Brief Papers

Classification With Truncated ℓ_1 Distance Kernel

Xiaolin Huang, *Member, IEEE*, Johan A. K. Suykens, *Fellow, IEEE*, Shuning Wang, Joachim Hornegger, and Andreas Maier, *Member, IEEE*

***Abstract*— This brief proposes a truncated ℓ_1 distance (TL1) kernel, which results in a classifier that is nonlinear in the global region but is linear in each subregion. With this kernel, the subregion structure can be trained using all the training data and local linear classifiers can be established simultaneously. The TL1 kernel has good adaptiveness to nonlinearity and is suitable for problems which require different nonlinearities in different areas. Though the TL1 kernel is not positive semidefinite, some classical kernel learning methods are still applicable which means that the TL1 kernel can be directly used in standard toolboxes by replacing the kernel evaluation. In numerical experiments, the TL1 kernel with a pregiven parameter achieves similar or better performance than the radial basis function kernel with the parameter tuned by cross validation, implying the TL1 kernel a promising nonlinear kernel for classification tasks.**

***Index Terms*— Classification, indefinite kernel, piecewise linear (PWL).**

I. Introduction

LINEAR classifiers are the simplest classifiers and have good performance in many applications. When the problem becomes complicated, nonlinear classifiers are needed, which can be induced by nonlinear kernels such as polynomial kernel, tanh kernel, and radial basis function (RBF) kernel. One drawback is that the nonlinearity of those kernels is usually controlled by a hyperparameter, which has global effect and is not suitable for the case that requires different nonlinearities in different regions.

One way to fit nonlinearity in different regions is local classification [1]–[3]. They generally consist of two stages: domain partition and local training. For local training, it is popular to use linear classifiers because of computational effectiveness. Though the local classifiers are linear, the overall performance is nonlinear. Specifically, the result is piecewise linear (PWL), of which the nonlinearity can be different for different regions and the nonlinearity comes from the existence of subregions. Training a linear classifier in a subregion is relatively easy, and hence, the main challenge of PWL classification is to find a suitable domain partition, for which local learning applies K-nearest neighboring or clustering. Another way to construct PWL classifiers is to use PWL mappings [4]–[6], for which adjusting subregions is equal to tuning nonlinear parameters. Both methods are time-consuming and cannot fully use global information.

To efficiently find suitable subregions, we turn to PWL kernel learning methods, for which support vectors (SVs) provide subregions and non-SVs do not. In this way, it can be expected that the subregion is globally adjusted and the corresponding local linear classifier is simultaneously trained. Then, a suitable PWL kernel adaptively coincides nonlinearity requirement for different regions: for the heavily nonlinear part, there could be many SVs such that the boundary changes dramatically; on the other part, there are a few SVs and the boundary is flat. One interesting PWL kernel is designed in [7] and [8] and is named *the additive kernel*. Though the classifier constructed from the additive kernel is separable, it illustrates promising performance in some computer vision tasks, showing the potential advantages of PWL kernels.

In this brief, we propose to use the ℓ_1-norm as a distance measure and apply a truncation technique to construct a new PWL kernel. It is named a *truncated ℓ_1-distance (TL1)* kernel. In numerical experiments, the TL1 kernel shows surprisingly good performance. Due to the adaptiveness to nonlinearity, the TL1 kernel is less sensitive to its parameter. Thus, we can pregive the TL1 kernel a parameter value without cross validation and achieve similar or even better performance than the RBF kernel with the parameter chosen by cross validation. In view of local learning, training a classifier for the TL1 kernel is to globally find the partition and locally fit nonlinearity, which makes the TL1 kernel a promising alternative for the RBF kernel.

Besides good aspects of the TL1 kernel, there is a crucial problem that the TL1 kernel does not satisfy Mercer's condition, since the corresponding kernel matrices are not positive semidefinite (PSD). Fortunately, recent studies make non-PSD kernels usable without a major problem. The interested reader is referred to [11]–[20] for theoretical discussion and algorithms. In this brief, we will conclude some properties and choose suitable algorithms for the TL1 kernel, but leave the detailed learning analysis for further study.

The remainder of this brief is organized as follows. In Section II, the TL1 kernel is proposed and investigated. The training algorithms are discussed in Section III. Section IV evaluates the proposed TL1 kernel by numerical experiments. Section V ends this brief with a brief conclusion.

II. Truncated ℓ_1 Distance Kernel

A. Piecewise Linear Kernel

We in this brief focus on binary classification, for which the training set is denoted by $\mathcal{Z} = \{\mathbf{x}_i, y_i\}_{i=1}^m$ with $\mathbf{x}_i \in \mathbb{R}^n$ and $y_i \in \{-1, +1\}$. The margin-based kernel methods, such as the support vector machine (C-SVM) [21], the least squares support

Manuscript received May 19, 2016; revised September 29, 2016 and January 24, 2017; accepted February 7, 2017. Date of publication March 1, 2017; date of current version April 16, 2018. The work of X. Huang was supported in part by the Alexander von Humboldt Foundation and in part by the National Natural Science Foundation of China under Grant 61603248. The work of J. Suykens was supported in part by ERC AdG A-DATADRIVE-B under Grant 290923, Grant KUL: GOA/10/09 MaNet, Grant CoE PFV/10/002 (OPTEC), and Grant BIL12/11T, in part by FWO under Grant G.0377.12, Grant G.088114N, and Grant SBO POM (100031), and in part by IUAP P7/19 DYSCO. The work of S. Wang was supported by NSFC under Grant 61134012 and Grant 61473165.

