研究ノート

# バックプロパゲーション・ニューラルネットワークにおける学習率について

田口　功

## 1. はじめに

バックプロパゲーション．ニューラルネットワークは、ニューロコンピュータの発達の歴史を考えても最も重要なものである。すべての学習させたい入力に対して有効な出力を取り出すための誤差関数を定義し、その誤差を最小にするための重み係数を決定する事が一つの目標である。さらに、学習回数を減少させることも研究目標の一つになっている。

特に、学習率は、現在の段階では、一般的に一つの定数として与え、すべての重み係数の更新を行っている。ヘッセ行列の一般逆行列を求めることによって各重み係数に対して、個々の学習係数を決定する研究も進んでいるがむずかしいとされている。

本稿では、3層ニューラルネットワークを考え、（入力層は、0層とする）重み係数の更新（物理的意味の説明）を行うための従来の方法を用いたシュミレーションによる実際の報告と問題点を示した。

## 2. 3層ニューラルネットワークにおける重み係数変化による誤差の変化について

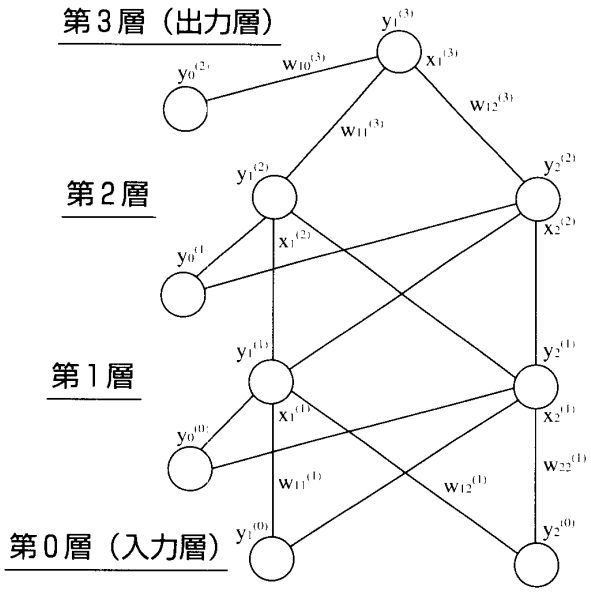


図2.1　3層ニューラルネットワーク

図2.1に示すように、入力層（0層）の入力を $y_1^{(0)}$、$y_2^{(0)}$ とし1層の出力を $y_1^{(1)}$、$y_2^{(1)}$ 2層の出力を $y_1^{(2)}$、$y_2^{(2)}$、3層の出力を $y_1^{(3)}$ とする。また、入力を持たない $y_0^{(0)}$ $y_0^{(1)}$、$y_0^{(2)}$ は、重み係数と共に、しきい値を表現するための出力である。重み係数は、0層から、1層の間を考えた場合、6個存在する。例えば、重み係数 $w_{21}^{(1)}$

とは、0層から1層の信号伝達を行い、0層の1番目の素子から、1層の2番目の素子の信号伝達の割合を決定する係数である。すなわち、$w_{ij}$と一般的に表現すれば、m-1層のj番目の素子から、m層のi番目に向かう信号伝達を行う重み係数を意味する。また、ここでは隠れ層は、1層および2層の場合を考える。

次に、0層に、入力信号が入力された場合の出力計算を行う。最初に、1層の1番目の素子への入力を計算すると、

$$x_1^{(1)} = y_1^{(0)} w_{11}^{(1)} + y_2^{(0)} w_{12}^{(1)} - y_0^{(0)} w_{10}^{(1)} \quad (2\text{-}1)$$

となる。同様に、

$$x_2^{(1)} = y_1^{(0)} w_{21}^{(1)} + y_2^{(0)} w_{22}^{(1)} - y_0^{(0)} w_{20}^{(1)} \quad (2\text{-}2)$$

