Universality of Fully-Connected Recurrent Neural Networks *

Kenji Doya
doya@crayfish.ucsd.edu
Department of Biology
University of California, San Diego
La Jolla, CA 92093-0322, USA

February 1, 1993

Abstract

It is shown from the universality of multi-layer neural networks that any discrete-time or continuous-time dynamical system can be approximated by discrete-time or continuous-time recurrent neural networks, respectively.

1 Introduction

It is widely known that multi-layer neural networks with a single hidden layer are universal function approximators. That is, any continuous function can be approximated to any degree of accuracy if there are enough hidden units (see, for example, [5] and the references therein). This universality theorem gives a theoretical basis for the applications of multi-layer neural networks to various problem domains. Accordingly, we may expect that recurrent neural networks are universal approximators of dynamical systems. Actually, it is easy to show that a two-layered network with delay elements in the feedback loop (Figure 1 (a)) can approximate any dynamical system because a two-layer network can approximate any map or vector field. Such multi-layer recurrent networks have been used in modeling sequence learning [6] and more elaborate architectures have been considered for modeling various classes of dynamical systems [7]. Note that the state of such networks is updated sequentially from the input layer to the output layer.

Another approach to modeling dynamical systems is the use of fully connected recurrent networks [2, 8, 9, 10]. In these models, each unit has either a discrete or continuous time delay and is updated in parallel. If such units are used to implement the nodes in the two-layer recurrent network (as in Figure 1 (a), but now all nodes have delays), the performance of the network is not obvious. In a discrete-time case, it takes two time-steps for the state vector

*submitted to: *IEEE Transactions on Neural Networks*

$x(t)$ to be updated, thus the states at odd and even time steps will evolve independently of each other. Furthermore, in a continuous-time case, it is not clear that a given vector field can be implemented by a hidden layer with continuous time delays, in which the units are usually leaky integrators.

In this article, we will show that any dynamical system can be approximated by a recurrent network in which all the units work in parallel with the same time delays, either discrete-time or continuous-time. The result indicates that a distributed representation of the state space is the key to the implementation of a complex dynamics by a single-layer mapping. Performance analysis of this kind of architecture is important for hardware implementations in which all the elements work in parallel with their inherent time-delay characteristics.

2 Discrete-Time Dynamics

In this section, we consider the approximation of a m dimensional discrete-time dynamical system (map)

$$\boldsymbol{x}(t+1) = F(\boldsymbol{x}(t), \boldsymbol{u}(t)), \tag{1}$$

where $\boldsymbol{x}(t) \in \mathbf{R}^m$ is a state vector, $\boldsymbol{u}(t) \in \mathbf{R}^l$ is an input vector, and $F(\,) : \mathbf{R}^m \times \mathbf{R}^l \to \mathbf{R}^m$ is a smooth map. We try to approximate this system by a discrete-time recurrent network model

$$\boldsymbol{y}(t+1) = S(W\boldsymbol{y}(t) + V\boldsymbol{u}(t)), \tag{2}$$

where $\boldsymbol{y}(t) \in R^n$ represents the outputs of the units, W and V are connection matrices, and $S(\,) : \mathbf{R}^n \to \mathbf{R}^n$ is a component-wise squashing function, for example, $S(\boldsymbol{x})_i = 1/(1 + \exp[-x_i])$. Apparently, this model can implement a very limited class of n dimensional dynamical systems.

From the universality theorem for a two-layer network [5], it is possible to approximate any map F by a two-layer network with m linear output units and n squashing hidden units for some large n. That is, for some matrices W_1 $(n \times m)$, W_2 $(m \times n)$, and V $(n \times l)$, we have

$$W_2 S(W_1\boldsymbol{x}(t) + V\boldsymbol{u}(t)) \simeq F(\boldsymbol{x}(t), \boldsymbol{u}(t)). \tag{3}$$

Therefore we can approximate the dynamics of (1) by

$$\boldsymbol{x}(t+1) = W_2 S(W_1\boldsymbol{x}(t) + V\boldsymbol{u}(t)). \tag{4}$$

