Consensus-Based Distributed Kernel One-class Support Vector Machine for Anomaly Detection

Tianyao Wang, Fan He, Ruikai Yang, Zhixing Ye, Xiaolin Huang*

***Abstract*—One-class support vector machine (OCSVM) is one of the most widely used methods for learning from imbalanced data and has been successfully applied to numerous tasks such as anomaly detection. However, the study on decentralized OCSVM is currently limited to linear cases. The main challenge is how to communicate the non-parametric and local-data-dependent decision functions between neighboring nodes. To tackle it, this paper proposes a projection consensus constraint to formulate a decentralized OCSVM, where local solutions are assumed to be the projection of the global optimum on local reproducing kernel Hilbert spaces. A fast non-parametric solving algorithm is then designed based on alternating direction method of multipliers. Experiments on real-world anomaly datasets indicate that our method outperforms the existing distributed OCSVM methods while reducing the communication cost.**

I. Introduction

One-class support vector machine (OCSVM, [1]) and its variations have been widely used in classification tasks with imbalanced data. As a kernel-based method, OCSVM enjoys the advantage of kernel trick that can implicitly capture the nonlinear mapping to well distinguish different classes. A typical application is anomaly detection, where the training set usually contains only normal data, see [2]–[5] for reference. In recent years, with the advent of new scenarios such as the Internet of Things, distributed data sources have become increasingly popular [6], which raises challenges for OCSVM regarding how to effectively share information across different nodes and meanwhile reduce the communication cost.

In this context, various designs for distributed algorithms have been proposed. They can be generally categorized into i) local methods, which use only local data to construct local models without any communication; ii) centralized methods, where a fusion center (FC) aggregates information from all nodes to construct a centralized model; iii) decentralized methods, where only communications between neighboring nodes are allowed and the FC is prohibitive. Among these three methods, the decentralized one is generally preferred because of its advantages in communication efficiency, data privacy, and framework stability.

This work is supported by National Natural Science Foundation of China (No. 61977046), Shanghai Science and Technology Program (No. 22511105600), and Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102).

T. Wang is with Department of Automation and also Ningbo Artificial Intelligence Institute, Shanghai Jiao Tong University, China. E-mail: wang-tianyao@sjtu.edu.cn.

F. He, R. Yang, Z. Ye and X. Huang are with Department of Automation and also with Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, China. E-mail: {HF-inspire, ruikai.yang, yzx1213, xiaolinhuang}@sjtu.edu.cn

*Corresponding author.

However, the majority of current decentralized OCSVM algorithms are limited to linear cases [7], [8], primarily due to the difficulty of achieving consistency of decision functions in nonlinear cases. Linear algorithms manage it by exchanging decision functions across neighboring nodes. However, when using a nonlinear mapping, the decision function is non-parametric and dependent on local data, which brings challenges for communication design. To deal with it, existing works employ function value consistency instead of function consistency [9]. Other works apply random Fourier features (RFFs, [10]) such that one can obtain parametric decision functions [11]–[13].

In this paper, we propose a non-parametric decentralized algorithm for OCSVM and analyze its performance by applying it to decentralized anomaly detection. The key novelty is to establish a projection consensus constraint for OCSVM, which could be regarded as an extension of our previous work [14] from unsupervised learning to supervised one. We also design a decentralized algorithm based on alternating direction method of multipliers (ADMM, [15]). The effectiveness in both accuracy and communication has been evaluated on real-world anomaly detection datasets with the comparisons with state-of-the-art methods.

II. Related Works

OCSVM [1] and its variants such as least squares OCSVM [16], fuzzy OCSVM [17], AdaBoost based OCSVM [18], graph embedded OCSVM [19] have attracted considerable attention and are demonstrated to be effective in anomaly detection. Using kernel techniques has improved OCSVM's capacity to manage high-dimensional and nonlinear data.

With the growing popularity of distributed data sources, distributed learning has received significant attention. Several distributed algorithms have been proposed for solving linear SVM [8], [20]–[22]. However, they can not be directly applied to nonlinear OCSVM, since the nonlinear mapping in kernel methods can not be explicitly expressed.

To the best of our knowledge, there are only a few works on distributed nonlinear SVM or OCSVM. The work [23] proposes a distributed SVM in which each node exchanges support vectors (SVs) with its neighbors and a FC combines the local results. To reduce the high communication cost of transmitting SVs, the work [24] proposes a distributed semi-parametric SVM that transmits a function of SVs. However, these FC-based methods have the following limitations: i) the high communication cost to transmit local information to the FC, ii) lack of robustness in case of FC failures.

2023 International Joint Conference on Neural Networks (IJCNN) | 978-1-6654-8867-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/IJCNN54540.2023.10191788

Therefore, a decentralized framework that only allows communications among neighboring nodes has attracted growing interest. Popular decentralized algorithms include diffusion-based [25] and consensus-based [26], [27] algorithms. However, there has not been satisfactory work for nonlinear OCSVM due to the implicit mapping functions. Existing works apply RFFs to approximate kernel functions by mapping the input to a higher dimensional randomized feature space [11], [12]. As a result, the nonlinear OCSVM is transformed to a linear model. However, these approximate methods are incapable of obtaining the exact solution of kernel machines, which often leads to suboptimal prediction accuracy. Another popular approach is to enforce consensus on a small subset sampled from global training data [9]. Although these methods are somewhat useful for dealing with nonlinear OCSVM, they may lead to significant accuracy loss and high computation or communication cost. More recently, the work [14] adopts projection consensus constraints for kernel principal component analysis and solves the optimization problem by a distributed solver.

III. Method

A. Preliminaries

OCSVM aims to find a hyperplane with the maximum margin for separating all the data from the origin. If a new input falls inside the hyperplane, it is considered normal; otherwise, it is considered abnormal. Generally, real-world data are high-dimensional and nonlinear; thus, usually they can not be well classified by linear OCSVM. Then, a feature mapping function $\Phi(\cdot) : \mathcal{X} \rightarrow \mathcal{H}$ that maps the input space to a higher dimensional space is needed. The corresponding nonlinear one-class support vector machine can be formulated as the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w},\rho,\{\xi_i\}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{\nu N}\sum_{i=1}^{N}\xi_i - \rho \\ \text{s.t.} \quad & \Phi(\mathbf{x}_i)^\top \mathbf{w} \geq \rho - \xi_i, \\ & \xi_i \geq 0, \end{aligned} \tag{1}$$

where $\mathbf{x}_i$ are input data, $\mathbf{w}$ is the weight vector, b is the bias term of the separating hyperplane, ξ_i are slack variables, N is the total number of training data, $\nu \in (0, 1]$ controls the fraction of anomalies and SVs. The optimal discriminant function $g^*(\mathbf{x}) = \Phi(\mathbf{x})^\top \mathbf{w}^* - \rho^*$ can be obtained by solving (1). For any input $\mathbf{x}$, it is considered normal if $g(\mathbf{x}) \geq 0$; otherwise, it is considered abnormal.

B. Distributed OCSVM

In this section, we present the consensus-based distributed formulation of OCSVM. We first introduce terminology and notation conventions. Consider an undirected graph $\mathcal{G}(\mathcal{J}, \mathcal{E})$ with a set of nodes $\mathcal{J} := \{1, \ldots, J\}$ and edges $\mathcal{E} \subseteq \mathcal{J} \times \mathcal{J}$. For each node $j \in \mathcal{J}$, $\mathcal{B}_j \subseteq \mathcal{J}$ represents its one-hop neighbors and $|\mathcal{B}_j|$ denotes the number of its one-hop neighbors. Node j could only exchange data with its neighbors $q \in \mathcal{B}_j$.

