# Indefinite Kernel Logistic Regression

Fanghui Liu
Institute of Image Processing and Pattern Recognition
Shanghai Jiao Tong University
Shanghai, China 200240
lfhsgre@outlook.com

Xiaolin Huang
Institute of Image Processing and Pattern Recognition
Shanghai Jiao Tong University
Shanghai, China 200240
xiaolinhuang@sjtu.edu.cn

Jie Yang*
Institute of Image Processing and Pattern Recognition
Shanghai Jiao Tong University
Shanghai, China 200240
jieyang@sjtu.edu.cn

## ABSTRACT

Traditionally, kernel learning methods require positive definitiveness on the kernel, which is too strict and excludes many sophisticated similarities, that are indefinite. To utilize those indefinite kernels, indefinite learning methods are of great interests. This paper aims at the extension of the logistic regression from positive definite kernels to indefinite ones. The proposed model, named indefinite kernel logistic regression (IKLR), keeps consistency to the regular KLR in formulation but it essentially becomes non-convex. Thanks to the positive decomposition of an indefinite kernel, IKLR can be transformed into a difference of two convex models, which follows the use of concave-convex procedure. Moreover, aiming at large-scale problems in practice, a concave-inexact-convex procedure (CCICP) algorithm with an inexact solving scheme is proposed with convergence guarantees. Experimental results on multi-modal datasets demonstrate the superiority of the proposed IKLR model over kernel logistic regression with positive definite kernels and other state-of-the-art indefinite learning based methods.

## KEYWORDS

indefinite kernel, logistic regression, concave-inexact-convex procedure

## 1 INTRODUCTION

Kernel methods [16] are powerful statistical machine learning techniques, which have been widely and successfully used. The representative kernel-based algorithms include Support Vector Machine (SVM, [19]), Kernel Logistic Regression (KLR, [24]), Kernel Fisher Discriminant Analysis (KFDA, [12]), and so on. In above kernel-based methods, the corresponding kernel matrix is required to be symmetric and positive semi-definite to satisfy Mercer's condition. Accordingly, these methods can be well analyzed in the Reproducing Kernel Hilbert Spaces (RKHS) [5].

However, in practice, we often meet some sophisticated similarity or dissimilarity measures that are either *indefinite* (real, symmetric, but not positive definite) or for which the Mercer condition is difficult to verify. For example, in multimedia area, one can use the human-judged similarities between concepts and words in music recommendation [20], video recommendation [17], or utilize dynamic time warping [9] for time series, or consider the Kullback-Leibler divergence between probability distributions. In these cases, many learning models boil down to be non-convex due to the used indefinite kernel which violates Mercer's condition. Hence, there is both practical and theoretical need to properly handle these measures.

To use indefinite similarities in classification task, there have been some discussions, mainly on SVM. In theory, learning with indefinite kernels is discussed in the Reproducing Kernel Kreĭn Spaces (RKKS) [10, 11], instead of the conventional reproducing kernel Hilbert spaces (RKHS) for positive definite kernels. In practice, two kinds of algorithms are considered to deal with indefinite kernels: i) kernel approximation and ii) non-convex optimization. Kernel approximation aims to transform the indefinite kernel matrix into a positive semi-definite matrix by spectrum modification. For example, "*flip*": the absolute value of the negative eigenvalues; "*clip*": the negative eigenvalues cut to zero; "*shift*": all eigenvalues plus a positive constant until the smallest eigenvalue is zero. However, above operations actually change the indefinite matrix itself, and thus may cause in the loss of some important information involved with the kernel. The second approach is to directly solve the corresponding non-convex problem. For SVM with indefinite kernles, [4] applies the SMO-type algorithm and [1, 22] uses the concave-convex procedure (CCCP) [23] algorithm that decomposes the objective function into the difference of two convex functions.

In this paper, we investigate the use of indefinite kernels on kernel logistic regression (KLR). It is a representative classifier and has been widely and successfully applied in many fields. However, indefinite kernel logistic regression (IKLR) has not yet been investigated in the past. To extend kernel used in KLR from positive definite kernels to indefinite ones, we need to carefully discuss the indefinite model and its corresponding algorithm. In formulation, based on the representor theorem in RKKS, the IKLR model shares the similar formulation with that of the regular KLR. However, using indefinite kernel makes the problem non-convex and hard to solve. To tackle this issue, we decompose the objective function into the difference of two convex functions and

*Corresponding author.

*MM'17, October 23–27, 2017, Mountain View, CA, USA.*
© 2016 Copyright held by the owner/author(s). ACM ISBN 978-1-4503-4906-2/17/10...$15.00
DOI: https://doi.org/10.1145/3123266.3123295

then the CCCP algorithm is applicable. Moreover, aiming at large-scale problems in practice, a concave-inexact-convex procedure (CCICP) algorithm is proposed to obtain early termination during each iteration. We theoretically demonstrate the convergence of CCICP with the provable guarantee. Experiments on various multi-modal datasets suggest that in most cases our IKLR method outperforms not only the conventional KLR with positive kernels but also other recent algorithms with indefinite kernels.

## 2 KERNEL LOGISTIC REGRESSION

Kernel logistic regression has been proven to be a powerful classifier with several merits [8] when compared with other traditional classifiers. It can naturally provide probabilities and straightforward extend to multi-class classification problems. Specifically, it only requires solving an unconstrained quadratic problem, and thus, the computation time can be much less than that of other methods, such as SVM which needs to solve a constrained quadratic optimization problem.

Here we briefly introduce KLR in the binary classification setting. In this setting, given a training set $\left\{(\mathbf{x}_i, y_i)\right\}_{i=1}^{n}$, an instance space $\mathcal{X}$, an output space $\mathcal{Y}$, and a training sample $\mathbf{x}_i \in \mathcal{X}$ with its corresponding label $y_i \in \{+1, -1\}$ in the space $\mathcal{Y}$. We aim to learn a function $f : \mathcal{X} \to \mathcal{Y}$ based on these $n$ training samples, so that when given a new input $\mathbf{z} \in \mathbb{R}^m$ ($m$ is the feature dimension) from the test sample set $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_s]$ with $s$ test samples, we can predict its label $y$. Many people have noted the relationship between a classifier (e.g. SVM, logistic regression) and regularized function estimation in the reproducing kernel Hilbert spaces (RKHS) [5]. For instance, fitting a logistic regression problem is equivalent to:

$$\min_{f \in \mathcal{H}} \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 + \frac{1}{n}\sum_{i=1}^{n} \ln\Big(1 + \exp\big(-y_i f(\mathbf{x}_i)\big)\Big)\,, \qquad (1)$$

where $\mathcal{H}$ is the RKHS generated by the kernel $\mathcal{K}(\cdot,\cdot)$, and $\lambda$ is the regularization parameter. Generally, the discriminant function is formulated as $f(\mathbf{x}) = \mathbf{w}^\top\mathbf{x} + b$ [1], where $\mathbf{w} \in \mathbb{R}^m$ is a weight vector parameterizing the space of linear functions mapping from $\mathcal{X}$ to $\mathcal{Y}$. By the representer theorem [15] in RKHS, the optimal $f^*(\mathbf{x})$ can be formulated as:

$$f^*(\mathbf{x}) = \sum_{i=1}^{n} \beta_i \mathcal{K}(\mathbf{x}, \mathbf{x}_i)\,,$$

