Approximationg many valued mappings using a recurrent neural network

著者	Tomikawa Y., Nakayama Kenji
journal or publication title	IEEE&INNS Proc. of IJCNN'98, Anchorage
volume	2
page range	1494-1497
year	1998-05-01
URL	http://hdl.handle.net/2297/6814

Approximating Many Valued Mappings Using A Recurrent Neural Network

Yoshihiro Tomikawa, R&D Division, Y KK Corporation.*
Kenji Nakayama, Faculty of Eng., Kanazawa Univ.**

Abstract— **In this paper, a recurrent neural network(RNN) is applied to approximating one to N many valued mappings. The RNN described in this paper has a feedback loop from an output to an input in addition to the conventional multi layer neural network(MLNN). The feedback loop causes dynamic output properties. The convergence property in these properties can be used for this approximating problem.**

In order to avoid conflict. by the overlapped target data $y*$s to the same input $x*$, the input data set $(x*, y*)$ and the target data $y*$ are presented to the network in learning phase. By this learning, the network function $f(x,z)$ which satisfies $y* = f(x*, y*)$ is formed. In recalling phase, the solutions y of y = $f(x, y)$ are detected by the feedback dynamics of RNN. The different solutions for the same input x can be gained by changing the initial output value of y.

It have been presented in our previous paper that the RNN can approximate many valued continuous mappings by introducing the differential condition to learning. However, if the mapping has discontinuity or changes of value number, it sometimes shows undesirable behavior. In this paper, the integral condition is proposed in order to prevent spurious convergence and to spread the attractive regions to the approximating points.

Keywords- **Recurrent Neural Network, Many Valued Mapping, Convergence Property.**

I. Introduction

The human beings can perceive different things from the same information. It is known to be caused by the feedback from knowledge. Rumelhart et al applied a Hopfield type of a neural network to the recognition problem of Necker cube[1]. This network is a type of recurrent neural networks(RNN).

In this paper, it is proposed how to apply a RNN to approximating one to N many valued mappings. It is well known as universal approximation theory [2]that a Multi-layer neural network(MLNN) can approximate single valued continuous mappings. However, since a feed-foward network such as a MLNN can not take different values for the same input, it is difficult to apply it to many valued mapping problems.

Whereas, a. RNN has a convergence property. It can converge to different points even for the same input. This property can be applied to approximating a many valued mapping. In our previous paper[3], it have been proposed that a RNN can approximate many valued continuous mappings which does not have changes of value number. However, it have been also indicated that it shows chaotic behavior or spurious convergence in the case of a many valued mapping with changes of value number. In general, it is difficult to control the convergence property of a RNN.

In this paper, the integral condition is proposed in order to prevent such undesirable behavior. By introducing this condition to learning, a RN N can be extended to approximating discontinuous mappings or many valued mappings with changes of value number.

*200, Yoshida, Kurobe, 938 **Japan.** E-mail:tomy@rd.ykk.co.jp.
2-40-20, Kodatsuno, Kanazawa, **920 Japan. **E-mail: nakayama** @ec.t.Kanazawa-u.ac.jp.

II. Learning and recalling method

The learning method of many valued mappings have already been proposed[3][4]. In this section, the main learning method and recalling method are described.

Let's consider the learning of the data sets $(x*, y*)$ in which different output $y*$s may be given for the same input $x*$. Since a conventional feed-forward type of neural network takes a only single output for the same input, it can not learn such the data sets. Then, we propose the learning method which lets a MLNN to learn the following data relations.

$$input\ \ vector\ \ (x*, y*) \rightarrow output\ \ y* \qquad (1)$$

In this learning, the output $y*$ is always different for the different $y*$ in the input vector even though the $x*$ in the input vector is the same. Then, the relation of Eq.(l) can be learned by a. conventional MLNN. In this case, the learned output function of the network satisfies the following equation.

$$y^* = f(x*, y^*) \qquad (2)$$

where $\boldsymbol{f}(x, z)$ is the output function of the MLNN with the input x and z.

