Laplacian Welsch Regularization for Robust Semisupervised Learning

Jingchen Ke, Chen Gong, *Member, IEEE*, Tongliang Liu, *Member, IEEE*, Lin Zhao, Jian Yang, *Member, IEEE*, and Dacheng Tao, *Fellow, IEEE*

***Abstract*—Semisupervised learning (SSL) has been widely used in numerous practical applications where the labeled training examples are inadequate while the unlabeled examples are abundant. Due to the scarcity of labeled examples, the performances of the existing SSL methods are often affected by the outliers in the labeled data, leading to the imperfect trained classifier. To enhance the robustness of SSL methods to the outliers, this article proposes a novel SSL algorithm called Laplacian Welsch regularization (LapWR). Specifically, apart from the conventional Laplacian regularizer, we also introduce a bounded, smooth, and nonconvex Welsch loss which can suppress the adverse effect brought by the labeled outliers. To handle the model nonconvexity caused by the Welsch loss, an iterative half-quadratic (HQ) optimization algorithm is adopted in which each subproblem has an ideal closed-form solution. To handle the large datasets, we further propose an accelerated model by utilizing the Nyström method to reduce the computational complexity of LapWR. Theoretically, the generalization bound of LapWR is derived based on analyzing its Rademacher complexity, which suggests that our proposed algorithm is guaranteed to obtain satisfactory performance. By comparing LapWR with the existing representative SSL algorithms on various benchmark and real-world datasets, we experimentally found that LapWR performs robustly to outliers and is able to consistently achieve the top-level results.**

***Index Terms*—Generalization bound, half-quadratic (HQ) optimization, Nyström method, semisupervised learning (SSL), Welsch loss.**

Manuscript received July 28, 2018; revised September 5, 2019 and November 6, 2019; accepted November 7, 2019. This work was supported in part by the NSF of China under Grant 61602246, Grant 61802189, Grant 61973162, and Grant U1713208, in part by the NSF of Jiangsu Province under Grant BK20171430 and Grant BK20180464, in part by the Fundamental Research Funds for the Central Universities under Grant 30918011319 and Grant 30918014107, in part by the Open Project of State Key Laboratory of Integrated Services Networks, Xidian University under Grant ISN19-03, in part by the "Summit of the Six Top Talents Program" under Grant DZXX-027, in part by the "Young Elite Scientists Sponsorship Program" by Jiangsu Province, in part by the "Young Elite Scientists Sponsorship Program" by CAST under Grant 2018QNRC001, in part by the Program for Changjiang Scholars, in part by the "111" Program under Grant AH92005, and in part by the Australian Research Council under Grant DE190101473. This article was recommended by Associate Editor Y. Jin. *(Corresponding author: Chen Gong.)*

J. Ke and C. Gong are with the PCA Lab, Key Laboratory of Intelligent Perception and Systems for High-Dimensional Information of Ministry of Education, Nanjing University of Science and Technology, Nanjing 210094, China, and also with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China (e-mail: csjke@njust.edu.cn; chen.gong@njust.edu.cn).

T. Liu and D. Tao are with the UBTECH Sydney Artificial Intelligence Centre, University of Sydney, Darlington, NSW 2008, Australia, and also with the School of Computer Science, Faculty of Engineering, University of Sydney, Darlington, NSW 2008, Australia (e-mail: tliang.liu@gmail.com; dacheng.tao@sydney.edu.au).

L. Zhao and J. Yang are with the PCA Lab, Key Laboratory of Intelligent Perception and Systems for High-Dimensional Information of Ministry of Education, Nanjing University of Science and Technology, Nanjing 210094, China, and also with the Jiangsu Key Laboratory of Image and Video Understanding for Social Security, School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: linzhao@njust.edu.cn; csjyang@njust.edu.cn).

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

Digital Object Identifier 10.1109/TCYB.2019.2953337

I. Introduction

In many machine learning and data mining applications, massive data can be easily collected due to the development of sensors or Internet. However, manually labeling them for model training is very expensive in terms of both time and labor cost. Therefore, it is often the case that only a small set of data are labeled while the vast majority of collected data are left unlabeled. In this case, the traditional fully supervised classification methods cannot be used due to the scarcity of the labeled data. To address this problem, semisupervised learning (SSL) was proposed [1] which has attracted increasing attention in the past few years. The SSL methods try to establish an accurate classifier by taking advantage of the supervision information carried by the limited labeled data as well as the distribution information revealed by the massive unlabeled data.

SSL is an active field, in which a large number of algorithms have been proposed so far. Among the existing SSL algorithms, the graph-based methods are commonly used and have attracted wide attention due to their good performance. These methods usually build a graph whose nodes correspond to data points and the edges connecting them encode their similarities. The labels of the unlabeled data are then learned from the graph in the way that the nearby data points in the graph will have similar labels. In general, researchers hypothesize a low-dimensional manifold structure along which labels are assumed to vary smoothly. To discover the manifold structure of the data, Zhu *et al.* [2] used the graph Laplacian to approximate the Laplace–Beltrami operator defined on the manifold, and proposed an algorithm called Gaussian field and harmonic functions (GFHF). However, this method is transductive and cannot generalize to the unseen test data. Therefore, Belkin *et al.* [3] developed the manifold regularization and proposed two variants, including Laplacian regularized least squares (LapRLS) and Laplacian support vector machines (LapSVM). Different

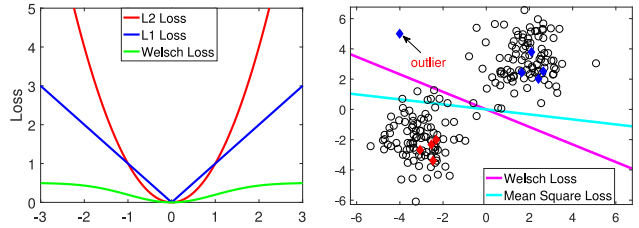


Fig. 1. Motivation of our method: (a) compares our adopted Welsch loss with the existing ℓ_2 loss and ℓ_1 loss; (b) compares the generated decision boundaries of our proposed LapWR with the Welsch loss (magenta line) and the LapRLS method with the ℓ_2 loss (cyan line). The red and blue diamonds represent the labeled positive examples and negative examples, respectively, among which one of the negative examples forms an outlier. The black circles are unlabeled examples.

from the above methods that only consider pairwise label smoothness, Wang *et al.* [4] assumed that every data point in the graph can be linearly reconstructed by its neighbors and proposed the linear neighborhood propagation (LNP) algorithm. Moreover, Yu *et al.* [5] proposed a feature selection-based SSL algorithm to handle the classification task with high-dimensional data. Afterward, they generated an auxiliary training set and adopt a multiobjective subspace selection process to obtain the reliable classifier [6]. Recently, different kinds of manifold regularization-based semisupervised algorithms have been proposed, such as Hessian regularization [7], p-Laplacian regularization [8], and hypergraph-based regularization [9]. Other typical graph-based algorithms include local and global consistency [10], Fick's law-assisted propagation [11], probabilistic pointwise smoothness [12], flexible semisupervised embedding [13], and optimal graph embedded semisupervised feature selection [14].

Besides the above manifold assumption, cluster assumption is also a major assumption widely adopted by many SSL algorithms. Cluster assumption assumes that the classes are well separated, such that the decision boundary falls into the low-density area in the feature space. The semisupervised support vector machine (S3VM) [15] is one of the most representative algorithms which focus on approaching an optimal low-density separator. Based on S3VM, Li and Zhou [16] proposed the safe S3VM (S4VM) model to exploit the candidate's low-density separators simultaneously to reduce the risk of identifying a poor separator with unlabeled data. Furthermore, Wang *et al.* [17] extended the cluster assumption of examples to class membership and developed a novel SSL methodology by introducing the membership vector. Recently, Sakai *et al.* [18], [19] studied SSL from the view of positive and unlabeled learning and proposed to conduct SSL by directly optimizing the area under curve (AUC) metric.

However, the above methods share a common drawback that they are not robust to the outliers. Taking LapRLS as an example, due to the adopted ℓ_2 loss on the labeled data which amplifies the negative impact of outliers, the generated decision is quite biased and does not fall into the margin between data clusters [see the cyan line in Fig. 1(b)]. To mitigate this drawback, various methods have been proposed to enhance the robustness of SSL. For example, Nie *et al.* [20] and Luo *et al.* [21], respectively, devised the adaptive elastic embedding and discriminative least-squares regression to alleviate the sensitivity of ℓ_2 loss to outliers. After that, Liu *et al.* [22] adapted the elastic constraint to semisupervised label propagation and used the loss without the square to make the learned model robust to the outliers. Apart from this, Gong *et al.* [23] used the degree of each node on the graph to determine the examples ambiguity and designed a novel smoothness regularizer based on the deformed graph Laplacian. Later, they proposed to deploy curriculum learning for semisupervised label propagation, so that the simple examples with definite labels are learned ahead of the difficult ambiguous examples [24], [25]. Their method contains teaching-to-learn step and learning-to-teach step, and these two steps interact in each iteration so that all unlabeled examples are utilized via an ordered sequence.

In this article, we focus on the loss function and present the Welsch loss to solve the robustness problem incurred by the outliers. By combining the proposed Welsch regularization and also the existing Laplacian regularization, our method is thus dubbed Laplacian Welsch regularization (LapWR). The main motivation of LapWR is that the conventional loss functions, such as ℓ_2 loss, are not robust to the outliers as the unboundedness of the convex loss functions would cause outliers to have large loss values. As a result, the decision boundary may deviate severely from the optimal one, which leads to the poor performance. From Fig. 1(a), we learn that the adopted Welsch loss is a bounded, smooth, and nonconvex loss and, thus, it is more robust than the commonly used ℓ_2 loss and ℓ_1 loss. To validate this, we generate two Gaussian data clusters with an outlier of negative class as indicated in Fig. 1(b). We intuitively show how this outlier influences the decision boundaries of the LapRLS method with the ℓ_2 loss and our method with Welsch loss. It can be clearly found that the decision boundary of LapRLS (i.e., the cyan line) is seriously influenced by the outlier, while that of our LapWR (i.e., the magenta line) successfully resists the perturbation of outlier and travels through the margin between two classes.

Moreover, we establish our model in the reproducing kernel Hilbert space (RKHS) to obtain a nonlinear decision boundary. To handle the nonconvexity brought by the Welsch loss, we adopt the half-quadratic (HQ) optimization [26] technique and further prove that an ideal closed-form solution can be found in each iteration, which means that our LapWR method can be easily implemented. Furthermore, to reduce the computational complexity of our method, we propose an extended accelerated model by applying the Nyström method [27] to speed up the original model. The Nyström method is probably one of the most well-studied and successful methods that have been used to scale up the kernel methods. Besides, we also present the generalization bound based on the derived Rademacher complexity. From the generalization bound, we observe that the generalization error will gradually decrease with the increase of training examples. Therefore, our proposed LapWR is guaranteed to obtain satisfactory performance. In the experiments, we compare our algorithm on both the synthetic and real-world

datasets with the state-of-the-art SSL algorithms, and the results confirm the effectiveness of our algorithm in the presence of data noises. Also, we show the convergence property and parametric stability of LapWR.

