Neurocomputing 205 (2016) 45–52

Contents lists available at ScienceDirect

Neurocomputing

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

Brief Papers

Using RBFs in a CMAC to prevent parameter drift in adaptive control

C.J.B. Macnab [1]

Department of Electrical and Computer Engineering, University of Calgary, Alberta, Canada T2N 1N4

ARTICLE INFO

Article history:
Received 14 October 2015
Received in revised form
23 February 2016
Accepted 24 April 2016
Communicated by Lixian Zhang
Available online 14 May 2016

Keywords:
Cerebellar Model Articulation Controller
Radial Basis Function Network
Direct adaptive control
Bursting
Lyapunov stability

ABSTRACT

A radial Basis Function Network (RBFN) works well as a nonlinear approximator in direct adaptive control, as long as the number of inputs is low. A Cerebellar Model Arithmetic Computer (CMAC) indexes basis functions efficiently and can handle many inputs, but is prone to adaptive-parameter drift and subsequent bursting. This paper proposes using overlapping RBFs inside a CMAC structure. Specifically the RBFs associated with past and future (predicted) CMAC cells on a CMAC layer are activated along with the currently indexed cell's RBF on that layer. The novel neural network structure achieves the computational efficiency of the CMAC, yet can avoid drift when RBF widths are wide enough. Simulation results with a pendulum compare the performance and robustness of CMAC, RBF, and the proposed RBFCMAC in both the disturbance-free case and with sinusoidal disturbance.

1. Introduction

Neural networks have found success in a wide variety of applications, for recent examples see [1,2]. Adaptive neural networks are often used in filtering, with some of the latest results as follows. In [3] an adaptive neural network filtering approach reduces electrocardiogram signal noise. An adaptive neural fuzzy inference system and a functional link neural network are used to construct a filter for removing artifacts from an electroencephalogram in [4]. An adaptive artificial neural network model restores severely corrupted images in [5]. Using the neural network as a universal approximator inside an adaptive control scheme is also a well-known technique and recent results are as follows. In [6] the control structure includes a proportional-derivative control term in the feedback loop and a radial-basis-function neural network in the feedforward loop (mimicking a human motor learning control mechanism) and is proposed for a class of uncertain Euler–Lagrange systems. An adaptive recurrent Chebyshev neural network control system is proposed to control a permanent magnet synchronous motor servo-drive electric scooter with V-belt continuously variable transmission in [7]. A nonlinear model predictive control method using radial Basis Function Networks is proposed to guarantee the system stability for a sampled-data nonlinear plant and compensate for network-induced delays in [8].

In this paper, we focus on control system applications where speed of learning/adaptation is a critical factor. Radial Basis Function Networks (RBFNs) using Gaussian functions provide smooth and accurate approximations of nonlinear functions and converge much faster than backpropagation networks. Lyapunov-stable update laws adjust the weights when RBFs are used in direct adaptive control. However, RBFNs suffer from the "curse of dimensionality", in that the number of basis functions required to achieve the same level of approximation accuracy goes up exponentially with the number of inputs when the centres are placed on a grid/lattice. For this reason, it is unlikely that an unmodified RBF will be used in a control application when the number of inputs is greater than three or four, due to the need to calculate all the basis functions in real time. This severely restricts the possible control applications of RBF networks. Various approaches involving careful placement of basis function centres (instead of on a grid/lattice) have been proposed to try and ameliorate this problem, including [9].

On the other hand, the Cerebellar Model Arithmetic Computer [10] (originally "Articulation Controller" but still CMAC [11]) is a type of neural network that also converges faster than backpropagation networks, but avoids the curse of dimensionality. Its basis functions have hypercube domains (*cells*), and only the indexed (*activated*) basis functions need to be calculated in real time. A CMAC does not achieve as smooth an approximation as an RBFN, but is not limited in the number of inputs: for a comparison of RBFNs and CMACs see [12]. Many have proposed modifications to the CMAC to improve training time and/or approximation ability including [13–15]. CMAC has found success in a number of control applications: for recent examples see [16,17].

E-mail address: cmacnab@ucalgary.ca
[1] Tel.: +1 403 220 4877; fax: +1 403 282 6855.

