Machine Intelligence Research
www.mi-research.net

19(1), February 2022, 63-74
DOI: 10.1007/s11633-022-1315-6

A Framework for Distributed Semi-supervised Learning Using Single-layer Feedforward Networks

Jin Xie San-Yang Liu Jia-Xi Chen

School of Mathematics and Statistics, Xidian University, Xi′an 710071, China

Abstract: This paper aims to propose a framework for manifold regularization (MR) based distributed semi-supervised learning (DSSL) using single layer feed-forward neural network (SLFNN). The proposed framework, denoted as DSSL-SLFNN is based on the SLFNN, MR framework, and distributed optimization strategy. Then, a series of algorithms are derived to solve DSSL problems. In DSSL problems, data consisting of labeled and unlabeled samples are distributed over a communication network, where each node has only access to its own data and can only communicate with its neighbors. In some scenarios, DSSL problems cannot be solved by centralized algorithms. According to the DSSL-SLFNN framework, each node over the communication network exchanges the initial parameters of the SLFNN with the same basis functions for semi-supervised learning (SSL). All nodes calculate the global optimal coefficients of the SLFNN by using distributed datasets and local updates. During the learning process, each node only exchanges local coefficients with its neighbors rather than raw data. It means that DSSL-SLFNN based algorithms work in a fully distributed fashion and are privacy preserving methods. Finally, several simulations are presented to show the efficiency of the proposed framework and the derived algorithms.

Keywords: Distributed learning (DL), semi-supervised learning (SSL), manifold regularization (MR), single layer feed-forward neural network (SLFNN), privacy preserving.

Citation: J. Xie, S. Y. Liu, J. X. Chen. A framework for distributed semi-supervised learning using single-layer feedforward networks. *Machine Intelligence Research*, vol.19, no.1, pp.63–74, 2022. http://doi.org/10.1007/s11633-022-1315-6

1 Introduction

Artificial neural networks (ANNs) play an important role in machine learning (ML) and perform well in many applications[1−3]. Among many ANNs, the single layer feed-forward neural network (SLFNN) attracts more attention owing to its simple form and excellent performance[4]. Although the SLFNN algorithm is effective, the training of the SLFNN algorithm is complicated because of the large number of parameters. Thus, many researchers focus on feed-forward neural networks with randomized weights (FNNRW). Li and Wang[5] discuss the practical issues and common pitfalls of the random vector functional-link (RVFL) built with a specific randomized algorithm. The input weights and biases of RVFL are randomly assigned and fixed during the training phase. Wang and Li[6] propose a stochastic configuration networks (SCNs) learner model, which is incrementally generated by stochastic configuration (SC) algorithms. The FNNRW attracts attention because of its simple configuration, fast training speed, and good performance in regression and classification tasks. Traditional algorithms based on SLFNN and FNNRW are supervised approaches, where the labels of the training data are known.

In the real world, it is difficult or costly to get labeled data. Therefore, some researchers have tried to make better use of unlabeled data based on labeled data and proposed many semi-supervised learning (SSL) algorithms. The task of SSL is to learn mappings from datasets, including labeled and unlabeled data. Up to now, SSL techniques have been theoretically studied and well developed. Some SSL algorithms are based on the manifold regularization (MR) theory[7−9]. According to the MR framework, training samples are assumed to be distributed on a low-dimension manifold, whose geometry only depends on the input data including labeled and unlabeled data. Correspondingly, many MR-based algorithms have been derived[10−12]. Another kind of SSL algorithm is based on the transductive learning (TL) theory[12−14]. The TL theory is based on the principle of structural risk minimization. Besides, there are also many SSL methods, including generative modeling[15], semi-supervised support vector machine (S3VM)[16], co-training[17], and fuzziness theory[18].

Although SLFNN based supervised learning (SL) and SSL algorithms can be used to solve many problems in practice, there are many scenarios in which data cannot be centrally dealt with. For example, personal information and commercial secrets cannot be shared due to pri-

Research Article
Manuscript received March 15, 2021; accepted August 19, 2021
Recommended by Associate Editor Bin Luo
Colored figures are available in the online version at https://link.springer.com/journal/11633

vacy protection. In addition, nodes in a wireless sensor network (WSN) can only transfer limited information due to bandwidth limitation. Data in these scenarios are separately stored over communication networks and cannot be shared, where traditional centralized algorithms cannot be applied. Thus, several distributed SSL (DSSL) algorithms have been proposed. Avrachenkov et al.[19] propose two asynchronously distributed approaches for graph-based SSL. The first approach is based on stochastic approximation and the second one uses the randomized Kaczmarz algorithm. Chang et al.[20] provide an error analysis for DSSL with kernel ridge regression based on a divide-and-conquer strategy. Shen et al.[21] propose two frameworks for distributed semi-supervised metric learning using the diffusion and alternating direction method of multipliers (ADMM) strategies. Scardapane et al.[22] investigate the problem of learning a semi-supervised support vector machine (SVM) with distributed data over a network of interconnected agents and propose a DSSL algorithm based on the in-network successive convex approximation (NEXT) framework. Fierimonte et al.[23] propose the D-LapRLS algorithm based on the kernel method and distributed average consensus (DAC) strategy. Moreover, they applied two techniques, namely the random projection and the nonlinear transformation, to calculate the global Euclidean distance matrix (EDM) with transformed data. Güler et al.[24] propose a kernel based privacy-aware DSSL algorithm by defining a metric that quantifies the number of candidate samples that are consistent with shared data. Among these kernel-based DSSL algorithms, the D-LapRLS algorithm performs best. However, it needs to calculate the global EDM with respect to total samples, which cost much time. In order to solve DSSL problems and overcome the drawback of kernel based DSSL algorithms, we redesigned the model and proposed the D-LapWNN algorithm in our previous work[25], inspired by [26–28]. It is based on the MR framework and the zero gradient sum (ZGS) strategy. Considering distributed data split by attributes or vertically partitioned data, we proposed the ADMM and MR based DSS-RVFL algorithm in [29].

In this paper, we extend our previous works and novelly propose a DSSL framework using SLFNN and distributed optimization strategies (DOS) such as the ZGS, ADMM, DAC, diffusion least mean square (DLMS), etc. We denote the proposed framework as DSSL-SLFNN. This framework takes advantage of SLFNN and DOS. According to the proposed framework, each node over the communication network is assigned an individual SLFNN with the same structure and exchanges local information with its neighbors. The convergence of the proposed framework is guaranteed by the corresponding DOS. Furthermore, we derive a series of DSSL algorithms from the DSSL-SLFNN framework by using different SLFNNs and DOSs. The task of DSSL-SLFNN is to calculate the globally optimal coefficients of SLFNN in a distributed manner. During the learning process, each node only exchanges local coefficients with its neighbors rather than the raw data. It means that the DSSL-SLFNN framework is able to preserve privacy. Moreover, the learning results of these DSSL-SLFNN based algorithms are proved to be similar to the centralized algorithm using SLFNN with all training samples over the network. The main contribution of this paper is to novelly propose a privacy-preserving DSSL framework, which is based on the MR theory but not necessary to estimate the global EDM.

The contributions of this paper are summarized as follows.

1) This paper proposes a novel DSSL framework using MR and SLFNN, denoted as the DSSL-SLFNN framework, to quickly solve DSSL problems. Then, a series of DSSL algorithms are derived based on the proposed framework.

2) The DSSL-SLFNN framework overcomes the common drawback of the kernel method based DSSL algorithm, which requires to calculate the global EDM with respect to the total data over the communication network.

3) The algorithms derived from the DSSL-SLFNN framework work in a fully distributed fashion and avoids sharing original or sensitive data. It means that the proposed DSSL-SLFNN framework is fully distributed and privacy-preserving.

The rest of this paper is organized as follows. At first, some preliminaries are introduced in Section 2. Then, in Section 3, we formulate the DSSL problem and propose a novel DSSL framework using SLFNN. Section 4 provides some numerical simulations to verify the efficiency of the proposed algorithms. Finally, conclusions are drawn in Section 5.

Notation. $\mathbf{R}$ stands for the real number set. $\boldsymbol{x} \in \mathbf{R}^n$ represents an $n \times 1$ real-value vector. $\boldsymbol{A} \in \mathbf{R}^{n\times n}$ denotes an $n \times n$ size real-value matrix. $\boldsymbol{A}^{\mathrm{T}}$ is the transpose of the matrix $\boldsymbol{A}$. $\|\cdot\|$ stands for the Euclidean norm. ∇f denotes the gradient of the function f.

2 Preliminaries

In this section, some definitions of SLFNN are introduced in Section 2.1. Then, some distributed optimization strategies are introduced in Section 2.2.

2.1 Single layer feed-forward neural networks

In the study of ANN, the research of SLFNN plays a very important role. SLFNN contains many kinds of neural networks such as wavelet neural network (WNN)[27, 30], radial basis function network (RBFN)[31, 32], and random vector functional link network (RVFLN)[12]. The general model of an SLFNN is given in the following form:

$$f(\boldsymbol{x}) = \sum_{l=1}^{L} w_l \psi_l(\boldsymbol{x}) + \epsilon \triangleq \tilde{f}(\boldsymbol{x}) + \epsilon \tag{1}$$

where ψ_l denotes the l-th neuron or basis function, L is the number of hidden neurons, and w_l is the coefficient or weight of neuron l. Coefficients of neurons are trained by using a dataset $\mathcal{S} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$ with $\boldsymbol{x}_i \in \mathbf{R}^d$ and $y_i \in \mathbf{R}, i = 1, \cdots, N$, ϵ represents the training error.