The data $\mathbf{X}$ are distributed over J nodes: $\mathbf{X} = [\mathbf{X}_1 \ \mathbf{X}_2 \ \ldots \ \mathbf{X}_J] \in \mathcal{R}^{M \times N}$. Each node j has the local data $\mathbf{X}_j = [\mathbf{x}_1^j \ \mathbf{x}_2^j \ \ldots \ \mathbf{x}_{N_j}^j] \in \mathcal{R}^{M \times N_j}$, where $\mathbf{x}_i^j \in \mathcal{R}^M$ and N_j is the amount of data at node j. $\Phi(\mathbf{X}_j) = [\Phi(\mathbf{x}_1^j) \ \Phi(\mathbf{x}_2^j) \ \ldots \ \Phi(\mathbf{x}_{N_j}^j)] \in \mathcal{R}^{P \times N_j}$, where $\Phi(\mathbf{x}^j)$ lies in the high-dimensional feature space $\mathcal{R}^P$. The kernel matrix $\mathbf{K}(\mathbf{X}_i, \mathbf{X}_j)$ is calculated by inner products between each pair of points in the feature space: $\mathbf{K}(\mathbf{X}_i, \mathbf{X}_j) = \left[\Phi(\mathbf{x}_p^i)^\top \Phi(\mathbf{x}_q^j)\right]_{1 \leq p \leq N_i, 1 \leq q \leq N_j} \in \mathcal{R}^{N_i \times N_j}$.

Then we present a distributed formulation of OCSVM. Replacing the global variable $(\mathbf{w}, \rho)$ by local variables $\{(\mathbf{w}_j, \rho_j)\}_{j=1}^J$ and adding consensus constraints among neighboring nodes, we can reformulate the problem (1) as the following decentralized problem,

$$\begin{aligned} \min_{\{\mathbf{w}_j,\rho_j,\boldsymbol{\xi}_j\}} \quad & \frac{1}{2}\sum_{j=1}^{J}\|\mathbf{w}_j\|^2 + \frac{1}{\nu N}\sum_{j=1}^{J}\mathbf{1}_j^\top \boldsymbol{\xi}_j - \sum_{j=1}^{J}\rho_j \\ \text{s.t.} \quad & \Phi(\mathbf{X}_j)^\top \mathbf{w}_j \succeq \rho_j \mathbf{1}_j - \boldsymbol{\xi}_j, \quad \forall j \in \mathcal{J}, \\ & \boldsymbol{\xi}_j \succeq \mathbf{0}, \quad \forall j \in \mathcal{J}, \\ & \mathbf{w}_j = \mathbf{w}_q, \quad \forall j \in \mathcal{J}, q \in \mathcal{B}_j, \\ & \rho_j = \rho_q, \quad \forall j \in \mathcal{J}, q \in \mathcal{B}_j. \end{aligned} \tag{2}$$

If the network is connected, the network-wide consensus can be achieved after convergence, namely, $\mathbf{w}_1 = \mathbf{w}_2 = \ldots = \mathbf{w}_J = \mathbf{w}$ and $\rho_1 = \rho_2 = \ldots = \rho_J = \rho$. For any input $\mathbf{x}$, each node j obtains its local discriminant function $g(\mathbf{x}) = \Phi(\mathbf{x})^\top \mathbf{w}_j - \rho_j$. But recall that $\mathbf{w}$ lies in the high-dimensional unknown space, such that the biggest challenge in algorithm design is to handle the constraint $\mathbf{w}_j = \mathbf{w}_q$.

C. Projection Consensus Constraints

In this paper, we propose a new projection consensus constraint instead of the strict consensus constraint $\mathbf{w}_j = \mathbf{w}_q$. The implicit mapping functions can be eliminated and converted into calculations of the kernel matrix during the solving process. Besides, through the information exchange among neighboring nodes, the entire network can reach great consensus after convergence.

Let $\mathbf{u}$ be the global optimal solution to the variable $\mathbf{w}$ in problem (1), we propose a new projection as shown in

$$\mathbf{w}_j = \Phi\left(\mathbf{X}_j\right)\mathbf{K}_j^{-1}\Phi\left(\mathbf{X}_j\right)^\top \mathbf{u}, \quad \forall j \in \mathcal{J}, \tag{3}$$

where the local kernel matrix is defined by $\mathbf{K}_j = \left[\Phi(\mathbf{x}_p^j)^\top \Phi(\mathbf{x}_q^j)\right]_{1 \leq p,q \leq N_j} \in \mathcal{R}^{N_j \times N_j}$. The local solution $\mathbf{w}_j$ is the projection vector of the global solution on the column space of local dataset. The projection has two advantages. Due to the data heterogeneous in different nodes, the data-dependent local solution $\mathbf{w}_j$ may differ from the global optimal solution $\mathbf{u}$. Compared with the strict consensus constraint $\mathbf{w}_j = \mathbf{w}_q = \ldots = \mathbf{w}$, the proposed projection provides a more appropriate representation between the local and global solution. On the other hand, the projection helps eliminate the implicit calculation of the mapping function in our subsequent solving process.

To keep consensus among neighboring nodes, we introduce local variables $\mathbf{z}_j$ in place of the global optimal solution $\mathbf{u}$,

$$\mathbf{w}_j = \Phi\left(\mathbf{X}_j\right)\mathbf{K}_j^{-1}\Phi\left(\mathbf{X}_j\right)^\top \mathbf{z}_j, \quad \forall j \in \mathcal{J}. \tag{4}$$

Adding the constraints $\mathbf{z}_j = \mathbf{z}_q, q \in \mathcal{B}_j$, we reformulate the optimization problem as follows,

$$\begin{aligned} \min_{\{\mathbf{w}_j,\rho_j,\boldsymbol{\xi}_j\}} \quad & \frac{1}{2}\sum_{j=1}^{J}\left\|\mathbf{w}_j\right\|^2 + \frac{1}{\nu N}\sum_{j=1}^{J}\mathbf{1}_j^\top\boldsymbol{\xi}_j - \sum_{j=1}^{J}\rho_j \\ \text{s.t.} \quad & \Phi(\mathbf{X}_j)^\top\mathbf{w}_j \succeq \rho_j\mathbf{1}_j - \boldsymbol{\xi}_j, \quad \forall j \in \mathcal{J}, \\ & \boldsymbol{\xi}_j \succeq \mathbf{0}, \quad \forall j \in \mathcal{J}, \\ & \mathbf{w}_j = \Phi\left(\mathbf{X}_j\right)\mathbf{K}_j^{-1}\Phi\left(\mathbf{X}_j\right)^\top\mathbf{z}_j, \quad \forall j \in \mathcal{J}, \\ & \mathbf{z}_j = \mathbf{z}_q, \quad \forall j \in \mathcal{J}, q \in \mathcal{B}_j, \\ & \rho_j = \rho_q, \quad \forall j \in \mathcal{J}, q \in \mathcal{B}_j. \end{aligned} \tag{5}$$

To simplify notations, the following augmented matrices are defined:

- Augmented matrix $\mathbf{Z} = [\mathbf{z}_1\ \mathbf{z}_2\ \ldots\ \mathbf{z}_J] \in \mathcal{R}^{P\times J}$.
- Augment column vector $\mathbf{e}_j = [0\ \ldots\ 1\ \ldots\ 0]^\top \in \mathcal{R}^J$, where the j^{th} entry is 1 and others are zeros.
- Augmented matrix $\mathbf{Q}_j = [\mathbf{e}_{q_1}\ \mathbf{e}_{q_2}\ \ldots] \in \mathcal{R}^{J\times|\mathcal{B}_j|}$, where $q_1, q_2, \ldots \in \mathcal{B}_j$.
- Augmented row vector $\mathbf{E}_j = [1\ \ldots\ 1] \in \mathcal{R}^{|\mathcal{B}_j|}$.

