

---

# Advice Refinement in Knowledge-Based Support Vector Machines

---

**Gautam Kunapuli**

KUNAPG@BIOSTAT.WISC.EDU

University of Wisconsin-Madison, 1300 University Avenue, Madison, WI 53705 USA

**Richard Maclin**

RMACLIN@D.UMN.EDU

University of Minnesota, Duluth, 1114 Kirby Drive, Duluth, MN 55812 USA

**Jude W. Shavlik**

SHAVLIK@CS.WISC.EDU

University of Wisconsin-Madison, 1300 University Avenue, Madison, WI 53705 USA

## Abstract

Knowledge-based support vector machines (KBSVMs) incorporate advice from experts, which can improve accuracy and generalization significantly. A major limitation occurs when the expert advice is noisy or incorrect which can lead to poorer models and decreased generalization. We propose a model that extends KBSVMs and learns not only from data and advice, but also simultaneously improves the advice. This model, which contains bilinear constraints for advice refinement, is effective for learning in domains with small data sets. We propose two approaches to handle the bilinearity of the formulation along with experimental results.

## 1. Introduction

For learning in complex environments, incorporating prior knowledge from experts can greatly improve generalization, often with many fewer labeled examples. Such approaches have been shown in rule learning methods, e.g., Pazzani & Kibler (1992); artificial neural networks (ANNs), e.g., Towell & Shavlik (1994); and more recently, in support vector machines, e.g., Fung et al. (2003a). One limitation of approaches to date concerns how well they adapt when the knowledge provided by the expert is inexact or partially correct. Many rule-learning methods implicitly focus on refining the given rules to learn better rules, while ANNs form the rules as portions of the network which could be refined. Further, ANN methods have been paired with rule extraction methods, e.g., Craven & Shavlik (1996), to try to understand the resulting learned network.

Here, we consider the framework of knowledge-based support vector machines (KBSVMs), which were first introduced by Fung et al. (2003a) for classification. KBSVMs have been further extended to incorporate kernels (Fung et al., 2003b), for kernel approximation (Mangasarian et al., 2004) and more recently, Kunapuli et al. (2010) derived an online version of KBSVMs. Extensive empirical results establish that expert advice can be effective, especially for applications such as breast-cancer diagnosis and tuberculosis spoligotype identification. KBSVMs are an effective and attractive methodology for knowledge discovery as they can use expert advice produce good models that generalize well *with a small amount of labeled data*.

Advice is usually provided by domain experts, typically as rules-of-thumb, based on the expert's accumulated experience in the domain and may not always be completely accurate. It is desirable that, rather than ignoring inaccurate rules or heavily penalizing approximate rules, the effectiveness of the advice can be *improved* by refining them. There are three reasons for this: first, improved rules result in the improvement of the overall generalization. Second, if the refinements to the advice are interpretable, it will help in the understanding of underlying phenomena for the experts. Finally, this is particularly appealing for applications where only a small number of labeled examples are available, and advice can be used in lieu of a possibly expensive labeling process.

To handle advice, KBSVMs minimize an objective function that contains three terms:

$$\text{model complexity} + \lambda \text{ data misfit} + \mu \text{ advice misfit.}$$

The parameter  $\mu$  trades-off regularization with *advice loss*, which is analogous to data loss and measures the closeness of fit to the advice (also see eq. 4). Furthermore, the KBSVM incorporates different rules into the hypothesis by taking linear combinations via *advice vectors*. Again, analogous to support vectors, only

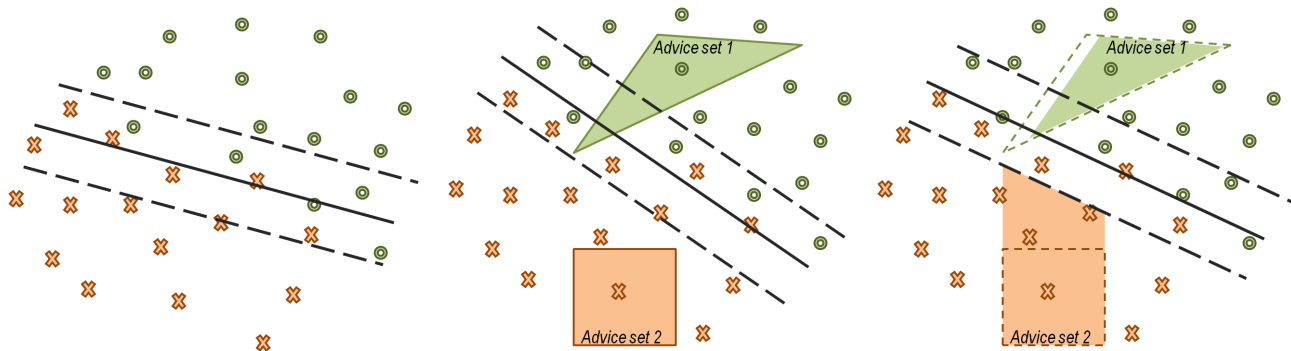


Figure 1. **(left)** Standard SVM, trades off between model complexity and error with respect to the data; **(center)** KBSVM, trades off between flatness and error with respect to the data *and* advice. A piece of advice set 1 extends over the margin, and is penalized as the advice error. No part of advice set 2 touches the margin and is ignored. **(right)** SVM that refines advice by truncating (advice set 1) or extending (advice set 2) advice so that the overall advice error is minimized.

