Learning With Asymmetric Kernels: Least Squares and Feature Interpretation

Mingzhen He, Fan He, Lei Shi, Xiaolin Huang, *Senior Member, IEEE*, and Johan A. K. Suykens, *Fellow, IEEE*

***Abstract*—Asymmetric kernels naturally exist in real life, e.g., for conditional probability and directed graphs. However, most of the existing kernel-based learning methods require kernels to be symmetric, which prevents the use of asymmetric kernels. This paper addresses the asymmetric kernel-based learning in the framework of the least squares support vector machine named *AsK-LS*, resulting in the first classification method that can utilize asymmetric kernels directly. We will show that AsK-LS can learn with asymmetric features, namely source and target features, while the kernel trick remains applicable, i.e., the source and target features exist but are not necessarily known. Besides, the computational burden of AsK-LS is as cheap as dealing with symmetric kernels. Experimental results on various tasks, including Corel, PASCAL VOC, Satellite, directed graphs, and UCI database, all show that in the case asymmetric information is crucial, the proposed AsK-LS can learn with asymmetric kernels and performs much better than the existing kernel methods that rely on symmetrization to accommodate asymmetric kernels.**

***Index Terms*—Asymmetric kernels, directed graphs, Kullback-Leibler kernel, least squares support vector machine.**

Manuscript received 2 February 2022; revised 30 September 2022; accepted 9 March 2023. Date of publication 15 March 2023; date of current version 30 June 2023. This work was supported in part by the National Natural Science Foundation of China under Grants 61977046 and 12171093, in part by Shanghai Science and Technology Program under Grants 22511105600, 20JC1412700, and 21JC1400600, in part by Shanghai Municipal Science and Technology Major Project under Grant 2021SHZDZX0102. The research leading to these results has received funding from the European Research Council under the European Union's Horizon 2020 research and innovation program / ERC Advanced Grant E-DUALITY under Grant 787960. This paper reflects only the authors' views and the Union is not liable for any use that may be made of the contained information. This work was also supported in part by Research Council KU Leuven: Optimization frameworks for deep kernel machines under Grant C14/18/068, in part by Flemish Government: FWO projects under Grant GOA4917 N (Deep Restricted Kernel Machines: Methods and Foundations), PhD/Postdoc grant. This research received funding from the Flemish Government (AI Research Program) and Leuven AI Institute. Recommended for acceptance by C.S. Ong. *(Corresponding authors: Xiaolin Huang; Lei Shi.)*

Mingzhen He, Fan He, and Xiaolin Huang are with the Department of Automation, MOE Key Laboratory of System Control and Information Processing, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: mingzhen_he@sjtu.edu.cn; hf-inspire@sjtu.edu.cn; xiaolinhuang@sjtu.edu.cn).

Lei Shi is with the Shanghai Key Laboratory for Contemporary Applied Mathematics, School of Mathematical Sciences, Fudan University, Shanghai 200433, China, and also with the Shanghai Artificial Intelligence Laboratory, Shanghai 200232, China (e-mail: leishi@fudan.edu.cn).

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

Digital Object Identifier 10.1109/TPAMI.2023.3257351

I. Introduction

KERNEL-BASED learning [1], [2] is an important scheme in machine learning and has been widely used in classification [3], [4], regression [5], clustering [6], and many other tasks. Traditionally, a kernel used in kernel-based learning should satisfy the Mercer's condition [7]. For the Mercer's condition, there are two well-known requirements on a kernel $\mathcal{K}(\cdot,\cdot):\mathbb{R}^d\times\mathbb{R}^d\mapsto\mathbb{R}$: for samples $\{\boldsymbol{x}_i,y_i\}_{i=1}^m$, where d and m are the dimension and number of data. The kernel matrix $\boldsymbol{K}:\boldsymbol{K}_{ij}=\mathcal{K}(\boldsymbol{x}_i,\boldsymbol{x}_j)$ should be i) symmetric and ii) positive semi-definite (PSD). When the latter condition is relaxed, the flexibility is enhanced and those methods are called *indefinite learning*, for which some interesting results could be found in [8], [9], [10], [11], [12] and a review [13]. However, discussion on relaxing the symmetry condition is rare. Many asymmetric similarities exist in real life. For example, in directed graphs, the connection similarity is asymmetric: $d(\boldsymbol{x}_i,\boldsymbol{x}_j)\neq d(\boldsymbol{x}_j,\boldsymbol{x}_i)$. The conditional probability, which has been widely used to measure the directional similarity [14], is also asymmetric: $p(\boldsymbol{x}_i|\boldsymbol{x}_j)\neq p(\boldsymbol{x}_j|\boldsymbol{x}_i)$. Those asymmetric measurements cannot be used in current kernel-based learning directly.

Let us consider the support vector machine (SVM, [3])

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^m}\ \frac{1}{2}\boldsymbol{\alpha}^\top\boldsymbol{H}\boldsymbol{\alpha}-\mathbf{1}^\top\boldsymbol{\alpha},\ \text{s.t.}\ \boldsymbol{Y}^\top\boldsymbol{\alpha}=0,0\leq\boldsymbol{\alpha}\leq C\mathbf{1},\quad(1)$$

where $C>0$ is a pre-given constant, $\mathbf{1}$ is a m-dimensional vector of all ones, $\boldsymbol{Y}=[y_1,\ldots,y_m]^\top$, $\boldsymbol{\alpha}$ is a dual variable vector, and $\boldsymbol{H}:\boldsymbol{H}_{ij}=y_i\boldsymbol{K}_{ij}y_j$. When $\boldsymbol{H}$ is asymmetric, at least in (1), we can directly use it. However, by noticing that

$$\boldsymbol{\alpha}^\top\boldsymbol{H}\boldsymbol{\alpha}=\boldsymbol{\alpha}^\top\frac{\boldsymbol{H}^\top+\boldsymbol{H}}{2}\boldsymbol{\alpha},\forall\boldsymbol{\alpha}\in\mathbb{R}^m,\boldsymbol{H}\in\mathbb{R}^{m\times m},$$

one can find that only the symmetric part of an asymmetric kernel is learned when directly using it in SVM.

Another popular kernel-based learning framework is the least squares support vector machine (LS-SVM, [1], [15]). Its dual form is the following linear system

$$\begin{bmatrix}0 & \boldsymbol{Y}^\top\\ \boldsymbol{Y} & \frac{\boldsymbol{I}}{\gamma}+\boldsymbol{H}\end{bmatrix}\begin{bmatrix}b\\ \boldsymbol{\alpha}\end{bmatrix}=\begin{bmatrix}0\\ \mathbf{1}\end{bmatrix},$$

where $\boldsymbol{I}$ is an identity matrix. An interesting point here is that using an asymmetric kernel in LS-SVM will not reduce to its symmetrization and asymmetric information can be learned. Then we can develop asymmetric kernels in the LS-SVM framework in a straightforward way. The corresponding kernel trick, feature

interpretation, and asymmetric information will be investigated in this paper. Notice that we do not claim that asymmetric kernels could not be applied in SVM, but it is not straightforward and requires further investigation. Similarly, for symmetric but indefinite kernels, the solving method in the LS-SVM framework keeps easy [10], while SVM needs delicately design in form, theory, and solving-algorithm [11], [16].

In this paper, a novel method called *AsK-LS* for learning with asymmetric kernels in the framework of least squares will be established. The most important discussion is to investigate the kernel trick and the feature interpretation on how the asymmetric information could be extracted by Ask-LS. Generally, there are two features involved in the kernel trick. Using the concept in directed graphs, which have two feature embeddings for source and target nodes [17], [18], [19], we call the features in AsK-LS as *source feature* and *target feature*. The name distinguishes two features but does not mean that it can only be used for directed graphs. For the singular value decomposition, asymmetric kernels could be introduced into the LS-SVM framework [20] where the two features are related to columns and rows of the matrix, respectively.

For the discussion on learning theory, classical PD kernels induce functions in reproducing kernel Hilbert spaces, which is then extended to reproducing kernel Kerĭn spaces for indefinite kernels. To formulate the asymmetric kernel methods, reproducing kernel Banach space (RKBS) provides a suitable mathematical framework. Various RKBSs have been introduced in a wide variety of fields, including, among many others, machine learning [21], sampling reconstruction [22], and sparse approximation [23]. For more recent progress on RKBSs, one can refer to [24] and references therein. However, most existing literature only focuses on the theoretical properties of RHBSs, while how to conduct optimization in these spaces has not yet been thoroughly investigated. In this paper, we apply a generalized kernel trick to construct asymmetric kernels in Definition 1. This kernel indeed defines a reproducing kernel of a RKBS constructed in [24] (see Section 2.2 of [24] for more details).

In the rest of this paper, we will first discuss asymmetric kernels and illustrate that there are two different features embedded in an asymmetric kernel; see Section II for details. Then in Section III we will formulate AsK-LS, discuss its feature interpretation, and design the solving algorithm. In principle, an asymmetric kernel contains more information than the symmetric one. Thus, the proposed AsK-LS will demonstrate advantages when the asymmetric information is crucial, as numerically evaluated in Section IV. Section V will end this paper with a brief conclusion.

II. Asymmetric Kernels

In the classical kernel-based learning, the kernel $\mathcal{K}$ is symmetric, i.e., the kernel matrix $\boldsymbol{K}_{ij} = \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for training samples is symmetric. But there could be many asymmetric kernels. For example, in image classification tasks, the Kullback-Leibler (KL, [25]) kernel could be used to measure the similarity between two probability distributions. The directed graph is another example, where the dissimilarity between two nodes is essentially asymmetric. Notice that similarity and dissimilarity could be readily converted. In text, we choose one following usual custom. While, for kernel value, we always use similarity, i.e., the larger the kernel value is, the higher similarity is, which is also the setting of the radial basis function (RBF) kernel, the most widely used kernel.

