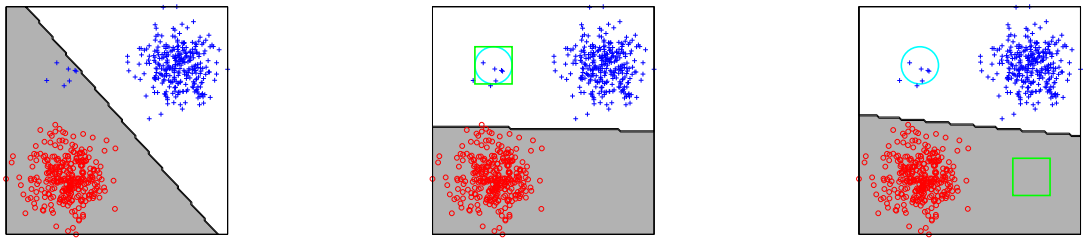
Since the objective function of (3.10) is strongly convex with a symmetric positive definite Hessian matrix, it has a unique solution obtained by setting its gradient equal to zero. This gives a system of $k + 1 + r$ *nonsingular linear equations* in as many unknowns (u, γ, v). We note that additional prior knowledge about the class -1 can be added to (3.10) in an analogous way. Finally, the prior knowledge can easily be imposed at many points by using the incremental technique of [10] since the size of the system of linear equations does not depend on ℓ. We note that in the linear programming approach of [7], both the number of constraints and the number of variables in the linear program grow linearly with ℓ. The fact that the size of the system of equations does not depend on ℓ allows our approach to potentially handle knowledge which is imposed at a huge number of points, which would be difficult to solve using the linear programming formulation.

We turn now to numerical examples of our proximal prior knowledge approach.

4 Computational Results

We illustrate the effectiveness of our proposed formulation in two ways: by comparing our approach to [7], and by comparing our approach to a proximal nonlinear kernel classifier *without* knowledge in new experiments in which prior knowledge is generated from ordinary datasets.

4.1 Comparing Linear Programming and Proximal Knowledge-Based Classification In [7], prior knowledge implications of the form in (2.3) were converted to linear *inequalities* and added to a linear program for nonlinear kernel classification. Prior knowledge was incorporated into three publicly available datasets: the Checkerboard dataset [17], the Spiral dataset [18], and the Wisconsin Prognostic Breast Cancer (WPBC) dataset [19]. Here we compare our proposed proximal formulation to the linear programming formulation using the same prior knowledge. Our results show that our proximal formulation can obtain solutions with accuracy similar to the linear programming formulation, while being



(a) (b) (c)

Figure 1: **An example showing that the set Γ_1 discretized in (3.10) need not contain the region $\{x|g(x)_+ = 0\}$ in which the left-hand side of the implication (2.5) is satisfied. Each of the figures (a), (b) and (c) depict 600 points denoted by "+" and "o" that are obtained from three bivariate normal distributions. Another 400 points from the same three distributions in the same proportions were used for training and tuning each of the three classifiers of figures (a), (b) and (c). For simplicity, we use the linear kernel $K(A, B') = AB'$ to obtain three different linear classifiers to discriminate between the +'s and o's. A linear classifier without prior knowledge is shown in (a). Note that some + points from the rarely sampled distribution are misclassified. Figures (b) and (c) show classifiers using the same data *and* the prior knowledge $(\|x - \binom{-3}{3}\| - 1)_+ = 0 \Longrightarrow K(x', B')u - \gamma = 1$. The left hand side of this implication is true in the region enclosed by the circle in (b) and (c) and contains most of the + points from the rarely sampled distribution. The prior knowledge is imposed over a single grid of 100 evenly spaced points in the square shown and the parameter σ of (3.10) was set to 1. In (b), the square contains the set $\{x|(\|x - \binom{-3}{3}\| - 1)_+ = 0\}$, but the square of (c) is highly disjoint from the set $\{x|(\|x - \binom{-3}{3}\| - 1)_+ = 0\}$. Nevertheless, the classifier of (c) is nearly indistinguishable from that in (b). Techniques such as [12], which incorporate prior knowledge by adding points which conform to the knowledge as "pseudo" points to the training set, will not make use of a discretization such as that of (c). Our approach is able to handle points in the discretization that are not in $\{x|g(x)_+ = 0\}$ partly because of the multiplier v in (2.6). At the solution shown in (c), $v'g(x)_+ > 1$ in the discretized region. For such x, (2.6) implies that x should have class -1. Thus, v can select which side of the separating surface (2.1) points with a relatively large $g(x)_+$ should lie on. In (3.10), the size of v is traded off against the fit to the data, the extent to which (2.6) is satisfied, and the solution complexity. This extreme example illustrates an important property of our approach: Proposition 2.1 does not require that Γ_1 match the set $\{x|g(x)_+ = 0\}$ closely.**

much faster to solve.

