Neurocomputing 000 (2017) 1–9

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Fast convergent distributed cooperative learning algorithms over networks

Yanfei Song, Weisheng Chen*, Hao Dai

School of Aerospace Science and Technology, Xidian University, Xi'an 710071, China

ARTICLE INFO

Article history:
Received 11 April 2017
Revised 23 October 2017
Accepted 30 October 2017
Available online xxx

Communicated by Prof. Dianhui Wang

Keywords:
Finite-time convergence
Distributed cooperative learning (DCL)
Lyapunov method
Feedforward neural networks
Random weights

ABSTRACT

This paper aims to design a fast convergent distributed cooperative learning (DCL) algorithm for feedforward neural networks with random weights (FNNRWs) over undirected and connected networks. First, a continuous-time fast convergent DCL algorithm is proposed, whose finite-time convergence is guaranteed based on the Lyapunov method. Second, we extend this algorithm to a discrete-time form by using the fourth-order Runge–Kutta method. Compared with the distributed alternating direction method of multipliers (ADMM) and the Zero-Gradient-Sum-based (ZGS-based) algorithms, the proposed algorithm has high learning capability and convergence speed. Simulation results demonstrate that the proposed algorithm has fast convergence rate, and the convergence rate may be adjusted by properly selecting some tuning parameters.

1. Introduction

The topic of learning in distributed environments has received increased attention in recent years [1–6]. With the advent of large data, new research in the field of large-scale machine learning has been widely concerned. Due to resource constraints, the large-scale data have to be scattered storage. In addition, the traditional centralized data processing could not satisfy the requirement of privacy and confidentiality in some cases. Therefore, the distributed machine learning problem is worthy of study.

In this paper, we aim to study a distributed machine learning problem in an undirected and connected communication network based on the feedforward neural network with random weights (FNNRWs). This problem is a type of distributed in-network data processing problem, where training data are gathered from a set of agents connected in a network, and each agent only has access to local information without the involvement of any centralized coordination. Researchers have been committed to finding a better way of cooperation for distributed cooperative learning (DCL) problems for the in-network data processing in a distributed manner. Several approaches are proposed to these problems including the use of alternating direction method of multipliers (ADMM) strategies [7,8], the distributed average consensus (DAC) strategies [7,9], the diffusion least-mean square (LMS) strategies [10,11], and so on.

In general, solving distributed machine learning problem is more challenging than solving traditional centralized machine learning problem. It is difficult to obtain a global information in a distributed manner when training data in the network cooperatively find an identical but un-known pattern only by sharing learned knowledge with their neighboring nodes. Due to this, distributed large-scale machine learning strategies need to be designed to model the communication between agents. In [12], the authors developed a DCL algorithm with ADMM optimization procedure for Echo State Networks, and it can solve the global problem in asymptotic convergence rate. Two algorithms for Random Vector Functional-link (RVFL) were proposed in [7] by using the DAC and the ADMM strategies, respectively. However, from the comparisons results with two strategies we can get the DCL algorithm with DAC-based performs better than the DCL algorithm with ADMM-based in terms of computational complexity and effectiveness. Furthermore, the ADMM-based DCL algorithm only can achieve asymptotic convergence rate, and the DAC-based DCL algorithm can obtain exponential convergence rate. Some DCL problems can be regarded as distributed convex optimization problems, especially for single layer feedforward neural networks. Zero-Gradient-Sum (ZGS) strategy is one of the effective ways to minimize the sum of the strong convex functions in a distributed manner [13–15]. The main idea is to design an algorithm that ensure the sum of all local functional gradients remain at zero [14], which is more helpful in finding the optimal value. In [16], the

* Corresponding author.
E-mail addresses: yanfei_song@126.com (Y. Song), wshchen@126.com (W. Chen), dai0519hao@163.com (H. Dai).

https://doi.org/10.1016/j.neucom.2017.10.056
0925-2312/

authors followed the ZGS strategy and proposed an algorithm for the FNNRWs for finding the optimal learning weight, which can reach asymptotic convergence rate. It is worth mentioning that these DCL algorithms can protect the privacy of data in the learning process, and can be suit for sundry large-scale distributed machine learning problems well.