The forms of basis functions in different SLFNN models are quite different. WNN is a class of ANN that combines the classic ANN and wavelet analysis. In the WNN model, $\psi_l(x) = \frac{1}{\sqrt{\boldsymbol{a}_l}}\psi(\frac{\boldsymbol{x} - \boldsymbol{b}_l}{\boldsymbol{a}_l})$ is the wavelet basis function. Here, $\boldsymbol{a}_l \in \mathbf{R}_+$ and $\boldsymbol{b}_l \in \mathbf{R}^d$ are called dilation and translation parameters, respectively.

RBFN is widely used because of its universal approximation and faster learning speed. In RBFN, the basis functions depend on the distance between the input vector and the core vector, which is defined as $\psi_l(\boldsymbol{x}) = \psi(\|\boldsymbol{x} - \boldsymbol{c}_l\|)$. Here, $\boldsymbol{c}_l \in \mathbf{R}^d$ is a reference vector denoted as the l-th center or core.

In the traditional fully connected ANN, the initial values of many parameters have little influence on the accuracy of the algorithm, especially the weight between the input layer and the hidden layer, namely input weight. For FNNRW, $\psi_l(\boldsymbol{x}) = \psi(\boldsymbol{a}_l^{\mathrm{T}}\boldsymbol{x} - \boldsymbol{b}_l)$ with $\boldsymbol{a}_l \in \mathbf{R}^d$ and $\boldsymbol{b}_l \in \mathbf{R}$ are randomly chosen.

The frequently-used basis functions of the SLFNNs mentioned above are expressed in Table 1.

Table 1 Basis functions and the corresponding formulations of different types of SLFNN

Type	Basis function	Formulation
Wavelet	Shannon	$\psi(x) = \frac{\sin(2\pi x) - \sin(\pi x)}{\pi x}$
	Morlet	$\psi(x) = \cos(5x)\mathrm{e}^{-\frac{x^2}{2}}$
	Mexican hat	$\psi(x) = (1 - x^2)\mathrm{e}^{-\frac{x^2}{2}}$
Radial	Gaussian	$\psi(x) = \mathrm{e}^{-\frac{x^2}{2\sigma^2}}$
	Multi quadric	$\psi(x) = (x^2 + \sigma^2)^{\frac{1}{2}}$
	Thin plate spline	$\psi(x) = x^2 \ln(x)$
Other	Sigmoid	$\psi(x) = \frac{1}{1 + \mathrm{e}^{-x}}$
	Sine	$\psi(x) = \sin(x)$

The goal of training an SLFNN is to find proper coefficients to minimize the training error ϵ, which is equivalent to minimizing the following expression:

$$\mathcal{J} = \frac{1}{2}\sum_{i}^{N}(y_i - \tilde{f}(\boldsymbol{x}_i))^2 + \frac{\lambda}{2}\sum_{l=1}^{L}\boldsymbol{w}_l^2 \tag{2}$$

where $\lambda > 0$ is the coefficient of the regularization term.

Expression (2) can be rewritten in the matrix form as

$$\mathcal{J} = \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{H}\boldsymbol{w}\|^2 + \frac{\lambda}{2}\|\boldsymbol{w}\|^2 \tag{3}$$

where $\boldsymbol{y} = [y_1, \cdots, y_N]^{\mathrm{T}}$, $\boldsymbol{w} = [w_1, \cdots, w_L]^{\mathrm{T}}$, and

$$\boldsymbol{H} = \begin{bmatrix} \psi_1(x_1) & \cdots & \psi_L(x_1) \\ \vdots & \ddots & \vdots \\ \psi_1(x_N) & \cdots & \psi_L(x_N) \end{bmatrix}_{N\times L}.$$

Thus, the optimal coefficient vector can be easily obtained by

$$\boldsymbol{w}^* = (\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H} + \boldsymbol{\lambda}\boldsymbol{I_L})^{-1}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{y} \tag{4}$$

where $\boldsymbol{I}_L$ denotes the L order identity matrix.

2.2 Distributed optimization strategies

There are many DOS used for calculating global optimums of optimization problems in distributed scenes, such as distributed average consensus (DAC)[33], zero gradient sum (ZGS)[34], alternating direction method of multipliers (ADMM)[35], diffusion least mean square (DLMS)[36], and such like[22].

1) Distributed average consensus: The DAC strategy is an iterative network protocol that is designed to compute the global average of the local measurement vector[33]. For a V nodes network, the k-th iteration of node i is shown as

$$\boldsymbol{x}_i(k) = \sum_{j=1}^{V}\boldsymbol{C}_{ij}\boldsymbol{x}_j(k-1). \tag{5}$$

It has been proved in [33] that if the network is connected and the connectivity matrix $\boldsymbol{C}$ is chosen appropriately, the iterative sequence (5) converges to the global average. It means that

$$\lim_{k\to+\infty}\boldsymbol{x}_i(k) = \frac{1}{V}\sum_{j=1}^{V}\boldsymbol{x}_j(0), \forall i = 1, 2, \cdots, V. \tag{6}$$

In order to ensure convergence in the case of undirected and connected networks[33], $\boldsymbol{C}$ is given by

$$\mathbf{C}_{ij} = \begin{cases} \frac{1}{d+1}, & \text{if } j \in \mathcal{N}_i \\ 1 - \frac{d_j}{d+1}, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

where d_j represents the degree of node j and d denotes the maximum degree of all nodes in the graph $\mathcal{G}$.

2) Alternating direction method of multipliers: The ADMM strategy proposed in [35] is an optimization procedure, that breaks the optimization problem into subproblems that are easier to handle. It focuses on solving optimization problems with variables $\boldsymbol{x} \in \mathbf{R}^{D_1}$ and $\boldsymbol{z} \in \mathbf{R}^{D_2}$, shown as

$$\begin{cases} \min & f(\boldsymbol{x}) + g(\boldsymbol{z}) \\ \text{s.t.} & \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} + \boldsymbol{c} = \boldsymbol{0} \end{cases} \tag{8}$$

where f and g are convex functions, $\boldsymbol{A} \in \mathbf{R}^{p\times D_1}$, $\boldsymbol{B} \in \mathbf{R}^{p\times D_2}$, and $\boldsymbol{c} \in \mathbf{R}^p$.

To solve the optimization problem (8), we rewrite the expression as the following augmented Lagrangian form:

$$\begin{aligned} \mathcal{F}_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{r}) =& f(\boldsymbol{x}) + g(\boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{A}\boldsymbol{s} + \boldsymbol{B}\boldsymbol{z} + \boldsymbol{c}\|^2 + \\ & \boldsymbol{r}^{\rm T}(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} + \boldsymbol{c}) \end{aligned} \tag{9}$$

where $\rho > 0$ is a tunable parameter and $\boldsymbol{r} \in \mathbf{R}^p$ is the Lagrange multipliers vector.

In order to solve the problem (8), the iteration procedure of the ADMM strategy is given by

$$\begin{cases} \boldsymbol{x}(k+1) = \arg\min\limits_{\boldsymbol{x}}\{\mathcal{F}_\rho(\boldsymbol{x}(k), \boldsymbol{z}(k), \boldsymbol{r}(k))\} \\ \boldsymbol{z}(k+1) = \arg\min\limits_{\boldsymbol{z}}\{\mathcal{F}_\rho(\boldsymbol{x}(k+1), \boldsymbol{z}(k), \boldsymbol{r}(k))\} \\ \boldsymbol{r}(k+1) = \boldsymbol{r}(k) + \rho(\boldsymbol{A}\boldsymbol{x}(k+1) + \boldsymbol{B}\boldsymbol{z}(k+1) + \boldsymbol{c}). \end{cases} \tag{10}$$

3) Zero gradient sum: The ZGS strategy is proposed to solve unconstrained, separable, convex optimization problems over undirected networks with fixed topologies[34]. Consider a non-constrained distributed optimization problem that consists of a series of simple subproblems. The formulation is given by

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} f(\boldsymbol{x}) = \arg\min_{\boldsymbol{x}} \sum\nolimits_{i=1}^{V} f_i(\boldsymbol{x}) \tag{11}$$

where f_i is a twice continuously differentiable and strongly convex function.

In our previous work[25, 27], the ZGS strategy has been developed as the following form:

$$\begin{cases} \nabla f_i(\boldsymbol{x}_i(k+1)) - \nabla f_i(\boldsymbol{x}_i(k)) = \\ \gamma \sum\nolimits_{j\in\mathcal{N}_i} a_{ij}(\boldsymbol{x}_j(k) - \boldsymbol{x}_i(k)) \\ \boldsymbol{x}_i(0) = {\boldsymbol{x}_i}^* \end{cases} \tag{12}$$

where $\gamma > 0$ is a tunable parameter, a_{ij} is the element of the weighted adjacency matrix $\mathcal{A}$ corresponding to the graph $\mathcal{G}$, and $\mathcal{N}_i$ denotes the neighbor index set of node i.

4) Diffusion least mean square: The DLMS strategy is proposed for adaptation and learning over networks[36]. It is used to solve problems like (11). The iteration of the DLMS strategy is given by

$$\begin{cases} \phi_i(k) = \boldsymbol{x}_i(k) - \alpha(k)\nabla f_i(\boldsymbol{x}_i(k)) \\ \boldsymbol{x}_i(k+1) = \sum\nolimits_{j=1}^{V} \boldsymbol{C}_{ij}\phi_j(k) \end{cases} \tag{13}$$

where $\boldsymbol{x}_i(0)$ is usually chosen as $\boldsymbol{0}$ or a random vector and $\alpha(k)$ is a step-size sequence.