4.1.1 Checkerboard and Spiral Problems

The frequently used Checkerboard [17, 3] and Spiral [18, 1] datasets are synthetic datasets for which prior knowledge can be easily constructed [7, 3]. The checkerboard dataset consists of points with labels "black" and "white" arranged in the shape of a checkerboard, while the spiral dataset consists of points from two concentric spirals. Table 1 shows both the accuracy and CPU time needed to run the experiments of [7] using both the linear programming formulation originally used in [7] and our proposed proximal formulation. On both datasets, 225 optimization problems were solved. In the checkerboard experiment, the matrix B has 16 rows, and the prior knowledge is imposed at 200 points. In the spiral dataset, the matrix B has 194 rows, and the knowledge is imposed at 194 points. For the checkerboard experiment, the knowledge consists only of the two leftmost squares of the bottom row, while for the spiral dataset, the knowledge was constructed by inspecting the source code used to generate the spiral. The experiments were performed using MATLAB 7.2 [20] under CentOS Linux 4.4 on an Intel Pentium IV 3 GHz processor with 1 gigabyte of RAM. The running times were calculated by the MATLAB profiler, and represent the total time during the experiment consumed in setting up and solving the optimization problem. Linear programs were solved using CPLEX 9.0 [21], and the linear systems of equations were solved using the `chol` routine of MATLAB. For the spiral dataset, flush-to-zero mode was enabled to speed the multiplication of numbers with very small magnitude. This change had a significant impact on the running time of our proximal formulation, and negligible impact on the linear programming formulation. Note that the proximal formulation has similar accuracy to the linear programming formulation, while being *approximately an order of*

magnitude faster to solve. We further note that considering only the time taken to solve the linear program or linear system of equations gives a similar result, thus we do not believe that the difference in computation time can be attributed to the setup procedure.

4.1.2 Predicting Breast Cancer Survival Time We have also tested our proposed proximal formulation on the Wisconsin Prognostic Breast Cancer (WPBC) dataset [19]. This dataset contains thirty cytological features obtained from a fine needle aspirate and two histological features, tumor size and the number of metastasized lymph nodes, obtained during surgery for breast cancer patients. The dataset also contains the amount of time before each patient experienced a recurrence of the cancer, if any. Here, we shall consider the task of predicting whether a patient will remain cancer free for at least 24 months. In [7] prior knowledge was used to achieve 91% correctness on this task using only the histological features. In this dataset, 81.9% of patients are cancer free after 24 months. To our knowledge, the best result on this dataset without the knowledge used in [7] is 86.3% correctness obtained by Bennett in [22]. We have repeated the experiment of [7], which we briefly describe below for completeness.

In order to evaluate our proposed proximal approach, we compared the misclassification rates of two classifiers on this dataset. One classifier is learned without prior knowledge, while the second classifier is learned using the prior knowledge from [7], which is depicted in Figure 2 (a). For both cases the rows of the matrices A and B of (3.10) were set to the usual values, that is to the coordinates of the points of the training set. The misclassification rates are computed using leave-one-out cross validation. For each fold, the parameter ν and the kernel parameter μ were chosen from the set $\{2^i | i = -7, \ldots, 7\}$ by using ten-fold cross validation on the training set of the fold. In the classifier with prior knowledge, the parameter σ was set to 10^6 for simplicity, which corresponds to very strict adherence to the prior knowledge. The results are summarized in Figure 2 (b). We note that both the linear programming and proximal formulations misclassify the same number of examples on this dataset, even though they generate slightly different separating surfaces. The reduction in misclassification rate indicates that our proximal approach can achieve the same 50% improvement in classification accuracy using prior knowledge as the linear programming formulation [7].

