ハイパフォーマンス
コンピューティング 84－3
（２０００．１２．８）

Derivative Expressions of Neural Networks

Tomoo AOYAMA*, Qianyi WANG, Hanxi ZHU, Umpei NAGASHIMA**

*The Faculty of Engineering, Miyazaki University
Gakuen-Kibanadai-Nishi 1-1, Miyazaki city, 889-2192
e_mail: t0b217u@cc.miyazaki-u.ac.jp
**National Institute for Advanced Interdisciplinary Research

Abstract

Derivative expressions for multi-layer neural networks are already known, however, their application fields are restricted. Because, neural networks give correct responses at learning points, but the reliability at unlearning points are not so certain. The information about the unlearning points is used in various industrial applications. The problem is arisen by the unsymmetrical character of learning equation. Since it is property of the conception of the equation, and that is inherited from the neuron system, we can't avoid the unsymmetrical character; then, we couldn't use the information enough. For complex processing, the scale of neural networks becomes large. In that case, the BP-learning is not so effective. We find a necessity to divide the neural network under reasonable CPU resource. If we divide a neural network, the learning methods among plural networks are interacted each other. The interactions should be controlled by the continuous condition and the tangential consistencies. Neuron computers that are constructed of combinations of neural networks are well worth considering as a non-Neumann's machine. The combination is not a plain assemble of independent networks, but interacted each other. To discuss the interactions, the derivative expressions for multi-layer neural networks are useful terms.

We derive the expressions until fifth order, and by using them, we introduce explicit learning equations for interacted neural networks.

階層型ニューラルネットワークの微分形

青山智夫，王謙軼，朱含希，長嶋雲兵
宮崎大学工学部電気電子工学科

概要

階層型ニューラルネットワークの微分形は既知である。しかし用途は少ない。ニューラルネットワークは学習点で正しい応答をする一方未学習点の応答の信頼性は低い。未学習点に関する情報は様々な工業的応用に使用される。この問題は学習方程式の非対称性により生じる。これは方程式の概念の性質である。それは元ニューロン系から継承された。我々はその非対称性を回避し得ない。それゆえ我々は未学習点の情報を十分に使用できない。

複雑な処理ではニューラルネットワークの規模は大きくなる。その場合BP学習は効果的でない。限りあるCPUの下ではニューラルネットワークを分割する必要がある。分割時、複数の学習方程式は相互に作用する。その相互作用は連続性、微分接続性であろう。ニューラルネットワークの結合で構築されるニューロンコンピュータは、非ノイマン機械の一つである。その結合は、独立したネットワークの単純集合ではなく相互に作用する。その相互作用を論じるために微分形は有用である。我々は5次項までの微分式を導き、それらから相互作用ネットワークの学習方程式を導いた。

1. Introduction

Derivative expressions for multi-layer neural networks have already known [1]. However, their applications are not completed yet. Because the expressions give information around learning points, therefore, if the information had not important certainly, the applied fields would be restricted [2]. Neural networks give correct responses at learning points, however, the reliability for responses around the learning points are not certain. The problem is arisen by the unsymmetrical character of BP-learning [3]. Since it is the property of BP, that is inherited from the neuron system, we can't avoid the problem; then, we must pay an attention to the information. For complex processing, the scale of neural networks is large. In that case, the BP-learning is not so effective. We find a necessity to divide the neural network under reasonable CPU resource [4]. If we divide a neural network, the learning methods among plural networks are interacted each other. The interactions should be controlled by the continuous condition and the tangential consistencies. Here, we find an application for the derivatives of neural networks.

2. Derivative of 3-layer neural network

A 3-layer neural network is a function that has plural input/output arguments. If differential functions are adopted for conductions of neurons in the networks, the differential actions of the 3-layer neural network are derived. The differential expression was already published, however, the higher order derivatives are not done; then, we show that.

