Pattern Recognition Letters 129 (2020) 1–7

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Sparsification of core set models in non-metric supervised learning

Frank-Michael Schleif[a,b,*], Christoph Raab[a], Peter Tino[b]

[a] School of Computer Science, University of Applied Sciences Würzburg-Schweinfurt, Sanderheinrichsleitenweg 20, Würzburg, 97074, Germany
[b] University of Birmingham School of Computer Science, Edgbaston, Birmingham, B15 2TT, UK

ARTICLE INFO

Article history:
Received 4 February 2019
Revised 27 August 2019
Accepted 21 October 2019
Available online 4 November 2019

Keywords:
Large scale indefinite learning
Krein space
Sparse models
Orthogonal matching pursuit
Core sets
Non-metric learning

ABSTRACT

Supervised learning employing positive semi definite kernels has gained wide attraction and lead to a variety of successful machine learning approaches. The restriction to positive semi definite kernels and a hilbert space is common to simplify the mathematical derivations of the respective learning methods, but is also limiting because more recent research indicates that non-metric, and therefore non positive semi definite, data representations are often more effective. This challenge is addressed by multiple approaches and recently dedicated algorithms for so called indefinite learning have been proposed. Along this line, the Krĕin space Support Vector Machine (KSVM) and variants are very efficient classifiers for indefinite learning problems, but with a non-sparse decision function. This very dense decision function prevents practical applications due to a costly out of sample extension. We focus on this problem and provide two post processing techniques to sparsify models as obtained by a Krĕin space SVM approach. In particular we consider the indefinite Core Vector Machine and indefinite Core Vector Regression Machine which are both efficient for psd kernels, but suffer from the same dense decision function, if the Krĕin space approach is used. We evaluate the influence of different levels of sparsity and employ a Nyström approach to address large scale problems. Experiments show that our algorithm is similar efficient as the non-sparse Krĕin space Support Vector Machine but with substantially lower costs, such that also problems of larger scale can be processed.

Learning of classification models for indefinite kernels received substantial interest with the advent of domain specific similarity measures. Indefinite kernels are a severe problem for most kernel based learning algorithms because classical mathematical assumptions such as positive definiteness, used in the underlying optimization frameworks are violated. As a consequence e.g. the classical Support Vector Machine (SVM) [38] has no longer a convex solution - in fact, most standard solvers will not even converge for this problem [16]. Researchers in the field of e.g. psychology [13], vision [28,40] and machine learning [4] have criticized the typical restriction to metric similarity measures. In [4] it is shown that many real life problems are better addressed by e.g. kernel functions which are not restricted to be based on a metric. Non-metric measures (leading to kernels which are not positive semi-definite (non-psd)) are common in many disciplines. The use of divergence measures [32,41] is very popular for spectral data analysis in chemistry, geo- and medical sciences [19,21], and are in general not metric. Also the popular Dynamic Time Warping (DTW) [27] algorithm provides a non-metric alignment score which is often used as a proximity measure between two one-dimensional functions of different length. In image processing and shape retrieval indefinite proximities are often obtained by means of the inner distance [15] - another non-metric measure. Further prominent examples for genuine non-metric proximity measures can be found in the field of bioinformatics where classical sequence alignment algorithms (e.g. smith-waterman score [9]) produce non-metric proximity values. Multiple authors argue that the non-metric part of the data contains valuable information and should not be removed [25,28]. Furthermore, it has been shown [16,29] that work-arounds such as eigenspectrum modifications are often inappropriate or undesirable, due to a loss of information and problems with the out-of sample extension. Nevertheless they are still often used and can serve as a baseline approach. Due to its strong theoretical foundations, Support Vector Machine (SVM) has been extended for indefinite kernels in a number of ways [8,10,17]. A recent survey on indefinite learning is given in [29]. In [16] a stabilization approach was proposed to calculate a valid SVM model in the Krĕin space which can be directly applied on indefinite kernel matrices. This approach has shown great promise in a number of learning problems, but has intrinsically quadratic to cubic complexity and provides a dense decision model. The approach can also be used for the recently proposed indefinite Core

* Corresponding author.
E-mail address: fschleif@techfak.uni-bielefeld.de (F.-M. Schleif).

https://doi.org/10.1016/j.patrec.2019.10.024
0167-8655/

Vector Machine (iCVM) [30] which has better scalability but still suffers from the dense model. The initial sparsification approach of the iCVM proposed in [30] is not always applicable and we will provide an alternative in this paper.

Another indefinite SVM formulation was provided in [1], but it is based on an empirical feature space technique, which changes the feature space representation. Additionally, the imposed input dimensionality scales with the number of input samples, which is unattractive in out of sample extensions.

The present paper improves the work of [30] by providing a sparsification approach such that the otherwise very dense decision model becomes sparse again. The new decision function approximates the original one with high accuracy and makes the application of the model practical.

We now review the main parts of the Krĕin space SVM provided in [16] showing why the obtained α-vector is dense. The effect is the same for to the Core Vector Machine as shown in [30]. For details on the iCVM derivation we refer the reader to [30].

1. Learning with non-psd kernels

Learning with non-psd kernels can be a challenging problem and may occur very quickly as discussed before, if domain specific measure are used or simply due to noise. The metric violations cause negative eigenvalues in the eigenspectrum of the kernel matrix K, leading to non-psd similarity matrices or indefinite kernels. Many learning algorithms are based on kernel formulations which have to be symmetric and psd. The mathematical meaning of a kernel is the inner product in some Hilbert space [33]. However, it is often loosely considered simply as a pairwise "similarity" measure between data items, leading to a similarity matrix S.

If a particular learning algorithm requires the use of Mercer kernels and the similarity measure does not fulfill the kernel conditions, steps must be taken to ensure a valid model.

1.1. Eigenspectrum approaches

A natural way to address the indefiniteness problem and to obtain a psd similarity matrix is to correct the eigenspectrum of the original similarity matrix S. Popular strategies include *flipping, clipping and shift correction*. The non-psd similarity matrix S is decomposed by an eigen decomposition: $S = U\Lambda U^\top$, where U contains the eigenvectors of S and Λ contains the corresponding eigenvalues. One can now adapt the eigenvalues to get rid of the negative eigenvalues and to end up with a psd kernel.

