Neurocomputing 157 (2015) 41–45

Contents lists available at ScienceDirect

Neurocomputing

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

Brief Papers

Adaptive natural gradient learning algorithms for Mackey–Glass chaotic time prediction

Junsheng Zhao *, Xingjiang Yu

School of Mathematics, Liaocheng University, Liaocheng 252059 China

ARTICLE INFO

Article history:
Received 13 August 2014
Received in revised form
8 November 2014
Accepted 10 January 2015
Communicated by Ning Wang
Available online 31 January 2015

Keywords:
Multilayer perceptron
Radial basis function networks
Adaptive gradient descent method
Mackey–Glass chaotic time series prediction

ABSTRACT

In this paper, we introduce the adaptive natural gradient method into the multilayer perceptrons (MLPs) and radial basis function (RBF) networks. We give a good performance for the Mackey–Glass chaotic time series prediction, and compare it with the LMA. Results show that the adaptive natural gradient methods for MLPs and RBFs, which are the online learning, can give almost the same performance with the LMA (MLPs), which is the batch mode learning. However, the performance of LMA (RBFs) is very poor and is very sensitive with the initial parameters.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

Natural gradient algorithm gives an on-line learning algorithm that takes the Riemannian metric of the parameter space into account [1,2]. It adopts the on-line learning mode. As is known to all on-line learning uses a training example once when it is observed, while batch learning stores all the examples and every example can be reused later. Therefore, the performance of the on-line learning is worse than batch learning. However, Amari shows that natural gradient algorithm achieves Fisher efficiency. It may obtain the Cramer–Rao bounds and gives the best asymptotic performance that any unbiased learning algorithm can achieve [1].

However, it is difficult to calculate the Fisher information matrix of multilayer perceptrons (MLPs). Even when it is obtained, its inversion is computationally expensive. Yang and Amari gave an explicit form of the Fisher information matrix for the MLPs [2]. Rattray et al. gave it in terms of statistical–mechanical order parameters [3]. As for the radial basis function (RBF) networks, Zhao et al. gave the explicit form of the Fisher information matrix [4]. These results show that it is difficult to implement natural gradient learning for practical large-scale problems, although it may give a good performance.

As a kind of nonlinear fitting problems, the Mackey–Glass chaotic time prediction has attracted more and more attentions [4–11]. As is known to all, the Mackey–Glass series is generated by the following nonlinear time delay differential equation:

$$\frac{dx(t)}{dt} = \frac{\beta x(t-\tau)}{1+x^n(t-\tau)} + \gamma x(t), \tag{1}$$

where β, γ, τ, n are real numbers. Depending on the values of the parameters, this equation displays a range of periodic and chaotic dynamics. Such a series has some short-range time coherence, but long-term prediction is very difficult. Although lots of the literatures about the Mackey–Glass fitting have arisen, almost all of them use the batch mode learning.

Many people want to know how to implement the natural gradient algorithm into the Mackey–Glass chaotic time prediction. In this paper, We will introduce the adaptive natural gradient algorithm for the MLPs and RBFs. They will give a good performance for the Mackey–Glass chaotic time prediction, and get almost the same performance with the Levenberg–Marquardt algorithm (LMA), which is a batch mode learning [12–14].

The rest of the paper is organized as follows. In Section 2, we will introduce the adaptive natural gradient method. Section 3 will give the adaptive natural gradient method for MLPs and RBFs. Mackey–Glass chaotic time prediction will be given in Section 4. Section 5 is the conclusions and discussions.

2. The adaptive natural gradient method

Considering a feedforward network, its input–output behavior is depicted as follows:

$$y = f(\mathbf{x}, \theta) + \xi, \tag{2}$$

* Corresponding author.

http://dx.doi.org/10.1016/j.neucom.2015.01.039
0925-2312/© 2015 Elsevier B.V. All rights reserved.

where $\mathbf{x} \in R^n$ is the input vector and $y \in R^1$ is the output; θ is the parameter that specialize the network; ξ is a random noise whose probability density function (pdf) is $p(x)$.