The 3-layer neural network is a forward propagator, then we write the propagations as,

$$Pj=\Sigma\, Vij*Xi, \quad Yj=f(Pj),$$

where X is input datum, V is connection matrix between 1st and 2nd neuron's layers, f is neuron's function on 2nd layer, and Y is output vector of 2nd layer. The expression between 2nd and 3rd-layers is,

$$Qk=\Sigma\, Wjk*Yj, \quad Ok=g(Qk),$$

where the meanings of g and W are same as that of f and V. On the expressions, we derived differential forms that are listed in appendix A, B.

$$\delta\, Pj=Vij*\delta\, Xi, \qquad \delta\, Yj=f(\delta\, Pj)=\Sigma\, n\{f^{(n)}(Pj)*\{Vij*\delta\, Xi\}^n/n!\}=\{\textbf{Appendix A}\}$$

Where $Vij*\delta\, Xi=Uij$, and $f^{(n)}$ is nth derivative of function f.

$$\delta\, Qk=Wjk*\delta\, Yj, \qquad \delta\, Ok=g(\delta\, Qk)=\Sigma\, n\{g^{(n)}(Qk)*\{Wjk*\delta\, Yj\}^n/n!\}=\{\textbf{Appendix B}\}$$

Then, the differential expressions are $\partial\, Ok/\partial\, Xi=g^{(1)}Wf^{(1)}V=\Sigma\, j\{g^{(1)}(Qk)Wkjf^{(1)}(Pi)Vij\}$, and higher order terms are listed in **Appendix C**. The expression is defined at kth and ith neurons on 3rd and 1st layers.

3. Clustering and BP-algorithm

3.1 Traditional BP

The back-propagation is a useful algorithm, but it is applicable to small neuron systems. If we wish to investigate complex problems, the target system would be large and include

more than 10^4 neurons, and the number of learning data would be 10^5 orders. It is difficult to generate such large systems, when it sometimes requires huge CPU times until convergent on learning. For large system, the back-propagation algorithm is not practical. When it is necessary, it is natural that the learning data set is divided to some clusters, and plural neural networks are learned for each cluster. Such networks are called "multi-modal." For learning data set, the divided neural networks give correct responses; but for unlearning data, the responses are not always precise. The precision depends on reasonability of clustering.

We consider a following energy that is evaluated by a learning datum,

$$E=(1/2)\,\Sigma\,k(Ok-Tk)^2.$$

We differentiate the E-term by Wjk.

$$\partial E/\partial Wjk=(Ok-Tk)\,\partial Ok/\partial Wjk=(Ok-Tk)g^{(1)}(Qk)yj=\Delta kj,$$

If $\Delta kj\rightarrow 0$, then an iterative equation, $Wjk^{(\xi)}=Wjk^{(\xi-1)}-\varepsilon\,\Delta kj$, where ε is a small coefficient, makes a stationary matrix Wjk; that is BP-algorithm. The iteration is defined on each learning datum. If you wished to define the iteration for all learning data, the following equations would be got.

$$E=(1/2)\;\Sigma\,\lambda\,\Sigma k\,(O(\lambda)k-T(\lambda)k)^2.$$

The E-term can be derived as following, $\partial E/\partial Wjk=\Sigma\,\lambda\,(O(\lambda)k-T(\lambda)k)\,\partial O(\lambda)k/\partial Wjk$

$=\Sigma\,\lambda\,(O(\lambda)k-T(\lambda)k)g^{(1)}(Q(\lambda)k)yj=\Sigma\,\lambda\,\Delta(\lambda)kj$, then $\Sigma\,\lambda\,\Delta kj\rightarrow 0$; so we get,

$$Wjk^{(\xi)}=Wjk^{(\xi-1)}-\varepsilon\;\Sigma\,\lambda\,\Delta(\lambda)kj.$$

Similar equations are also given by Vij, and make a stationary matrix.

In case of $\partial E/\partial Vij$, we introduce following replacements.

$\{\,(Ok-Tk)\rightarrow\Sigma k\,(Ok-Tk)g^{(1)}(Qk)Wjk,\quad g^{(1)}(Qk)\rightarrow f^{(1)}(Pj),\quad yj\rightarrow xi\}$, so we get,

$$\partial E/\partial Vij=[\,\Sigma k\,(Ok-Tk)g^{(1)}(Qk)Wjk]\,f^{(1)}(Pj)\,xi=\Delta ij,\quad \Delta ij\rightarrow 0,\quad Vij^{(\xi)}=Vij^{(\xi-1)}-\varepsilon\;\Delta ij.$$

Where ε is a positive small number. For all learning data, the following iteration is obtained.

$$\Sigma\,\lambda\,\Delta ij\rightarrow 0,\quad Vij^{(\xi)}=Vij^{(\xi-1)}-\varepsilon\;\Sigma\,\lambda\,\Delta(\lambda)ij.$$

