Neurocomputing 194 (2016) 117–125

Contents lists available at ScienceDirect

Neurocomputing

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

Adaptive control of rapidly time-varying discrete-time system using initial-training-free online extreme learning machine

Xiang Hui Gao [a], Ka In Wong [a], Pak Kin Wong [a,*], Chi Man Vong [b]

[a] Department of Electromechanical Engineering, University of Macau, Macau
[b] Department of Computer and Information Science, University of Macau, Macau

ARTICLE INFO

Article history:
Received 5 August 2015
Received in revised form
18 November 2015
Accepted 26 January 2016
Communicated by: G.-B. Huang
Available online 22 February 2016

Keywords:
Adaptive control
System identification
Time-varying discrete systems
Machine learning

ABSTRACT

While multiple model adaptive control (MMAC) scheme provides a solution to systems with unknown and rapidly time-varying parameters, many offline samples must be obtained beforehand, and the number of models is difficult to be found if no prior knowledge is given. This paper proposes a new adaptive control strategy to handle such systems. The principle is to use a change detection mechanism to check if there is an abrupt change, and immediately train a new model if a change is detected. A novel online identification algorithm, namely initial-training-free online extreme learning machine (ITF-OELM), is also proposed to allow the model to be trained anytime without concerns on prior data. With this strategy, only one model is necessary as compared to MMAC, resulting in reduction on computational complexity and memory usage. Simulation results show that the proposed strategy is effective. Besides, although the use of forgetting factor in ITF-OELM can accelerate the convergence speed for system identification, sometimes it may lead to ill-conditioned covariance matrix in the recursively updating process. This paper shows that such issue can be solved by the change detection mechanism.

1. Introduction

Most conventional control approaches cannot provide satisfactory control performance for complex uncertain nonlinear systems due to the difficulties in developing exact mathematical models for the unknown dynamics (i.e. uncertainties) of the systems. Neural networks (NNs) were therefore suggested for adaptive control of uncertain nonlinear systems early in 1990s [1–3], owing to their superior ability in identifying the internal dynamical behavior of unknown systems. Since then, adaptive control using NNs has received significant attention [4], and a large number of related studies exist in the literature [5–9]. The principle is to estimate the unknown parameters of the controlled plant by adjusting (or so-called training) the weights of the network online based on the real-time information from the plant, and then compute the corresponding control signal based on the trained network. Although various adaptive NN control schemes have already been proposed, most of them mainly focus on systems of which the unknown parameters are constant or vary slowly with time. In many practical applications, however, the underlying time-varying systems are subject to fast changing environments such that the unknown parameters may vary largely and rapidly [10,11]. In such cases, the transient performance of these adaptive control schemes are often unsatisfactory [12].

To handle systems with rapidly time-varying parameters, two strategies are available in the literature. The first strategy is to use multiple NNs, which is also known as the multiple model adaptive control (MMAC) scheme [11–13]. Its principle is to, via a switching mechanism, select in real time the best model of the plant among a set of pre-trained NNs and switch to the selected model to derive the corresponding control signal. While MMAC has been proven to be particularly useful for this kind of systems [13–17], a group of models has to be trained in advanced. Thus, a lot of off-line training data must be collected beforehand to train the multiple models. Yet for some practical cases, collection of a large chunk of representative initial data in advance is very difficult, time-consuming and labor-demanding. Moreover, if no prior knowledge is given or the available knowledge is not sufficient, it is extremely difficult to determine the necessary number of models. Besides, the use of multiple models increases the controller complexity, which further increases the computational burden [8]. Hence, MMAC may not be a good solution to practical applications, especially for onboard control cases that the memory and computational capacity of the embedded equipment (e.g. electronic control unit of a car) are limited.

The alternative strategy to deal with rapidly time-varying parameters is by using modified recursive least-squares (RLS) algorithms. Such strategy involves (i) a change detection

* Corresponding author.
E-mail address: fstpkw@umac.mo (P.K. Wong).

http://dx.doi.org/10.1016/j.neucom.2016.01.071
0925-2312/© 2016 Elsevier B.V. All rights reserved.

mechanism for determining in real time the "parameter jumping" locations and (ii) a corresponding adjustment on the RLS algorithms at the jumping locations [10]. The difficulty in using this strategy, however, is that a proper adjustment has to be made in order to guarantee the tracking performance. Modifying the covariance matrix [10,18] and varying the forgetting factor [19,20] in the RLS algorithms are two possible approaches, but there is currently no effective way to select the most suitable parameters for these two mechanisms as they are case-dependent; an improper choice of parameters (e.g. forgetting factor) may result in large misadjustment, poor stability and slow convergence [21].

