JMLR: Workshop and Conference Proceedings 39:32–47, 2014 ACML 2014

Support vector machines with indefinite kernels

Ibrahim Alabdulmohsin IBRAHIM.ALABDULMOHSIN@KAUST.EDU.SA
Xin Gao XIN.GAO@KAUST.EDU.SA
Xiangliang Zhang XIANGLIANG.ZHANG@KAUST.EDU.SA
Computer, Electrical, and Mathematical Sciences & Engineering (CEMSE) Divison, King Abdullah University of Science & Technology (KAUST), Thuwal 23955, Saudi Arabia.

Editor: Dinh Phung and Hang Li

Abstract

Training support vector machines (SVM) with indefinite kernels has recently attracted attention in the machine learning community. This is partly due to the fact that many similarity functions that arise in practice are not symmetric positive semidefinite, i.e. the Mercer condition is not satisfied, or the Mercer condition is difficult to verify. Previous work on training SVM with indefinite kernels has generally fallen into three categories: (1) positive semidefinite kernel approximation, (2) non-convex optimization, and (3) learning in Krein spaces. All approaches are not fully satisfactory. They have either introduced sources of inconsistency in handling training and test examples using kernel approximation, settled for approximate local minimum solutions using non-convex optimization, or produced non-sparse solutions. In this paper, we establish both theoretically and experimentally that the 1-norm SVM, proposed more than 10 years ago for embedded feature selection, is a better solution for extending SVM to indefinite kernels. More specifically, 1-norm SVM can be interpreted as a structural risk minimization method that seeks a decision boundary with large *similarity margin* in the original space. It uses a linear programming formulation that remains convex even if the kernel matrix is indefinite, and hence can always be solved quite efficiently. Also, it uses the indefinite similarity function (or distance) directly without any transformation, and, hence, it always treats both training and test examples consistently. Finally, it achieves the highest accuracy among all methods that train SVM with indefinite kernels with a statistically significant evidence while also retaining sparsity of the support vector set.

Keywords: Support vector machines, Indefinite kernels, Similarity-based classification, Supervised learning, Linear programming

1. Introduction

Support vector machines (SVM) is one of the most popular classification algorithms today. It is inspired by deep theoretical foundations, which make use of the Vapnik-Chervonenkis (VC) dimension to establish the generalization ability of such family of classifiers (Vapnik, 1999; Burges, 1998). However, SVM has its limitations, which motivated development of numerous variants including the Distance Weighted Discrimination algorithm (DWD) to deal with the "data piling" phenomenon observed in large dimensions (Marron et al., 2007) and second order cone programming (SOCP) techniques for handling uncertain or missing values assuming availability of second order moments of data (Shivaswamy et al., 2006).

One fundamental limiting factor in SVM is the need for positive semidefinite (PSD) kernels. This follows from the fact that SVM is usually solved in its dual form:

$$\begin{aligned} \underset{\alpha}{\text{minimize}} \quad & \frac{1}{2}\,\alpha^T\, Y\, K\, Y\, \alpha - \mathbf{1}^T\,\alpha \\ \text{subject to} \quad & 0 \le \alpha \le C\,\mathbf{1}, \quad y^T\,\alpha = 0 \end{aligned} \tag{1}$$

Here, $Y = \text{diag}(y)$, where $y \in \{-1, +1\}^m$ is a vector of m class labels, while C is a tradeoff constant. In the dual-form in Eq 1, the kernel matrix K has to be symmetric positive semidefinite, i.e. satisfies the Mercer condition, in order to guarantee convexity of the optimization problem and the existence of a reproducing Hilbert kernel space (RHKS).

In real-life applications, however, many similarity functions exist that are either indefinite or for which the Mercer condition is difficult to verify. For example, one can incorporate the longest common subsequence in defining distance between genetic sequences, use BLAST similarity score between protein sequences, use set operations such as union/intersection in defining similarity between transactions, use human-judged similarities between concepts and words, use the symmetrized Kullback-Leibler divergence between probability distributions, use dynamic time warping for time series, or use the tangent distance and shape matching distance in computer vision (Chen et al., 2009a; Wu et al., 2005; Ying et al., 2009; Haasdonk, 2005). Extending SVM to indefinite kernels will greatly expand its applicability.

Recent work on training SVM with indefinite kernels has generally fallen into three categories: (1) positive semidefinite (PSD) kernel approximation, (2) non-convex optimization, and (3) learning in Krein spaces. In the PSD kernel approximation approach, the kernel matrix of training examples is altered so that it becomes PSD. One example is the *denoise* method, which sets all negative eigenvalues to zero. The motivation behind such approach is to assume that negative eigenvalues are caused by noise (Pekalska et al., 2001). A second example is the *flip* method, which flips sign of the negative eigenvalues. This method aims at retaining some of the information coded in those negative eigenvalues (Pekalska et al., 2001; Graepel et al., 1999). A third example is to formulate a max-min optimization problem that both seeks support vectors as well as a PSD kernel that approximates the indefinite similarity matrix. The latter approach was introduced by Luss and d'Aspremont in 2007 with improvements in training time reported in the following years (Chen and Ye, 2008; Luss and dAspremont, 2009; Chen et al., 2009b).

All the kernel approximation methods above guarantee that the optimization problem remains convex during training. During testing, however, the original indefinite kernel function is used. Hence, training and test examples are treated inconsistently. In addition, such methods are only useful when the similarity matrix is approximable by a PSD matrix. For other similarity functions such as the sigmoid kernel that can occasionally yield a *negative semidefinite* matrix for certain values of its hyperparameters, the kernel approximation approach cannot be utilized.