となる。$x_1^{(1)}$および$x_2^{(1)}$に対して、出力は、一般的にシグモイド関数を用いて1層目の1番目の出力および2番目の出力は計算され、次に示す式で表される。

$$y_1^{(1)} = F(x_1^{(1)}) \quad (2\text{-}3)$$

$$y_2^{(1)} = F(x_2^{(1)}) \quad (2\text{-}4)$$

$$y_0^{(1)} = 1 \quad (2\text{-}5)$$

ここで、(2-3) 式における$y_1^{(1)}$は、隠れ層1層目の1番目出力、(2-4) 式の$y_2^{(1)}$は、隠れ層1層目の2番目の出力を示す。隠れ層2層目の入力$x_1^{(2)}$、$x_2^{(2)}$は、同様に

$$x_1^{(2)} = y_1^{(1)} w_{11}^{(2)} + y_2^{(1)} w_{12}^{(2)} - y_0^{(1)} w_{10}^{(2)} \quad (2\text{-}6)$$

$$x_2^{(2)} = y_1^{(1)} w_{21}^{(2)} + y_2^{(1)} w_{22}^{(2)} - y_0^{(1)} w_{20}^{(2)} \quad (2\text{-}7)$$

$$y_1^{(2)} = F(x_1^{(2)}) \quad (2\text{-}8)$$

$$y_2^{(2)} = F(x_2^{(2)}) \quad (2\text{-}9)$$

$$y_0^{(2)} = 1 \quad (2\text{-}10)$$

となる。また3層目の入力$x_1^{(3)}$は、

$$x_1^{(3)} = y_1^{(2)} w_{11}^{(3)} + y_2^{(2)} w_{12}^{(3)} - y_0^{(2)} w_{10}^{(3)} \quad (2\text{-}11)$$

$$y_1^{(3)} = F(x_1^{(3)}) \quad (2\text{-}12)$$

となる。ここで入力に対して期待出力を$d_1$と置くと、その誤差は$y_1^{(3)} - d_1$となる。従って、2乗誤差関数をEとすると、

$$E = 1/2 (y_1^{(3)} - d_1)^2 \quad (2\text{-}13)$$

となる。従って、出力層に対する重み係数の偏微分関数$\partial E/\partial w_{10}^{(3)}$、$\partial E/\partial w_{11}^{(3)}$、$\partial E/\partial w_{12}^{(3)}$は、偏微分に関する連鎖率とシグモイド関数の微分に関する性質を利用すれば、

$$\begin{aligned}\partial E/\partial w_{10}^{(3)} &= \partial E/\partial x_1^{(3)} \cdot \partial x_1^{(3)}/\partial w_{10}^{(3)} \\ &= \partial E/\partial y_1^{(3)}\ \partial y_1^{(3)}/\partial x_1^{(3)} \cdot \partial x_1^{(3)}/\partial w_{10}^{(3)} \\ &= \partial/\partial y_1^{(3)}\ \{(1/2 (y_1^{(3)} - d_1)^2)\}\ \{y_1^{(3)}(1 - y_1^{(3)})\}\ \{-y_0^{(2)}\} \\ &= (y_1^{(3)} - d_1)\ y_1^{(3)}(1 - y_1^{(3)})(-y_0^{(2)}) \quad (2\text{-}14)\end{aligned}$$

$$\begin{aligned}\partial E/\partial w_{11}^{(3)} &= \partial E/\partial x_1^{(3)} \cdot \partial x_1^{(3)}/\partial w_{11}^{(3)} \\ &= \partial E/\partial y_1^{(3)}\ \partial y_1^{(3)}/\partial x_1^{(3)} \cdot \partial x_1^{(3)}/\partial w_{11}^{(3)} \\ &= (y_1^{(3)} - d_1)\ y_1^{(3)}(1 - y_1^{(3)})(y_1^{(2)}) \quad (2\text{-}15)\end{aligned}$$

$$\begin{aligned}\partial E/\partial w_{12}^{(3)} &= \partial E/\partial x_1^{(3)} \cdot \partial x_1^{(3)}/\partial w_{12}^{(3)} \\ &= \partial E/\partial y_1^{(3)}\ \partial y_1^{(3)}/\partial x_1^{(3)} \cdot \partial x_1^{(3)}/\partial w_{12}^{(3)} \\ &= (y_1^{(3)} - d_1)\ y_1^{(3)}(1 - y_1^{(3)})(y_2^{(2)}) \quad (2\text{-}16)\end{aligned}$$