If we take

$$\boldsymbol{y}(t+1) \equiv S(W_1\boldsymbol{x}(t) + V\boldsymbol{u}(t)) \tag{5}$$

as a new state variable, then we have $\boldsymbol{x}(t) = W_2\boldsymbol{y}(t)$ and therefore the network dynamics is expressed in terms of $\boldsymbol{y}$ as

$$\boldsymbol{y}(t+1) = S(W_1 W_2\boldsymbol{y}(t) + V\boldsymbol{u}(t)). \tag{6}$$

This dynamics can be implemented by a recurrent network of the form (2) with an $n \times n$ connection matrix $W = W_1 W_2$. This is a consequence of the linearity of the time-delay operation. The dynamics of $\boldsymbol{x}(t)$ is unaffected by the exchange of the order of time delay and matrix operation, as shown in Figure 1 (b).

Note that the rank of the $n \times n$ matrix $W_1 W_2$ is m or lower. When the mapping $F(\,)$ is complex enough, we can assume $n > m$. In such a case, for a fixed value of the input $\boldsymbol{u}$, the state vector $\boldsymbol{y}(t)$ of the network model (6) is constrained in a m-dimensional surface in the n dimensional space. This apparently redundant, distributed representation of the state space is essential for realizing a complex state transition by a single-layer mapping.

If we want to explicitly retrieve the state $\boldsymbol{x}(t)$ of the original system (1), a linear matrix operation $\boldsymbol{x}(t) = W_2 \boldsymbol{y}(t)$ must be additionally carried out. The resulting network will have a similar architecture to a "simple recurrent network" [3].

3 Continuous-Time Dynamics

In this section, we investigate if an arbitrary continuous-time dynamical system (vector field)

$$\frac{d}{dt}\boldsymbol{x}(t) = F(\boldsymbol{x}(t), \boldsymbol{u}(t)) \tag{7}$$

can be approximated by a continuous-time recurrent network of the form

$$\tau \frac{d}{dt}\boldsymbol{y}(t) = -\boldsymbol{y}(t) + S(W\boldsymbol{y}(t) + V\boldsymbol{u}(t)). \tag{8}$$

The idea is similar to the discrete-time case. First, we rewrite (7) in terms of leaky integration as

$$\tau \frac{d}{dt}\boldsymbol{x}(t) = -\boldsymbol{x}(t) + [\boldsymbol{x}(t) + \tau S(W\boldsymbol{y}(t) + V\boldsymbol{u}(t))],$$

and approximate the input to the leaky integrator by a two-layer network. For some large n and matrices W_1 $(n \times m)$, W_2 $(m \times n)$, and V $(n \times l)$, we have

$$W_2 S(W_1 \boldsymbol{x}(t) + V\boldsymbol{u}(t)) \simeq \boldsymbol{x}(t) + \tau F(\boldsymbol{x}(t), \boldsymbol{u}(t)). \tag{9}$$

Thus we can approximate (7) by

$$\tau \frac{d}{dt}\boldsymbol{x}(t) = -\boldsymbol{x}(t) + W_2 S(W_1 \boldsymbol{x}(t) + V\boldsymbol{u}(t)). \tag{10}$$

Because a leaky integrator is a linear filter, we can move the linear transformation by W_2 to the output side of the filter. We take a new variable $y(t)$ defined as the solution of the differential equation

$$\tau \frac{d}{dt}\boldsymbol{y}(t) = -\boldsymbol{y}(t) + S(W_1 \boldsymbol{x}(t) + V\boldsymbol{u}(t)) \tag{11}$$

From (10), we have

$$\tau \frac{d}{dt}\boldsymbol{x}(t) + \boldsymbol{x}(t) = W_2(\tau \frac{d}{dt}\boldsymbol{y}(t) + \boldsymbol{y}(t))$$

and hence

$$\tau \frac{d}{dt}(\boldsymbol{x}(t) - W_2\boldsymbol{y}(t)) = -(\boldsymbol{x}(t) - W_2\boldsymbol{y}(t)). \tag{12}$$