Clip eigenvalue correction: All negative eigenvalues in Λ are set to 0. Spectrum clip leads to the nearest psd matrix S in terms of the Frobenius norm [12].

Flip eigenvalue correction: All negative eigenvalues in Λ are set to $\lambda_i := |\lambda_i| \forall i$ which at least keeps the absolute values of the negative eigenvalues and can be relevant if these eigenvalue contain important information [25].

Shift eigenvalue correction: The shift operation was already discussed earlier by different researchers [5] and modifies Λ such that $\Lambda := \Lambda - \min_{ij} \Lambda$. Spectrum shift enhances all the self-similarities by the amount of ν and does not change the similarity between any two different data points, but it may also increase the intrinsic dimensionality of the data space and amplify noise contributions.

In the experiments we will only compare with the clip and flip approach. The latter one is also an algorithmic part of the Krĕin space SVM model. If one of the former corrections is applied to the input kernel any standard kernel based learning method like SVM can be used. One major drawback of these approaches is the rather complicated out of sample extension to new test points but also that the data representation may have changed completely, leading to inferior results.

1.2. Krĕin space SVM

The Krĕin Space SVM (KSVM) [16], replaced the classical SVM minimization problem by a stabilization problem in the Krĕin space. The respective equivalence between the stabilization problem and a standard convex optimization problem was shown in [16]. Let $x_i \in X, i \in \{1, \ldots, N\}$ be training points in the input space X, with labels $y_i \in \{-1, 1\}$, representing the class of each point. The input space X is often considered to be $\mathbb{R}^d$, but can be any suitable space due to the kernel trick. For a given positive C, SVM is the minimum of the following regularized empirical risk functional

$$J_C(f,b) = \min_{f\in\mathcal{H}, b\in\mathbb{R}} \frac{1}{2}\|f\|^2_{\mathcal{H}} + C \cdot H(f,b)$$
$$H(f,b) = \sum_{i=1}^{N} \max(0, 1 - y_i(f(x_i) + b)) \quad (1)$$

Using the solution of Eq. (1) as $(f_C^*, b_C^*) := \arg\min J_C(f,b)$ one can introduce $\tau = H(f_C^*, b_C^*)$ and the respective convex quadratic program (QP)

$$\min_{f\in\mathcal{H}, b\in\mathbb{R}} \frac{1}{2}\|f\|^2_{\mathcal{H}} \quad s.t. \sum_{i=1}^{N} \max(0, 1 - y_i(f(x_i) + b)) \leq \tau \quad (2)$$

where we detail the notation in the following. This QP can be also seen as the problem of retrieving the orthogonal projection of the null function in a Hilbert space $\mathcal{H}$ onto the convex feasible set. The view as a projection will help to link the original SVM formulation in the Hilbert space to a KSVM formulation in the Krein space. First we need a few definitions, widely following [16]. A Krĕin space is an *indefinite* inner product space endowed with a Hilbertian topology.

Definition 1 (Inner products and inner product space)**.** Let $\mathcal{K}$ be a real vector space. An inner product space with an indefinite inner product $\langle\cdot,\cdot\rangle_{\mathcal{K}}$ on $\mathcal{K}$ is a bi-linear form where all $f, g, h \in \mathcal{K}$ and $\alpha \in \mathbb{R}$ obey the following conditions:

- Symmetry: $\langle f, g\rangle_{\mathcal{K}} = \langle g, f\rangle_{\mathcal{K}}$,
- linearity: $\langle \alpha f + g, h\rangle_{\mathcal{K}} = \alpha\langle f, h\rangle_{\mathcal{K}} + \langle g, h\rangle_{\mathcal{K}}$
- and $\langle f, g\rangle_{\mathcal{K}} = 0\ \forall g \in \mathcal{K}$ implies $f = 0$.

An inner product is positive definite if $\forall f \in \mathcal{K},\ \langle f, f\rangle_{\mathcal{K}} \geq 0$, negative definite if $\forall f \in \mathcal{K},\ \langle f, f\rangle_{\mathcal{K}} \leq 0$, otherwise it is indefinite. A vector space $\mathcal{K}$ with inner product $\langle\cdot,\cdot\rangle_{\mathcal{K}}$ is called inner product space.

Definition 2 (Krĕin space and pseudo Euclidean space)**.** An inner product space $(\mathcal{K}, \langle\cdot,\cdot\rangle_{\mathcal{K}})$ is a Krĕin space if there exist two Hilbert spaces $\mathcal{H}_+$ and $\mathcal{H}_-$ spanning $\mathcal{K}$ such that $\forall f \in \mathcal{K},\ f = f_+ + f_-$ with $f_+ \in \mathcal{H}_+,\ f_- \in \mathcal{H}_-$ and $\forall f, g \in \mathcal{K},\ \langle f, g\rangle_{\mathcal{K}} = \langle f_+, g_+\rangle_{\mathcal{H}_+} - \langle f_-, g_-\rangle_{\mathcal{H}_-}$. A finite-dimensional Krĕin-space is a so called pseudo Euclidean space (pE).

If $\mathcal{H}_+$ and $\mathcal{H}_-$ are reproducing kernel hilbert spaces (RKHS), $\mathcal{K}$ is a reproducing kernel Krĕin space (RKKS). For details on RKHS and RKKS see e.g. [25]. In this case the uniqueness of the functional decomposition (the nature of the RKHSs $\mathcal{H}_+$ and $\mathcal{H}_-$) is not guaranteed. In [23] the reproducing property is shown for a RKKS $\mathcal{K}$. There is a unique symmetric kernel $k(x, x)$ with $k(x,\cdot) \in \mathcal{K}$ such that the reproducing property holds (for all $f \in \mathcal{K}, f(x) = \langle f, k(x,\cdot)\rangle_{\mathcal{K}}$) and $k = k_+ - k_-$ where k_+ and k_- are the reproducing kernels of the RKHSs $\mathcal{H}_+$ and $\mathcal{H}_-$.