Intuitively, $\boldsymbol{K}$ being asymmetric contains more information than that being symmetric. For symmetric kernels, the kernel trick (there are additional conditions for the existence of kernel trick; see, e.g, [7]) means that there is a feature mapping ϕ such that $\mathcal{K}(\boldsymbol{u}, \boldsymbol{v}) = \langle \phi(\boldsymbol{u}), \phi(\boldsymbol{v}) \rangle$ for two samples $\boldsymbol{u}$ and $\boldsymbol{v}$. Then it is expected that there are more features for asymmetric kernels. Fig. 1 illustrates a simple case for asymmetric (dis)similarity. In Fig. 1(b-d), we illustrate three methods which make the kernel matrix symmetric. For source dissimilarity, one can extract a nonlinear feature mapping, denoted as $\phi_s(\boldsymbol{u})$. Meanwhile, a target nonlinear feature mapping, denoted as $\phi_t(\boldsymbol{u})$ which is generally different from $\phi_s(\boldsymbol{u})$, can be also extracted. However, in the existing kernel-based learning, only symmetric kernels are acceptable and hence one has to use: symmetric similarity, for example, $(\boldsymbol{K}^\top + \boldsymbol{K})/2$ or $\boldsymbol{K}^\top \boldsymbol{K}$, which indicates that those symmetrization methods may lose asymmetric information.

Besides KL kernel and directed graphs, there are other tasks where the asymmetric kernels may be superior. In kernel density estimation problems, asymmetric kernels performed better than symmetric ones where the underlying random variables were bounded [26], [27], [28]. In Gaussian process regression tasks, Pintea et al. argued that it was helpful to set an individual kernel parameter for each data center, which enabled each data center to learn a proper kernel parameter in its neighborhood and resulted in an asymmetric kernel matrix [29]. In federated learning tasks [30], an asymmetric neural tangent kernel was established to address the issue that the gradient of the global machine was not determined by local gradient directions directly.

Our aim in this paper is to propose a novel method to directly learn with asymmetric kernels and can correspondingly learn with two feature mappings. For a long time, symmetrization is the main way for dealing with asymmetric kernels. In an early paper [31], Tsuda let the asymmetric kernel matrix $\boldsymbol{S}$ be symmetric by multiplying its transpose, then a new symmetric matrix $\boldsymbol{Q}$ was obtained as $\boldsymbol{Q} = \boldsymbol{S}^\top \boldsymbol{S}$. Munoz et al. utilized a pick-out method to convert the asymmetric kernel into the symmetric one [32]. Moreno et al. studied the KL divergence kernel $D(P, Q)$ in SVM on multimedia data [25]. They defined $D(P, Q) = \mathrm{KL(P\|Q)} + \mathrm{KL(Q\|P)}$ to satisfy the Mercer's condition, but the asymmetric information disappeared. Koide and Yamashita proposed an asymmetric kernel method and applied it to the Fisher's discriminant (AKFD) [33]. They claimed that an asymmetric kernel $\mathcal{K}(\boldsymbol{x}, \boldsymbol{y}) = \langle \phi_1(\boldsymbol{x}), \phi_2(\boldsymbol{y}) \rangle$ was generated by the inner product between two different feature mappings. In the AKFD, the decision function was assumed to be spanned by $\{\phi_1(\boldsymbol{x}_i)\}_{i=1}^m$ and input data were mapped by ϕ_2. However, the assumption of the AKFD was very strict and the situation that the decision function was spanned by $\{\phi_2(\boldsymbol{x}_i)\}_{i=1}^m$ was not considered. Wu et al. proposed a hyper asymmetric kernel method to learn with asymmetric kernels between data from two different input spaces such as query space $\mathcal{X}$ and document space $\mathcal{Y}$ [34], while an asymmetric kernel degenerated to a symmetric one when two spaces were identical, i.e., $\mathcal{X} = \mathcal{Y}$. In summary,

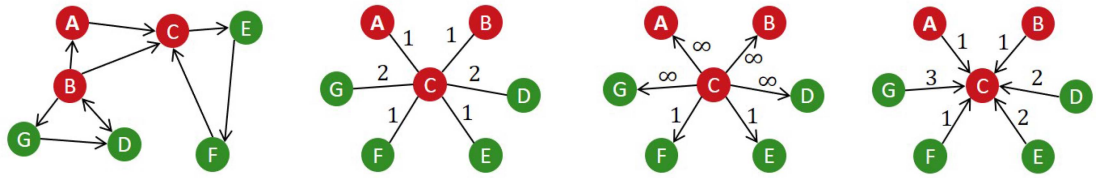

Fig. 1. Simple illustration for asymmetric dissimilarity. In order to show the asymmetric information, the dissimilarity between sample C and the other samples is shown in sub-figures (b), (c) and (d) as an example. (a) There are seven samples on a directed graph where red and green colors indicate two categories. The dissimilarity from one sample to another one is defined as the shortest path from the first sample to the latter, for example, the dissimilarity from sample A to sample E is 2 and if the sample can not be reached, dissimilarity is ∞. (b) Symmetrization. The directed edges are replaced by undirected edges. (c) Source space. The dissimilarity from C to others. (d) Target space. The dissimilarity from other samples to C.

these works used symmetrization methods at the optimization level, which canceled the asymmetric information and was not expected in the asymmetric kernel-based learning.

It was interesting that the matrix singular value decomposition (SVD) could be merged in the LS-SVM framework [20], [35]. The matrix to be decomposed could be asymmetric and even non-square, implying that LS-SVM could tolerate asymmetric kernels. From the viewpoint of the LS-SVM setting, the matrix SVD was related to two feature maps acting on the column vectors and the row vectors of the matrix, respectively. For directed graphs, it was also possible to use the adjacency matrix without the label to extract embeddings as the source and target features, respectively [17], [18], [19]. These works, although in an unsupervised setting, demonstrated that asymmetric kernels can be studied rather than through the symmetrization process.

III. Asymmetric Kernels in LS-SVM

A. PD and Indefinite Kernels in LS-SVM

Given training samples $\{\boldsymbol{x}_i, y_i\}_{i=1}^m$ with $\boldsymbol{x} \in \mathbb{R}^d$ and $y \in \{+1, -1\}$, a discriminant function $f : \mathbb{R}^d \to \mathbb{R}$ is constructed to classify the input samples. For linearly inseparable problems, a non-linear feature mapping $\phi : \mathbb{R}^d \to \mathbb{R}^p$ is needed, where $\mathbb{R}^p$ is a high-dimensional space.

LS-SVMs with positive definite (PD) kernels can be solved by the following optimization problem [15]

$$\begin{aligned} \min_{\boldsymbol{\omega},b,\xi} \quad & \frac{1}{2}\boldsymbol{\omega}^\top\boldsymbol{\omega} + \frac{\gamma}{2}\sum_{i=1}^m \xi_i^2 \\ \text{s.t.} \quad & y_i(\boldsymbol{\omega}^\top\phi(\boldsymbol{x}_i) + b) = 1 - \xi_i \\ & \forall i \in \{1, 2, \ldots, m\}, \end{aligned} \tag{2}$$

where the discriminant function f is formulated as $f(\boldsymbol{x}) = \boldsymbol{\omega}^\top\phi(\boldsymbol{x}) + b$. When the kernel is generalized to a non-PD one, the primal problem is as follows [10]

$$\begin{aligned} \min_{\boldsymbol{\omega},b,\xi} \quad & \frac{1}{2}\left(\boldsymbol{\omega}_+^\top\boldsymbol{\omega}_+ - \boldsymbol{\omega}_-^\top\boldsymbol{\omega}_-\right) + \frac{\gamma}{2}\sum_{i=1}^m \xi_i^2 \\ \text{s.t.} \quad & y_i\left(\boldsymbol{\omega}_+^\top\phi_+(\boldsymbol{x}_i) + \boldsymbol{\omega}_-^\top\phi_-(\boldsymbol{x}_i) + b\right) = 1 - \xi_i \\ & \forall i \in \{1, 2, \ldots, m\}, \end{aligned} \tag{3}$$

where $\phi_+ : \mathbb{R}^d \to \mathbb{R}^{p_1}$ and $\phi_- : \mathbb{R}^d \to \mathbb{R}^{p_2}$ are two non-linear feature mappings, both $\mathbb{R}^{p_1}$ and $\mathbb{R}^{p_2}$ are potential high-dimensional spaces. The discriminant function here is formulated as $f(\boldsymbol{x}) = \boldsymbol{\omega}_+^\top\phi_+(\boldsymbol{x}_i) + \boldsymbol{\omega}_-^\top\phi_-(\boldsymbol{x}_i) + b$.

Although the feature interpretations of (2) and (3) are not the same, their dual problems share the same formulation as below

$$\begin{bmatrix} 0 & \boldsymbol{Y}^\top \\ \boldsymbol{Y} & \frac{\boldsymbol{I}}{\gamma} + \boldsymbol{H} \end{bmatrix} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ \boldsymbol{1} \end{bmatrix}. \tag{4}$$

The kernel trick, which gives the feature interpretation of (4), is different for different types of kernels. If the kernel is PD then

$$\mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle\phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j)\rangle,$$

If the kernel is non-PD but has a positive decomposition associated with a reproducing kernel Kreĭn space (RKKS), for which the conceptual condition and a practice judgment can be found in [12], the kernel trick becomes

$$\begin{aligned} \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \langle\phi_+(\boldsymbol{x}_i), \phi_+(\boldsymbol{x}_j)\rangle - \langle\phi_-(\boldsymbol{x}_i), \phi_-(\boldsymbol{x}_j)\rangle \\ &= \mathcal{K}_+(\boldsymbol{x}_i, \boldsymbol{x}_j) - \mathcal{K}_-(\boldsymbol{x}_i, \boldsymbol{x}_j), \end{aligned}$$

where $\mathcal{K}_+$ and $\mathcal{K}_-$ are two PD kernels.

B. AsK-LS

When looking at the framework of LS-SVM from the viewpoint of solving (4), there is not any problem if $\boldsymbol{K}$ is asymmetric. It is still well-defined and a solution can be readily obtained. The key problem is to analyze what we really learn if $\boldsymbol{K}$ is asymmetric.

First, we define a generalized kernel trick to present a kernel as an inner product of two mappings ϕ_s and ϕ_t.