となる。ここで (2-14) 式、(2-15) 式および (2-16) 式をみると、第3層の出力層への重み係数変化による誤差関数偏微分式は、教師信号と実際の

出力差およびシグモイド関数の微分値および第2層からの出力（重み係数の位置によって異なる）の三つの要素の積によって決定されることがわかる。さらに、第1層から第2層に結合している重み係数変化による誤差変化（$\partial E/\partial w_{21}^{(2)}$）を例として計算すると、

$$\begin{aligned}\partial E/\partial w_{21}^{(2)} &= \partial E/\partial x_2^{(2)} \cdot \partial x_2^{(2)}/\partial w_{21}^{(2)} \\ &= \partial E/\partial y_2^{(2)}\ \partial y_2^{(2)}/\partial x_2^{(2)} \cdot \\ &\quad \partial x_2^{(2)}/\partial w_{21}^{(2)} \\ &= \partial E/\partial x_1^{(3)}\ \partial x_1^{(3)}/\partial y_2^{(2)} \\ &\quad \partial y_2^{(2)}/\partial x_2^{(2)} \cdot \partial x_2^{(2)}/\partial \\ &\quad w_{21}^{(2)} \\ &= \partial E/\partial x_1^{(3)}\ w_{12}^{(3)}\ (y_2^{(2)}) \\ &\quad (1-y_2^{(2)})\ y_1^{(1)} \\ &= \partial E/\partial y_1^{(3)}\ \partial y_1^{(3)}/\partial x_1^{(3)} \\ &\quad w_{12}^{(3)}\ y_2^{(2)}(1-y_2^{(2)})\ y_1^{(1)} \\ &= (y_1^{(3)}-d_1)\ y_1^{(3)}(1-y_1^{(3)}) \\ &\quad w_{12}^{(3)}\ y_2^{(2)}(1-y_2^{(2)})\ y_1^{(1)}\end{aligned} \tag{2-17}$$

となる。(2-17) 式をよくみると、第2層から第3層の重み係数変化による誤差関数の偏微分式は、$w_{12}^{(3)}$ の重み係数を通して、出力誤差（$y_1^{(3)}-d_1$）および第3層の出力微分（$y_1^{(3)}(1-y_1^{(3)})$）および $w_{21}^{(2)}$ と結合している出力微分と $w_{21}^{(2)}$ と結合している1層前の出力の積となることがわかる。全く同様な考えで他の第2層における偏微分式は導くことが出来るので、その結果を以下に示す。

$$\begin{aligned}\partial E/\partial w_{11}^{(2)} &= (y_1^{(3)}-d_1)\ y_1^{(3)}(1-y_1^{(3)}) \\ &\quad w_{11}^{(3)}\ y_1^{(2)}\ (1-y_1^{(2)}) \\ &\quad y_1^{(1)}\end{aligned} \tag{2-18}$$

$$\begin{aligned}\partial E/\partial w_{12}^{(2)} &= (y_1^{(3)}-d_1)\ y_1^{(3)}(1-y_1^{(3)}) \\ &\quad w_{11}^{(3)}\ y_1^{(2)}\ (1-y_1^{(2)}) \\ &\quad y_2^{(1)}\end{aligned} \tag{2-19}$$

$$\begin{aligned}\partial E/\partial w_{10}^{(2)} &= (y_1^{(3)}-d_1)\ y_1^{(3)}(1-y_1^{(3)}) \\ &\quad w_{11}^{(3)}\ y_1^{(2)}\ (1-y_1^{(2)}) \\ &\quad y_0^{(1)}\end{aligned} \tag{2-20}$$