As shown in [23] for any symmetric non-positive kernel k that can be decomposed as the difference of two positive kernels k_+ and k_-, a RKKS can be associated to it. In [16] it was shown how

the classical SVM problem can be reformulated by means of a stabilization problem. This is necessary because a classical norm as used in Eq. (2) does not exist in the RKKS but instead the norm is reinterpreted as a projection which still holds in RKKS and is used as a regularization technique [16]. This allows to define SVM in RKKS (viewed as Hilbert space) as the orthogonal projection of the null element onto the set [16]:

$$S = \{f \in \mathcal{K}, b \in \mathbb{R} | H(f,b) \leq \tau\} \text{ and } 0 \in \partial_b H(f,b)$$

where ∂_b denotes the sub differential with respect to b. The set S leads to a unique solution for SVM in a Krĕin space [16]. As detailed in [16] one finally obtains a stabilization problem which allows one to formulate a SVM in a Krĕin space.

$$\text{stab}_{f\in\mathcal{K},b\in\mathbb{R}} \frac{1}{2}\langle f,f\rangle_{\mathcal{K}} \quad \text{s.t.} \sum_{i=1}^{l} \max(0, 1 - y_i(f(x_i)+b)) \leq \tau \qquad (3)$$

where stab means stabilize as detailed in the following: In a classical SVM in RKHS the solution is regularized by minimizing the norm of the function f. In Krĕin spaces however minimizing such a norm is meaningless since the dot-product contains both the positive and negative components. That's why the regularization in the original SVM through minimizing the norm f has to be transformed in the case of Krĕin spaces into a min-max formulation, where we jointly minimize the positive part and maximize the negative part of the norm. The authors of [23] termed this operation the stabilization projection, or stabilization. Further mathematical details can also be found in [11]. An example illustrating the relations between minimum, maximum and the projection/stabilization problem in the Krĕin space is illustrated in [16].

In [16] it is further shown that the stabilization problem Eq. (3) can be written as a minimization problem using a semi-definite kernel matrix. By defining a projection operator with transition matrices it is also shown how the dual RKKS problem for the SVM can be related to the dual in the RKHS. We refer the interested reader to [16]. One - finally - ends up with a flipping operator applied to the eigenvalues of the indefinite kernel matrix[1] K as well as to the α parameters obtained from the stabilization problem in the Krĕin space, which can be solved using classical optimization tools on the flipped kernel matrix. This permits to apply the obtained model from the Krĕin space directly on the non-positive input kernel without any further modifications. The algorithm is shown in Algorithm 1. There are four major steps: 1) an eigen-decomposition of the full kernel matrix, with cubic costs (which can be potentially restricted to a few dominating eigenvalues - referred to as KSVM-L); 2) a flipping operation; 3) the solution of an SVM solver on the modified input matrix; 4) the application of the projection operator obtained from the eigen-decomposition on the α vector of the SVM model. U in Algorithm 1 contains the eigenvectors, D is a diagonal matrix of the eigenvalues and S is a matrix containing only $\{1, -1\}$ on the diagonal as obtained from the respective function sign.

Algorithm 1 Krĕin Space SVM (KSVM) - adapted from [16].

Krĕin SVM:
$[U, D]$:= EigenDecomposition(K)
$\hat{K} := USDU^\top$ with $S := sign(D)$
$[\alpha, b]$:= SVMSolver($\hat{K}$, Y, C)
$\tilde{\alpha} := USU^\top\alpha$ (now $\tilde{\alpha}$ is **dense**)
return $\tilde{\alpha}, b$;

As pointed out in [16], this solver produces an exact solution for the stabilization problem. The main weakness of this Algorithm is, that it requires the user to pre-compute the whole kernel matrix and to decompose it into eigenvectors/eigenvalues. Further today's SVM solvers have a theoretical, worst case complexity of $\approx \mathcal{O}(N^2)$. The other point to mention is that the final solution $\tilde{\alpha}$ is not sparse. The iCVM from [30] has a similar derivation and leads to a related decision function, again with a dense $\tilde{\alpha}$, but the model fitting costs are $\approx \mathcal{O}(N)$.

2. Sparsification of iCVM

2.1. Sparsification of iCVM by OMP

We can formalize the objective to approximate the decision function, which is defined by the $\tilde{\alpha}$ vector, obtained by KSVM or iCVM (both are structural identical), by a sparse alternative with the following mathematical problem:

$$\min \quad |\tilde{\alpha}|_0 \quad \text{s.t. } \textstyle\sum_m \tilde{\alpha}_m \Phi(x_m)^\top \Phi(x) \approx f(x)$$

It is well-known that this problem is NP hard in general, and a variety of approximate solution strategies exist in the literature. Here, we rely on a popular and very efficient approximation offered by **orthogonal matching pursuit** (OMP) [6,24]. Given an acceptable error $\epsilon > 0$ or a maximum number n of non-vanishing components of the approximation, a greedy approach is taken: the algorithm iteratively determines the most relevant direction and the optimum coefficient for this axes to minimize the remaining residual error.

In line 2 of Algorithm 2 we define the initial residuum to be the vector $K\tilde{\alpha}$ as part of the decision function. In line 4 we identify the most contributing dimension (assuming an empirical feature space representation of our kernel - it becomes the dictionary). Then in line 6 we find the current approximation of the sparse $\tilde{\alpha}$-vector - called $\tilde{\gamma}$ to avoid confusion, where $^+$ indicates the pseudo inverse. In line 7 we update the residuum by removing the approximated $K\tilde{\alpha}$ from the original unapproximated one. A Nyström based approximation of the Algorithm 2 is straight forward using the concepts provided in [7,31]. There it is also shown that the Nyström approximation holds for non-psd kernels, with a simplified proof given in [22]. With the Nyström technique a symmetric matrix psd [39] or non-pdf [7,31] is approximated by a low-rank approach using a subset of the original datapoints, called landmarks. As shown in [7,39] this approximation is exact if the rank of original data is smaller or equal to the number of landmarks. The landmarks are often chosen randomly, with more advanced strategies proposed e.g. in [20].

Algorithm 2 OMP to approximate the α vector.