http://dx.doi.org/10.1016/j.neucom.2016.04.022
0925-2312/

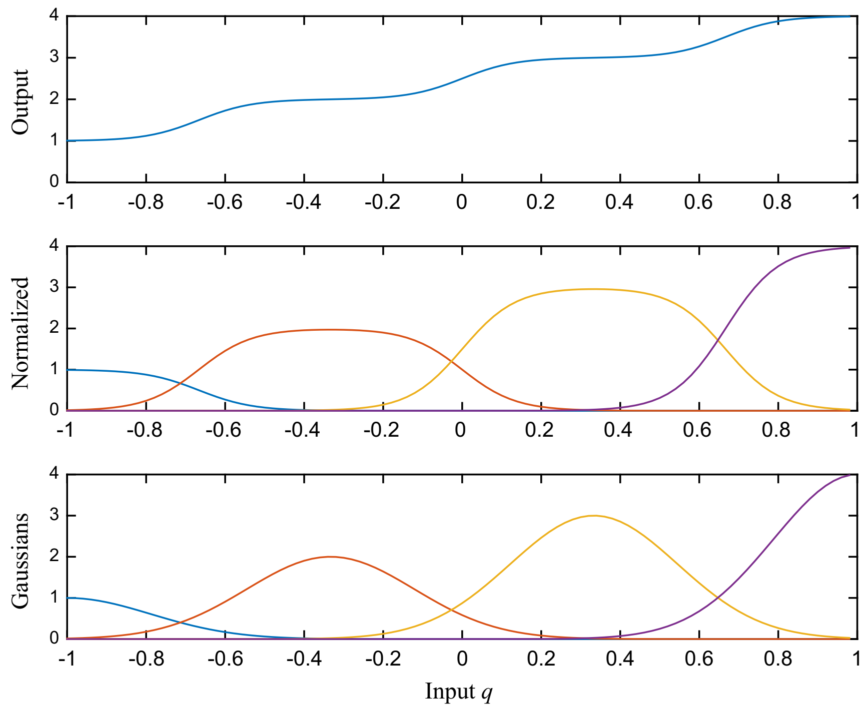

Fig. 1. Single-input normalized Gaussian RBFN: Evenly spaced Gaussians (bottom graph) are normalized (middle graph) before providing the output (top graph).

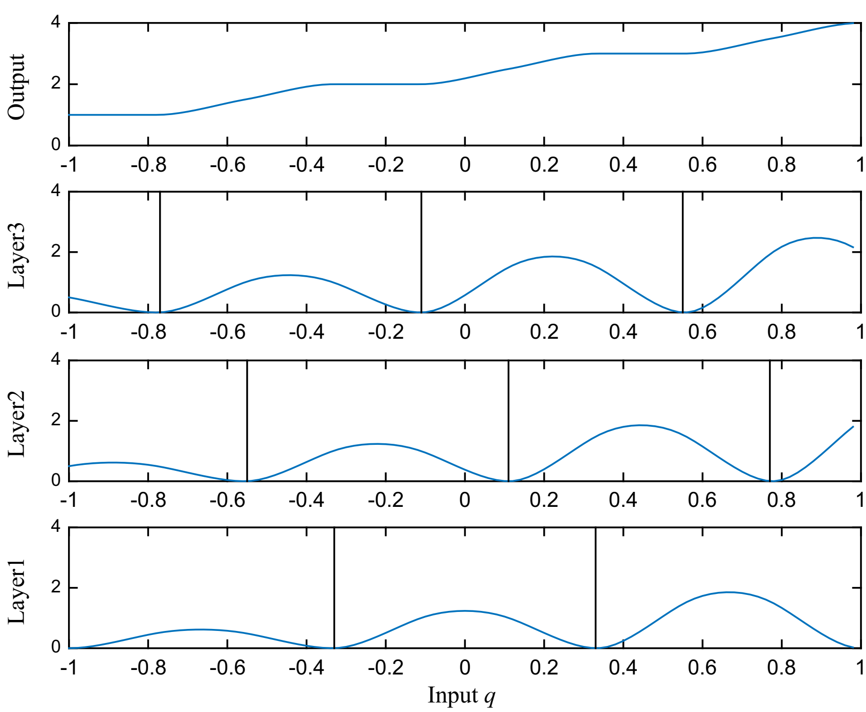

Fig. 2. Single-input spline CMAC: Three layers of cells and their normalized spline basis functions. The output is shown in the top graph.

The main limitation of the CMAC in adaptive control applications is that it often exhibits adaptive-parameter drift (*weight drift*) and bursting [18–20]. In bursting, the error at first converges to a low value while the weights continue to grow in magnitude; the weights eventually grow too large and adversely affect the control signal, causing the error to suddenly increase. Weight drift easily occurs when there are persistent oscillations in the input: for example mechanical systems with underdamped elastic degrees of freedom or any system subject to external sinusoidal disturbances. In the CMAC, this is due to oscillations between two (or more) cells and across the origin, where the weight in one cell drifts to positive values and in the adjacent cell toward negative values. Robust modifications to the training rule that can prevent weight drift (e.g. deadzone [21] and e-modification [22]) must be made large enough to affect performance in the case of CMAC with input oscillations [23]. Contrast this to RBFNs, where weight drift can be prevented in practise by choosing the variance (the *width*) of the basis functions large