X. Huang is with the Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, also with the MOE Key Laboratory of System Control and Information Processing, Shanghai 200400, China, and also with the Pattern Recognition Lab, the Friedrich-Alexander-Universität Erlangen-Nürnberg, 91058 Erlangen, Germany (e-mail: xiaolinhuang@sjtu.edu.cn).

J. A. K. Suykens is with the Department of Electrical Engineering (ESAT-STADIUS), KU Leuven, B-3001 Leuven, Belgium (email: johan.suykens@esat.kuleuven.be).

S. Wang is with the Department of Automation, Tsinghua University, Beijing 10084, China (e-mail: swang@mail.tsinghua.edu.cn).

J. Hornegger and A. Maier are with the Pattern Recognition Lab, the Friedrich-Alexander-Universität Erlangen-Nürnberg, 91058 Erlangen, Germany (e-mail: joachim.hornegger@fau.de; andreas.maier@fau.de).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2017.2668610

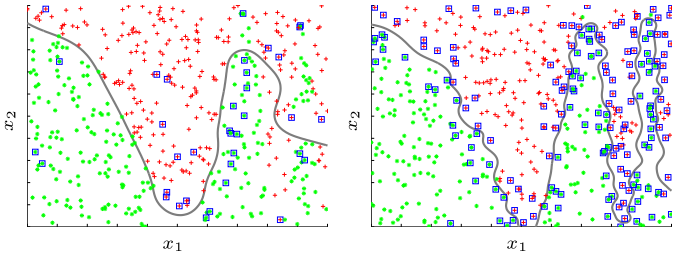

Fig. 1. Classification boundaries for data set "cosexp" are displayed by solid curves and the SVs are marked by squares. Left: RBF kernel with $\sigma = 1$. Right: RBF kernel with $\sigma = 0.2$.

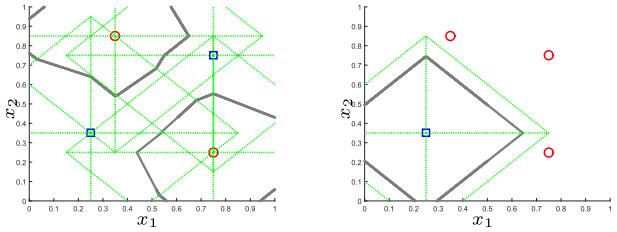

Fig. 2. Potential subregion boundaries are shown by dotted lines, and a possible classification boundary that has tuning points when meeting subregion boundaries is shown by black solid lines. Left: four SVs. Right: one SV.

vector machine (LS-SVM) [22], and the linear programming support vector machine (LP-SVM) [23], have been widely and successfully applied. For a kernel $\mathcal{K}(\mathbf{u}, \mathbf{v})$, those methods train the corresponding kernel coefficients $\{\alpha_i\}_{i=1}^m$ and an offset α_0 to construct the following classification function:

$$f(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i y_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}) + \alpha_0. \tag{1}$$

Obviously, a nonlinear kernel leads to a nonlinear classifier. Consider the RBF kernel $\mathcal{K}(\mathbf{u}, \mathbf{v}) = \exp(-\|\mathbf{u} - \mathbf{v}\|_{\ell_2}^2/\sigma^2)$, where the bandwidth σ plays an important role for determining the nonlinearity. When σ is large, the classification curve is flat, or equally speaking, the nonlinearity is light. If heavy nonlinearity is needed, i.e., the ideal classification curve has large curvature, then a small σ performs well. However, for the situation that the ideal classifier has heavy nonlinearity on one part but performs linearly on another part, it is hard to find a suitable σ value. As a simple example, we use LP-SVM (see Section III) with the RBF kernel to establish a classifier for a 2-D problem "cosexp." In Fig. 1, the sampling points in two classes are displayed by red crosses and green stars, respectively. It can be expected that the ideal classifier changes smoothly on the left part of the domain and has sharp vertices on the right part. To pursue a flat classification boundary, a relatively large σ is preferred. We set $\sigma = 1$ and display the obtained decision boundary by a solid curve in Fig. 1(left). The left part is classified quite well but the right part is not. To achieve high accuracy for the right part, we need to decrease σ value to, e.g., $\sigma = 0.2$. The obtained classification boundary is plotted in Fig. 1 (right). It has better performance on the right part but is not as flat as we expect on the left part, where there are many SVs in order to follow the desired trend.

One possibility to handle this problem is to use a PWL classifier, which equals a linear classifier in one subregion and all the subregions tessellate the whole domain. Since a PWL classifier performs linearly in each subregion, the nonlinearity replies on the subregion structure. In parametric PWL models [5], [30], [31], the subregion structure is determined by nonlinear parameters and hard to tune. In local classification methods [1]–[3], it is obtained by clustering or neighbor search, which do not use the global information.

To adaptively find a suitable subregion structure with all training data, we propose to use a *PWL kernel* $\mathcal{K}(\mathbf{u}, \mathbf{v})$ that is PWL to $\mathbf{u}$ and $\mathbf{v}$, respectively. One pioneering work is given in [7] and [8], which designed the following additive kernel:

$$\mathcal{K}(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^{n} \min\{\mathbf{u}(i), \mathbf{v}(i)\}. \tag{2}$$

Suppose that $\mathbf{x}_i$ is an SV, then it partitions the domain into hypercubes $\{\mathbf{x} : \mathbf{x}(j) \geq (\text{or } \leq)\ \mathbf{x}_i(j), \forall j\}$. If $\alpha_i = 0$, then there is no subregion boundary and the classifier keeps linear around $\mathbf{x}_i$. Since α_i can be trained by kernel learning methods using all training data, one essentially gets a globally trained subregion structure and the resulted classifier can adaptively fit local nonlinearity. Thus, the additive kernel gets success in some imaging process tasks. However, the potential subregion structure induced from the additive kernel is not flexible enough, which can be seen from that the classifiers based on the additive kernel are separable, i.e., they can be written as the sum of several univariate PWL functions. Therefore, the performance of the additive kernel on problems with strongly coupled variables is not good, and in general problems, it is not comparable with the RBF kernel.