1: **OMP:**
2: $I := \emptyset$; $r := y := K\tilde{\alpha}$; % initial residuum
3: **while** $|I| < n$ **do**
4: $l_0 := \text{argmax}_l |[Kr]_l|$; % find relevant direction + index
5: $I := I \cup \{l_0\}$ % track relevant indices
6: $\tilde{\gamma} := (K_{,I})^+ \cdot y$ % restricted (inverse) projection
7: $r := y - (K_{,I}) \cdot \tilde{\gamma}$ % residuum of the approximated decision function
8: **end while**
9: **return** $\tilde{\gamma}$ (as the new **sparse** $\tilde{\alpha}$)

2.2. Sparsification of iCVM by late subsampling

The parameters $\tilde{\alpha}$ are dense as already noticed in [16]. A naive sparsification by using only $\tilde{\alpha}_i$ with large absolute magnitude is not possible as can be easily checked by counter examples. One may now approximate $\tilde{\alpha}$ by using the (for this scenario slightly modified) OMP algorithm from the former section or by the following strategy, both compared in the experiments.

[1] Obtained by evaluating $k(x, y)$ for training points x, y.

As a second sparsification strategy we can use the approach suggested by Schleif and Tiňo [30], to restrict the projection operator and hence the transformation matrix of iCVM to a subset of the original training data. We refer to this approach as iCVM-sparse-sub.

To get a consistent solution we have to recalculate parts of the eigen-decomposition as shown in Algorithm 3. To obtain the respective subset of the training data we use the samples which are core vectors.[2] The number of core vectors is guaranteed to be very small [36] and hence even for a larger number of classes the solution remains widely sparse. The suggested approach is given in Algorithm 3. We assume that the original projection function (line 6 of Algorithm 3, detailed in [16]), is smooth and can be potentially restricted to a small number of construction points with low error. We observed that in general few construction points are sufficient to keep high accuracy, as seen in the experiments.

Algorithm 3 Sparsification of iCVM by late subsampling.

1: **Sparse iCVM:**
2: Apply iCVM - see (Schleif and Tiňo, 2017)
3: ζ - vector of projection points by using the core set points
4: construct a reduced K' using indices ζ as $\bar{K}$
5: [U,D] := EigenDecomposition($\bar{K}$); $S := sign(D)$
6: $\bar{\alpha} := USU^\top \alpha$ %U restricted to core set indices
7: $\tilde{\alpha} := 0 \quad \tilde{\alpha}_\zeta := \bar{\alpha}$ % assign $\bar{\alpha}$ to $\tilde{\alpha}$ using indices of ζ
8: $b := Y\tilde{\alpha}^\top$ % recalculate bias using the sparse $\tilde{\alpha}$
9: **return** $\tilde{\alpha}, b$;

3. Indefinite Core-Vector-Regression - iCVR

As already indicated in [30] the Krěin space approach considered before can also be used in similar minimum enclosing ball (MEB) based optimization problems. In particular we will consider the sparsification in the context of core vector regression for indefinite kernels, subsequently referred to as iCVR.

Assume points $x_i \in \mathbb{R}^d, i \in \{1, \ldots, N\}$ and real-valued outputs $y_i \in \mathbb{R}$ are given. Further, we assume a kernel function k (for the moment it is assumed this kernel is a psd kernel) is given with a feature map Φ. A kernel regression trains a function of the following form: $x \mapsto w^\top \Phi(x) + b$, where w is a normal vector of a decision plane and b a bias term. The training objective is to get as many points as possible approximately right while preserving a large margin. In the classical core vector regression (CVR) [36] an ϵ-tube formalization is used to achieve this objective. For an ϵ-tube a data point is correctly predicted iff its image is within ϵ of the desired value. The corresponding dual core vector regression is described by (for details see [36]):

$$\max_{\alpha_i, \alpha_i^* \geq 0, \sum \alpha_i + \alpha_i^* = 1} -\frac{1}{2} \cdot \sum_{i,j} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)(k_{ij} + 1)$$
$$-\frac{1}{2} \sum_{i=1}^{N} \alpha_i^2 / C - \frac{1}{2} \cdot \sum_{i=1}^{m} (\alpha_i^*)^2 / C + \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) y_i$$

This problem is of the form:

$$\max_{\alpha_i, \alpha_i^* \geq 0, \sum \alpha_i + \alpha_i^* = 1} -\frac{1}{2} \begin{pmatrix} \alpha \\ \alpha^* \end{pmatrix}^\top \tilde{K} \begin{pmatrix} \alpha \\ \alpha^* \end{pmatrix} + \begin{pmatrix} \alpha \\ \alpha^* \end{pmatrix}^\top \begin{pmatrix} y \\ -y \end{pmatrix} \tag{4}$$

where

$$\tilde{K} = \begin{pmatrix} K + \mathbf{1}\mathbf{1}^\top + 1/C \cdot \mathbf{I} & -(K + \mathbf{1}\mathbf{1}^\top) \\ -(K + \mathbf{1}\mathbf{1}^\top) & K + \mathbf{1}\mathbf{1}^\top + 1/C \cdot \mathbf{I} \end{pmatrix}$$

[2] A similar strategy for KSVM may be possible but is much more complicated because typically quite many points are support vectors and special sparse SVM solvers would be necessary.

$\tilde{K}$ is a valid kernel and as shown in [36] a core / MEB algorithm can be used to solve Eq. (4). If the underlying kernel function k is indefinite also $\tilde{K}$ becomes indefinite. We can now use the same argumentation as for iCVM [30] and following the work in [16] to modify the kernel $\tilde{K}$ by a flipping operation, to calculate a valid CVR model. Using the projection approach of [16] the obtained solution vector can again be mapped into the Krěin space to obtain a model for iCVR. This final solution does not need any kernel modification for new test points to be applied. The whole algorithmic workflow to derive a iCVR model is described in Algorithm 4. Once more a Nyström approximation can be employed in Algorithm 4 line 2, for an indefinite kernel using the concepts proposed in [7,31] and also in line 4 following [39] for psd kernel matrices.

Algorithm 4 Calculating a iCVR model.