In the second approach, non-convex optimization methods are used. For example, SMO-type decomposition might be used in finding a local minimum with indefinite similarity functions (Lin and Lin, 2003). Haasdonk interprets this as a method of minimizing the distance between reduced convex hulls in a pseudo-Euclidean space (Haasdonk, 2005). However, because such approach can terminate at a local minimum, it does not guarantee learning

(Chen et al., 2009a). Similar to the previous approach, this method only works well if the similarity matrix is approximately PSD.

The third approach that has been proposed in the literature is to extend SVM into Krein spaces, in which a reproducing kernel is decomposed into the sum of one positive semidefinite kernel and one negative semidefinite kernel (Ong et al., 2004; Loosli et al., 2013). Instead of minimizing regularized risk, the objective function is now *stabilized*. One fairly recent algorithm that has been proposed to solve the stabilization problem is called eigen-decomposition SVM (ESVM) (Loosli et al., 2013). While this algorithm has been shown to outperform all previous methods, its primary drawback is that it does not produce sparse solutions, hence the entire list of training examples are often needed during prediction.

The main contribution of this paper is to establish both theoretically and experimentally that the 1-norm SVM (Zhu et al., 2004), which was proposed more than 10 years ago, is a better solution for extending SVM to indefinite kernels. More specifically, 1-norm SVM can be interpreted as a structural risk minimization method that seeks a decision boundary with large *similarity margin* in the original space. It uses a linear programming (LP) formulation that remains convex even if the kernel matrix is indefinite, and hence can always be solved quite efficiently. It uses the indefinite similarity function (or distance) directly without any transformation, and, hence, it always treats both training and test examples consistently. In addition, it achieves the highest accuracy among all methods that train SVM with indefinite kernels, with a statistically significant evidence, while also retaining sparsity of the support vector set. Further details are provided in Section 3.

In the literature, 1-norm SVM is often used as an *embedded* feature selection method, where learning and feature selection are performed simultaneously (Bradley and Mangasarian, 1998; Zhu et al., 2004; Fung and Mangasarian, 2004; Zou, 2007; Hilario and Kalousis, 2008; Liu et al., 2010). It was studied in (Zhu et al., 2004), where it was argued that 1-norm SVM has an advantage over standard 2-norm SVM when there are redundant noise features. To the knowledge of the authors, the advantage of using 1-norm SVM in handling indefinite kernels has never been established in the literature.

The rest of the paper is structured as follows. First, we describe the 1-norm SVM and how it can be adapted to handle binary classification with indefinite kernels. We provide two motivations behind its formulation; namely that it can be interpreted as a method of finding a decision boundary with a large similarity margin and that it can be interpreted as a method of structural risk minimization. After that, we present experimental results using both synthetic and real datasets, which validate the advantage of using 1-norm SVM in handling indefinite kernels over all other methods.

2. Material and Methods

Given a training set of m examples $\{(x_i, y_i)\}_{i=1,\ldots,m}$, where $x_i \in \mathcal{X}$ and $y_i \in \{-1, +1\}$, we would like to use the training set to infer a classifier of the form $\hat{y}(x) = \mathbf{sign}(f(x))$ for some function $f(\cdot) : \mathcal{X} \to \mathbb{R}$. The standard SVM formulation uses the decision rule obtained by

the Representer Theorem (Schölkopf and Smola, 2002):

$$f(x) = b + \sum_{i=1}^{m} y_i \, \alpha_i \, K(x_i, \, x), \tag{2}$$

for some offset $b \in \mathbb{R}$ and a reproducing kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Here, b and α_i are the solutions to the optimization problem in (1). The original formulation of 1-norm support vector machines (Zhu et al., 2004), on the other hand, uses a dictionary of basis functions $\mathcal{D} = \{h_1(\cdot), \, h_2(\cdot), \, \ldots\}$, where $h_j(\cdot) : \mathcal{X} \to \mathbb{R}$, and considers classification using:

$$f(x) = b + \sum_{j} \lambda_j \cdot h_j(x) \tag{3}$$

In the above expression for $f(\cdot)$, the basis functions $h_j(\cdot)$ are fixed and the only variables to be optimized are b and λ_j. Zhu et al. (2004) proposed the following optimization problem for finding the weights λ (Eq 5 in (Zhu et al., 2004))[1]:

$$\begin{aligned} \underset{\lambda, \, \xi, \, b}{\text{minimize}} \quad & \sum_{j} |\lambda_j| + C \, \sum_{i=1}^{m} \xi_i \\ \text{subject to} \quad & y_i \cdot \big(b + \sum_{j} \lambda_j \cdot h_j(x_i)\big) \geq 1 - \xi_i \\ & \xi_i \; \geq 0, \qquad \text{for all } i = 1, \, 2, \ldots, m \end{aligned}$$

Here, C is a tradeoff parameter between regularization and fitting. We will provide several motivations behind such formulation shortly. To utilize the above method in handling indefinite kernels, we set $h_j(\cdot) = y_j \, S(x_j, \, \cdot)$, where $S(x_j, \, \cdot) : \mathcal{X} \to \mathbb{R}$ measures similarity to example x_j. In addition, we impose the non-negativity constraint $\lambda_j \geq 0$ to ensure that any example x_j can be representative to its own class y_j only. This gives us the following linear program (LP):

$$\begin{aligned} \underset{\lambda, \, \xi, \, b}{\text{minimize}} \quad & \sum_{i=1}^{m} \lambda_i + C \, \sum_{i=1}^{m} \xi_i \\ \text{subject to} \quad & Q \, \lambda + b \, y \geq 1 - \xi \\ & \lambda, \; \xi \; \geq 0 \end{aligned} \tag{4}$$