However, these algorithms can only achieve asymptotical convergence. In reality, a finite convergence time is needed, especially for systems that require high control precision and good robustness. The purpose of this work is to develop a fast convergent DCL algorithm for FNNRWs that is able to solve distributed machine learning problems with fast time by local interactions. First, we propose a continuous-time fast convergent DCL algorithm. Second, the convergence analysis based on the Lyapunov method is given. The theoretical analysis demonstrates that our algorithm has faster convergence speed in a fully distributed fashion compared with other DCL algorithms [12,16]. Third, for practical use, we extend this continuous-time algorithm to a discrete-time form by using the fourth-order Runge–Kutta method. Moreover, we show the comparison with the distributed ADMM and the ZGS-based algorithms from four datasets. Simulation results show that our algorithm illustrates the effectiveness and better performance in convergence rate than others. Finally, we subsequently show the performance of our algorithm with different parameters. Simulation results verify that the convergence rate can be adjusted by properly selecting some tuning parameters.

The remainder of the paper is organized as follows. Section 2 gives some preliminaries. In Section 3, we show the problem formulation. Our fast convergent DCL algorithm and the comparative analysis are described in Section 4. In Section 5, the numerical simulations are given. Section 6 concludes the paper.

Notations . $\mathbb{R}$ and $\mathbb{R}^+$ are the set of real numbers and the set of nonnegative real numbers, respectively; $\|\cdot\|$ represents Euclidean norm in $\mathbb{R}^n$; $\boldsymbol{C}\otimes\boldsymbol{D}=\{c_{11}\boldsymbol{D},\dots,c_{1m}\boldsymbol{D};\dots;c_{n1}\boldsymbol{D},\dots,c_{nm}\boldsymbol{D}\}\in\mathbb{R}^{np\times mq}$, where $\boldsymbol{C}=[c_{ij}]\in\mathbb{R}^{n\times m},\boldsymbol{D}\in\mathbb{R}^{p\times q}$, and $\otimes$ is called the Kronecker product; $\boldsymbol{I}_n\in\mathbb{R}^{n\times n}$ is the identity matrix; ∇f and $\nabla^2 f$ denote the gradient and the Hessian matrix of $f:\mathbb{R}^n\to\mathbb{R}$, respectively; $\boldsymbol{a}\odot\boldsymbol{b}$ is defined by $[a_1b_1,\ a_2b_2,\ \dots a_nb_n]^T$, where $\boldsymbol{a}=(a_1,a_2,\dots,a_n)^T\in\mathbb{R}^n,\boldsymbol{b}=(b_1,b_2,\dots,b_n)^T\in\mathbb{R}^n$; $\mathrm{Sig}(\boldsymbol{a})=(\mathrm{sig}(a_1),\mathrm{sig}(a_2),\dots,\mathrm{sig}(a_n))^T$, where $\mathrm{sig}(\cdot)$ denotes the sign function; $|\boldsymbol{a}|^\beta=(|a_1|^\beta,|a_2|^\beta,\dots,|a_n|^\beta)^T$, where $\beta>0$ is a constant; A $\leq$ B means A - B is a negative semidefinite matrix).

2. Preliminaries

In this section, some useful preliminaries for the proof of the main results are provided as follows, such as graph theory, the structure of FNNRWs and the finite-time stability theorem of continuous-time system.

2.1. Graph theory

Consider a network with N agents labeled as 1, 2, ..., N. The communication topology among the agents can be modeled by an undirected and connected graph $\mathcal{G}(\mathcal{V}(\mathcal{G}),\mathcal{E}(\mathcal{G}))$, where $\mathcal{V}(\mathcal{G})=\{1,\dots,N\}$ and $\mathcal{E}(\mathcal{G})\subset\{\{i,j\}:,i,j\in\mathcal{V},i\neq j\}$ denote the vertex set and the edge set, respectively. The graph $\mathcal{G}$ is undirected which indicates that for arbitrary $i,j\in\mathcal{V}(\mathcal{G})$, if $(i,j)\in\mathcal{E}(\mathcal{G})$, $(j,i)\in\mathcal{E}(\mathcal{G})$. Moreover, we let $\mathcal{A}(\mathcal{G})=[a_{ij}]\in\mathbb{R}^{N\times N}$ be the adjacency matrix, for which $a_{ij}=a_{ji}$ if $(i,\ j)$ is an edge in $\mathcal{G}$, and $a_{ij}=0$, otherwise. Suppose that each node has no self edge, i.e. $a_{ii}=0$. $\mathcal{D}(\mathcal{G})=\mathrm{diag}\{d_1,\dots,d_N\}$ represents the degree matrix of $\mathcal{G}$ with $d_i=\sum_{j=1}^N a_{ij}$. $\mathcal{L}(\mathcal{G})=\mathcal{D}(\mathcal{G})-\mathcal{A}(\mathcal{G})$ is known as the Laplacian matrix of $\mathcal{G}$. If $\bar{\mathcal{G}}$ is the complete graph of $\mathcal{G}$, we can get the properties:

$$\begin{cases}\mathcal{L}(\bar{\mathcal{G}})^2=N\mathcal{L}(\bar{\mathcal{G}}),\\ \lambda_2\mathcal{L}(\bar{\mathcal{G}})\leq N\mathcal{L}(\mathcal{G}),\end{cases}\tag{1}$$

where λ_2 is the second smallest eigenvalue of $\mathcal{L}(\mathcal{G})$, also known as the algebraic connectivity of $\mathcal{G}$.