A. Recalling by a RNN

In recalling, the network output y is feed-backed to the input z as shown in Fig.(1). A neural network which has this feedback operation is called a. R.NN. This operation is expressed by the following equation.

$$y(n) = f\,(x, y(n-1)) \quad n \rightarrow \infty \qquad (3)$$

0-7803-4859-1/98 $10.0001998 IEEE

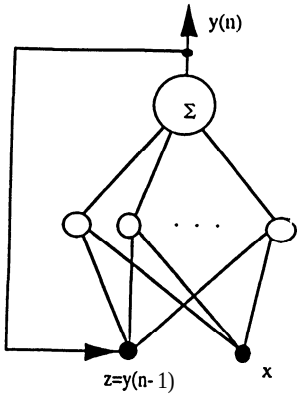


Fig. 1. A RNN model for recalling

This feedback operation is repeated until the output $y(n)$ does not change. This unchanged state means that the network have been converged. This type of RNN usually takes many convergent states.

B. Differential condition

If the network state of this RNN is at the point which satisfies Eq.(2), the state never change. This point is called an equilibrium point. However, the network state is not always guaranteed to converge to an equilibrium point. In order to converge to an equilibrium point, the point must satisfy an asymptotic stable condition. In this RNN, the asymptotic stable condition at the point $(x*, y*)$ can be expressed as follows

$$\frac{\partial f(x*, y*)}{\partial z} < 1 \tag{4}$$

This condition is called the differential condition in this paper, since it uses a differential form.

C. Learning by the method of the steepest descent

The parameters of a MLNN, such as connection weights and thresholds, sometimes can be adjusted by the back propagation method. This learning method minimizes the total square error between the network output and the target dada. This back propagation method is principally based on the method of the steepest descent.

The proposed learning method uses the method of the steepest descent but it minimizes the integrated total error E_{total} of the network output and the differential condition. This total errors E_{total} is written as follows, by using the output error E_{out} and the differential error E_{diff}.

$$E_{total} = E_{out} + P\&if\_f \tag{5}$$

where β is a small positive constant. Since the differential condition does not have to be zero as shown in Eq.(4), the weight coefficient β is used.

Using the given data $x*$ and $y*$, the output error is written by the following equation.

$$E_{out} = \sum \{y* - f(x*, y*)\}^2 \tag{6}$$

And the differential error is written as follows,

$$E_{diff} = \sum \{\frac{\partial f(x*, y*)}{\partial z}\}^2 \tag{7}$$

where $\sum$ means the summation of a.11 the given learning data..

By the method of the steepest descent, the network parameters are updated as follows,

$$w(n)_{ij}^{(L)} = w(n-1)_{ij}^{(L)} - \eta \frac{\partial E_{total}(n-1)}{\partial w_{ij}^{(L)}} \tag{8}$$

$$\theta(n)_i^{(L)} = \theta(n)_i - \eta \frac{\partial E_{total}(n-1)}{\partial \theta_i^{(L)}} \tag{9}$$

where $w(n)_{ij}^{(L)}$ is the weight of the i-th neuron for the j-th input in the L-th layer at the n-th update. $\theta_i^{(L)}$ is the threshold. η is the learning constant.

III. CONTROL OF SPURIOUS CONVERGENCE

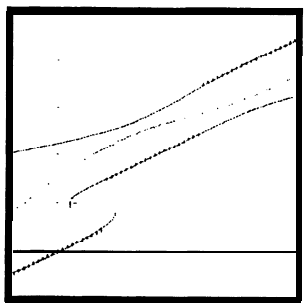
If the given data do not have discontinuity, a. RNN can approximate them by the learning method described in Section 2. However, in the case of approximating a discontinuous mapping or a mapping with value changes, the RNN sometimes converges to the unlearned points, that is a. spurious convergence. Figure 2 shows the convergence points from the uniform initial values $y(0)$ by the RNN, which learns the points "*" by the learning method described in section *2*. The horizontal coordinate shows the input x and the vertical coordinate shows the converged output y.