Here, $y \in \{-1, \, +1\}^m$ is a vector of class labels for all m training examples and $Q \in \mathbb{R}^{m \times m}$ is given by [2]:

$$Q_{i,j} = y_i \, y_j \, S(x_i, \, x_j)$$

The above formulation is a simple LP that can be solved quite efficiently using, for example, the Gurobi solver (Gurobi Optimization, 2012). Note that unlike the standard formulation of SVM, the LP formulation above remains convex even when the matrix Q is

1. In (Zhu et al., 2004) Eq 5, the Hinge loss is used explicitly in the objective function, which is equivalent to the use of slack variables in our formulation.
2. To reiterate, the similarity function $S : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is determined by the application at hand, and not by the learning system. Therefore, we assume a similarity function is given, and do not address whether or not it is suitable for the learning task.

not PSD because both the objective function and inequality constraints are linear in the optimization variables (λ, ξ, b). Once the above optimization problem is solved, we classify a new example x_t using the rule:

$$\hat{y}_t = \textbf{sign}\{b + \sum_{i=1}^{m} y_i \, \lambda_i \, S(x_i, \, x_t)\} \tag{5}$$

The decision rule in Eq 5 can be motivated in several ways. For instance, it is identical to the decision rule obtained by the Representer Theorem in Eq 2. Moreover, it has been established that decision rules of the form given above are capable of *universal function approximation* when the similarity function $S : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is radial, i.e. a function of distance (Park and Sandberg, 1991).

The 1-norm SVM algorithm can be interpreted as an l_1-regularized SVM applied to the *empirical kernel map*. Given a training set $\{(x_i, \, y_i)\}_{i=1,\ldots,m}$, one can introduce the new mapping:

$$\Phi(\cdot) = (y_1 \, S(x_1, \, \cdot), \, \ldots, \, y_m \, S(x_m, \, \cdot)) \; : \; \mathcal{X} \rightarrow \mathbb{R}^m,$$

which is similar to the *empirical kernel map* (Schölkopf and Smola, 2002) except for the presence of class labels. The 1-norm SVM seeks a separating hyperplane in the new space $\Phi(\cdot)$ with $||\cdot||_1$ regularization, hence the name. Although it is possible to learn the weights λ in the new space $\Phi(\cdot)$ using other approaches, such as SVM with $||\cdot||_2$ regularization, we focus on the approach employed by 1-norm SVM chiefly because it produces sparse solutions. To reiterate, the central claim of this paper is that 1-norm SVM is the best method to learn with indefinite kernels *while also retaining sparsity of the support vector set* [3].

Training examples x_i with $\lambda_i > 0$ are analogous to the *support vectors* in standard SVM, and we will refer to them as support vectors here as well. As depicted in Figure 1, each support vector is 'carefully' placed in the plane to guard a region dominated by its respective class. In practice, because the regularization term in the objective function minimizes the 1-norm of λ, the vector λ tends to be sparse and the number of support vectors tends to be small.

Next, we provide two motivations for using the formulation in (4). First, we show that the 1-norm SVM can be interpreted as a method of finding a decision boundary with a large similarity margin in the original space. Second, we show that the objective function in (4) can be interpreted as a method of minimizing an upper bound on expected test error rate.

2.1. Large similarity margins

Given a similarity function $S(x_i, \, x_j) : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ between examples x_i and x_j, we can define similarity between an example x_t and a class $y = l$ to be a *weighted* sum of similarities

3. Sparse solutions are important for at least two reasons. First, only a small subset of the training set is needed during prediction, and hence prediction can be carried out quite efficiently. Second, minimizing the number of support vectors can be interpreted as a method of minimizing an upper bound on expected test error rate (see for example Eq 93 in (Burges, 1998) in the case of SVM and the discussion in Section 2.2 in the case of 1-norm SVM).

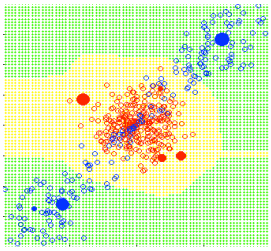

Figure 1: In this figure, two classes are shown in RED and BLUE. The **solid** markers are the support vectors, where size is proportional to the weights λ_i. Classification regions for RED and BLUE classes are shown in YELLOW and GREEN respectively

with all of its examples. In other words, we may write:

$$\mathbb{S}(x_t,\, l) = \sum_{i=1}^{m} \lambda_i \, S(x_i, x_t) \cdot \mathbb{I}\{y_i = l\}$$

to denote class similarity between x_t and a class $y = l$. Here, the weight λ_i represents *importance* of the example x_i to its class y_i. In addition, we can introduce an offset b that quantifies prior preference. Such offset plays a role that is similar to the *prior* in Bayesian methods, the activation threshold in neural networks, and the offset in SVM. Thus, we consider classification using the rule:

$$\hat{y}_t = \mathbf{sign}\{\mathbb{S}(x_t,\, +1) - \mathbb{S}(x_t,\, -1) + b\}, \tag{6}$$

which is identical to the classification rule of 1-norm SVM given in Eq 5. Moreover, we define the *similarity margin* M_t for example x_t in the usual sense:

$$M_t = \mathbb{S}(x_t,\, y_t) - \mathbb{S}(x_t,\, -y_t) + y_t \, b$$

Maximizing the minimum similarity margin can be formulated as a linear program (LP). First, we write:

$$\begin{aligned} \underset{\lambda,\, b,\, M}{\text{maximize}} \quad & M \\ \text{subject to} \quad & \mathbb{S}(x_i,\, y_i) - \mathbb{S}(x_i,\, -y_i) + y_i \, b \geq M, \quad (\text{for all } i) \\ & \lambda \geq 0 \end{aligned}$$