As mentioned above, in this article, we attempt to devise a loss function to handle the robustness problem incurred by the outliers. Actually, there are also some other robust loss functions developed so far that are upper bounded like our adopted Welsch loss, such as capped l_1 loss [28] and nonconvex squared loss [29]. However, they are inferior to our Welsch loss due to the induced optimization issues. Specifically, capped l_1 loss and nonconvex squared loss are nonsmooth, so the gradient that is critical for model optimization cannot be computed at some nondifferentiable points. Therefore, in this article, we deploy the Welsch loss to deal with the robustness problem incurred by the outliers in SSL. Moreover, the bisquare loss appeared in [30] can also be used here, and it leads to the comparable performance to the Welsch loss as their functional behaviors are quite similar.

The remainder of this article is as follows. In Section II, we describe the LapWR model and its optimization algorithm. Then, the accelerated model which relies on the Nyström method will be presented in Section III. Section IV derives the generalization bound based on the Rademacher complexity. Experimental results are shown in Section V. Finally, this article is summarized in Section VI.

II. Model Description

In this section, we first introduce some background knowledge of graph-based SSL, and then present the proposed LapWR and also its solution.

A. Graph-Based Semisupervised Learning

The SSL problem is mathematically defined as follows. Without loss of generality, we take the binary classification as an example. Given a dataset $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_l, \mathbf{x}_{l+1}, \ldots, \mathbf{x}_n\}$ of size n and a label set $\mathcal{Y} \in \{1, -1\}$, where the first l data examples $\mathbf{x}_i (i \leq l)$ in $\mathcal{X}$ are labeled as $y_i \in \{1, -1\}$ and the remaining u examples $\mathbf{x}_i (l+1 \leq i \leq n, n = l+u)$ are unlabeled with $y_i = 0$. We use $\mathcal{L} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_l, y_l)\}$ to denote the labeled set drawn from the joint distribution $\mathcal{P}$ defined on $\mathcal{X} \times \mathcal{Y}$, and $\mathcal{U} = \{\mathbf{x}_{l+1}, \mathbf{x}_{l+2}, \ldots, \mathbf{x}_{l+u}\}$ to represent the unlabeled set drawn from the unknown marginal distribution $\mathcal{P}_\mathcal{X}$ of $\mathcal{P}$. In the graph-based method, we model the whole dataset $\mathcal{X}$ as a graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$, where $\mathcal{V}$ is the set of nodes which is composed by the elements in $\mathcal{X}$ and $\mathcal{E}$ is the set of edges which records the relationship among all the nodes. $\mathbf{W}$ is the adjacency matrix of graph $\mathcal{G}$ which can be defined as $w_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/(2\sigma^2))$, where w_{ij} denotes the similarity between examples $\mathbf{x}_i$ and $\mathbf{x}_j$; the variance σ is a free parameter that should be manually tuned. Based on the adjacency matrix $\mathbf{W}$, the Laplacian matrix $\mathbf{L}$ of the graph $\mathcal{G}$ can be computed by $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{D}$ is an $n \times n$ diagonal matrix with its ith diagonal element being equal to the sum of the ith row of $\mathbf{W}$.

TABLE I
Important Mathematical Notations

Notation	Description
l	Number of labeled examples
$\mathcal{L}$	Labeled set
u	Number of unlabeled examples
$\mathcal{U}$	Unlabeled set
$\mathbf{x}_i$	The i^{th} example
y_i	Label of $\mathbf{x}_i$, $y_i \in \{\pm 1\}$
$\mathbf{W}$	Adjacency matrix
$\mathbf{L}$	Laplacian matrix
$\mathbf{D}$	Degree matrix
$\mathcal{G}$	Weighted similarity graph
$\mathbf{K}$	Kernel matrix
$V(\cdot)$	Loss function
$f(\mathbf{x})$	Decision function

The existing Laplacian regularized graph-based SSL models are usually established in the RKHS, which has the formation

$$Q(f) = \sum\nolimits_{i=1}^{l} V(y_i, f(\mathbf{x}_i)) + \lambda \|f\|_\mathcal{H}^2 + \mu \mathbf{f}^\top \mathbf{L} \mathbf{f} \tag{1}$$

where the vector $\mathbf{f} = (f_1, f_2, \ldots, f_n)^\top$ records the determined soft labels of all n examples. The first term of $Q(f)$ is the fitting term which consists of the loss function on the labeled set $\mathcal{L}$. As mentioned before, the existing methods usually use the ℓ_1 loss [31], ℓ_2 loss [3], or hinge loss [15] for $V(y_i, f(\mathbf{x}_i))$ which are not sufficiently robust. The second term is the regularization term to prevent overfitting. The third term is the smoothness term which requires that similar examples in the feature space $\mathcal{X}$ also obtain similar labels in $\mathcal{Y}$ space. In (1), λ and μ are non-negative tradeoff parameters governing the relative weights of the two terms. By extending the conventional representer theorem to the semisupervised case, the closed-form solution of (1) can be easily found [3]. For convenience, we list the important notations which will be later used in this article in Table I.

B. Laplacian Welsch Regularization

As mentioned in Section I, the loss functions $V(\cdot)$ adopted by the existing SSL algorithms are often unbounded, and this will cause large loss value when outliers appear. Therefore, the SSL algorithms with such nonrobust losses are very likely to be influenced by the outliers and produce the deviated decision boundary from the expected one (see Fig. 1). To make matters worse, SSL only harnesses very few labeled examples to train a classifier, so the negative impact of outliers can be significantly amplified due to the existing nonrobust losses. Therefore, in this article, we focus on designing a robust loss function to make the classifier stable to the outliers in the labeled set.

To suppress the adverse impact caused by the outliers, we propose to incorporate the Welsch loss to the framework of SSL. The Welsch loss is a bounded, smooth, and nonconvex loss which is very robust to the outliers. It is defined as

$$V(z) = \frac{c^2}{2}\left[1 - \exp\left(-\frac{z^2}{2c^2}\right)\right] \tag{2}$$

where c is a tuning parameter controlling the degree of penalty to the outliers. Fig. 2 shows the function curve of the Welsch

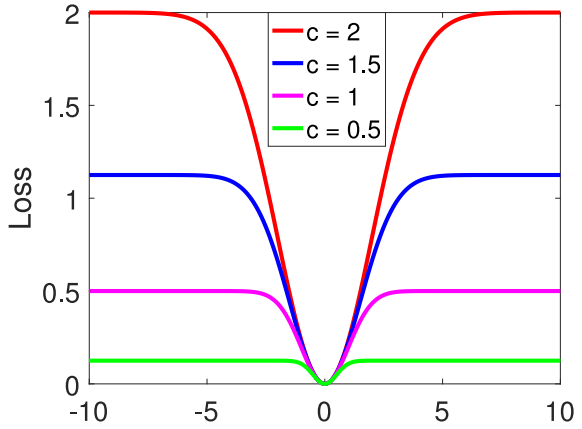


Fig. 2. Welsch loss with different selections of parameter c.

loss $V(z)$ under different values of c which changes from 0.5 to 2. We see that the upper bound of the Welsch loss increases and converges slowly when c gradually increases. By driving z to infinity, it can be easily found that the upper bound of the Welsch loss is $c^2/2$ from (2), which means that the influences of abnormal examples are capped during model training. Consequently, the Welsch loss is able to resist the disturbances of outliers.

Due to the robustness of the Welsch loss, in this article, we incorporate the loss (2) to the general framework of SSL in (1), and thus the proposed method called LapWR. Since our model is established in RKHS, there is a unique positive semidefinite kernel on $\mathcal{X} \times \mathcal{X}$ [32]. Therefore, by adopting the representer theorem, the decision function can be expressed as $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \mathbf{K}(\mathbf{x}, \mathbf{x}_i)$, where $\mathbf{K}$ is the $n \times n$ Gram matrix over labeled and unlabeled examples and α_i is the coefficient. Consequently, our LapWR model is formally represented as

$$\min_{\boldsymbol{\alpha}} \quad \sum_{i=1}^{l} \frac{c^2}{2}\left[1 - \exp\left(-\frac{(y_i - f(\mathbf{x}_i))^2}{2c^2}\right)\right] + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 + \frac{\mu}{2(l+u)^2}\mathbf{f}^\top \mathbf{L}\mathbf{f} \tag{3}$$

where l is the number of labeled examples, u is the number of unlabeled examples, and λ and μ are the non-negative regularization parameters. By denoting the parameter vector as $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)^\top$, our goal is to find a suitable $\boldsymbol{\alpha}$ to minimize the objective of (3). However, due to the introduction of the Welsch loss, the optimization of nonconvex (3) becomes very difficult. There are several well-known algorithms for dealing with nonconvex problem, such as iterative reweighted least squares (IRLS) [33] and concave–convex procedure (CCCP) [34]. Although these above algorithms can achieve reasonable solutions, they are not the optimal solutions for solving LapWR. Specifically, they will result in a slow convergence rate due to that each iteration also requires iteratively solving an optimization subproblem. Consequently, in this article, we simply follow [26] and use the HQ programming method to handle the nonconvexity problem caused by the Welsch loss. For our problem, HQ programming has a closed-form solution in every iteration, so HQ can be implemented more efficiently than IRLS and CCCP. We will show the detailed optimization procedure of HQ in the next section.