some of the rules are incorporated and these are called *support constraints*. The rules that are not support constraints are ignored, typically when the cost of trying to fit them into the hypothesis becomes too great.

To see this, consider Figure 1. The figure on the left is the classical SVM, which trades off regularization with the data error. Figure 1 (center) illustrates standard KBSVMs (Fung et al., 2003a). Expert rules are specified as polyhedral *advice regions* in input space and introduce a bias to focus the learner on a model that also includes the advice:

$$\forall x \in \text{advice region } i, \text{ class}(x) = 1,$$

and similarly for  $\text{class}(x) = -1$ . In the KBSVM in Figure 1 (center), a piece of advice region 1 extends beyond the margin, and this is penalized as the *advice error*. Each advice region contributes to the final hypothesis in a KBSVM via its *advice vector*,  $\mathbf{u}^1$  and  $\mathbf{u}^2$  (see Section 2). As no part of advice set 2 touches the margin ( $\mathbf{u}^2 = 0$ ), none of its rules contribute anything to the final classifier and *are ignored*. As some of the rules in advice set 1 touch (in this case, intersect) the margin, ( $\mathbf{u}^1 \neq 0$ ), and the corresponding rules are called the *support constraints*, which are analogous to support vectors. Consequently, in the final classifier only uses advice set 1, with advice error. However, though the rules are inaccurate, they are able to improve the overall generalization compared to the SVM.

Now consider an SVM that is capable of refining inaccurate advice (Figure 1, right). Inaccurate advice (set 1) that intersects the hyperplane is truncated such that it minimizes the advice error with respect to the optimal classifier. Previously ignored advice (set 2) is extended, or refined to cover as much more of the input space as is feasible with respect to the optimal margin. Note that in the latter case, as advice region 2 is extended, it pushes the classifier up. The optimal classifier now has minimized the error with respect to

the data and the *refined* the advice. This allows it to further improve upon the performance of not just the SVM but also the KBSVM. This is the motivation behind this proposed approach.

Our approach generalizes the work of Maclin et al. (2007), to produce a model that corrects the polyhedral advice regions of KBSVMs. The resulting mathematical program is no longer a linear or quadratic program owing to *bilinear* correction factors in the constraints. We propose two algorithmic techniques to solve the resulting bilinear program, one based on successive linear programming (Maclin et al., 2007), and a concave-convex procedure (Yuille & Rangarajan, 2001). Before detailing the refinement algorithms, we briefly introduce KBSVMs.

## 2. Knowledge-Based SVMs

In KBSVMs, advice is specified in the form **IF antecedent THEN consequent**. The antecedents are convex, polyhedral regions in the input space of data. Advice can be specified about *every* potential data point in the input space that satisfies certain advice constraints. We illustrate with the following example.

Consider a classification task of learning to identify if a patient is at risk for diabetes, based on various features such as body mass index, blood glucose level, age etc. The National Institute for Health (NIH) web site on risks for Type-2 Diabetes<sup>1</sup> provides the following guidelines to establish risk for diabetes, according to which, a person who is obese, characterized by high body mass index ( $\text{BMI} \geq 30$ ) and high `bloodglucose` level ( $\geq 126$ ) is at strong risk for diabetes, while a person who is at normal weight ( $\text{BMI} \leq 25$ ) and low `bloodglucose` level ( $\leq 100$ ) is unlikely to have diabetes. As `BMI` and `bloodglucose` are

<sup>1</sup><http://diabetes.niddk.nih.gov/DM/pubs/~riskfortype2>

features of the data set, we can give advice by combining these conditions into conjunctive rules, one for each class:

$$\begin{aligned} (\text{BMI} \leq 25) \wedge (\text{bloodglucose} \leq 100) &\Rightarrow \neg \text{diabetes} \\ (\text{BMI} \geq 30) \wedge (\text{bloodglucose} \geq 126) &\Rightarrow \text{diabetes} \end{aligned} \quad (1)$$

In general, rules such as the ones above define a polyhedral region of the input space and are expressed as the implication

$$D\mathbf{x} \leq \mathbf{d} \Rightarrow z(\mathbf{w}'\mathbf{x} - b) \geq 1, \quad (2)$$

where  $z = +1$  indicates that all points  $\mathbf{x}$  that satisfy the constraints  $D\mathbf{x} \leq \mathbf{d}$  (i.e., lie in the polyhedral region) belong to class  $+1$ , while  $z = -1$  indicates the same for the other class.