Here, $\alpha(k) \in (0,1)$, $\sum_{k=0}^{\infty}\alpha(k) = \infty$, and $\sum_{k=0}^{\infty}\alpha(k)^2 < \infty$. According to [22], $\alpha(k)$ can be chosen as

$$\alpha(k) = \frac{\alpha_0}{(k+1)^\delta} \tag{14}$$

where α_0 and δ are positive tunable parameters.

3 Framework for distributed semi-supervised learning

This section contains three parts. Firstly, we reformulate the DSSL problem in Section 3.1. Then, we propose a novel DSSL framework and derive three DSSL algorithms in Section 3.2.

3.1 Problem formulation

Consider the distributed situation of SSL problems, which is illustrated in Fig. 1. All of the training data, including labeled and unlabeled samples, are separately distributed on V different nodes over the communication network. Here, we only care about undirected and connected networks.

Corresponding to the definition of the dataset in SSL, a dataset in DSSL problems can be described as

$$\mathcal{S} = \bigcup_{i=1}^{V} \mathcal{S}_i = \bigcup_{i=1}^{V} (\mathcal{S}_i^{\mathfrak{l}} \cup \mathcal{S}_i^{\mathfrak{u}})$$

where $\mathcal{S}_i^{\mathfrak{l}} = \{(\boldsymbol{x}_j^{\mathfrak{l}}, y_j^{\mathfrak{l}})\}_{j=1}^{N_i^{\mathfrak{l}}}$ stands for the labeled samples, $\mathcal{S}_i^{\mathfrak{u}} = \{\boldsymbol{x}_j^{\mathfrak{u}}\}_{j=1}^{N_i^{\mathfrak{u}}}$ represents the unlabeled data, and $N_i = N_i^{\mathfrak{l}} + N_i^{\mathfrak{u}}$ denotes the number of training samples of node i, respectively, $i = 1, \cdots, V$.

In the distributed case, the goal of DSSL is to obtain globally optimal parameters using distributed data. However, as shown in Fig. 1, each node has only access to its own data and the parameters transmitted by neighboring nodes. For each node, the SSL problem is just a centralized problem. Thus, the following global loss function is defined by accumulating the local loss function:

$$\mathcal{J} = \sum_{i=1}^{V}\left(\frac{1}{2}\|\boldsymbol{H}_i^{\mathfrak{l}}\boldsymbol{w} - \boldsymbol{y}_i^{\mathfrak{l}}\|^2 + \frac{\lambda_i}{2}\|\boldsymbol{w}\|^2 + \frac{\eta_i}{2}\boldsymbol{w}^{\rm T}\boldsymbol{H}_i^{\rm T}\tilde{\boldsymbol{L}}_i\boldsymbol{H}_i\boldsymbol{w}\right) \tag{15}$$

where λ_i and η_i are positive parameters. For simplicity, we set $\lambda_1 = \cdots = \lambda_V = \lambda$ and $\eta_1 = \cdots = \eta_V = \eta$.

The above loss function can be rewritten as $\mathcal{J} = \sum_{i=1}^{V}\mathcal{J}_i$, where $\mathcal{J}_i$ is the local loss function. For node i, the local loss function $\mathcal{J}_i$ is known. The global loss function $\mathcal{J}$ is a measure of the computational model for all nodes using distributed data. However, it is agnostic for any node.

Since the training data are distributed over the communication network, minimizing the loss function (15) is

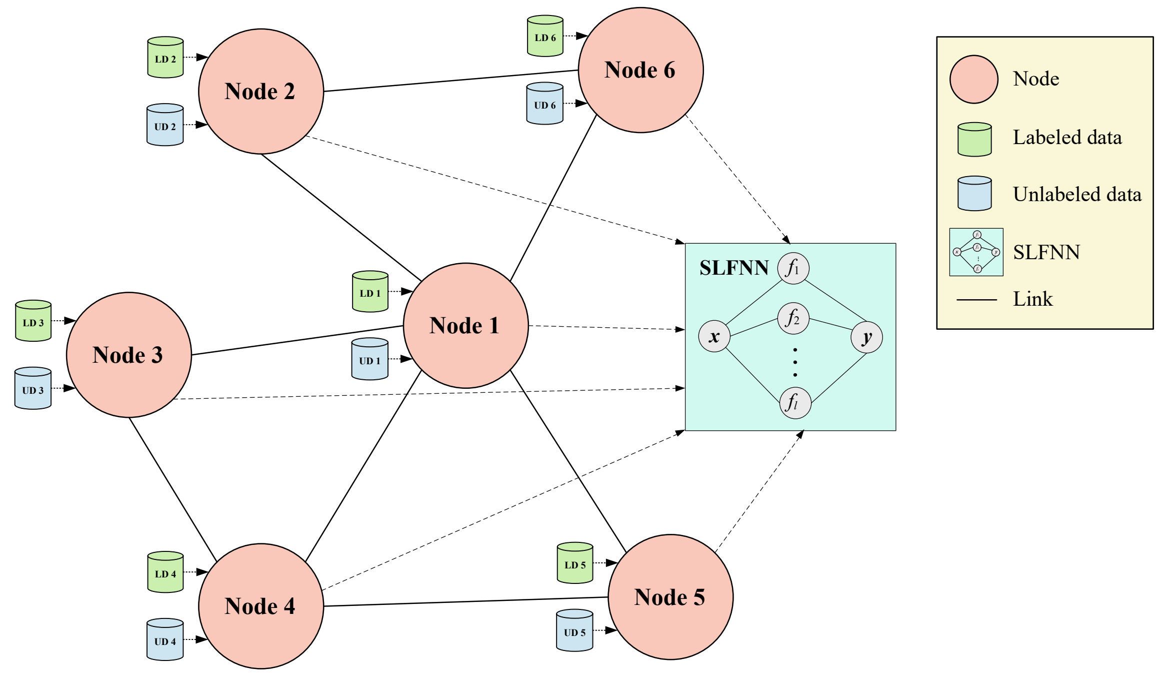


Fig. 1 An illustration of the communication network, where node 1 and node 4 are the neighbors of node 3. Each node is individually assigned an SLFNN with the same basis functions.

equivalent to solving the optimization problem given by

$$\begin{cases} \min \quad \sum_{i=1}^{V}\Big(\frac{1}{2}\|\boldsymbol{H}_i^l\boldsymbol{w}_i-\boldsymbol{y}_i^l\|^2+\frac{\lambda}{2}\|\boldsymbol{w}_i\|^2+ \\ \quad \frac{\eta}{2}\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{H}_i^{\mathrm{T}}\tilde{\boldsymbol{L}}_i\boldsymbol{H}_i\boldsymbol{w}_i\Big), \\ \text{s.t.} \quad \boldsymbol{w}_i=\boldsymbol{w}_j, i,j=1,\cdots,V. \end{cases} \tag{16}$$

The optimization problem (16) cannot be solved directly by traditional centralized methods. Thus, our task is to calculate the global optimal coefficient vector $\boldsymbol{w}^*$ defined as

$$\boldsymbol{w}^*=\arg\min_{\boldsymbol{w}\in\mathbf{R}^L}\mathcal{J}. \tag{17}$$

3.2 DSSL-SLFNN framework and derived algorithms

The optimization problem (17) is a distributed optimization problem, which cannot be directly solved because it contains distributed data. However, it can be solved by using DOSs, such as ZGS, ADMM, DLMS, etc.

When changing the regular terms or norms in problem (17), some DOSs may not work. For example, the ZGS strategy achieves a high convergence speed but requires the local loss function $\mathcal{J}_i$ to be twice continuously differentiable and strongly convex. On the other hand, the ADMM strategy does not require the local loss function to be differentiable, but the calculation is time-consuming.

Therefore, we summarize the existing algorithms and propose a novel DSSL-SLFNN framework on the basis of the SLFNN, MR-based SSL framework, and DOS methods. First, the flow chart of the DSSL-SLFNN framework is illustrated in Fig. 2. Then, we derive it into a series of DSSL algorithms.