C. HQ Optimization for LapWR

Before we use the HQ optimization algorithm to optimize LapWR, we first rewrite (3) as

$$\max_{\boldsymbol{\alpha}} G_1(\boldsymbol{\alpha}) \tag{4}$$

where

$$G_1(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \exp\left(-\frac{(y_i - f(\mathbf{x}_i))^2}{2c^2}\right) - \frac{\lambda}{c^2}\|f\|_{\mathcal{H}}^2 - \frac{\mu}{c^2(l+u)^2}\mathbf{f}^\top \mathbf{L}\mathbf{f}. \tag{5}$$

To facilitate the following derivations, we then define a convex function $g(v) = -v\log(-v) + v$, where $v < 0$. Based on the conjugate function theory [35], we have

$$\exp\left(-\frac{(y - f(a))^2}{2c^2}\right) = \sup_{v<0}\left\{v\frac{(y - f(a))^2}{2c^2} - g(v)\right\} \tag{6}$$

which the supremum is achieved at

$$v = -\exp\left(-\frac{(y - f(a))^2}{2c^2}\right). \tag{7}$$

With (6), we can rewrite $G_1(\boldsymbol{\alpha})$ in (4) as

$$\begin{aligned} G_1(\boldsymbol{\alpha}) &\overset{1}{=} \sum_{i=1}^{l} \sup_{v_i<0}\left\{v_i\frac{(y_i - f(\mathbf{x}_i))^2}{2c^2} - g(v_i)\right\} - Z(f) \\ &\overset{2}{=} \sup_{\mathbf{v}<0}\left\{\sum_{i=1}^{l}\left(v_i\frac{(y_i - f(\mathbf{x}_i))^2}{2c^2} - g(v_i)\right)\right\} - Z(f) \\ &\overset{3}{=} \sup_{\mathbf{v}<0}\left\{\sum_{i=1}^{l}\left(v_i\frac{(y_i - f(\mathbf{x}_i))^2}{2c^2} - g(v_i)\right) - Z(f)\right\} \end{aligned} \tag{8}$$

where $Z(f) = (\lambda/c^2)\|f\|_{\mathcal{H}}^2 + (\mu/c^2(l+u)^2)\mathbf{f}^\top\mathbf{L}\mathbf{f}$; $\mathbf{v} = (v_1, \ldots, v_l)^\top \in \mathbb{R}^l$ with $v_i < 0$ for $i = 1, 2, \ldots, l$. The second equation above holds due to the fact that the v_is $(i = 1, \ldots, l)$ in the first term are independent to each other, and the third equation comes from the fact that $Z(f)$ is a constant that is irrelevant to v_i. By using (8), we further derive (4) as

$$\max_{\boldsymbol{\alpha}, \mathbf{v}<0} G_2(\boldsymbol{\alpha}, \mathbf{v}) \tag{9}$$

where

$$G_2(\boldsymbol{\alpha}, \mathbf{v}) = \sum_{i=1}^{l}\left[v_i\frac{(y_i - f(\mathbf{x}_i))^2}{2c^2} - g(v_i)\right] - \frac{\lambda}{c^2}\|f\|_{\mathcal{H}}^2 - \frac{\mu}{c^2(l+u)^2}\mathbf{f}^\top \mathbf{L}\mathbf{f}. \tag{10}$$

Now we can use the HQ optimization algorithm to optimize (9). Note that there are two variables to be optimized in (9), so we may alternatively optimize one of $\boldsymbol{\alpha}$ and $\mathbf{v}$ while keeping the other one unchanged. Suppose that we already have $\boldsymbol{\alpha}^s$, where the superscript s denotes the result of

Algorithm 1 HQ Optimization Algorithm for Solving (9)

Input: The kernel matrix $\mathbf{K}$; the free parameters c, λ and μ; the label vector $\mathbf{y}$; and the maximum iteration number S.
Output: $\boldsymbol{\alpha}$ in (16)

```
Set s = 0, ψ = 10^{-6} and initialize v^s;
while s < S do
    Construct Ω from v^s, Ω = diag(−v^s, 0);
    Obtain α^{s+1} by solving (16);
    Obtain v^{s+1} by solving (12);
    Check the convergence condition:
        ‖α^s − α^{s+1}‖_2 < ψ.
    Set s = s + 1;
end while
return α = α^s.
```

the sth iteration, then the optimization problem with respect to $\mathbf{v}$ becomes

$$\max_{\mathbf{v}^s<0} \sum_{i=1}^{l} \left[v_i^s \frac{(y_i - f^s(\mathbf{x}_i))^2}{2c^2} - g\big(v_i^s\big) \right]. \tag{11}$$

According to (6) and (7), the analytical solution of (11) is

$$v_i^s = -\exp\left(-\frac{(y_i - f^s(\mathbf{x}_i))^2}{2c^2} \right). \tag{12}$$

Second, after obtaining $\mathbf{v}^s$, we can obtain $\boldsymbol{\alpha}^{s+1}$ by solving the following problem:

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{l} \left[v_i^s \frac{(y_i - f^s(\mathbf{x}_i))^2}{2c^2} \right] - \frac{\lambda}{c^2} \|f\|_{\mathcal{H}}^2 - \frac{\mu}{c^2(l+u)^2} \mathbf{f}^\top \mathbf{L} \mathbf{f}. \tag{13}$$

Equation (13) can be rewritten in a compact matrix formation as follows:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2c^2} (\mathbf{y} - \mathbf{J}\mathbf{K}\boldsymbol{\alpha})^\top \boldsymbol{\Omega} (\mathbf{y} - \mathbf{J}\mathbf{K}\boldsymbol{\alpha}) \\ & + \frac{\lambda}{c^2} \boldsymbol{\alpha}^\top \mathbf{K}\boldsymbol{\alpha} + \frac{\mu}{c^2(l+u)^2} \boldsymbol{\alpha}^\top \mathbf{K}\mathbf{L}\mathbf{K}\boldsymbol{\alpha} \end{aligned} \tag{14}$$

where $\mathbf{y}$ is an n-dimensional label vector given by $\mathbf{y} = [y_1, \ldots, y_l, 0, \ldots, 0]^\top$; $\mathbf{J}$ is an $n \times n$ diagonal matrix given by $\mathbf{J} = \text{diag}(1, \ldots, 1, 0, \ldots, 0)$ with the first l diagonal entries being 1 and the remaining u elements being 0; and $\boldsymbol{\Omega}$ is also an $n \times n$ diagonal matrix given by $\boldsymbol{\Omega} = \text{diag}(-\mathbf{v}^s, \mathbf{0})$, where $\mathbf{0}$ is an all-zero vector.

By computing the derivative of (14) to $\boldsymbol{\alpha}$ and setting the result to zero, we obtain

$$\boldsymbol{\Omega}(\mathbf{y} - \mathbf{J}\mathbf{K}\boldsymbol{\alpha})(-\mathbf{J}\mathbf{K}) + 2\lambda \mathbf{K}\boldsymbol{\alpha} + \frac{2\mu}{(l+u)^2} \mathbf{K}\mathbf{L}\mathbf{K}\boldsymbol{\alpha} = \mathbf{0} \tag{15}$$

which leads to the following solution:

$$\boldsymbol{\alpha} = \left[\boldsymbol{\Omega}\mathbf{J}\mathbf{K} + 2\lambda \mathbf{I} + \frac{2\mu}{(l+u)^2} \mathbf{L}\mathbf{K} \right]^{-1} \boldsymbol{\Omega}\mathbf{y}. \tag{16}$$

Above subproblems regarding $\boldsymbol{\alpha}$ and $\mathbf{v}$ iterate and the solution of (9) can be finally obtained. Algorithm 1 summarizes the entire optimization procedure, in which $\mathbf{v}^0$ is initialized by setting it to -1. Also, we set the convergence condition as $\|\boldsymbol{\alpha}^s - \boldsymbol{\alpha}^{s+1}\|_2 < \psi$, where $\psi = 10^{-6}$. Note that in our HQ optimization, every subproblem has a closed-form solution, so the model (3) can be efficiently solved.

Although our model is derived for binary classification, it can be easily extended to multiclass situations by using the one-vs-the-rest strategy.

III. Model Acceleration

Due to the matrix inversion in (16), the computational complexity of direct implementation of LapWR is as high as $\mathcal{O}(Sn^3)$, which indicates that LapWR is computationally expensive. In this section, we propose to use the Nyström approximation method [27] to reduce the computational complexity of LapWR.

It is obvious that $\boldsymbol{\alpha}$ in (16) cannot be easily computed when the number of examples n increases. In this case, we try to reduce the dimension of kernel matrix $\mathbf{K}$ which is actually the Gram matrix over labeled and unlabeled examples. To this end, we try to find a low-rank approximation of $\mathbf{K}$ with rank m to replace the full-rank $\mathbf{K}$ in (16). It should be noted that the value of m should be settled carefully in practical use as it controls the tradeoff between efficiency and accuracy for matrix approximation. The Nyström method [27] is a popular low-rank approximation method which can help to reduce the computational complexity of LapWR. As mentioned before, we focus on generating an approximation of $\mathbf{K}$ (i.e., $\tilde{\mathbf{K}}$) based on a sample of $m \ll n$ of its columns. First, let $\mathbf{C}$ denote the $n \times m$ matrix formed by these columns and $\mathbf{Q}$ denote the $m \times m$ matrix consisted of the intersection of these m columns with the corresponding m rows of $\mathbf{K}$. Without loss of generality, the columns and rows of $\mathbf{K}$ can be rearranged based on this sampling so that $\mathbf{K}$ and $\mathbf{C}$ are written as

$$\mathbf{C} = \begin{bmatrix} \mathbf{Q} \\ \mathbf{E} \end{bmatrix} \quad \text{and} \quad \mathbf{K} = \begin{bmatrix} \mathbf{Q} & \mathbf{E}^\top \\ \mathbf{E} & \mathbf{F} \end{bmatrix} \tag{17}$$

where $\mathbf{E} \in \mathbb{R}^{(n-m) \times m}$ represents the $n - m$ columns with the corresponding m rows of $\mathbf{K}$; and $\mathbf{F} \in \mathbb{R}^{(n-m) \times (n-m)}$ represents the $n - m$ columns with the corresponding $n - m$ rows of $\mathbf{K}$. In this way, we divide $\mathbf{K}$ into four parts which further compose the matrix $\mathbf{C}$. Second, the Nyström method utilizes the above $\mathbf{Q}$ and $\mathbf{C}$ to approximate $\mathbf{K}$. By uniform sampling, a set of columns of K, the Nyström method generates a rank-r approximation $\tilde{\mathbf{K}}$ of $\mathbf{K}$ for $r < n$ defined by

$$\tilde{\mathbf{K}} = \mathbf{C}\mathbf{Q}_r^\dagger \mathbf{C}^\top \tag{18}$$

where $\mathbf{Q}_r$ is the optimal rank-r approximation of the $m \times m$ inner matrix $\mathbf{Q}$, and $\mathbf{Q}_r^\dagger$ represents the pseudo inverse of $\mathbf{Q}_r$. Now we obtain the low-rank approximation of $\mathbf{K}$ which can be decomposed as $\mathbf{C}\mathbf{Q}_r^\dagger \mathbf{C}^\top$.