According to the above definitions, $\mathbf{z}_j$ can be represented by $\mathbf{Z}\mathbf{e}_j$, $[\mathbf{z}_{q_1}\ \mathbf{z}_{q_2}\ \ldots]$ can be represented by $\mathbf{Z}\mathbf{Q}_j$. Replacing $\mathbf{w}_j = \Phi\left(\mathbf{X}_j\right)\mathbf{K}_j^{-1}\Phi\left(\mathbf{X}_j\right)^\top\mathbf{z}_j$ and $\mathbf{z}_j = \mathbf{z}_q$ with $\mathbf{w}_j\mathbf{E}_j = \Phi\left(\mathbf{X}_j\right)\mathbf{K}_j^{-1}\Phi\left(\mathbf{X}_j\right)^\top\mathbf{Z}\mathbf{Q}_j$, we reformulate the problem (5) as

$$\begin{aligned} \min_{\{\mathbf{w}_j,\rho_j,\boldsymbol{\xi}_j\}} \quad & \frac{1}{2}\sum_{j=1}^{J}\left\|\mathbf{w}_j\right\|^2 + \frac{1}{\nu N}\sum_{j=1}^{J}\mathbf{1}_j^\top\boldsymbol{\xi}_j - \sum_{j=1}^{J}\rho_j \\ \text{s.t.} \quad & \Phi(\mathbf{X}_j)^\top\mathbf{w}_j \succeq \rho_j\mathbf{1}_j - \boldsymbol{\xi}_j, \quad \forall j \in \mathcal{J}, \\ & \boldsymbol{\xi}_j \succeq \mathbf{0}, \quad \forall j \in \mathcal{J}, \\ & \mathbf{w}_j\mathbf{E}_j = \Phi\left(\mathbf{X}_j\right)\mathbf{K}_j^{-1}\Phi\left(\mathbf{X}_j\right)^\top\mathbf{Z}\mathbf{Q}_j, \quad \forall j \in \mathcal{J}, \\ & \rho_j = \rho_q, \quad \forall j \in \mathcal{J}, q \in \mathcal{B}_j. \end{aligned} \tag{6}$$

Consequently, the local discriminant function for any input $\mathbf{x}_t$ on node j is written as:

$$\begin{aligned} g(\mathbf{x}_t) &= \operatorname{sgn}\left(\Phi(\mathbf{x}_t)^\top\mathbf{w}_j - \rho_j\right) \\ &= \operatorname{sgn}\left(\Phi(\mathbf{x}_t)^\top\Phi(\mathbf{X}_j)\mathbf{K}_j^{-1}\Phi(\mathbf{X}_j)^\top\mathbf{z}_j - \rho_j\right). \end{aligned} \tag{7}$$

IV. Algorithm and Analysis

In this section, a fully distributed nonlinear OCSVM algorithm is developed. Alternating direction method of multipliers (ADMM, [15]) has been been demonstrated to be a powerful approach in distributed learning with a strong convergence property. Here ADMM is used to solve the proposed global consensus problem.

Introducing the Lagrange multipliers $\boldsymbol{\lambda}_j$ and $\boldsymbol{\mu}_j$ corresponding to the inequality constraints, $\boldsymbol{\alpha}_j$ and γ_j corresponding to the equality constraints, we can formulate the augmented Lagrangian function as (8). The term "augmented" indicates that two quadratic terms are added to further regularize the equality constraints. The quadratic terms could ensure strict convexity of $\mathcal{L}$ with respect to $\mathbf{w}_j$ and ρ_j, thus ensure convergence to the optimal solution. η is used to control the convergence rate and dispersion level of solutions among nodes.

$$\begin{aligned} \mathcal{L} = &\frac{1}{2}\sum_{j=1}^{J}\left\|\mathbf{w}_j\right\|^2 + \frac{1}{\nu N}\sum_{j=1}^{J}\mathbf{1}_j^\top\boldsymbol{\xi}_j - \sum_{j=1}^{J}\rho_j \\ &- \sum_{j=1}^{J}\boldsymbol{\lambda}_j{}^\top\left(\Phi\left(\mathbf{X}_j\right)^\top\mathbf{w}_j - \rho_j\mathbf{1}_j + \boldsymbol{\xi}_j\right) - \sum_{j=1}^{J}\boldsymbol{\mu}_j{}^\top\boldsymbol{\xi}_j \\ &+ \operatorname{tr}\left[\sum_{j=1}^{J}\boldsymbol{\alpha}_j^\top\left(\mathbf{w}_j\mathbf{E}_j - \Phi\left(\mathbf{X}_j\right)\mathbf{K}_j^{-1}\Phi\left(\mathbf{X}_j\right)^\top\mathbf{Z}\mathbf{Q}_j\right)\right] \\ &+ \frac{\eta}{2}\sum_{j=1}^{J}\left\|\mathbf{w}_j\mathbf{E}_j - \Phi\left(\mathbf{X}_j\right)\mathbf{K}_j^{-1}\Phi\left(\mathbf{X}_j\right)^\top\mathbf{Z}\mathbf{Q}_j\right\|^2 \\ &+ \sum_{j=1}^{J}\sum_{q\in\mathcal{B}_j}\gamma_j(\rho_j - \rho_q) + \frac{\eta}{2}\sum_{j=1}^{J}\sum_{q\in\mathcal{B}_j}\left\|\rho_j - \rho_q\right\|^2. \end{aligned} \tag{8}$$

Then we update variables by minimizing $\mathcal{L}$ with respect to one set of variables while fixing others. Lagrange multipliers $\boldsymbol{\alpha}_j$ and $\boldsymbol{\gamma}_j$ are updated using gradient ascent.

To simplify the solving of $\mathbf{w}_j$ and $\boldsymbol{\xi}_j$, we can transform the primal problem into its dual problem. According to the Karush-Kuhn-Tucker (KKT) optimality conditions, we take the derivative of $\mathcal{L}$ with respect to $\mathbf{w}_j$ and $\boldsymbol{\xi}_j$ equal to zero, then obtain

$$\begin{aligned} \mathbf{w}_j = \frac{1}{a_j}\Big(&\eta\Phi\left(\mathbf{X}_j\right)\mathbf{K}_j^{-1}\Phi\left(\mathbf{X}_j\right)^\top\mathbf{Z}\mathbf{Q}_j\mathbf{E}_j^\top \\ &+\Phi\left(\mathbf{X}_j\right)\boldsymbol{\lambda}_j - \boldsymbol{\alpha}_j\mathbf{E}_j^\top\Big), \end{aligned} \tag{9}$$

$$\frac{J}{\nu N}\mathbf{1}_j - \boldsymbol{\lambda}_j - \boldsymbol{\mu}_j = 0, \tag{10}$$

where $a_j = 1 + \eta|\mathcal{B}_j|$. The optimality conditions also require $\boldsymbol{\lambda}_j \succeq \mathbf{0}_j$ and $\boldsymbol{\mu}_j \succeq \mathbf{0}_j$, allowing (10) to be replaced by $\mathbf{0}_j \preceq \boldsymbol{\lambda}_j \preceq \frac{1}{\nu N}\mathbf{1}_j$. Substituting (9) and (10) into (8), we obtain the Lagrangian dual function $\tilde{\mathcal{L}}$. $\boldsymbol{\lambda}_j^{(t+1)}$ can be obtained through

$$\begin{aligned} \boldsymbol{\lambda}_j^{(t+1)} &= \underset{\mathbf{0}_j\preceq\boldsymbol{\lambda}_j\preceq\frac{1}{\nu N}\mathbf{1}_j}{\arg\max}\ \tilde{\mathcal{L}}(\boldsymbol{\lambda}_j) \\ &= \underset{\mathbf{0}_j\preceq\boldsymbol{\lambda}_j\preceq\frac{1}{\nu N}\mathbf{1}_j}{\arg\max}\ -\frac{1}{2a_j}\boldsymbol{\lambda}_j{}^\top\mathbf{K}_j\boldsymbol{\lambda}_j \\ &- \left[\frac{1}{a_j}\left(\eta\Phi(\mathbf{X}_j)^\top\mathbf{Z}^{(t)}\mathbf{Q}_j - \Phi(\mathbf{X}_j)^\top\boldsymbol{\alpha}_j^{(t)}\right)\mathbf{E}_j^\top - \rho_j\mathbf{1}_j^\top\right]^\top\boldsymbol{\lambda}_j. \end{aligned} \tag{11}$$