KBSVMs learn a linear classifier ( $\mathbf{w}'\mathbf{x} = b$ ) given data  $(\mathbf{x}^i, y_i)_{i=1}^{\ell}$  with  $\mathbf{x}^i \in \mathbb{R}^n$  and labels  $y_i \in \{\pm 1\}$ . The data points are collected row-wise in the matrix  $X \in \mathbb{R}^{\ell \times n}$ , and  $Y = \text{diag}(y)$ . We assume that  $m$  advice sets  $(D_i, \mathbf{d}^i, z_i)_{i=1}^m$  are given in addition to the data, and if the  $i$ -th advice set has  $k_i$  constraints, we have  $D_i \in \mathbb{R}^{k_i \times n}$ ,  $\mathbf{d}^i \in \mathbb{R}^{k_i}$  and  $z_i = \{\pm 1\}$ . The absolute value of a scalar  $y$  is denoted  $|y|$ , the 1-norm of a vector  $\mathbf{x}$  is denoted  $\|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i|$ , and the *entrywise* 1-norm of a  $m \times n$  matrix  $A$  is denoted  $\|A\|_1 = \sum_{i=1}^m \sum_{j=1}^n |A_{ij}|$ . Finally,  $\mathbf{e}$  is a vector of ones of appropriate dimension.

The linear SVM formulation for classification optimizes *model complexity*  $+$   $\lambda$  *training error*:

$$\begin{aligned} \min_{\mathbf{w}, b, (\xi \geq 0)} \quad & \|\mathbf{w}\|_1 + \lambda \mathbf{e}'\xi \\ \text{s.t.} \quad & Y(X\mathbf{w} - \mathbf{e}b) + \xi \geq \mathbf{e}. \end{aligned} \quad (3)$$

The implication (2), for the  $i$ -th advice set, can be incorporated into (3) using the nonhomogeneous Farkas theorem of the alternative (Fung et al., 2003a) that introduces advice vectors  $\mathbf{u}^i$ . The advice vectors perform the same role as the dual multipliers  $\alpha$  in the classical SVM. Recall that points with non-zero  $\alpha$ 's are the *support vectors* which additively contribute to  $\mathbf{w}$ . Here, for *each* advice set, the constraints of the set which have non-zero  $\mathbf{u}^i$ 's are called *support constraints*.

The resulting formulation is the KBSVM:

$$\begin{aligned} \min_{\mathbf{w}, b, (\xi, \mathbf{u}^i, \boldsymbol{\eta}^i, \zeta_i \geq 0)} \quad & \|\mathbf{w}\|_1 + \lambda \mathbf{e}'\xi + \mu \sum_{i=1}^m (\mathbf{e}'\boldsymbol{\eta}^i + \zeta_i) \\ \text{s.t.} \quad & Y(X\mathbf{w} - \mathbf{e}b) + \xi \geq \mathbf{e}, \\ & -\boldsymbol{\eta}^i \leq D_i' \mathbf{u}^i + z_i \mathbf{w} \leq \boldsymbol{\eta}^i, \\ & -\mathbf{d}^i' \mathbf{u}^i - z_i b + \zeta_i \geq 1, \quad i = 1, \dots, m. \end{aligned} \quad (4)$$

In the case of noisy advice, the advice errors  $\boldsymbol{\eta}^i$  and  $\zeta_i$  soften the advice constraints analogous to the data errors  $\xi$ . Returning to Figure 1, for advice set 1,  $\boldsymbol{\eta}^1$

and  $\zeta_2$  are non-zero, while for advice set 2,  $\mathbf{u}^2 = 0$ . The influence of data and advice is determined by the choice of the parameters  $\lambda$  and  $\mu$  which reflect the user's trust in the data and advice respectively.