In order to apply the Nyström method to LapWR, we first rewrite (15) as

$$\left(2\lambda \mathbf{I} + \left(\boldsymbol{\Omega}\mathbf{J} + \frac{2\mu}{(l+u)^2} \mathbf{L} \right) \mathbf{K} \right) \boldsymbol{\alpha} = \boldsymbol{\Omega}\mathbf{y}. \tag{19}$$

To facilitate the following derivations, we define $\mathbf{A} = \boldsymbol{\Omega}\mathbf{J} + 2\mu/(l+u)^2 \mathbf{L}$, then (19) is equal to

$$\left(2\lambda \mathbf{A}^{-1} + \mathbf{K} \right) \boldsymbol{\alpha} = \mathbf{A}^{-1} \boldsymbol{\Omega}\mathbf{y}. \tag{20}$$

Now we use the Nyström method to reduce the dimension of $\mathbf{K}$ by substituting $\tilde{\mathbf{K}}$ for $\mathbf{K}$ in the above equation, which arrives at

$$\left(2\lambda\mathbf{A}^{-1}+\mathbf{C}\mathbf{Q}_r^{\dagger}\mathbf{C}^{\top}\right)\boldsymbol{\alpha}=\mathbf{A}^{-1}\boldsymbol{\Omega}\mathbf{y}. \tag{21}$$

After decomposing $\mathbf{K}$, we try to compute the inverse of $2\lambda\mathbf{A}^{-1}+\mathbf{C}\mathbf{Q}_r^{\dagger}\mathbf{C}^{\top}$, which reminds us to use the Sherman–Morrison–Woodbury formula [36] that is formulated as

$$(\mathbf{P}+\mathbf{U}\mathbf{M}\mathbf{V})^{-1}=\mathbf{P}^{-1}-\mathbf{P}^{-1}\mathbf{U}\left(\mathbf{M}^{-1}+\mathbf{V}\mathbf{P}^{-1}\mathbf{U}\right)^{-1}\mathbf{V}\mathbf{P}^{-1} \tag{22}$$

where $\mathbf{P}\in\mathbb{R}^{n\times n}$, $\mathbf{U}\in\mathbb{R}^{n\times m}$, $\mathbf{M}\in\mathbb{R}^{m\times m}$, and $\mathbf{V}\in\mathbb{R}^{m\times n}$. In this case, we only need to compute the inverse of a small $m\times m$ matrix with $m\ll n$, making the computational cost of LapWR acceptable. By applying (22) to $2\lambda\mathbf{A}^{-1}+\mathbf{C}\mathbf{Q}_r^{\dagger}\mathbf{C}^{\top}$, we obtain

$$\begin{aligned}&\left(2\lambda\mathbf{A}^{-1}+\mathbf{C}\mathbf{Q}_r^{\dagger}\mathbf{C}^{\top}\right)^{-1}\\&=\frac{1}{2\lambda}\mathbf{A}-\left(\frac{1}{2\lambda}\right)^2\mathbf{A}\mathbf{C}\left(\left(\mathbf{Q}_r^{\dagger}\right)^{-1}+\frac{1}{2\lambda}\mathbf{C}^{\top}\mathbf{A}\mathbf{C}\right)^{-1}\mathbf{C}^{\top}\mathbf{A}\end{aligned} \tag{23}$$

where $((\mathbf{Q}_r^{\dagger})^{-1}+(1/2\lambda)\mathbf{C}^{\top}\mathbf{A}\mathbf{C})\in\mathbb{R}^{m\times m}$. Note that the Nyström approximation of $\mathbf{K}$ only needs to be computed once and its computational complexity is $\mathcal{O}(rmn)$. Now, (16) can be rewritten as

$$\boldsymbol{\alpha}=\left[\frac{1}{2\lambda}\mathbf{I}-\left(\frac{1}{2\lambda}\right)^2\mathbf{A}\mathbf{C}\left(\left(\mathbf{Q}_r^{\dagger}\right)^{-1}+\frac{1}{2\lambda}\mathbf{C}^{\top}\mathbf{A}\mathbf{C}\right)^{-1}\mathbf{C}^{\top}\right]\boldsymbol{\Omega}\mathbf{y}. \tag{24}$$

To summarize, in the accelerated LapWR, we use (24) to replace (16) in Algorithm 1. Recall that when $m\ll n$, the computational complexity of accelerated LapWR is $\mathcal{O}(rmn+Sm^3)$ which is obviously smaller than the computational complexity of original LapWR which is $\mathcal{O}(Sn^3)$. Furthermore, several low-rank approximation methods which are variants or extensions of the adopted Nyström method have recently been proposed to obtain good decomposition results, such as [37] and [38], and they can also be used here to further reduce the computational complexity.

IV. Generlizaion Bound

In this section, we derive the generalization bound of the proposed LapWR based on its Rademacher complexity.

Definition 1 [39]: For a sample $\{x_1,\ldots,x_n\}$ generated by a distribution $\mathcal{D}$ and a real-valued function class $\mathcal{F}$ with domain $\mathcal{X}$, the empirical Rademacher complexity of $\mathcal{F}$ is defined as

$$\hat{R}_n(\mathcal{F})=\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\left|\frac{2}{n}\sum_{i=1}^{n}\sigma_i f(x_i)\right|\right] \tag{25}$$

where the expectation is taken over $\boldsymbol{\sigma}=(\sigma_1,\ldots,\sigma_n)^{\top}$ with $\sigma_i\in\{-1,+1\}$ $(i\in[1,n])$ being independent uniform random variables. The Rademacher random variables satisfy the probability $P\{\sigma_i=+1\}=P\{\sigma_i=-1\}=1/2$. Then, the Rademacher complexity of $\mathcal{F}$ is

$$R_n(\mathcal{F})=\mathbb{E}_{\mathbf{x}}\left[\hat{R}_n(\mathcal{F})\right]=\mathbb{E}_{\mathbf{x}\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\left|\frac{2}{n}\sum_{i=1}^{n}\sigma_i f(x_i)\right|\right]. \tag{26}$$

Based on Definition 1, the generalization bound for a function $f\in\mathcal{F}$ is given in the following theorem.

Theorem 1 [40]: Let $\mathcal{F}$ be a class of functions mapping from $\mathcal{Z}=\mathcal{X}\times\mathcal{Y}$ to $[0,1]$. Given n examples drawn independently from a distribution $\mathcal{D}$, then with probability $1-\delta$, every $f\in\mathcal{F}$ satisfies

$$err(f)\le\hat{err}(f)+\hat{R}_n(\mathcal{F})+3\sqrt{\frac{\ln(2/\delta)}{2n}} \tag{27}$$

where $err(f)$ represents the expected error, and $\hat{err}(f)$ denotes the empirical error of f.

This bound is quite general and applicable to various learning algorithms if an empirical Rademacher complexity $\hat{R}_n(\mathcal{F})$ of the function class $\mathcal{F}$ can be found. Meanwhile, it is easy to bound the empirical Rademacher complexity for kernelized algorithms by using the trace of the kernel matrix.

Theorem 2 [41]: If $\mathbf{K}:\mathcal{X}\times\mathcal{X}\mapsto\mathbb{R}$ is a kernel, and $\{x_1,\ldots,x_n\}$ is a sample from $\mathcal{X}$, then the empirical Rademacher complexity of the class $\mathcal{F}(B)$ with the bounded norm $\|f\|_{\mathcal{H}}\le B$ satisfies

$$\hat{R}_n(\mathcal{F}(B))\le\frac{2B}{n}\sqrt{\sum_{i=1}^{n}\mathbf{K}(x_i,x_i)}. \tag{28}$$

If $\mathbf{K}(x,x)\le T^2$ for all $x\in\mathcal{X}$ and $\mathbf{K}$ is a normalized kernel, we can rewrite (28) as

$$\hat{R}_n(\mathcal{F}(B))\le\frac{2B}{n}\sqrt{\sum_{i=1}^{n}\mathbf{K}(x_i,x_i)}\le 2B\sqrt{\frac{T^2}{n}}. \tag{29}$$

Based on the empirical Rademacher complexity and Theorem 2, it is easy to bound the generalization error of our LapWR. To be specific, we may first find the value of B in Theorem 2, and then combine (27) and (28) to derive the generalization bound of LapWR. The result is presented in the following theorem.

Theorem 3 (Generalization Bound): Let $err(f)$ and $\hat{err}(f)$ be the expected error and the empirical error of LapWR; l and u be the amounts of labeled examples and unlabeled examples, respectively. Suppose $n=l+u$ and $\mathbf{K}(x,x)\le T^2$, then for any $\delta>0$, with probability at least $1-\delta$, the generalization error of LapWR is

$$\begin{aligned}|err(f)-\hat{err}(f)|\le{}&2cT\sqrt{\frac{ln}{\lambda n^2+\mu\beta_1}\left[1-\exp\left(-\frac{1}{2c^2}\right)\right]}\\&+3\sqrt{\frac{\ln(2/\delta)}{2n}}.\end{aligned} \tag{30}$$

Proof: In order to use Theorem 2, we are going to find the value of B in (28) which is the upper bound of $\|f\|_{\mathcal{H}}^2$. To this end, recalling that the objective function of LapWR expressed in RKHS $\mathcal{H}$ is

$$\tilde{Q}(f)=\sum\nolimits_{i=1}^{l}V(y_i,f(\mathbf{x}_i))+\frac{\lambda}{2}\|f\|_{\mathcal{H}}^2+\frac{\mu}{2n^2}\mathbf{f}^{\top}\mathbf{L}\mathbf{f}. \tag{31}$$

Let $f^*=\operatorname{argmin}_{f\in\mathcal{H}}\tilde{Q}(f)$ be the solution of (31), then we have $\tilde{Q}(f^*)\le\tilde{Q}(\mathbf{0})$. Therefore, we further obtain

$$\frac{\lambda}{2}\|f^*\|_{\mathcal{H}}^2+\frac{\mu}{2n^2}\mathbf{f}^{\top}\mathbf{L}\mathbf{f}\le\tilde{Q}(\mathbf{0}). \tag{32}$$

Assume that the nonzero eigenvalues of $\mathbf{L}$ are $\beta_1 < \beta_2 < \cdots < \beta_d$, where d is the rank of $\mathbf{L}$, among which β_1 represents the minimum eigenvalue and β_d represents the maximum eigenvalue, then we obtain

$$\beta_1 \|f^*\|_{\mathcal{H}}^2 \leq \mathbf{f}^\top \mathbf{L}\mathbf{f} \leq \beta_d \|f^*\|_{\mathcal{H}}^2. \tag{33}$$

It is obvious that

$$\begin{aligned}\left(\frac{\lambda}{2} + \frac{\mu\beta_1}{2n^2}\right)\|f^*\|_{\mathcal{H}}^2 &\leq \frac{\lambda}{2}\|f^*\|_{\mathcal{H}}^2 + \frac{\mu}{2n^2}\mathbf{f}^\top \mathbf{L}\mathbf{f} \\ &\leq \left(\frac{\lambda}{2} + \frac{\mu\beta_d}{2n^2}\right)\|f^*\|_{\mathcal{H}}^2. \end{aligned} \tag{34}$$

By combining (32) and (34), we arrive at

$$\left(\frac{\lambda}{2} + \frac{\mu\beta_1}{2n^2}\right)\|f^*\|_{\mathcal{H}}^2 \leq \tilde{Q}(\mathbf{0}). \tag{35}$$