Now we consider the update of $\mathbf{Z}$. Since there exists an association between $\mathbf{w}_j$ and $\mathbf{z}_j$, we can eliminate $\mathbf{w}_j$ and only update $\mathbf{z}_j$ by substituting the expression of $\mathbf{w}_j$

into the Lagrangian function. In this way, computation and communication cost can be both reduced. $\tilde{\mathcal{L}}(\mathbf{Z})$ is given by

$$\begin{aligned}\tilde{\mathcal{L}}(\mathbf{Z}) = &\frac{\eta}{2a_j}\sum_{j=1}^{J}\left\|\Phi\left(\mathbf{X}_j\right)\mathbf{K}_j^{-1}\Phi\left(\mathbf{X}_j\right)^\top\mathbf{Z}\mathbf{Q}_j\right\|^2\\ &-\sum_{j=1}^{J}\frac{\eta}{a_j}\left(\Phi\left(\mathbf{X}_j\right)^\top\mathbf{Z}\mathbf{Q}_j\mathbf{E}_j^\top\right)^\top\boldsymbol{\lambda}_j\\ &-\operatorname{tr}\left(\sum_{j=1}^{J}\frac{1}{a_j}\boldsymbol{\alpha}_j^\top\Phi\left(\mathbf{X}_j\right)\mathbf{K}_j^{-1}\Phi\left(\mathbf{X}_j\right)^\top\mathbf{Z}\mathbf{Q}_j\right).\end{aligned} \tag{12}$$

Taking the derivative with respect to $\mathbf{Z}$ equal to zero, we can obtain the update of $\mathbf{Z}$. However, since $\Phi\left(\mathbf{X}_j\right)\mathbf{K}_j^{-1}\Phi\left(\mathbf{X}_j\right)^\top$ is implicit, it is hard to obtain an exact solution of $\mathbf{Z}$. Thus, we consider a relaxation that $\Phi\left(\mathbf{X}_j\right)\mathbf{K}_j^{-1}\Phi\left(\mathbf{X}_j\right)^\top\mathbf{Z}\mathbf{Q}_j$ is approximated as $\mathbf{Z}\mathbf{Q}_j$. After that, $\tilde{\mathcal{L}}(\mathbf{Z})$ is reduced to $\mathcal{U}(\mathbf{Z})$ and the update of $\mathbf{Z}^{(t+1)}$ can be obtained from the optimality condition as the following,

$$\mathbf{Z}^{(t+1)} = \sum_{j=1}^{J}\Phi(\mathbf{X}_j)\left(\mathbf{K}_j^{-1}\Phi(\mathbf{X}_j)^\top\boldsymbol{\alpha}_j^{(t)} + \eta\boldsymbol{\lambda}_j^{(t+1)}\mathbf{E}_j\right)\mathbf{Q}_j^\top\mathbf{H}, \tag{13}$$

where

$$\mathbf{H} = \left(\eta\sum_{j=1}^{J}\mathbf{Q}_j\mathbf{Q}_j^\top\right)^{-1} = \operatorname{diag}\left\{\frac{1}{\eta\left|\mathcal{B}_1\right|},\dots,\frac{1}{\eta\left|\mathcal{B}_J\right|}\right\}. \tag{14}$$

Further we get

$$\begin{aligned}\mathbf{z}_j^{(t+1)} &= \mathbf{Z}^{(t+1)}\mathbf{e}_j\\ &= \sum_{l=1}^{J}\Phi(\mathbf{X}_l)\left(\mathbf{K}_l^{-1}\Phi\left(\mathbf{X}_l\right)^\top\boldsymbol{\alpha}_l^{(t)} + \eta\boldsymbol{\lambda}_l^{(t+1)}\mathbf{E}_l\right)\mathbf{Q}_l^\top\mathbf{H}\mathbf{e}_j\\ &= \sum_{l\in\mathcal{B}_j}\Phi(\mathbf{X}_l)\left(\mathbf{K}_l^{-1}\Phi\left(\mathbf{X}_l\right)^\top\boldsymbol{\alpha}_l^{(t)} + \eta\boldsymbol{\lambda}_l^{(t+1)}\mathbf{E}_l\right)\mathbf{Q}_l^\top\mathbf{H}\mathbf{e}_j.\end{aligned} \tag{15}$$

Left-multiplying by $\Phi\left(\mathbf{X}_q\right)^\top$ and recalling that $\mathbf{K}(\mathbf{X}_q,\mathbf{X}_l) = \Phi\left(\mathbf{X}_q\right)^\top\Phi\left(\mathbf{X}_l\right)$, we obtain

$$\begin{aligned}&\Phi(\mathbf{X}_q)^\top\mathbf{z}_j^{(t+1)} =\\ &\sum_{l\in\mathcal{B}_j}\mathbf{K}(\mathbf{X}_q,\mathbf{X}_l)\left(\mathbf{K}_l^{-1}\Phi(\mathbf{X}_l)^\top\boldsymbol{\alpha}_l^{(t)} + \eta\boldsymbol{\lambda}_l^{(t+1)}\mathbf{E}_l\right)\mathbf{Q}_l^\top\mathbf{H}\mathbf{e}_j.\end{aligned} \tag{16}$$

Next, we consider the update of ρ_j. Take the derivative of $\tilde{L}(\rho_j)$ with respect to ρ_j equal to zero. $\rho_j^{(t+1)}$ is given by

$$\rho_j^{(t+1)} = \frac{1}{a_j-1}\left(\eta\sum_{q\in\mathcal{B}_j}\rho_q^{(t+1)} + 1 - |\mathcal{B}_j|\gamma_j^{(t)} - \mathbf{1}_j^\top\boldsymbol{\lambda}_j^{(t+1)}\right). \tag{17}$$

Algorithm 1 PND-OCSVM

```
Input: Data $\mathbf{X}_j$, the ADMM hyper-parameter $\eta$.
Output: $\{\Phi(\mathbf{X}_j)^\top\mathbf{z}_j^{(t+1)}\}$ and $\{\rho_j\}$
 1: Randomly initialize $\Phi(\mathbf{X}_j)^\top\mathbf{z}_j^{(0)}$ and $\rho_j^{(0)}$; initialize
    $\Phi(\mathbf{X}_j)^\top\boldsymbol{\alpha}_j^{(0)} = \mathbf{0}_{N_j\times|\mathcal{B}_j|}$ and $\boldsymbol{\gamma}_j^{(0)} = 0$.
 2: for $j = 1,\dots,\mathcal{J}$ do in parallel:
 3: Distribute $\mathbf{X}_j$ to neighbors $q\in\mathcal{B}_j$.
 4: Calculate $\mathbf{K}(\mathbf{X}_p,\mathbf{X}_q), \forall p,q\in\mathcal{B}_j$.
 5: Set $t = 0$.
 6: repeat
 7:     Compute $\boldsymbol{\lambda}_j^{(t+1)}$ via (11).
 8:     Broadcast $\boldsymbol{\lambda}_j^{(t+1)}$ to neighbors $q$.
 9:     Compute $\Phi(\mathbf{X}_q)^\top\mathbf{z}_j^{(t+1)}$ and $\rho_j^{(t+1)}$ via (16), (17).
10:     Broadcast $\Phi(\mathbf{X}_q)^\top\mathbf{z}_j^{(t+1)}$ and $\rho_j^{(t+1)}$ to neighbors $q$.
11:     Compute $\Phi(\mathbf{X}_j)^\top\boldsymbol{\alpha}_j^{(t+1)}$ and $\gamma_j^{(t+1)}$ via (19), (20).
12:     Broadcast $\Phi(\mathbf{X}_j)^\top\boldsymbol{\alpha}_j$ to neighbors $q$.
13:     $t = t + 1$.
14: until the criteria is achieved.
```