Definition 1. A kernel trick for a kernel $\mathcal{K} : \mathbb{R}^{d_s} \times \mathbb{R}^{d_t} \to \mathbb{R}$ can be defined by the inner product of two different feature mappings as follows:

$$\mathcal{K}(\boldsymbol{u}, \boldsymbol{v}) = \langle\phi_s(\boldsymbol{u}), \phi_t(\boldsymbol{v})\rangle, \forall \boldsymbol{u} \in \mathbb{R}^{d_s}, \boldsymbol{v} \in \mathbb{R}^{d_t},$$

where $\phi_s : \mathbb{R}^{d_s} \to \mathbb{R}^p$, $\phi_t : \mathbb{R}^{d_t} \to \mathbb{R}^p$, and $\mathbb{R}^p$ is a high-dimensional even an infinite-dimensional space.

Different from the classical kernel trick, the above definition allows different ϕ_s and ϕ_t, of which even the dimension d_s and d_t could be different. In this article, we consider the case

$d := d_s = d_t$ then both $\mathcal{K}(\boldsymbol{u}, \boldsymbol{v})$ and $\mathcal{K}(\boldsymbol{v}, \boldsymbol{u})$ are well-defined and the kernel matrix for training data is square but asymmetric. Definition 1 is compatible with the existing symmetric kernels, including PD and indefinite ones.

1) The symmetric and positive definite kernel $\mathcal{K}(u, v)$ can be defined as follows:

$$\mathcal{K}(\boldsymbol{u}, \boldsymbol{v}) = \langle \phi(\boldsymbol{u}), \phi(\boldsymbol{v}) \rangle, \forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^d,$$

in the situation when, two feature mappings ϕ_s and ϕ_t are identical $\phi := \phi_s = \phi_t$. $\phi_s, \phi_t \in \mathbb{R}^d \to \mathbb{R}^p$.

2) The symmetric and indefinite kernel $\mathcal{K}(u, v)$ can be defined as follows:

$$\begin{aligned}\mathcal{K}(\boldsymbol{u}, \boldsymbol{v}) &= \mathcal{K}_1(\boldsymbol{u}, \boldsymbol{v}) - \mathcal{K}_2(\boldsymbol{u}, \boldsymbol{v}) \\ &= \langle \phi_1(\boldsymbol{u}), \phi_1(\boldsymbol{v}) \rangle - \langle \phi_2(\boldsymbol{u}), \phi_2(\boldsymbol{v}) \rangle \\ &= \begin{bmatrix} \phi_1(\boldsymbol{u}) \\ \phi_2(\boldsymbol{u}) \end{bmatrix}^\top \begin{bmatrix} \phi_1(\boldsymbol{v}) \\ -\phi_2(\boldsymbol{v}) \end{bmatrix} \\ &:= \langle \phi_s(\boldsymbol{u}), \phi_t(\boldsymbol{v}) \rangle, \forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^d,\end{aligned}$$

in the situation when two feature mappings ϕ_s and ϕ_t are not identical, $\mathcal{K}_1$ and $\mathcal{K}_2$ are two PD kernels and $\phi_1 : \mathbb{R}^d \to \mathbb{R}^{p_1}$ and $\phi_2 : \mathbb{R}^d \to \mathbb{R}^{p_2}$ are two high-dimensional feature mappings corresponding to $\mathcal{K}_1$ and $\mathcal{K}_2$, respectively. $\mathbb{R}^{p_1}$ and $\mathbb{R}^{p_2}$ are two high-dimensional spaces.

The kernel trick associated with an asymmetric kernel contains two different feature mappings. Using the concept from directed graphs, we call them *source* and *target* features, respectively. Then for each sample, e.g., a node in a directed graph, we can extract two features from different views and classify the sample in the framework of the least squares support vector machine, which is hence called *Ask-LS*. AsK-LS in the primal space takes the following form

$$\begin{aligned}\min_{\boldsymbol{\omega}, \boldsymbol{\nu}, b_1, b_2, \boldsymbol{e}, \boldsymbol{h}} \quad & \boldsymbol{\omega}^\top \boldsymbol{\nu} + \frac{\gamma}{2} \sum_{i=1}^m e_i^2 + \frac{\gamma}{2} \sum_{i=1}^m h_i^2 \\ \text{s.t.} \quad & y_i(\boldsymbol{\omega}^\top \phi_s(\boldsymbol{x}_i) + b_1) = 1 - e_i \\ & y_i(\boldsymbol{\nu}^\top \phi_t(\boldsymbol{x}_i) + b_2) = 1 - h_i \\ & \forall i \in \{1, 2, \ldots, m\},\end{aligned} \tag{5}$$

where $e_i, h_i \in \mathbb{R}$ and γ is the regularization coefficient of misclassification errors.

In the objective function of (5), there are three terms. The latter two are to require the output to be around the label, i.e., we not only pursue classification accuracy but also prefer small within-class scatter. It is as same as that in LS-SVM and the only difference is that (5) considers both the source and target feature spaces simultaneously. The first term, i.e., $\boldsymbol{\omega}^\top \boldsymbol{\nu}$, is a regularization term in pseudo-euclidean (pE) spaces. In the SVM framework, the geometric meaning about margin in pE spaces has been discussed in [9]. A pE space $\mathbb{R}^{(p,q)}$ can be seen as a product space of real and imaginary euclidean vector spaces $\mathbb{R}^p \times \mathrm{i}\mathbb{R}^q$, where $\mathrm{i} = \sqrt{-1}$, $p, q \in \mathbb{N}_0$. A bilinear but not necessarily positive definite inner product is defined by $\langle \boldsymbol{z}, \boldsymbol{z}' \rangle_{\mathrm{pE}} = \boldsymbol{z}^\top \boldsymbol{M} \boldsymbol{z}'$ and a square norm is defined by $\|\boldsymbol{z}\|_{\mathrm{pE}}^2 = \langle \boldsymbol{z}, \boldsymbol{z} \rangle_{\mathrm{pE}}$ for indefinite learning where $\boldsymbol{z}, \boldsymbol{z}' \in \mathbb{R}^{(p,q)}$ and $\boldsymbol{M} = \mathrm{diag}(\mathbf{1}_p, -\mathbf{1}_q)$. [9] minimizes $\frac{1}{2}\|\boldsymbol{z}\|_{\mathrm{pE}}^2$ as a margin regularization. In a word, $\boldsymbol{\omega}^\top \boldsymbol{\nu}$ plays as a regularization term in the pE space $\mathbb{R}^{(p,p)}$

$$\boldsymbol{\omega}^\top \boldsymbol{\nu} = \frac{1}{2} \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{\nu} \end{bmatrix}^\top \begin{bmatrix} 0 & \boldsymbol{I_p} \\ \boldsymbol{I_p} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{\nu} \end{bmatrix} = \frac{1}{2} \boldsymbol{\mu}^\top \boldsymbol{B} \boldsymbol{\mu},$$

where $\boldsymbol{I_p}$ is a $p \times p$ identity matrix and $\boldsymbol{\mu}, \boldsymbol{B}$ are defined by

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{\nu} \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} 0 & \boldsymbol{I_p} \\ \boldsymbol{I_p} & 0 \end{bmatrix}.$$

$\boldsymbol{B}$ is a real symmetric matrix and has a spectral decomposition $\boldsymbol{B} = \boldsymbol{\Gamma} \boldsymbol{\Lambda} \boldsymbol{\Gamma}^\top$, where $\boldsymbol{\Gamma}$ is a orthogonal matrix and $\boldsymbol{\Lambda}$ is a real diagonal matrix. Notice that $\boldsymbol{B}\boldsymbol{B}^\top = \boldsymbol{\Gamma} \boldsymbol{\Lambda}^2 \boldsymbol{\Gamma}^\top = \boldsymbol{I_{2p}}$, from which it follows that $\boldsymbol{\Lambda}^2 = \boldsymbol{I_{2p}}$, i.e., eigenvalues λ_i of $\boldsymbol{B}$ are ± 1. Besides, the trace of $\boldsymbol{B}$ is 0, i.e., $\sum_{i=1}^{2p} \lambda_i = 0$. Combining the above two conditions, we know that the diagonal matrix $\boldsymbol{\Lambda} = \mathrm{diag}(\mathbf{1}_p, -\mathbf{1}_p)$ and then

$$\begin{aligned}\boldsymbol{\omega}^\top \boldsymbol{\nu} &= \frac{1}{2} \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{\nu} \end{bmatrix}^\top \begin{bmatrix} 0 & \boldsymbol{I_p} \\ \boldsymbol{I_p} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{\nu} \end{bmatrix} = \frac{1}{2} \boldsymbol{\mu}^\top \boldsymbol{B} \boldsymbol{\mu} \\ &= \frac{1}{2} \boldsymbol{\mu}^\top \boldsymbol{\Gamma} \begin{bmatrix} \boldsymbol{I_p} & 0 \\ 0 & -\boldsymbol{I_p} \end{bmatrix} \boldsymbol{\Gamma}^\top \boldsymbol{\mu} \\ &= \frac{1}{2} \langle \boldsymbol{\Gamma}^\top \boldsymbol{\mu}, \boldsymbol{\Gamma}^\top \boldsymbol{\mu} \rangle_{\mathrm{pE}} \\ &= \frac{1}{2} \|\boldsymbol{\Gamma}^\top \boldsymbol{\mu}\|_{\mathrm{pE}}^2 \\ &= \frac{1}{2} \left\| \boldsymbol{\Gamma}^\top \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{\nu} \end{bmatrix} \right\|_{\mathrm{pE}}^2.\end{aligned}$$

Therefore, similarly to the discussion in [9], $\boldsymbol{\omega}^\top \boldsymbol{\nu}$ in (5) can be regarded as a Tikhonov margin regularization in the pE space $\mathbb{R}^{(p,p)}$. This term also has appeared in discussing singular value decomposition [20], where the column and row vectors can also be regarded as two different features but SVD works in an unsupervised learning manner.

Now let us investigate the dual problem of (5), of which the kernel trick for an asymmetric kernel is crucial.