1: **Indefinite CVR (iCVR):**
2: $[U, D]$ = EigenDecomposition(K)
3: $\hat{K} = USDU^\top$ with $S = sign(D)$
4: $[\alpha]$ = CoreVectorRegressionSolver($\hat{K}$, Y, C)
5: $\tilde{\alpha} = USU^\top \alpha \qquad b = Y\tilde{\alpha}^\top$
6: **return** $\tilde{\alpha}, b$;

One can easily see that the solution vector obtained in Algorithm 4 is non-sparse. We will therefore apply again the two post-processing approaches suggested before to sparsify the iCVR model. This can be done in the same way as for iCVM with results given in the experimental section.

4. Experiments - iCVM

This part contains a series of experiments that show that our approach leads to a substantially lower complexity, while keeping similar prediction accuracy compared to the non-sparse approach. To allow for large datasets with two much hassle we provide sparse results only for the MEB approaches, namely iCVM and iCVR. The modified OMP approach will work also for sparse KSVM or KSVR but the late sampling sparsification is not well suited if many support vectors are given in the original model, asking for a sparse SVM implementation. We follow the experimental design given in [16]. Methods that require to modify test data are excluded as also done in [16]. Finally we compare the experimental complexity of the different solvers. The used data are explained in Table 1. Additional larger data sets have been added to motivate our approach in the line of learning with large scale indefinite kernels.

4.1. Experimental setting

For each dataset, we have run 20 times the following procedure: A random split to produce a training and a testing set, a 5-fold cross validation to tune each parameter (the number of parameters depending on the method) on the training set, and the evaluation on the testing set. If $N > 1000$ we use $m = 200$ randomly chosen landmarks from the given classes to approximate the kernel matrix using the Nyström technique. If the input data are vectorial data we used a tanh kernel with parameters $[a = 1, r = 1]$ to obtain an indefinite kernel. Where tanh is given as: $k(x, y) = \tanh(a < x, y > +r)$.

4.2. Results

In Table 2 we show the results for large scale data (having at least 1000 points) using iCVM with sparsification. We observe much smaller models, especially for larger datasets with often comparable prediction accuracy with respect to the non-sparse

Table 1
Overview of the different datasets. We provide the dataset size (N) and the origin of the indefiniteness. For vectorial data the indefiniteness is caused artificial by using the tanh kernel.

Dataset	#samples	Proximity measure and data source
Sonatas	1068	normalized compression distance on midi files [29]
Delft	1500	dynamic time warping [29]
a1a	1605	tanh kernel [17]
zongker	2000	template matching on handwritten digits [26]
prodom	2604	pairwise structural alignment on proteins [26]
PolydistH57	4000	Hausdorff distance [26]
chromo	4200	edit distance on chromosomes [26]
Mushrooms	8124	tanh kernel [35]
swiss-10k	$\approx 10k$	smith waterman alignment on protein sequences [29]
checker-100k	100.000	tanh kernel (indefinite)
skin	245.057	tanh kernel (indefinite)[37]
checker	1 Mill	tanh kernel (indefinite)

Table 2
Prediction errors (mean ± std.-dev.) on the test sets. The percentage of projection points (pts) is calculated using the unique set over core vectors over all classes in comparison to all training points. All sparse-OMP models use only 10 points in the final models. Best results are shown in bold. Best sparse results are underlined. Datasets with substantially reduced prediction accuracy are marked by ⊙ (anova $p < 5\%$).

Dataset	#samples	iCVM (sparse-sub)	pts	iCVM (sparse-OMP)	iCVM (non-sparse)	Others
Sonatas	1068	**12.64 ± 1.71**	76.84%	22.56 ± 4.16⊙	13.01 ± 3.82	11.52 ± 0.20 [29]
Delft	1500	16.53 ± 2.79⊙	52.48%	3.27 ± 0.6	**3.20 ± 0.84**	1.80 ± 1.48[29]
a1a	1605	39.50 ± 2.88⊙	1.25%	27.85 ± 2.8	**20.56 ± 1.34**	17.08% [17]
zongker	2000	29.20 ± 2.48⊙	52.81%	7.50 ± 1.7	**6.40 ± 2.11**	4.4 ± 0.6 [26]
prodom	2604	2.89 ± 1.17	26.31%	3.12 ± 0.11	**0.87 ± 0.64**	1.3 ± 0.5 [26]
PolydistH57	4000	6.12 ± 1.38	12.92%	29.35 ± 8⊙	**0.70 ± 0.19**	5.4 ± 1.3 [26]
chromo	4200	11.50 ± 1.17	33.76%	**3.74 ± 0.58**	6.10 ± 0.63	7.7 ± 0.4 [26]
Mushrooms	8124	7.84 ± 2.21	6.46%	18.39 ± 5.7⊙	**2.54 ± 0.56**	–
swiss-10k	$\approx 10k$	35.90 ± 2.52⊙	17.03%	**6.73 ± 0.72**	12.08 ± 3.47	n.a.
checker-100k	100.000	**8.54 ± 2.35**	2.26%	19.54 ± 2.1⊙	9.66 ± 2.32	n.a.
skin	245.057	9.38 ± 3.30	0.06%	9, 43 ± 2.41	**4.22 ± 1.11**	n.a.
checker	1 Mill	8.94 ± 0.84	0.24%	**1.44 ± 0.3**	9.38 ± 2.73	n.a.

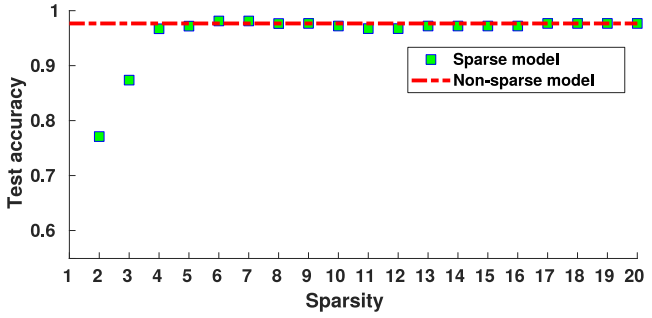


Fig. 1. Prediction results for the protein dataset using a varying level of sparsity and the OMP sparsity methods. For comparison the prediction accuracy of the non-sparse model is shown by a straight line.

model. The runtimes are similar to the non-sparse case but in general slightly higher due to the extra eigen-decompositions on a reduce set of the data as shown in Algorithm 3. But the focus is not on a faster runtime (which is linear for iCVM and iCVR), but on a simple, sparse model and hence an easy out of sample extension. A typical result for the protein data set using the OMP-sparsity technique and various values for sparsity is shown in Fig. 1.