Finally, we update the multiplier $\boldsymbol{\alpha}_j$ by

$$\begin{aligned}&\boldsymbol{\alpha}_j^{(t+1)} = \boldsymbol{\alpha}_j^{(t)}\\ &+\eta\Bigg[\Phi(\mathbf{X}_j)\mathbf{K}_j^{-1}\Phi(\mathbf{X}_j)^\top\mathbf{Z}^{(t+1)}\mathbf{Q}_j\left(\frac{\eta}{a_j}\mathbf{E}_j^\top\mathbf{E}_j - \mathbf{I}_j\right)\\ &+\frac{1}{a_j}\left(\Phi(\mathbf{X}_j)\boldsymbol{\lambda}_j^{(t+1)} - \boldsymbol{\alpha}_j^{(t)}\mathbf{E}_j^\top\right)\mathbf{E}_j\Bigg].\end{aligned} \tag{18}$$

Since Φ is implicit and $\Phi(\mathbf{X}_j)^\top\boldsymbol{\alpha}_j^{(t+1)}$ occurs in (11) and (16), we left-multiply (18) by $\Phi(\mathbf{X}_j)^\top$ and recall that $\mathbf{K}_j\mathbf{K}_j^{-1} = \mathbf{I}$, which yields

$$\begin{aligned}\Phi(\mathbf{X}_j)^\top\boldsymbol{\alpha}_j^{(t+1)} &= \Phi(\mathbf{X}_j)^\top\boldsymbol{\alpha}_j^{(t)}\\ &+\eta\Bigg[\Phi(\mathbf{X}_j)^\top\mathbf{Z}^{(t+1)}\mathbf{Q}_j\left(\frac{\eta}{a_j}\mathbf{E}_j^\top\mathbf{E}_j - \mathbf{I}_j\right)\\ &+\frac{1}{a_j}\left(\mathbf{K}_j\boldsymbol{\lambda}_j^{(t+1)} - \Phi(\mathbf{X}_j)^\top\boldsymbol{\alpha}_j^{(t)}\mathbf{E}_j^\top\right)\mathbf{E}_j\Bigg].\end{aligned} \tag{19}$$

Likewise, γ_j is updated by

$$\gamma_j^{(t+1)} = \gamma_j^{(t)} + \eta\sum_{q\in\mathcal{B}_j}(\rho_j^{(t+1)} - \rho_q^{(t+1)}). \tag{20}$$

The proposed algorithm, named as projection nonlinear distributed one-class support vector machine (PND-OCSVM), is summarized in Alg. 1. The message dissemination among nodes in a 5-node network is shown in Fig. 1. Each node j acquires raw data from neighboring nodes in order to calculate and store the kernel matrices $\mathbf{K}(\mathbf{X}_p,\mathbf{X}_q)$. Next, node j updates its local variables and exchanges messages with neighboring nodes until the algorithm converges. In PND-OCSVM, messages are only exchanged between neighboring nodes, but local nodes can still communicate implicitly with multi-hops neighbors. As shown in Fig. 1, node 2 and 3 cannot communicate directly but they are both connected with node 1 and 5. Node 1 and 5 calculate the kernel matrix $\mathbf{K}_{(2,3)}$ with

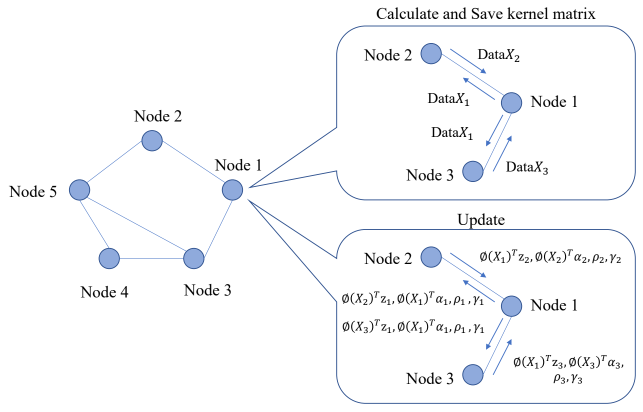
Calculate and Save kernel matrix
Node 2
DataX_2
DataX_1
Node 1
DataX_1
Node 3
DataX_3
Node 2
Node 1
Node 5
Node 4
Node 3
Update
Node 2
$\emptyset(X_1)^T z_2, \emptyset(X_2)^T \alpha_2, \rho_2, \gamma_2$
$\emptyset(X_2)^T z_1, \emptyset(X_1)^T \alpha_1, \rho_1, \gamma_1$
Node 1
$\emptyset(X_3)^T z_1, \emptyset(X_1)^T \alpha_1, \rho_1, \gamma_1$
$\emptyset(X_1)^T z_3, \emptyset(X_3)^T \alpha_3,$
ρ_3, γ_3
Node 3

Fig. 1. Message dissemination among nodes in a 5-node network.

$\mathbf{X}_2$ and $\mathbf{X}_3$, then the information of $\mathbf{K}_{(2,3)}$ is packaged in $\Phi(\mathbf{X}_2)^\top[\mathbf{z}_1 \quad \mathbf{z}_5]$ and $\Phi(\mathbf{X}_3)^\top[\mathbf{z}_1 \quad \mathbf{z}_5]$, and is sent to node 2 and 3 respectively. As a result, node 2 and 3 both receive the information of node 3 and 2.

Then we provide an analysis of the communication and computation cost of PND-OCSVM. The communication cost can be calculated by the number of scalars transmitted during each iteration. At each iteration, node j first broadcasts $\boldsymbol{\lambda}_j$ to its neighbors $q \in \mathcal{B}_j$. Next, it broadcasts $\Phi(\mathbf{X}_q)^\top \mathbf{z}_j$ along with ρ_j to its neighbors. Finally node j broadcasts $\Phi(\mathbf{X}_j)^\top \boldsymbol{\alpha}_j$ to its neighbors. Note that $\boldsymbol{\lambda}_j$ and $\Phi(\mathbf{X}_q)^\top \mathbf{z}_j$ are $N_j \times 1$ vectors, $\Phi(\mathbf{X}_j)^\top \boldsymbol{\alpha}_j$ is a $N_j \times |B_j|$ matrix, and ρ_j is a scalar. Therefore, the communication cost of each node in one ADMM iteration is $\mathcal{O}(N_j|\mathcal{B}_j| + \sum_{q\in\mathcal{B}_j} N_q + |\mathcal{B}_j| + N_j|\mathcal{B}_j|^2)$. When data are evenly distributed, that is, $N_1 = N_2 = \ldots = N_J$, the communication cost of each node in a single iteration is $\mathcal{O}((N_j|\mathcal{B}_j| + 2N_j + 1)|\mathcal{B}_j|)$.