2.2. FNNRWs

Definition 1. ([17,18]). For input vector $\boldsymbol{z}\in\mathbb{R}^m$ with l hidden neurons, the n-dimensional output function of FNNRW with zero bias of output neuron can be mathematically modeled as:

$$f_l(\boldsymbol{z})=\sum_{i=1}^{l}s_i(\boldsymbol{z};\epsilon_i,b_i)\boldsymbol{w}_i,\tag{2}$$

where s_i denotes the output function of the ith hidden neuron and is often referred to as functional link; $\epsilon_i\in\mathbb{R}^m$ is the weight vector connecting the ith hidden neuron, and $b_i\in\mathbb{R}$ denotes the bias of output of the ith hidden neuron. At the beginning of training, they are randomly chosen from a predefined probability distribution and fixed during the training process, which independently of the training data; $\boldsymbol{w}_i\in\mathbb{R}^n$ is the output weight vector, which connects the output neurons with the ith hidden neuron.

A set with M samples is denoted as $D=\{\boldsymbol{X},\boldsymbol{Y}\}$, where $\boldsymbol{X}\in\mathbb{R}^{M\times m}$ is the input set and $\boldsymbol{Y}\in\mathbb{R}^{M\times n}$ is the output set. The cost function of all samples can be written as follows

$$E(\boldsymbol{W})^{glob}=\frac{1}{2}\|\boldsymbol{Y}-\boldsymbol{S}(\boldsymbol{X})\boldsymbol{W}\|_2^2+\frac{K}{2}\|\boldsymbol{W}\|_2^2,\tag{3}$$

where

$$S(\boldsymbol{X})=\begin{pmatrix}s_1(\boldsymbol{x}_1)&\cdots&s_l(\boldsymbol{x}_1)\\ \vdots&\ddots&\vdots\\ s_1(\boldsymbol{x}_M)&\cdots&s_l(\boldsymbol{x}_M)\end{pmatrix}_{M\times l},\quad \boldsymbol{W}=\begin{pmatrix}\boldsymbol{w}_1^T\\ \vdots\\ \boldsymbol{w}_l^T\end{pmatrix}_{l\times n},$$

and $K>0$ is a tunable parameter which provides a tradeoff between the regularized item and the training errors.

To minimize the cost function is equivalent to find the optimal learning weights, which can be formulated as a standard regularized least-square problem. Obviously, whose solution is given by

$$\boldsymbol{W}^*=(\boldsymbol{S}(\boldsymbol{X})^T\boldsymbol{S}(\boldsymbol{X})+K\boldsymbol{I})^{-1}\boldsymbol{S}(\boldsymbol{X})^T\mathbf{Y},\tag{4}$$

where $\boldsymbol{W}^*$ is the global optimal weight. This result can be obtained without the iterative-training process. Furthermore, this result can be widely used in multi-valued regression and multi-class classification.

2.3. Finite-time stability

Lemma 1([19,20]). Consider the system

$$\dot{\boldsymbol{x}}=g(t,\boldsymbol{x}(t)),g(0,t)=0,\boldsymbol{x}\in U_0\subset\mathbb{R}^n,\tag{5}$$

where $g:U_0\times\mathbb{R}^+\to\mathbb{R}^n$ is continuous in an open neighborhood U_0 of origin $\boldsymbol{x}=0$. Supposing there is a continuous positive definite Lyapunov function $V(\boldsymbol{x}(t))$ defined on $U\times\mathbb{R}^+$, where $U\subset U_0$ is the neighborhood of the origin. If there are real numbers $\lambda>0$, $\alpha\in(0,1)$, such that $\dot{V}\leq-\lambda V^\alpha$ is established on U, then the system (5) is finite-time stable with the setting time T bounded by

$$T\leq\frac{V^{1-\alpha}(x(t_0))}{\lambda(1-\alpha)}.\tag{6}$$