Figure 2.(a) shows the example of a. discontinuous mapping. Many spurious convergence points appeared near the discontinuous parts. Figure 2. (b) shows the example of a. mapping with value changes. The spurious convergence points appeared along the prolongation of the center line. This results show that the learning method described in section 2 is not efficient to control the network convergence for approximating many valued mappings.

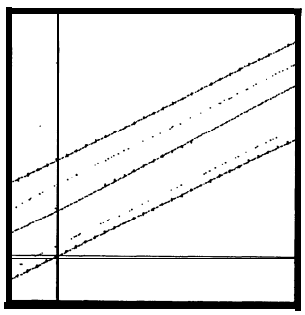
A. Integral condition

In this section, the integral condition is proposed in order to enlarge the attractive region to the learned point and to control spurious convergence.

Figure(3) shows the distribution of discontinuous mapping data which are learned by the proposed network. The mesh shows the learned output function for



(a) Approximation of a discontinuous mapping using only differential condition.(β=0.01)



(b) Approximation of a many valued mapping with changes of value number using only differential condition.(β=0.02)

Fig. 2. Spurious convergence by a R.NN

the input x $-$ z. In learning of Eq.(1), the input data $(x*, y*)$ is always a local graph in the $x-z$ space as shown in this figure. Then, the input, data exist only in the restricted partial space. Whereas, the attractive property of the RNN is attributed to the network output function for all input space.

The differential condition of Eq.(4) means the flat form of the network function near the data points in Fig.(S). However, in this case, the flat form appears in the region far from the data. points. In order to reduce spurious convergence, the output function in the region far from the data points must be changed.It is difficult to control the output function in this part only by local learning data.

Then, we propose the learning condition which sets the network output to be y* also for all z except y*. The error function of this condition can be written by the following equation.

$$E_{int} = \sum \int e^{-\frac{(y*-z)^2}{\sigma}} \{y* - f(x*, z)\}^2 dz \rightarrow 0 \quad (10)$$

This condition is called the integral condition in this paper, since it uses a. integral form.

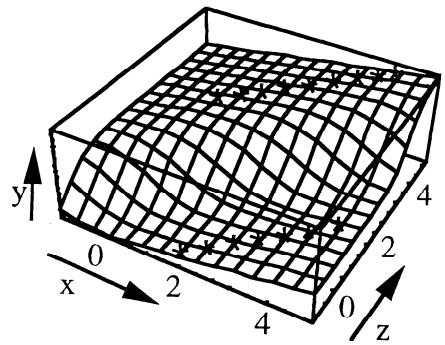


Fig. 3. Local learning data(*) in the $x-z$ input space and network output function

In the equation (IO), the square factor means the error between the purpose output $y*$ and the network output $f(x*, z)$. The exponential factor means the weight to attach the more importance to the nearer z to y*. By the effect of this exponential factor: the integral condition enlarges the attractive region to the learned value $y*$ nearer to an initial value z, even though many $y*$s exist for the same $x*$. The parameter σ means the spread of the weight.

B. Auxiliary data

A RNN could learn the integral condition by the method of steepest descent, if the integral error E_{int} could be included in the integrated total error E_{total} as follows.

$$E_{total} = E_{out} + \beta E_{diff} + \gamma E_{int} \quad (11)$$

where γ is a. small constant,. As well as E_{diff}, E_{int} does not have to be zero. Then, the weight factor is used also for E_{int}. However: by adjusting β and γ, the convergence property delic.a.tely changes.