5. Experiments - iCVR

We show the effectiveness of iCVR on a number of simulated and real life benchmark regression problems and compare with solutions as obtained by using standard CVR but for flipped (all signs of the eigenspectrum become positive) and clipped eigenspectra (negative einvalues are set to 0) of the respective kernel matrices. Data are given as $X \in \mathbb{R}^D$. Target function values $y_i \in \mathbb{R}^1$. The following one-dimensional simulated datasets have been used:

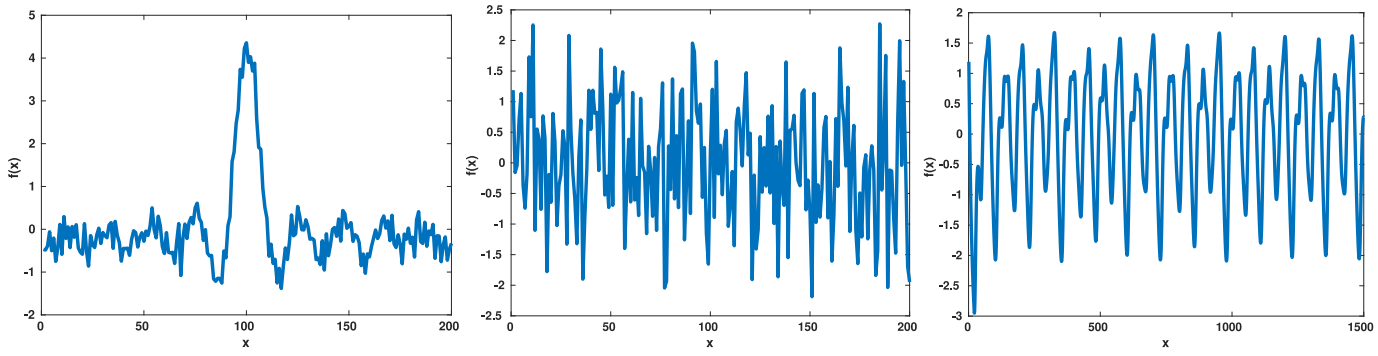
- (SIM1) basic sinc sample, with 200 samples, $f(x) = sinc(x/\pi) + 0.05 \cdot \sigma$ where σ is gaussian noise and x is linearly spread in $[-30, 30]$
- (SIM2) Friedman function, with 200 samples, $f(x) = 10 \cdot \sin(\pi \cdot \sigma_1 \cdot \sigma_2) + 20 \cdot (\sigma_3 - 0.5)^2 + 10 \cdot \sigma_4 + 5 \cdot \sigma_5 + \sigma$; and uniform noise $\sigma_1, \ldots, \sigma_5$, σ is gaussian noise
- (SIM3) The Mackey glass data,with 12000 samples, in 1 dimension as detailed in [18]

Further we used the following real life regression datasets.

- (DS1) Abalone - age prediction, with 4177 samples, $D = 8$ taken from [14]
- (DS2) Forest fires, with 517 samples, $D = 13$, dimension 13 was used as output variable, taken from [3]
- (DS3) Breast cancer (radius) prediction, with 569 samples, $D = 32$, dimension 3 was used as output variable, taken from [14] (wdbc)
- (DS4) White wine quality (scored 0–10), with 4898 samples, $D = 12$, dimension 12 was used as output variable, taken from [2][3]

The indefiniteness was caused using a Manhattan kernel $K_m = -||X - X^\top||$. The regression profiles for SIM1-SIM3 are depicted in Fig. 2. In the experiments we apply the iCVR approach on the given datasets and compare it with the standard CVR algorithm were the indefinite input kernel was corrected by applying a flip or clip eigenspectrum transformation.

[3] Available at: http://www3.dsi.uminho.pt/pcortez/wine/.



(a) SIM1 (b) SIM2 (c) SIM3

Fig. 2. Plots of the simulated data.

Table 3
Mean square error (mean ± std-dev.) in the 10-fold crossvalidation. The percentage of projection points (pts) is calculated using the unique set over core vectors over all classes in comparison to all training points. All sparse-OMP models use only 50 points in the final models. Best results are shown in bold.

Dataset	iCVR (non-sparse)	iCVR-flip	iCVR-clip	iCVR (sparse-sub)	pts	iCVR (sparse-OMP)	SVR-flip	SVR-clip
SIM1	**0.25 ± 0.12**	0.44 ± 0.43	0.46 ± 0.50	0.33 ± 0.13	17.25%	**0.25 ± 0.12**	0.11 ± 0.01	0.11 ± 0.03
SIM2	**0.14 ± 0.16**	0.15 ± 0.18	0.15 ± 0.16	0.15 ± 0.16	50%	0.15 ± 0.18	0.17 ± 0.04	0.18 ± 0.05
SIM3	**0.01 ± 0.00**	**0.01 ± 0.00**	**0.01 ± 0.00**	**0.01 ± 0.00**	18.68%	0.06 ± 0.01	≈0 ± ≈0	0.01 ± ≈0
DS1	0.83 ± 0.09	0.81 ± 0.07	2.46 ± 4.64	0.85 ± 0.06	7.83%	**0.77 ± 0.08**	**0.49 ± 0.04**	0.70 ± 0.34*
DS2	1.34 ± 0.57	1.19 ± 0.38	2.15 ± 0.61	1.57 ± 1.08	5.03%	**1.12 ± 0.15**	1.47 ± 0.23	1.85 ± 1.00*
DS3	**0.0 ± 0.0**	**0.01 ± 0.00**	**0.01 ± 0.00**	**0.02 ± 0.00**	23.04%	0.05 ± 0.00	**0.01 ± ≈0**	0.50 ± 0.31*
DS4	1.17 ± 0.28	1.24 ± 0.23	1.20 ± 0.16	1.29 ± 0.19	5.22%	**0.75 ± 0.05**	**0.67 ± 0.03**	0.70 ± 0.03

