Information Sciences 364–365 (2016) 197–212

Contents lists available at ScienceDirect

Information Sciences

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

Randomized algorithms for nonlinear system identification with deep learning modification

Erick de la Rosa, Wen Yu*

Departamento de Control Automatico, CINVESTAV-IPN (National Polytechnic Institute), Av. IPN 2508, Mexico City 07360, Mexico

ARTICLE INFO

Article history:
Received 7 May 2015
Revised 14 September 2015
Accepted 26 September 2015
Available online 8 October 2015

Keywords:
Deep learning
Nonlinear system modeling
Randomized algorithms

ABSTRACT

Both randomized algorithms and deep learning techniques have been successfully used for regression and classification problems. However, the random hidden weights of the randomized algorithms require suitable distributions in advance, and the deep learning methods do not use the output information in system identification.

In this paper, the distributions of the hidden weights are obtained by the restricted Boltzmann machines. This deep learning method uses input data to construct the statistical features of the hidden weights. The output weights of the neural model are trained by normal randomized algorithms. So we successfully combine the unsupervised training (deep learning) and the supervised learning method (randomized algorithm), and take advantages from both of them. The proposed randomized algorithms with deep learning modification are validated with three benchmark problems.

1. Introduction

Neural networks use a family of statistical learning algorithms to estimate or approximate functions or nonlinear systems. The most used neural model is the multilayer perceptrons (MLP). The neural modeling process can be divided in two tasks: structure selection and parameter identification. The structure selection usually lies on a substantial amount of heuristic observation to express proper strategy's knowledge. It is often tackled by trial-and-error approaches, like the unbias criterion [20], evolutionary algorithms [15], fuzzy grid [14], and data mining methods [18]. However, these algorithms do not improve a lot identification accuracy, because they only try to find the number of hidden neurons, and they do not determine the number of hidden layers.

From universal approximation theory, a single hidden layer neural network can approximate any nonlinear function to any prescribed accuracy if sufficient hidden neurons are provided. Also if the hidden neuron number is equal to the number of training examples, the training error arrives zero. However, the training examples are usually much larger than the hidden nodes. Increasing hidden layers with less hidden nodes can also achieve desired training accuracy. This is the basic idea of deep neural networks.

1.1. Randomized algorithms

Randomized algorithms have been initially proposed in [21] and deeply studied in [12] for single hidden layer neural networks, where the hidden weights are chosen randomly and the pseudoinverse approach (or the least square method) is applied

* Corresponding author. Tel.: +52 5557541613.
E-mail address: yuw@ctrl.cinvestav.mx (W. Yu).

http://dx.doi.org/10.1016/j.ins.2015.09.048
0020-0255/© 2015 Elsevier Inc. All rights reserved.

to calculate the output weights. The advantages of using the pseudoinverse are: it gives an optimal solution in the sense of least square and finds the optimal weights with minimal norm. Igelnik and Pao [12] extended the above algorithms to random sampling: the hidden weights are sampled from a continuous distribution. It shows that for the single hidden layer neural network, the optimization for the hidden layer parameters does not improve the generalization behavior significantly, while updating the output weights is more effective. Randomized algorithms have been successfully applied to nonlinear system identification in [25].

The parameter identification of neural models is usually addressed by some gradient descent variants, e.g., the least squares algorithm, back-propagation, and the Levenberg–Marquardt method. Even though these methods have been widely used, they may converge very slowly and have the local minima problem [13]. Since the identification error space is unknown, the neural model can be settled down in a local minima easily if the initial weights of the neural model are not suitable [9]. There are some techniques to overcome the local minima in the error space and to force the neural model near the global minimum, such as noise-shaping modification for the gradient descent algorithm [6], adding momentum term [17], and combining nonlinear clustering [16]. These algorithms modify the gradient descent algorithms to avoid the local minima problem, but they do not solve the key problem of the local minima: wrong initial weights.