4.2 Generating Prior Knowledge from Ordinary Classification Datasets In order to further demonstrate the effectiveness of our proposed formulation, we generate prior knowledge from a subset of an ordinary classification dataset. In these experiments, we will ensure that the prior knowledge is generated without knowing the true distribution of the data, or inspecting the full data set. By using ordinary datasts, we are also able to easily demonstrate our proposed formulation on datasets with more than two features. In order for prior knowledge to improve classification accuracy when combined with ordinary data, the prior knowledge and the data must contain different information about the "true" dataset. Thus, we simulate a situation where a knowledge-based classifier using both prior knowledge and data will be superior to a classifier that uses either the data or prior knowledge alone. In our scenario, the set $\mathcal{M}^+$ will consist mostly of points from the class $+1$ and will be used only to generate prior knowledge, while the set $\mathcal{M}^-$ will consist mostly of points from the class -1 and will be used only as ordinary data. We varied the percentage of negative points in $\mathcal{M}^+$. As this percentage approaches fifty percent one expects that $\mathcal{M}^+$ and $\mathcal{M}^-$ will contain the same information, and the gain due to incorporating prior knowledge will be minimal. Construction of the sets $\mathcal{M}^+$ and $\mathcal{M}^-$ is illustrated in Figure 3 (a). The motivation for this scenario is a situation in which prior knowledge is available about data in the set $\mathcal{M}^+$, while only the set $\mathcal{M}^-$ is available as ordinary data. Thus, the learning algorithm will need to incorporate both prior knowledge about $\mathcal{M}^+$ and the conventional data in $\mathcal{M}^-$ in order to generalize well to new points.

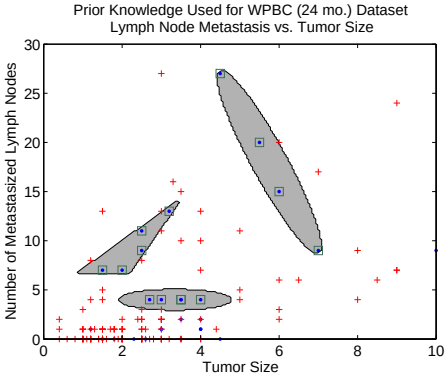
One can imagine many methods of automatically generating prior knowledge from $\mathcal{M}^+$, such as [23]. However, we used the simple approach of learning a proximal support vector machine on the points in $\mathcal{M}^+$. The knowledge we used was the following:

(4.11)
$$(-\phi(x))_+ = 0 \Longrightarrow K(x', B')u - \gamma = 1, \ \forall x \in \Gamma_1,$$

where $\phi(x)$ is the classifier function (2.1) learned on the set $\mathcal{M}^+$. This knowledge simply states that if the proximal support vector machine represented by $\phi(x)$ labels the point as $+1$, then the point should be labeled $+1$ by the classifier which combines both data and knowledge. We impose the prior knowledge of (4.11) at a random sample from a multivariate normal distribution fit to the points in $\mathcal{M}^+$. We chose the multivariate normal distribution for simplicity, but one can easily imagine using a more sophisticated distribution. Investigation of different methods of generating prior knowledge is left to future research.

Table 1: **Accuracy and CPU time in seconds for the linear programming formulation [7] and the proposed proximal formulation. Each running time result is the total time needed to set up and solve the optimization problem, either as a linear program or a linear system of equations, 225 times. The time ratio is the time for the linear programming formulation divided by the time for the proximal formulation.**

Dataset	Linear Programming SVM [7] Accuracy CPU Time in Seconds	Proximal SVM Accuracy CPU Time in Seconds	Time Ratio
Checkerboard without Knowledge	89.2% 2.3	94.2% 0.2	11.5
Checkerboard with Knowledge	100% 26.4	98.0% 3.2	8.3
Spiral without Knowledge	79.9% 21.3	80.4% 4.3	5.0
Spiral with Knowledge	100% 300.2	100% 19.0	15.8



(a)

Classifier	Misclassification Rate
Without knowledge	0.1806
With knowledge	**0.0903**
Improvement due to knowledge	50.0%

(b)

Figure 2: **(a) Number of metastasized lymph nodes versus tumor size for the WPBC (24 mo.) dataset. The solid dots represent patients who experienced a recurrence within 24 months of surgery, while the crosses represent the cancer free patients. The shaded regions which correspond to the areas in which the left-hand side of one of the three implications in the form of Equation (2.3) is true simulate an oncological surgeon's prior knowledge regarding patients that are likely to have a recurrence. Prior knowledge was enforced at the points enclosed in squares. (b) Leave-one-out misclassification rate of classifiers with and without knowledge on the WPBC (24 mo.) dataset. Best result is in bold.**