Proposition 1. Let $b_1^\star, b_2^\star, \boldsymbol{\alpha}^\star, \boldsymbol{\beta}^\star$ be the solution of the problem below, where $\boldsymbol{H}_{ij} = y_i \phi_s(\boldsymbol{x}_i)^\top \phi_t(\boldsymbol{x}_j) y_j = y_i \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) y_j$ with an asymmetric kernel $\mathcal{K}$

$$\begin{bmatrix} 0 & 0 & \boldsymbol{Y}^\top & 0 \\ 0 & 0 & 0 & \boldsymbol{Y}^\top \\ \boldsymbol{Y} & 0 & \frac{\boldsymbol{I}}{\gamma} & \boldsymbol{H} \\ 0 & \boldsymbol{Y} & \boldsymbol{H}^\top & \frac{\boldsymbol{I}}{\gamma} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \mathbf{1} \\ \mathbf{1} \end{bmatrix}, \tag{6}$$

1) $\boldsymbol{\omega}^\star$ and $\boldsymbol{\nu}^\star$ are spanned by $\{\phi_t(\boldsymbol{x}_i)\}_{i=1}^m$ and $\{\phi_s(\boldsymbol{x}_i)\}_{i=1}^m$, respectively

$$\boldsymbol{\omega}^\star = \sum_{i=1}^m \beta_i^\star y_i \phi_t(\boldsymbol{x}_i), \quad \boldsymbol{\nu}^\star = \sum_{i=1}^m \alpha_i^\star y_i \phi_s(\boldsymbol{x}_i),$$

where $(\boldsymbol{\omega}^\star, \boldsymbol{\nu}^\star)$ is a stationary point of the primal problem (5)

2) The primal problem (5) results in two discriminant functions f_s and f_t as follows:

$$\begin{cases} f_s(x) = \mathcal{K}(\boldsymbol{x}, \boldsymbol{X})(\boldsymbol{\beta}^\star \odot \boldsymbol{Y}) + b_1^\star \\ f_t(x) = \mathcal{K}(\boldsymbol{X}, \boldsymbol{x})(\boldsymbol{\alpha}^\star \odot \boldsymbol{Y}) + b_2^\star, \end{cases} \tag{7}$$

where $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1}^m$ is a training set and $\odot$ denotes a element-wise product,
$\mathcal{K}(\boldsymbol{x}, \boldsymbol{X}) = [\mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_1), \ldots, \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_m)]$,
$\mathcal{K}(\boldsymbol{X}, \boldsymbol{x}) = [\mathcal{K}(\boldsymbol{x}_1, \boldsymbol{x}), \ldots, \mathcal{K}(\boldsymbol{x}_m, \boldsymbol{x})]$.

Proof. The Lagrangian function of the primal problem (5) is formulated as follows:

$$\begin{aligned} \mathcal{L}(\Theta; \boldsymbol{\alpha}, \boldsymbol{\beta}) = \boldsymbol{\omega}^\top \boldsymbol{\nu} + \frac{\gamma}{2} \sum_{i=1}^m e_i^2 + \frac{\gamma}{2} \sum_{i=1}^m h_i^2 \\ + \sum_{i=1}^m \alpha_i (1 - e_i - y_i(\boldsymbol{\omega}^\top \phi_s(\boldsymbol{x}_i) + b_1)) \\ + \sum_{i=1}^m \beta_i (1 - h_i - y_i(\boldsymbol{\nu}^\top \phi_t(\boldsymbol{x}_i) + b_2)), \end{aligned} \tag{8}$$

where $\Theta = \{\boldsymbol{\omega}, \boldsymbol{\nu}, b_1, b_2, \boldsymbol{e}, \boldsymbol{h}\}$ is the parameter set. Then the condition of stationary points requires the following equations

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial \boldsymbol{\nu}} = \boldsymbol{\omega} - \sum_{i=1}^m \beta_i y_i \phi_t(\boldsymbol{x}_i) = 0 \\ \frac{\partial \mathcal{L}}{\partial \boldsymbol{\omega}} = \boldsymbol{\nu} - \sum_{i=1}^m \alpha_i y_i \phi_s(\boldsymbol{x}_i) = 0 \\ \frac{\partial \mathcal{L}}{\partial b_1} = \sum_{i=1}^m \alpha_i y_i = 0 \\ \frac{\partial \mathcal{L}}{\partial b_2} = \sum_{i=1}^m \beta_i y_i = 0 \\ \frac{\partial \mathcal{L}}{\partial e_i} = \gamma e_i - \alpha_i = 0 \\ \frac{\partial \mathcal{L}}{\partial h_i} = \gamma h_i - \beta_i = 0 \\ \frac{\partial \mathcal{L}}{\partial \alpha_i} = 1 - e_i - y_i(\boldsymbol{\omega}^\top \phi_s(\boldsymbol{x}_i) + b_1) = 0 \\ \frac{\partial \mathcal{L}}{\partial \beta_i} = 1 - h_i - y_i(\boldsymbol{\nu}^\top \phi_t(\boldsymbol{x}_i) + b_2) = 0. \end{cases} \tag{9}$$

The last two conditions can be converted into the following equations

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial \alpha_i} = 1 - \frac{\alpha_i}{\gamma} - y_i b_1 - y_i \sum_{j=1}^m \beta_j y_j \phi_s^\top(\boldsymbol{x}_i) \phi_t(\boldsymbol{x}_j) = 0 \\ \frac{\partial \mathcal{L}}{\partial \beta_i} = 1 - \frac{\beta_i}{\gamma} - y_i b_2 - y_i \sum_{j=1}^m \alpha_j y_j \phi_t^\top(\boldsymbol{x}_i) \phi_s(\boldsymbol{x}_j) = 0. \end{cases} \tag{10}$$

The (10) can be formulated as a linear system as follows:

$$\begin{bmatrix} 0 & 0 & \boldsymbol{Y}^\top & 0 \\ 0 & 0 & 0 & \boldsymbol{Y}^\top \\ \boldsymbol{Y} & 0 & \frac{\boldsymbol{I}}{\gamma} & \boldsymbol{Z_s Z_t}^\top \\ 0 & \boldsymbol{Y} & \boldsymbol{Z_t Z_s}^\top & \frac{\boldsymbol{I}}{\gamma} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \boldsymbol{1} \\ \boldsymbol{1} \end{bmatrix}, \tag{11}$$

where

$$\begin{cases} \boldsymbol{Z_s} = [y_1 \phi_s^\top(\boldsymbol{x}_1); \cdots; y_m \phi_s^\top(\boldsymbol{x}_m)] \\ \boldsymbol{Z_t} = [y_1 \phi_t^\top(\boldsymbol{x}_1); \cdots; y_m \phi_t^\top(\boldsymbol{x}_m)]. \end{cases}$$

According to Definition 1, an asymmetric kernel is defined as follows:

$$\mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi_s(\boldsymbol{x}_i), \phi_t(\boldsymbol{x}_j) \rangle.$$

Then, the linear system (11) can be reformulated as follows:

$$\begin{bmatrix} 0 & 0 & \boldsymbol{Y}^\top & 0 \\ 0 & 0 & 0 & \boldsymbol{Y}^\top \\ \boldsymbol{Y} & 0 & \frac{\boldsymbol{I}}{\gamma} & \boldsymbol{H} \\ 0 & \boldsymbol{Y} & \boldsymbol{H}^\top & \frac{\boldsymbol{I}}{\gamma} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \boldsymbol{1} \\ \boldsymbol{1} \end{bmatrix},$$

where $\boldsymbol{H}_{ij} = y_i \phi_s(\boldsymbol{x}_i)^\top \phi_t(\boldsymbol{x}_j) y_j = y_i \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) y_j$ with a given asymmetric kernel $\mathcal{K}$.

Suppose $b_1^\star, b_2^\star, \boldsymbol{\alpha}^\star, \boldsymbol{\beta}^\star$ be the solution of (6), according to partial derivative (9), a stationary point $(\boldsymbol{\omega}, \boldsymbol{\nu})$ can be formulated as below

$$\boldsymbol{\omega}^\star = \sum_{i=1}^m \beta_i^\star y_i \phi_t(\boldsymbol{x}_i), \quad \boldsymbol{\nu}^\star = \sum_{i=1}^m \alpha_i^\star y_i \phi_s(\boldsymbol{x}_i).$$

Since a stationary point of the primal problem (5) is obtained, two functions which classify samples from source and target points of view respectively can be formulated as follows:

$$\begin{cases} f_s(\boldsymbol{x}) = \boldsymbol{\omega}^{\star\top} \phi_s(\boldsymbol{x}) + b_1^\star = \mathcal{K}(\boldsymbol{x}, \boldsymbol{X})(\boldsymbol{\beta}^\star \odot \boldsymbol{Y}) + b_1^\star \\ f_t(\boldsymbol{x}) = \boldsymbol{\nu}^{\star\top} \phi_t(\boldsymbol{x}) + b_2^\star = \mathcal{K}(\boldsymbol{X}, \boldsymbol{x})(\boldsymbol{\alpha}^\star \odot \boldsymbol{Y}) + b_2^\star. \end{cases}$$

■

The primal-dual relationship between (5) and (6) for asymmetric kernels is characterized by Proposition 1, which also includes symmetric kernels. In that case, both the primal and dual formulations reduce to the classical LS-SVM, as shown below.

Proposition 2. When the kernel $\mathcal{K}$ in (5) is symmetric,
1) two functions f_s and f_t are identical;
2) two linear systems (6) and (4) are equivalent.

Proof. According to Definition 1, symmetric kernels can be also defined in the asymmetric kernel framework. Thus, kernels in AsK-LS can be also positive semi-definite, even indefinite.