The computation cost comes from the following parts: for each node, updating $\boldsymbol{\lambda}_j$ costs $\mathcal{O}(N_j^3)$, updating $\Phi(\mathbf{X}_q)^\top \mathbf{z}_j$ costs $\mathcal{O}\left(|\mathcal{B}_j| \sum_{l\in\mathcal{B}_j} N_l \left(\max_{l\in\mathcal{B}_j}\{N_l\}\right)\right)$, updating ρ_j costs $\mathcal{O}(N_j)$, updating $\Phi(\mathbf{X}_j)^\top \boldsymbol{\alpha}_j$ costs $\mathcal{O}(N_j^2|\mathcal{B}_j|)$, and updating γ_j costs $\mathcal{O}(\mathcal{B}_j)$. When data are evenly distributed, the computation cost of each node in one iteration is $\mathcal{O}(\max\{N^3, |\mathcal{B}_j|^2 N^2\})$. Compared with the Centralized OCSVM, which solves the original problem at a central node and thus costs $\mathcal{O}((JN)^3)$, PND-OCSVM has considerably lower computation cost since $|\mathcal{B}_j| \ll J$.

In addition, the complexity of each node in one iteration in PND-OCSVM is compared with those in other OCSVM algorithms in Tab. I, including Centralized OCSVM and two advanced decentralized OCSVM algorithms: MoM-NDOCSVM [9], doOCSVM [11]. Recall that M is the dimension of original input. In MoM-NDOCSVM, L is the size of the preselected global matrix. In doOCSVM, D is the size of randomized feature space after mapping. The communication cost of PND-OCSVM depends on the size of local dataset and the number of neighbors which are usually small in distributed environments. Subsequent experiments also show that PND-OCSVM significantly reduces the communication cost while achieving high accuracy compared with other distributed OCSVM algorithms.

TABLE I
COMPLEXITY COMPARISONS

	Communication	Computation
PND-OCSVM	$\mathcal{O}(\lvert\mathcal{B}_j\rvert N)$	$\mathcal{O}(\max\{N^3, \lvert\mathcal{B}_j\rvert^2 N^2\})$
Centralized OCSVM	$\mathcal{O}(JNM)$	$\mathcal{O}(J^3N^3)$
MoM-NDOCSVM [9]	$\mathcal{O}(\lvert\mathcal{B}_j\rvert L)$	$\mathcal{O}(\max\{N^3, N^2L, NL^2\})$
doOCSVM [11]	$\mathcal{O}(\lvert\mathcal{B}_j\rvert D)$	$\mathcal{O}(D(\lvert\mathcal{B}_j\rvert + 7))$

V. EXPERIMENTS

This section evaluates the proposed PND-OCSVM on synthetic and real-world anomaly detection datasets. Methods for comparison include the Centralized OCSVM, Local OCSVM, and several state-of-the-art distributed OCSVM methods.

A. *Experimental Setting*

Hardware and Software: All the experiments are carried out in the Python environment on a 64-bit CPU 8 Core (clock speed of 3.50 GHz) with 64GB of memory.

Datasets: For the synthetic dataset, normal samples are randomly selected inside a circle with center $C(-0.5, -0.5)$, radius $R = 1$ and a circle with center $C(2, 2)$, radius $R = \sqrt{2}$, while abnormal samples are drawn from a uniform distribution with the interval $[-4, 4]$ outside the aforementioned circles. The entire dataset contains $N = 6000$ samples, of which 90% are normal and 10% are abnormal. Besides, we select four real-world datasets from different applications. The detailed information is summarized in Tab. II. Each dataset is normalized to $[0, 1]$. In the training phase, only normal data are used, and in the testing phase, both normal and abnormal data are used. To form the datasets in a distributed environment, the training data are randomly distributed across local nodes. All the testing data are evaluated for each node.

TABLE II
DATASETS SUMMARY

Datasets	Application Domain	Attributes	No.instances (Training/Testing (normal+abnormal))
KDDCup99 [28]	Intrusion Detection	49	6000/2200(2000+200)
Cardio [29]	Healthcare	21	1000/600(450+150)
Backdoor [30]	Network	196	6000/3000(2000+1000)
Fraud [29]	Finance	29	4000/2492(2000+492)

Evaluation Measures: We first introduce two metrics: True Positive Rate (TPR) and False Positive Rate (FPR). TPR is the ratio of correctly identified abnormal samples to actual abnormal samples, and FPR is the ratio of normal samples incorrectly identified as abnormal to actual normal samples. High TPR and low FPR are generally preferred. Besides, F1-score is commonly used in anomaly detection to provide an overall evaluation. Since class imbalance always exists in anomaly detection, area under ROC curve (AUC) is also applied.

B. Parameter settings

In PND-OCSVM, we select the inverse length scale γ of Gaussian kernel from $\gamma \in \{0.01, 0.1, 0.5, 1, 5, 10\}$ and the hyperparameter ν from $\nu \in \{10^{-2}, 10^{-3}, 10^{-4}\}$ via gird search using the performance (F1-score or AUC) on a small holdout set (20% of randomly drawn testing samples).

C. Competing Methods

PND-OCSVM is compared with Centralized and Local OCSVM first. Since PND-OCSVM uses neighboring data for calculating the kernel matrix, it is also compared with Local OCSVM using local and neighboring data. Other distributed OCSVM algorithms are also compared, including MoM-NDOCSVM [9] and doOCSVM [11]. The specific descriptions of comparison methods are as follows:

- Centralized OCSVM: Each node gets access to all data from other nodes and trains a centralized OCSVM.
- Local OCSVM: Each node trains a local OCSVM with local data independently.
- Local OCSVM with Neighbors: Each node trains a local OCSVM using local and neighboring data.
- MoM-NDOCSVM: It reformulates the centralized nonlinear SVM into sub-problems linked via consensus constraints and pursues the consensus on a preselected global matrix [9]. This method is applied to OCSVM in this paper. Preselected vectors are sampled from global training data and the number of samples is L.
- doOCSVM: It applies RFFs to approximate the mapping function and then reduces nonlinear OCSVM to a linear model. D is the size of randomized feature space.

All competing methods use the same parameter settings for γ and ν as PND-OCSVM. To keep the same communication for PND-OCSVM, MoM-NDOCSVM and doOCSVM, $L = D = N_j(|\mathcal{B}_j| + 2)$ can be set. Besides, changing the values of L and D can produce different results in terms of accuracy and communication cost.

D. Experimental Results

1) Performance: Consider a network with $J = 20$ nodes and $|\mathcal{B}_j| = 2$ neighbors per node. For fair comparisons, the communication cost for MoM-NDOCSVM, doOCSVM, and PND-OCSVM is maintained at the same level. Besides, experiments of doOCSVM* are conducted, which doubles the D of doOCSVM and thus doubles the communication cost. Tab. III shows the average results over 20 nodes on synthetic dataset. Centralized OCSVM shows the best performance on all metrics. PND-OCSVM has a higher TPR and lower FPR than Local methods and other distributed methods, is closer to Centralized OCSVM. Since AUC and F1-score provide overall evaluations of models, we mainly focus on the two metrics when evaluating different algorithms.

In addition, real-world anomaly detection datasets are used to evaluate these algorithms. The results are reported in

TABLE III
PERFORMANCE COMPARISONS OF DIFFERENT METHODS ON THE SYNTHETIC DATASET.

Methods	TPR	FPR	F1-score	AUC
Centralized	0.9894	0.0011	0.9894	0.9989
Local	0.9716	0.0219	0.8915	0.9945
Local with Neighbors	0.9701	0.0071	0.9522	0.9950
MoM-NDOCSVM	0.9659	0.0112	0.9323	0.9943
doOCSVM	0.9780	0.0102	0.9460	0.9968
doOCSVM*	0.9741	**0.0064**	0.9586	**0.9989**
PND-OCSVM	**0.9894**	0.0072	**0.9616**	**0.9989**

Tab. IV. The average F1 and AUC scores along with their standard deviations are included.

PND-OCSVM is superior to other distributed algorithms, and performs significantly better than Local OCSVM, approaching the performance of Centralized OCSVM. Only doOCSVM* with a large approximate dimension performs slightly better on some datasets than PND-OCSVM, but the communication cost is twice that of PND-OCSVM. In MoM-NDOCSVM, a global matrix is required to enforce consensus on a subset. In doOCSVM, kernel function is replaced by random approximate function. They both lead to larger approximation errors to Centralized OCSVM when compared with PND-OCSVM. When local nodes get access to a small number of training samples such as Cardio dataset, the F1-score of Local OCSVM decreases dramatically. However, PND-OCSVM performs comparably to the Centralized one.