1) ZGS based DSSL algorithm: According to our previous work, the global loss function (15) can be regarded as the sum of the local loss function of node i as follows:

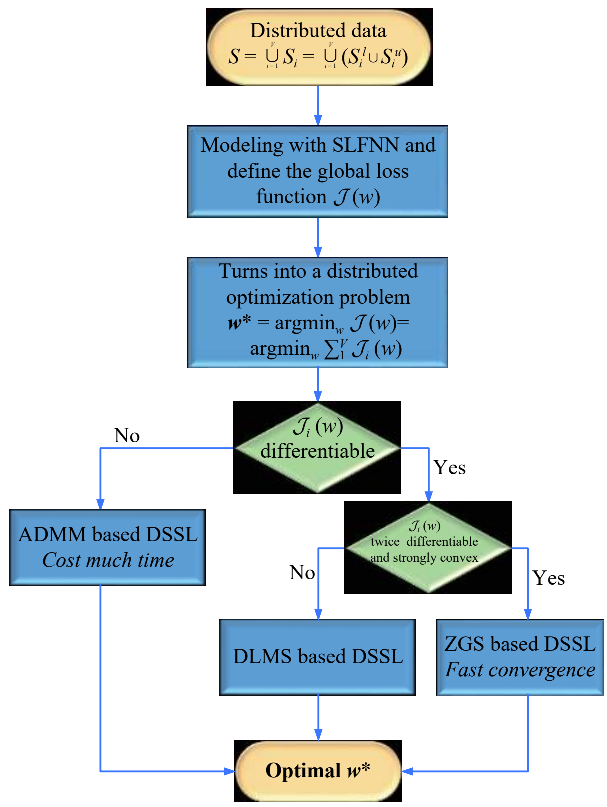


Fig. 2 Flow chart of the proposed DSSL-SLFNN framework

$$\mathcal{J}_i=\frac{1}{2}\left\|\boldsymbol{H}_i^{\mathfrak{l}}\boldsymbol{w}-\boldsymbol{y}_i^{\mathfrak{l}}\right\|^2+\frac{\eta_i}{2}\boldsymbol{w}^{\rm T}\boldsymbol{H}_i^{\rm T}\tilde{\boldsymbol{L}}_i\boldsymbol{H}_i\boldsymbol{w}+\frac{\lambda_i}{2}\|\boldsymbol{w}\|^2 \quad (18)$$

which is a twice continuously differentiable and strongly convex function.

Denoting $\boldsymbol{w}_i(k)$ as $\boldsymbol{w}_i$, the partial derivative of $\mathcal{J}_i(k)$ with respect to $\boldsymbol{w}_i(k)$ is given by

$$\nabla\mathcal{J}_i(\boldsymbol{w}_i)=\left(\boldsymbol{H}_i^{\mathfrak{l}\rm T}\boldsymbol{H}_i^{\mathfrak{l}}+\eta\boldsymbol{H}_i^{\rm T}\tilde{\boldsymbol{L}}_i\boldsymbol{H}_i+\lambda\boldsymbol{I}_{\boldsymbol{L}}\right)\boldsymbol{w}_i-\boldsymbol{H}_i^{\mathfrak{l}\rm T}\boldsymbol{y}_i^{\mathfrak{l}}. \quad (19)$$

Thus, we have the following expression:

$$\nabla\mathcal{J}_i(\boldsymbol{w}_i(k+1))-\nabla\mathcal{J}_i(\boldsymbol{w}_i(k))=$$
$$\left(\boldsymbol{H}_i^{\mathfrak{l}\rm T}+\eta\boldsymbol{H}_i^{\rm T}\tilde{\boldsymbol{L}}_i\boldsymbol{H}_i+\lambda\boldsymbol{I}_L\right)\left(\boldsymbol{w}_i(k+1)-\boldsymbol{w}_i(k)\right). \quad (20)$$

According to the ZGS strategy, the extended algorithm is defined as

$$\begin{cases}\boldsymbol{w}_i(k+1)=\gamma\left(\boldsymbol{H}_i^{\mathfrak{l}\rm T}\boldsymbol{H}_i^{\mathfrak{l}}+\eta\boldsymbol{H}_i^{\rm T}\tilde{\boldsymbol{L}}_i\boldsymbol{H}_i+\lambda\boldsymbol{I}_{\boldsymbol{L}}\right)^{-1}\times\\ \quad\displaystyle\sum_{j\in\mathcal{N}_i}a_{ij}\left(\boldsymbol{w}_j(k)-\boldsymbol{w}_i(k)\right)+\boldsymbol{w}_i(k)\\ \boldsymbol{w}_i(0)=\boldsymbol{w}_i^*.\end{cases} \quad (21)$$

where a_{ij} is an element of the adjacency matrix $\mathcal{A}$.

The pseudo-code of the ZGS-DSSL algorithm is shown in Algorithm 1.

Algorithm 1. ZGS-DSSL

1) Initialize SLFNN and parameters
2) **for** $i\in\mathcal{V}$ **do**
3) Calculate $\boldsymbol{H}_i^{\mathfrak{l}}$, $\boldsymbol{H}_i$ and $\tilde{\boldsymbol{L}}_i$
4) $\boldsymbol{Q}_i\leftarrow\boldsymbol{H}_i^{\mathfrak{l}\rm T}\boldsymbol{H}_i^{\mathfrak{l}}+\eta\boldsymbol{H}_i^{\rm T}\tilde{\boldsymbol{L}}_i\boldsymbol{H}_i+\lambda\boldsymbol{I}_L$
5) $\boldsymbol{w}_i(0)\leftarrow\boldsymbol{Q}_i^{-1}\boldsymbol{H}_i^{\mathfrak{l}\rm T}\mathbf{y}_i^{\mathfrak{l}}$
6) **end**
7) **for** $k\leftarrow 0$ **to** $K-1$ **do**
8) **for** $i\in\mathcal{V}$ **do**
9) $\boldsymbol{w}_i(k+1)\leftarrow\boldsymbol{w}_i(k)+\gamma\,\boldsymbol{Q}_i^{-1}\sum_{j\in\mathcal{N}_i}a_{ij}\left(\boldsymbol{w}_j(k)-\boldsymbol{w}_i(k)\right)$
10) **end**
11) **end**
12) $\boldsymbol{w}^*\leftarrow\boldsymbol{w}_1(K)$
13) **return** $\boldsymbol{w}^*$

Remark 1. According to our previous work[25], the convergence of this kind of algorithm is ensured by the Lyapunov theory. It can be easily derived that $\boldsymbol{w}_i(k)$ converges to $\boldsymbol{w}^*$ if the communication network is undirected and connected, and the parameter $0<\gamma<\min\left\{\frac{\underline{\theta}}{\zeta},\frac{\overline{\Theta}}{\lambda_2}\right\}$. Here, $\lambda_2>0$ is the smallest nonzero eigenvalue of the Laplacian matrix $\mathcal{L}$, $\zeta=\lambda_{\max}(\mathcal{L})$, $\underline{\theta}=\min_{v_i\in\mathcal{V}}\{\lambda_{\min}(\boldsymbol{Q}_i)\}$ and $\overline{\Theta}=\max_{v_i\in\mathcal{V}}\{\lambda_{\max}(\boldsymbol{Q}_i)\}$ with $\boldsymbol{Q}_i=\boldsymbol{H}_i^{\mathfrak{l}\rm T}\boldsymbol{H}_i^{\mathfrak{l}}+\eta\boldsymbol{H}_i^{\rm T}\tilde{\boldsymbol{L}}_i\boldsymbol{H}_i+\lambda\boldsymbol{I}_{\boldsymbol{L}}$.

2) ADMM based DSSL algorithm: According to the ADMM strategy, we rewrite the optimization problem (16) as the following form:

$$\begin{cases}\min\quad\sum_{i=1}^{V}\Big(\frac{1}{2}\left\|\boldsymbol{H}_i^{\mathfrak{l}}\boldsymbol{w}_i-\boldsymbol{y}_i^{\mathfrak{l}}\right\|^2+\frac{\lambda}{2}\|\boldsymbol{w}_i\|^2+\\ \quad\frac{\eta}{2}\boldsymbol{w}_i^{\rm T}\boldsymbol{H}_i^{\rm T}\tilde{\boldsymbol{L}}_i\boldsymbol{H}_i\boldsymbol{w}_i\Big)\\ \text{s.t.}\quad\boldsymbol{w}_i-\boldsymbol{z}=\boldsymbol{0},i=1,\cdots,V.\end{cases} \quad (22)$$

We rewrite the loss function (15) as

$$\mathcal{J}=\sum_{i=1}^{V}\Big(\frac{1}{2}\left\|\boldsymbol{H}_i^{\mathfrak{l}}\boldsymbol{w}_i-\boldsymbol{y}_i^{\mathfrak{l}}\right\|^2+\frac{\eta}{2}\boldsymbol{w}_i^{\rm T}\boldsymbol{H}_i^{\rm T}\tilde{\boldsymbol{L}}_i\boldsymbol{H}_i\boldsymbol{w}_i+$$
$$\frac{\lambda}{2}\|\boldsymbol{z}\|^2+\frac{\gamma}{2}\|\boldsymbol{w}_i-\boldsymbol{z}\|^2+\boldsymbol{r}_i^{\rm T}(\boldsymbol{w}_i-\boldsymbol{z})\Big). \quad (23)$$

According to the ADMM strategy, the optimization problem (23) can be rewritten as

$$\begin{cases}\boldsymbol{w}_i(k+1)=\arg\min\limits_{\boldsymbol{w}_i}\Big\{\frac{1}{2}\left\|\boldsymbol{H}_i^{\mathfrak{l}}\boldsymbol{w}_i-\boldsymbol{y}_i^{\mathfrak{l}}\right\|^2+\frac{\gamma}{2}\|\boldsymbol{w}_i-\boldsymbol{z}(k)\|^2+\\ \quad\boldsymbol{r}_i(k)^{\rm T}(\boldsymbol{w}_i-\boldsymbol{z}(k))+\frac{\eta}{2}\boldsymbol{w}_i^{\rm T}\boldsymbol{H}_i^{\rm T}\tilde{\boldsymbol{L}}_i\boldsymbol{H}_i\boldsymbol{w}_i\Big\}\\ \boldsymbol{z}(k+1)=\arg\min\limits_{\boldsymbol{z}}\Big\{\sum_{i=1}^{V}\boldsymbol{r}_i(k)^{\rm T}(\boldsymbol{w}_i(k+1)-\boldsymbol{z})+\\ \quad\frac{V\lambda}{2}\|\boldsymbol{z}\|^2+\frac{\gamma}{2}\sum_{i=1}^{V}\|\boldsymbol{w}_i(k+1)-\boldsymbol{z}\|^2\Big\}\\ \boldsymbol{r}_i(k+1)=\boldsymbol{r}_i(k)+\gamma\big(\boldsymbol{w}_i(k+1)-\boldsymbol{z}(k+1)\big).\end{cases} \quad (24)$$