where $\mathcal{K}$ is a kernel function in RKHS and the coefficient vector $\boldsymbol{\beta} \in \mathbb{R}^n$. Accordingly, the formulation of kernel logistic regression can be obtained as:

$$\min_{\boldsymbol{\beta}} \frac{\lambda}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\beta_i\beta_j\mathbf{K}_{ij} + \frac{1}{n}\sum_{i=1}^{n}\ln\Big(1 + \exp\big(-y_i\sum_{j=1}^{n}\beta_j\mathbf{K}_{ij}\big)\Big)\,, \qquad (2)$$

where $\mathbf{K}_{ij} = \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$ is a kernel matrix. With some abuse of notation, in [24], Eq. (2) can be written in a compact form:

$$\min_{\boldsymbol{\beta}} \frac{\lambda}{2}\boldsymbol{\beta}^\top\mathbf{K}\boldsymbol{\beta} + \frac{1}{n}\mathbf{1}^\top \ln\Big(\mathbf{1} + \exp(-\mathbf{y} \odot \mathbf{K}\boldsymbol{\beta})\Big)\,, \qquad (3)$$

where $\mathbf{1}$ denotes the all-one vector, the operator $\odot$ is element-wise multiplication, and $\mathbf{y} = (y_1, y_2, \cdots, y_n)^\top$. Traditionally, in Eq. (3), we require the positive semi-definite property on the kernel matrix $\mathbf{K}$, and thus the optimization problem is formulated as a convex unconstrained quadratic programming. To find the optimal $\boldsymbol{\beta}$, the Newton-Raphson method can be used to iteratively solve such optimization problem.

## 3 INDEFINITE LEARNING IN KERNEL LOGISTIC REGRESSION

### 3.1 IKLR Model

In indefinite learning, using indefinite kernels in Eq. (3) makes Mercer's theorem not applicable, which means that the functional space spanned by indefinite kernels does not belong to RKHS. To tackle indefinite kernels in theory, the Reproducing Kernel Kreĭn Spaces (RKKS) [11] is introduced to provide a justification for feature space interpretation. In this case, the primal optimization problem of our IKLR model is formulated as a stabilization problem instead of a minimization problem. We reformulate Eq. (1) in RKKS as follows:

$$\operatorname*{stablize}_{f \in \mathcal{H}_K} \frac{\lambda}{2}\|f\|_{\mathcal{H}_K}^2 + \frac{1}{n}\sum_{i=1}^{n}\ln\Big(1 + \exp\big(-y_i f(\mathbf{x}_i)\big)\Big)\,, \qquad (4)$$

where $\mathcal{H}_K$ is the RKKS generated by the kernel $\mathcal{K}(\cdot,\cdot)$. In [11], Ong *et al.* verify the existence of the representer theorem in RKKS. That is, if the optimization problem in Eq. (4) has a saddle point, it admits the following expansion:

$$f^* = \sum_{i=1}^{n}\beta_i\mathcal{K}(\mathbf{x}_i, \cdot)\,,$$

where $\mathcal{K}$ is a kernel function in RKKS and $\boldsymbol{\beta}$ is the coefficient vector. Since this condition is easily satisfied, the logistic regression problem with indefinite kernels can be expressed in RKKS, which arrives at:

$$\operatorname*{stab}_{\boldsymbol{\beta}} \frac{\lambda}{2}\boldsymbol{\beta}^\top\mathbf{K}\boldsymbol{\beta} + \frac{1}{n}\mathbf{1}^\top\ln\Big(\mathbf{1} + \exp(-\mathbf{Y}\mathbf{K}\boldsymbol{\beta})\Big)\,, \qquad (5)$$

where the label matrix $\mathbf{Y} \in \mathbb{R}^{n\times n}$ is a diagonal matrix, of which the $i$th diagonal element is $y_i$. It can be seen that Eq. (5) shares the similar formulation with Eq. (3). However, due to the indefinite property of the kernel matrix $\mathbf{K}$ in Eq. (5), such non-convex optimization problem must be analysed in the Kreĭn space.

### 3.2 Kernels in Kreĭn Space

The feature space in indefinite learning is given by a Kreĭn space [6], which is an indefinite inner product space endowed with a Hilbertain topology, yet its inner product is not necessarily non-positive. The Kreĭn space is with the following explicit definition in [3].

[1] We omit the bias term in theatrical discussions for simplicity but include it in numerical experiments.

*Definition 3.1.* An inner product space is a Kreĭn space $\mathcal{H}_K$ if there exist two Hilbert spaces $\mathcal{H}_+$ and $\mathcal{H}_-$ spanning $\mathcal{H}_K$ such that i) All $f \in \mathcal{H}_K$ can be decomposed into $f = f_+ + f_-$, where $f_+ \in \mathcal{H}_+$ and $f_- \in \mathcal{H}_-$, respectively. ii) $\forall f, g \in \mathcal{H}_K$, $\langle f, g\rangle_{\mathcal{H}_K} = \langle f_+, g_+\rangle_{\mathcal{H}_+} - \langle f_-, g_-\rangle_{\mathcal{H}_-}$.

The existence of RKKS implies that an indefinite kernel $\mathcal{K}$ has a positive decomposition on a given set $\mathcal{X}$ such that:

$$\mathcal{K}(\mathbf{u},\mathbf{v}) = \mathcal{K}_+(\mathbf{u},\mathbf{v}) - \mathcal{K}_-(\mathbf{u},\mathbf{v}), \forall \mathbf{u},\mathbf{v} \in \mathcal{X}\,,$$

where $\mathcal{K}_+$ and $\mathcal{K}_-$ are two positive definite kernels. Thus the objective function in Eq. (5) can be rewritten as:

$$\operatorname*{stab}_{\boldsymbol{\beta}} f(\boldsymbol{\beta}) = \frac{\lambda}{2}\boldsymbol{\beta}^\top(\mathbf{K}_+ - \mathbf{K}_-)\boldsymbol{\beta} + \frac{1}{n}\mathbf{1}^\top \ln\Big(\mathbf{1} + \exp(-\mathbf{Y}\mathbf{K}\boldsymbol{\beta})\Big)\,. \tag{6}$$

To obtain $\mathbf{K}_+$ and $\mathbf{K}_-$, one can decompose the symmetric indefinite kernel matrix $\mathbf{K}$ by eigenvalue decomposition, namely $\mathbf{K} = V^\top \Lambda V$, where $V$ is an orthogonal matrix and the diagonal matrix $\Lambda$ is defined as $\Lambda = \operatorname{diag}(\mu_1, \mu_2, \cdots, \mu_n)$, of which elements are eigenvalues of $\mathbf{K}$ with $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_n$. Without loss of generality, we assume that the first $v$ eigenvalues in $\Lambda$ are nonnegative and the remaining $n - v$ eigenvalues are smaller than zero. As a result, $\mathbf{K}_+$ and $\mathbf{K}_-$ can be formulated as:

$$\begin{cases} \mathbf{K}_+ = V^\top \operatorname{diag}(\mu_1 + \rho, \cdots, \mu_v + \rho, \rho, \cdots, \rho)V; \\ \mathbf{K}_- = V^\top \operatorname{diag}(\rho, \cdots, \rho, \rho - \mu_{v+1}, \cdots, \rho - \mu_n)V. \end{cases},$$

where $\rho$ is chosen as $\rho > -\mu_n$ to guarantee these two matrices $\mathbf{K}_+$ and $\mathbf{K}_-$ positive definite. By this decomposition of $\mathbf{K}$, the objective function in Eq. (6) can be decomposed as $f(\boldsymbol{\beta}) = g(\boldsymbol{\beta}) - h(\boldsymbol{\beta})$ with:

$$\begin{cases} g(\boldsymbol{\beta}) = \dfrac{\lambda}{2}\boldsymbol{\beta}^\top \mathbf{K}_+ \boldsymbol{\beta} + \dfrac{1}{n}\mathbf{1}^\top \ln\Big(\mathbf{1} + \exp(-\mathbf{Y}\mathbf{K}\boldsymbol{\beta})\Big), \\ h(\boldsymbol{\beta}) = \dfrac{\lambda}{2}\boldsymbol{\beta}^\top \mathbf{K}_- \boldsymbol{\beta}. \end{cases} \tag{7}$$