### 3. Advice-Refining KBSVMs

Previously, Maclin et al. (2007) formulated a model to refine advice in KBSVMs. However, in their model, only the terms  $\mathbf{d}^i$  are refined, that is, they only attempt to refine each rule such that

$$D_i \mathbf{x} \leq (\mathbf{d}^i - \mathbf{f}^i) \Rightarrow z_i(\mathbf{w}'\mathbf{x} - b) \geq 0, \quad i = 1, \dots, m. \quad (5)$$

The resulting formulation adds the refinement terms into the KBSVM model (4) in the advice constraints, as well as in the objective. The latter allows for the overall extent of the refinement to be controlled by the *refinement parameter*  $\nu > 0$ . This formulation was called Refining-Rules Support Vector Machine (RRSVM):

$$\begin{aligned} \min_{\substack{\mathbf{w}, b, \mathbf{f}^i, \\ (\xi, \mathbf{u}^i, \boldsymbol{\eta}^i, \zeta_i \geq 0)}} \quad & \|\mathbf{w}\|_1 + \lambda \mathbf{e}'\xi + \mu \sum_{i=1}^m (\mathbf{e}'\boldsymbol{\eta}^i + \zeta_i) \\ & + \nu \sum_{i=1}^m \|\mathbf{f}^i\|_1 \\ \text{s.t.} \quad & Y(X\mathbf{w} - \mathbf{e}b) + \xi \geq \mathbf{e}, \\ & -\boldsymbol{\eta}^i \leq D_i' \mathbf{u}^i + z_i \mathbf{w} \leq \boldsymbol{\eta}^i, \\ & -(\mathbf{d}^i - \mathbf{f}^i)' \mathbf{u}^i - z_i b + \zeta_i \geq 1, \quad i = 1, \dots, m. \end{aligned} \quad (6)$$

The RRSVM is no longer an LP owing to bilinear terms  $\mathbf{f}^i' \mathbf{u}^i$ . Here, we consider a full generalization of RRSVMs, which allows for much more flexibility in refining the advice based on the data, while still retaining interpretability of the resulting refined advice:

$$(D_i - F_i) \mathbf{x} \leq (\mathbf{d}^i - \mathbf{f}^i) \Rightarrow z_i(\mathbf{w}'\mathbf{x} - b) \geq 0, \quad i = 1, \dots, m. \quad (7)$$

The formulation (6) now changes to include the additional refinement terms  $F_i$ :

$$\begin{aligned} \min_{\substack{\mathbf{w}, b, F_i, \mathbf{f}^i, \\ (\xi, \mathbf{u}^i, \boldsymbol{\eta}^i, \zeta_i \geq 0)}} \quad & \|\mathbf{w}\|_1 + \lambda \mathbf{e}'\xi + \mu \sum_{i=1}^m (\mathbf{e}'\boldsymbol{\eta}^i + \zeta_i) \\ & + \nu \sum_{i=1}^m (\|F_i\|_1 + \|\mathbf{f}^i\|_1) \\ \text{s.t.} \quad & Y(X\mathbf{w} - \mathbf{e}b) + \xi \geq \mathbf{e}, \\ & -\boldsymbol{\eta}^i \leq (D_i - F_i)' \mathbf{u}^i + z_i \mathbf{w} \leq \boldsymbol{\eta}^i, \\ & -(\mathbf{d}^i - \mathbf{f}^i)' \mathbf{u}^i - z_i b + \zeta_i \geq 1, \quad i = 1, \dots, m. \end{aligned} \quad (8)$$

The objective function of (8) trades-off the effect of refinement in each of the advice sets via the *refinement parameter*  $\nu$ . This is the Advice-Refining KBSVM (arkSVM); it improves upon the work of Maclin et al. in two important ways. First, refining  $\mathbf{d}$  alone is highly restrictive as it allows only for the *translation* of the boundaries of the polyhedral advice; the generalized refinement offered by arkSVMs allows for much more flexibility owing to the fact that the boundaries

**Algorithm 1** arkSVM via SLP (arkSVM-slp)

- 1: **initialize:**  $\hat{F}_i^1 = 0, \hat{\mathbf{f}}^{i,1} = 0$
- 2: **while**  $\sum_j (\|F_j^t - F_j^{t+1}\| + \|\mathbf{f}_j^t - \mathbf{f}_j^{t+1}\|) > \epsilon$  **do**
- 3: (**estimation step**) solve for  $\{\hat{\mathbf{u}}^{i,t+1}\}_{i=1}^m$

$$\begin{aligned} \min_{\substack{\mathbf{w}, b, (\boldsymbol{\xi}, \boldsymbol{\eta}^i, \zeta_i \geq 0) \\ \mathbf{u}^i \geq 0 \\ \text{s.t.}}} \quad & \|\mathbf{w}\|_1 + \lambda \mathbf{e}' \boldsymbol{\xi} + \mu \sum_{i=1}^m (\mathbf{e}' \boldsymbol{\eta}^i + \zeta_i) \\ & Y(X\mathbf{w} - b\mathbf{e}) + \boldsymbol{\xi} \geq \mathbf{e}, \\ & -\boldsymbol{\eta}^i \leq (D_i - \hat{F}_i^t)' \mathbf{u}^i + z_i \mathbf{w} \leq \boldsymbol{\eta}^i, \\ & -(\mathbf{d}^i - \hat{\mathbf{f}}^{i,t})' \mathbf{u}^i - z_i b + \zeta_i \geq 1, \\ & i = 1, \dots, m. \end{aligned}$$