However, the decision rule given by Eq. (6) does not change when we multiply the weights λ by any fixed positive constant including constants that are arbitrarily large. This is because the decision rule only looks into the sign of its argument. In particular, we can always rescale the weights λ to be arbitrarily large, for which $M \to \infty$. This degree of freedom implies that we need to maximize the ratio $M/||\lambda||$ instead of maximizing M in absolute terms. Here, any norm $||\cdot||$ suffices but the 1-norm is preferred because it produces sparse solutions and because it gives better accuracy in practice.

Since our objective is to maximize the ratio $M/||\lambda||_1$, we can fix $M = 1$ and minimize $||\lambda||_1$. In addition, to avoid over-fitting outliers or noisy samples and to be able to handle

the case of non-separable classes, soft-margin constraints are needed as well. This results in the LP formulation of the 1-norm SVM given earlier in (4). Hence, 1-norm SVM can be interpreted as a method of finding a decision boundary with a large similarity margin in the original space. Such interpretation holds regardless of whether or not the similarity function is PSD. Thus, we expect 1-norm SVM to work well even for indefinite kernels.

2.2. True error rate bound

Similar to the original SVM, one can interpret 1-norm SVM as a method of striking a balance between estimation bias and variance.

Lemma 1 *Suppose m training examples were used to build the 1-norm SVM classifier in (4). Let e_{LOO} be the expected leave-one-out validation error rate on the same training set. Then:*

$$e_{LOO} \leq \frac{||\lambda||_0}{m} + \frac{||\xi||_0}{m} \tag{7}$$

Here, $||z||_0$ denotes the number of non-zero entries in z.

Proof Let $\lambda^\star$ and $\xi^\star$ be the optimal solutions to the 1-norm SVM in (4). If $\xi_i^\star = \lambda_i^\star = 0$, then the i-th training example was classified correctly and it will continue to be classified correctly if it is the only example removed from the training set. The latter statement holds because removing the i-th example from the training set is equivalent to adding the new constraint $\lambda_i = \xi_i = 0$ to the formulation (4), which is the original optimal value of $\lambda_i^\star$ and $\xi_i^\star$. Because the new feasibility region is a subset of the original feasibility region and it contains the original optimal solution, the optimal solution remains unchanged. Hence:

$$e_{LOO} \leq \frac{||\,\lambda^\star \circ \xi^\star\,||_0}{m} \leq \frac{||\lambda^\star||_0 + ||\xi^\star||_0}{m},$$

where $\circ$ is the Hadamard (elementwise) product. ■

Corollary 2 *Let e_{tst} be the true error rate (true risk) when 1-norm SVM is trained on a randomly selected training set S_m with m training examples. Then, the expected test error rate satisfies:*

$$\mathbb{E}_{S_{m-1}}[e_{tst}] \leq \frac{\mathbb{E}_{S_m}||\lambda||_0}{m} + \frac{\mathbb{E}_{S_m}||\xi||_0}{m} \tag{8}$$

Here, expectation of the test error rate is taken over all possible training sets of size $m-1$ whereas remaining expectations are taken over all possible training sets of size m.

Proof By the Luntz-Brailovsky theorem (Luntz and Brailovsky, 1969; Vapnik and Chapelle, 2000), we have:

$$\mathbb{E}_{S_{m-1}}[e_{tst}] = \mathbb{E}_{S_m}[e_{LOO}], \tag{9}$$

where e_{LOO} is the leave-one-out validation error. Using Eq 9 and Lemma 1 yields the desired result. ■

The tradeoff in Eq 8 is analogous to the classical tradeoff in estimation between bias and variance (Hastie et al., 2001). On one hand, one can fit the training set perfectly, e.g. by using radial similarity functions with sufficiently large bandwidth that effectively turn 1-norm SVM into a 1-NN classifier, but the fraction of support vectors becomes at its worst, hence high variance. On the other hand, one can choose a very small number of support vectors but this tends to increase the training error rate, hence high bias. In the 1-norm SVM formulation in (4), the cost function penalizes both training error (bias) and the number of support vectors (variance) simultaneously by penalizing the $||\cdot||_1$ of slack variables ξ and weights λ. Because minimizing $||\cdot||_1$ promotes *sparsity* (Boyd and Vandenberghe, 2004), Corollary 2 states that the 1-norm SVM can be interpreted as a method of minimizing expected true risk.

3. Experiments and Results

In this section, we present experimental results of applying 1-norm SVM to synthetic and real-world classification problems, and demonstrate its effectiveness in handling indefinite similarity functions.

3.1. Synthetic Datasets.

First, 1-norm SVM was tested on six synthetic datasets depicted in Figure 2. In these datasets, the radial basis function (RBF) $S(x_i, x_j) = \exp\{-\gamma\,||x_i - x_j||_2^2\}$ was used, where the bandwidth parameter γ was selected using a grid search on a separate validation set. Figure 3 plots test error rate as a function of training set size m, with the Bayes rate for each classification problem indicated in the legend bar. As shown in Figure 3, test error rate approaches the optimal Bayes rate for sufficiently large training sets in all six classification problems. This test verifies that 1-norm SVM is capable of producing accurate decision boundary for various complex mixtures of classes.