In general, the integral of Eq.(10) can not solve analytically. However, it can be approximated by the compositions for discrete z_n s.

$$E_{int} \simeq \Delta z \sum_{i}^{N} \sum e^{-\frac{(y*-z_n)^2}{\sigma}} \{y* - f(x*, z_n)\}^2 \quad (12)$$

If the exponential term is 1, the error E_{int} can be consider to be the output error when the input vector

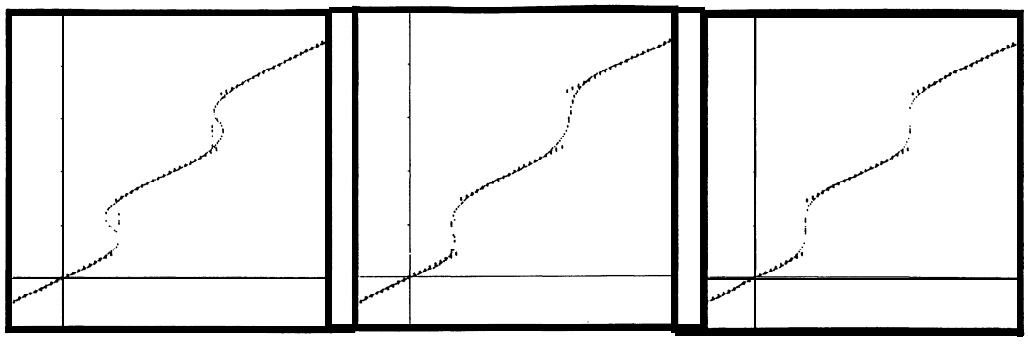


Fig. 4. Results for approximation **of a** discontinuous mapping

$(x*, z_n)$ and the target data $y*$ are presented. Besides the exponential term can be thought the weight factor to this learning data..

Then, in this paper, the integral condition is considered to be given by an auxiliary data. weighted by the exponential term and γ.

IV. Simulation

Figure 4 shows the approximation results for the discontinuous line data, using the RNN trained by the learning method in section 2 and the integral condition in section 3. It shows the convergence points from the uniform initial values $y(0)$ as the same as Fig.%.(a). By including the integral condition in learning, spurious convergence points could be well reduced. Increasing the weight factor γ further, the approximation of a discontinuous mapping could be realized. Especially the discontinuous parts could be well reproduced.

Figure 5 shows the approximation result for the many valued mapping with changes of value number. In this case, spurious convergence points could be also reduced by using the integral condition. However, if the weight factor β was strong, spurious convergence points were left near the parts of value number change. Whereas, the weight factor γ was stronger, the precise of approximation became worse. Figure 5 shows the best results of our simulations. The adjusting of the weight factors β and γ is very important in the proposed method but it is the future problem.

V. Conclusion

Iii this paper, the approximation method of many valued mappings using a. recurrent neural network is introduced. And the integral condition is proposed to extend it to approximating discontinuous mappings or many valued mappings with changes of value number. It is confirmed that the integral condition can well reduce spurious convergence of RNN and that a recurrent neural network can be applied to approximation problem of a. discontinuous mapping and a. many valued mapping with changes of value number.

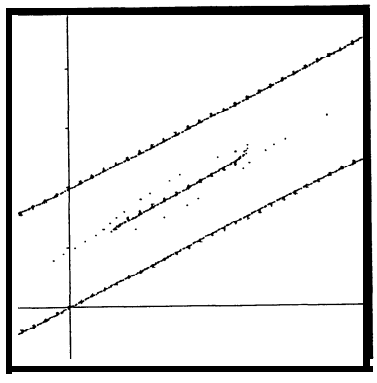


Fig. 5. Results for approximation **of a** many **valued** mapping **with changes of value** number

References

[1] D.E.Rumelhart,J.L. McClelland, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition,I&II.*, MITPress, Cambridge MA, 1986.

[2] S.Haykin, *Neural Networks.* IEEE PRESS, 1994

[3] Y. Tomikawa, H.Akiyama, K. Nakayama: *A Multilayer Neural Network With A Feedback Loop Approximating Many-Valued Functions.* Technical report of IEICE., NC95-167, pp99-106, 1996. (In Japanese)

[4] Y. Tomikawa, K. Nakayama: *A Multilayer Neural Network With A Feedback Loop Approximating Many-Valued Functions.* Trans. of IEICE., D-II,No.9, pp2493-2501, 1996. (In Japanese)