The loss function is $l(y|\mathbf{x};\theta)$, which is the logarithm of the conditional probability density function $p(y|\mathbf{x};\theta)$, then the Fisher information matrix is depicted as follows [1]:

$$G = E_{\mathbf{x}\sim p(\mathbf{x})}[E_{y\sim p(y|\mathbf{x};\theta)}\left[\frac{\partial l(y|\mathbf{x};\theta)}{\partial\theta}\frac{\partial l(y|\mathbf{x};\theta)}{\partial\theta^T}\right]. \tag{3}$$

Given the training samples $(\mathbf{x}_1,y_1),(\mathbf{x}_2,y_2),\ldots,(\mathbf{x}_L,y_L)$, the natural gradient method is

$$\theta_{t+1} = \theta_t - \eta_t G^{-1}(\theta)\frac{\partial l(y_t|\mathbf{x}_t;\theta)}{\partial\theta_t}, \tag{4}$$

where η_t is the learning rate.

According to the definition of the Fisher information matrix, we have to know the pdf of the input $\mathbf{x}$, which is hardly given in practical problems. Amari [1] and Zhao [4] had given the explicit expressions of the Fisher information matrix for MLPs and RBFs respectively [1,4]. However, when the number of the hidden units is large, the computation of the Fisher information matrix and its inverse is very difficult. Sometimes the inverse of the Fisher information matrix does not exist.

Amari gives the adaptive gradient method for MLPs by using the well known technique of the Kalman filter. It can obtain the inverse of the Fisher information matrix directly, as depicted as follows:

$$G_{t+1}^{-1} = (1+\epsilon_t)\widehat{G}_t^{-1} - \epsilon_t\widehat{G}_t^{-1}\times\frac{\partial f(\theta)}{\partial\theta_t}\times\frac{\partial f(\theta)}{\partial\theta_t^T}\widehat{G}_t^{-1}, \tag{5}$$

where ϵ_t is a time-dependent learning rate. We usually select $\epsilon_t = c/t$ and $\epsilon_t = \epsilon$.

3. Adaptive natural gradient method for MLPs and RBFs

3.1. Adaptive natural gradient method for MLPs

Let us consider a multilayer perceptron (MLP) written as

$$y = \sum_{i=1}^{m} v_i\varphi(\omega_i^T\cdot\mathbf{x}+b_\alpha)+b_0+\xi, \tag{6}$$

where $\omega_i = (\omega_{1i},\omega_{2i},\ldots,\omega_{ni})^T$ is an n-dimensional connection weight vector from the input to the ith hidden unit ($i=1,2,\ldots,m$), $\varphi(\mathbf{x}) = 1/(1+e^{-\mathbf{x}})$ is a sigmoidal activation function. The MLPs can be depicted in Fig. 1.

We know

$$f(\mathbf{x};\theta) = \sum_{i=1}^{m} v_i\varphi(\omega_i\cdot\mathbf{x}+b_\alpha)+b_0, \tag{7}$$

where $\theta = (\omega_1^T,\ldots,\omega_m^T,v_1,\ldots,v_m)^T \in R^{(n+1)m}$.

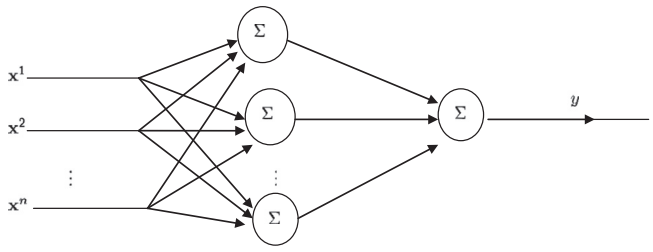

Fig. 1. Two multilayer perceptrons.

Derivatives of $f(\mathbf{x};\theta)$ with respect to θ are

$$\frac{\partial f(\mathbf{x};\theta)}{\partial\theta} = \begin{bmatrix} v_1\varphi(\omega_1\cdot\mathbf{x})(1-\varphi(\omega_1\cdot\mathbf{x}))\cdot\mathbf{x} \\ \vdots \\ v_m\varphi(\omega_m\cdot\mathbf{x})(1-\varphi(\omega_2\cdot\mathbf{x}))\cdot\mathbf{x} \\ \varphi(\omega_1\cdot\mathbf{x}) \\ \vdots \\ \varphi(\omega_m\cdot\mathbf{x}) \end{bmatrix}, \tag{8}$$

so we can get the inverse of the Fisher information matrix directly by the following formula:

$$G_{t+1}^{-1} = (1+\epsilon_t)\widehat{G}_t^{-1} - \epsilon_t\widehat{G}_t^{-1}\times\frac{\partial f(\theta)}{\partial\theta_t}\times\frac{\partial f(\theta)}{\partial\theta_t^T}\widehat{G}_t^{-1}, \tag{9}$$

where

$$\frac{\partial f(\theta)}{\partial\theta_t}\times\frac{\partial f(\theta)}{\partial\theta_t^T} = \begin{bmatrix} v_1\varphi(\omega_1\cdot\mathbf{x})(1-\varphi(\omega_1\cdot\mathbf{x}))\cdot\mathbf{x} \\ \vdots \\ v_m\varphi(\omega_m\cdot\mathbf{x})(1-\varphi(\omega_m\cdot\mathbf{x}))\cdot\mathbf{x} \\ \varphi(\omega_1\cdot\mathbf{x}) \\ \vdots \\ \varphi(\omega_m\cdot\mathbf{x}) \end{bmatrix}$$
$$\left[v_1\varphi(\omega_1\cdot\mathbf{x})(1-\varphi(\omega_1\cdot\mathbf{x}))\cdots \quad v_m\varphi(\omega_m\cdot\mathbf{x})\right.$$
$$\left.\times(1-\varphi(\omega_m\cdot\mathbf{x}))\cdot\mathbf{x}\varphi(\omega_1\cdot\mathbf{x})\cdots\varphi(\omega_m\cdot\mathbf{x})\right] \tag{10}$$

The adaptive natural gradient method for MLPs [15] is

$$\theta_{t+1} = \theta_t - \eta_t G^{-1}(\theta)\frac{\partial l(y_t|\mathbf{x}_t;\theta)}{\partial\theta_t}. \tag{11}$$

3.2. Adaptive natural gradient method for RBFs

As the radial basis function (RBF) networks are concerned, the input–output relation can be written as

$$y = \sum_{i=1}^{m} v_i\varphi(\mathbf{x},\omega_i)+\xi, \tag{12}$$

where $\omega_i = (\omega_{1i},\omega_{2i},\ldots,\omega_{ni})^T$ is the center of the ith hidden unit ($i=1,2,\ldots,m$), $\varphi(\mathbf{x},\omega) = \exp^{-\|\mathbf{x}-\omega\|^2/2}$ is the Gaussian activation function. We know

$$f(\mathbf{x};\theta) = \sum_{i=1}^{m} v_i\varphi(\mathbf{x},\omega_i), \tag{13}$$

where $\theta = (\omega_1^T,\omega_2^T,\ldots,\omega_m^T,v_1,\ldots,v_m)^T \in R^{(n+1)m}$.

Derivatives of $f(\mathbf{x};\theta)$ with respect to θ are

$$\frac{\partial f(\mathbf{x};\theta)}{\partial\theta} = \begin{bmatrix} v_1\varphi(\mathbf{x},\omega_1)(\mathbf{x}-\omega_1) \\ \vdots \\ v_m\varphi(\mathbf{x},\omega_m)(\mathbf{x}-\omega_m) \\ \varphi(\mathbf{x},\omega_1) \\ \vdots \\ \varphi(\mathbf{x},\omega_m) \end{bmatrix}, \tag{14}$$

so we can get the inverse of the Fisher information matrix directly by the following formula:

$$G_{t+1}^{-1} = (1+\epsilon_t)\widehat{G}_t^{-1} - \epsilon_t\widehat{G}_t^{-1}\times\frac{\partial f(\theta)}{\partial\theta_t}\times\frac{\partial f(\theta)}{\partial\theta_t^T}\widehat{G}_t^{-1}, \tag{15}$$

where

$$\frac{\partial f(\theta)}{\partial\theta_t}\times\frac{\partial f(\theta)}{\partial\theta_t^T} = \frac{\partial f(\mathbf{x};\theta)}{\partial\theta} = \begin{bmatrix} v_1\varphi(\mathbf{x},\omega_1)(\mathbf{x}-\omega_1) \\ \vdots \\ v_m\varphi(\mathbf{x},\omega_m)(\mathbf{x}-\omega_m) \\ \varphi(\mathbf{x},\omega_1) \\ \vdots \\ \varphi(\mathbf{x},\omega_m) \end{bmatrix}$$

Download English Version:

https://daneshyari.com/en/article/409380

Download Persian Version:

https://daneshyari.com/article/409380

Daneshyari.com