The Time Complexity Analysis for a Kind of Neural Network with Rational Spline Functions

Daiyuan ZHANG & Hongyan BAO
College of Computer, Nanjing University of Posts and Telecommunications, Nanjing, China

ABSTRACT: Based on the results of weight function neural networks, the time complexity for a kind of neural network with rational spline functions is analyzed. We derive the complexity results by interpolation theories and algorithms. We find a linear relationship between the time complexity and the number of patterns. We also conclude that the time complexity is proportional to the input and output dimensions of the neural network. It shows that the results proposed in this paper have many potential applications.
KEYWORD: Neural Network; Rational Spline Function; Weight Function; Time Complexity

1 INTRODUCTION

In engineering practice and scientific experiments, we often require a set of observation data (x_i, y_i), $i = 0,1,2,\cdots,N+1$ from some experiments, which reveals the relationship between independent variant *x* and dependent variables *y*. It can generally be approximated by a function $y = f(x)$.

Neural network can be used to find the approximate relationship between *x* and *y*. Literature [1]puts forward the fundamental theory for a new kind of neural networks based on cubic spline weight functions. However, with further research we find that polynomial interpolation method may not meet all the requirements. So we use rational interpolation method in this paper.

2 NETWORK'S STRUCTURE

The neural network is shown in Figure 1.

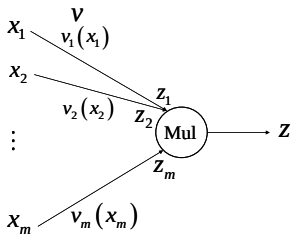

Figure 1. The neural network with only one output node

Let $\boldsymbol{x}_i$ be the *m* dimensional input of the *i*-th component, it can be written in the following $\boldsymbol{x}_i = \left(x_{i0}, x_{i1}, \cdots, x_{i(N+1)}\right)$. At the same time, the corresponding output vector is denoted as $\boldsymbol{z} = \left(z_0, z_1, \cdots, z_{N+1}\right)$. So that

$$\begin{aligned}\boldsymbol{z}_i &= \left(v_i\left(x_{i0}\right)\ v_i\left(x_{i1}\right)\cdots v_i\left(x_{i(N+1)}\right)\right)\\ &= \left(z_0^{\rho_i}\ z_1^{\rho_i}\ \cdots\ z_{N+1}^{\rho_i}\right)\end{aligned} \tag{1}$$

The corresponding weight function interpolation point can be expressed as

$$\boldsymbol{Ip}_i = \left\{\left(x_{i0}, z_0^{\rho_i}\right), \left(x_{i1}, z_1^{\rho_i}\right), \cdots, \left(x_{i(N+1)}^{\rho_i}, z_{N+1}^{\rho_i}\right)\right\} \tag{2}$$

So the weight function $s_i(x_i)$ can be obtained by the method of interpolation.

3 COMPLEXITY ANALYSIS

3.1 *Fundamentals of Training algorithm*

According to the theory of rational spline structure [2-3], the 3/2 rational spline weight function can be written as follows:

$$R(x) = \frac{P(x)}{Q(x)} \tag{3}$$

Let $\theta = (x - x_i)/h_i$, $h = x_{x+1} - x_i$, from equation (3) the following holds

$$\begin{aligned}P(x) = &(1-\theta)^3 f_{i-1} + (1-\theta)^2 \theta\left(r_i f_{i-1} + h_i t_{i-1}\right)\\ &+ (1-\theta)\theta^2\left(r_i f_i - h_i t_i\right) + \theta^3 f_i\end{aligned} \tag{4}$$

$$Q(x)=(1-\theta)^2+(1-\theta)\theta(r_i-1)+\theta^2 \tag{5}$$

where

$$\begin{cases} f(x_i)=f_i \\ f(x_{i+1})=f_{i+1} \\ f'(x_i)=t_i \\ f'(x_{i+1})=t_{i+1} \end{cases} \tag{6}$$

On the basis of Hermite interpolation theorem, we have

$$R_i(x)=\sum_{j=0}^{3} f(x_j)\varphi_{ij}(x) \tag{7}$$

where

$$\begin{cases} \varphi_{i0}(x)=\dfrac{(1-\theta)^2((r_i-1)\theta+1)}{(1-\theta)^2+(1-\theta)\theta(r_i-1)+\theta^2} \\ \varphi_{i1}(x)=\dfrac{\theta^3+\theta^2(1\text{-}\theta)r_i}{(1-\theta)^2+(1-\theta)\theta(r_i-1)+\theta^2} \\ \varphi_{i2}(x)=\dfrac{\theta(1-\theta)^2 h_i}{(1-\theta)^2+(1-\theta)\theta(r_i-1)+\theta^2} \\ \varphi_{i3}(x)=\dfrac{-h_i(1-\theta)\theta^2}{(1-\theta)^2+(1-\theta)\theta(r_i-1)+\theta^2} \end{cases} \tag{8}$$