Therefore, $W_2\boldsymbol{y}(t)$ approaches $\boldsymbol{x}(t)$ exponentially with time constant τ. Assuming $\boldsymbol{x}(t) = W_2\boldsymbol{y}(t)$, we can represent the dynamics of $\boldsymbol{y}(t)$ as follows.

$$\tau \frac{d}{dt}\boldsymbol{y}(t) = -\boldsymbol{y}(t) + S(W_1W_2\boldsymbol{y}(t) + V\boldsymbol{u}(t)). \tag{13}$$

This is a fully connected continuous-time recurrent neural network of the form (8) with connection weight W_1W_2.

A slightly different continuous-time model is obtained by exchanging W_1 and the leaky integration in (10). If we take

$$\boldsymbol{z}(t) \equiv W_1\boldsymbol{x}(t) \tag{14}$$

as a new state variable, we have from (10)

$$\tau \frac{d}{dt}\boldsymbol{z}(t) = W_1\tau \frac{d}{dt}\boldsymbol{x}(t) = -W_1\boldsymbol{x}(t) + W_1W_2S(W_1\boldsymbol{x}(t) + V\boldsymbol{u}(t))$$

and therefore

$$\tau \frac{d}{dt}\boldsymbol{z}(t) = -\boldsymbol{z}(t) + W_1W_2S(\boldsymbol{z}(t) + V\boldsymbol{u}(t)). \tag{15}$$

This is another form of continuous-time models considered in [4, 2].

4 Discussion

Using the universality theorem of two-layer networks, we showed that any discrete or continuous-time dynamical system can be modeled by a fully-connected discrete or continuous-time recurrent network, respectively, provided the network consists of enough units. However, it does not mean that such approximations can be readily achieved by learning schemes like error gradient descent. It has been shown in many empirical studies that multi-layer networks can approximate fairly complex mappings using back-propagation learning. In contrast, relatively few cases have been reported showing the successful learning of complex dynamics by fully connected recurrent neural networks. One of the reasons for this is the cost for computing the error gradient in large scale recurrent networks. However, another important aspect is the characteristics of the error surface in the parameter space. It is possible in a nonlinear dynamical system that a small change in a parameter can elicit a qualitative change of the behavior of the system, which is known as a bifurcation. The problems of gradient based learning with regard to bifurcations of the network dynamics are considered in [1, 9].

Acknowledgments

The author thanks Gary Cottrell, John Lewis, and Fu-Sheng Tsung for their helpful comments. Part of this study was supported by ONR grant N00014-91-J-1720.

References

[1] K. Doya. Bifurcations in the learning of recurrent neural networks. *Proceedings of 1992 IEEE International Symposium on Circuits and Systems*, 6:2777–2780, 1992.

[2] K. Doya and S. Yoshizawa. Adaptive neural oscillator using continuous-time back-propagation learning. *Neural Networks*, 2:375–386, 1989.

[3] J. L. Elman. Finding structure in time. *Cognitive Science*, 14:179–211, 1990.

[4] J. J. Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of National Academy of Science*, 81:3088–3092, 1984.

[5] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4:251–257, 1991.

[6] M. I. Jordan. Serial order: A parallel distributed processing approach. In J. L. Elman and D. E. Rumelhart, editors, *Advances in Connectionist Theory: Speech*. Erlbaum, 1989.

[7] K. Narendra and K. Parthasarathy. Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, 1:4–27, 1990.

[8] B. A. Pearlmutter. Learning state space trajectories in recurrent neural networks. *Neural Computation*, 1:263–269, 1989.

[9] F. J. Pineda. Dynamics and architecture for neural computation. *Journal of Complexity*, 4:216–245, 1988.

[10] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1:270–280, 1989.