# 4 IKLR MODEL WITH THE CCICP ALGORITHM

In this section, we present a CCICP algorithm to efficiently solve such non-convex problem. Further, the convergence analysis of the CCICP algorithm in IKLR is theoretically demonstrated.

## 4.1 Solving with CCICP

Based on the above discussions, the objective function $f(\boldsymbol{\beta})$ in Eq. (6) can be formulated as the difference of two convex functions $g(\boldsymbol{\beta})$ and $h(\boldsymbol{\beta})$. Therefore the CCCP algorithm is an appropriate choice to solve such problem. Here we briefly introduce the main idea of the CCCP and then detail our CCICP algorithm.

The CCCP algorithm decomposes the non-convex objective function $f(\boldsymbol{\beta})$ into the difference of two convex functions $g(\boldsymbol{\beta})$ and $h(\boldsymbol{\beta})$: $f(\boldsymbol{\beta}) = g(\boldsymbol{\beta}) - h(\boldsymbol{\beta})$. In each iteration, $h(\boldsymbol{\beta})$ is replaced by its first order Taylor approximation $\tilde{h}(\boldsymbol{\beta})$ around its current solution, and then the original non-convex objective function $f(\boldsymbol{\beta})$ can be approximated by the convex function $\tilde{f}(\boldsymbol{\beta}) = g(\boldsymbol{\beta}) - \tilde{h}(\boldsymbol{\beta})$. Accordingly, the sub-problem $\tilde{f}(\boldsymbol{\beta})$ is formulated as a simpler convex form and then solved by an off-the-shelf convex solver (e.g. a gradient descent method). Theoretical analyses suggest that CCCP is able to converge to a local minima [18].

Nonetheless, it can be observed that such sub-problem needs to be solved at each iteration in CCCP, which makes the solving process inefficient especially for a large-scale dataset. To tackle this issue, we propose a concave-inexact-convex procedure (CCICP), that only requires an inexact solution for the sub-problem. By doing so, the CCICP algorithm is able to effectively speed up the solving process. To be specific, the inexact solution $\boldsymbol{\beta}^{(t+1)}$ lies in an $\delta$-neighborhood around the actual result $\boldsymbol{\beta}_*^{(t)} = \operatorname*{argmin}_{\boldsymbol{\beta}} \tilde{f}(\boldsymbol{\beta})$. Since a gradient descent algorithm is used, it satisfies $\tilde{f}(\boldsymbol{\beta}^{(t+1)}) \leq \tilde{f}(\boldsymbol{\beta}_*^{(t)})$. Here $\boldsymbol{\beta}^{(t+1)}$ is bounded by $\boldsymbol{\beta}_*^{(t)}$ with the following formula:

$$\boldsymbol{\beta}^{(t+1)} \in \mathrm{U}_\delta(\boldsymbol{\beta}_*^{(t)}) \triangleq \left\{\boldsymbol{\beta} \mid \|\boldsymbol{\beta} - \boldsymbol{\beta}_*^{(t)}\| \leq \delta\right\}.$$

In this case, the Karush-Kuhn-Tucker (KKT) condition for $\boldsymbol{\beta}^{(t+1)}$ does not hold, namely:

$$\nabla_{\boldsymbol{\beta}} \tilde{f}(\boldsymbol{\beta})|_{\boldsymbol{\beta}=\boldsymbol{\beta}^{(t+1)}} \neq 0\,.$$

Without loss of generality, we assume that:

$$\nabla_{\boldsymbol{\beta}} \tilde{f}(\boldsymbol{\beta})|_{\boldsymbol{\beta}=\boldsymbol{\beta}^{(t+1)}} = \varepsilon\|\boldsymbol{\beta}^{(t)}\|\,, \tag{8}$$

where $\varepsilon$ corresponds to the bounded error, and its choice will be discussed in Section 4.2.

Based on the above analyses, we detail the CCICP algorithm in our IKLR model. The function $h(\boldsymbol{\beta})$ is linearized by its Taylor approximation at $\boldsymbol{\beta}^{(t)}$: $\tilde{h}(\boldsymbol{\beta}^{(t)}) = \lambda \boldsymbol{\beta}^{(t)\top}\mathbf{K}_-(\boldsymbol{\beta} - \boldsymbol{\beta}^{(t)})$. As a result, the sub-problem is reformulated as:

$$\tilde{f}(\boldsymbol{\beta}, \boldsymbol{\beta}^{(t)}) = \frac{\lambda}{2}\boldsymbol{\beta}^\top \mathbf{K}_+ \boldsymbol{\beta} + \frac{1}{n}\mathbf{1}^\top \ln\Big(\mathbf{1} + \exp(-\mathbf{Y}\mathbf{K}\boldsymbol{\beta})\Big) - \tilde{h}(\boldsymbol{\beta}^{(t)})\,. \tag{9}$$

We employ the gradient descent method to solve this convex optimization problem, in which the gradient of $\tilde{f}(\boldsymbol{\beta}, \boldsymbol{\beta}^{(t)})$ with respect to $\boldsymbol{\beta}$ is computed as:

$$\nabla_{\boldsymbol{\beta}} \tilde{f}(\boldsymbol{\beta}, \boldsymbol{\beta}^{(t)}) = \lambda \mathbf{K}_+ \boldsymbol{\beta} - \frac{1}{n}\mathbf{K}\mathbf{Y}\mathbf{W}\mathbf{q} - \lambda \mathbf{K}_- \boldsymbol{\beta}^{(t)}\,, \tag{10}$$

where $\mathbf{W} = \operatorname{diag}[\exp(-\mathbf{Y}\mathbf{K}\boldsymbol{\beta})]$ is a diagonal matrix whose $i$th diagonal element is $\exp(-y_i \mathbf{K}^{(i)}\boldsymbol{\beta})$, and $\mathbf{q} = (q_1, q_2, \cdots, q_n)^\top$ by defining

$$q_i = \frac{1}{1 + \exp\big(-y_i \sum_{j=1}^n \beta_j \mathbf{K}_{ij}\big)},\ \forall i = 1, 2, \cdots, n. \tag{11}$$

To obtain the inexact solution $\boldsymbol{\beta}^{(t+1)} \approx \operatorname*{argmin}_{\boldsymbol{\beta}} \tilde{f}(\boldsymbol{\beta}, \boldsymbol{\beta}^{(t)})$ in the sub-problem, the early termination scheme occupied by Eq. (8) is executed to obtain early stop during each iteration. Specifically, under the inexact solving scheme with the bounded error assumption, the rationality of such approximation and the convergence of the CCICP algorithm will be theoretically demonstrated in Section 4.2. The detailed procedure of the CCICP algorithm for IKLR is summarized in Algorithm 1.