Thus, the ADMM based algorithm is described as

$$\begin{cases}\boldsymbol{w}_i(k+1)=\left(\boldsymbol{H}_i^{\mathfrak{l}\rm T}\boldsymbol{H}_i^{\mathfrak{l}}+\eta\boldsymbol{H}_i^{\rm T}\tilde{\boldsymbol{L}}_i\boldsymbol{H}_i+\lambda\boldsymbol{I}_L\right)^{-1}\times\\ \quad\left(\boldsymbol{H}_i^{\mathfrak{l}\rm T}\boldsymbol{y}_i^{\mathfrak{l}}-\boldsymbol{r}_i(k)+\gamma\boldsymbol{z}(k)\right)\\ \boldsymbol{z}(k+1)=\dfrac{1}{\lambda+\gamma}\left(\gamma\bar{\boldsymbol{w}}+\bar{\boldsymbol{r}}\right)\\ \boldsymbol{r}_i(k+1)=\boldsymbol{r}_i(k)+\gamma(\boldsymbol{w}_i(k+1)-\boldsymbol{z}(k+1))\end{cases} \quad (25)$$

where $\boldsymbol{w}_i(0)$ is a random vector, $\bar{\boldsymbol{w}}=\frac{1}{V}\sum_{i=1}^{V}\boldsymbol{w}_i(k+1)$ and $\bar{\boldsymbol{r}}=\frac{1}{V}\sum_{i=1}^{V}\boldsymbol{r}_i(k)$, which can be calculated using the DAC strategy.

The pseudo-code of the proposed ADMM-DSSL algorithm is shown in Algorithm 2.

Algorithm 2. ADMM-DSSL

1) Choose the proper SLFNN and parameters
2) Initialize $\boldsymbol{z}(0)$ and $\boldsymbol{r}_i(0)$ to $\boldsymbol{0}$
3) **for** $i\in\mathcal{V}$ **do**
4) Calculate $\boldsymbol{H}_i^{\mathfrak{l}}$, $\boldsymbol{H}_i$ and $\tilde{\boldsymbol{L}}_i$
5) $\boldsymbol{Q}_i\leftarrow\boldsymbol{H}_i^{\mathfrak{l}\rm T}\boldsymbol{H}_i^{\mathfrak{l}}+\eta\boldsymbol{H}_i^{\rm T}\tilde{\boldsymbol{L}}_i\boldsymbol{H}_i+\lambda\boldsymbol{I}_L$
6) **end**
7) **for** $k\leftarrow 0$ **to** $K-1$ **do**
8) **for** $i\in\mathcal{V}$ **do**
9) $\boldsymbol{w}_i(k+1)\leftarrow\boldsymbol{Q}_i^{-1}\left(\boldsymbol{H}_i^{\rm T}\boldsymbol{C}_i\tilde{\boldsymbol{y}}_i-\boldsymbol{r}_i(k)+\gamma\boldsymbol{z}(k)\right)$
10) **end**
11) $\bar{\boldsymbol{w}}\leftarrow\frac{1}{V}\sum_{i=1}^{V}\boldsymbol{w}_i(k+1)$ using the DAC strategy
12) $\bar{\boldsymbol{r}}\leftarrow\frac{1}{V}\sum_{i=1}^{V}\boldsymbol{r}_i(k)$ using the DAC strategy
13) $\boldsymbol{z}(k+1)\leftarrow\frac{1}{\lambda+\gamma}\left(\gamma\bar{\boldsymbol{w}}+\bar{\boldsymbol{r}}\right)$
14) **for** $i\in\mathcal{V}$ **do**
15) $\boldsymbol{r}_i(k+1)\leftarrow\boldsymbol{r}_i(k)+\gamma(\boldsymbol{w}_i(k+1)-\boldsymbol{z}(k+1))$
16) **end**

17) **if** $\|\boldsymbol{w}(k)-\boldsymbol{z}(k)\|<\epsilon$ **then**
18) **break**
19) **end**
20) **end**
21) $\boldsymbol{w}^* \leftarrow \boldsymbol{z}(K)$
22) **return** $\boldsymbol{w}^*$

Remark 2. Compared with the ZGS-DSSL algorithm, the ADMM-DSSL algorithm consumes more time owing to two extra DAC steps. However, the ADMM strategy has a low limit on the sub-optimization function, namely convex rather than twice continuously differentiable and strongly convex. From this point of view, the ADMM-DSSL algorithm is more flexible and ordinary.

3) DLMS based DSSL algorithm: Similar to the definitions in the ZGS based DSSL algorithms, the local loss function and its derivation are described in expressions (18) and (19), respectively. When we denote $\alpha(k)=\dfrac{\alpha_0}{(k+1)^\delta}$ and $\delta=1$ in the DLMS strategy expressed in (13), the DLMS based DSSL algorithm is given by

$$\begin{cases}\phi_i(k)=\boldsymbol{w}_i(k)-\dfrac{\alpha_0}{k+1}\Big(\big(\boldsymbol{H}_i^{\text{lT}}\boldsymbol{H}_i^{\text{l}}+\eta\boldsymbol{H}_i^{\text{T}}\tilde{\boldsymbol{L}}_i\boldsymbol{H}_i+\\ \quad\lambda\boldsymbol{I}_L\big)\boldsymbol{w}_i(k)-\boldsymbol{H}_i^{\text{lT}}\boldsymbol{y}_i^{\text{l}}\Big)\\ \boldsymbol{w}_i(k+1)=\phi_i(k)+\dfrac{1}{1+d}\sum_{j\in\mathcal{N}_i}\big(\phi_j(k)-\phi_i(k)\big)\end{cases} \tag{26}$$

where $\boldsymbol{w}_i(0)=\boldsymbol{0}$.

The pseudo-code of the DLMS-DSSL framework is shown in Algorithm 3.

Algorithm 3. DLMS-DSSL
1) Choose the proper SLFNN and parameters
2) Initialize $\boldsymbol{w}(0)$ to $\boldsymbol{0}$
3) **for** $i\in\mathcal{V}$ **do**
4) Calculate $\boldsymbol{H}_i^{\text{l}}$, $\boldsymbol{H}_i$ and $\tilde{\boldsymbol{L}}_i$
5) $\boldsymbol{Q}_i \leftarrow \boldsymbol{H}_i^{\text{lT}}\boldsymbol{H}_i^{\text{l}}+\eta\boldsymbol{H}_i^{\text{T}}\tilde{\boldsymbol{L}}_i\boldsymbol{H}_i+\lambda\boldsymbol{I}_L$
6) **end**
7) **for** $k\leftarrow 0$ **to** $K-1$ **do**
8) **for** $i\in\mathcal{V}$ **do**
9) $\phi_i(k)\leftarrow\boldsymbol{w}_i(k)-\dfrac{\alpha_0}{k+1}\big(\boldsymbol{Q}_i\boldsymbol{w}_i(k)-\boldsymbol{H}_i^{\text{lT}}\boldsymbol{y}_i^{\text{l}}\big)$
10) $\boldsymbol{w}_i(k+1)\leftarrow\phi_i(k)+\dfrac{1}{1+d}\sum_{j\in\mathcal{N}_i}\big(\phi_j(k)-\phi_i(k)\big)$
11) **end**
12) **end**
13) $\boldsymbol{w}^*\leftarrow\boldsymbol{w}_1(K)$
14) **return** $\boldsymbol{w}^*$

Remark 3. Compared with the ZGS-DSSL algorithm, the requirements for optimization functions of the DLMS-DSSL algorithm are more flexible, but the convergence speed is lower. In contrast, compared with the ADMM-DSSL algorithm, the DLMS-DSSL costs more time. Therefore, the DSSL algorithm should be selected reasonably according to the actual situation.

4 Simulations

In this section, we will apply the algorithms proposed in this paper to different datasets, introduced in Table 2, to verify the effectiveness of these algorithms. First, some simulation results using the proposed algorithms are shown in Section 4.1. Then, we analyze the parameters of these DSSL algorithms in Section 4.2.