Using $R''_{i-1}(x_i-0)=R''_i(x_i+0)$, the result is

$$\begin{aligned} &\frac{(2r_i^3-2r_i^2-16r_i-16)}{h_{i-1}} f_{i-1} \\ &+\frac{(-6r_i^3+24r_i^2-14r_i-6)}{h_{i-1}} f_i \\ &+\frac{2}{h_{i-1}} t_{i-1}+\frac{2r_i^2-12r_i+10}{h_{i-1}} t_i \\ &=0+\frac{2r_i}{h_i} f_{i+1}+\frac{-2(r_i-1)}{h_i} t_i+\frac{-2}{h_i} t_{i+1} \end{aligned} \tag{9}$$

By simplifying, the equation can be obtained as follows:

$$\begin{aligned} &\frac{2}{h_{i-1}} t_{i-1}+\left(\frac{2r_i^2-12r_i+10}{h_{i-1}}+\frac{2(r_i-1)}{h_i}\right) t_i+\frac{2}{h_i} t_{i+1} \\ &=\frac{-2r_i^3+2r_i^2+16r_i+16}{h_{i-1}} f_{i-1} \\ &\quad+\frac{6r_i^3-24r_i^2+14r_i+6}{h_{i-1}} f_i+\frac{2r_i}{h_i} f_{i+1} \end{aligned} \tag{10}$$

Eq.(10) can be written as a linear equation

$$At=C \tag{11}$$

Where

$$A=\begin{pmatrix} q_0 & q_1 & 0 & \cdots & 0 & 0 & 0 \\ \dfrac{2}{h_0} & H_1 & \dfrac{2}{h_1} & \cdots & 0 & 0 & 0 \\ 0 & \dfrac{2}{h_1} & H_2 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \dfrac{2}{h_{N-1}} & H_N & \dfrac{2}{h_N} \\ 0 & 0 & 0 & \cdots & 0 & q_N & q_{N+1} \end{pmatrix} \tag{12}$$

$$\begin{aligned} C_i=&\frac{-2r_i^3+2r_i^2+16r_i+16}{h_{i-1}} f_{i-1} \\ &+\frac{6r_i^3-24r_i^2+14r_i+6}{h_{i-1}} f_i \\ &+\frac{2r_i}{h_i} f_{i+1}, i=1,2,...,N \end{aligned} \tag{13}$$

$$H_i=\frac{2r_i^2-12r_i+10}{h_{i-1}}+\frac{2(r_i-1)}{h_i}, i=1,2,\cdots,N \tag{14}$$

Decompose formula (12), we know that L is a lower triangular matrix, and U is a unit upper triangular matrix. L and U are as follows:

$$L=\begin{pmatrix} \alpha_1 & 0 & \cdots & 0 & 0 \\ \gamma_2 & \alpha_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \alpha_N & 0 \\ 0 & 0 & \cdots & \gamma_{N+2} & \alpha_{N+2} \end{pmatrix} \tag{15}$$

$$U=\begin{pmatrix} 1 & \beta_1 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & \beta_N \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \tag{16}$$

and

$$\begin{cases} \dfrac{2r_i^2-12r_i+10}{h_{i-1}}+\dfrac{2(r_i-1)}{h_i}=\gamma_i\beta_{i-1}+\alpha_i \\ q_0=\alpha_1 \\ q_1=\alpha_1\beta_1 \\ \dfrac{2}{h_i}=\gamma_i \\ \dfrac{2}{h_i}=\alpha_i\beta_i \end{cases} \tag{17}$$

Thus,

$$\beta_i = \frac{\frac{2}{h_i}}{\frac{2r_i^2 - 12r_i + 10}{h_{i-1}} + \frac{2(r_i - 1)}{h_i} - \frac{2}{h_i}\beta_{i-1}} \tag{18}$$

3.2 *Analysis of the time complexity*

When using Thomas algorithm for solving trigonometric group, we set $Ut = m$, then get $Lm = C$ where $At = C$. C_i is showed in (13).

After the analysis of β_1 and β_i we know that it needs $5N + 5$ additions and subtractions, $10N + 11$ multiplications and divisions. After the analysis of $Lm = C$, it shows that it needs $6N + 6$ additions and subtractions, $10N + 11$ multiplications and divisions. After the analysis of $Lm = C$ we know that it needs $N + 1$ additions and subtractions, $N + 1$ multiplications and divisions.

So altogether it needs $12N + 12$ additions and subtractions, $21N + 23$ multiplications and divisions.

For (13), it needs $8N$ additions and subtractions, $19N$ multiplications and divisions.

In summary, we find that it requires $40N + 23$ multiplications and divisions, $20N + 12$ additions and subtractions.

It is well known that the running time of the algorithm depends on the basic arithmetic like addition, subtraction, the size of the data, computer hardware, operation environment and so on. The main factor for the time of algorithm is the scale of the problem. Assuming that the time of operating addition is t_1, making a multiplication needs t_2, and making a power operation spends t_3.