B. TL1 Kernel

We want to have a more flexible PWL kernel for classification tasks. For this aim, the RBF kernel, the currently most popular nonlinear kernel, is first investigated. The RBF kernel $\mathcal{K}(\mathbf{u}, \mathbf{v}) = \exp(-\|\mathbf{u} - \mathbf{v}\|_{\ell_2}^2/\sigma^2)$ is a composition of two operators: 1) $\|\mathbf{u}-\mathbf{v}\|_{\ell_2}$ measures the distance and 2) $\exp(-a^2/\sigma^2)$ normalizes the distance to a bounded value. Following this idea, we use two PWL functions to establish a PWL kernel: the ℓ_1-norm to measure the distance and a triangle function to normalize the distance. Specifically, we propose the following kernel:

$$\mathcal{K}(\mathbf{u}, \mathbf{v}) = \max\{\rho - \|\mathbf{u} - \mathbf{v}\|_{\ell_1}, 0\}. \tag{3}$$

Since $\max\{a, 0\}$ is a truncation operator, (3) is named *truncated* ℓ_1 *distance (TL1)* kernel. Obviously, the truncation operator and the ℓ_1-norm distance are both PWL, and their composition hence gives a PWL function. Both the truncation operator and the ℓ_1-norm distance are one-level maximum operators, and thus, the TL1 kernel (3) is a two-level deep PWL model. In contrast, the additive kernel (2) is a one-level deep PWL model.

In (3), there is a kernel parameter ρ, which is the truncation threshold: when the ℓ_1 distance between $\mathbf{u}$ and $\mathbf{v}$ is larger than ρ, the corresponding kernel value is zero. Throughout this brief, we assume that each component of the training data is in the range of $[0, 1]$. If not, one can easily normalize the training data into $[0, 1]^n$. In this case, the maximum value of $\|\mathbf{u}-\mathbf{v}\|_{\ell_1}$ is n and the minimum value is zero. Thus, a reasonable value of ρ is between $[0, n]$. ρ also controls the sparsity of the kernel matrix: a smaller ρ tends to have more SVs, similar to the kernel width σ in the RBF kernel.

In Fig. 2 (left), potential subregion boundaries corresponding to four SVs (blue squares and red circles stand for two classes) are shown by green dotted lines. The subregion structure can result in a very flexible decision boundary. In a 2-D space, the boundary consists of a series of segments. In Fig. 2 (left), one possible boundary is plotted by solid lines that easily solve this XOR problem. This boundary is linear in each subregion and has tuning points on the junctions of subregions. When the problem is simple, e.g., in Fig. 2 (right), the top-right SV has the same label as the two nearby ones, training procedure gives a sparse result that there is only one SV, i.e., the left-bottom point. Then, the subregion structure

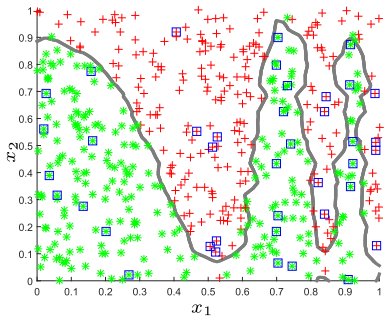

Fig. 3. Classification result for "cosexp" using the TL1 kernel. Nonlinearity is adjusted by SVs (marked by squares), which determine the boundaries of different linear subregions. On the left part, there are a few SVs; on the right part, the density of SVs is high. The performance of the RBF kernel with different signal kernel parameters can be found in Fig. 1.

and the potential classification boundary become simple, as shown in Fig. 2 (right).

The above explanation shows the flexibility of the classifier induced by the TL1 kernel. Considering the "cosexp" data set shown in Fig. 1, we now use the TL1 kernel and LP-SVM to train a classifier. The classifier and the SVs are displayed in Fig. 3, from which one can find the mechanism of nonlinearity adjustment: on the left part, there are a few SVs and the classification curve is flat; on the right part, the classifier is heavily nonlinear due to the existence of many SVs.

C. Properties of the TL1 Kernel

The proposed TL1 kernel is a PWL, compactly supported, and indefinite kernel:

1) *Piecewise Linearity:* The classification function (1) induced by the TL1 kernel is continuous and PWL with respect to $\mathbf{x}$, which also means that the classification boundary $f(\mathbf{x}) = 0$ is continuous and PWL. The continuity comes from the fact that the composition of two continuous functions is continuous, which is valid for PWL functions as well. This property could be extended to a deeper structure, i.e., a multilayer kernel with nested-TL1 kernel is PWL. With more nested levels, the kernels will have more complicated subregion structures and then have more flexibility. Its training can draw from the experience of deep convolutional networks [24] where the maximum operator is used as the basic computational unit. In this brief, we first focus on the TL1 kernel.
2) *Compact Support:* The TL1 kernel is a compactly supported kernel. A compactly supported kernel means that the corresponding supports are limited to a finite window [25]. The TL1 kernel is compactly supported because of the use of the triangle operator, which is a common way to obtain locally supported kernels. Via deforming the whole space to an ℓ_2-ball, compactly supported RBF kernel, compactly supported polynomial kernel, Epanechnikov kernel, and triangular kernel, have been established [25]– [29]. Different from these kernels that are based on the ℓ_2-norm, the proposed TL1 kernel is to use the ℓ_1 distance, which leads to polygonal subregions and linear performance in subregion. Many existing results on compact supported kernels are also applicable to the TL1 kernel. However, the analysis that relies on smoothness and positive semidefiniteness is not valid for the TL1 kernel and this extension is a challenge.
3) *Indefiniteness:* In a space higher than one dimension, the TL1 kernel is not PSD. Being an indefinite kernel makes the learning behavior of the TL1 kernel questionable. In fact, the formulation for the TL1 kernel has appeared in [26] and [27], however, as a counterexample because of the lack of positive definiteness, which prevented researchers to seriously investigate the TL1 kernel and evaluate its performance in the past. In Section III, we will discuss the recent advances on indefinite learning, with which it is possible to use the TL1 kernel nowadays.

III. Indefinite Learning Algorithms

For indefinite kernels including the proposed TL1 kernel, researchers need to carefully investigate the training algorithms. One important issue is that the functional space induced from an indefinite kernel is not a reproducing kernel Hilbert space (RKHS). Instead, it is a reproducing kernel Kreĭn space (RKKS) and the readers are referred to [18] and references therein for detailed discussion. As proved in [11], a general representer theorem is still true and a regularized risk can be defined in RKKS. Specifically, a classifier, which is generated by an indefinite kernel and stabilizes one regularized risk, also has the same expression (1). Therefore, when the training data are given, learning for the TL1 kernel still can be formulated as a finite-dimensional optimization problem on α_i.

According to the representer theorems for RKHS and RKKS, learning with PSD or indefinite kernels can be concluded in the same optimization framework

$$\min_{f\in\mathcal{S}_{\mathcal{K},\mathcal{Z}},\alpha_0\in\mathbb{R}} \ \frac{1}{2}\Omega(f) + C\sum_{i=1}^{m} L(1 - y_i(f(\mathbf{x}_i) + \alpha_0)) \tag{4}$$

where Ω is the regularization term, L is the classification loss function, $C > 0$ is the tradeoff coefficient between them, and $\mathcal{S}_{\mathcal{K},\mathcal{Z}} = \{f : f(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i y_i \mathcal{K}(\mathbf{x}, \mathbf{x}_i)\}$ is a finite-dimensional space spanned by $\mathcal{K}$ and $\mathcal{Z}$.

If $\mathcal{K}$ satisfies Mercer's condition, there is a nonlinear feature mapping $\phi(\mathbf{x})$, of which the outputs could be in an infinite dimensional space, such that $\mathcal{K}(\mathbf{u}, \mathbf{v}) = \phi(\mathbf{u})^T\phi(\mathbf{v})$. In this case, one can choose a norm induced by $\mathcal{K}$ as the regularization term, choose the hinge function as the loss, and then formulate (4) as the well-known C-SVM [21]. Its dual problem is

$$\begin{aligned} \min_{\alpha} \ & \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} y_i\alpha_i\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)\alpha_j y_j - \sum_{i=1}^{m}\alpha_i \\ \text{s.t.} \ & \sum_{i=1}^{m} y_i\alpha_i = 0, \quad 0 \le \alpha_i \le C \quad \forall i. \end{aligned} \tag{5}$$

There have been many algorithms to solve (5). For instance, when the sampling data size is not very huge, the sequential minimization optimization (SMO) developed in [32] can efficiently find the optimal dual variables.

However, if indefinite kernels, such as the TL1 kernel, are used, we cannot find the corresponding feature mapping, i.e., there is no kernel trick. Moreover, (5) is no longer convex. An alternative way is to find an approximate PSD kernel $\tilde{\mathcal{K}}$ for an indefinite one $\mathcal{K}$, and then solve (5) for $\tilde{\mathcal{K}}$. $\tilde{\mathcal{K}}$ can be constructed by eigenvalue modification [9] or an additional optimization problem [13]. Since training and classification are based on two different kernels, the above methods are working only when $\mathcal{K}$ and $\tilde{\mathcal{K}}$ are similar. To overcome the inconsistency, an eigendecomposition support vector machine was established in [18]. This method was reported to outperform the other PSD approximate methods.

The above methods which transfer the non-PSD kernels to PSD ones have been reviewed and compared in [19]. That review also discusses another type of indefinite learning which directly uses non-PSD kernels in the methods that are insensitive to metric violations. For example, in the dual problem (5), we can directly