The training data consisting of labeled and unlabeled samples are averagely allocated to each node over the

Table 2 Description of the datasets used in simulations

Dataset	Instance	Feature	Type	Source
2-Moon	10 000	2	Cla	Artificial
SinC	10 000	1	Reg	Artificial
Concrete	1 027	8	Reg	UCI repository
WDBC	569	30	Cla	UCI repository

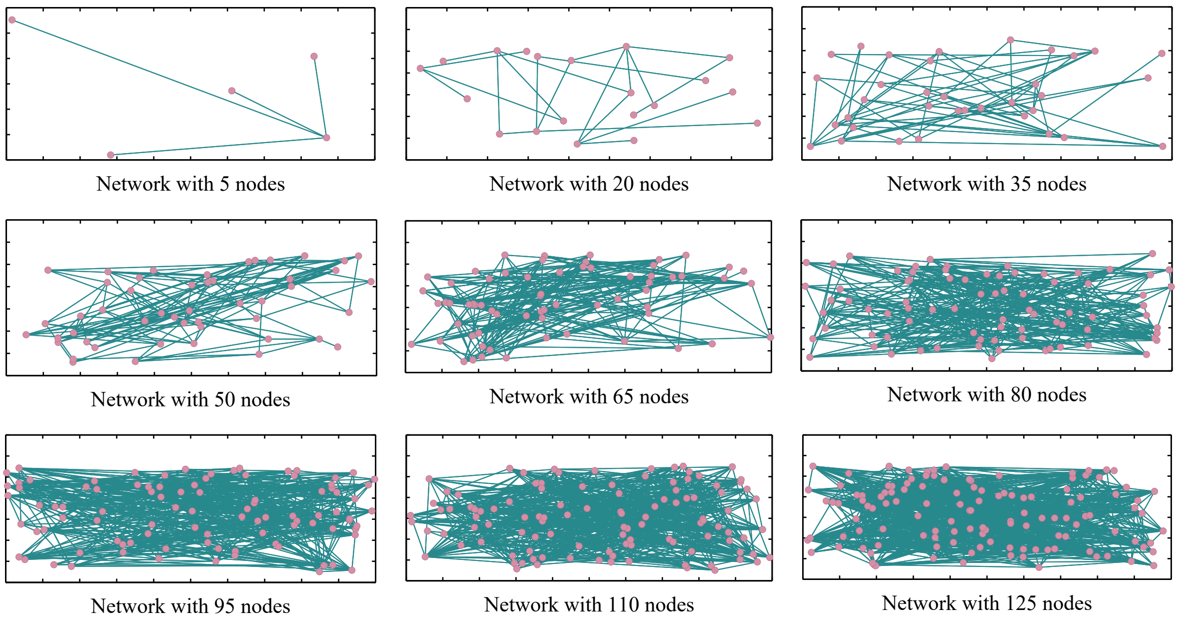


Fig. 3 Topologies for communication networks of different sizes, where the number of nodes ranges from 5 to 125

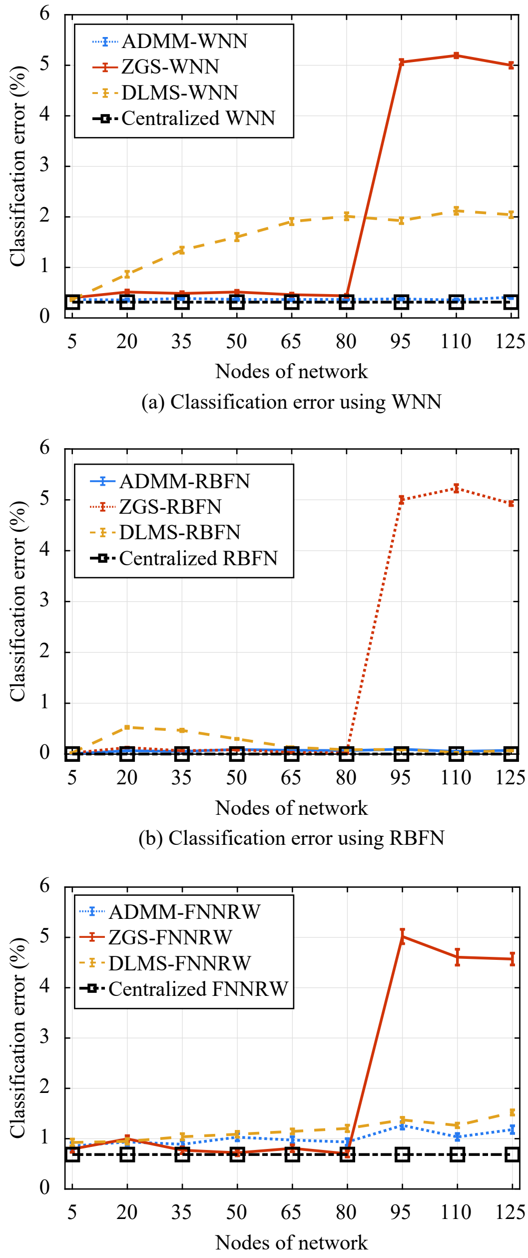
communication networks. For simplicity, we only consider the connected and undirected networks with different kinds of topologies shown in Fig. 3.

The proposed framework is used to solve distributed problems, whose goal is to achieve the effect of centralized learning. Therefore, it focuses on how to realize distributed semi-supervised learning rather than specific SLFNN. Moreover, all the following simulations are repeated ten times for verification.

4.1 Simulation results

The derived DSSL algorithms are applied to the datasets illustrated in Table 2 over the communication networks with different topologies shown in Fig. 3. The simulation results are shown in Figs. 4–7. In addition, we list the results running in the 5-node network from Fig. 4 and compare the derived DSSL algorithms with the centralized SLFNNs, LapRLS, and D-LapRLS algorithms, listed in Table 3. The parameters used in these algorithms are listed in Table 4.

The simulation results show that the proposed DSSL-SLFNN framework and the derived DSSL algorithms are efficient enough in many applications with different datasets. Fig. 4 shows that the proposed DSSL algorithms achieve the similar accuracy compared with the centralized SLFNNs. Furthermore, the ZGS based DSSL algorithms perform worse compared with the other two algorithms. Table 3 shows that the kernel-based algorithms and centralized SSL algorithms using different SLFNNs are time-consuming due to the large-scale Laplacian matrix calculation.



(a) Classification error using WNN

(b) Classification error using RBFN

(c) Classification error using FNNRW

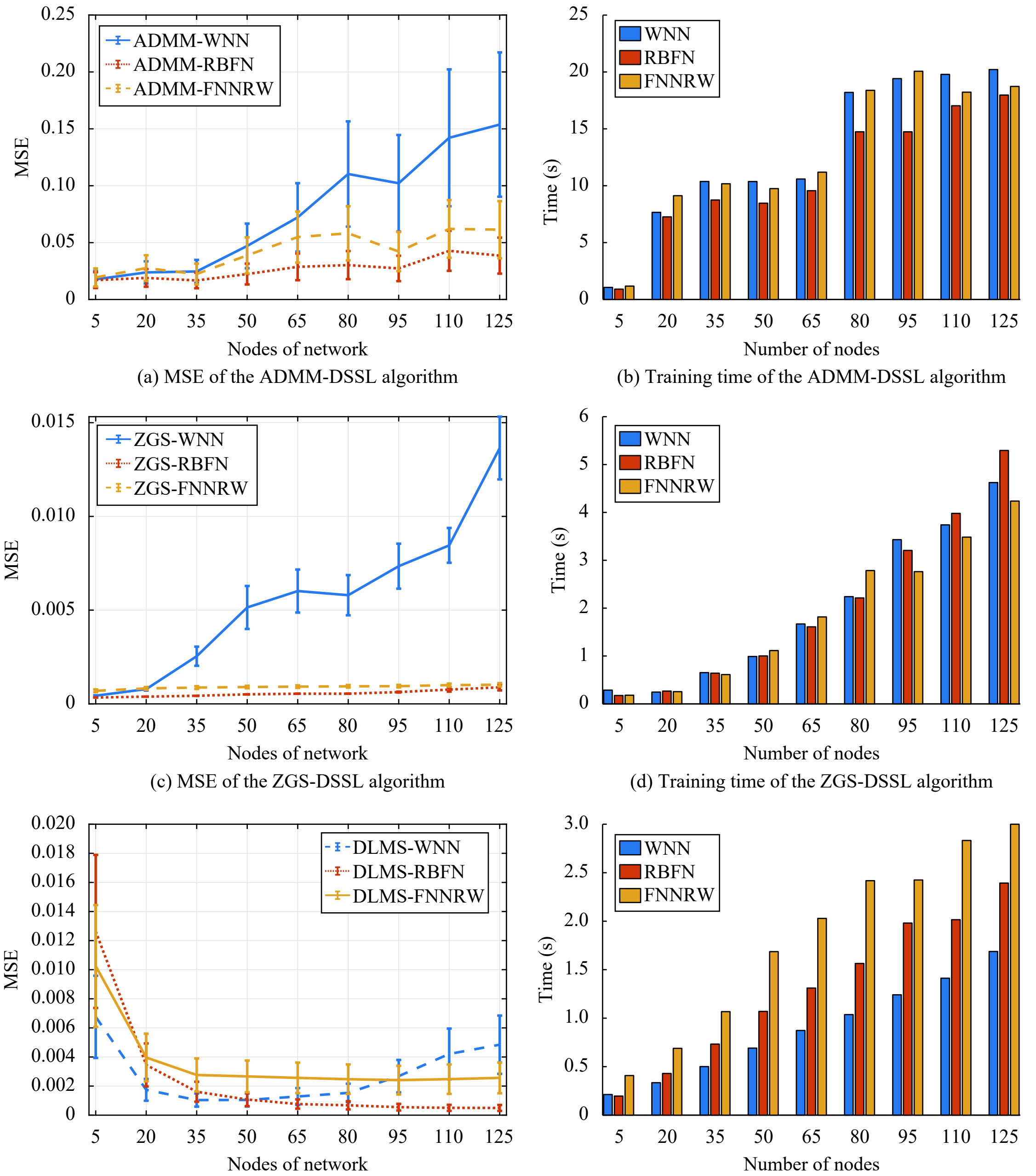
Fig. 4 Classification error of the derived algorithms with the dataset of "2-Moon" using different SLFNNs