After obtaining the output $\tilde{\boldsymbol{\beta}}$ by Algorithm 1, in the test process, we firstly construct the test kernel matrix $\mathbf{K}$ associated with the training sample set $\mathbf{X}$ and the test sample set $\mathbf{Z}$,

**Algorithm 1:** CCICP for indefinite kernel logistic regression.

```
Input: the indefinite kernel matrix $\mathbf{K}$ and two positive
       semi-definite kernel matrices $\mathbf{K}_+$ and $\mathbf{K}_-$, the
       label matrix $\mathbf{Y}$, and the regularization
       parameter $\lambda$.
Output: the coefficient vector $\boldsymbol{\beta}$.
Set: stopping criteria: $t_{\max} = 15$, the stepsize $\eta = 0.2$,
and the decay factor $\tau = 0.5$;
Initialize $t = 0$ and $\boldsymbol{\beta}^{(0)}$, and compute $\varepsilon$;
Repeat
    Obtain $\tilde{h}(\boldsymbol{\beta}^{(t)}) = \lambda \mathbf{K}_- \boldsymbol{\beta}^{(t)}$;
    Obtain the sub-problem $\tilde{f}(\boldsymbol{\beta})$ by Eq. (9);
    // Inner Loop:  Solve $\boldsymbol{\beta}^{(t+1)} = \operatorname*{argmin}_{\boldsymbol{\beta}} \tilde{f}(\boldsymbol{\beta})$.
    Initialize $k = 0$ and $\boldsymbol{\beta}_k^{(t)} := \boldsymbol{\beta}^{(t)}$;
    while $\|\nabla \tilde{f}(\boldsymbol{\beta}_k^{(t)})\| > \varepsilon \|\boldsymbol{\beta}_k^{(t)}\|$ do
        Obtain the gradient $\nabla \tilde{f}(\boldsymbol{\beta}_k^{(t)})$ by Eq. (10);
        $\boldsymbol{\beta}_{k+1}^{(t)} := \boldsymbol{\beta}_k^{(t)} - \tau^k \eta \nabla \tilde{f}(\boldsymbol{\beta}_k^{(t)})$;
        $k := k + 1$;
    end
    Output $\boldsymbol{\beta}^{(t+1)} := \boldsymbol{\beta}_*^{(t)}$ that minimizes Eq. (9);
    // Inner Loop completes.
    $t := t + 1$;
Until $t = t_{\max} \vee \frac{\|\boldsymbol{\beta}^{(t)} - \boldsymbol{\beta}^{(t-1)}\|_2}{\|\boldsymbol{\beta}^{(t)}\|_2} \leq \varepsilon$;
Output the stationary point $\tilde{\boldsymbol{\beta}}$ that minimizes Eq. (7).
```

namely $\mathbf{K}_{ij} = \mathcal{K}(\mathbf{x}_i, \mathbf{z}_j)$, and then compute the classification score of the $i$th test sample $p(\mathbf{z}_i)$, which is defined as:

$$p(\mathbf{z}_i) = \frac{\exp\big(\mathbf{K}^{(i)}\mathbf{z}_i\big)}{1 + \exp\big(\mathbf{K}^{(i)}\mathbf{z}_i\big)}, \forall i = 1, 2, \cdots, s,$$

where $\mathbf{K}^{(i)}$ represents the $i$th row of the test kernel matrix $\mathbf{K}$. If the classification score $p(\mathbf{z}_i) > 0.5$, we label $\mathbf{z}_i$ with $+1$, otherwise it is assigned to $-1$, which completes a predict progress for a test sample.

## 4.2 Convergence Analysis of CCICP

With the aforementioned inexact operation, the CCICP algorithm is expected to speed up the optimization process. For the ease of such algorithm in theory, we carefully consider the convergence of CCICP by investigating an inexact sequence $\{\boldsymbol{\beta}^{(t)}\}_{t=1}^{\infty}$ generated by Algorithm 1, and then further analyse its convergence rate in the proposed IKLR model.

The key convergence analysis result of the CCICP is summarized by Theorem 4.2, that is, when the error $\varepsilon$ is upper bounded, the sequence $\{\boldsymbol{\beta}^{(t)}\}_{t=1}^{\infty}$ generated by a given point $\boldsymbol{\beta}^{(0)} \in \mathbb{R}^n$ still converges to a stationary point.

Before proving Theorem 4.2, we need the following Lemma 4.1 to aid the proof.

Lemma 4.1. *Given a sigmoid function $R(x) = (1 + \mathrm{e}^{cx})^{-1}$ where $c \in \{+1, -1\}$, and for two arbitrary variables $x_1, x_2 \in (-\infty, +\infty)$, there exists a bound such that*

$$\big|R(x_1) - R(x_2)\big| \leq \frac{1}{4}\big|x_1 - x_2\big|. \tag{12}$$

Proof. Because $R(x)$ is a differential function, by Lagrange mean value theorem, there exists at least one point $\xi \in \big(\min(x_1, x_2), \max(x_1, x_2)\big)$ such that

$$\big|R(x_1) - R(x_2)\big| = \big|(x_1 - x_2)R'(\xi)\big|,$$

where the range of $|R'(\xi)|$ satisfies:

$$|R'(\xi)| = \frac{\mathrm{e}^{a\xi}}{(1 + \mathrm{e}^{a\xi})^2} = \frac{1}{\mathrm{e}^{a\xi} + \mathrm{e}^{-a\xi} + 2} \leq \frac{1}{4}.$$

Then we can conclude the proof:

$$\big|R(x_1) - R(x_2)\big| = \big|(x_1 - x_2)R'(\xi)\big| \leq \frac{1}{4}|x_1 - x_2|.$$

□

Next we are ready to prove Theorem 4.2 as follows.

Theorem 4.2. *The sequence $\{\boldsymbol{\beta}^{(t)}\}_{t=1}^{\infty}$ with an inexact operation generated by CCICP still converges to a local minimum or a stationary point if the bound error $\varepsilon$ in Eq. (8) (i.e. $\varepsilon_1$ and $\varepsilon_2$ in Eqs. (14) and (15)) satisfies:*

$$\max\Big\{\varepsilon_1, \varepsilon_2\Big\} < \lambda\Big(\|\mathbf{K}_+\| - \|\mathbf{K}_-\|\Big) - \frac{\|\mathbf{K}\|^2}{4n}. \tag{13}$$

Proof. Let $\phi : U \subset \mathbb{R}^n \to \mathbb{R}^n$ be a point-to-set map, $\boldsymbol{\beta}^{(t+1)} \in \phi(\boldsymbol{\beta}^{(t)})$ such that:

$$\phi(\boldsymbol{\beta}^{(t)}) = \operatorname*{argmin}_{\boldsymbol{\beta}} \tilde{f}(\boldsymbol{\beta}, \boldsymbol{\beta}^{(t)}),$$

which generates an inexact sequence $\{\boldsymbol{\beta}^{(t)}\}_{t=1}^{\infty}$ through the rule $\boldsymbol{\beta}^{(t+1)} \in \phi(\boldsymbol{\beta}^{(t)})$, where $\phi(\boldsymbol{\beta}^{(t)})$ satisfies the bounded error assumption, that is:

$$\nabla_{\boldsymbol{\beta}} \tilde{f}(\boldsymbol{\beta}, \boldsymbol{\beta}^{(t)})|_{\boldsymbol{\beta} = \phi(\boldsymbol{\beta}^{(t)})} = \varepsilon \|\boldsymbol{\beta}^{(t)}\|.$$

Specifically, the map $\phi$ is said to be *global convergent*[2] if for any chosen initial point $\boldsymbol{\beta}^{(0)}$, the sequence converges to a point for which a necessary condition of optimality holds. Therefore, the key is to prove that the map $\phi$ is a contraction mapping for two arbitrary points $\mathbf{a}, \mathbf{b} \in \mathit{int}(U)$ such that:

$$\big\|\phi(\mathbf{a}) - \phi(\mathbf{b})\big\| \leq \alpha \|\mathbf{a} - \mathbf{b}\|,$$

for a distance metric $\|\cdot\|$, where $\alpha \in [0, 1)$.