The pseudoinverse approach of the randomized algorithm can solve the local minima problem without considering the hidden weights [2]. By the sensitivity ratio analysis, Song [23] gives a method to calculate the initial weights of a recurrent neural network. In [27], the initial weights are obtained by finding the support vectors of the input data. However, the above papers do not consider one important issue: the initial hidden weights depend on the statistical features of the input data [12].

1.2. Deep learning models

A deep neural network has the same structure as a MLP. The depth of the neural network is defined as the hidden layer number [3]. A deep structure has at least two hidden layers [3], and usually needs fewer neurons (or weights) than a MLP [10]. However, increasing the number of hidden layers causes exponentially increasing model complexity and requires more training examples [8]. Restricted Boltzmann machines [10] are main deep learning methods, they use energy-based learning models. The training process of deep learning is unsupervised, i.e., it uses input information.

The deep learning has two goals: (a) it guides the weights to regions of minimal norm, (b) it sets the weights in zones of the parameter space where the likelihood of a global minimum is maximum [3]. The results of [8] show that the unsupervised training can drive the neural model away from the local minima for the classification problems. However, deep learning methods cannot be applied to system identification directly, because the input/output values are non-binary as classification problems. Most of deep learning techniques also use binary data, for example the conditional probability transformation in the restricted Boltzmann machines needs binary values [10].

Deep learning techniques for system identification can be regarded as a pre-training stage. Only input data are used for this unsupervised learning stage. The objective of this stage is to learn the probability distribution of input data $P(x)$. This helps to decide the conditional probability distribution $P(y|x)$, which is the objective of system identification [8]. Since the unsupervised deep learning minimizes the variance and introduces bias into the input space X, the supervised learning for X and Y can be improved. This is explained by Bengio and Delalleau [3]: in the unsupervised learning stage, the input information are sent to hidden layers to construct useful statistical features. This mechanism improves the corresponding input/output representation. The input distribution $P(x)$ appears in the hidden units via the deep learning method.

In this paper, we take both advantages of the deep learning and the randomized algorithm for nonlinear system identification. The neural model has deep structure, which increases hidden layers and decreases hidden neurons. The complexity of the neural model does not change, while the modeling capacity is improved. The restricted Boltzmann machines are modified to train the hidden weights with input data. Then we use the randomized algorithm to train the output weights. Three benchmark examples are applied to show that the randomized algorithm with deep learning modification can improve the identification accuracy for nonlinear system identification.

2. Nonlinear system identification with deep neural networks

Consider the following unknown discrete-time nonlinear system

$$\bar{x}(k+1) = f[\bar{x}(k), u(k)], \quad y(k) = g[\bar{x}(k)] \tag{1}$$

where $u(k) \in \Re^u$ is the input vector, $\bar{x}(k) \in \Re^x$ is an internal state vector, and $y(k) \in \Re^m$ is the output vector. f and g are general nonlinear smooth functions $f, g \in C^\infty$. Denoting $Y(k) = [y^{\mathrm{T}}(k), y^{\mathrm{T}}(k+1), \dots y^{\mathrm{T}}(k+n-1)]^{\mathrm{T}}$, $U(k) = [u^{\mathrm{T}}(k), u^{\mathrm{T}}(k+1), \dots u^{\mathrm{T}}(k+n-2)]^{\mathrm{T}}$, if $\frac{\partial Y}{\partial \bar{x}}$ is non-singular at $\bar{x} = 0$, $U = 0$, this leads to the following NARMA model

$$y(k) = \Gamma[x(k)] \tag{2}$$

where

$$x(k) = [y^T(k-1), y^T(k-2), \dots u^T(k), u^T(k-1), \dots]^{\mathrm{T}}$$

$\Gamma(\cdot)$ is an unknown nonlinear difference equation representing the plant dynamics, $u(k)$ and $y(k)$ are measurable scalar input and output. The nonlinear system (2) is a NARMA model. We can also regard the input of the nonlinear system as $x(k) = [x_1 \dots x_n]^{\mathrm{T}} \in \Re^n$, the output as $y(k) \in \Re^m$.