A solution to the primal problem (5) is given by the linear system (6) which can be reformulated as follows, according to the matrix column transformation (CT) and row transformation (RT) formulas,

$$\begin{bmatrix} 0 & 0 & \boldsymbol{Y}^\top & 0 \\ 0 & 0 & 0 & \boldsymbol{Y}^\top \\ \boldsymbol{Y} & 0 & \frac{\boldsymbol{I}}{\gamma} & \boldsymbol{H} \\ 0 & \boldsymbol{Y} & \boldsymbol{H}^\top & \frac{\boldsymbol{I}}{\gamma} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \boldsymbol{1} \\ \boldsymbol{1} \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & \boldsymbol{Y}^\top & 0 \\ 0 & 0 & 0 & \boldsymbol{Y}^\top \\ \boldsymbol{Y} & 0 & \frac{\boldsymbol{I}}{\gamma} & \boldsymbol{H}^\top \\ 0 & \boldsymbol{Y} & \boldsymbol{H} & \frac{\boldsymbol{I}}{\gamma} \end{bmatrix} \begin{bmatrix} b_2 \\ b_1 \\ \boldsymbol{\beta} \\ \boldsymbol{\alpha} \end{bmatrix}.$$

Feeding the symmetric kernel $\mathcal{K}$, $\boldsymbol{H}$ is then a symmetric matrix, $\boldsymbol{H}_{ij} = y_i \phi_s(\boldsymbol{x}_i)^\top \phi_t(\boldsymbol{x}_j) y_j = y_i \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) y_j$, i.e.,

$\boldsymbol{H} = \boldsymbol{H}^\top$. The following equation holds

$$\begin{bmatrix} 0 & 0 & \boldsymbol{Y}^\top & 0 \\ 0 & 0 & 0 & \boldsymbol{Y}^\top \\ \boldsymbol{Y} & 0 & \frac{\boldsymbol{I}}{\gamma} & \boldsymbol{H} \\ 0 & \boldsymbol{Y} & \boldsymbol{H}^\top & \frac{\boldsymbol{I}}{\gamma} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} = \begin{bmatrix} 0 & 0 & \boldsymbol{Y}^\top & 0 \\ 0 & 0 & 0 & \boldsymbol{Y}^\top \\ \boldsymbol{Y} & 0 & \frac{\boldsymbol{I}}{\gamma} & \boldsymbol{H} \\ 0 & \boldsymbol{Y} & \boldsymbol{H}^\top & \frac{\boldsymbol{I}}{\gamma} \end{bmatrix} \begin{bmatrix} b_2 \\ b_1 \\ \boldsymbol{\beta} \\ \boldsymbol{\alpha} \end{bmatrix}.$$

The above equation can be simplified by moving the right term to the left.

$$\begin{bmatrix} 0 & 0 & \boldsymbol{Y}^\top & 0 \\ 0 & 0 & 0 & \boldsymbol{Y}^\top \\ \boldsymbol{Y} & 0 & \frac{\boldsymbol{I}}{\gamma} & \boldsymbol{H} \\ 0 & \boldsymbol{Y} & \boldsymbol{H}^\top & \frac{\boldsymbol{I}}{\gamma} \end{bmatrix} \begin{bmatrix} b_1 - b_2 \\ b_2 - b_1 \\ \boldsymbol{\alpha} - \boldsymbol{\beta} \\ \boldsymbol{\beta} - \boldsymbol{\alpha} \end{bmatrix} = \boldsymbol{0}$$

$$\xrightarrow{\text{RT}} \begin{bmatrix} \boldsymbol{Y} & 0 & \frac{\boldsymbol{I}}{\gamma} & \boldsymbol{H} \\ 0 & \boldsymbol{Y} & \boldsymbol{H}^\top & \frac{\boldsymbol{I}}{\gamma} \\ 0 & 0 & \boldsymbol{Y}^\top & 0 \\ 0 & 0 & 0 & \boldsymbol{Y}^\top \end{bmatrix} \begin{bmatrix} b_1 - b_2 \\ b_2 - b_1 \\ \boldsymbol{\alpha} - \boldsymbol{\beta} \\ \boldsymbol{\beta} - \boldsymbol{\alpha} \end{bmatrix}$$

$$\triangleq \boldsymbol{A} \cdot \begin{bmatrix} b_1 - b_2 \\ b_2 - b_1 \\ \boldsymbol{\alpha} - \boldsymbol{\beta} \\ \boldsymbol{\beta} - \boldsymbol{\alpha} \end{bmatrix} = \boldsymbol{0}, \tag{12}$$

where the matrix $\boldsymbol{A} \in \mathbb{R}^{(2m+2)\times(2m+2)}$ denotes the system matrix. Notice that $\boldsymbol{A}$ has a full rank and the (12) indicates that $b_1 = b_2$ and $\boldsymbol{\alpha} = \boldsymbol{\beta}$. Then two functions in (7) are identical as below,

$$\begin{aligned} f_s(\boldsymbol{x}) &= \mathcal{K}(\boldsymbol{x}, \boldsymbol{X})(\boldsymbol{\beta}^\star \odot \boldsymbol{Y}) + b_1^\star \\ &= \mathcal{K}(\boldsymbol{X}, \boldsymbol{x})(\boldsymbol{\alpha}^\star \odot \boldsymbol{Y}) + b_2^\star = f_t(\boldsymbol{x}). \end{aligned}$$

Since $b_1 = b_2, \boldsymbol{\alpha} = \boldsymbol{\beta}$ and $\boldsymbol{H}$ is a symmetric matrix, the linear system (6) can be simplified into a lower-dimensional linear system as follows:

$$\begin{bmatrix} 0 & \boldsymbol{Y}^\top \\ \boldsymbol{Y} & \frac{\boldsymbol{I}}{\gamma} + \boldsymbol{H} \end{bmatrix} \begin{bmatrix} b_1 \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ \boldsymbol{1} \end{bmatrix},$$

which is equivalent to (4) and we complete the proof.

Symmetric and asymmetric kernels lead to a unified linear system. When the kernel is asymmetric, we simultaneously obtain two functions f_s, f_t. The predictor can be f_s only or f_t only or an ensemble of f_s and f_t. Which predictor to use is actually determined by users and problems. To fully use the asymmetric information, we merge them together by two ensemble methods. Averaging is a simple way and other ensemble methods like the stacked generalization [36], [37] can be also used. Overall, we summarize the algorithm for AsK-LS in Algorithm 1.

Algorithm 1: Learning With an Asymmetric Kernel in AsK-LS.

Input: The asymmetric kernel $\mathcal{K}$, the regularization parameter γ, training data $(\boldsymbol{X}, \boldsymbol{Y})$, and test data $\boldsymbol{Z}$.
Output: The prediction $f(\boldsymbol{Z})$ of test data $\boldsymbol{Z}$.
1: Calculate an asymmetric kernel matrix $\boldsymbol{H}$ by the asymmetric kernel $\mathcal{K}$ and training data $(\boldsymbol{X}, \boldsymbol{Y})$.
2: $[\boldsymbol{\alpha}^\star, \boldsymbol{\beta}^\star, b_1^\star, b_2^\star] \leftarrow$ solve the linear system (6).
3: Predict test data from source and target views, i.e., $f_s(\boldsymbol{Z})$ and $f_t(\boldsymbol{Z})$.
4: Merge two classifiers $f_s(\boldsymbol{Z})$ and $f_t(\boldsymbol{Z})$ to obtain the final prediction $f(\boldsymbol{Z})$.
5: **return** $f(\boldsymbol{Z})$.

The complexity of AsK-LS is as same as that of symmetric LS-SVMs with the same matrix size. Specifically, the memory complexity of is $\mathcal{O}(m^2)$ and the calculation complexity is $\mathcal{O}(m^3)$. For the sparsity, the performance is also as same as that in a symmetric LS-SVM: all the training samples are support vectors which participate in constructing a decision functions. The reason is that the constrains of LS-SVMs are equalities instead of inequalities and thus are always active. When the data size is not very large, (6) is easy to solve. For large-scale problems, one can consider the fixed-size method [38], [39], Nyström approximation [40], and random Fourier features (RFF) [12], [41], [42]. For Nyström methods, the spectral decomposition does no longer hold for asymmetric kernels. In order to approximate asymmetric kernels via low-rank matrices, one could consider their singular value expansion. For RFF, an asymmetric kernel does not correspond to any well-defined probability and thus both of them are worthy of further research.

IV. Numerical Experiments

The aim of designing the AsK-LS is to learn with asymmetric kernels. As discussed before, asymmetric kernels could come from asymmetric metrics and directed graphs. The following experiments are not to claim that asymmetric kernels are better than symmetric kernels, which is surely not true since the choice of kernels is problem-dependent. Instead, we will show that when the asymmetric information is crucial, our AsK-LS can largely improve the performance of the existing kernel learning methods. The experiments are implemented in MATLAB on a PC with Intel i7-10700 K CPU (3.8 GHz) and 32 GB memory. All the reported results are the average over 10 trials. The source code of our implementation can be found in https://github.com/AlexHe123/AsK-LS.

A. Kullback-Leibler Kernel

Kullback-Leibler divergence is the measure of how one probability distribution Q is different from another probability distribution P

$$\mathrm{KL}(P||Q) = \int_{-\infty}^{\infty} P(x) \log \frac{P(x)}{Q(x)} dx, \tag{13}$$

TABLE I
MICRO-/MACRO-F1 SCORES (MEAN ± STD) OF DIFFERENT METHODS WITH THE KL KERNEL BETWEEN GMMS ON SEVERAL DATASETS

Datasets	#classes	#train	#test	F1 Score	SVM	LS-SVM	AsK-LS
Corel	10	900	100	Micro	0.830±0.001	0.840±0.001	**0.916±0.005**
				Macro	0.822±0.001	0.832±0.001	**0.914±0.005**
PASCAL	10	900	200	Micro	0.337±0.002	0.308±0.006	**0.397±0.002**
				Macro	0.337±0.003	0.293±0.007	**0.394±0.002**
Satellite	4	1600	160	Micro	0.959±0.003	0.969±0.001	**0.990±0.003**
				Macro	0.959±0.003	0.969±0.001	**0.990±0.003**

which is asymmetric. KL divergence has been successfully used in many fields, e.g., VAE [43] and GAN [44], [45]. From KL divergence, one can evaluate the similarity between two probability distributions by the KL kernel as follows:

$$\mathcal{K}(s_i, s_j) = \exp\left(-a \cdot \mathrm{KL}(P_i||P_j)\right), \tag{14}$$

where $a > 0$ is the hyperparameter for kernel bandwidth, s denotes a sample, and P_i, P_j are probability distributions estimated from samples s_i and s_j, respectively. a is tuned by 10-fold cross-validation on a parameter grid $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}\}$. Since KL kernel is asymmetric, the kernel matrix is also asymmetric. AsK-LS can be utilized to directly learn with the asymmetric KL kernel rather than its symmetry [25].