Suppose that $\phi(\mathbf{a})$ and $\phi(\mathbf{b})$ satisfy:

$$\nabla_{\boldsymbol{\beta}} \tilde{f}(\boldsymbol{\beta}, \mathbf{a})|_{\boldsymbol{\beta} = \phi(\mathbf{a})} = \varepsilon_1 \|\mathbf{a}\|, \tag{14}$$

$$\nabla_{\boldsymbol{\beta}} \tilde{f}(\boldsymbol{\beta}, \mathbf{b})|_{\boldsymbol{\beta} = \phi(\mathbf{b})} = \varepsilon_2 \|\mathbf{b}\|, \tag{15}$$

where $\varepsilon_1$ and $\varepsilon_2$ correspond to the bounded error, that lead to the inexact sequence $\{\boldsymbol{\beta}^{(t)}\}_{t=1}^{\infty}$. For simplicity, suppose $\varepsilon_1 \leq \varepsilon_2$, and the subtraction between Eqs. (14) and (15) can be formulated as[3]:

$$\lambda \mathbf{K}_+\big[\phi(\mathbf{a}) - \phi(\mathbf{b})\big] = \lambda \mathbf{K}_-(\mathbf{a} - \mathbf{b}) + \frac{1}{n}\mathbf{K}\mathbf{Y}\mathbf{h} + \varepsilon_1 \|\mathbf{a}\| - \varepsilon_2 \|\mathbf{b}\|, \tag{16}$$

[2] It does not imply convergence to a global optimum for all initial values $\boldsymbol{\beta}^{(0)}$.

[3] If $\varepsilon_1 > \varepsilon_2$, we use the subtraction between Eq. (15) and (14).

where $\mathbf{h}$ is a $n$-dimensional vector, of which the $i$th element is defined as:

$$h_i = \frac{1}{1+\exp\big(y_i\mathbf{K}^{(i)}\phi(\mathbf{a})\big)} - \frac{1}{1+\exp\big(y_i\mathbf{K}^{(i)}\phi(\mathbf{b})\big)}\,.$$

By Lemma 4.1, we have:

$$|h_i| \leq \frac{1}{4}|\mathbf{K}^{(i)}\phi(\mathbf{a}) - \mathbf{K}^{(i)}\phi(\mathbf{b})|,\ \forall i = 1,2,\cdots,n,$$

and then $\|\mathbf{h}\|_\infty$ satisfies[4]:

$$\|h\|_\infty \leq \frac{1}{4}|\mathbf{K}^{(s)}\phi(\mathbf{a}) - \mathbf{K}^{(s)}\phi(\mathbf{b})| \leq \frac{1}{4}\|\mathbf{K}^{(s)}\|_1\cdot\|\phi(\mathbf{a})-\phi(\mathbf{b})\|_\infty$$
$$\leq \frac{1}{4}\|\mathbf{K}\|_\infty\|\phi(\mathbf{a}) - \phi(\mathbf{b})\|_\infty\,,$$

where $s = \underset{i}{\operatorname{argmin}}\big|\mathbf{K}^{(i)}\phi(\mathbf{a}) - \mathbf{K}^{(i)}\phi(\mathbf{b})\big|,\ i = 1,2,\cdots,n.$

Due to the positiveness of $\mathbf{K}_+$, Eq. (16) can be reformulated as:

$$\phi(\mathbf{a})-\phi(\mathbf{b}) = \frac{1}{\lambda}\mathbf{K}_+^{-1}\Big\{\lambda\mathbf{K}_-(\mathbf{a}-\mathbf{b}) + \frac{1}{n}\mathbf{KYh} + \varepsilon_1\|\mathbf{a}\| - \varepsilon_2\|\mathbf{b}\|\Big\}\,.$$

Subsequently, it can be bounded by using $\|\cdot\|_\infty$ (we omit the notation for simplicity), that is:

$$\|\phi(\mathbf{a})-\phi(\mathbf{b})\| \leq \frac{1}{\lambda}\mathbf{K}_+^{-1}\Big\{\lambda\mathbf{K}_-(\mathbf{a}-\mathbf{b}) + \frac{1}{n}\mathbf{KYh} + \varepsilon_1\|\mathbf{a}\| - \varepsilon_2\|\mathbf{b}\|\Big\}$$
$$\leq \big\|\mathbf{K}_+^{-1}\mathbf{K}_-\big\|\,\|\mathbf{a}-\mathbf{b}\| + \frac{1}{\lambda n}\big\|\mathbf{K}_+^{-1}\mathbf{KY}\big\|\,\|\mathbf{h}\| + \frac{\varepsilon_2}{\lambda}\|\mathbf{K}_+^{-1}\|\,\Big|\|\mathbf{a}\| - \|\mathbf{b}\|\Big|$$
$$\leq \big\|\mathbf{K}_+^{-1}\big\|\big\|\mathbf{K}_-\big\|\ \|\mathbf{a}-\mathbf{b}\| + \frac{1}{4\lambda n}\big\|\mathbf{K}_+^{-1}\mathbf{K}\big\|\big\|\mathbf{K}\big\|\ \|\phi(\mathbf{a}) - \phi(\mathbf{b})\|$$
$$+\ \frac{\varepsilon_2}{\lambda}\big\|\mathbf{K}_+^{-1}\big\|\ \|\mathbf{a}-\mathbf{b}\|\,.$$

Hence we can obtain:

$$\|\phi(\mathbf{a}) - \phi(\mathbf{b})\| \leq \frac{\|\mathbf{K}_-\| + \frac{\varepsilon_2}{\lambda}}{\|\mathbf{K}_+\| - \frac{1}{4\lambda n}\|\mathbf{K}\|^2}\|\mathbf{a}-\mathbf{b}\|\,. \tag{17}$$

Likewise, if $\varepsilon_2 < \varepsilon_1$, the above formulation can be rewritten as:

$$\|\phi(\mathbf{b}) - \phi(\mathbf{a})\| \leq \frac{\|\mathbf{K}_-\| + \frac{\varepsilon_1}{\lambda}}{\|\mathbf{K}_+\| - \frac{1}{4\lambda n}\|\mathbf{K}\|^2}\|\mathbf{b}-\mathbf{a}\|\,. \tag{18}$$