Aiming to address these aforementioned issues, this paper develops a new strategy for adaptive control of rapidly time-varying system, of which the idea is to use a change detection mechanism to check whether an abrupt change of the system parameters occurs, and if a change is detected, immediately start a re-training process of the network to identify the newly changed system. As compared to MMAC, only one NN is necessary throughout the whole control process in this proposed strategy. Moreover, as compared to the strategy using modified RLS algorithms, optimal parameter selection is no longer required. Hence, the computational burden, human intervention and experimental cost can be greatly reduced. The core of this proposed strategy, however, is an algorithm that can easily train a NN without using prior offline data and must have fast convergence speed in identifying the newly changed parameters. A novel algorithm called initial-training-free online extreme learning machine (ITF-OELM) is therefore proposed.

The proposed ITF-OELM is an improved version of online sequential extreme learning machine (OS-ELM). Originally, OS-ELM is an efficient online identification algorithm for training single-hidden-layer feed-forward NNs, of which the hidden layer weights are initialized randomly and remain fixed while only the output weights between the hidden layer and the output layer are updated using a learning algorithm similar to RLS [22]. This OS-ELM algorithm has been recently used for adaptive control of nonlinear discrete-time system under the aforementioned MMAC scheme [23]. The proposed ITF-OELM, on the other hand, introduces a regularization factor to the training algorithm of OS-ELM so that the constraint on the maximum number of hidden nodes in OS-ELM can be resolved [24], and employs an initialization way different from OS-ELM such that no prior offline training data is necessary for the network construction. Consequently, ITF-OELM allows the NN be re-trained anytime when necessary without concerns on the network architecture and prior data acquisition. As a result, the aforementioned new strategy can be developed with this advantageous feature of ITF-OELM so that when a rapid change of parameters is detected, a new identification process can be started immediately for the NN to adapt to the newly changed system.

The organization of this paper is as follows. A brief review of OS-ELM and the details of ITF-OELM are provided in Section 2. The proposed strategy for adaptive control of rapidly time-varying discrete-time system using ITF-OELM is presented in Section 3. To evaluate the proposed strategy, simulations under some nonlinear discrete-time systems with rapidly changing parameters are conducted and the simulation results are given in Section 4. Conclusions are finally drawn in Section 5.

2. Details of OS-ELM and ITF-OELM

2.1. OS-ELM

OS-ELM is an online learning algorithm originated from extreme learning machine (ELM) for single-hidden-layer feedforward NNs [22]. A single-hidden-layer feedforward NN with d input nodes, L hidden nodes and one output node has the following form:

$$f(\mathbf{x}) = \sum_{i=1}^{L} \beta_i G(\boldsymbol{a}_i, b_i, \mathbf{x}) = \sum_{i=1}^{L} \beta_i h_i(\mathbf{x}) = \mathbf{h}(\mathbf{x})\boldsymbol{\beta} \tag{1}$$

where the subscript i indicates the ith hidden node, $G(\cdot)$ is the active function of the hidden nodes with parameters $\boldsymbol{a}_i \in \boldsymbol{R}^d$ and $b_i \in R$, $\boldsymbol{\beta} = [\beta_1, \beta_2, \dots, \beta_L]^{\mathrm{T}}$ is the output weight vector containing the weights between the hidden layer and the output node, and $\mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x}), h_2(\mathbf{x}), \dots, h_L(\mathbf{x})] = [G(\boldsymbol{a}_1, b_1, \mathbf{x}), G(\boldsymbol{a}_2, b_2, \mathbf{x}), \dots, G(\boldsymbol{a}_L, b_L, \mathbf{x})]$ represents the hidden layer output vector with respect to the input$\mathbf{x}$.