We conduct image classification experiments on Corel [46], Satellite[1], and PASCAL VOC 2007 [47]. The Corel database contains 80 concept groups, including, e.g., autumn, aviation, dog, and elephant. Of these groups, 10 classes are picked: beaches, bus, dinosaurs, elephants, flowers, foods, horses, monuments, snow mountains, and African people & villages. There are 100 images per class: 90 for training and 10 for test. The Satellite dataset is a remote sensing image set containing four classes: cloudy, desert, green area, and water. There are 440 images for each class: 400 for training and 40 for test. The PASCAL VOC 2007 dataset has four concept groups: person, animal, vehicle and indoor. 10 classes in those groups are chosen: aeroplane, bicycle, bird, boat, bus, car, cat, dog, horse and person. There are 110 images for each class: 90 for training and 20 for test. For each image, we follow the standard feature extraction method [25] to obtain a 64-dimensional discrete cosine transform feature vector $\boldsymbol{X}$.

In the experiment, we use the Gaussian mixture model (GMM) with 256 diagonal Gaussian components to estimate the probability distribution $P_i(\boldsymbol{X}_i)$ of the image feature vector sequence $\boldsymbol{X}_i$. Since the KL divergence for two GMMs can not be calculated by (13) directly, we use the Monte Carlo method to calculate it. Then kernel value between two images is given by the KL kernel similarity (14) of these two probability distributions.

The corresponding asymmetric KL divergence has been widely used in learning tasks. However, due to that KL divergence violates the symmetry requirement, it has never been directly used in the kernel-based learning. Now with the proposed AsK-LS, we can use it in classification tasks. As a comparison, SVM and LS-SVM are used with a symmetric KL kernel [25]. For the parameters, i.e., the hyperparameter a and regularization parameter γ in all the three models is tuned by 10-fold cross-validation. Our AsK-LS outputs two classifiers and here we simply average them.

The image classification is a multi-classes task where we utilize the one-vs-rest scheme, for which the average Micro-F1 and Macro-F1 scores are reported in Table I. AsK-LS achieves better performance on the PASCAL, Corel and Satellite datasets, showing that learning with the asymmetric information can help.

B. Directed Adjacency Matrix

Nodes classification with an asymmetric adjacency matrix is a task that needs to learn from asymmetric metrics. In this task, nodes in a directed graph $\mathcal{H} = \{\mathcal{N}, \mathcal{E}\}$ with nodes set $\mathcal{N}$ and edges set $\mathcal{E}$ are classified. Its adjacency matrix showing the connection among nodes is defined as follows:

$$\mathcal{A}_{ij} = \begin{cases} 1, & \text{if } j \to i, \\ 0, & \text{otherwise.} \end{cases}$$

where $j \to i$ means that there is a link pointing to node i from node j. This model has wide applications and here we consider five directed graphs, namely Cora, Citeseer, and Pubmed [48], AM-photo, and AM-computer [49]. Details of the data set could be found in Table II. For the first three widely used graphs, the nodes and edges present documents and citations, respectively. For the latter two, the nodes present goods, and the edges mean the two goods are frequently bought together. The node classification is originally a multi-classes task where we utilize the one-vs-rest scheme and focus on Micro-F1 and Macro-F1 scores.

A directed graph is fully characterized by its adjacency matrix. However, in existing kernel methods, one cannot directly use the adjacency matrix as a kernel. Therefore, the current mainstream is to first do feature embedding [17], [18], [19] to extract the asymmetric information and then do classification based on the extracted features. In the experiment, For the embedding, we use a SOTA embedding method NERD [19], which utilizes random walk to extract node embeddings $\phi_s(v)$ and $\phi_t(v)$ as source feature and target feature of each node $v \in \mathcal{N}$ on a directed graph. We combine them as a unified feature $\phi(v) = [\phi_s^\top(v), \phi_t^\top(v)]^\top$ which then defines a symmetric kernel

[1]https://www.kaggle.com/datasets/mahmoudreda55/satellite-image-classification

TABLE II
THE DETAILED INFORMATION OF USED DIRECTED GRAPH DATASETS

Datasets	Cora	Citeseer	Pubmed	AM-photo	AM-computer
#Classes	7	6	3	8	10
#Nodes	2078	3327	19717	7650	13752
#Edges	5429	4732	44338	143663	287209
λ_{max}	1.2688	1.3398	1.2578	1.4195	1.5163
λ_{min}	-0.2060	-0.1684	-0.2986	-0.9778	-1.1002

TABLE III
MICRO-/MACRO-F1 SCORES (MEAN ± STD) OF DIFFERENT ALGORITHMS ON THE NODES CLASSIFICATION TASK

Datasets	F1 Score	Embedding		Graph		
		SVM	LS-SVM	SVM	LS-SVM	AsK-LS
Cora	Micro	0.740±0.010	0.733±0.009	0.305±0.010	0.713±0.015	**0.753±0.013**
	Macro	0.730±0.012	0.724±0.011	0.077±0.004	0.708±0.016	**0.748±0.013**
Citeseer	Micro	0.497±0.013	0.507±0.010	0.393±0.042	0.596±0.011	0.605±0.012
	Macro	0.454±0.013	0.452±0.010	0.301±0.044	0.560±0.012	0.579±0.012
Pubmed	Micro	**0.732±0.006**	0.726±0.004	0.602±0.022	0.635±0.006	0.719±0.004
	Macro	0.680±0.006	0.669±0.005	0.451±0.018	0.503±0.006	**0.698±0.005**
AM-photo	Micro	0.858±0.005	0.834±0.006	0.319±0.022	0.799±0.008	**0.885±0.005**
	Macro	0.834±0.005	0.767±0.006	0.155±0.018	0.732±0.008	**0.874±0.005**
AM-computer	Micro	0.606±0.005	0.609±0.004	0.400±0.005	0.619±0.014	**0.868±0.002**
	Macro	0.429±0.004	0.454±0.003	0.107±0.003	0.443±0.016	**0.846±0.005**

Datasets	F1 Score	Graph				
		Flip	Clip	Shift	KSVM	IKSVM-DC
Cora	Micro	0.311±0.012	0.310±0.013	0.309±0.012	0.726±0.015	0.715±0.001
	Macro	0.079±0.005	0.078±0.006	0.079±0.006	0.726±0.014	0.712±0.001
Citeseer	Micro	0.403±0.043	0.397±0.046	0.401±0.048	**0.609±0.014**	0.592±0.011
	Macro	0.312±0.049	0.307±0.048	0.312±0.053	**0.584±0.012**	0.568±0.011
Pubmed	Micro	0.594±0.012	0.616±0.019	0.610±0.023	0.658±0.012	0.396±0.002
	Macro	0.444±0.010	0.463±0.016	0.457±0.020	0.627±0.013	0.190±0.002
AM-photo	Micro	0.390±0.039	0.344±0.018	0.324±0.019	0.832±0.012	0.237±0.017
	Macro	0.221±0.027	0.188±0.015	0.159±0.013	0.818±0.014	0.080±0.013
AM-computer	Micro	0.406±0.006	0.401±0.005	0.401±0.007	0.790±0.012	0.375±0.001
	Macro	0.119±0.003	0.115±0.003	0.104±0.003	0.747±0.024	0.055±0.001

that can be used in classical kernel-based learning methods, e.g., SVM and LS-SVM, and results are reported in Table III.

Existing kernel machines based on symmetric similarities such as SVMs and LS-SVMs and their variations are chosen as compared methods. The adjacency matrix by symmetrization $(\mathcal{A}+\mathcal{A}^\top)/2$ is fed to these classical but symmetric methods. The maximum and minimum eigenvalues of symmetric adjacency matrices are shown in Table II. It can be observed that the Gram matrices are non-PSD/indefinite. There are generally two ways to deal with indefinite kernels: spectrum modification and non-convex optimization. In spectrum modification, the indefinite kernel matrix is converted into a PSD matrix by operations on spectrum. We consider three methods: "Flip" using the absolute values of eigenvalues [50]; "Clip" setting negative eigenvalues be zeros [51]; "Shift" adding a positive numbers to all eigenvalues until the smallest eigenvalue is zero [52]. In non-convex optimization, we consider two methods: "KSVM" [16] and "IKSVM-DC" [53]. The former regrades the indefinite

TABLE IV
CLASSIFICATION ACCURACY (MEAN ± STD) OF LS-SVM AND ASK-LS WITH SEVERAL KERNELS ON THE UCI DATABASE

Datasets	LS-SVM			AsK-LS	
	TL1	tanh	RBF	SNE	T
heart	0.828±0.019	0.775±0.047	**0.837±0.032**	0.824±0.025	0.825±0.031
sonar	0.732±0.008	0.537±0.064	0.856±0.001	0.854±0.028	**0.865±0.027**
monks1	0.741±0.010	0.762±0.007	**0.791±0.012**	0.790±0.014	0.778±0.012
monks2	0.671±0.001	0.699±0.036	0.841±0.007	**0.866±0.016**	**0.866±0.014**
monks3	**0.961±0.016**	0.914±0.001	0.936±0.003	0.936±0.004	0.901±0.004
pima	0.750±0.017	0.749±0.029	0.738±0.026	0.749±0.026	**0.752±0.021**
australian	0.856±0.018	0.855±0.020	**0.862±0.015**	0.854±0.019	0.859±0.032
spambase	0.918±0.014	0.919±0.004	**0.925±0.007**	0.908±0.007	0.922±0.018

SVM as a min-max problem in reproducing kernel Kreĭn spaces. The latter reformulates the indefinite problem as a difference of convex functions and use the related algorithm [54].

Since the asymmetric information can be extracted by the embedding method, the performance of that is much better than using existing kernel methods with symmetric adjacency matrices, of which micro-/macro- F1 scores are listed in Table III.