Therefore, we can restrict the search for f^* to a ball in $\mathcal{H}$ of radius $g = \sqrt{\tilde{Q}(\mathbf{0})/(\lambda/2 + \mu\beta_1/2n^2)}$. Let $\mathcal{H}_g := \{f \in \mathcal{H} : \|f\|_{\mathcal{H}} \leq g\}$ denote the ball of radius g in RKHS $\mathcal{H}$, then according to Theorem 1, the generalization bound of LapWR can be written as

$$|err(f) - \hat{err}(f)| \leq \hat{R}_n(\mathcal{H}_g) + 3\sqrt{\frac{\ln(2/\delta)}{2n}} \tag{36}$$

where

$$\hat{R}_n(\mathcal{H}_g) \leq 2g\sqrt{\frac{T^2}{n}}. \tag{37}$$

Suppose we set the parameter vector $\boldsymbol{\alpha} = (0, \ldots, 0)^\top$ which makes the last two terms in (31) equal zero, then we can have $\tilde{Q}(\mathbf{0}) = \sum_{i=1}^{l} V(y_i, f(\mathbf{0})) = lc^2/2[1 - \exp(-1/2c^2)]$. By further considering (35) we obtain

$$\left(\frac{\lambda}{2} + \frac{\mu\beta_1}{2n^2}\right)\|f^*\|_{\mathcal{H}}^2 \leq \frac{lc^2}{2}\left[1 - \exp\left(-\frac{1}{2c^2}\right)\right] \tag{38}$$

which reveals that

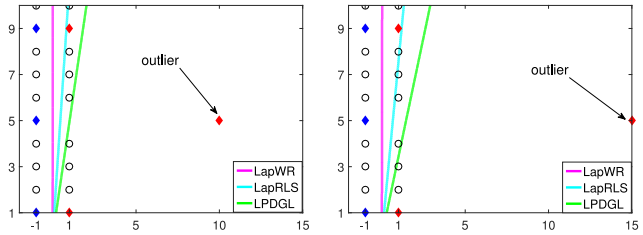
$$g = \sqrt{\frac{lc^2n^2}{\lambda n^2 + \mu\beta_1}\left[1 - \exp\left(-\frac{1}{2c^2}\right)\right]}. \tag{39}$$

By plugging (39) into (37), we find

$$\hat{R}_n(\mathcal{H}_g) \leq 2cT\sqrt{\frac{ln}{\lambda n^2 + \mu\beta_1}\left[1 - \exp\left(-\frac{1}{2c^2}\right)\right]}. \tag{40}$$

■

Finally, Theorem 3 can be proved by putting (40) into (36). Theorem 3 reveals that the LapWR has a profound generalizability with the convergence rate of order $\mathcal{O}(1/\sqrt{n})$, which means that the more training examples are adopted, the lower generalization bound of LapWR we will have. This is also consistent with our general understanding.



(a) (b)

Fig. 3. Linear decision boundaries generated by LapRLS, LPDGL, and LapWR on the synthetic *DoubleLine* dataset. The red and blue diamonds represent the labeled positive examples and negative examples, respectively, and the hollow circles denote unlabeled examples. The lines with x-coordinates -1 and 1 correspond to negative class and positive class separately. In (a), an outlier positive example is located at (10, 5); and in (b), this outlier is moved to (15, 5).

V. Experiments

In this section, we first validate the proposed LapWR on an artificial toy dataset, and then compare LapWR with the state-of-the-art SSL algorithms on some real-world collections. Finally, we investigate the convergence property and the parametric sensitivity of our LapWR method. Several popular SSL algorithms serve as baselines for comparison, including S3VM [15], S4VM [16], LapRLS [3], LapSVM [3], label prediction via deformed graph Laplacian (LPDGL) [23], SSL with elastic embedding (SEE) [20], SSL based on PN and PU classification (PNU) [18], and semisupervised AUC optimization (SSAUC) [19].

A. Toy Data

To intuitively show the robustness of our LapWR with the Welsch loss, we generate an artificial dataset called "DoubleLine" contaminated by different outliers as illustrated in Fig. 3. In this figure, we see that the data points belonging to two vertical lines, respectively, constitute two different classes, where the points with x-coordinate 1 correspond to positive class while the points with x-coordinate -1 represent negative class. Among these data points, the red diamonds are labeled as positive examples and the blue diamonds are labeled as negative examples. Note that there are only three examples labeled in each class, while the remaining hollow circles represent the unlabeled data. Furthermore, in Fig. 3(a), we observe that a labeled positive example located at (10, 5) is far away from the normal positive data, and this point is treated as an outlier that may have a large influence on determining the decision boundary. In Fig. 3(b), we move this outlier to a farther place [i.e., (15, 5)], to see how it affects the classifier training.

Apart from presenting the decision boundary of our proposed LapWR, we also show the results of two existing SSL models (i.e., LapRLS [3] and LPDGL [23]) that utilize the ℓ_2-norm-based mean square loss. All the three methodologies employ the k-nearest neighbor (k-NN) graph with $k = 3$, and the linear kernel is adopted for illustration. For LapWR, we set the tradeoff parameters λ and μ, the normalizing constant parameter c, and the maximum iteration number S to 1000, 10, 1, and 5, respectively. As for the compared algorithms,

TABLE II
STATISTICS OF THE ADOPTED UCI DATASETS

	Dataset	#Examples	#Attributes	#Classes
D1	Pima	768	8	2
D2	Redwine	1599	11	3
D3	Whitewine	4898	11	3
D4	Waveform	5000	21	3
D5	Pendigits	7494	16	10

we, respectively, set γ_A and γ_I to 100 and 1 in LapRLS, and tune α, γ, and β to 1 in LPDGL as recommended by the authors. From Fig. 3(a), we observe that the decision boundaries of LapRLS and LPDGL are seriously influenced by the outlier (i.e., cyan line and green line). Furthermore, Fig. 3(b) shows that as the outlier goes further, the decision boundaries produced by these two methods get worse, which indicates that LapRLS and LPDGL are not robust to outlier. In contrast, the Welsch loss mentioned before has the upper bound which can suppress the negative influences caused by the outlier. Thanks to the adopted Welsch loss, the decision boundaries of LapWR [i.e., magenta line in Fig. 3(a) and (b)] correctly discriminate the positive and negative classes no matter how far the outlier is from its normal position. More notably, even though the outlier has been pulled to a farther place in Fig. 3(b), the decision boundary of LapWR is almost the same with that in Fig. 3(a), which confirms that LapWR is very robust to the outliers. Therefore, the argument that our method with a bounded loss is better than the existing methods with the unbounded losses has been empirically verified.

B. UCI Data

In this section, we choose five University of California Irvine (UCI) machine learning repository datasets [42], that is, *Pima* (D1), *Redwine* (D2), *Whitewine* (D3), *Waveform* (D4), and *Pendigits* (D5), to compare the performance of LapWR with other baselines. D1 comes from the Indian diabetes dataset consisted of 768 examples with 8 attributes. D2 and D3 are related to red and white wine quality evaluation. We divide the wine examples into three levels, including "good" (the score is above 5), "normal" (the score equals to 5), and "bad" (the score is below 5). As for D4, it contains 5000 examples with 21 attributes belonging to three kinds of waveforms (i.e., classes). D5 is a multiclass classification dataset which is pen-based recognition of handwritten digits with 16 integer attributes and 10 classes. The detailed information of these five datasets is summarized in Table II. From Table II, we see that our experiments not only involve the binary classification but also contain multiclass classification. In this article, we adopt the one-vs-the-rest strategy to deal with the multiclass problems.

For all the UCI datasets we use 70% of the examples as the training data and the remaining 30% as the test data. For each dataset, we consider three different cases in which 5%, 10%, and 15% of the training examples are labeled. To achieve a fair comparison, we randomly pick up a subset of the training examples as labeled, and the selected labeled examples are kept identical for all compared methods for each case. Then, we repeat the above process 10 times for each dataset and then calculate the average test accuracies of compared algorithms to measure their performances. For fair comparison, SEE, LapSVM, LapRLS, LPDGL, and LapWR are trained on the same k-NN graph for each of the datasets. In D2 and D3, we construct the 10-NN graphs, and we build 12-NN graphs for D1, D4, and D5. The parameters λ and μ in LapWR are set to 0.1 and 1, respectively, and the maximum iteration number is decided as $S = 5$. Because the parameter c determines the upper bound of the loss function in our LapWR, we fix $c = 1$ to get small loss values for the outliers in all experiments. The parameters γ_A and γ_I are set to 0.1 and 1 in LapRLS, while the same parameters are set to 0.1 and 0.15 in LapSVM in all the UCI datasets. In LPDGL, we optimally set α, γ, and β to 1 via cross-validation. As for the parameters of other baselines, we adopt the default values which are provided by the authors.

Table III shows the mean value with the standard deviation of ten independent runs of all methods on different datasets, which reveals that our LapWR can consistently obtain the best results when compared with other baselines. In D1, we observe that LPDGL achieves 69% accuracy with 15% labeled examples, while LapWR can obtain almost the same accuracy with 5% labeled examples. With 15% labeled examples, LapWR can further improve the accuracy to almost 73%. LapWR also produces a very encouraging performance in D2. Its test accuracies are 3 or 4 percent higher than those of the baselines no matter how many training data labeled. As for D3, LapWR also achieves a better performance than the baselines with very few labeled examples. LapWR can get approximately 70% accuracy while the second best algorithm just yields 66% accuracy when 15% training examples are labeled. Also, we notice that LapWR obtains a very small standard deviation while other baselines render relatively large ones, such as SEE, PNU and SSAUC. This means that the performance of our LapWR is stable on this dataset with respect to different selections of labeled examples. In D4 and D5, we observe that some of the baselines achieve very good performances, such as LPDGL, LapSVM, and SSAUC. However, the accuracies obtained by them can still be improved by the LapWR, which demonstrates the strength of our algorithm. Besides, since the paired t-test is a statistical tool to determine whether two sets of observations are essentially the same, we use the paired t-test with 90% confidence level to examine whether the accuracies output by LapWR are significantly higher than the baselines. From Table III, we can see that the performances of LapWR are significantly better than other algorithms in most situations. In very rare cases, LapWR achieves comparable performances with the competitive baseline algorithms.

C. Object Recognition

Object recognition has been widely studied as a traditional research area of computer vision because of the extensive practical demands. We apply the proposed LapWR to the object recognition problem. COIL20[1] is a popular object recognition dataset, which contains 1440 object images belonging to

[1] http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php

TABLE III
COMPARISON OF TEST ACCURACIES (MEAN±STD) OF VARIOUS METHODS ON UCI DATASETS. •/◦ INDICATES THAT LAPWR IS SIGNIFICANTLY BETTER/WORSE THAN THE CORRESPONDING METHOD (PAIRED t-TEST WITH 90% CONFIDENCE LEVEL). THE BEST RESULT FOR EACH DATASET IS MARKED IN RED