- 4: (**refinement step**) solve for  $(\hat{F}_i^{t+1}, \hat{\mathbf{f}}^{i,t+1})_{i=1}^m$

$$\begin{aligned} \min_{\substack{F_i, \mathbf{f}^i, \\ \mathbf{w}, b, (\boldsymbol{\xi}, \boldsymbol{\eta}^i, \zeta_i \geq 0) \\ \text{s.t.}}} \quad & \|\mathbf{w}\|_1 + \lambda \mathbf{e}' \boldsymbol{\xi} + \mu \sum_{i=1}^m (\mathbf{e}' \boldsymbol{\eta}^i + \zeta_i) \\ & + \nu \sum_{i=1}^m (\|F_i\|_1 + \|\mathbf{f}^i\|_1) \\ & Y(X\mathbf{w} - b\mathbf{e}) + \boldsymbol{\xi} \geq \mathbf{e}, \\ & -\boldsymbol{\eta}^i \leq (D_i - F_i)' \hat{\mathbf{u}}^{i,t+1} + z_i \mathbf{w} \leq \boldsymbol{\eta}^i, \\ & -(\mathbf{d}^i - \mathbf{f}^i)' \hat{\mathbf{u}}^{i,t+1} - z_i b + \zeta_i \geq 1, \\ & i = 1, \dots, m. \end{aligned}$$

- 5: **end while**

of the advice can be translated *and rotated* (see Figure 2). Second, the newly added refinement terms,  $F_i' \mathbf{u}^i$ , are bilinear also, and do not make the overall problem more complex; in addition to the successive linear programming approach of Maclin et al. (2007), we also propose a concave-convex procedure that leads to an approach based on successive quadratic programming. We provide details of both approaches next.

### 3.1. Successive Linear Programming (SLP)

One approach is to solve a sequence of LPs while alternately fixing the bilinear variables. This approach has been used to solve various machine learning formulations, for instance Bennett & Bredensteiner (1997), and also used to solve RRSVMs. We solve the LPs arising from alternately fixing the sources of bilinearity:  $(F_i, \mathbf{f}^i)_{i=1}^m$  and  $\{\mathbf{u}^i\}_{i=1}^m$ .

- (**Estimation Step**) When  $(\hat{F}_i^t, \hat{\mathbf{f}}^{i,t})_{i=1}^m$  are fixed we solve a KBSVM to find a data-estimate of the advice vectors with respect to the current refinement of advice:  $(D_j - \hat{F}_j^t) \mathbf{x} \leq (\mathbf{d}^j - \hat{\mathbf{f}}^{j,t})$ .
- (**Refinement Step**) When  $\{\hat{\mathbf{u}}^{i,t}\}_{i=1}^m$  are fixed, the resulting LP attempts to further refine the advice regions based on estimates from data computed in the previous step.

Algorithm 1 describes the above approach.

### 3.2. Successive Quadratic Programming

**Algorithm 2** arkSVM via SQP (arkSVM-sqp)

- 1: **initialize:**  $\hat{F}_i^1 = 0, \hat{\mathbf{f}}^{i,1} = 0$
- 2: **while**  $\sum_j (\|F_j^t - F_j^{t+1}\| + \|\mathbf{f}_j^t - \mathbf{f}_j^{t+1}\|) > \epsilon$  **do**
- 3: solve for  $\{\hat{\mathbf{u}}^{i,t+1}\}_{i=1}^m$

$$\begin{aligned} \min_{\substack{F_i, \mathbf{f}^i, (\mathbf{u}^i \geq 0) \\ \mathbf{w}, b, (\boldsymbol{\xi}, \boldsymbol{\eta}^i, \zeta_i \geq 0) \\ \text{s.t.}}} \quad & \|\mathbf{w}\|_1 + \lambda \mathbf{e}' \boldsymbol{\xi} + \mu \sum_{i=1}^m (\mathbf{e}' \boldsymbol{\eta}^i + \zeta_i) \\ & + \nu \sum_{i=1}^m (\|F_i\|_1 + \|\mathbf{f}^i\|_1) \\ & Y(X\mathbf{w} - b\mathbf{e}) + \boldsymbol{\xi} \geq \mathbf{e}, \\ & \text{eqns (10–12)} \\ & i = 1, \dots, m, j = 1, \dots, n \end{aligned}$$