Now we have the AsK-LS and can directly learn with the adjacency matrix without the feature embedding. Before sending the adjacency matrix to AsK-LS, we pre-process it by its in-degree $d_i = \sum_{j=1}^{m} \mathcal{A}_{ij}$ (the same pre-process is also applied for SVM and LS-SVM). The Micro- and Macro-F1 scores of AsK-LS are reported in the last column in Table III. The performance is much better than SVM and LS-SVM with the symmetrization of the kernel, indicating that the asymmetric information is helpful. For the comparison with the SOTA embedding methods, AsK-LS is better or at least comparable, showing the effectiveness of AsK-LS for extracting the asymmetric information.

C. Symmetric or Asymmetric Kernels

We have shown that the proposed AsK-LS can learn with asymmetric kernels. Since asymmetric kernels are more general than symmetric ones, learning with asymmetric kernels is promising to get improvement, if there is an efficient method to get a suitable kernel. In the case in our paper, kernels are pre-given, then the performance is determined by the choice of kernels. In the previous sections, the asymmetric information, i.e., measuring the difference by KL divergence and directed distance, is important, thus the corresponding asymmetric kernels lead to a good performance. Although the asymmetric kernels are more flexible, a specific asymmetric kernel is not necessarily better than a symmetric one. Especially, there have been study on symmetric kernels that have good performance on many tasks, e.g., the RBF kernel [55], the truncated ℓ_1 (TL1) kernel [56], and the tanh kernel [57]

We conduct classification experiments on several datasets from the UCI database [58], where 60% of the data are randomly picked up for training and the rest for test. Two asymmetric kernels, called *SNE kernel* and *T kernel*, are considered here. They have been used for dimension reduction [14] but have not been used for classification, since before there was no classification method that can learn with asymmetric kernels directly. The formulations are given as below,

1) The SNE kernel with the parameter $\sigma \in \mathbb{R}$

$$\mathcal{K}_{\mathrm{SNE}}(\boldsymbol{x}, \boldsymbol{y}) = \frac{\exp(-\|\boldsymbol{x} - \boldsymbol{y}\|_2^2/\sigma^2)}{\sum_{\boldsymbol{z}\in\boldsymbol{Z}} \exp(-\|\boldsymbol{x} - \boldsymbol{z}\|_2^2/\sigma^2)},$$

2) The T kernel

$$\mathcal{K}_{\mathrm{T}}(\boldsymbol{x}, \boldsymbol{y}) = \frac{(1 + \|\boldsymbol{x} - \boldsymbol{y}\|_2^2)^{-1}}{\sum_{\boldsymbol{z}\in\boldsymbol{Z}} (1 + \|\boldsymbol{x} - \boldsymbol{z}\|_2^2)^{-1}},$$

3) The TL1 kernel with the parameter ρ

$$\mathcal{K}_{\mathrm{TL1}}(\boldsymbol{x}, \boldsymbol{y}) = \max\{\rho - \|\boldsymbol{x} - \boldsymbol{y}\|_{\ell_1}, 0\},$$

4) The tanh kernel with parameters c, t

$$\mathcal{K}_{\mathrm{T}}(\boldsymbol{x}, \boldsymbol{y}) = \tanh\left(c\boldsymbol{x}^\top \boldsymbol{y} + t\right),$$

5) The RBF kernel with the parameter $\sigma \in \mathbb{R}$

$$\mathcal{K}_{\mathrm{RBF}}(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\|\boldsymbol{x} - \boldsymbol{y}\|_2^2/\sigma^2).$$

where $\boldsymbol{Z}$ stands for the training set, $\boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{Z}$ and $\boldsymbol{z}$ is defined as an element belonging to $\boldsymbol{Z}$. All the parameters are tuned by 10-fold cross-validation. The parameter σ is turned on a grid $\{2^{-3}, 2^{-2}, 2^{-1}, 1, 2, 5, 10\}$. The TL1 kernel is an indefinite kernel. As discussed in [56], the performance of the TL1 kernel is not sensitive to ρ, we set $\rho = 0.7\ d$ as suggested where d is the input dimension. The tanh kernel is indefinite when $c < 0$. c is turned by 10-folds cross-validation on a grid, of which the values uniformly lie on $[-10, 10]$. And d is turned by 10-folds cross-validation on $\{0, 1, 5, 10, 15\}$

The classification accuracy of AsK-LS with the two asymmetric kernels is reported in Table IV, together with the accuracy of LS-SVM with the TL1, tanh and RBF kernels. Generally, the best choice of kernels is problem-dependent and one cannot assert which kernel is good in advance. But at least, the proposed AsK-LS makes it possible to use asymmetric kernels. With delicately designing or efficiently learning, asymmetric kernels could lead to a good performance.

V. CONCLUSION

In this paper, we investigate the least squares support vector machine with asymmetric kernels in theoretical and algorithmic aspects. The proposed AsK-LS is the first model that can learn

with asymmetric kernels. The primal and dual representations for AsK-LS are given, showing the feature interpretation that there are two different functions, can be simultaneously learned from the source and the target views. In numerical experiments, when the asymmetric information is physically important, the AsK-LS with asymmetric kernels significantly outperforms SVM and LS-SVM that can only deal with symmetric kernels.

The most significant contribution of this paper is to make asymmetric kernels useful in the kernel-based learning. In methodology, the least squares framework is not the unique way to accommodate asymmetric kernels. Models from other kernel-based methods, e.g., the support vector machine and the logistic regression, etc., are worthy to be investigated. In theory, the functional spaces associated with asymmetric kernels are interesting, which is beyond reproducing kernel Hilbert spaces for PD kernels or reproducing kernel Kreĭn spaces for indefinite kernels, but falls in a reproducing kernel Banach space. The proposed algorithm fully exploits the structure of the feature mapping and the data similarity encoding the asymmetry of the kernels. The theoretical behavior of our algorithm can be further studied under the framework of [24], which we will leave as further work. In application, one can design asymmetric kernels for different tasks, especially those that involve directional relationships, including but not limited to directed graphs, the distribution distance [59], the causality analysis [60], [61], and the optimal transport [62], [63].

Acknowledgments

The authors would like to thank the anonymous reviewers for their insightful comments.

References

[1] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle, *Least Squares Support Vector Machines*. Singapore: World Scientific, 2002.
[2] V. Vapnik, *The Nature of Statistical Learning Theory*. Berlin, Germany: Springer Science & Business Media, 2013.
[3] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.
[4] K. Soman, R. Loganathan, and V. Ajay, *Machine Learning With SVM and Other Kernel Methods*, Delhi, India: PHI Learning Pvt. Ltd., 2009.
[5] H. Drucker et al., "Support vector regression machines," *Adv. Neural Inf. Process. Syst.*, vol. 9, pp. 155–161, 1997.
[6] M. Girolami, "Mercer kernel-based clustering in feature space," *IEEE Trans. Neural Netw.*, vol. 13, no. 3, pp. 780–784, May 2002.
[7] H. Jeffreys, B. Jeffreys, and B. Swirles, *Methods of Mathematical Physics*. Cambridge, U.K.: Cambridge Univ. Press, 1999.
[8] C. S. Ong, X. Mary, S. Canu, and A. J. Smola, "Learning with non-positive kernels," in *Proc. 21st Int. Conf. Mach. Learn.*, 2004, pp. 639–646.
[9] 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.
[10] 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.*, vol. 43, no. 1, pp. 162–172, 2017.
[11] D. Oglic and T. Gärtner, "Scalable learning in reproducing kernel Kreĭn spaces," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 4912–4921.
[12] F. Liu, X. Huang, Y. Chen, and J. A. K. Suykens, "Fast learning in reproducing kernel Kreĭn spaces via signed measures," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2021, pp. 388–396.
[13] F.-M. Schleif and P. Tino, "Indefinite proximity learning: A review," *Neural Comput.*, vol. 27, no. 10, pp. 2039–2096, 2015.
[14] G. Hinton and S. T. Roweis, "Stochastic neighbor embedding," in *Proc. Adv. Neural Inf. Process. Syst.*, Citeseer, 2002, pp. 833–840.
[15] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*, vol. 9, no. 3, pp. 293–300, 1999.
[16] G. Loosli, S. Canu, and C. S. Ong, "Learning SVM in Kreĭn spaces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 6, pp. 1204–1216, Jun. 2015.
[17] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric transitivity preserving graph embedding," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 1105–1114.
[18] C. Zhou, Y. Liu, X. Liu, Z. Liu, and J. Gao, "Scalable graph embedding for asymmetric proximity," in *Proc. AAAI Conf. Artif. Intell.*, 2017, pp. 2942–2948.
[19] M. Khosla, J. Leonhardt, W. Nejdl, and A. Anand, "Node representation learning for directed graphs," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discov. Databases*, Springer, 2019, pp. 395–411.
[20] J. A. K. Suykens, "SVD revisited: A new variational principle, compatible feature maps and nonlinear extensions," *Appl. Comput. Harmon. Anal.*, vol. 40, no. 3, pp. 600–609, 2016.
[21] H. Zhang, Y. Xu, and J. Zhang, "Reproducing kernel banach spaces for machine learning," *J. Mach. Learn. Res.*, vol. 10, no. 12, pp. 2741–2775, 2009.
[22] M. Z. Nashed and Q. Sun, "Sampling and reconstruction of signals in a reproducing kernel subspace of LP (RD)," *J. Funct. Anal.*, vol. 258, no. 7, pp. 2422–2452, 2010.
[23] Y. Xu and Q. Ye, *Generalized Mercer Kernels and Reproducing Kernel Banach Spaces*. American Mathematical Society, 2019.
[24] R. R. Lin, H. Z. Zhang, and J. Zhang, "On reproducing kernel banach spaces: Generic definitions and unified framework of constructions," *Acta Mathematica Sinica, English Ser.*, vol. 38, no. 8, pp. 1459–1483, 2022.
[25] P. J. Moreno, P. Ho, and N. Vasconcelos, "A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications," in *Proc. Adv. Neural Inf. Process. Syst.*, 2003, pp. 1385–1392.
[26] M. Mackenzie and A. K. Tieu, "Asymmetric kernel regression," *IEEE Trans. Neural Netw.*, vol. 15, no. 2, pp. 276–282, Mar. 2004.
[27] C. Kuruwita, K. Kulasekera, and W. Padgett, "Density estimation using asymmetric kernels and bayes bandwidths with censored data," *J. Statist. Plan. Inference*, vol. 140, no. 7, pp. 1765–1774, 2010.
[28] H. L. Koul and W. Song, "Large sample results for varying kernel regression estimates," *J. Nonparametric Statist.*, vol. 25, no. 4, pp. 829–853, 2013.
[29] S. L. Pintea, J. C. van Gemert, and A. W. Smeulders, "Asymmetric kernel in gaussian processes for learning target variance," *Pattern Recognit. Lett.*, vol. 108, pp. 70–77, 2018.
[30] B. Huang, X. Li, Z. Song, and X. Yang, "FL-NTK: A neural tangent kernel-based framework for federated learning analysis," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 4423–4434.
[31] K. Tsuda, "Support vector classifier with asymmetric kernel functions," in *Proc. Eur. Symp. Artif. Neural Netw.*, 1999, pp. 183–188.
[32] A. Munoz, I. M. de Diego, and J. M. Moguerza, "Support vector machine classifiers for asymmetric proximities," in *Proc. Artif. Neural Netw. Neural Inf. Process.*, Springer, 2003, pp. 217–224.
[33] N. Koide and Y. Yamashita, "Asymmetric kernel method and its application to fisher's discriminant," in the *Proc. 18th Int. Conf. Pattern Recognit.*, 2006, pp. 820–824.
[34] W. Wu, J. Xu, H. Li, and S. Oyama, "Asymmetric kernel learning," Microsoft Research, Tech. Rep. MSR-TR-2010-85, Jun. 2010. [Online]. Available: https://www.microsoft.com/en-us/research/publication/asymmetric-kernel-learning/
[35] J. A. K. Suykens, "Deep restricted kernel machines using conjugate feature duality," *Neural Comput.*, vol. 29, no. 8, pp. 2123–2163, 2017.
[36] D. H. Wolpert, "Stacked generalization," *Neural Netw.*, vol. 5, no. 2, pp. 241–259, 1992.
[37] K. M. Ting and I. H. Witten, "Stacking bagged and dagged models," in *Proc. 14th Int. Conf. Mach. Learn.*, 1997, pp. 367–375.
[38] M. Espinoza, J. A. K. Suykens, and B. De Moor, "Fixed-size least squares support vector machines: A large scale application in electrical load forecasting," *Comput. Manage. Sci.*, vol. 3, no. 2, pp. 113–129, 2006.
[39] K. De Brabanter, J. De Brabanter, J. A. K. Suykens, and B. De Moor, "Optimized fixed-size kernel models for large data sets," *Comput. Statist. Data Anal.*, vol. 54, no. 6, pp. 1484–1504, 2010.
[40] C. Williams and M. Seeger, "Using the Nyström method to speed up kernel machines," in *Proc. 14th Annu. Conf. Neural Inf. Process. Syst.*, 2001, pp. 682–688.