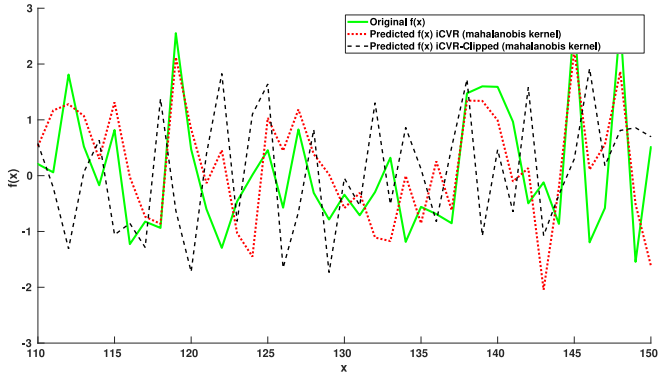


Fig. 3. Zoom in a plot of the Friedman output function (green, line). We also show the predicted output using CVR on a clipped mahalanobis kernel (black dashed+dotted) and a prediction of the output function using iCVR on the indefinite mahalanobis kernel (red, dashed). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

In Fig. 3 a plot of the output function for SIM2 and its prediction using iCVR and CVR on a clipped kernel is shown. The plot shows substantial prediction errors on the clipped kernel in contrast to the prediction of iCVR with the indefinite mahalanobis kernel. Considering the results shown in Table 3 we observe that the clipping is in general worse than flipping or the iCVR. The sparse models of iCVR are in general only slightly worse than the non-sparse model. In parts we can even see a better performance of the sparse iCVR model (see last column) compared to the iCVR. This may be due to a denoising effect, caused by the implicit low rank approach used in OMP.

It should be noted that an application of the standard CVR on the indefinite kernels is not possible, which was also experimentally verified, because the obtained problem becomes non-convex and the solver is unable to provide a solution to the optimization problem.

6. Complexity analysis

The original KSVM has runtime costs (with full eigendecomposition) of $\mathcal{O}(N^3)$ and memory storage $\mathcal{O}(N^2)$, where N is the number of points. The iCVM or respectively iCVR involves an extra Nyström approximation of the kernel matrix to obtain $K_{(N,m)}$ and $K^{-1}_{(m,m)}$, if not already given. If we have m landmarks, $m \ll N$, this gives memory costs of $\mathcal{O}(mN)$ for the first matrix and $\mathcal{O}(m^3)$ for the second, due to the matrix inversion. Further a Nyström approximated eigendecomposition has to be done to apply the eigenspectrum flipping operator. This leads to runtime costs of $\mathcal{O}(N \times m^2)$. The runtime costs for the sparse iCVM/iCVR are $\mathcal{O}(N \times m^2)$ and the memory complexity is the same as for iCVM/iCVR. Due to the used Nyström approximation the prior costs only hold if $m \ll N$, which is the case for many datasets as shown in the experiments.

The application of a new point to a KSVM, iCVM or iCVR model requires the calculation of kernel similarities to all N training points, for the sparse iCVM/iCVR this holds only in the worst case. In general the sparse iCVM/iCVR provides a simpler out of sample extension as shown in Table 2, but is data dependent.

The (i)CVM/(i)CVR model generation has not more than N iterations or even a constant number of 59 points, if the probabilistic sampling trick is used [34,36]. As show in [36] the classical CVM has runtime costs of $\mathcal{O}(1/\epsilon^2)$. The evaluation of a kernel function using the Nyström approximated kernel can be done with cost of $O(m^2)$ in contrast to constant costs if the full kernel is available. Accordingly, If we assume $m \ll N$ the overall runtime and memory complexity of iCVM/iCVR is linear in N, this is two magnitudes less as for KSVM for reasonable large N and for low rank input kernels.

7. Discussions and conclusions

As discussed in [16], there is no good reason to enforce positive-definiteness in kernel methods. A very detailed discussion on reasons for using KSVM or iCVM is given in [16], explaining why

a number of alternatives or pre-processing techniques are in general inappropriate. Our experimental results show that an appropriate Krĕin space model provides very good prediction results and using one of the proposed sparsification strategies this can also be achieved for a sparse model in most cases. The proposed iCVM-sparse-OMP is only slightly better than the former iCVM-sparse-sub model with respect to the prediction accuracy but has very few final modeling vectors, with an at least competitive prediction accuracy in the vast majority of data sets. Similar observations are found for the iCVR in comparison to CVR with flipping or clipping. As is the case for KSVM, the presented approach can be applied without the need for transformation of test points, which is a desirable property for practical applications.

Declaration of Competing Interest

There is no conflict of interest.

Acknowledgements

We thank Gaelle Bonnet-Loosli for providing support with the Krĕin Space SVM and R. Duin, Delft University for various support with distools and prtools. PT was supported by the EC Horizon 2020 ITN SUNDIAL (SUrvey Network for Deep Imaging Analysis and Learning), Project ID: 721463. FMS, CR are supported by the FuE program of the StMWi, project OBerA, grant number IUK-1709-0011//IUK530/010.