The purpose of OS-ELM is to search for a set of parameters in Eq. (1) that can approximate a given function (even with highly nonlinear nature) as much as possible in online manner. According to theories of ELM [25,26], even though the hidden parameters are generated randomly and untuned, the network trained by OS-ELM still retains universal approximation capability. In other words, it is only necessary for OS-ELM to train the output weights of the NN. To train the output weights, OS-ELM mainly uses two phases: an initialization phase and an online sequential learning phase. In the initialization phase, a base network is firstly initialized using basic ELM algorithm with a small chuck of offline training samples. For an initial training dataset with N_0 training samples $\{(\mathbf{x}_i, T_i)\}_{i=1}^{N_0}$, basic ELM aims to minimize the norm of output error:

$$\text{Minimize}: \|\boldsymbol{H}_0\boldsymbol{\beta}^{(0)} - \boldsymbol{T}_0\|^2 \tag{2}$$

where $\boldsymbol{T}_0 = [T_1, \cdots T_{N_0}]$ is target output vector, and $\boldsymbol{H}_0 = [\mathbf{h}(\mathbf{x}_1), \mathbf{h}(\mathbf{x}_2), \dots, \mathbf{h}(\mathbf{x}_{N_0})]^{\mathrm{T}}$ is the $N_0 \times L$ hidden layer output matrix with each row of $\boldsymbol{H}_0$ a training sample after feature mapping. The solution of $\boldsymbol{\beta}^{(0)}$ is:

$$\boldsymbol{\beta}^{(0)} = \boldsymbol{H}_0^{\dagger}\boldsymbol{T}_0 \tag{3}$$

where $\boldsymbol{H}_0^{\dagger}$ is the Moore–Penrose pseudo-inverse of matrix $\boldsymbol{H}_0$. If ${\boldsymbol{H}_0}^{\mathrm{T}}\boldsymbol{H}_0$ is nonsingular, the orthogonal projection method can be used to calculate the pseudo-inverse of $\boldsymbol{H}_0$:

$$\boldsymbol{\beta}^{(0)} = \boldsymbol{P}_0\boldsymbol{H}_0\boldsymbol{T}_0 \tag{4}$$

$$\boldsymbol{P}_0 = (\boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{H}_0)^{-1}. \tag{5}$$

After a base network is initialized using Eqs. (4) and (5), the network can then be used for online incremental learning through the second phase of OS-ELM. For new arriving training data of size N_{k+1}, the weight updating algorithm used in OS-ELM takes a similar form to RLS algorithm [22]:

$$\boldsymbol{\beta}^{(k+1)} = \boldsymbol{\beta}^{(k)} + \boldsymbol{P}_{k+1}\boldsymbol{H}_{k+1}^{\mathrm{T}}\left(\boldsymbol{T}_{k+1} - \boldsymbol{H}_{k+1}\boldsymbol{\beta}^{(k)}\right) \tag{6}$$

$$\boldsymbol{P}_{k+1} = \boldsymbol{P}_k - \boldsymbol{P}_k\boldsymbol{H}_{k+1}^{\mathrm{T}}(\boldsymbol{I}_{N_{k+1}} + \boldsymbol{H}_{k+1}\boldsymbol{P}_k\boldsymbol{H}_{k+1}^{\mathrm{T}})^{-1}\boldsymbol{H}_{k+1}\boldsymbol{P}_k \tag{7}$$

where $\boldsymbol{T}_{k+1}$ indicates the target of $(k+1)$th arriving training data with k starting from zero, $\boldsymbol{H}_{k+1}$ is the hidden layer output for the $(k+1)$th arriving training data, and $\boldsymbol{I}_{N_{k+1}}$ is an identity matrix of size N_{k+1}.

It is notable that in OS-ELM, the number of distinct initial training data for training the base network should not be less than the number of hidden nodes in the network, in order to guarantee that the hidden layer output matrix $\boldsymbol{H}_0$ is full rank and the term ${\boldsymbol{H}_0}^{\mathrm{T}}\boldsymbol{H}_0$ is non-singular. However, under this constraint, the OS-ELM algorithm may still encounter the ill-posed problems [24]. To simultaneously solve the singular and ill-posed problems of OS-ELM, a regularized version of OS-ELM was proposed in [24] based on Tikhonov regularization. Instead of Eq. (2), the cost function