[41] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Proc. 20th Int. Conf. Neural Inf. Process. Syst.*, 2007, pp. 1177–1184.
[42] F. Liu, X. Huang, Y. Chen, and J. A. K. Suykens, "Random features for kernel approximation: A survey on algorithms, theory, and beyond," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 10, pp. 7128–7148, Oct. 2021.
[43] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in the *Proc. 2nd Int. Conf. Learn. Representations*, 2014.
[44] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4401–4410.
[45] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, "A review on generative adversarial networks: Algorithms, theory, and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 4, pp. 3313–3332, Apr. 2021.
[46] J. Z. Wang, J. Li, and G. Wiederhold, "SIMPLIcity: Semantics-sensitive integrated matching for picture libraries," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 9, pp. 947–963, Sep. 2001.
[47] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes challenge 2007 (VOC2007) results," 2007. [Online]. Available: http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html
[48] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, pp. 93–93, 2008.
[49] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann, "Pitfalls of graph neural network evaluation," 2018, *arXiv:1811.05868*.
[50] T. Graepel, R. Herbrich, P. Bollmann-Sdorra, and K. Obermayer, "Classification on pairwise proximity data," in *Proc. 11th Int. Conf. Neural Inf. Process. Syst.*, 1998, pp. 438–444.
[51] E. Pekalska, P. Paclik, and R. P. Duin, "A generalized kernel approach to dissimilarity-based classification," *J. Mach. Learn. Res.*, vol. 2, pp. 175–211, 2001.
[52] V. Roth, J. Laub, M. Kawanabe, and J. M. Buhmann, "Optimal cluster preserving embedding of nonmetric proximity data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 12, pp. 1540–1551, Dec. 2003.
[53] H.-M. Xu, H. Xue, X.-H. Chen, and Y.-Y. Wang, "Solving indefinite kernel support vector machine with difference of convex functions programming," in *Proc. AAAI Conf. Artif. Intell.*, vol. 31, no. 1, 2017.
[54] H. A. Le Thi and T. Pham Dinh, "DC programming and DCA: Thirty years of developments," *Math. Program.*, vol. 169, no. 1, pp. 5–68, 2018.
[55] J.-P. Vert, K. Tsuda, and B. Schölkopf, "A primer on kernel methods," *Kernel Methods Comput. Biol.*, vol. 47, pp. 35–70, 2004.
[56] X. Huang, J. A. K. Suykens, S. Wang, J. Hornegger, and A. Maier, "Classification with truncated ℓ_1 distance kernel," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 2025–2030, May 2018.
[57] 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," *Neural Comput.*, vol. 3, pp. 1–32, 2003.
[58] C. Blake and C. Merz, "UCI repository of machine learning databases," 1998. [Online]. Available: http://archive.ics.uci.edu/ml
[59] S. K. Zhou and R. Chellappa, "From sample similarity to ensemble similarity: Probabilistic distance measures in reproducing kernel Hilbert space," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 6, pp. 917–929, Jun. 2006.
[60] K. Radinsky, S. Davidovich, and S. Markovitch, "Learning causality for news events prediction," in *Proc. 21st Int. Conf. World Wide Web*, 2012, pp. 909–918.
[61] H. Xu, M. Farajtabar, and H. Zha, "Learning granger causality for Hawkes processes," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1717–1726.
[62] N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonjy, "Optimal transport for domain adaptation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 9, pp. 1853–1865, Sep. 2016.
[63] B. Su and G. Hua, "Order-preserving optimal transport for distances between sequences," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 12, pp. 2961–2974, Dec. 2019.

Mingzhen He received the BS degree from the South China University of Technology, Guangzhou, China, in 2020. He is currently working toward the PhD degree with the Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, Shanghai, China. His current research interest is kernel learning.

Fan He received the BS and ME degrees from Shanghai Jiao Tong University, Shanghai, China, in 2017 and 2019, respectively. She is currently working toward the PhD degree with the Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University. Her current research interests include kernel learning and distributed algorithms.

Lei Shi received the joint PhD degree in applied mathematics from the City University of Hong Kong, Hong Kong, and the University of Science and Technology of China, Hefei, China, in 2010. He is now a full professor with the School of Mathematical Sciences, Fudan University, Shanghai, China. His current research interests include learning theory and approximation theory.

Xiaolin Huang (Senior Member, IEEE) received the BS degree in control science and engineering, the BS degree in applied mathematics from Xi'an Jiaotong University, Xi'an, China, in 2006, and the PhD degree in control science and engineering from Tsinghua University, Beijing, China. From 2012 to 2015, he worked as a postdoctoral researcher with ESAT-STADIUS, KU Leuven, Leuven, Belgium. After that he was selected as an Alexander von Humboldt fellow and working with Pattern Recognition Lab, the Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany. From 2016, he has been an associate professor with the Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, Shanghai, China. In 2017, he was awarded by "1000-Talent Plan" (Young Program). His current research interests include machine learning and optimization, especially for robustness and sparsity of both kernel learning and deep neural networks.

Johan A. K. Suykens (Fellow, IEEE) received the master's degree in electro-mechanical engineering and the PhD degree in applied sciences from the Katholieke Universiteit Leuven, in 1989 and 1995, respectively. In 1996 he has been a visiting postdoctoral researcher with the University of California, Berkeley. He has been a postdoctoral researcher with the Fund for Scientific Research FWO Flanders and is currently a full Professor with KU Leuven. He is author of the books "Artificial Neural Networks for Modelling and Control of Non-linear Systems" (Kluwer Academic Publishers) and "Least Squares Support Vector Machines" (World Scientific), co-author of the book "Cellular Neural Networks, Multi-Scroll Chaos and Synchronization" (World Scientific) and editor of the books "Nonlinear Modeling: Advanced Black-Box Techniques" (Kluwer Academic Publishers), "Advances in Learning Theory: Methods, Models and Applications" (IOS Press) and "Regularization, Optimization, Kernels, and Support Vector Machines" (Chapman & Hall/CRC). In 1998 he organized an International Workshop on Nonlinear Modelling with Time-series Prediction Competition. He has served as associate editor for *IEEE Transactions on Circuits and Systems* (1997-1999 and 2004-2007), *IEEE Transactions on Neural Networks* (1998-2009), *IEEE Transactions on Neural Networks and Learning Systems* (from 2017) and *IEEE Transactions on Artificial Intelligence* (from April 2020). He received an IEEE Signal Processing Society 1999 Best Paper Award, a 2019 Entropy Best Paper Award and several Best Paper Awards at International Conferences. He is a recipient of the International Neural Networks Society INNS 2000 Young Investigator Award for significant contributions in the field of neural networks. He has served as a director and organizer of the NATO Advanced Study Institute on Learning Theory and Practice (Leuven 2002), as a program co-chair for the International Joint Conference on Neural Networks 2004 and the International Symposium on Nonlinear Theory and its Applications 2005, as an organizer of the International Symposium on Synchronization in Complex Networks 2007, a co-organizer of the NIPS 2010 workshop on Tensors, Kernels and Machine Learning, and chair of ROKS 2013.He has been awarded an ERC Advanced Grant 2011 and 2017, has been elevated IEEE fellow 2015 for developing least squares support vector machines, and is ELLIS fellow. He is currently serving as program director of master AI with KU Leuven.