		LapWR	LapRLS [3]	LapSVM [3]	LPDGL [23]	SEE [20]	PNU [18]	SSAUC [19]	S3VM [15]	S4VM [16]
D1	5%	69.13±3.16	65.11±1.98 •	64.16±5.74 •	64.33±4.15 •	66.02±1.43 •	62.47±9.32	60.09±6.96 •	61.08±5.91 •	61.00±5.58 •
	10%	71.99±2.86	67.53±2.27 •	67.53±4.89 •	67.92±3.84 •	67.45±1.89 •	67.66±5.31 •	57.10±13.29 •	67.84±4.28 •	68.48±4.83 •
	15%	72.77±3.09	67.88±2.13 •	68.66±2.87 •	69.10±1.54 •	67.32±2.20 •	67.49±8.60	62.68±9.90 •	67.23±3.03 •	66.19±4.57 •
D2	5%	64.31±3.50	60.35±3.03 •	60.54±3.49 •	62.08±2.02 •	61.81±2.56 •	52.46±9.50 •	39.44±11.42 •	55.13±6.01 •	56.88±5.23 •
	10%	68.37±2.43	64.40±2.66 •	63.87±3.42 •	62.79±1.36 •	66.42±2.40 •	52.71±12.49 •	52.40±8.42 •	57.79±4.04 •	57.96±4.50 •
	15%	69.06±1.85	66.50±1.74 •	66.69±3.00 •	63.60±0.96 •	66.98±1.59 •	57.40±10.12 •	54.00±5.97 •	60.50±3.94 •	62.00±3.56 •
D3	5%	67.50±2.34	65.39±3.16	54.68±6.75 •	64.71±0.32 •	65.36±3.20	50.52±14.29 •	38.07±18.70 •	51.71±7.34 •	53.75±5.28 •
	10%	69.63±1.68	67.91±2.70	59.83±6.58 •	64.87±0.86 •	59.55±12.17•	49.93±14.32 •	56.82±12.10 •	55.33±7.22 •	58.54±5.72 •
	15%	70.47±0.74	66.14±3.58 •	56.35±6.09 •	64.97±0.63 •	56.11±14.80•	56.88±10.08 •	54.77±20.71 •	55.25±7.19 •	58.92±8.31 •
D4	5%	80.58±0.80	78.21±0.56 •	78.15±0.78 •	74.07±1.95 •	80.55±2.12	78.74± 4.86 •	44.71±16.27 •	78.80±1.03 •	80.13±1.09
	10%	82.07±0.97	80.12±1.09 •	78.41±0.94 •	76.13±0.79 •	79.72±1.17 •	80.96± 1.08 •	68.85±21.89 •	79.92±1.06 •	80.57±1.43 •
	15%	82.14±0.94	80.93±0.85 •	80.19±1.01 •	78.09±1.21 •	80.44±1.38 •	80.13± 1.56 •	80.97±3.66	79.69±1.59 •	80.02±1.54 •
D5	5%	94.61±0.91	92.29±1.31 •	93.56±1.46 •	94.04±1.18	75.57±2.09 •	90.09± 1.20 •	36.43±31.60 •	91.22±1.09 •	93.47±1.04 •
	10%	96.85±0.81	95.76±0.73 •	96.64±0.74	96.14±0.62 •	76.30±1.45 •	91.86± 1.98 •	57.01±32.66 •	94.00±0.95 •	95.62±0.91 •
	15%	97.56±0.30	96.31±0.31 •	97.32±0.31 •	96.80±0.42 •	75.85±1.65 •	92.01± 2.13 •	62.76±31.34 •	95.04±0.79 •	96.13±0.43 •

Fig. 4. Example images from the COIL20 dataset.

20 classes. Fig. 4 shows the example images of 20 classes from COIL20. The size of each image is 32×32 with 256 gray levels per pixel. Each image is represented by a 1024-D vector. Like the UCI datasets, we select 70% examples from the entire dataset to form the training data, and the remaining 30% examples are treated as the test data. Also, we consider three different labeling ratios, including 5%, 10%, and 15%. To make the accuracies reliable, we run all the investigated methods ten times with randomly selected labeled examples and compute the average accuracy with the standard deviation.

We build a 5-NN graph with $\lambda = 0.1$, and set $\mu = 10$, $c = 1$, and $S = 5$ for LapWR in COIL20. For fair comparison, we build the same graph for SEE, LapRLS, LapSVM, and LPDGL. In LapRLS and LapSVM, we set both γ_A and γ_I to 1 to obtain the best performance. The test accuracies of compared algorithms are reported in Table IV, in which the best performance under each percent is marked in red. Note that S4VM is not compared here as this algorithm cannot deal with the dataset with high dimensionality. It is observed that LapWR achieves very satisfactory results and significantly outperforms other algorithms.

TABLE IV
TEST ACCURACIES OF COMPARED METHODS ON THE COIL20 DATASET. •/◦ INDICATES THAT LAPWR IS SIGNIFICANTLY BETTER/WORSE THAN THE CORRESPONDING METHOD (PAIRED t-TEST WITH 90% CONFIDENCE LEVEL)

	5%	10%	15%
LapRLS [3]	81.52±1.08•	85.98±1.24•	87.90±1.85•
LapSVM [3]	81.57±1.18•	85.95±1.25•	87.88±1.80•
LPDGL [23]	76.74±3.01•	82.95±1.30•	85.76±1.46•
SEE [20]	63.64±1.81•	74.19±1.92•	75.50±2.94•
PNU [18]	65.26±3.00•	79.40±1.80•	82.74±2.00•
SSAUC [19]	64.60±1.86•	75.00±0.11•	78.93±1.39•
S3VM [15]	56.35±6.75•	78.57±1.67•	83.81±1.90•
S4VM [16]	—	—	—
LapWR	83.69±1.76	88.40±1.41	89.64±1.32

In particular, the proposed LapWR achieves very high accuracy with limited labeled examples, for example, 88%, when only 10% of the training examples are labeled, which is better than other algorithms when 15% labeled examples are available. With 15% labeled examples, LapWR can achieve almost 90% accuracy which is very impressive. Also, we use the paired t-test to statistically demonstrate such superiority of LapWR to other methods. As we can see, LapWR has better performance than all the baselines on this dataset.

D. Text Classification

Besides object recognition, text classification is also an important task that deserves academic study. Here, we utilize the proposed LapWR to classify the text examples from the Reuters Corpus Volume I (RCV1)[2] to verify the advantages of LapWR for tackling text data. RCV1 is a dataset recording the corpus of newswire stories which contains 9625 examples with 29 992 distinct words, and these textual examples are divided into four classes, such as "C15", "ECAT", "GCAT," and "MCAT".

A 9-NN graph is established to evaluate the performances of SEE, LapRLS, LapSVM, LPDGL, and LapWR. Other parameters in LapWR are $\lambda = 0.001$, $\mu = 1$, $c = 1$, and $S = 5$.

[2]http://www.daviddlewis.com/resources/testcollections/rcv1/

TABLE V
EXPERIMENTS ON RCV1 DATASET. •/◦ INDICATES THAT LAPWR IS SIGNIFICANTLY BETTER/WORSE THAN THE CORRESPONDING METHOD (PAIRED t-TEST WITH 90% CONFIDENCE LEVEL)

	5%	10%	15%
LapRLS [3]	77.22±1.22•	83.49±0.65•	86.69±0.63•
LapSVM [3]	79.26±2.43•	86.35±0.84	88.28±0.66
LPDGL [23]	60.15±0.15•	70.55±0.17•	71.69±0.16•
SEE [20]	—	—	—
PNU [18]	61.92±9.04•	66.81±8.30•	72.79±1.02•
SSAUC [19]	62.04±1.88•	65.04±7.59•	67.45±0.01•
S3VM [15]	72.45±1.61•	83.12±0.61•	87.36±0.77•
S4VM [16]	—	—	—
LapWR	82.83±0.65	87.91±0.61	89.88±0.64

The parameters γ_A and γ_I in LapRLS are set to 0.1 and 1, while in LapSVM they are set to 0.1 and 0.5 to obtain the optimal results. Because the RCV1 dataset is sparse, the singular problem appears in SEE, so this method is not compared here. Besides, S4VM cannot handle this dataset as the feature dimensionality is very high. The results of other methodologies are given in Table V, in which the best result under each labeling rate is marked in red. We observe that LPDGL and SSAUC generate very low accuracies on this dataset, of which the accuracies are around 70% and 65%, respectively. In contrast, LapRLS is slightly better than PNU. Among the baseline methods, LapSVM and S3VM achieve very high accuracies, however, they are still inferior to LapWR with the margin 1% in terms of test accuracy. The paired t-test also statistically confirms the superiority of LapWR to the compared baselines.

E. Image Classification

In this section, we apply LapWR to image classification problem. We choose the CIFAR-10[3] dataset to test the performance of LapWR for image classification. CIFAR-10 consists of 60 000 32×32 color images in ten classes, with 6000 images per class. Fig. 5 shows ten images which are randomly selected from ten classes in CIFAR-10. We use the output of the first fully connected layer of VGGNet-16 to extract the CNN features for each image, therefore, the dimensionality of a feature vector is 4096. For our experiment, we randomly choose 3500 images from each class as training images, and the remaining images as testing. Similar to the above experiments, we also study the test accuracies of all methods with different sizes of labeled sets.

We build a 15-NN graph for model comparison, and the key parameters in LapWR are $\lambda = 0.5$, $\mu = 500$, $c = 1$, and $S = 5$. In order to obtain the best performance, we set γ_A and γ_I to 0.1 and 1 in LapRLS. In LapSVM, we set these two parameters to 0.5 and 1. The results are presented in Table VI where the best performance has been marked as red. The SEE and S4VM are not compared as they are not scalable to this dataset. From Table VI, we can observe that LapRLS and LapSVM are generally the best methods among the baselines. In contrast, our LapWR can still obtain better results than them no matter how many examples are labeled. Furthermore, we conduct the t-test on the CIFAR-10 which shows that LapWR is significantly better than all the baselines.

[3]http://www.cs.toronto.edu/ kriz/cifar.html

Fig. 5. Random sample of images from the CIFAR-10 dataset. There are ten image categories in the dataset, and each row represents a category.

TABLE VI
RESULTS OF ACCURACY ON THE CIFAR-10 DATASET. •/◦ INDICATES THAT LAPWR IS SIGNIFICANTLY BETTER/WORSE THAN THE CORRESPONDING METHOD (PAIRED t-TEST WITH 90% CONFIDENCE LEVEL)

	5%	10%	15%
LapRLS [3]	80.78±0.06•	81.11±0.86•	84.73±0.23•
LapSVM [3]	79.41±0.36•	81.73±0.39•	83.04±0.28•
LPDGL [23]	74.85±0.41•	76.56±0.24•	76.72±0.44•
SEE [20]	—	—	—
PNU [18]	78.27±0.31•	78.78±0.21•	78.57±0.87•
SSAUC [19]	54.11±9.73•	57.77±3.06•	60.12±2.77•
S3VM [15]	72.90±0.11•	77.06±0.11•	78.29±0.23•
S4VM [16]	—	—	—
LapWR	83.94±0.21	85.46±0.33	86.38±0.06

F. Effectiveness of Accelerated Model

As we mentioned before, the computational complexity of direct implementation of LapWR is as high as $\mathcal{O}(Sn^3)$. Therefore, in Section III, we proposed the accelerated model which uses the Nyström approximation to reduce the computational complexity of LapWR. In this section, we compare the original model with accelerated model on RCV1 and CIFAR-10 to show the effectiveness of model acceleration. We repeat the process ten times under three different ratios of labeled examples to compare the accuracy and CPU time of these two settings. For the accelerated model, m is a key tuning parameter which indicates the rank of the approximation of the kernel matrix. We set m to 500, 700, and 1200 in RCV1 when 5%, 10%, and 15% examples are labeled. As for CIFAR-10, m is set to 4000, 5000, and 6000 under 5%, 10%, and 15% labeled data.