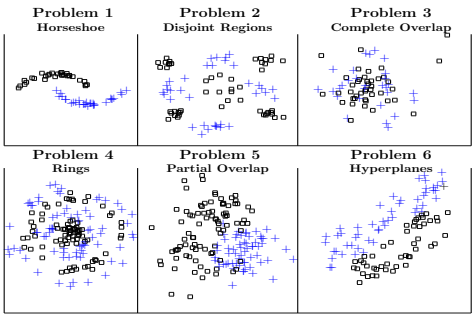

Figure 2: The six synthetic datasets that are used in evaluating 1-norm SVM.

3.2. Real Datasets.

For real datasets, we compared performance of 1-norm SVM against popular classification algorithms for both PSD and non-PSD similarity functions. We will first describe the datasets and test methodology, and discuss test results after that.

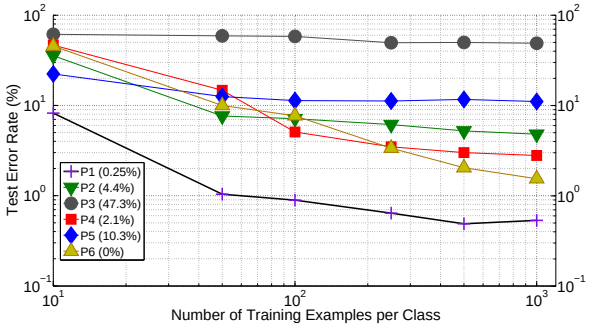

Figure 3: Performance of 1-norm SVM on the six synthetic datasets. Each error rate, plotted in a log-scale, is an average of five i.i.d training/test sets. P1,..., P6 stands for Problem 1,..., Problem 6 shown in Figure 2. The optimal Bayes rates are indicated in the legends bar. In the y-axis, each grid line between 10^z and 10^{z+1} is to be read as 1×10^z, 2×10^z,... 9×10^z. For example, the grid lines from 10^1 to 10^2 correspond to the values 10, 20, ..., 80, 90.

3.2.1. Datasets:

The following datasets and similarity functions are used.

(A) **IMDB**: This is a graph-based dataset that contains movies released between 1996 and 2001 (Macskassy and Provost, 2007). Class label identifies whether the opening weekend box-office receipts exceeded $2 million. An edge weight between two movies is the number of common production companies, actors, producers, or directors. In our implementation, all edge weights were normalized to fall in the range [0, 1]. The following similarity functions are used:

(a) *PSD*: The Jaccard index $S_{i,j} = \frac{\sum_k \min\{w_{i,k}, w_{j,k}\}}{\sum_k \max\{w_{i,k}, w_{j,k}\}}$, where $w_{i,k}$ is edge weight.

(b) *Non-PSD*: Edge weight $S_{i,j} = w_{i,j}$ and $S_{i,i} = 1$.

(B) **Word-Sim-353**: This dataset contains human-judged similarities between English words (Finkelstein et al., 2002). All similarities are again normalized to fall in the range [0, 1] and self-similarity is set to unity. We grouped words into two categories: 'living' vs. 'non-living', and used the two similarity functions specified earlier for the IMDB dataset. Examples of the 'living' class include *children*, *Maradona*, *brother*, *carnivore*, and *mammal*.

(C) **Caltech-101**: This dataset contains images of various objects (Fei-Fei et al., 2004). We grouped images of 'Big Cats', 'Winged Insects', and 'Flowers' into three classes and trained three separate binary classifiers between every pair of classes. Each image was converted into a histogram using the two MATLAB commands `rgb2gray` and `imhist` and used Laplace normalization. This effectively represents the i-th image by a probability distribution p_i. We, then, used the following two similarity functions:

(a) *PSD*: Intersection (a.k.a. overlapping coefficient) $S_{i,j} = \sum_k \min\{p_{i,k}, p_{j,k}\}$.

(b) *Non-PSD*: We used $S_{i,j} = \max\{0, 1 - 0.1 \times D(p_i \,||p_j)\}$, where $D(p_i||p_j)$ is the symmetrized Kullback-Leibler divergence [4].

(D) **Splice**: This is a biological sequence classification dataset (Noordewier et al., 1991) that was downloaded from the UCI repository (Blake and Merz, 1998). Each example is a 60-letter DNA sequence. We performed classification between the two classes EI and IE. The similarity functions are:

(a) *PSD*: We used the implementation of string kernels given in (Soman et al., 2009). Because string kernels can grow quite rapidly, we normalized using the cosine similarity: $S_{i,j} = \frac{K_{i,j}}{\sqrt{K_{i,i} \cdot K_{j,j}}}$.

(b) *Non-PSD*: We used the longest-common-subsequence (LCS) between two strings. Because each string is 60 letters in length, we set $S_{i,j} = LCS(x_i, x_j)/60$.

(E) **CNAE-9**: This is a text classification dataset available at the UCI repository, where each text is represented using bag-of-words. The dataset contains 9 classes and we randomly selected five binary classification problems: 1-vs-5, 5-vs-4, 6-vs-8, 3-vs-9, and 2-vs-7 [5]. These are represented by P1 through P5 in Table 1 respectively. The two similarity functions are:

(a) *PSD*: The cosine similarity $S_{i,j} = \frac{x_i^T x_j}{||x_i|| \cdot ||x_j||}$, which is commonly used for text classification tasks (Chen et al., 2009a).

(b) *Non-PSD*: The second similarity function used is a variant to the first. Specifically, we have $S_{i,j} = \frac{v^T v}{||x_i|| \cdot ||x_j||}$, where $v_k = \min\{x_{i,k}, x_{j,k}\}$.