3.2 Clustering BP

We consider a link between two neural networks, "1" and "2", whose junction is defined at a specify datum that is written by μ. At the μ-point, the outputs from the two neural networks must be equivalent each other; then, the difference E is define as,

$$E=(1/2)\Sigma k(O1(\mu)k-O2(\mu)k)^2.$$

The E-term must be zero when the junction is completed for infinitesimal change of W1jk,

$$\partial E/\partial W1jk=(O1(\mu)k-O2(\mu)k)\,\partial O1(\mu)k/\partial W1jk$$

$$=(O1(\mu)k-O2(\mu)k)g1^{(1)}(Q1(\mu)k)y1j=\Delta 1(\mu)kj.$$

This is formed for network "1." The similar equation is derived for network "2."

$$\partial E/\partial W2jk=(O1(\mu)k-O2(\mu)k)\,\partial O2(\mu)k/\partial W2jk$$

$$=(O1(\mu)k-O2(\mu)k)g2^{(1)}(Q2(\mu)k)y2j=\Delta 2(\mu)kj.$$

The two conditions must be satisfied at the same time, therefore,

$$\{\Delta 1kj\}\rightarrow 0,\ \{\Delta 2kj\}\rightarrow 0;\ W1jk^{(\xi)}=W1jk^{(\xi-1)}-\varepsilon\,\Delta 1(\mu)kj,\ W2jk^{(\xi)}=W2jk^{(\xi-1)}-\varepsilon\,\Delta 2(\mu)kj.$$

This is a BP-algorithm for a linked neural network.

The tangential E-term must be zero when the junction is completed precisely for infinitesimal changes W1jk, W2jk, V1ij, and V2ij. On first order differentials, we get,

$$E=\Sigma k[\ \Sigma j\{g1(\mu)^{(1)}(Q1(\mu)k)W1jkf1(\mu)^{(1)}(P1(\mu)iV1ij)$$

$$-\Sigma j\{g2(\mu)^{(1)}(Q2(\mu)k)W2jkf2(\mu)^{(1)}(P2(\mu)iV2ij)\}\ ]^2.$$

Then, we can derive followings,

$$\partial E/\partial W1jk=2\Sigma i\{g1(\mu)^{(1)}(Q1(\mu)k)W1jkf1(\mu)^{(1)}(P1(\mu)iV1ij\}$$

$$*\{g1(\mu)^{(1)}(Q1(\mu)k)f1(\mu)^{(1)}(P1(\mu)iV1ij)=\Delta 1(\mu)kj,$$

$$\partial E/\partial W2jk=-2\Sigma i\{\ g2(\mu)^{(1)}(Q2(\mu)k)W2jkf2(\mu)^{(1)}(P2(\mu)iV2ij\}$$

$$*\{g2(\mu)^{(1)}(Q2(\mu)k)f2(\mu)^{(1)}(P2(\mu)iV2ij)=\Delta 2(\mu)kj,$$

$$\partial E/\partial V1ij=2\Sigma k\{g1(\mu)^{(1)}(Q1(\mu)k)W1jkf1(\mu)^{(1)}(P1(\mu)iV1ij))$$

$$*\{g1(\mu)^{(1)}(Q1(\mu)k)W1jkf1(\mu)^{(1)}(P1(\mu)i)=\Delta 1(\mu)ij,$$

$$\partial E/\partial V2ij=-2\Sigma k\{\ g2(\mu)^{(1)}(Q2(\mu)k)W2kjf2(\mu)^{(1)}(P2(\mu)iV2ij\}$$

$$*\{g2(\mu)^{(1)}(Q2(\mu)k)W2jkf2(\mu)^{(1)}(P2(\mu)i)=\Delta 2(\mu)ij,$$

$$\{\Delta 1(\mu)kj\}\rightarrow 0,\ \{\Delta 2(\mu)kj\}\rightarrow 0,\ \{\Delta 1(\mu)ij\}\rightarrow 0,\ \{\Delta 2(\mu)ij\}\rightarrow 0;\ \text{then, we get,}$$

$$W1jk^{(\xi)}=W1jk^{(\xi-1)}-\varepsilon\,\Delta 1(\mu)kj,\quad V1ij^{(\xi)}=V1ij^{(\xi-1)}-\varepsilon\,\Delta 1(\mu)ij,$$

$$W2jk^{(\xi)}=W2jk^{(\xi-1)}-\varepsilon\,\Delta 2(\mu)kj,\quad V2ij^{(\xi)}=V2ij^{(\xi-1)}-\varepsilon\,\Delta 2(\mu)ij.$$

We consider a combination between continuous and tangential E's,

$$E=\Sigma k(O1(\mu)k-O2(\mu)k)^2+\rho\ \Sigma k[\Sigma j\{\boldsymbol{g1}(\mu)^{(1)}(Q1(\mu)k)W1jk\boldsymbol{f1}(\mu)^{(1)}(P1(\mu)iV1ij)$$

$$-\Sigma j\{\boldsymbol{g2}(\mu)^{(1)}(Q2(\mu)k)W2jk\boldsymbol{f2}(\mu)^{(1)}(P2(\mu)iV2ij)\}]^2,$$

where ρ is a coefficient. Since the iteration, derived from $\partial E/\partial W1jk$ and so on, would not converge to true-zero, the coefficient is introduced as emphasizing important terms. The individual iterations are omitted, that are combinations of previous BP-terms.