TABLE I
AVERAGE ACCURACY, STANDARD DEVIATION, AND NUMBER OF SUPPORT VECTORS

dataset	m	RBF kernel C-SVM	RBF kernel LS-SVM	TL1 kernel C-SVM	TL1 kernel LS-SVM	TL1 kernel LP-SVM
DBWords	32	$84.81 \pm 5.79\%$ (31)	$85.01 \pm 6.45\%$ (32)	$\underline{85.75} \pm 6.41\%$ (30)	$85.01 \pm 5.51\%$ (32)	$84.38 \pm 6.12\%$ (20)
Fertility	50	$88.56 \pm 2.48\%$ (49)	$88.45 \pm 2.67\%$ (50)	$88.84 \pm 2.45\%$ (19)	$\underline{89.43} \pm 2.84\%$ (50)	$88.25 \pm 2.84\%$ (3)
Spect	80	$77.54 \pm 3.11\%$ (77)	$82.35 \pm 1.07\%$ (80)	$83.42 \pm 2.47\%$ (68)	$83.03 \pm 1.37\%$ (80)	$\underline{84.49} \pm 1.45\%$ (30)
Planning	91	$70.76 \pm 2.99\%$ (66)	$71.48 \pm 3.77\%$ (91)	$71.09 \pm 4.38\%$ (54)	$72.03 \pm 4.32\%$ (91)	$\underline{72.23} \pm 3.23\%$ (31)
Sonar	104	$83.24 \pm 3.52\%$ (72)	$\underline{83.33} \pm 2.66\%$ (104)	$83.04 \pm 3.55\%$ (74)	$83.00 \pm 4.02\%$ (103)	$82.11 \pm 3.43\%$ (51)
Statlog	135	$82.96 \pm 2.22\%$ (71)	$82.76 \pm 2.47\%$ (135)	$83.09 \pm 2.37\%$ (76)	$\underline{83.53} \pm 1.84\%$ (135)	$83.32 \pm 0.68\%$ (10)
Monk1	169	$81.02 \pm 1.89\%$ (65)	$79.05 \pm 1.35\%$ (124)	$81.25 \pm 1.69\%$ (80)	$\underline{85.19} \pm 0.00\%$ (124)	$83.33 \pm 2.11\%$ (36)
Monk2	169	$86.34 \pm 1.71\%$ (102)	$\underline{87.59} \pm 0.62\%$ (169)	$86.57 \pm 1.40\%$ (130)	$86.57 \pm 2.01\%$ (169)	$86.57 \pm 1.93\%$ (69)
Monk3	122	$93.59 \pm 0.73\%$ (42)	$93.74 \pm 0.59\%$ (122)	$\underline{97.22} \pm 0.00\%$ (37)	$\underline{97.22} \pm 0.00\%$ (122)	$96.30 \pm 0.70\%$ (20)
Climate	270	$\underline{94.28} \pm 1.78\%$ (113)	$93.25 \pm 1.22\%$ (270)	$93.82 \pm 1.42\%$ (67)	$92.07 \pm 0.72\%$ (270)	$93.66 \pm 1.89\%$ (46)
Liver	292	$72.59 \pm 1.86\%$ (289)	$71.38 \pm 2.60\%$ (292)	$72.66 \pm 2.36\%$ (290)	$72.02 \pm 2.08\%$ (292)	$\underline{73.13} \pm 1.66\%$ (80)
Austr.	345	$84.75 \pm 1.34\%$ (133)	$85.08 \pm 1.98\%$ (345)	$\underline{86.07} \pm 2.01\%$ (117)	$85.90 \pm 1.70\%$ (345)	$85.62 \pm 2.11\%$ (65)
Breast	349	$96.58 \pm 0.65\%$ (46)	$96.30 \pm 0.69\%$ (349)	$\underline{96.87} \pm 0.70\%$ (83)	$\underline{96.87} \pm 0.61\%$ (349)	$96.38 \pm 0.52\%$ (16)
Trans.	374	$76.53 \pm 1.76\%$ (193)	$77.60 \pm 1.27\%$ (374)	$\underline{77.87} \pm 1.07\%$ (193)	$76.91 \pm 1.52\%$ (374)	$75.53 \pm 1.63\%$ (9)

use a non-PSD $\mathcal{K}$ as suggested in [10]. Then, there are two crucial problems arising. First, there is no feature mapping ϕ such that $\mathcal{K}(\mathbf{u}, \mathbf{v}) = \phi(\mathbf{u})^T\phi(\mathbf{v})$, from which it follows that C-SVM with a non-PSD kernel cannot be explained as maximizing the margin in a feature space. Instead, it is to minimize the distance between two convex hulls in a pseudo-Euclidean space. The reader is referred to [11] and [12] for this important interpretation. The second problem is that a non-PSD kernel makes (5) nonconvex, and many algorithms cannot be used if they rely on global optimality. Fortunately, descent algorithms, such as SMO, are still applicable, and then, a stationary point can be achieved efficiently.

Comparing the above two categories of indefinite learning methods, we find that transferring a non-PSD kernel to an approximate PSD one takes significantly longer time than directly training. Thus, for the aim of practical use, we choose SMO to solve (5) for the proposed TL1 kernel. This is also a fair comparison with the RBF kernel, which is commonly trained by (5) with SMO. Notice that when $\mathcal{K}$ in (5) is not PSD, the result of SMO could differ for different starting points. Since we prefer a sparse solution for a low model complexity, we suggest the zero vector as the initial solution.

The approach of directly using a non-PSD kernel is also applicable to LS-SVM [22], another popular kernel learning method. In the dual space, LS-SVM is to solve the following linear system:

$$\begin{bmatrix} 0 & \mathbf{y}^T \\ \mathbf{y} & \mathcal{H} + \dfrac{1}{C}\mathcal{I} \end{bmatrix} [\alpha_0, \alpha_1, \ldots, \alpha_m]^T = \begin{bmatrix} 0 \\ \mathbf{1} \end{bmatrix} \tag{6}$$

where $\mathcal{I}$ is an identity matrix, $\mathbf{1}$ is an all ones vector with the proper dimension, and $\mathcal{H}$ is given by $\mathcal{H}_{ij} = y_i y_j \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$. Similarly to the discussion about C-SVM, we directly use the TL1 kernel in (6) as well. Different to C-SVM (5), where a non-PSD kernel makes that problem nonconvex, using a non-PSD kernel in LS-SVM (6) is still easy to solve. But the kernel trick is not valid and theoretical discussions on LS-SVM with non-PSD kernels could be found in [20].