$$\begin{aligned}\partial E/\partial w_{22}^{(2)} &= (y_1^{(3)}-d_1)\ y_1^{(3)}(1-y_1^{(3)}) \\ &\quad w_{12}^{(3)}\ y_2^{(2)}\ (1-y_2^{(2)}) \\ &\quad y_2^{(1)}\end{aligned} \tag{2-21}$$

$$\begin{aligned}\partial E/\partial w_{20}^{(2)} &= (y_1^{(3)}-d_1)\ y_1^{(3)}(1-y_1^{(3)}) \\ &\quad w_{12}^{(3)}\ y_2^{(2)}\ (1-y_2^{(2)}) \\ &\quad y_0^{(1)}\end{aligned} \tag{2-22}$$

次に第0層（入力層）と第1層の間の重み係数変化に対する誤差評価関数偏微分式を考える。同様に、

$$\begin{aligned}\partial E/\partial w_{12}^{(1)} &= \partial E/\partial x_1^{(1)} \cdot \partial x_1^{(1)}/\partial w_{12}^{(1)} \\ &= \partial E/\partial x_1^{(1)} \cdot y_2^{(0)} \\ &= \partial E/\partial y_1^{(1)} \cdot \partial y_1^{(1)}/\partial x_1^{(1)} \\ &\quad y_2^{(0)} \\ &= \partial E/\partial y_1^{(1)}\ y_1^{(1)}(1-y_1^{(1)})\ y_2^{(0)} \\ &= \partial E/\partial y_1^{(1)}\ y_1^{(1)}(1-y_1^{(1)})\ y_2^{(0)} \\ &= (\partial E/\partial x_1^{(2)}\ \partial x_1^{(2)}/\partial y_1^{(1)} + \\ &\quad \partial E/\partial x_2^{(2)}\ \partial x_2^{(2)}/\partial y_1^{(1)}) \\ &\quad y_1^{(1)}\ (1-y_1^{(1)})\ y_2^{(0)}\end{aligned} \tag{2-14}$$

となる。ここで、(2-17) 式を導く過程で使用した式を用いて、

$$\begin{aligned}\partial E/\partial x_1^{(2)} &= (y_1^{(3)}-d_1)\ y_1^{(3)}(1-y_1^{(3)}) \\ &\quad w_{11}^{(3)}\ y_1^{(2)}(1-y_1^{(2)}) \\ &= A\end{aligned} \tag{2-24}$$

$$\begin{aligned}\partial E/\partial x_2^{(2)} &= (y_1^{(3)}-d_1)\ y_1^{(3)}(1-y_1^{(3)}) \\ &\quad w_{12}^{(3)}\ y_2^{(2)}(1-y_2^{(2)}) \\ &= B\end{aligned} \tag{2-25}$$

と置くと、

$$\partial E/\partial w_{12}^{(1)} = (A\ w_{11}^{(2)} + B\ w_{21}^{(2)})\ y_1^{(1)}$$

$$(1 - y_1^{(1)}) \quad y_2^{(0)} \qquad (2\text{-}26)$$

となる。ここで（2-26）式の（A　$w_{11}^{(2)}$ ＋B　$w_{21}^{(2)}$）は、重み係数を2回通しての出力差、$y_1^{(1)}$ $(1 - y_1^{(1)})$ は、重み係数の先の出力微分、$y_2^{(0)}$ は、重み係数の手前からの入力と考える事が出来る。他の重み係数偏微分も同様に考えると、

$$\partial E/\partial w_{11}^{(1)} = (A \quad w_{11}^{(2)} + B \quad w_{21}^{(2)}) \quad y_1^{(1)} \quad (1 - y_1^{(1)}) \quad y_1^{(0)} \qquad (2\text{-}27)$$

$$\partial E/\partial w_{21}^{(1)} = (A \quad w_{12}^{(2)} + B \quad w_{22}^{(2)}) \quad y_2^{(1)} \quad (1 - y_2^{(1)}) \quad y_1^{(0)} \qquad (2\text{-}28)$$

$$\partial E/\partial w_{22}^{(1)} = (A \quad w_{12}^{(2)} + B \quad w_{22}^{(2)}) \quad y_2^{(1)} \quad (1 - y_2^{(1)}) \quad y_2^{(0)} \qquad (2\text{-}29)$$