Since $\phi(x)$ is learned with few negative examples, it will likely not be accurate over the entire dataset. In fact, in our experiments $\phi(x)$ alone always had similar accuracy on the test set to the classifier built using only the data in $\mathcal{M}^-$. Intuitively, since $\phi(x)$ is a kernel classifier, the knowledge we imposed states that points which are "close" to the $+1$ points in $\mathcal{M}^+$ used to construct $\phi(x)$ should be labeled $+1$. However, enforcement of this knowledge is balanced against fitting the ordinary data points in $\mathcal{M}^-$. Thus, the combination of data and prior knowledge is necessary to obtain high accuracy.

Figure 3 (b) shows the result of applying the above procedure to Thompson's Normally Distributed Clusters on Cubes (NDCC) dataset [24]. This dataset generates points according to multivariate normal distributions centered at the vertices of two concentric 1-norm cubes. Points are mostly labeled according to the cube they were generated from, with some specified fraction of noisy labels. We generated a dataset of 20000 points in R^{50}, with ten percent label noise. We used 300 points as a training set, 2000 separate points to choose parameters from the set $\{2^i | i = -7, \ldots, 7\}$, and the remaining 17700 points to evaluate the classifiers. In Figure 3 (b), we compare an approach using only the data in the set $\mathcal{M}^-$ and no prior knowledge to an approach based on the same data *plus* prior knowledge obtained from the points in $\mathcal{M}^+$, with σ equal to ν. The knowledge was imposed on $|\mathcal{M}^+|$ randomly sampled points as described above, where $|\mathcal{M}^+|$ is the cardinality of $\mathcal{M}^+$. In our experience on this dataset, reducing the number of sampled points to less than half $|\mathcal{M}^+|$ had very little impact on accuracy. Determining the appropriate number of points to sample for a given dataset is left to future work. The approach using prior knowledge is able to approach ten percent misclassification error even when relatively few points in $\mathcal{M}^-$ have label $+1$.

Figure 4 shows the result of applying the above procedure to the publicly available Wisconsin Diagnostic Breast Cancer (WDBC) and Ionosphere datasets [19]. In the WDBC dataset, the task is to classify tumors as either malignant or benign based on the 30 features given. To simulate the scenario in which most information about malignant tumors is available only through prior knowledge, while information about benign tumors is more readily gathered, we label malignant tumors $+1$. In the Ionosphere dataset, the task is to classify radar returns as either good or bad based on the 34 features given. We chose to label bad radar returns $+1$. To asses the generalization performance of our approach, we computed ten-fold cross validation misclassification rates. We chose all parameters from the set $\{2^i | i = -7, \ldots, 7\}$ using ten-fold cross validation on the training set. When using prior knowledge, we set σ equal to ν. In carrying out the cross validation experiment, $\mathcal{M}^+$ and $\mathcal{M}^-$ were formed from the training set for each fold. In Figure 4, three different approaches are compared. In the first approach, represented by squares, the classifier is learned using only the data in $\mathcal{M}^-$ with *no* prior knowledge. This classifier performs poorly until a sufficient number of $+1$ points are present in $\mathcal{M}^-$. The second approach, represented by circles, learns a classifier using the data in $\mathcal{M}^-$ *plus* the prior knowledge from $\mathcal{M}^+$ described by (4.11). The knowledge was imposed at $|\mathcal{M}^+|$ randomly generated points as described above. Note that the use of prior knowledge results in considerable improvement, especially when there are few points in $\mathcal{M}^+$ with class -1. For reference, we include an approach represented by triangles which uses no prior knowledge, but *all* the data. Note that this classifier has the same misclassification rate regardless of the fraction of negative points in $\mathcal{M}^+$. Including this approach illustrates that our approach is able to use the prior knowledge generated from $\mathcal{M}^+$ to recover most of the information in $\mathcal{M}^+$. Recall that we are simulating a situation in which $\mathcal{M}^+$ is only available as prior knowledge.