One drawback of (6) is that its results lack of sparsity. We expect that for the flat part of the ideal classifier, a PWL classifier can adaptively perform linearly, which requires sparsity on α. To pursue sparsity, the ℓ_1-norm and the hinge loss are chosen in the framework of (4), leading to the following LP-SVM (see [23] for PSD kernels and [17] for indefinite kernels):

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{m}|\alpha_i| + C\sum_{i=1}^{m}\max\left\{0, 1 - y_i\left(\sum_{j=1}^{m} y_j\alpha_j\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) + \alpha_0\right)\right\}. \tag{7}$$

IV. NUMERICAL EXPERIMENTS

In Section II and III, we proposed the TL1 kernel and discussed three potential training algorithms. This section will compare the TL1 kernel with the RBF kernel. The data are downloaded from UCI Repository of Machine Learning data sets [33] and LibSVM data sets [34]. For some problems, both the training and test sets are provided. Otherwise, we randomly pick half of the data as the training set and use the remaining for test. We will report the average accuracy, the standard deviation, and the average number of SVs (if $|\alpha_i| > 10^{-3}$) on ten trials. Kernel parameters and regularization coefficients are tuned by tenfold cross validation based on misclassification. As discussed before, the reasonable range of ρ for the TL1 kernel is between 0 and n and the value candidate set is $\{0.1n, 0.3n, 0.5n, 0.7n, 0.9n\}$. For C-SVM, LS-SVM, and LP-SVM, the regularization coefficients are chosen from {0.01, 0.1, 0.5, 1, 2, 10}. All the experiments are done with MATLAB 2014b on Windows 7 with Core i5-3.10 GHz and 8-GB RAM.

There have been efficient and popular toolboxes for the RBF kernel, e.g., libsvm [34] for C-SVM (5) and LS-SVMlab [35] for LS-SVM (6). The kernel parameter σ is crucial to its performance and we apply the automatic parameter selection provided by [36] to find suitable parameters for LibSVM and apply the default tuning process for LS-SVMlab. In Table I, the performance for small-size problems is reported, and the highest accuracy is underlined.

Comparing the RBF and the TL1 kernel with the same training algorithm, we find that the TL1 kernel gives at least a comparable result for each data set, which confirms the universality of the TL1 kernel. Moreover, on some data sets, like "Spect," "Monk3," and "Austr.," the performance is improved with respect to the RBF kernel.

PWL kernel learning is motivated by the adaptiveness for nonlinearity in different areas. This also implies that the classification performance is less sensitive to the kernel parameter, and then for the TL1 kernel, one parameter value is suitable for different nonlinearities. Thus, for different problems, we can use a pregiven ρ without tuning process. In order to verify this property, classification accuracy versus parameter value is plotted in Fig. 4, where C-SVM is used.

In the three data sets, the classification accuracy of the TL1 kernel is similar for a quite large range of ρ values. The performance for ρ between $0.6n$ to $0.9n$ is generally stable and setting $\rho = 0.7n$ always leads to a good result. In contrast, the best σ of the RBF kernel for different problems differs a lot: for "Monk3", the best σ is around 0.15; for "Austr.," it is about 0.7, and for "Breast", it is about 0.02. The stability of ρ in the TL1 kernel allows us to prechose

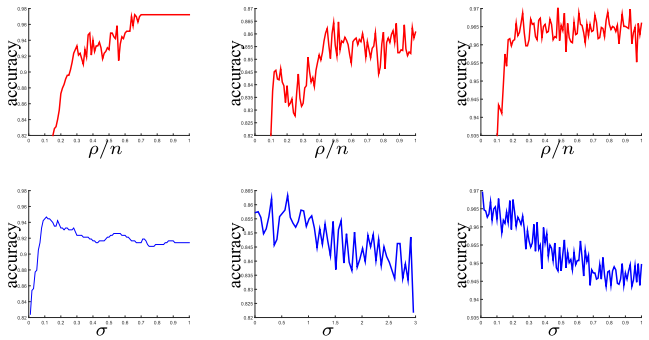

Fig. 4. Classification accuracy on test data versus kernel parameters (top: the TL1 kernel; bottom: the RBF kernel) for different data sets (left: "Monk3"; middle: "Austr."; right: "Breast"). For the considered three data sets, setting $\rho = 0.7\ n$ for the TL1 kernel can give satisfactory results, while for the RBF kernel, the suitable value for σ varies a lot for different problems.

TABLE II
AVERAGE TIME FOR TUNING PARAMETERS (IN SECONDS)

dataset	RBF kernel (σ by CV)			TL1 kernel ($\rho = 0.7n$)		
	C-SVM	LS-SVM	LP-SVM	C-SVM	LS-SVM	LP-SVM
Monk3	3.09	0.62	92.7	0.89	0.08	47.2
Austr.	6.02	5.44	160	3.74	0.57	86.2
Breast	7.62	6.14	168	1.34	0.62	130