4. Conclusions

We discussed the fifth order derivatives of multi-layer neural networks; and using them, we derived many kinds of BP-learning algorithms for a collective or linked neural network that was constructed of plural neural networks whose structures have different structures. The algorithms give iterations to search an object quantity (if we used wording in molecular orbital, it was *the expectation.*), where connection weights among neurons are stationary.

Acknowledgment

The research is financially supported by the Sasagawa Scientific Research Grant from the Japan Science Society.

References

[1] T.Aoyama, H.Ichikawa,"Restricted BP-learning method", IPSJ SIG Notes, NA-37-8, pp.59-66 (1991.7).

[2] R. Masuoka, M. Yamada,"Neural networks learning higher order differentials", IEICE Technical Report, NC94-66, pp.49-56, Feb.1995.

[3] H. Zhu, T. Aoyama, U. Nagashima,"Symmetry on the contour map calculated by multi-layer neural networks", IPSJ SIG Notes, 2000-HPC-82, pp.197-202 (2000.8.5).

[4] M.M. Moya, D.R. Hush,"Network constrains and multi-objective optimization for one-class classification", Neural Networks, vol.9, pp.463-474(1996).

Appendix A:

$$\delta Yj=[f^{(1)}(Pj)+f^{(2)}(Pj)Uij/2+f^{(3)}(Pj)Uij^2/6+f^{(4)}(Pj)Uij^3/24+f^{(5)}(Pj)Uij^4/120+O(6)]Uij$$

$$(\delta Yj)^2=(f(\delta Pj))^2=[\{f^{(1)}(Pj)\}^2+f^{(1)}(Pj)f^{(2)}(Pj)Uij+\{f^{(2)}(Pj)\}^2Uij^2/4+f^{(1)}(Pj)f^{(3)}(Pj)Uij^2/3]$$
$$+f^{(1)}(Pj)f^{(4)}(Pj)Uij^3/12+f^{(2)}(Pj)f^{(3)}(Pj)Uij^3/6+f^{(1)}(Pj)f^{(5)}(Pj)Uij^4/60$$
$$+f^{(2)}(Pj)f^{(4)}(Pj)Uij^4/24+\{f^{(3)}(Pj)\}^2Uij^4/36+O(7)]Uij^2$$

$$(\delta Yj)^3=(f(\delta Pj))^3=[\{f^{(1)}(Pj)\}^3+(3/2)\{f^{(1)}(Pj)\}^2f^{(2)}(Pj)Uij+(3/4)f^{(1)}(Pj)\{f^{(2)}(Pj)\}^2Uij^2$$
$$+(1/2)\{f^{(1)}(Pj)\}^2f^{(3)}(Pj)Uij^2+(1/8)\{f^{(1)}(Pj)\}^2f^{(4)}(Pj)Uij^3+(1/2)f^{(1)}(Pj)f^{(2)}(Pj)f^{(3)}(Pj)Uij^3$$
$$+(1/8)\{f^{(2)}(Pj)\}^3Uij^3+(1/40)\{f^{(1)}(Pj)\}^2f^{(5)}(Pj)Uij^4+(1/8)f^{(1)}(Pj)f^{(2)}(Pj)f^{(4)}(Pj)Uij^4$$
$$+(1/12)f^{(1)}(Pj)\{f^{(3)}(Pj)\}^2Uij^4+(1/8)\{f^{(2)}(Pj)\}^2f^{(3)}(Pj)Uij^4+O(8)]Uij^3$$

$$(\delta Yj)^4=(f(\delta Pj))^4=[\{f^{(1)}(Pj)\}^4+2\{f^{(1)}(Pj)\}^3f^{(2)}(Pj)Uij+(2/3)\{f^{(1)}(Pj)\}^3f^{(3)}(Pj)Uij^2$$
$$+(9/4)\{f^{(1)}(Pj)\}^2\{f^{(2)}(Pj)\}^2Uij^2+O(7)]Uij^4$$