Accordingly, Eqs. (17) and (18) can be reformulated into a uniform framework as follows:

$$\|\phi(\mathbf{a}) - \phi(\mathbf{b})\| \leq \frac{\|\mathbf{K}_-\| + \frac{\max\{\varepsilon_1,\varepsilon_2\}}{\lambda}}{\|\mathbf{K}_+\| - \frac{1}{4\lambda n}\|\mathbf{K}\|^2}\|\mathbf{a}-\mathbf{b}\|\,.$$

Further, to guarantee that the map $\phi$ is a contraction mapping, we require:

$$\alpha \triangleq \frac{\|\mathbf{K}_-\| + \frac{\max\{\varepsilon_1,\varepsilon_2\}}{\lambda}}{\|\mathbf{K}_+\| - \frac{1}{4\lambda n}\|\mathbf{K}\|^2} < 1\,.$$

After some straightforward algebraic manipulations, $\varepsilon_1$ and $\varepsilon_2$ can be upper bounded as shown in Eq. (13). Finally, the map $\phi$ served as a contraction mapping is well theoretical demonstrated if the error is upper bounded. By the fixed point theorem, we can conclude the proof. □

## 4.3 The Convergence Rate of our CCICP Algorithm

Here we are also interested in the convergence rate of the CCICP in our IKLR model. Salakhutdinov *et al.* [14] have studied the local convergence of the CCCP, that is, depending on the curvature of $g(\boldsymbol{\beta})$ and $h(\boldsymbol{\beta})$, CCCP would exhibit either quasi-Newton behavior with fast, typically superlinear convergence or first-order convergence behavior. Assume that the sequence $\{\boldsymbol{\beta}^{(t)}\}_{t=1}^{\infty}$ converges to the fixed point $\tilde{\boldsymbol{\beta}}$: $\tilde{\boldsymbol{\beta}} = \phi(\tilde{\boldsymbol{\beta}})$, we can Taylor expand it in the neighborhood of the fixed point $\tilde{\boldsymbol{\beta}}$ since the mapping $\phi$ is continuous and differentiable. That is:

$$\boldsymbol{\beta}^{(t+1)} - \tilde{\boldsymbol{\beta}} \approx M'(\tilde{\boldsymbol{\beta}})(\boldsymbol{\beta}^{(t)} - \tilde{\boldsymbol{\beta}})\,,$$

where $M'(\tilde{\boldsymbol{\beta}}) = \frac{\partial M}{\partial \boldsymbol{\beta}}\big|_{\boldsymbol{\beta}=\tilde{\boldsymbol{\beta}}}$, termed as the convergence matrix which controls the quasi-Newton behavior. Near the local optimum, this matrix is related to the curvature of the convex function $g(\boldsymbol{\beta})$ and the concave function $-h(\boldsymbol{\beta})$, namely:

$$M'(\tilde{\boldsymbol{\beta}}) = \left[\frac{\partial^2 h(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}\boldsymbol{\beta}^\top}\Big|_{\boldsymbol{\beta}=\tilde{\boldsymbol{\beta}}}\right]\left[\frac{\partial^2 g(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}\boldsymbol{\beta}^\top}\Big|_{\boldsymbol{\beta}=\tilde{\boldsymbol{\beta}}}\right]^{-1},$$

which can be interpreted as a ratio of concave curvature to convex curvature.

In the proposed CCICP algorithm, the fixed point $\tilde{\boldsymbol{\beta}}$ generated by the sequence $\{\boldsymbol{\beta}^{(t)}\}_{t=1}^{\infty}$ with a bounded error. In this case, it can be also approximated by the Taylor expansion around the actual fixed point. As a result, we can analyse the local convergence of CCICP in our model as abovementioned. After two Hessian matrices $\nabla^2_{\boldsymbol{\beta}}h(\boldsymbol{\beta})$ and $\nabla^2_{\boldsymbol{\beta}}g(\boldsymbol{\beta})$ obtained, the convergence matrix is determined by:

$$M'(\tilde{\boldsymbol{\beta}}) = \lambda\mathbf{K}_-\Big(\frac{1}{n}\mathbf{K}^\top\mathbf{H}(\tilde{\boldsymbol{\beta}})\mathbf{K} + \lambda\mathbf{K}_+\Big)^{-1},$$

where $\mathbf{H} = \operatorname{diag}\big(q_1(1-q_1),\cdots,q_n(1-q_n)\big)$, and $q_i$ is defined in Eq. (11). Given an indefinite kernel matrix $\mathbf{K}$, the convergence rate is determined by the ratio of $\mathbf{K}_-$ from the concave part and $\mathbf{K}_+$ from the convex part. Generally, in indefinite kernel learning, eigenvalues of $\mathbf{K}_+$ are usually much larger than that of $\mathbf{K}_-$. In this case, $\mathbf{K}_+$ occupies a dominant position for the convergence. Hence, the CCICP algorithm will exhibit a quasi-Newton behavior and possess fast, typically superlinear convergence. In the experiments, such condition will be satisfied in real-world dataset and the convergence of CCICP will be further demonstrated. Note that different convex-concave decompositions do not change the final results of our algorithm; while they only change the convergence rate.

# 5 EXPERIMENTS

In this section, we evaluate the IKLR model on two benchmarks with a collection of multi-modal dataset from multimedia and machine learning areas.

## 5.1 Experiment Setup

For the kernel setting, we choose a truncated $\ell_1$ distance (TL1) indefinite kernel [7] incorporated into our model, which is defined as $\mathcal{K}(\mathbf{u},\mathbf{v}) = \max\{\tau - \|\mathbf{u}-\mathbf{v}\|_1, 0\}$. As discussed

[4] Here we use $|\mathbf{a}^\top\mathbf{b}| = \|\mathbf{a}\|_p\|\mathbf{b}\|_q$, where $\frac{1}{p}+\frac{1}{q}=1$.

**Table 1: Statistics for various datasets with $n$ training samples represented by a $m$-dimensional feature. The notations $\mu_{\max}$ and $\mu_{\min}$ denote the maximum and minimum eigenvalues of the TL1 kernel over training samples. Specifically, the large-scale data sets are highlighted by bold.**

| Dataset | $m$(feature) | $n$(#num) | $\mu_{\min}$ | $\mu_{\max}$ |
|---|---|---|---|---|
| monks1 | 6 | 124 | -2.094 | 94.077 |
| monks2 | 6 | 169 | -2.535 | 131.14 |
| monks3 | 6 | 122 | -1.764 | 95.376 |
| parkinsons | 23 | 195 | 0.127 | 1200.4 |
| sonar | 60 | 208 | 1.452 | 3024.6 |
| SPECT | 21 | 80 | -1.145 | 353.11 |
| transfusion | 4 | 748 | -0.336 | 818.74 |
| splice | 60 | 1000 | -1.325 | 2885.3 |
| **EEG** | 14 | 14980 | -0.444 | 7312.0 |
| **guide1-t** | 4 | 4000 | -0.805 | 4116.7 |
| **madelon** | 500 | 2000 | 14.825 | 27015 |

in [7], the performance of the TL1 kernel is not very sensitive to the parameter $\tau$, and thus it is fixed to $\tau = 0.7m$ as suggested. In addition, as a representative positive definite kernel, the radial basis function (RBF) kernel is added for comparison, defined as: $\mathcal{K}(\mathbf{u}, \mathbf{v}) = \exp(-\|\mathbf{u} - \mathbf{v}\|_2/\sigma^2)$. The regularization parameter $\lambda$, and the kernel width in Gaussian kernel $\sigma$, the trade-off parameter $C$ in SVM are respectively tuned via a five-fold cross validation over the values $\{0.0001, 0.001, 0.01, 0.1, 1, 5, 10\}$ on the training set: one of these five subsets is used for validation in turn and the remaining ones for training.