Table 3 Classification error and training time of the dataset "2-Moon" using the proposed DSSL algorithms and the D-LapRLS algorithm on the 5 nodes communication network shown in Fig. 3

SLFN	DOS	Classification error	Time (s)
WNN	ADMM	3.65E−02(±2.18E−02)	1.41E+00(±3.37E−01)
	ZGS	3.99E−02(±2.55E−02)	4.24E−02(±6.45E−03)
	DLMS	3.64E−02(±2.23E−02)	8.18E−02(±2.54E−02)
	Central	3.13E−02(±2.03E−02)	4.12E+02(±5.22E+01)
RBFN	ADMM	8.90E−04(±8.42E−04)	1.93E+00(±4.49E−01)
	ZGS	2.25E−03(±2.57E−03)	4.48E−02(±3.99E−03)
	DLMS	2.74E−03(±3.68E−03)	4.83E−01(±7.66E−02)
	Central	2.10E−04(±2.07E−04)	4.09E+02(±7.95E+01)
FNNRW	ADMM	**8.90E−04(±8.42E−04)**	2.07E+00(±4.23E−01)
	ZGS	7.94E−02(±6.39E−02)	4.68E−02(±3.72E−03)
	DLMS	9.25E−02(±6.71E−02)	3.89E−01(±5.42E−02)
	Central	6.85E−02(±6.33E−02)	4.14E+02(±6.65E+01)
Kernel	D-LapRLS	1.05E−03(±1.48E−03)	**8.27E+02(±7.26E+01)**
	LapRLS	1.05E−03(±1.48E−03)	4.17E+02(±2.57E+01)

Table 4 Parameters used in the derived DSSL algorithms in the experiment section

	SSL	SSL	ADMM	ZGS	DLMS
Dataset	λ	η	γ	γ	α_0
SinC	10^{-3}	10^{-4}	10^{-3}	10^{-4}	10^{-3}
2-Moon	10^{-4}	10^{-5}	10^{-3}	10^{-5}	10^{-2}
Concrete	10^{-4}	10^{-5}	10^{-4}	10^{-4}	10^{-3}
WDBC	10^{-3}	10^{-4}	10^{-3}	10^{-4}	10^{-2}



(a) MSE of the ADMM-DSSL algorithm
(b) Training time of the ADMM-DSSL algorithm
(c) MSE of the ZGS-DSSL algorithm
(d) Training time of the ZGS-DSSL algorithm
(e) MSE of the DLMS-DSSL algorithm
(f) Training time of the DLMS-DSSL algorithm

Fig. 5 Performance of the derived algorithms with the dataset of "SinC"

4.2 Parameter analyses

In this part, we analyze the influence of the tunable parameters on the performance of the proposed algorithms. The proposed algorithms in this paper use many parameters including common parameters like λ, η, and specific parameters like γ, ρ, and α_0. For convenience, these parameters are treated as global parameters in this paper shared among the communication network nodes. The cost of parameter exchange is avoided, which can simplify the implementation of the proposed algorithms.

The influence of parameters γ and ρ on the ZGS based and ADMM based DSSL algorithms have been analyzed similarly in [25] and [29], respectively. Thus, we will not analyze these two parameters repeatedly. As for the parameter α_0 used in the DLMS-DSSL algorithm, we set the value from 10^{-5} to 1 and applied it to different communication networks. The result is shown in Fig. 8. Then, we analyzed the influence of the accuracy of the centralized SSL algorithms on the parameters λ and η. The results, shown in Fig. 9, imply that the SSL algorithms perform better with the decrease of λ and η.

In order to analyze the influence of the ratio of labeled

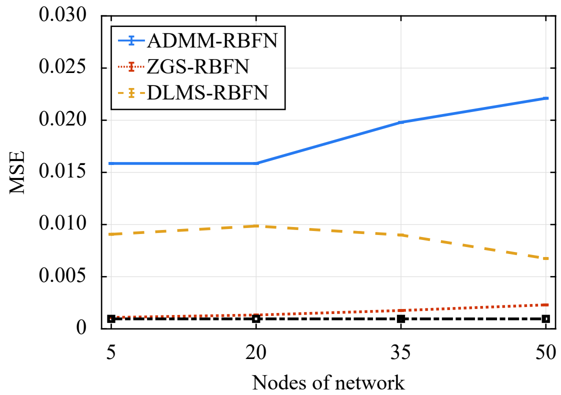


Fig. 6 MSE of regression on the "Concrete" dataset using the proposed DSSL algorithms over the communication networks with different scale

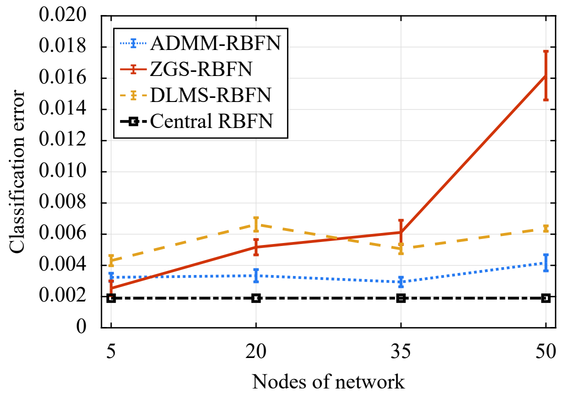


Fig. 7 Classification error of the proposed DSSL algorithms using the "WDBC" dataset

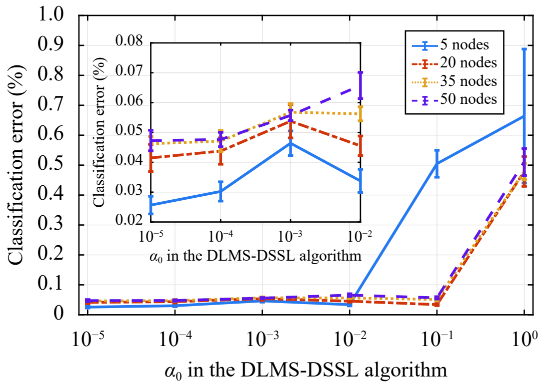


Fig. 8 Influence of initial values of the parameter α_0 on the performance of the DLMS-DSSL algorithm using the "WDBC" dataset

data on the performance of the centralized and proposed distributed SSL algorithms, we applied the proposed algorithms on the benchmark datasets with different labeled samples by setting a different ratio of labeled samples. The results are shown in Fig. 10. As the results shown, all centralized and distributed SSL algorithms perform better when the ratio of labeled samples is increased. However, when the labeled samples are increased to a certain proportion, the improvement of the SSL algorithms will become inconspicuous.

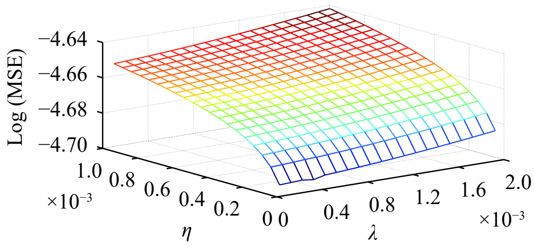


Fig. 9 Log (MSE) of regression on the "Concrete" dataset using the centralized RBFN, varying the parameters λ and η during the iteration process.

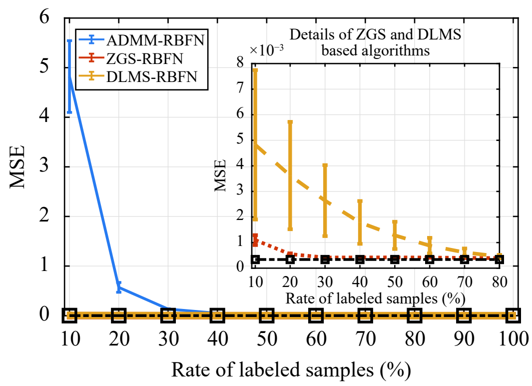


Fig. 10 Influence of the ratio of labeled samples on the performance of SSL algorithms using the "Concrete" dataset

5 Conclusions

This paper proposes a novel DSSL framework for solving SSL problems in distributed scenes, denoted as the DSSL-SLFNN framework. By decomposing a DSSL problem into a series of subproblems on SSL with consensus constraints, we reformulate DSSL problems as distributed optimization problems that DOS can solve over communication networks.

Then, we derive a series of DSSL algorithms from the DSSL-SLFNN framework. According to these DSSL algorithms, each node over the communication network shares the SLFNN with the initial parameters and has only access to its own data. Besides, these nodes can exchange local information iteratively. During the learning process of the proposed DSSL algorithms, the estimated coefficients of the local SLFNN are exchanged between neighbors at every iteration.

Finally, some future work is to generalize the communicating networks into the case of directed or time-varying networks, which are more practical in applications. Another important task to perform is to design finite-time distributed learning algorithms.

Acknowledgments

This work was supported by National Natural Science Foundation of China (Nos. 61877047, 61877046, 62106186 and 62063031), and the Fundamental Research Funds for the Central Universities (Nos. JB210701 and JB210718).

References

[1] W. Jia, J. Gao, W. Xia, Y. Zhao, H. Min, J. T. Lu. A performance evaluation of classic convolutional neural networks for 2D and 3D palmprint and palm vein recognition. *International Journal of Automation and Computing*, vol. 18, no. 1, pp. 18–44, 2021. DOI: 10.1007/s11633-020-1257-9.