(F) **Ionosphere, Australian, Breast Cancer, Haberman, and Diabetes**: These are five binary classification problems with numeric features available at the UCI repository. We used the following similarity functions:

(a) *PSD*: The RBF kernel $S_{i,j} = e^{-\gamma ||x_i - x_j||_2^2}$.

(b) *Non-PSD*: The sigmoid kernel $S_{i,j} = \tanh\{\gamma \cdot x_i^T x_j + r\}$, which is popular due to its origins in neural networks. To ensure that the kernel matrix is not PSD, we fixed $r = -1$ [6].

3.2.2. Test Methodology and Results:

When the similarity function is PSD, we compared performance of 1-norm SVM vs. standard SVM. For each dataset, the value of the tradeoff constant C was selected using 5-fold cross validation for $C \in \{2, 4, 8, 16, 32\}$. When the RBF kernel is used, the bandwidth γ is also selected using 5-fold cross validation in the grid $\gamma \in \{2^{-15}, 2^{-14}, \ldots, 2^{-1}, 1\}$. SVM was implemented using the LIBSVM library (Chang and Lin, 2001), whereas 1-norm SVM

4. The reason behind choosing 0.1 is because 95% of pairwise distances are less than 10.
5. We perfomred a random permutation of the set of integers $\{1, 2, \ldots, 9\}$. Each pair of adjacent labels was used as a binary classification problem, where the 9^{th} label is trainied vs. the 1^{st}.
6. It has been shown that the sigmoid kernel is PSD only if $r \geq 0$ (Burges, 1999). However, using $r < 0$ tends to perform better (Lin and Lin, 2003).

was implemented using the Gurobi solver (Gurobi Optimization, 2012). In all classification problems, we reported the average test error rate of five random training-to-test splits, with a training-to-split ratio of 4:1. The same split is always used in both SVM and 1-norm SVM.

When the similarity function is non-PSD, we compared performance of 1-norm SVM against the three popular methods proposed previously in the literature:

1. *Non-convex optimization*: This was implemented using the LIBSVM library with its `-t 4` option. When the similarity matrix is non-PSD, the LIBSVM package seeks a stationary point using non-convex optimization (Lin and Lin, 2003).

2. *Kernel Approximation*: PSD kernel approximation was tested using three approaches: (1) the *denoise* method, (2) the *flip* method, and (3) the indefinite SVM formulation proposed by Luss and d'Aspremont (Luss and dAspremont, 2009). The denoise and flip methods were implemented by supplying the modified (PSD) kernel matrix to LIBSVM using the `-t 4` option. The indefinite SVM method was tested using the implementation available for download at the authors' website.

3. *SVM in Krien Spaces*: SVM in Krien spaces was implemented using the ESVM algorithm described in (Loosli et al., 2013). ESVM comprises of two main steps: (1) eigen-decomposition, and (2) SVM training. LIBSVM was used for the SVM training step.

In all methods, hyper-parameters were selected using cross validation and grid search, implemented separately for each individual method. Test results are shown in Table 1. All results reported here are based on the best selected hyper-parameters of these methods.

4. Discussion

As shown in columns 2 and 3 of Table 1, when the similarity function is PSD, performance of 1-norm SVM is comparable to that of SVM. When running statistical significance tests, we find no statistically significant evidence that one method outperforms the other at the 95% confidence level. For example, the two-tailed Wilcoxon's signed rank test (Demšar, 2006) gives a value of $p = 0.155$. This validation verifies that 1-norm SVM is a viable algorithm for binary classification even when the similarity function is positive semidefinite (PSD). Such experimental evidence agrees with earlier conclusions (Zhu et al., 2004).

For non-PSD kernels, on the other hand, we compare the error rate of 1-norm SVM (shown in column 5 of Table 1) with the other methods (in columns 6-10). The rows in Table 1 are ordered by the value of β shown in column 4, which is a measure of how indefinite the similarity matrix is. In particular, a value of $\beta = 0$ corresponds to positive semidefinite matrices while $\beta = 1$ corresponds to negative semidefinite matrices. As shown in the table, 1-norm SVM and ESVM (i.e. SVM in Krein spaces) both outperform all remaining classification methods in nearly all the datasets. Performance of ESVM, however, is very similar to that of 1-norm SVM, which is quite intriguing given the very different approaches employed by the two algorithms!

Nevertheless, unlike ESVM whose solution is quite dense, the 1-norm SVM method yields very sparse solutions so that prediction time is faster. In fact, 1-norm SVM yields

Table 1: Average test error rate results on 16 Datasets using the similarity functions described in Section 3.2.

