Engineering Applications of Artificial Intelligence 26 (2013) 281–292

Contents lists available at SciVerse ScienceDirect

Engineering Applications of Artificial Intelligence

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

Dynamic neural networks with hybrid structures for nonlinear system identification

Jiamei Deng

School of Mechanical and Automotive Engineering, Kingston University London, London SW15 3DW, UK

ARTICLE INFO

Article history:
Received 15 August 2011
Received in revised form
14 March 2012
Accepted 1 May 2012
Available online 5 July 2012

Keywords:
Recurrent neural networks
System identification
Nonlinear systems
Feedback linearisation
Nonlinear control

ABSTRACT

Dynamic neural networks (DNNs) have important properties that make them convenient to be used together with nonlinear control approaches based on state space models and differential geometry, such as feedback linearisation. However the mapping capability of DNNs are quite limited due to their fixed structure, that is, the number of layers and the number of hidden units. An example shown in this paper has demonstrated this limitation of DNNs. The development of novel DNN structures, which has good mapping capability, is a relevant challenge being addressed in this paper. Although the structure is changed minorly only, the mapping capability of the new designed DNN in this paper has been improved dramatically. Previous work [J. Deng et al., 2005. The dynamic neural network of a hybrid structure for nonlinear system identification. In: 16th IFAC World Congress, Prague.] presents a new dynamic neural network structure which is suitable for the identification of highly nonlinear systems, which needs the outputs from the real system for training and operation. This paper presents a hybrid dynamic neural network structure which presents a similar idea of serial–parallel hybrid structure, but it uses an output from another neural network for training and operation classified as a serial–parallel model. This type of DNNs does not require the output of the plant to be used as an input to the model. This neural network has the advantages of good mapping capabilities and flexibilities in training complicated systems, compared to the existed DNNs. A theoretical proof showing how this hybrid dynamic neural network can approximate finite trajectories of general nonlinear dynamic systems is given. To illustrate the capabilities of the new structure, neural networks are trained to identify a real nonlinear 3D crane system.

1. Introduction

The introduction of artificial neural network methods for the identification and control of dynamical systems two decades ago has had a significant impact in control systems research (Narendra and Parthasarathy, 1990; Miller et al., 1990). Many industrial control systems exhibit nonlinear characteristics. Most conventional control schemes use linear models. However, the use of linear models can result in a serious deterioration of control performance for many nonlinear plants. The presence of nonlinearities in control systems complicates the design stages and may cause performance problems if not considered appropriately. Research over decades has produced several nonlinear control strategies based on mathematical foundations. One of the most important control techniques is the input–output linearisation of nonlinear systems. Based on differential geometric concepts and properties of the system, input–output linearisation and decoupling introduce a new input variable v and a nonlinear transformation which uses state feedback to compute the original input u, so that a n-input n-output nonlinear system of the form $\dot{x}=f(x)+g(x)u$ is transformed into n single-input single-output systems which exhibit linear dynamics between the inputs and outputs. A linear controller can be applied to the linearised system to achieve the final design requirements. The main drawback of the input–output linearisation technique is that it depends on the exact knowledge of a nonlinear model of the process. When a suitable first principles model is not available, this may be overcome by using a dynamic neural network to identify a model to be suitable for the input–output linearisation and decoupling. A recurrent neural network is a closed loop system, with feedback paths introducing dynamics into the model. They can be trained to learn the system dynamics without assuming much knowledge about the structure of the system under consideration.

Dynamic neural networks (DNNs) have important properties that make them convenient to be used together with nonlinear control approaches based on state space models and differential geometry (Garces et al., 2003), such as feedback linearisation. However, the mapping capability of DNNs are quite limited due to

E-mail address: jiameideng@hotmail.com

0952-1976/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.engappai.2012.05.003

their fixed structure, that is, the number of layers and the number of hidden units (Lippman, 1987; Mabhan and Zomaya, 1994). An example shown in this paper has demonstrated this limitation of DNNs. The development of novel DNN structures, which has good mapping capability, is a relevant challenge being addressed in this paper and previous work (Deng et al., 2005). Although the structure is changed minorly only, the mapping capability of the new designed DNN in this paper has been improved dramatically. Deng et al. (2005) presented a new dynamic neural network structure which is suitable for the identification of highly nonlinear systems, which needs the outputs from the real system for training and operation. This paper presents a hybrid dynamic neural network structure which presents a similar idea of serial–parallel hybrid structure, but it uses an output from another neural network for training and operation classified as a serial-parallel model (Narendra and Parthasarathy, 1990). This type of DNNs does not require the output of the plant to be used as an input to the model. This neural network has much better mapping capabilities and is more flexible in training complicated systems, compared to the DNNs in Garces et al. (2003). A theoretical proof showing how this hybrid dynamic neural network can approximate finite trajectories of general nonlinear dynamic systems is given. To illustrate the capabilities of the new structure, neural networks are trained to identify a real nonlinear 3D crane system.

The paper is organized as follows. Section 2 discusses the universal approximation property of static multilayer perceptrons. Section 3 introduces the class of dynamic neural networks of interest in this paper. Section 4 discusses theoretical results on the approximation ability of dynamic neural networks. Section 5 presents an example. Finally, Section 6 gives concluding remarks.

2. Different types of dynamic neural networks

Dynamic neural networks are made of interconnected dynamic neurons, also called units. The class of neuron of interest in this paper is described by the following differential equation:

$$\dot{x}_i = -\beta_i x_i + \sum_{j=1}^{N} \omega_{ij}\sigma(y_j) + \sum_{j=1}^{m} \gamma_{ij} u_j, \tag{1}$$

where β_i, ω_{ij} and γ_{ij} are adjustable weights, with $1/\beta_i$ a positive time constant and x_i is the activation state of the ith unit, y_j is the actual system output or the hidden state of the jth unit, $\sigma : \mathbb{R} \to \mathbb{R}$ is a sigmoid function and $u_1, \ldots, u_m$ are the input signals.