Besides, the standard deviation of F1-scores for each algorithm varies due to the different representations of consensus constraints. Local OCSVM has a relatively large standard deviation, indicating that models trained by local data are distinct. PND-OCSVM shows a smaller standard deviation than Local OCSVM and MoM-NDOCSVM, indicating that it achieves great consensus among nodes. Despite having a slightly larger standard deviation than doOCSVM, PND-OCSVM outperforms doOCSVM, which demonstrates the effectiveness of the projection consensus constraints.

2) Penalty Parameter and Convergence: The influence of the penalty parameter η and convergence of PND-OCSVM are discussed. Consider a network with $J = 20$ and $|\mathcal{B}_j| = 4$. Fig. 2 illustrates the iteration process of the mean and standard deviation of F1-score with different values of η. In limited iterations, larger values of η lead to a faster convergence rate and smaller dispersion of local solutions. However, a very large η may force the algorithm to reach consensus first, which may lead to sharper fluctuations during the iteration. In Fig. 2, the performance of PND-OCSVM closely converges to that of Centralized OCSVM with suitable values of η after 10 iterations. A small performance reduction to Centralized OCSVM is acceptable since the iterations are limited in Fig. 2.

3) The Influence of Network Topology: First, the effect of network connectivity is explored. Consider 40-node networks with different number of neighboring nodes ($|\mathcal{B}_j| = \{2, 4, 6, 8\}$). Fig. 3 shows the corresponding performance on Fraud Dataset.

TABLE IV
AVERAGE RESULTS WITH STD (AVG% ± THE STD%) FOR DIFFERENT ALGORITHMS OVER 20 NODES.

Datasets	KDDCup		Cardio		Backdoor		Fraud	
	F1-score	AUC	F1-score	AUC	F1-score	AUC	F1-score	AUC
Centralized	98.87	99.77	86.87	97.70	90.06	93.21	86.65	94.33
Local	87.44±4.29	99.60±0.14	77.59±5.28	94.33±2.85	78.88±2.87	92.64±0.51	79.96±1.69	93.31±0.64
Local with Neighbors	95.01±1.41	99.72±0.08	82.02±2.50	96.01±0.94	85.71±1.25	92.62±0.49	84.46±0.95	94.03±0.52
MoM-NDOCSVM	92.48±2.88	99.69±0.10	82.16±2.44	96.41±0.48	83.28±1.22	92.90±0.49	84.12±1.47	93.92±0.56
doOCSVM	93.29±0.31	99.73±0.00	82.09±0.20	95.99±0.00	87.91±0.54	92.93±0.02	85.12±0.13	93.96±0.00
doOCSVM*	93.49±0.17	**99.74±0.00**	83.90±0.13	96.48±0.00	**89.25±0.53**	**93.21±0.02**	**85.85±0.25**	**94.31±0.00**
PND-OCSVM	**95.68±1.25**	**99.74±0.00**	**84.51±0.92**	**96.72±0.19**	88.26±1.05	93.09±0.23	85.83±0.91	94.25±0.10

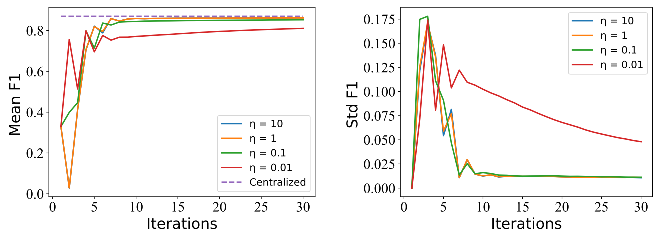
Fig. 2. F1-score of PND-OCSVM with different η on Fraud Dataset.

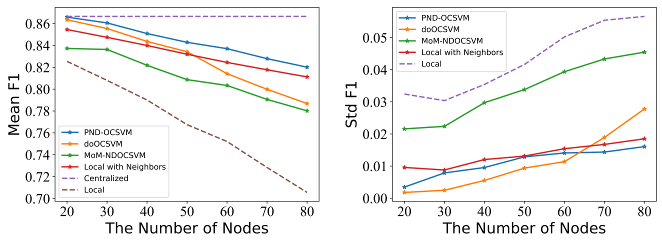
Fig. 4. Comparisons of F1-score for PND-OCSVM, Centralized OCSVM, Local OCSVM, doOCSVM and MoM-NDOCSVM with respect to the number of nodes on Fraud Dataset.

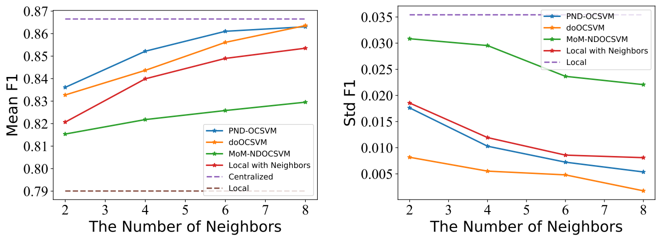
Fig. 3. Comparisons of F1-score for PND-OCSVM, Centralized OCSVM, Local OCSVM, Local OCSVM with Neighbors, doOCSVM and MoM-NDOCSVM with respect to the number of neighbors in a 40-node network on Fraud Dataset.

On the whole, the performance of distributed algorithms improves as the number of neighbors increases. PND-OCSVM outperforms local methods, doOCSVM and MoM-NDOCSVM, and is closer to the Centralized OCSVM. The mean F1-score of PND-OCSVM has a more significant improvement from $|\mathcal{B}_j| = 2$ to $|\mathcal{B}_j| = 4$ and the increasing trend tends to saturate when the number of neighbors is large enough. When $|\mathcal{B}_j| \geq 6$, PND-OCSVM performs closely to the centralized one. This is because the communications among nodes enable local nodes to acquire sufficient global information. Even when network connectivity is relatively poor, PND-OCSVM outperforms local methods and other distributed methods. Besides, PND-OCSVM shows a smaller deviation than MoM-NDOCSVM and local methods, and a slightly higher deviation than doOCSVM.

Then the effect of network size is evaluated. Consider networks with J ($J = \{20, 30, 40, 50, 60, 70, 80\}$) nodes and $|\mathcal{B}_j| = 4$ neighbors. Fig. 4 shows the corresponding performance with different J on Fraud Dataset. Note that the total number of training samples in the network remains unchanged.

In Fig. 4, as the number of nodes increases, the performance of distributed and local algorithms decreases. The decline of local OCSVM is more pronounced since local nodes get access to fewer data that are insufficient to train a suitable model. However, PND-OCSVM outperforms local methods and other distributed methods as network size increases, which proves that PND-OCSVM can achieve excellent performance in large and sparse networks.

4) Communication Cost Comparisons: Consider a network with $J = 20$ and $|\mathcal{B}_j| = 2$. The communication cost of different distributed algorithms are evaluated on Fraud dataset. According to the analysis in Section IV, the communication cost in PND-OCSVM mainly depends on the number of local training samples, whereas in doOCSVM it depends on the size of randomized feature space and in MoM-NDOCSVM it depends on the size of preselected matrix. Therefore, the values of L and D can be varied to obtain different results under different communication cost.

Fig. 5 shows the performance of distributed methods in terms of the communication cost on Fraud dataset. PND-OCSVM achieves superior performance than doOCSVM and MoM-NDOCSVM with lower communication cost. The results demonstrate the effectiveness of PND-OCSVM in improving accuracy and reducing communication cost, thereby demonstrating the applicability of PND-OCSVM in resource-constrained applications.