Dataset (m)	PSD		Non-PSD						
					Non-Convex Optimization	Kernel Approximation			svm in Krein space
	1-norm svm	svm	$\beta^{(1)}$	1-norm svm	svm	den	flip	Indefinite svm	esvm
IMDB (1441)	16.0%	**15.6%**	0.014	18.8%	17.7%	**17.6%**	**17.6%**	18.1%	18.8%
Word-Sim-353 (437)	**12.9%**	13.7%	0.033	**14.7%**	15.6%	15.2%	15.4%	**14.7%**	15.8%
Caltech-101-P2 (368)	26.0%	**24.0%**	0.048	22.1%	44.0%	62.0%	41.6%	37.5%	**20.7%**
Caltech-101-P3 (379)	19.8%	**19.2%**	0.080	**22.8%**	40.7%	54.5%	36.5%	32.0%	24.1%
Caltech-101-P1 (387)	**31.7%**	**31.7%**	0.089	**30.1%**	40.9%	50.8%	39.5%	38.0%	31.7%
Splice (1527)	6.56%	**5.79%**	0.096	5.70%	5.74%	5.97%	6.95%	5.64%	**4.86%**
CNAE-9-P1 (240)	0.56%	**0%**	0.318	**0%**	13.2%	6.17%	0.17%	**0%**	**0%**
CNAE-9-P5 (240)	2.05%	**1.15%**	0.332	3.72%	10.9%	6.61%	1.94%	**1.25%**	3.50%
CNAE-9-P3 (240)	1.67%	**0.94%**	0.340	**2.50%**	25.3%	25.4%	8.67%	3.75%	3.33%
CNAE-9-P2 (240)	**0.44%**	1.22%	0.343	**0.33%**	21.6%	20.7%	5.78%	2.50%	1.67%
CNAE-9-P4 (240)	1.89%	**0.83%**	0.344	2.56%	16.3%	9.67%	2.17%	**1.25%**	1.67%
Ionosphere (351)	7.14%	**6.57%**	0.999	**10.3%**	37.7%	70.3%	66.0%	36.0%	**10.3%**
Australian (690)	**16.6%**	16.9%	1.0	**12.2%**	40.7%	40.7%	91.9%	47.7%	15.1%
Breast Cancer (699)	**3.15%**	3.51%	1.0	**4.32%**	32.9%	32.9%	96.4%	30.1%	5.76%
Haberman (398)	**30.2%**	31.2%	1.0	**26.6%**	**26.6%**	40.1%	27.2%	*(2)	**26.6%**
Diabetes (768)	28.9%	**27.9%**	1.0	**22.7%**	33.3%	70.3%	54.2%	35.8%	25.9%

(1) $\beta = \dfrac{\sum_i |\lambda_i| \cdot \mathbb{I}\{\lambda_i < 0\}}{\sum_i |\lambda_i|}$, where λ_i are eigenvalues, is a measure of how negative semidefininte the similarity matrix is.

(2) The algorithm failed to terminate.

Table 2: The number of support vectors (SVs) used by 1-norm SVM and ESVM for the 16 classification problems with indefinite similarity functions.

Datasets	No. of training examples	No. of SVs in 1-norm SVM	No. of SVs in ESVM
IMDB	1441	630	1438
Word-Sim-353	437	26	436
Caltech-101-P2	368	39	386
Caltech-101-P3	379	47	379
Caltech-101-P1	387	35	386
Splice	1527	224	1527
CNAE-9-P1	240	2	233
CNAE-9-P5	240	21	240
CNAE-9-P3	240	23	238
CNAE-9-P2	240	2	235
CNAE-9-P4	240	23	240
Ionosphere	351	39	351
Australian	690	7	690
Breast Cancer	699	20	699
Haberman	398	28	398
Diabetes	768	13	768

solutions that are often 10-20 times, sometimes even 100 times, sparser than ESVM. Table 2 lists the number of support vectors used by both methods.

Last but not the least, in order to verify statistical significance at the 95% confidence level, we used Holm's step-down procedure for multiple comparisons applied to the two-tailed Wilcoxon's signed rank test (Demšar, 2006; Holm, 1979). More specifically, each null hypothesis H_i asserts that 1-norm SVM and the i-th alternative classifier have similar performance. When H_i is tested using the two-tailed Wilcoxon's signed rank test, the resulting p values are shown in Table 3. Using a confidence level of 95% in Holm's step down procedure, we find that the null hypothesis is rejected for non-convex optimization and all kernel approximation methods. This confirms that 1-norm SVM outperforms non-convex optimization and kernel approximation with a statistically significant evidence. However, there is no statistically significant evidence at the 95% confidence level that 1-norm SVM outperforms ESVM in terms of predictive accuracy. Here, it is perhaps worth reiterating that the 1-norm SVM significantly outperforms ESVM in terms of sparsity of solutions as shown in Table 2. Therefore, the 1-norm SVM method achieves the highest predictive accuracy among all methods that learn with indefinite kernels, while also retaining sparsity of the support vector set.

Finally, it is worth pointing out that indefinite similarity functions in our evaluation led to lower error rates than PSD similarity functions in roughly 50% of the datasets. This includes, most notably, the datasets: caltech-101-p2, australian, haberman, and diabetes. Therefore, *even for classification problems where PSD similarity functions are readily available, learning with non-PSD kernels remains important because it can result in a better classification accuracy.*

Table 3: In this table, the second column lists the p values in increasing order of the two-tailed Wilcoxon's signed rank test. The last column shows the critical values when Holm's step-down procedure is used at the 95% confidence level.

Null Hypothesis (H_i)	p value	Adjusted Critical Value
1-norm SVM vs. SVM with non-convex optimization	0.0003	0.0100
1-norm SVM vs. Denoise	0.0008	0.0125
1-norm SVM vs. Flip	0.0052	0.0167
1-norm SVM vs. Indefinite SVM	0.0107	0.0250
1-norm SVM vs. ESVM	0.0771	0.0500

5. Conclusion

Extensive research effort has been devoted recently to training support vector machines (SVM) with indefinite kernels. In this paper, we establish theoretically and experimentally that a variant of the 1-norm support vector machines is a better method for handling indefinite kernels. The 1-norm SVM method formulates large-margin separation as a convex linear programming (LP) problem without requiring that the kernel matrix be positive semidefinite (PSD). It uses the indefinite similarity function directly without any transformation, and, hence, it always treats both training and test examples consistently. In addition, 1-norm SVM achieves the highest accuracy among all methods that train SVM with indefinite kernels, with a statistically significant evidence, while also retaining sparsity of the support vector set.