5 Conclusion and Outlook

We have proposed a computationally effective framework for handling general nonlinear prior knowledge in proximal kernel classification problems. We have reduced such prior knowledge to an easily implemented linear equation that can be incorporated into an unconstrained strongly convex quadratic programming problem. We have demonstrated the effectiveness of our approach on a number of publicly available datasets. One possible future extension is to even more general prior knowledge, such as that where the right hand side of the implication (2.3) is replaced by a very general nonlinear inequality involving the classification function (2.1). Other interesting future extensions include proximal knowledge-based approximation, and the construction of an interface which allows users to easily specify arbitrary regions to be used as prior knowledge.

References

[1] G. Fung and O. L. Mangasarian. Proximal support vector machine classifiers. In F. Provost and R. Srikant, editors, *Proceedings KDD-2001: Knowledge Discovery and Data Mining, August 26-29,*

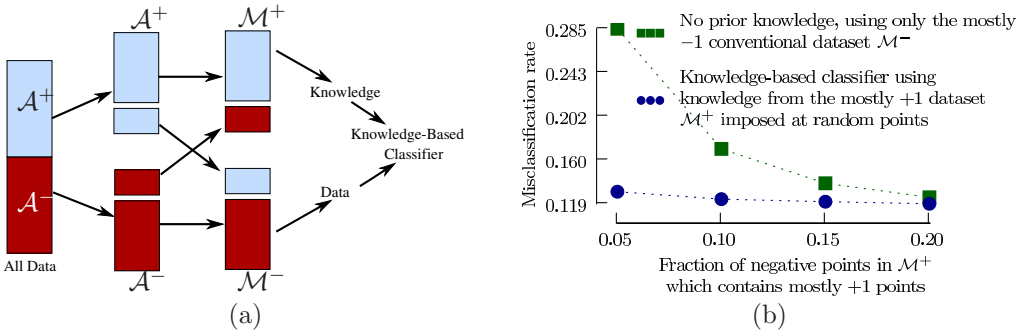


Figure 3: **(a) Generation of prior knowledge from a standard dataset. The dataset is first separated into the datasets $\mathcal{A}^+$ which consists of all $+1$ points, and $\mathcal{A}^-$ which consists of all -1 points. Then the mostly $+1$ dataset $\mathcal{M}^+$ is formed by replacing a small fraction of $+1$ points in $\mathcal{A}^+$ with an equal number of -1 points from $\mathcal{A}^-$. The mostly -1 dataset $\mathcal{M}^-$ is formed from the points not used in $\mathcal{M}^+$. We use $\mathcal{M}^+$ to produce prior knowledge, and $\mathcal{M}^-$ as ordinary data. Combining the knowledge from $\mathcal{M}^+$ and the data from $\mathcal{M}^-$ leads to a knowledge-based classifier which is superior to a classifier formed using either $\mathcal{M}^+$ as pure knowledge or $\mathcal{M}^-$ as pure data alone. (b) Prior knowledge experiment on the NDCC dataset: 300 points in R^{50}.**

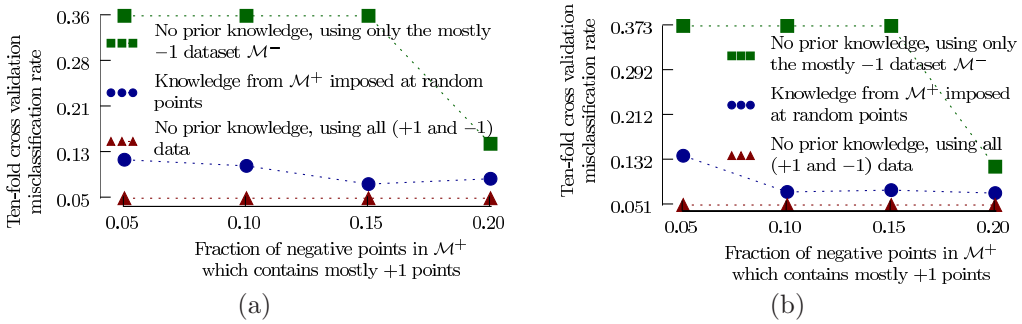


Figure 4: **Prior knowledge experiment on (a) the WDBC dataset: 569 points in R^{30}, with 212 malignant tumors labeled $+1$; and (b) the Ionosphere dataset: 351 points in R^{34}, with 126 bad radar returns labeled $+1$.**

2001, San Francisco, CA, pages 77–86, New York, 2001. Association for Computing Machinery.

[2] G. Fung, O. L. Mangasarian, and J. Shavlik. Knowledge-based support vector machine classifiers. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 521–528. MIT Press, Cambridge, MA, October 2003.