TABLE III
AVERAGE ACCURACY, STANDARD DEVIATION, AND NUMBER OF SVs

dataset	m	RBF kernel by C-SVM (σ by cross-validation)	TL1 kernel by C-SVM ($\rho = 0.7n$)
Qsar	528	$\underline{86.92} \pm 1.31\%$ (209)	$86.05 \pm 1.21\%$ (300)
Splice	1000	$89.83 \pm 0.09\%$ (651)	$\underline{92.74} \pm 0.02\%$ (229)
Guide3	1243	$84.15 \pm 3.45\%$ (482)	$\underline{97.56} \pm 0.00\%$ (487)
Madelon	2000	$58.83 \pm 0.00\%$ (1852)	$\underline{61.33} \pm 0.00\%$ (1845)
Spamb.	2300	$93.32 \pm 0.60\%$ (464)	$\underline{94.05} \pm 0.56\%$ (453)
ML-prove	3059	$72.48 \pm 0.32\%$ (1604)	$\underline{79.08} \pm 0.00\%$ (1684)
Guide1	3089	$96.84 \pm 0.16\%$ (308)	$\underline{97.12} \pm 0.04\%$ (397)
Wilt	4339	$85.80 \pm 0.74\%$ (130)	$\underline{86.80} \pm 0.44\%$ (475)
Phish.	5528	$\underline{95.92} \pm 0.30\%$ (2136)	$93.83 \pm 0.48\%$ (1657)
Magic	9510	$\underline{86.48} \pm 0.45\%$ (3124)	$86.04 \pm 0.43\%$ (4342)
RNA	59535	$\underline{96.66} \pm 0.20\%$ (7072)	$95.74 \pm 0.22\%$ (13656)

it without cross validation. This is particularly useful when the data size is large. To illustrate this benefit, we report parameter-tuning time for the RBF kernel with $\sigma \in \{0.1, 0.5, 1, 5, 10\}$ and the TL1 kernel with $\rho = 0.7n$. Besides, there is a regularization coefficient to be tuned by cross validation. Fig. 4 shows the classification accuracy and Table II gives the parameter-tuning time.

In the following experiments, we fix $\rho = 0.7n$ for the TL1 kernel and compare it with the RBF kernel with a parameter tuning process. For the aim of practical applications, we use C-SVM for the TL1 kernel for larger data sets. The classification performance of the TL1 kernel with $\rho = 0.7n$ is reported in Table III, together with the results of LibSVM for the RBF kernel, of which the kernel parameter is tuned by cross validation.

In Table III, the TL1 kernel shows pretty good performance. For data set "Splice," "ML-prove," and "Guide3," the classification accuracy is improved from the RBF kernel. For other problems, the performance of the two kernels is similar. Notice that in this experiment, the parameter of the RBF kernel is tuned by cross validation, while that of the TL1 kernel is set to be $\rho = 0.7n$. With one parameter less to tune, the computation time can be largely saved.

V. CONCLUSION

In this brief, we designed the truncated ℓ_1 distance kernel and evaluated it on numerical experiments. The TL1 kernel results in a PWL classifier, for which the subregion structure can be trained by kernel learning methods. Though the TL1 kernel is not PSD, the numerical experiments show that three popular kernel learning methods, including C-SVM, LS-SVM, and LP-SVM, can be directly applied to train classifiers for the TL1 kernel. With these training algorithms, the TL1 kernel illustrates good performance of global partitioning and local linear training. The adaptiveness to nonlinearity also makes the TL1 kernel less sensitive to its kernel parameter. Therefore, without parameter tuning, we can use a pregiven parameter for the TL1 kernel and achieve similar performance as the RBF kernel with cross validation. In various experiments, the TL1 kernel is comparable with the RBF kernel, and for some, it significantly outperforms the RBF kernel, which indicates that the TL1 kernel is a promising nonlinear kernel for classification. It can be implemented in popular classification toolboxes, such as LibSVM and LS-SVMlab, by simply changing the kernel evaluation function. With further investigations, other techniques, such as random features, Nyström approximation, and fixed-size method, are also possible to use the TL1 kernel.

ACKNOWLEDGMENT

The authors would like to thank Dr. Y. Feng in KU Leuven for discussions on local kernels. They would also like to thank the anonymous reviewers for their helpful comments.

REFERENCES

[1] J. Sklansky and L. Michelotti, "Locally trained piecewise linear classifiers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 2, no. 2, pp. 101–111, Mar. 1980.
[2] H. Zhang, A. C. Berg, M. Maire, and J. Malik, "SVM-KNN: Discriminative nearest neighbor classification for visual category recognition," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2006, pp. 2126–2136.
[3] N. Segata and E. Blanzieri, "Fast and scalable local kernel machines," *J. Mach. Learn. Res.*, vol. 11, pp. 1883–1926, Mar. 2010.
[4] Y. Li, B. Liu, X. Yang, Y. Fu, and H. Li, "Multiconlitron: A general piecewise linear classifier," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 276–289, Feb. 2011.
[5] X. Huang, S. Mehrkanoon, and J. A. K. Suykens, "Support vector machines with piecewise linear feature mapping," *Neurocomputing*, vol. 117, pp. 118–127, Oct. 2013.
[6] Y. Li and Q. Leng, "Alternating multiconlitron: A novel framework for piecewise linear classification," *Pattern Recognit.*, vol. 48, no. 3, pp. 968–975, 2015.
[7] S. Maji, A. C. Berg, and J. Malik, "Classification using intersection kernel support vector machines is efficient," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2008, pp. 1–8.
[8] S. Maji, A. C. Berg, and J. Malik, "Efficient classification for additive kernel SVMs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 66–77, Jan. 2013.
[9] E. Pekalska, P. Paclik, and R. P. W. Duin, "A generalized kernel approach to dissimilarity-based classification," *J. Mach. Learn. Res.*, vol. 2, pp. 175–211, Mar. 2002.
[10] 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," Dept. Comput. Sci., Nat. Taiwan Univ., Taipei, Taiwan, Tech. Rep., 2003. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/papers/tanh.pdf
[11] C. S. Ong, X. Mary, S. Canu, and A. J. Smola, "Learning with non-positive kernels," in *Proc. 21st Int. Conf. Mach. Learn. (ICML)*, 2004, pp. 639–646.
[12] B. Haasdonk, "Feature space interpretation of SVMs with indefinite kernels," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 4, pp. 482–492, Apr. 2005.
[13] R. Luss and A. d'Aspremont, "Support vector machine classification with indefinite kernels," in *Advances in Neural Information Processing Systems*, J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, Eds. Cambridge, MA, USA: MIT Press, 2008, pp. 953–960.