## 5.2 Results on UCI Dataset

In this section, eleven real-world datasets from UCI Machine Learning Repository [2] are used to evaluate the performance of IKLR with other five algorithms. For each dataset normalized to $[0, 1]$, we randomly pick up half of the data for training and the rest for test. Table 1 lists a brief description of these datasets including the feature dimension $m$, the number of training samples $n$, the minimum and maximum eigenvalues of the training TL1 kernels. It can be observed that the absolute value of the maximum eigenvalue in each dataset is always much larger than that of the minimum one, which means that the CCICP will possess fast in our IKLR model as discussed in Section 4.3.

We compare IKLR with other representative state-of-the-art indefinite kernel learning based algorithms including: "Flip", "Clip", and "Shift" [21]: three methods directly convert the indefinite kernel matrix generated by TL1 kernel into a positive semi-definite matrix using the spectrum transformation. Then we take the modified kernel matrix into kernel logistic regression. "KSVM" [10]: a method transforms TL1 kernel from RKKS to RKHS, and then trains the convex dual form of SVM. "KLR" [24]: a representative classification method uses logistic regression with the RBF kernel just for self-verification.

We test the above algorithms on these eight small-scale datasets, where the procedure is repeated 10 times, and then the average classification accuracy and its standard deviation on test data are reported in Table 2. The best classification accuracy on each dataset in the sense of average accuracy is highlighted in bold.

In terms of the results in Table 2, we firstly analyze six datasets in which the training TL1 kernel is indefinite, namely: *monks1*, *monk2*, *monks3*, *SPECT*, *transfusion*, and *splice*. It can be observed that IKLR achieves a promising performance in most of datasets. Specifically, in *monk1* and *splice* datasets, the proposed IKLR method achieves the improvement with respective 7.0% and 14.3% than the second best one. The huge promotion benefits from the fact that the TL1 kernel with $\tau = 0.7m$ is robust and has good adaptiveness to different non-linearity in different areas among the data distribution. In addition, in the remaining six datasets, our algorithm ranks the first on three datasets and the second on the other two datasets. Specifically, compared to the representative indefinite learning based algorithm KSVM, the proposed IKLR method shows a favorable performance.

Lastly, we analyze two *parkinsons* and *sonar* datasets in which the training TL1 kernel is still positive definite. It can be observed that all compared algorithm achieve a similar classification accuracy without distinct difference. As a result, designing an advanced and delicate kernel in kernel logistic regression is more flexible to achieve promising performance, not limited to a positive definite kernel.

To further validate the effectiveness of the proposed inexact scheme, we investigate the performance of our methods on three large-scale data sets in Table 3. One can see that CCCP without any inexact scheme achieves the best performance on classification accuracy. However, the early stop condition makes our CCICP algorithm much efficient on training time.

Above results demonstrate that the proposed IKLR model not only outperforms non-convex optimization and kernel approximation with a statistically significant evidence on the indefinite training kernel, but also achieves a favorable classification accuracy on the training dataset with positive definite kernels. Moreover, the inexact scheme can effectively speed up the training process of the proposed algorithm.

## 5.3 Results on ESC Dataset

Environmental sound classification (ESC) is one of the obstacles in research activities. We accomplish this auditory recognition task by the proposed IKLR model on ESC-10 dataset [13]. The ESC-10 dataset is a selection of 10 classes that represents three general groups of sounds, namely transient sounds (*sneezing*, *dog barking*, *clock ticking*), sounds events with strong harmonic content (*crying baby*, *crowing rooster*), and sound event with structured noise (*rain*, *sea waves*, *fire crackling*, *helicopter*, *chainsaw*).

In the experiment, we extract a ubiquitous feature in speech processing, namely mel-frequenct cepstral coefficients

**Table 2: Test classification accuracy of (mean±std. deviation) of each compared algorithm on UCI datasets. The best performance is highlighted in bold.**

| | KLR(RBF) [24] | Flip | Clip | Shift | KSVM [10] | CCICP |
|---|---|---|---|---|---|---|
| monks1 | 0.668±0.052 | 0.695±0.075 | 0.648±0.070 | 0.685±0.063 | 0.586±0.102 | **0.765**±0.065 |
| monks2 | 0.662±0.071 | 0.498±0.110 | 0.506±0.116 | 0.489±0.092 | 0.626±0.037 | **0.669**±0.093 |
| monks3 | 0.779±0.073 | 0.723±0.090 | 0.805±0.021 | **0.870**±0.036 | 0.640±0.083 | 0.830±0.072 |
| parkinsons | **1.000**±0.000 | 0.990±0.010 | 0.999±0.003 | 0.998±0.007 | 0.945±0.039 | **1.000**±0.000 |
| sonar | 0.789±0.022 | 0.546±0.045 | 0.539±0.042 | 0.504±0.054 | 0.608±0.072 | **0.794**±0.060 |
| SPECT | 0.737±0.092 | 0.652±0.026 | 0.706±0.022 | 0.667±0.034 | **0.893**±0.024 | 0.764±0.059 |
| splice | 0.642±0.093 | 0.513±0.017 | 0.619±0.057 | 0.604±0.033 | 0.515±0.029 | **0.785**±0.050 |
| transfusion | 0.741±0.048 | 0.734±0.095 | 0.717±0.020 | 0.736±0.038 | **0.762**±0.006 | 0.726±0.129 |

**Table 3: Results of CCCP and CCICP on several large-scale data sets.**

| Dataset | EEG | | guide1-t | | madelon | |
|---|---|---|---|---|---|---|
| Method | CCCP | CCICP | CCCP | CCICP | CCCP | CCICP |
| Accuracy | 0.769±0.042 | 0.725±0.042 | 0.962±0.003 | 0.955±0.003 | 0.624±0.080 | 0.609±0.051 |
| Training time | 17171.0 | 848.885 | 1314.3 | 47.229 | 305.29 | 8.1293 |
| Test time | 0.1237 | 0.1304 | 0.0020 | 0.0028 | 0.0008 | 0.0064 |

**Table 4: Comparison of average classification accuracy (%) of different algorithms where $\mu_{\max}$ and $\mu_{\min}$ denote the maximum and minimum eigenvalues of the TL1 kernel over training samples.**

| $\mu_{\min}$ | $\mu_{\max}$ | SVM(RBF) | KSVM | IKLR |
|---|---|---|---|---|
| −0.068 | 722.45 | 64.3% | 68.1% | 75.7% |

(MFCC), where each speech clip is divided into numerous frames. For each frame, a 12-dimensional MFCC is extracted to represent the current frame in each clip with default settings[5]. By doing so, a speech clip is represented by a MFCC matrix where each row of this matrix is a 12-dimensional MFCC for a frame. Then we compute their means and standard deviations across frames with average pooling operation. As a result, a feature vector created in this way is treated as an input to effectively represent a speech clip. For these speech clips in ten classes, we randomly divide these clips in each class into two non-overlapping training and testing sets which contain almost half of the samples in each class. Learning is performed with a 5-fold cross-validation regime.