- 4: **end while**

Denote the  $j$ -th components of  $\mathbf{w}$  and  $\boldsymbol{\eta}^i$  to be  $w_j$  and  $\eta_j^i$  respectively. Now, consider the  $j$ -th constraint of (7), which can be rewritten as  $D_{ij}' \mathbf{u}^i + z_i w_j - \eta_j^i - F_{ij}' \mathbf{u}^i \leq 0$ . A general bilinear term  $r's$ , which is non-convex, can be written as the difference of two convex terms:  $\frac{1}{4}\|r+s\|^2 - \frac{1}{4}\|r-s\|^2$ , and we have

$$D_{ij}' \mathbf{u}^i + z_i w_j - \eta_j^i + \frac{1}{4}\|F_{ij} - \mathbf{u}^i\|^2 \leq \frac{1}{4}\|F_{ij} + \mathbf{u}^i\|^2, \quad (9)$$

and both sides of the constraint above are now convex and quadratic. We now linearize the right-hand side of (9) around some current estimate of the bilinear variables  $(\hat{F}_{ij}^t, \hat{\mathbf{u}}^{i,t})$ . This produces the following constraint

$$\begin{aligned} D_{ij}' \mathbf{u}^i + z_i w_j - \eta_j^i + \frac{1}{4}\|F_{ij} - \mathbf{u}^i\|^2 &\leq \frac{1}{4}\|\hat{F}_{ij}^t + \hat{\mathbf{u}}^{i,t}\|^2 \\ &+ \frac{1}{2}(\hat{F}_{ij}^t + \hat{\mathbf{u}}^{i,t})' \left( (F_{ij} - \hat{F}_{ij}^t) + (\mathbf{u}^i - \hat{\mathbf{u}}^{i,t}) \right). \end{aligned} \quad (10)$$

Similarly, the constraint  $-(D_{ij} - F_{ij})' \mathbf{u}^i - z_i w_j - \eta_j^i \leq 0$ , can be replaced by

$$\begin{aligned} -D_{ij}' \mathbf{u}^i - z_i w_j - \eta_j^i + \frac{1}{4}\|F_{ij} + \mathbf{u}^i\|^2 &\leq \frac{1}{4}\|\hat{F}_{ij}^t - \hat{\mathbf{u}}^{i,t}\|^2 \\ &+ \frac{1}{2}(\hat{F}_{ij}^t - \hat{\mathbf{u}}^{i,t})' \left( (F_{ij} - \hat{F}_{ij}^t) - (\mathbf{u}^i - \hat{\mathbf{u}}^{i,t}) \right), \end{aligned} \quad (11)$$

while  $\mathbf{d}^{i'} \mathbf{u}^i + z_i b + 1 - \zeta_i - \mathbf{f}^{i'} \mathbf{u}^i \leq 0$  is replaced by

$$\begin{aligned} \mathbf{d}^{i'} \mathbf{u}^i + z_i b + 1 - \zeta_i + \frac{1}{4}\|\mathbf{f}^i - \mathbf{u}^i\|^2 &\leq \frac{1}{4}\|\hat{\mathbf{f}}^{i,t} + \hat{\mathbf{u}}^{i,t}\|^2 \\ &+ \frac{1}{2}(\hat{\mathbf{f}}^{i,t} + \hat{\mathbf{u}}^{i,t})' \left( (\mathbf{f}^i - \hat{\mathbf{f}}^{i,t}) + (\mathbf{u}^i - \hat{\mathbf{u}}^{i,t}) \right), \end{aligned} \quad (12)$$

for  $i = 1, \dots, m$  and  $j = 1, \dots, n$ . The right-hand sides in (10–12) are now affine. Replacing the original bilinear constraints of (8) with the convexified relaxations results in a quadratically-constrained linear program (QCLP), which we call the *restricted problem*. Now, we can iteratively solve the resulting QCLP. At the  $t$ -th iteration, the restricted problem uses the current estimate to construct a new feasible point and