[2] X. B. Fu, S. L. Yue, D. Y. Pan. Camera-based basketball scoring detection using convolutional neural network. *International Journal of Automation and Computing*, vol. 18, no. 2, pp. 266–276, 2021. DOI: 10.1007/s11633-020-1259-7.

[3] K. Aukkapinyo, S. Sawangwong, P. Pooyoi, W. Kusakunniran. Localization and classification of rice-grain images using region proposals-based convolutional neural network. *International Journal of Automation and Computing*, vol. 17, no. 2, pp. 233–246, 2020. DOI: 10.1007/s11633-019-1207-6.

[4] T. Matias, F. Souza, R. Araujo, C. H. Antunes. Learning of a single-hidden layer feedforward neural network using an optimized extreme learning machine. *Neurocomputing*, vol. 129, pp. 428–436, 2014. DOI: 10.1016/j.neucom.2013.09.016.

[5] M. Li, D. H. Wang. Insights into randomized algorithms for neural networks: Practical issues and common pitfalls. *Information Sciences*, vol. 382-383, pp. 170–178, 2017. DOI: 10.1016/j.ins.2016.12.007.

[6] D. H. Wang, M. Li. Stochastic configuration networks: Fundamentals and algorithms. *IEEE Transactions on Cybernetics*, vol. 47, no. 10, pp. 3466–3479, 2017. DOI: 10.1109/TCYB.2017.2734043.

[7] M. Belkin, P. Niyogi. Semi-supervised learning on Riemannian manifolds. *Machine Learning*, vol. 56, no. 1–3, pp. 209–239, 2004. DOI: 10.1023/B:MACH.0000033120.25363.1e.

[8] M. Belkin, P. Niyogi, V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, vol. 7, pp. 2399–2434, 2006.

[9] O. Chapelle, B. Scholkopf, A. Zien. *Semi-supervised Learning*, Cambridge, USA: MIT Press, 2006.

[10] S. Melacci, M. Belkin. Laplacian support vector machines trained in the primal. *Journal of Machine Learning Research*, vol. 12, pp. 1149–1184, 2011.

[11] Z. Q. Qi, Y. J. Tian, Y. Shi. Laplacian twin support vector machine for semi-supervised classification. *Neural Networks*, vol. 35, pp. 46–53, 2012. DOI: 10.1016/j.neunet.2012.07.011.

[12] S. Scardapane, D. Comminiello, M. Scarpiniti, A. Uncini. A semi-supervised random vector functional-link network based on the transductive framework. *Information Sciences*, vol. 364–365, pp. 156–166, 2016. DOI: 10.1016/j.ins.2015.07.060.

[13] S. R. Sain. The nature of statistical learning theory. *Technometrics*, vol. 38, no. 4, Article number 409, 1996.

[14] J. Wang, X. Shen. On transductive support vector machines. *Prediction & Discovery*, vol. 27, no. 6, pp. 1463–1462, 2007.

[15] D. P. Kingma, D. J. Rezende, S. Mohamed, M. Welling. Semi-supervised learning with deep generative models. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, MIT Press, Cambridge, USA, pp. 3581−3589, 2014.

[16] Y. F. Li, J. T. Kwok, Z. H. Zhou. Cost-sensitive semi-supervised support vector machine. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, AAAI Press, Atlanta, USA, pp. 500−505, 2010.

[17] M. F. A. Hady, F. Schwenker, G. Palm. Semi-supervised learning for tree-structured ensembles of RBF networks with Co-training. *Neural Networks*, vol. 23, no. 4, pp. 497–509, 2010. DOI: 10.1016/j.neunet.2009.09.001.

[18] T. Miyato, S. I. Maeda, M. Koyama, S. Ishii. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1979–1993, 2018. DOI: 10.1109/TPAMI.2018.2858821.

[19] K. Avrachenkov, V. S. Borkar, K. Saboo. Distributed and asynchronous methods for semi-supervised learning. In *Proceedings of the 13th International Workshop on Algorithms and Models for the Web-Graph*, Springer, Montreal, Canada, pp. 34−46, 2016. DOI: 10.1007/978-3-319-49787-7_4.

[20] X. Y. Chang, S. B. Lin, D. X. Zhou. Distributed semi-supervised learning with kernel ridge regression. *Journal of Machine Learning Research*, vol. 18, no. 1, pp. 1493–1514, 2017.

[21] P. C. Shen, X. Du, C. G. Li. Distributed semi-supervised metric learning. *IEEE Access*, vol. 4, pp. 8558–8571, 2016. DOI: 10.1109/ACCESS.2016.2632158.

[22] S. Scardapane, R. Fierimonte, P. Di Lorenzo, M. Panella, A. Uncini. Distributed semi-supervised support vector machines. *Neural Networks*, vol. 80, pp. 43–52, 2016. DOI: 10.1016/j.neunet.2016.04.007.

[23] R. Fierimonte, S. Scardapane, A. Uncini, M. Panella. Fully decentralized semi-supervised learning via privacy-preserving matrix completion. *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 11, pp. 2699–2711, 2017. DOI: 10.1109/TNNLS.2016.2597444.

[24] B. Güler, A. S. Avestimehr, A. Ortega. Privacy-aware distributed graph-based semi-supervised learning. In *Proceedings of the 29th International Workshop on Machine Learning for Signal Processing*, IEEE, Pittsburgh, USA, pp. 1–6, 2019. DOI: 10.1109/MLSP.2019.8918797.

[25] J. Xie, S. Y. Liu, H. Dai. A distributed semi-supervised learning algorithm based on manifold regularization using wavelet neural network. *Neural Networks*, vol. 118, pp. 300–309, 2019. DOI: 10.1016/j.neunet.2018.10.014.

[26] W. Ai, W. S. Chen, J. Xie. A zero-gradient-sum algorithm for distributed cooperative learning using a feedforward neural network with random weights. *Information Sciences*, vol. 373, pp. 404–418, 2016. DOI: 10.1016/j.ins.2016.09.016.

[27] J. Xie, W. S. Chen, H. Dai. Distributed cooperative learning algorithms using wavelet neural network. *Neural Com-*

puting and Applications, vol. 31, no. 4, pp. 1007–1021, 2019. DOI: 10.1007/s00521-017-3134-1.

[28] J. X. Chen, J. M. Li. Global FLS-based consensus of stochastic uncertain nonlinear multi-agent systems. *International Journal of Automation and Computing*, vol. 18, no. 5, pp. 826−837. DOI: 10.1007/s11633-021-1279-y.

[29] J. Xie, S. Y. Liu, H. Dai, Y. Rong. Distributed semi-supervised learning algorithms for random vector functional-link networks with distributed data splitting across samples and features. *Knowledge-Based Systems*, vol. 195, Article number 105577, 2020. DOI: 10.1016/j.knosys.2020.105577.

[30] Y. H. Chen, B. Yang, J. W. Dong. Time-series prediction using a local linear wavelet neural network. *Neurocomputing*, vol. 69, no. 4−6, pp. 449–465, 2006. DOI: 10.1016/j.neucom.2005.02.006.

[31] D. S. Broomhead, D. Lowe. Radial basis functions, multi-variable functional interpolation and adaptive networks. *Advances in Neural Information Processing Systems*, vol. 4148, pp. 728–734, 1988.

[32] K. Z. Mao. RBF neural network center selection based on Fisher ratio class separability measure. *IEEE Transactions on Neural Networks*, vol. 13, no. 5, pp. 1211–1217, 2002. DOI: 10.1109/TNN.2002.1031953.

[33] R. Olfati-Saber, J. A. Fax, R. M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007. DOI: 10.1109/JPROC.2006.887293.

[34] J. Lu, C. Y. Tang. Zero-gradient-sum algorithms for distributed convex optimization: The continuous-time case. *IEEE Transactions on Automatic Control*, vol. 57, no. 9, pp. 2348–2354, 2012. DOI: 10.1109/TAC.2012.2184199.

[35] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends ® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011. DOI: 10.1561/2200000016.

[36] A. H. Sayed. Adaptive networks. *Proceedings of the IEEE*, vol. 102, no. 4, pp. 460–497, 2014. DOI: 10.1109/JPROC.2014.2306253.

Jin Xie received the B. Sc. degree in mathematics from North China Electric Power University, China in 2013, and received the Ph. D. degree in mathematics from Xidian University, China in 2020. He is currently a lecturer with School of Mathematics and Statistics, Xidian University, China.

His research interests include machine learning, neural networks, semi-supervised learning, and distributed learning.

E-mail: xj6417@126.com

ORCID iD: 0000-0001-8007-5683

San-Yang Liu received the B. Sc. degree in applied mathematics from Shaanxi Normal University, China in 1982, received the M. Sc. degree in applied mathematics from Xidian University, China in 1984, and received the Ph. D. degree in mathematics from Xi′an Jiao Tong University, China in 1989. Currently, he is the Director of Institute of Industrial and Applied Mathematics, Director of the Center for Mathematics and Cross Science of Xidian University, China.

His research interests include optimization methods and their applications, nonlinear analysis, and system modeling.

E-mail: liusanyang@126.com

Jia-Xi Chen received the M. Sc. and Ph. D. degrees in applied mathematics from Xidian University, China in 2018 and 2020, respectively. He is currently a lecturer at Department of Applied Mathem- atics, Xidian University, China.

His research interests include adaptive control, multi-agent systems and Takagi-Sugeno (T-S) fuzzy systems.

E-mail: jxchen208@163.com (Corresponding author)

ORCID iD: 0000-0002-9667-811X