Here we choose three representative classifiers including SVM with RBF, KSVM [10] with TL1 kernel, and our IKLR algorithm to evaluate the classification performance. Table 4 reports the average test accuracy (%) across above three algorithms. We can see that the average classification accuracy ranges from 64.3% for SVM with the RBF kernel to 75.7% for our IKLR method, with KSVM with the middle (68.1%). This result reinforces to demonstrate the effectiveness of our IKLR algorithm with the TL1 indefinite kernel.

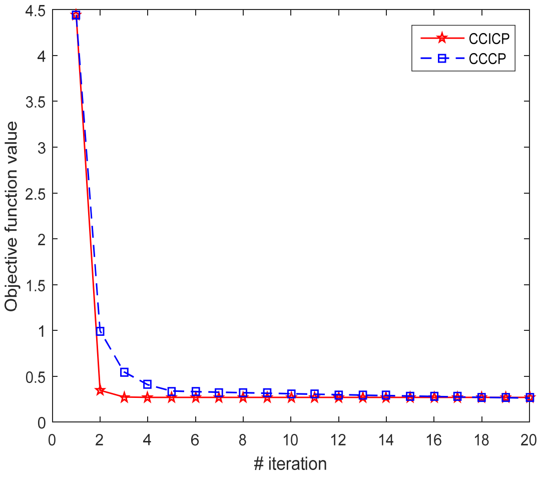

**Figure 1: Convergence plots for CCICP (red) and CCCP (blue) on the *monks1* dataset, with objective value versus iteration.**

## 5.4 Algorithm Convergence

The experiments about the convergence of CCICP algorithm are conducted on the *monks1* dataset as shown in Fig. 1. One can see that CCICP only takes 5 iterations to converge on the *monks1* dataset, while CCCP converges with 16 iterations. Therefore, such inexact scheme makes the proposed IKLR model much more efficient, which demonstrates the efficiency of our algorithm in each iteration.

[5] http://dx.doi.org/10.5281/zenodo.12714

## 6 CONCLUSION

This paper introduced the IKLR model to consider the indefinite kernel learning in logistic regression algorithm. Despite that it shares the similar formulation with that of KLR, it is in essence non-convex and thus has to be analysed in RKKS with explicit demonstration. The proposed CCICP algorithm is able to effectively solve such non-convex problem by decomposition methods, and adopts an inexact scheme with early stopping the sub-problem to decrease the computational complexity. The convergence of our algorithm has been demonstrated with theoretical guarantees and experimental validation. Specifically, the CCICP exhibits quasi-Newton behavior or typically superlinear convergence because the convex part in our IKLR model dominates the concave part. Extensive comparative experiments from multi-modal datasets validate the superiority of the proposed IKLR model to other algorithms with positive definite/indefinite kernels. Further, the results also enlighten us to design a proper indefinite kernel and does not limit to a positive definite kernel.

## ACKNOWLEDGEMENT

This research is partly supported by NSFC, China (No. 61572315, 6151101179, 61603248), 863 Plan, China (No. 2015AA042308).

## REFERENCES

[1] FranÇois Bertrand Akoa. 2008. Combining DC Algorithms (DCAs) and Decomposition Techniques for the Training of Nonpositive-Semidefinite Kernels. *IEEE Transactions on Neural Networks* 19, 11 (2008), 1854–1872.

[2] Catherine Blake and Christopher J. Merz. 1998. UCI Repository of Machine Learning Databases. (1998). http://archive.ics.uci.edu/ml/

[3] János Bognár. 2012. *Indefinite Inner Product Spaces*. Springer Science and Business Media.

[4] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2 (2011), 27:1–27:27. Issue 3. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[5] Theodoros Evgeniou, Massimiliano Pontil, and Tomaso Poggio. 2000. Regularization Networks and Support Vector Machines. *Advances in Computational Mathematics* 13, 1 (2000), 1–50.

[6] Bernard Haasdonk. 2005. Feature Space Interpretation of SVMs with Indefinite Kernels. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27, 4 (2005), 482–492.

[7] Xiaolin Huang, Johan AK Suykens, Shuning Wang, Joachim Hornegger, and Andreas Maier. 2017. Classification With Truncated $\ell_1$ Distance Kernel. *IEEE Transactions on Neural Networks and Learning Systems* (2017).

[8] Tommi S. Jaakkola and David Haussler. 1998. Probabilistic Kernel Regression Models. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics.*

[9] Eamonn Keogh and Chotirat Ann Ratanamahatana. 2005. Exact indexing of dynamic time warping. *Knowledge and Information Systems* 7, 3 (2005), 358–386.

[10] Gaëlle Loosli, Stéphane Canu, and Soon Ong Cheng. 2016. Learning SVM in Kreĭn Spaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38, 6 (2016), 1204–1216.

[11] Cheng Soon Ong, Xavier Mary, and Alexander J. Smola. Learning with non-positive kernels. In *Proceedings of the twenty-first International Conference on Machine Learning.* 81–89.

[12] Elżbieta Pękalska and Bernard Haasdonk. 2009. Kernel Discriminant Analysis for Positive Definite and Indefinite Kernels. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31, 6 (2009), 1017–1032.

[13] Karol J Piczak. 2015. ESC: Dataset for Environmental Sound Classification. In *Proceedings of the ACM International Conference on Multimedia.* 1015–1018.

[14] Ruslan Salakhutdinov, Sam Roweis, and Zoubin Ghahramani. 2012. On the Convergence of Bound Optimization Algorithms. In *Proceedings of the nineteenth Conference on Uncertainty in Artificial Intelligence.* 509–516.

[15] Bernhard Schölkopf, Ralf Herbrich, and Alex J Smola. 2000. A Generalized Representer Theorem. In *Proceedings of the Conference on Computational Learning Theory.* 416–426.

[16] Bernhard Schölkopf and Alexander J Smola. 2003. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press.

[17] Rajiv Ratn Shah, Yi Yu, and Roger Zimmermann. 2014. ADVISOR: Personalized Video Soundtrack Recommendation by Late Fusion with Heuristic Rankings. In *Proceedings of the ACM International Conference on Multimedia.* 607–616.

[18] Bharath K. Sriperumbudur and Gert R. G. Lanckriet. 2009. On the Convergence of the Concave-Convex Procedure. In *Proceedings of the International Conference on Neural Information Processing Systems.* 1759–1767.

[19] Vladimir N. Vapnik. 2000. *The Nature of Statistical Learning Theory.* Springer.

[20] Xinxi Wang and Ye Wang. 2014. Improving Content-based and Hybrid Music Recommendation using Deep Learning. In *Proceedings of the ACM International Conference on Multimedia.* 627–636.

[21] Gang Wu, Edward Y Chang, and Zhihua Zhang. 2005. An Analysis of Transformation on Non-positive Semidefinite Similarity Matrix for Kernel Machines. In *Proceedings of the ACM twenty-second International Conference on Machine Learning.*

[22] Haiming Xu, Hui Xue, Xiaohong Chen, and Yunyun Wang. 2017. Solving Indefinite Kernel Support Vector Machine with Difference of Convex Functions Programming. In *AAAI.* 1610–1616.

[23] Alan L. Yuille and Anand Rangarajan. 2003. The Concave-Convex Procedure. *Neural Computation* 15, 4 (2003), 915–936.

[24] Ji Zhu and Trevor Hastie. 2002. Kernel Logistic Regression and the Import Vector Machine. *Journal of Computational and Graphical Statistics* 14, 1 (2002), 185–205.