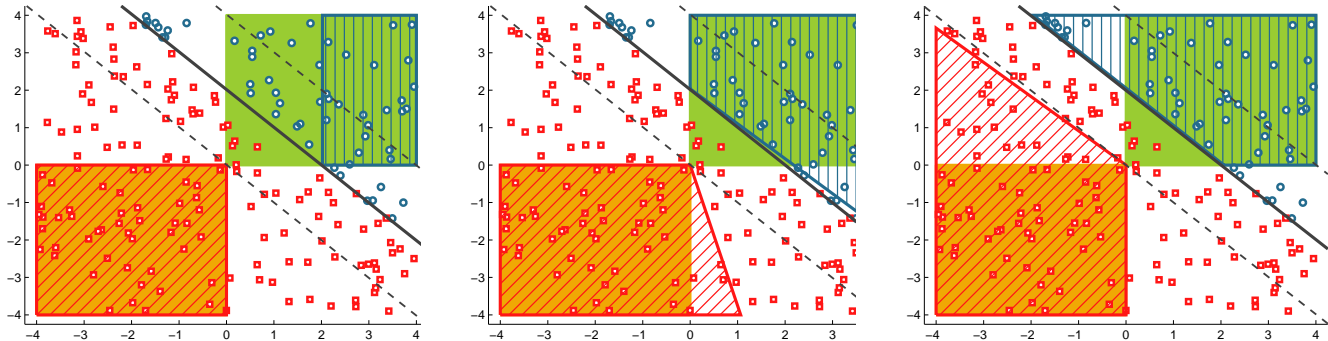


Figure 2. Advice refinement for the polyhedral regions of the toy data set described in Section 4.1 using (left) RRSVM (center) arkSVM-sla (right) arkSVM-sqp. The orange and green unhatched regions show the originally supplied advice. The dark line shows the learned model and the dashed lines, the margin. For each method, we show the refined advice after learning (vertically hatched for Class +1, and diagonally hatched for Class -1).

iterating this procedure produces a sequence of feasible points with decreasing objective values. The full algorithm is shown in Algorithm 2.

This approach is essentially the constrained concave-convex procedure described in the context of machine learning approaches by Yuille & Rangarajan (2001), and Smola & Vishwanathan (2005), who also derived conditions under which the algorithm converges to a local solution.

## 4. Experiments

We present the results of several experiments that compare the performance of three algorithms: RRSVMs, arkSVM-SLP and arkSVM-sqp. The LPs were solved using QSOPT<sup>2</sup>, while the QCLPs were solved using SDPT-3 (Tütüncü et al., 2003).

### 4.1. Synthetic Experiment

In this experiment, we consider a simple two-dimensional data set with shown in Figure 2. The data set consists of 200 points separated by the hyperplane  $x_1 + x_2 = 2$ . There are two advice sets, which essentially span the the positive and negative half quadrants of  $\mathbb{R}^2$   $\{S_1 : (x_1, x_2) \geq 0 \Rightarrow \text{class} = +1\}$ ,  $\{S_2 : (x_1, x_2) \leq 0 \Rightarrow \text{class} = -1\}$ . The final  $(\mathbf{w}, b)$  as well as the margin  $\|\mathbf{w}\|_\infty$  are shown. Both of the arkSVMs are able to refine the knowledge sets such that the no part of  $S_1$  lies on the wrong side of the final hyperplane. In addition, the refinement terms allow for sufficient modification (extension) of the advice sets  $D\mathbf{x} \leq \mathbf{d}$  so that they fill the input space as much as possible, without violating the margin. Comparing to the RRSVM, we see that refinement is very restrictive owing to corrections applied only to the right-hand-side terms of the polyhedral advice sets, rather than fully correcting the advice.

<sup>2</sup><http://www2.isye.gatech.edu/~wcook/qsopt/>

### 4.2. USPS Data Sets

The well-known USPS data consists of 9298 data points described by  $16 \times 16$  pixel greyscale images to represent handwritten digit images. We investigated 1-vs-rest classification for each of the 10 digits. The digit data sets were rescaled to  $8 \times 8$ . In prior experiments using this data set (Keyser et al., 2000), it was observed that the performance of various classifiers improved statistically significantly when the data set was augmented with up to 2,400 machine-printed digits. We exploit the ability of KBSVMs to accept advice that covers regions of input space rather than providing examples directly.

To create rules for each digit, rather than augmenting the data sets, we hand drew ten images for each of the digits (more specifically, our view of what the canonical digits should look like) and then performed a blurring of the images together using principal components analysis. We took the principal components of the resulting image and used them as templates for rules based on matching expected and observed pixels. This is done by computing thresholds at which the digit’s pixels that are on are exceeded for the correct digit and not, for all others (see Figure 3). While the advice provided thus for each digit is reasonable, it is far from perfect and can be refined.

For each of the 10 digits, 10 data sets with a very small amount of labeled data (100 examples) were randomly drawn, with positive examples being the digit, while the negative examples being randomly and uniformly drawn from all other digits. The rest of the examples are held out for testing. The parameters for the methods were selected using 10-fold cross validation and the test errors for each of the 10 tasks is shown in Figure 4. It is clear that for most of the classification tasks, either one of the arkSVM approaches is able to improve the test-set error compared to KBSVMs or standard SVMs. Furthermore, compared to