[3] G. Fung, O. L. Mangasarian, and J. Shavlik. Knowledge-based nonlinear kernel classifiers. In M. Warmuth and B. Schölkopf, editors, *Conference on Learning Theory (COLT 03) and Workshop on Kernel Machines*, pages 102–113, Berlin, 2003. Springer-Verlag.

[4] O. L. Mangasarian, J. W. Shavlik, and E. W. Wild. Knowledge-based kernel approximation. *Journal of Machine Learning Research*, 5:1127–1141, 2004.

[5] R. Maclin, J. Shavlik, L. Torrey, T. Walker, and E. Wild. Giving advice about preferred actions to reinforcement learners via knowledge-based kernel regression. In *Proceedings 20th National Conference on Artificial Intelligence*, pages 819–824, 2005.

[6] O. L. Mangasarian and E. W. Wild. Nonlinear knowledge in kernel approximation. *IEEE Transactions on Neural Networks*, 18:300–306, 2007. ftp://ftp.cs.wisc.edu/pub/dmi/tech-reports/05-05.pdf.

[7] O. L. Mangasarian and E. W. Wild. Nonlinear knowledge-based classification. Technical Report 06-04, Data Mining Institute, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, August 2006. ftp://ftp.cs.wisc.edu/pub/dmi/tech-reports/06-04.pdf, *IEEE Transactions on Knowledge and Data Engineering*, submitted.

[8] O. L. Mangasarian. Generalized support vector machines. In A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 135–146, Cambridge, MA, 2000. MIT Press.

[9] G. Fung and O. L. Mangasarian. Multicategory proximal support vector machine classifiers. *Machine Learning*, pages 77–97, 2005.

[10] G. Fung and O. L. Mangasarian. Incremental support vector machine classification. In H. Mannila R. Grossman and R. Motwani, editors, *Proceedings of the Second SIAM International Conference on Data Mining, Arlington, Virginia, April 11-13,2002*, pages 247–260, Philadelphia, 2002. SIAM.

[11] Q. V. Le, A. J. Smola, and T. Gärtner. Simpler knowledge-based support vector machines. In *Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA, 2006*, 2006.

[12] R. Maclin, J. Shavlik, T. Walker, and L. Torrey. A simple and effective method for incorporating advice into kernel methods. In *Proceedings 21st National Conference on Artificial In telligence*, 2006.

[13] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[14] O. L. Mangasarian. Generalized support vector machines. In A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 135–146, Cambridge, MA, 2000. MIT Press. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/98-14.ps.

[15] Y.-J. Lee and O. L. Mangasarian. RSVM: Reduced support vector machines. In *Proceedings of the First SIAM International Conference on Data Mining, Chicago, April 5-7, 2001, CD-ROM*, 2001.

[16] O. L. Mangasarian. *Nonlinear Programming*. SIAM, Philadelphia, PA, 1994.

[17] T. K. Ho and E. M. Kleinberg. Checkerboard dataset, 1996. http://www.cs.wisc.edu/math-prog/mpml.html.

[18] A. Wieland. Twin spiral dataset. http://www-cgi.cs.cmu.edu/afs/cs.cmu.edu/project/ai-repository/ai/areas/neural/bench/cmu/0.html.

[19] P. M. Murphy and D. W. Aha. UCI machine learning repository, 1992. www.ics.uci.edu/∼mlearn/MLRepository.html.

[20] MATLAB. *User's Guide*. The MathWorks, Inc., Natick, MA 01760, 1994-2006. http://www.mathworks.com.

[21] ILOG, Incline Village, Nevada. *ILOG CPLEX 9.0 User's Manual*, 2003. http://www.ilog.com/products/cplex/.

[22] K. P. Bennett. Decision tree construction via linear programming. In M. Evans, editor, *Proceedings 4th Midwest Artificial Intelligence and Cognitive Science Society Conference*, pages 97–101, Utica, Illinois, 1992.

[23] M. Craven and J. Shavlik. Learning symbolic rules using artificial neural networks. In *Proceedings 10th International Conference on Machine Learning*, pages 73–80, Amherst, MA, 1993. Morgan Kaufmann.

[24] M. E. Thompson. NDCC: Normally distributed clustered datasets on cubes, 2006. www.cs.wisc.edu/dmi/svm/ndcc/.