$$(\delta Yj)^5=(f(\delta Pj))^5=[\{f^{(1)}(Pj)\}^5+O(6)]\ Uij^5$$

Appendix B:

$$\delta Ok = g(\delta Qk) = \Sigma n\{g^{(n)}(Qk)*\{Wjk*\delta Yj\}^n/n!\} = g^{(1)}(Qk)Wjk\,\delta Yj + g^{(2)}(Qk)(Wjk\,\delta Yj)^2/2$$
$$+ g^{(3)}(Qk)(Wjk\,\delta Yj)^3/6 + g^{(4)}(Qk)(Wjk\,\delta Yj)^4/24 + g^{(5)}(Qk)(Wjk\,\delta Yj)^5/120 + O(6)$$

$$\delta Ok = g^{(1)}(Qk)\,\Sigma j\{Wkj\,f^{(1)}(Pj)\}Vji\,\delta Xi$$
$$+(1/2)\,\Sigma j\,\{\,g^{(1)}(Qk)Wkj\,f^{(2)}(Pj) + g^{(2)}(Qk)Wkj^2(f^{(1)}(Pj))^2\}Vji^2(\delta Xi)^2$$
$$+(1/6)\,\Sigma j\,\{\,g^{(1)}(Qk)Wkj f^{(3)}(Pj) + 3g^{(2)}(Qk)Wkj^2 f^{(1)}(Pj) f^{(2)}(Pj) + g^{(3)}(Qk)Wkj^3(f^{(1)}(Pj))^3\}Vji^3(\delta Xi)^3$$
$$+(1/24)\,\Sigma j\,\{\,g^{(1)}(Qk)Wkj f^{(4)}(Pj) + g^{(2)}(Qk)Wkj^2[3(f^{(2)}(Pj))^2 + 4 f^{(1)}(Pj) f^{(3)}(Pj)]$$
$$+6g^{(3)}(Qk)Wkj^3(f^{(1)}(Pj))^2 f^{(2)}(Pj) + g^{(4)}(Qk)Wkj^4(f^{(1)}(Pj))^4\}Vji^4(\delta Xi)^4$$
$$+(1/120)\,\Sigma j\,\{g^{(1)}(Qk)Wkj f^{(5)}(Pj) + 5g^{(2)}(Qk)Wkj^2 f^{(1)}(Pj) f^{(4)}(Pj) + 5g^{(3)}(Qk)Wkj^3[3 f^{(1)}(Pj)\{f^{(2)}(Pj)\}^2$$
$$+2\{f^{(1)}(Pj)\}^2 f^{(3)}(Pj)] + 10g^{(4)}(Qk)Wkj^4\{f^{(1)}(Pj)\}^3 f^{(2)}(Pj) + g^{(5)}(Qk)Wkj^5\{f^{(1)}(Pj)\}^5\}Vji^5(\delta Xi)^5 + O(6)$$

Appendix C:

$$\partial^2 Ok/\partial Xi^2 = (1/2)\{g^{(1)}Wf^{(2)} + g^{(2)}W^2(f^{(1)})^2\}V^2 = (1/2)\,\Sigma j\{g^{(1)}(Qk)Wkj f^{(2)}(Pi) + g^{(2)}(Qk)Wkj^2(f^{(1)}(Pi))^2\}Vij^2$$
$$\partial^3 Ok/\partial Xi^3 = (1/6)\{g^{(1)}Wf^{(3)} + 3g^{(2)}W^2 f^{(1)} f^{(2)} + g^{(3)}W^3(f^{(1)})^3\}V^3$$
$$\partial^4 Ok/\partial Xi^4 = (1/24)\{g^{(1)}Wf^{(4)} + g^{(2)}W^2[3(f^{(2)})^2 + 4 f^{(1)} f^{(3)}] + 6g^{(3)}W^3(f^{(1)})^2 f^{(2)} + g^{(4)}W^4(f^{(1)})^4\}V^4$$
$$\partial^5 Ok/\partial Xi^5 = (1/120)\{g^{(1)}Wf^{(5)} + 5g^{(2)}W^2 f^{(1)} f^{(4)} + 5g^{(3)}W^3[3 f^{(1)}(f^{(2)})^2 + 2(f^{(1)})^2 f^{(3)}]$$
$$+10g^{(4)}W^4(f^{(1)})^3 f^{(2)} + g^{(5)}W^5(f^{(1)})^5\}V^5$$