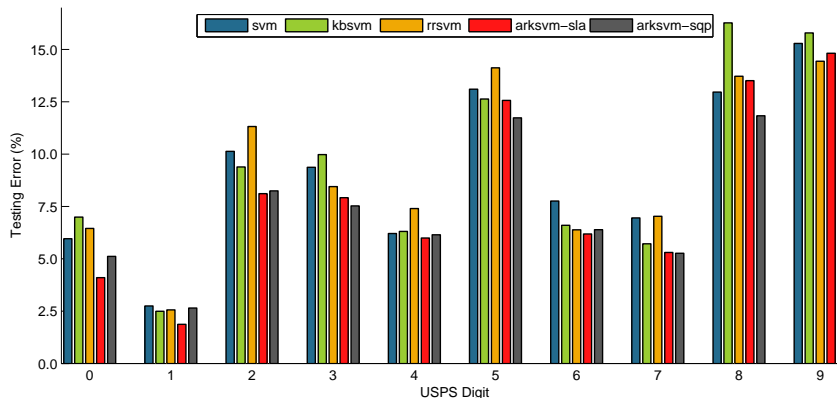


Figure 4. Results for the USPS data set for each digit vs the rest averaged over 10 runs.

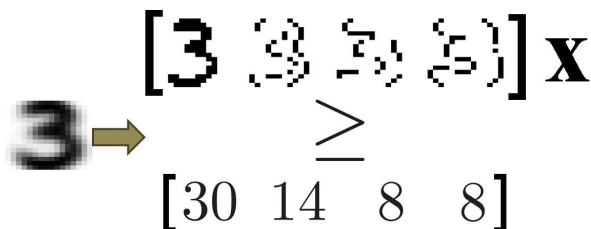


Figure 3. Advice in the form  $Dx \leq \mathbf{d}$  from principal components of hand-written digits (left). The advice templates for all possible digits  $\mathbf{x} \in \mathbb{R}^{64 \times 1}$  are the rows of  $D$  while the thresholds,  $\mathbf{d}$ , are the number of bits that should be on for each digit.

RRSVMs, which only refine thresholds of the advice, arkSVMs perform significantly better for every task as they are able to refine the advice comprehensively.

### 5. Conclusions and Future Work

In this work, we have presented two knowledge-discovery methods: arkSVM-sla and arkSVM-sqp that allow support vector methods to not only make use of advice provided by human experts but to refine that advice using labeled data to improve the advice. These methods are an advance over previous knowledge-based SVM methods which either did not refine advice (Fung et al., 2003a) or could only refine simple aspects of the advice (Maclin et al., 2007).

Experimentally, the resulting methods outperform SVMs, KBSVMs and RRSVMs. This allows the approaches to learn in domains using only a small amount of labeled data and imperfect advice.

### Acknowledgements

The authors gratefully acknowledge support of the Defense Advanced Research Projects Agency under DARPA grant FA8650-06-C-7606. Views and conclusions contained in this document are those of the authors and do not necessarily represent the official opinion or policies, either expressed or implied of the US government or of DARPA.

### References

Bennett, K. P. and Bredensteiner, E. J. A parametric optimization method for machine learning. *INFORMS Journ. on Comp.*, 9(3):311–318, 1997.

Craven, M. W. and Shavlik, J. W. Extracting tree-structured representations of trained networks. In *NIPS*, volume 8, pp. 24–30, 1996.

Fung, G., Mangasarian, O. L., and Shavlik, J. W. Knowledge-based support vector classifiers. In *NIPS*, volume 15, pp. 521–528, 2003a.

Fung, G., Mangasarian, O. L., and Shavlik, J. W. Knowledge-based nonlinear kernel classifiers. In *COLT*, pp. 102–113, 2003b.

Keyser, D., Dahmen, J., and Ney, H. A probabilistic view on tangent distance. In *Mustererkennung 2000, 22. DAGM-Symposium*, pp. 107–114, 2000.

Kunapuli, G., Bennett, K. P., Shabbeer, A., Maclin, R., and Shavlik, J. W. Online knowledge-based support vector machines. In *ECML*, pp. 145–161, 2010.

Maclin, R., Wild, E. W., Shavlik, J. W., Torrey, L., and Walker, T. Refining rules incorporated into knowledge-based support vector learners via successive linear programming. In *AAAI*, pp. 584–589, 2007.

Mangasarian, O. L., Shavlik, J. W., and Wild, E. W. Knowledge-based kernel approximation. *JMLR*, 5:1127–1141, 2004.

Pazzani, M. and Kibler, D. The utility of knowledge in inductive learning. *Mach. Learn.*, 9:57–94, 1992.

Smola, A. J. and Vishwanathan, S. V. N. Kernel methods for missing variables. In *AISStats*, pp. 325–332, 2005.

Towell, G. G. and Shavlik, J. W. Knowledge-based artificial neural networks. *AIJ*, 70(1–2):119–165, 1994.

Tütüncü, R. H., Toh, K. C., and Todd, Michael J. Solving semidefinite-quadratic-linear programs using sdpt3. *Math. Prog.*, 95(2), 2003.

Yuille, A. L. and Rangarajan, A. The concave-convex procedure (CCCP). In *NIPS*, volume 13, 2001.