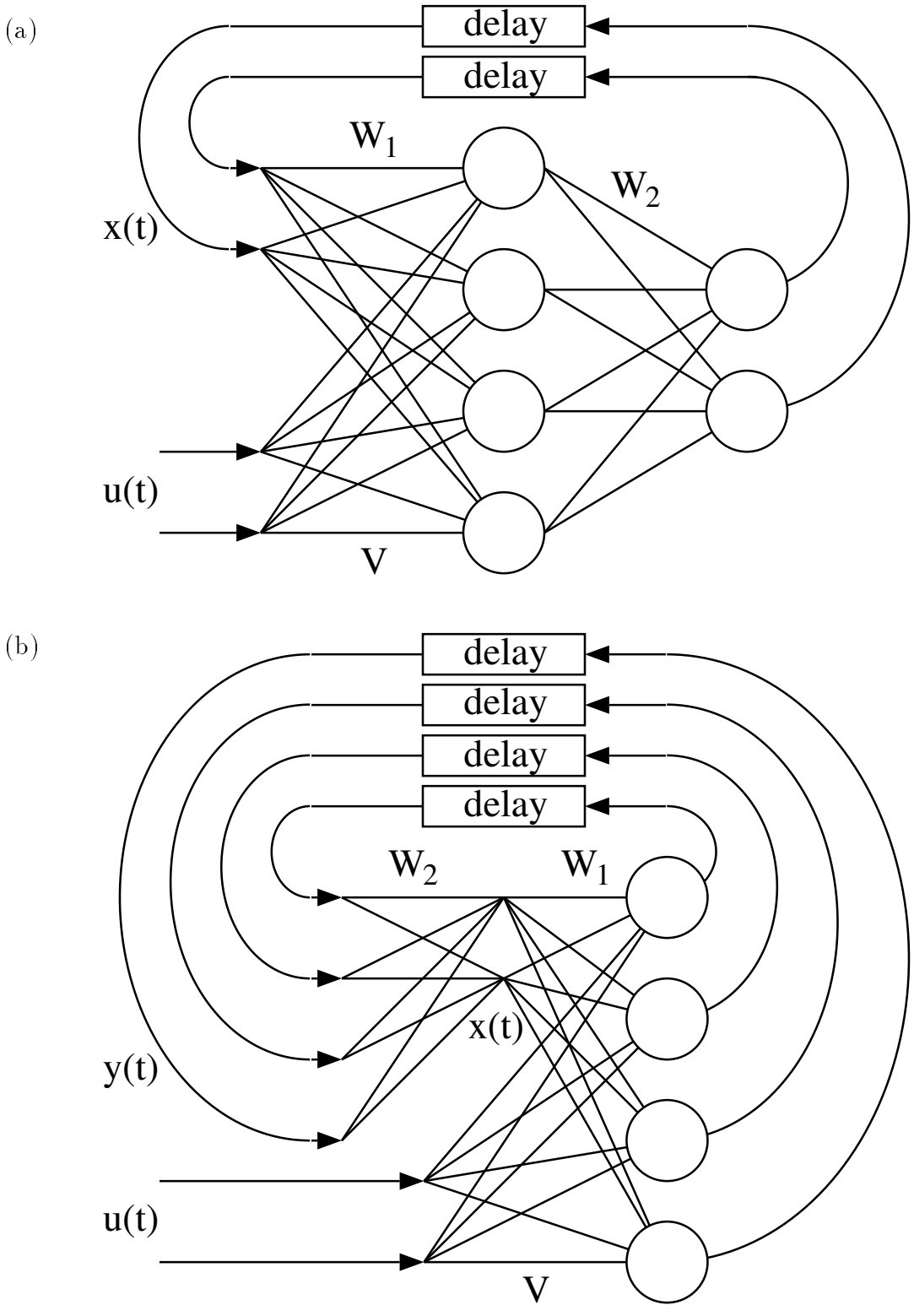

Figure 1: (a) A two-layer recurrent neural network is a universal approximator of dynamical systems. (b) A two-layer network with linear output units can be transformed to a single-layer network after exchanging the order of the time delay and a matrix operation.