[14] J. Chen and J. Ye, "Training SVM with indefinite kernels," in *Proc. 25th Int. Conf. Mach. Learn. (ICML)*, 2008, pp. 136–143.
[15] Y. Ying, C. Campbell, and M. Girolami, "Analysis of SVM with indefinite kernels," in *Advances in Neural Information Processing Systems*, Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, Eds. Cambridge, MA, USA: MIT Press, 2009, pp. 2205–2213.
[16] S. Gu and Y. Guo, "Learning SVM classifiers with indefinite kernels," in *Proc. 26th AAAI Conf. Artif. Intell.*, 2012, pp. 942–948.
[17] I. Alabdulmohsin, X. Gao, and X. Zhang, "Support vector machines with indefinite kernels," in *Proc. Asian Conf. Mach. Learn.*, 2014, pp. 32–47.
[18] G. Loosli, S. Canu, and C. S. Ong, "Learning SVM in Krein spaces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 6, pp. 1204–1216, Jun. 2016.
[19] F.-M. Schleif and P. Tino, "Indefinite proximity learning: A review," *Neural Comput.*, vol. 27, no. 10, pp. 2039–2096, 2015.
[20] X. Huang, A. Maier, J. Hornegger, and J. A. K. Suykens, "Indefinite kernels in least squares support vector machines and principal component analysis," *Appl. Comput. Harmon. Anal.*, to be published, doi: 10.1016/j.acha.2016.09.001.
[21] V. Vapnik, *Statistical Learning Theory*. New York, NY, USA: Wiley, 1998.
[22] J. A. K. Suykens, T. Van Gestel, B. De Moor, J. De Brabanter, and J. Vandewalle, *Least Squares Support Vector Machines*. Singapore: World Scientific, 2002.
[23] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani, "1-Norm support vector machines," in *Advances in Neural Information Processing Systems*, S. Thrun, L. K. Saul, and B. Schölkopf, Eds. Cambridge, MA, USA: MIT Press, 2004, pp. 49–56.
[24] F. Anselmi, L. Rosasco, C. Tan, and T. Poggio. (Aug. 2015). "Deep convolutional networks are hierarchical kernel machines." [Online]. Available: https://arxiv.org/abs/1508.01084
[25] L. Remaki and M. Cheriet, "KCS-new kernel family with compact support in scale space: Formulation and impact," *IEEE Trans. Image Process.*, vol. 9, no. 6, pp. 970–981, Jun. 2000.
[26] N. Cressie, *Statistics for Spatial Data*. New York, NY, USA: Wiley, 1993.
[27] M. G. Genton, "Classes of kernels for machine learning: A statistics perspective," *J. Mach. Learn. Res.*, vol. 2, pp. 299–312, Mar. 2002.
[28] B. Hamers, J. A. K. Suykens, and B. De Moor, "Compactly supported RBF kernels for sparsifying the Gram matrix in LS-SVM regression models," in *Proc. Int. Conf. Artif. Neural Netw. (ICANN)*, 2002, pp. 720–726.
[29] S. Saryazdi and M. Cheriet, "PKCS: A polynomial kernel family with compact support for scale-space image processing," *IEEE Trans. Image Process.*, vol. 16, no. 9, pp. 2299–2308, Sep. 2007.
[30] L. Breiman, "Hinging hyperplanes for regression, classification, and function approximation," *IEEE Trans. Inf. Theory*, vol. 39, no. 3, pp. 999–1013, May 1993.
[31] S. Wang, X. Huang, and K. M. Junaid, "Configuration of continuous piecewise-linear neural networks," *IEEE Trans. Neural Netw.*, vol. 19, no. 8, pp. 1431–1445, Aug. 2008.
[32] J. C. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in kernel methods*. Cambridge, MA, USA: MIT Press, 1999, pp. 185–208.
[33] A. Frank and A. Asuncion, "UCI machine learning repository," School Inf. Comput. Sci., Univ. California, Irvine, CA, USA, Tech. Rep., 2010. [Online]. Available: http://archive.ics.uci.edu/ml
[34] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 27:1–27:27, 2011.
[35] K. De Brabanter, P. Karsmakers, F. Ojeda, C. Alzate, J. De Brabanter, K. Pelckmans, B. De Moor, J. Vandewalle, and J.A.K. Suykens, "LS-SVMlab toolbox user's guide, version 1.8," ESAT-SISTA, Leuven, Belgium, Tech. Rep. 10-146, 2010.
[36] K. Kampa. *Automatic Parameter Selection, Integrated Brain Imaging Center*, UW Medical Center, Seattle, WA, USA. [Online], accessed on Feb. 22, 2016. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm/faq.html#/Q9:_MATLAB_interface