Table VII shows the average CPU time and accuracy of ten independent runs on the real-world datasets RCV1 and

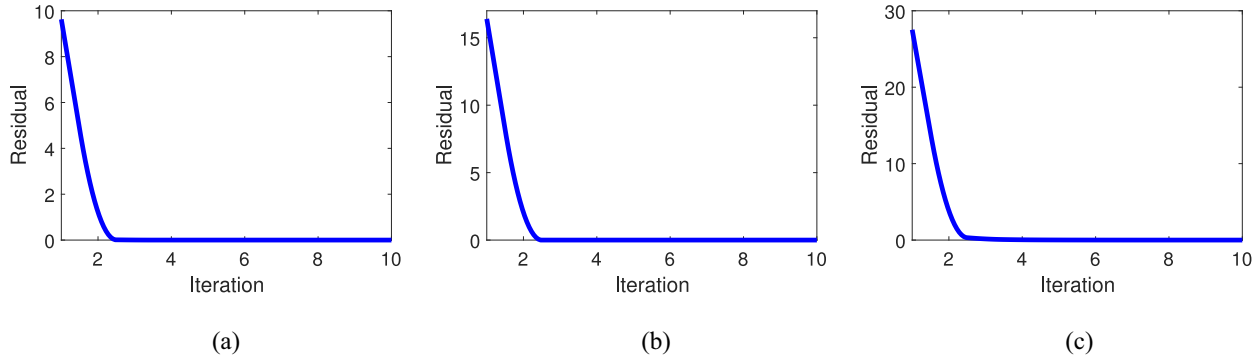


Fig. 6. Convergence behaviors of LapWR. (a)–(c) Convergence curves of LapWR on COIL20, RCV1 and CIFAR-10, respectively.

TABLE VII
AVERAGE ACCURACY AND CPU TIME OF LAPWR AND LAPWR* ON RCV1 AND CIFAR-10 DATASETS. LAPWR* REPRESENTS THE ACCELERATED MODEL

			LapWR	LapWR*
RCV1	Time (s)	5%	1027.70	816.02
		10%	1036.21	830.06
		15%	1028.56	855.11
	Accracy (%)	5%	82.83	82.19
		10%	87.91	87.65
		15%	89.88	89.60
CIFAR-10	Time (s)	5%	95651.16	30685.55
		10%	95725.96	46797.13
		15%	95745.38	52533.81
	Accracy (%)	5%	83.941	81.19
		10%	85.46	85.35
		15%	86.38	86.30

CIFAR-10, where LapWR* stands for the accelerated model and LapWR is the original model. From Table VII, we see that LapWR* can obtain comparable classification accuracies with LapWR on the two datasets, but the computational time is significantly less than LapWR. This indicates that our accelerated model can reduce the computational burden without sacrificing too much performance. In CIFAR-10, the acceleration effect of LapWR* is more obvious. The original LapWR needs more than 95 000 s for model training, while the accelerated LapWR* only consumes 30 685, 46 797, and 52 533 s when 5%, 10%, and 15% training examples are labeled. In other words, LapWR* saves 68%, 51%, and 45% of the training time when compared with LapWR under different ratios of labeled data.

G. Convergence Study

Because the Welsch loss is nonconvex, we cannot find a closed-form solution of LapWR. Therefore, we propose to use the HQ optimization to solve problem (3) in Section II. To this end, we develop Algorithm 1 to alternatively optimize the involved parameters. Therefore, in this section, we empirically study the convergence property of Algorithm 1.

In Fig. 6, we plot the convergence curves of our algorithm on the COIL20, RCV1, and CIFAR-10 datasets with 10% training data labeled, where the y-axis represents the residual $\|\boldsymbol{\alpha}^s - \boldsymbol{\alpha}^{s+1}\|_2$ and x-axis is the iteration times s. Similar to above experiments, we set ψ to 10^{-6} and stop the iteration process when the residual is less than ψ. The $\boldsymbol{\alpha}^0$ is initialized to all-zero vector $\mathbf{0}$, and all parameters settings remain the same as mentioned above. From Fig. 6, we can find that

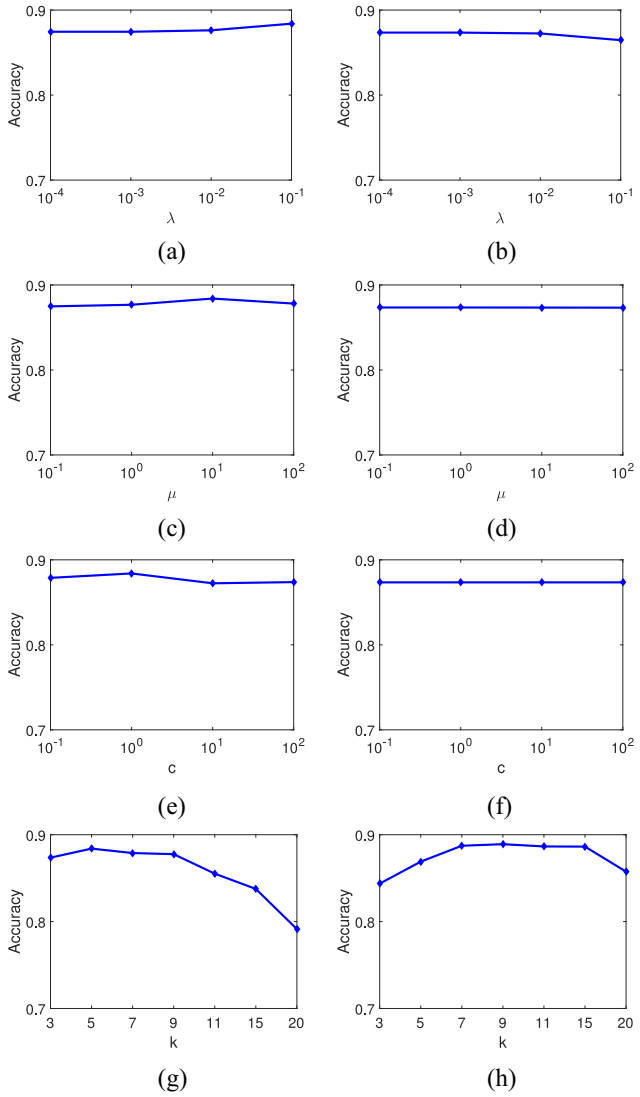


Fig. 7. Parametric sensitivity of LapWR. The first and second column, respectively, correspond to the COIL20 and RCV1 datasets. (a) and (b) Accuracy with respect to the change of λ when μ, c, and k are fixed. (c) and (d) Influence of μ to final accuracy when λ, c, and k are fixed. (e) and (f) Effects of c to the model accuracy when λ, μ, and k are fixed. (g) and (h) Impacts of k to the final accuracy when λ, μ, and c are fixed.

the residual gradually goes down and touches zero around the third iteration. Therefore, we conclude that LapWR can convergence to a stationary point rapidly. This also explains the reason that we set the maximum iteration number to 5 in all the above experiments.

H. Parametric Sensitivity

In this section, we investigate the parametric sensitivity of the tradeoff parameters λ and μ, the normalization parameter c, and the number of nearest neighbors k for graph construction. We examine the classification performance when one of them is changing while the others are fixed. The above two real-world datasets COIL20 and RCV1 are adopted here. Fig. 7 shows the model accuracies on these two datasets with 10% labeled examples. We tune λ from 10^{-4} to 10^{-1}, μ from 10^{-1} to 10^{2}, c from 10^{-1} to 10^{2}, and k from 3 to 20. From Fig. 7, we can observe that in COIL20, λ, μ, and c only have tiny effect on the accuracy. In RCV1, λ has a slight influence on the performance while μ and c have almost no influences. As for the parameter k, we find that the performance of our method will drop if k is too small or too large. Specifically, if k is too small, the graph may not be a connected graph and the isolated graph nodes will not receive label information. If k is too large, the graph will be very dense, which usually leads to worse performance than a sparse graph as reported in many prior works [43], [44]. In conclusion, we generally find that the parameters in LapWR can be easily tuned for practical implementations.

VI. Conclusion

This article proposed a novel SSL algorithm called LapWR, which is robust to the outliers in the labeled data. LapWR critically inherits a robust Welsch loss which upper bounds the large losses that are incurred by the outliers. To enhance the discriminability of LapWR, our model is established in RKHS so that a nonlinear classifier can be obtained. Because of the nonconvexity caused by the Welsch loss, we reformulate our model and use the HQ optimization algorithm to iteratively optimize the related model variables. Moreover, to reduce the computational complexity on large datasets, we propose an accelerated model based on the Nyström approximation method. We theoretically proved the Rademacher complexity and generalization bound of LapWR, which suggests that the test examples can be classified reliably and accurately. The experiments on the toy dataset, the UCI benchmark datasets, and the real-world datasets reveal that our LapWR is superior to the state-of-the-art SSL methods in terms of robustness and classification accuracy. The convergence property as well as the parametric stability of LapWR are also empirically verified. In the future, we plan to apply the Welsch loss to more deep learning-based SSL methods, such as [45] and [46] to improve their robustness. Besides, we may also consider extending our method to more related SSL scenarios, such as semisupervised dictionary learning, semisupervised dimensionality reduction, and semisupervised domain adaptation, as all these topics contain the data reconstruction process which may be heavily influenced by the outliers.

References

[1] O. Chapelle, B. Scholkopf, and A. Zien, "Semi-supervised learning," *IEEE Trans. Neural Netw.*, vol. 20, no. 3, p. 542, Mar. 2009.

[2] X. Zhu, Z. Ghahramani, and J. D. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. Int. Conf. Mach. Learn.*, 2003, pp. 912–919.

[3] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *J. Mach. Learn. Res.*, vol. 7, pp. 2399–2434, Nov. 2006.

[4] J. Wang, F. Wang, C. Zhang, H. C. Shen, and L. Quan, "Linear neighborhood propagation and its applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 9, pp. 1600–1615, Sep. 2009.

[5] Z. Yu *et al.*, "Adaptive semi-supervised classifier ensemble for high dimensional data classification," *IEEE Trans. Cybern.*, vol. 49, no. 2, pp. 366–379, Feb. 2019.

[6] Z. Yu *et al.*, "Multiobjective semisupervised classifier ensemble," *IEEE Trans. Cybern.*, vol. 49, no. 6, pp. 2280–2293, Jun. 2019.

[7] W. Liu and D. Tao, "Multiview hessian regularization for image annotation," *IEEE Trans. Image Process.*, vol. 22, no. 7, pp. 2676–2687, Jul. 2013.

[8] W. Liu, X. Ma, Y. Zhou, D. Tao, and J. Cheng, "*p*-Laplacian regularization for scene recognition," *IEEE Trans. Cybern.*, vol. 49, no. 8, pp. 2927–2940, Aug. 2019.

[9] X. Ma, W. Liu, S. Li, D. Tao, and Y. Zhou, "Hypergraph *p*-Laplacian regularization for remotely sensed image recognition," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 3, pp. 1585–1595, Mar. 2019.

[10] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Proc. Adv. Neural Inf. Process. Syst.*, 2003, pp. 321–328.