References

C. L. Blake and C. J. Merz. UCI repository of machine learning databases, 1998.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Paul S Bradley and Olvi L Mangasarian. Feature selection via concave minimization and support vector machines. In *ICML*, 1998.

C. J. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

Christopher Burges. Geometry and invariance in kernel based methods. *Advances in kernel methodssupport vector learning*, pages 89–116, 1999.

C. Chang and C. J. Lin. LIBSVM: A library for support vector machines, 2001. [Online]: http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Jianhui Chen and Jieping Ye. Training SVM with indefinite kernels. In *Proceedings of ICML*, pages 136–143, 2008.

Yihua Chen, Eric K Garcia, Maya R Gupta, Ali Rahimi, and Luca Cazzanti. Similarity-based classification: Concepts and algorithms. *JMLR*, 10:747–776, 2009a.

Yihua Chen, Maya R Gupta, and Benjamin Recht. Learning kernels from indefinite similarities. In *Proceedings of ICML*, pages 145–152, 2009b.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *JMLR*, 7:1–30, 2006.

L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In *IEEE CVPR: Workshop on Generative-Model Based Vision*, 2004.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing search in context: The concept revisited. *ACM Transactions on Information Systems*, 20(1):116–131, January 2002.

Glenn M Fung and Olvi L Mangasarian. A feature selection Newton method for support vector machine classification. *Computational optimization and applications*, 28:185–202, 2004.

Thore Graepel, Ralf Herbrich, Peter Bollmann-Sdorra, and Klaus Obermayer. Classification on pairwise proximity data. *Advances in NIPS*, pages 438–444, 1999.

Inc. Gurobi Optimization. Gurobi optimizer reference manual, 2012. URL http://www.gurobi.com.

Bernard Haasdonk. Feature space interpretation of svms with indefinite kernels. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(4):482–492, 2005.

T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning: data mining, inference, and prediction*. 2001.

Melanie Hilario and Alexandros Kalousis. Approaches to dimensionality reduction in proteomic biomarker studies. *Briefings in Bioinformatics*, 9(2):102–118, 2008.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pages 65–70, 1979.

H.-T. Lin and C.-J. Lin. A study on sigmoid kernels for SVM and the training of non-PSD kernels by SMO-type methods. Technical report, Department of Computer Science, National Taiwan University, 2003. URL http://www.csie.ntu.edu.tw/~cjlin/papers/tanh.pdf.

Huan Liu, Hiroshi Motoda, Rudy Setiono, and Zheng Zhao. Feature selection: An ever evolving frontier in data mining. *Journal of Machine Learning Research (JMLR) - Workshop and Conference Proceeding*, 10:4–13, 2010.

Gaelle Loosli, Cheng Soon Ong, and Stephane Canu. SVM in Krein spaces. Technical report, 2013. URL http://hal.archives-ouvertes.fr/hal-00869658/.

A Luntz and V Brailovsky. On estimation of characters obtained in statistical procedure of recognition. *Technicheskaya Kibernetica*, 3(6), 1969.

Ronny Luss and Alexandre dAspremont. Support vector machine classification with indefinite kernels. *Mathematical Programming Computation*, 1(2-3):97–118, 2009.

Sofus A. Macskassy and Foster Provost. Classification in networked data: A toolkit and a univariate case study. *JMLR*, 8:935–983, May 2007.

J. S. Marron, M. J. Todd, and J. Ahn. Distance-weighted discrimination. *Journal of the American Statistical Association*, 102(480):1267–1271, 2007.

M. O. Noordewier, G. G. Towell, and J. W. Shavlik. Training knowledge-based neural networks to recognize genes in dna sequences. In *Advances in NIPS*, 1991.

Cheng Soon Ong, Xavier Mary, Stephane Canu, and Alexander J. Smola. Learning with non-positive kernels. In *ICML*, 2004.

Jooyoung Park and Irwin W Sandberg. Universal approximation using radial-basis-function networks. *Neural computation*, 3(2):246–257, 1991.

Elzbieta Pekalska, Pavel Paclik, and Robert PW Duin. A generalized kernel approach to dissimilarity-based classification. *JMLR*, 2:175–211, 2001.

Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

Pannagadatta K Shivaswamy, Chiranjib Bhattacharyya, and Alexander J Smola. Second order cone programming approaches for handling missing and uncertain data. *JMLR*, 7: 1283–1314, 2006.

KP Soman, R Loganathan, and V Ajay. *Machine Learning with SVM and other Kernel methods*. 2009.

Vladimir Vapnik and Olivier Chapelle. Bounds on error expectation for support vector machines. *Neural Computation*, 12(9):2013–2036, 2000.

Vladimir N Vapnik. An overview of statistical learning theory. *Neural Networks, IEEE Transactions on*, 10(5):988–999, 1999.

Gang Wu, Zhihua Zhang, and Edward Y. Chang. An analysis of transformation on non-positive semidefinite similarity matrix for kernel machines. Technical report, UCSB, 2005.

Yiming Ying, Colin Campbell, and Mark Girolami. Analysis of SVM with indefinite kernels. *Advances in NIPS*, 22:2205–2213, 2009.

Ji Zhu, Saharon Rosset, Trevor Hastie, and Rob Tibshirani. 1-norm support vector machines. *Advances in neural information processing systems (NIPS)*, 16:49–56, 2004.

Hui Zou. An improved 1-norm SVM for simultaneous classification and variable selection. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, pages 675–681, 2007.