References

[1] I.M. Alabdulmohsin, M. Cissé, X. Gao, X. Zhang, Large margin classification with indefinite similarities, Mach. Learn. 103 (2) (2016) 215–237.
[2] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, J. Reis, Modeling wine preferences by data mining from physicochemical properties, Decis. Support Syst. 47 (4) (2009) 547–553.
[3] P. Cortez, A. Morais, A Data Mining Approach to Predict Forest Fires using Meteorological Data, in: J. Neves, M.F. Santos, J. Machado (Eds.), Proc. EPIA 2007, 2007, pp. 512–523.
[4] R.P.W. Duin, E. Pekalska, Non-euclidean dissimilarities: Causes and informativeness, in: SSPR&SPR 2010, 2010, pp. 324–333.
[5] M. Filippone, Dealing with non-metric dissimilarities in fuzzy central clustering algorithms, Int. J. of Approx. Reason. 50 (2) (2009) 363–384.
[6] Z.Z. Geoffrey M. Davis Stephane G. Mallat, Adaptive time-frequency decompositions, SPIE J. Opt. Eng. 33 (1) (1994) 21832191.
[7] A. Gisbrecht, F. Schleif, Metric and non-metric proximity transformations at linear costs, Neurocomputing 167 (2015) 643–657.
[8] S. Gu, Y. Guo, Learning SVM classifiers with indefinite kernels, in: Proc. of the 26th AAAI Conf. on AI, July 22–26, 2012.
[9] D. Gusfield, Algorithms on strings, trees, and sequences: Computer science and computational biology, Cambridge University Press, 1997.
[10] B. Haasdonk, Feature space interpretation of SVMs with indefinite kernels, IEEE TPAMI 27 (4) (2005) 482–492.
[11] B. Hassibi, Indefinite metric spaces in estimation, control and adaptive filtering, Stanford Univ., Dept. of Elec. Eng., Stanford, 1996 Ph.D. thesis.
[12] N. Higham, Computing a nearest symmetric positive semidefinite matrix, Linear Algebra Appl. 103 (C) (1988) 103–118.
[13] C. Hodgetts, U. Hahn, Similarity-based asymmetries in perceptual matching, Acta Psychol. 139 (2) (2012) 291–299.
[14] M. Lichman, UCI machine learning repository, 2013 http://archive.ics.uci.edu/ml.
[15] H. Ling, D.W. Jacobs, Shape classification using the inner-distance, IEEE Trans. Pattern Anal. Mach. Intell. 29 (2) (2007) 286–299.
[16] G. Loosli, S. Canu, C.S. Ong, Learning svm in Krĕin spaces, IEEE Trans. Pattern Anal. Mach. Intell. 38 (6) (2016) 1204–1216.
[17] R. Luss, A. d'Aspremont, Support vector machine classification with indefinite kernels, Math. Program. Comput. 1 (2–3) (2009) 97–118.
[18] M. Mackey, L. Glass, Oscillation and chaos in physiological control systems, Science 197 (4300) (1977) 287–289.
[19] F. van der Meer, The effectiveness of spectral similarity measures for the analysis of hyperspectral imagery, Int. J. Appl. Earth Obs. Geoinf. 8 (1) (2006) 3–17.
[20] C. Musco, C. Musco, Recursive sampling for the Nyström method, in: Proc. of NIPS 2017, 2017, pp. 3836–3848.
[21] E. Mwebaze, P. Schneider, F.-M. Schleif, Divergence based classification in learning vector quantization, Neurocomputing 74 (2010) 1429–1435.
[22] D. Oglic, T. Gärtner, Scalable learning in reproducing kernel Krĕin spaces, in: Proc. of the 36th Int. Conf. on ML, ICML 2019, Long Beach, California, USA, 2019, pp. 4912–4921.
[23] C.S. Ong, X. Mary, S. Canu, A.J. Smola, Learning with non-positive kernels, (ICML 2004), 2004.
[24] Y.C. Pati, R. Rezaiifar, P.S. Krishnaprasad, Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition, in: Proc. 27th Conf. on Signals, Sys. & Comp., 1, 1993, pp. 40–44.
[25] E. Pekalska, R. Duin, The Ddissimilarity Representation for Pattern Recognition, World Scientific, 2005.
[26] E. Pekalska, B. Haasdonk, Kernel discriminant analysis for positive definite and indefinite kernels, IEEE Trans. Pattern Anal. Mach. Intell. 31 (6) (2009) 1017–1031.
[27] H. Sakoe, S. Chiba, Dynamic programming algorithm optimization for spoken word recognition, Acoust. Speech Signal Process. IEEE Trans. 26 (1) (1978) 43–49.
[28] W.J. Scheirer, M.J. Wilber, M. Eckmann, T.E. Boult, Good recognition is non–metric, Pattern Recognit. 47 (8) (2014) 2721–2731.
[29] F. Schleif, P. Tiňo, Indefinite proximity learning: a review, Neural Comput. 27 (10) (2015) 2039–2096.
[30] F. Schleif, P. Tiňo, Indefinite core vector machine, Pattern Recognit. 71 (2017) 187–195.
[31] F.-M. Schleif, A. Gisbrecht, Data analysis of (non-)metric proximities at linear costs, in: Proceedings of SIMBAD 2013, 2013, pp. 59–74.
[32] D. Schnitzer, A. Flexer, G. Widmer, A fast audio similarity retrieval method for millions of music tracks, Multimedia Tools Appl. 58 (1) (2012) 23–40.
[33] J. Shawe-Taylor, N. Cristianini, Kernel Methods for Pattern Analysis and Discovery, Cambridge University Press, 2004.
[34] A.J. Smola, B. Schölkopf, Sparse greedy matrix approximation for machine learning, in: P. Langley (Ed.), Proceedings of the 17th Int. Conf. on Machine Learning (ICML 2000), Morgan Kaufmann, 2000, pp. 911–918.
[35] A. Srisuphab, J. Mitrpanont, Gaussian kernel approx. algorithm for feedforward nn design, Appl. Math. Comp. 215 (7) (2009) 2686–2693.
[36] I.W.-H. Tsang, J.T.-Y. Kwok, J.M. Zurada, Generalized core vector machines, IEEE TNN 17 (5) (2006) 1126–1140.
[37] UCI, Skin segmentation database, 2016 http://archive.ics.uci.edu/ml/datasets/Skin+Segmentation.
[38] V. Vapnik, The nature of statistical learning theory, Statistics for engineering and information science, Springer, 2000.
[39] C.K.I. Williams, M. Seeger, Using the Nyström Method to Speed Up Kernel Machines, in: Advances in Neural Information Processing Systems 13: NIPS'2000, 2000, pp. 682–688.
[40] W. Xu, R. Wilson, E. Hancock, Determining the cause of negative dissimilarity eigenvalues, LNCS 6854 LNCS (PART 1) (2011) 589–597.
[41] Z. Zhang, B.C. Ooi, S. Parthasarathy, A.K.H. Tung, Similarity search on bregman divergence: towards non-metric indexing, Proc. VLDB Endow. 2 (1) (2009) 13–24.