Therefore, the total time required for solving function (3) is

$$\begin{aligned}&(20N + 12)t_1 + (40N + 23)t_2 \\ &= (20t_1 + 40t_2)N + 12t_1 + 23t_2\end{aligned} \tag{19}$$

Consequently, the time complexity is $O(N)$. For the second type of neural network[1] of m dimensional input and 1 dimensional output, it holds

$$z = \prod_{i=1}^{m} z_i \tag{20}$$

And $z_i = v_i(x)$. $v_i(x)$ is the second class of weight function [1]. Then through the weighted coefficient ρ_i, z_i and z are connected to $z_i = z^{\rho_i}$, and

$$z = v_i^{-\rho_i}(x_i) \tag{21}$$

Therefore the total time of m dimensional input and 1 dimensional output is

$$T(N) = \left[(20t_1 + 40t_2)N + 12t_1 + 23t_2 + t_3\right]m \tag{22}$$

The total time of m dimensional input and n dimensional output is

$$T(N) = \left[(20t_1 + 40t_2)N + 12t_1 + 23t_2 + t_3\right]mn \tag{23}$$

where t_1, t_2 and t_3 can be viewed as constants, so the complexity is $O(mnN)$.

4 SIMULATION

Cubic spline interpolation function has the properties of monotonicity preserving. The properties are proved to meet only in some special cases. And it is easy to cause unexpected turbulence. But rational spline function has a high precision in approximation of quadric surface function.

More importantly, when using the spline method to construct cubic spline curves and surfaces, the accuracy and the shape are determined for the given interpolation conditions. In the real life and production, there will be a certain gap between the resulting curve and actual demand.

Rational spline method proposed in this paper not only holds the advantages of simple structure and easily calculate, but also can improve the precision of interpolation and adjust the shape of curves and surfaces. So this method is applied to solve the issues raised above. So it is necessary to compare the time complexity of them. If the difference between the time complexity within reasonable limits, then we can use rational spline interpolation method proposed in this paper in some applications.

The results for the analysis of the complexity of cubic spline weight function neural network[4] is as follows:

The total time of m dimensional input and 1 dimensional output is

$$T(N) = \left((7N + 3)t_1 + (10N + 7)t_2\right)m \tag{24}$$

So the complexity is $O(mN)$. Therefore the asymptotic time complexity of *m* dimensional input and *n* dimensional output is $O(mnN)$.

As compared with the complexity of second types of spline weight function neural network with cubic spline function neural network[1], it is not difficult to see that both of them have the same asymptotic time complexity. The only difference is the coefficient of these two expressions.

The CPU time depends on the algorithm, the computer architecture and the operating system[5]. This paper uses MATLAB for the simulation example. Figure 2 is the results of the simulation experiments on the dimension of output samples. As can be seen from Figure 2, the CPU time between

the two training algorithms is similar, the results of the rational spline shows that the CPU time is almost linear. That is to say, the experimental results are consistent with the theoretical analysis proposed in this paper.

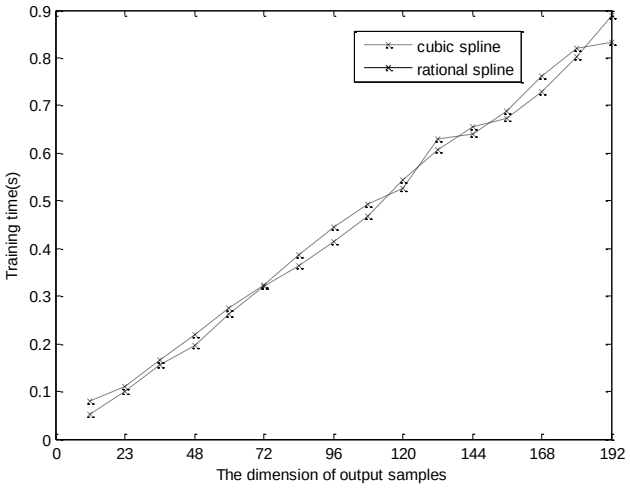

Figure 2. The relationship between the CPU time and the dimension of the output samples

5 SUMMARY AND CONCLUSIONS

This paper analyzes the complexity of second types of spline weight function neural network with 3/2 rational spline functions. The good results for the complexity obtained in this paper imply that this kind of neural network has many potential applications.

REFERENCES

[1] Zhang, D.Y. 2006. New Theories and Methods on Neural Networks. Beijing: Tsinghua University Press.

[2] Habib, Z.F.Q. Sarfraz, M.H. & Sakai, M.B. 2005. Rational Cubic Spline Interpolation with Shape Control. Computer & Graphics, 29:594-605.

[3] Duan, Q. Wang, L.Q. & Twizell, E.H. 2004. A New Weighted Rational Cubic Interpolation and its Approximation. Applied Mathematics & Computation, 168:990-1003.

[4] Liu, C.F. 2013. The Algorithm Complexity Research of Cubic Spline Weight Function Neural Network and its Application. Nangjing: Nanjing University of Posts and Telecommunications for the Degree of Master of Engineering.

[5] Zhang, D.Y. 2009. Computer Organization (Second Edition). Beijing: Tsinghua University Press.