[11] C. Gong, D. Tao, K. Fu, and J. Yang, "Fick's law assisted propagation for semisupervised learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 9, pp. 2148–2162, Sep. 2015.

[12] Y. Fang, K. C.-C. Chang, and H. W. Lauw, "Graph-based semi-supervised learning: Realizing pointwise smoothness probabilistically," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 406–414.

[13] F. Dornaika and Y. El Traboulsi, "Learning flexible graph-based semi-supervised embedding," *IEEE Trans. Cybern.*, vol. 46, no. 1, pp. 206–218, Jan. 2016.

[14] M. Luo, X. Chang, L. Nie, Y. Yang, A. G. Hauptmann, and Q. Zheng, "An adaptive semisupervised feature analysis for video semantic recognition," *IEEE Trans. Cybern.*, vol. 48, no. 2, pp. 648–660, Feb. 2018.

[15] T. Joachims, "Transductive inference for text classification using support vector machines," in *Proc. Int. Conf. Mach. Learn.*, vol. 99, 1999, pp. 200–209.

[16] Y.-F. Li and Z.-H. Zhou, "Towards making unlabeled data never hurt," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 1, pp. 175–188, Jan. 2015.

[17] Y. Wang, S. Chen, and Z.-H. Zhou, "New semi-supervised classification method based on modified cluster assumption," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 5, pp. 689–702, May 2012.

[18] T. Sakai, M. C. Du Plessis, G. Niu, and M. Sugiyama, "Semi-supervised classification based on classification from positive and unlabeled data," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2998–3006.

[19] T. Sakai, G. Niu, and M. Sugiyama, "Semi-supervised AUC optimization based on positive-unlabeled learning," *Mach. Learn.*, vol. 107, no. 4, pp. 767–794, 2018.

[20] F. Nie, H. Wang, H. Huang, and C. Ding, "Adaptive loss minimization for semi-supervised elastic embedding," in *Proc. Int. Joint Conf. Artif. Intell.*, 2013, pp. 1565–1571.

[21] M. Luo, L. Zhang, F. Nie, X. Chang, B. Qian, and Q. Zheng, "Adaptive semi-supervised learning with discriminative least squares regression," in *Proc. Int. Joint Conf. Artif. Intell.*, 2017, pp. 2421–2427.

[22] Y. Liu, Y. Guo, H. Wang, F. Nie, and H. Huang, "Semi-supervised classifications via elastic and robust embedding," in *Proc. AAAI Conf. Artif. Intell.*, 2017, pp. 2294–2300.

[23] C. Gong, T. Liu, D. Tao, K. Fu, E. Tu, and J. Yang, "Deformed graph Laplacian for semisupervised learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 10, pp. 2261–2274, Oct. 2015.

[24] C. Gong, D. Tao, S. J. Maybank, W. Liu, G. Kang, and J. Yang, "Multi-modal curriculum learning for semi-supervised image classification," *IEEE Trans. Image Process.*, vol. 25, no. 7, pp. 3249–3260, Jul. 2016.

[25] C. Gong, D. Tao, W. Liu, L. Liu, and J. Yang, "Label propagation via teaching-to-learn and learning-to-teach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 6, pp. 1452–1465, Jun. 2017.

[26] R. He, B. Hu, X. Yuan, and L. Wang, "M-estimators and half-quadratic minimization," in *Proc. Robust Recognit. Via Inf. Theor. Learn.*, 2014, pp. 3–11.

[27] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, "Spectral grouping using the Nystrom method," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 2, pp. 214–225, Feb. 2004.

[28] W. Jiang, F. Nie, and H. Huang, "Robust dictionary learning with capped ℓ1-norm," in *Proc. 24th Int. Conf. Artif. Intell.*, 2015, pp. 3590–3596.

[29] H. Pei, K. Wang, Q. Lin, and P. Zhong, "Robust semi-supervised extreme learning machine," *Knowl. Based Syst.*, vol. 159, pp. 203–220, Nov. 2018.

[30] K. Chen, Q. Lv, Y. Lu, and Y. Dou, "Robust regularized extreme learning machine for regression using iteratively reweighted least squares," *Neurocomputing*, vol. 230, pp. 345–358, Mar. 2017.

[31] F. Nie, H. Wang, H. Huang, and C. Ding, "Unsupervised and semi-supervised learning via ℓ1-norm graph," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 2268–2273.

[32] N. Aronszajn, "Theory of reproducing kernels," *Trans. Amer. Math. Soc.*, vol. 68, no. 3, pp. 337–404, 1950.

[33] C. S. Burrus, "Iterative reweighted least squares, version 1.12," Dept. Elect. Comput. Eng., Rice Univ., Houston, TX, USA, Rep., 2012.

[34] A. L. Yuille and A. Rangarajan, "The concave-convex procedure," in *Proc. Adv. Neural Inf. Process. Syst.*, 2002, pp. 1033–1040.

[35] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[36] G. H. Golub and C. F. Van Loan, *Matrix Computations*, vol. 3. Baltimore, MD, USA: Johns Hopkins Univ. Press, 2012.

[37] F. Chierichetti, S. Gollapudi, R. Kumar, S. Lattanzi, R. Panigrahy, and D. P. Woodruff, "Algorithms for ℓ_p low-rank approximation," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 806–814.

[38] P. Indyk, A. Vakilian, T. Wagner, and D. Woodruff, "Sample-optimal low-rank approximation of distance matrices," *arXiv preprint arXiv:1906.00339*, 2019.

[39] T. Liu and D. Tao, "Classification with noisy labels by importance reweighting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 3, pp. 447–461, Mar. 2016.

[40] P. L. Bartlett and S. Mendelson, "Rademacher and Gaussian complexities: Risk bounds and structural results," *J. Mach. Learn. Res.*, vol. 3, pp. 463–482, Nov. 2002.

[41] P. L. Bartlett, O. Bousquet, and S. Mendelson, "Local rademacher complexities," *Ann. Stat.*, vol. 33, no. 4, pp. 1497–1537, 2005.

[42] A. Frank and A. Asuncion, *UCI Machine Learning Repository*, Univ. California at Irvine, Irvine, CA, USA, 2010. [Online]. Available: http://archive.ics.uci.edu/ml

[43] E. Tu, Y. Zhang, L. Zhu, J. Yang, and N. Kasabov, "A graph-based semi-supervised k nearest-neighbor method for nonlinear manifold distributed data classification," *Inf. Sci.*, vols. 367–368, pp. 673–688, Nov. 2016.

[44] C. Gong, H. Shi, J. Yang, J. Yang, and J. Yanga, "Multi-manifold positive and unlabeled learning for visual analysis," *IEEE Trans. Circuits Syst. Video Technol.*, to be published, doi: 10.1109/TCSVT.2019.2903563.

[45] Z. Yang, W. W. Cohen, and R. Salakhutdinov, "Revisiting semi-supervised learning with graph embeddings," *arXiv preprint arXiv:1603.08861,* 2016.

[46] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling, "Semi-supervised learning with deep generative models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3581–3589.

Jingchen Ke received the B.E. degree from Northwestern Polytechnical University, Xi'an, China, in 2013. He is currently pursuing the M.S. degree with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China.

His research interests include machine learning and data mining.

Chen Gong (M'16) received the dual Doctoral degrees from Shanghai Jiao Tong University (SJTU), Shanghai, China, in 2016 and University of Technology Sydney, Ultimo, NSW, Australia, in 2017.

He is currently a Full Professor with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China. He has published more than 70 technical papers at prominent journals and conferences, such as the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON CYBERNETICS, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, the IEEE TRANSACTIONS ON MULTIMEDIA, the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, and the *ACM Transactions on Intelligent Systems and Technology*, and conferences NeurIPS, CVPR, AAAI, IJCAI, and ICDM.

Prof. Gong received the "Excellent Doctorial Dissertation" awarded by SJTU and Chinese Association for Artificial Intelligence and the "Wu Wen-Jun AI Excellent Youth Scholar Award." He was enrolled by the "Young Elite Scientists Sponsorship Program" of Jiangsu Province and China Association for Science and Technology. He also serves as the reviewer for more than 20 international journals, such as AIJ, the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, the IEEE TRANSACTIONS ON IMAGE PROCESSING, and also the SPC/PC member of several top-tier conferences, such as ICML, NeurIPS, CVPR, AAAI, IJCAI, ICDM, and AISTATS.

Tongliang Liu (M'l4) is currently a Lecturer with the School of Computer Science and the Faculty of Engineering, and a Core Member with the UBTECH Sydney AI Centre, University of Sydney, Darlington, NSW, Australia. He has authored and coauthored over 60 research papers, including the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, the IEEE TRANSACTIONS ON IMAGE PROCESSING, and the IEEE TRANSACTIONS ON CYBERNETICS, and conferences ICML, NeurIPS, AAAI, IJCAI, CVPR, ECCV, KDD, and ICME. His research interests include machine learning, computer vision, and data mining.

Mr. Liu is a recipient of the 2019 ICME Best Paper Award and DECRA from the Australian Research Council.

Lin Zhao received the Ph.D. degree in pattern recognition and intelligent systems form Xidian University, Xi'an, China, in 2017.

He is currently a Lecture with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China. His research interests include human-related computer vision tasks, such as human pose estimation and tracking, 3-D human shape reconstruction, and abnormally detection.

Jian Yang (M'08) received the Ph.D. degree in pattern recognition and intelligence systems from the Nanjing University of Science and Technology (NUST), Nanjing, China, in 2002.

In 2003, he was a Post-Doctoral Researcher with the University of Zaragoza, Zaragoza, Spain. From 2004 to 2006, he was a Post-Doctoral Fellow with Biometrics Centre, Hong Kong Polytechnic University, Hong Kong. From 2006 to 2007, he was a Post-Doctoral Fellow with the Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA. He is currently a Chang-Jiang Professor with the School of Computer Science and Technology, NUST. He has authored more than 200 scientific papers in pattern recognition and computer vision. His papers have been cited more than 5000 times in the Web of Science, and 13 000 times in Google Scholar. His research interests include pattern recognition, computer vision and machine learning.

Dr. Yang is/was an Associate Editor of *Pattern Recognition*, *Pattern Recognition Letters*, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, and *Neurocomputing*. He is a fellow of IAPR.

Dacheng Tao (F'15) is a Professor of computer science and an ARC Laureate Fellow with the School of Computer Science and the Faculty of Engineering, and the Inaugural Director of the UBTECH Sydney Artificial Intelligence Centre, University of Sydney, Darlington, NSW, Australia. His research results in artificial intelligence have expounded in one monograph and over 200 publications at prestigious journals and prominent conferences, such as the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, the *International Journal of Computer Vision*, and the *Journal of Machine Learning Research*, and conferences AAAI, IJCAI, NeurIPS, ICML, CVPR, ICCV, ECCV, ICDM, and KDD, with several best paper awards.

Prof. Tao received the 2018 IEEE ICDM Research Contributions Award and the 2015 Australian Scopus-Eureka prize. He is a fellow of the Australian Academy of Science.