VI. CONCLUSION

In this work, we proposed a novel projection consensus constraint for kernel OCSVM and developed a fully distributed solver based on ADMM. The proposed algorithm enables local

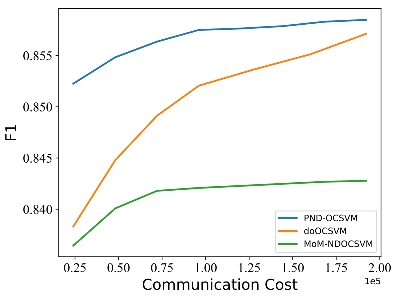

Fig. 5. Communication cost of P-NDPCSVM, doOCSVM and MoM-NDOCSVM for a network with $J = 20$ and $|\mathcal{B}_j| = 2$ on Fraud dataset.

nodes to acquire global information efficiently through the communications only between neighboring nodes.

Compared with other distributed OCSVM algorithms, the proposed method achieves superior performance while reducing communication cost. In the future, we can explore new data representations to replace the raw data exchange between neighboring nodes in order to further protect data privacy or reduce the communication cost.

References

[1] B. Scholkopf, R. Williamson, A. J. Smola, J. Shawe-Taylor, and J. Platt, “Support vector method for novelty detection,” in Proc. Adv. Neural Inf. Process. Syst., vol. 12, no. 3, pp. 582–588, 2000.

[2] L. Maglaras, J. Jiang, and T. Cruz, “Integrated OCSVM mechanism for intrusion detection in SCADA systems,” Electron. Lett., vol. 50, no. 25, pp. 1935–1936, 2014.

[3] V. Mygdalis, A. Iosifidis, A. Tefas, and I. Pitas, “Exploiting subclass information in one-class support vector machine for video summarization,” in Proc. IEEE Int. Conf. Acoust. Speech Signal Process., pp. 2259–2263, 2015.

[4] Y. Guerbai, Y. Chibani, and B. Hadjadji, “The effective use of the one-class SVM classifier for handwritten signature verification based on writer-independent parameters,” Pattern Recognit., vol. 48, no. 1, pp. 103–113, Jan. 2015

[5] Y. Xiao, H. Wang, W. Xu, and J. Zhou, “Robust one-class SVM for fault detection,” Chemometr. Intell. Lab. Syst., vol. 151, pp. 15–25, Feb. 2016.

[6] F. Angiulli, S. Basta, S. Lodi, and C. Sartori, “Distributed strategies for mining outliers in large data sets,” IEEE Trans. Knowl. Data Eng., vol. 25, no. 7, pp. 1520–1532, Jul. 2013.

[7] S. Li, M. Shao, and Y. Fu, “Locality linear fitting one-class SVM with low-rank constraints for outlier detection,” in Proc. Int. Joint Conf. Neural Netw., Jul. 2014, pp. 676–683.

[8] C. Ma, V. Smith, M. Jaggi, M. I. Jordan, P. Richtárik, and M. Takác,ˇ “Adding vs. averaging in distributed primal-dual optimization,” in Proc. 32nd Int. Conf. Mach. Learn., vol. 37, pp. 1973–1982, 2015.

[9] P. A. Forero, A. Cano, and G. B. Giannakis, “Consensus-based distributed support vector machines,” J. Mach. Learn. Res., vol. 11, pp. 1663–1707, May 2010.

[10] A. Rahimi and B. Recht, “Random features for large-scale kernel machines,” in Proc. Adv. Neural Inf. Process. Syst., pp. 1177–1184, 2007.

[11] X. Miao, Y. Liu, H. Zhao, and C. Li, “Distributed online one-class support vector machine for anomaly detection over networks,” IEEE Trans. Cybern., vol. 49, no. 4, pp. 1475-1488, 2019.

[12] W. Wang, Q. Chen, T. Liu, X. He and L. Tang, “A distributed online learning approach to detect anomalies for virtualized network slicing,” IEEE Glob. Commun. Conf., pp. 1-6, 2021.

[13] S. M. Erfani, M. Baktashmotlagh, S. Rajasegarar, S. Karunasekera, and C. Leckie, “R1SVM: A randomised nonlinear approach to large-scale anomaly detection,” in Proc. Assoc. Adv. Artif. Intell., pp. 432–438, 2015.

[14] F. He, R. Yang, L. Shi, and X. huang, “A decentralized framework for kernel PCA with projection consensus constraints,” arXiv preprint arXiv:2211.15953, 2022.

[15] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed optimization and statistical learning via the alternating direction method of multipliers,” Found. Trends Mach. Learn., vol. 3, no. 1, pp. 1–122, Jan. 2011.

[16] Y.-S. Choi, “Least squares one-class support vector machine,” Pattern Recognit. Lett., vol. 30, no. 13, pp. 1236–1240, Oct. 2009.

[17] Y. Liu, B. Zhang, B. Chen, Y. Yang, “Robust solutions to fuzzy one-class support vector machine,” Pattern Recognit. Lett., vol. 71, pp. 73-77, 2016.

[18] H.-J. Xing and W.-T. Liu, “Robust AdaBoost based ensemble of one-class support vector machines,” Inf. Fusion, vol. 55, pp. 45–58, Mar. 2020.

[19] V. Mygdalis, A. Iosifidis, A. Tefas, and I. Pitas, “Graph embedded oneclass classifiers for media data classification,” Pattern Recognit., vol. 60, pp. 585–595, Dec. 2016.

[20] L. Guan, T. Sun, L. Qiao, Z. Yang, D. Li, K. Ge, et al, “An efficient parallel and distributed solution to nonconvex penalized linear SVMs,” Front. Inform. Technol. Electron. Eng., vol. 21, no. 4, pp. 587-603, 2020.

[21] M. Jaggi et al., “Communication-efficient distributed dual coordinate ascent,” in Proc. Adv. Neural Inf. Process. Syst., vol. 2, pp. 3068–3076, 2014.

[22] C.-P. Lee and D. Roth, “Distributed box-constrained quadratic optimization for dual linear SVM,” in Proc. 32nd Int. Conf. Mach. Learn., pp. 987–996, 2015.

[23] Y. Lu, V. Roychowdhury, and L. Vandenberghe, “Distributed parallel support vector machine in strongly connected networks,” IEEE Trans. Neural Netw., vol. 19, no. 7, pp. 1167–1178, Jul. 2008.

[24] A. Navia-Vázquez, D. Gutierrez-Gonzalez, E. Parrado-Hernández, and J. J. Navarro-Abellan, “Distributed support vector machines,” IEEE Trans. Neural Netw., vol. 17, no. 4, pp. 1091–1097, Jul. 2006.

[25] S. Chouvardas and M. Draief, “A diffusion kernel LMS algorithm for nonlinear adaptive networks,” in Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 4164–4168, 2016.

[26] S. Macua, P. Belanovic, and S. Zazo, “Consensus-based distributed principal component analysis in wireless sensor networks,” in Proc. IEEE 11th Int. Workshop Signal Process. Adv. Wireless Commun., pp. 1–5, Jun. 2010.

[27] C. O'Reilly, A. Gluhak, M. A. Imran, “Distributed anomaly detection using minimum volume elliptical principal component analysis,” IEEE Trans. Knowl. Data Eng., vol. 28, no. 9, pp. 2320–2333, Sep. 2016.

[28] D. Dheeru and G. Casey, “UCI machine learning repository,” http://archive.ics.uci.edu/ml

[29] G. Pang, C. Shen, and A. van den Hengel, “Deep anomaly detection with deviation networks,” in Proc. Int. Conf. Knowl. Discovery Data Mining, 2019, pp. 353–362.

[30] N. Moustafa and J. Slay, “UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set),” in Proc. Mil. Commun. Inf. Syst. Conf., 2015, pp. 1–6.