A dynamic neural network is formed by a single layer of N units. The first n units are taken as the output of the network, leaving $N-n$ units as hidden neurons. A type 1 DNN is defined by the following vectorised expression:

$$\dot{x} = -\beta x + \omega\sigma(y) + \gamma u,$$

$$y_n = C_n x, \tag{2}$$

where x are coordinates on $\mathbb{R}^N$, $\beta \in \mathbb{R}^{N\times N}$ is a diagonal matrix with diagonal elements $\{\beta_1, \ldots, \beta_N\}$, $\omega \in \mathbb{R}^{N\times N}$, $\gamma \in \mathbb{R}^{N\times m}$ are weight matrices, $\sigma(x) = [\sigma(x_1), \ldots, \sigma(x_N)]^T$ is a vector sigmoid function, $u \in \mathbb{R}^m$ is the input vector, $y_n \in \mathbb{R}^n$ is the plant output vector, $y = [y_n^T, x_{n+1}, \ldots, x_N]^T$, $C_n = [I_{n\times N}, 0_{n\times(N-n)}]$.

A type 1 DNN differs from the dynamic neural network described in Chapter 4 of the book Garces et al. (2003), which in this paper is known as type 2 DNN, in the argument of the vector sigmoid function $\sigma(\cdot)$. A type 2 DNN is described by the following vectorised expression:

$$\dot{x} = -\beta x + \omega\sigma(x) + \gamma u,$$

$$y_n = C_n x. \tag{3}$$

Define the output state vector $x^p = [x_1^p, \ldots, x_n^p]^T = y_n$ as the internal state of the n output units. Define the hidden state vector $x^h = [x_1^h, \ldots, x_{N-n}^h]^T$ as the internal state of the $N-n$ hidden units. A type 1 DNN uses plant output and the hidden state in the argument of the vector sigmoid function $\sigma(\cdot)$, while a type 2 DNN uses the whole state vector of the network, which consists of the output states and the hidden states, in the argument of the vector sigmoid function. The difference is illustrated in Figs. 1 and 2.

A type 3 DNN is defined by the following vectorised expression:

$$\dot{x} = -\beta x + \omega\sigma(\hat{y}) + \gamma u,$$

$$y_n = C_n x, \tag{4}$$

where x are coordinates on $\mathbb{R}^N$, $\beta \in \mathbb{R}^{N\times N}$ is a diagonal matrix with diagonal elements $\{\beta_1, \ldots, \beta_N\}$, $\omega \in \mathbb{R}^{N\times N}$, $\gamma \in \mathbb{R}^{N\times m}$ are weight matrices, $\sigma(x) = [\sigma(x_1), \ldots, \sigma(x_N)]^T$ is a vector sigmoid function, $u \in \mathbb{R}^m$ is the input vector, $x^e \in \mathbb{R}^n$ is the estimated output vector of another neural network, $\hat{y} = [x^{eT}, x_{n+1}, \ldots, x_N]^T$, $C_n = [I_{n\times N}, 0_{n\times(N-n)}]$.

A type 3 DNN differs from a type 1 DNN, in the argument of the vector sigmoid function $\sigma(\cdot)$. A type 3 DNN is different from a type 1 DNN in that a type 3 DNN uses the outputs from another neural network and the hidden states in the argument of the vector sigmoid function $\sigma(\cdot)$, while a type 1 DNN uses the plant outputs and the hidden states of the network, which consists of the output states and the hidden states, in the argument of the vector sigmoid function. The type 3 DNN is illustrated in Fig. 3.

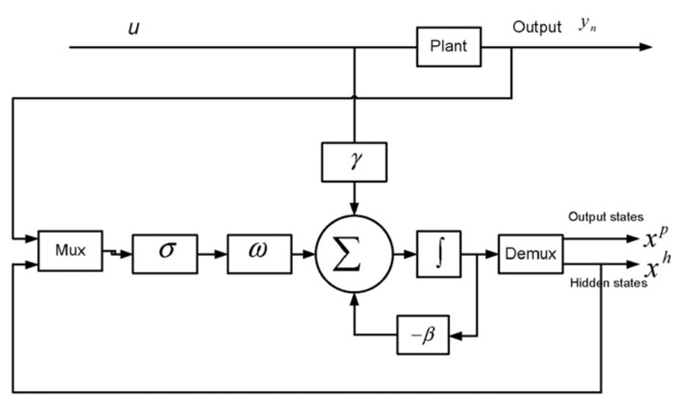

Fig. 1. Block diagram of type 1 DNN.

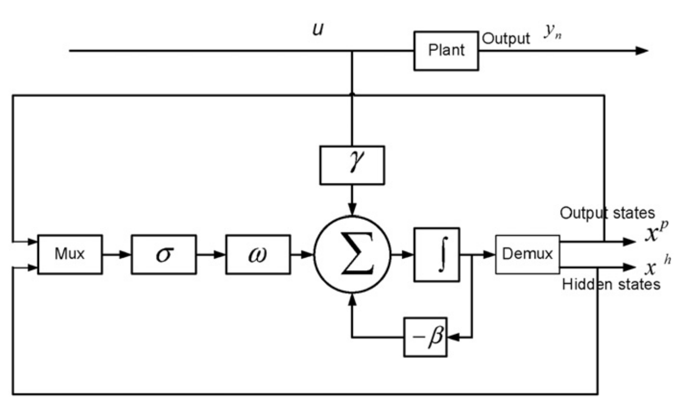

Fig. 2. Block diagram of type 2 DNN.