となる。以上をまとめると次の事が言える。

1. 誤差関数の偏微分式は、出力差の形式は異なるが、重み係数の先の出力差の項を含む。
2. 誤差関数の偏微分式は、重み係数の先の出力微分の項を含む。
3. 誤差関数の偏微分式は、重み係数の手前からの入力を含む項が存在する。

## 3. 重み係数の更新について

排他的論理和を学習する場合は非常によく知られている。ニューラルネットワークに入力データを4組与え、期待出力を4組与える問題である（表3.1）。

| A | B | C |
|---|---|---|
| 1．0 | 1．0 | 0．0 |
| 0．0 | 1．0 | 1．0 |
| 1．0 | 0．0 | 1．0 |
| 0．0 | 0．0 | 0．0 |

表3.1　排他的論理問題

表1において、AとBの排他的論理和をとると、その結果はCになる。すなわち、入力のどちらか一方が1の時出力は1となる。ニューラルネットワークに入力ABを与えた場合には、0に近い値または、1に近い値が出力されれば学習が成功したことになるわけである（実際の実行結果は後で示す）。

次に、重み係数の更新を考える前に、実際の動作について述べる。

1. 重み係数の初期値を乱数を使用して与える（例えば、－0.1から0.1）。
2. 入力パターンをネットに与える。
3. 1.および2.によって与えられたデータによって出力計算を行う。
4. 誤差評価を行うために2乗誤差関数Eの重み係数 $w_{ij}^{(m)}$ に関する偏微分の計算を行う。
5. 重み係数の更新を次式に従って行う。

$$w_{ij}^{(m)}[n+1] = w_{ij}^{(m)}[n] + \eta \left(\partial E/\partial w_{ij}^{(m)}\right) \qquad (3\text{-}1)$$

ここで $\eta$ は、学習率を表す。

6. Eが十分小さくなるまで2.から5.までを繰り返す。
7. 入力パターンが1個の場合には、これで終了となる。

また、学習データが複数存在する場合には、期待出力データも同数存在し、複数となる。更新方法には2通りの考え方が存在する。

（方法1）一組の学習データに対する2乗誤差式を考え、順次学習データをネットに提示し、上で述べた2.から6.に従って重み係数を更新する。各学習データに対するEがすべて十分小さくなるまで繰り返す（逐次更新）。

（方法2）学習データが4組存在し、期待出力も4組存在する場合を考える。期待出力とネットの実際の出力の2乗誤差の総和は、

$$E = (1/2)(E_1 + E_2 + E_3 + E_4) \qquad (3\text{-}2)$$

とし、Eの各層の重み係数に対する偏微分 $\partial E/\partial w_{ij}^{(m)}$ は、各学習データに対する和として、

表4.1

```
*** ﾈｯﾄﾜｰｸﾊﾟﾗﾒｰﾀｰﾉｾｯﾃｲ ***
ｶｸﾚﾝｿｳﾉｶｽﾞ (max:5) : 1
ﾆｭｳﾘｮｸｿｳﾉﾕﾆｯﾄﾉｶｽﾞ : 2
ｶｸﾚﾝｿｳ 1 ｿｳﾒﾉﾕﾆｯﾄﾉｶｽﾞ : 2
ｼｭﾂﾘｮｸｿｳﾉﾕﾆｯﾄﾉｶｽﾞ:1
ｵﾓﾐｺｳｼﾝｹｲｽｳn : 0.3
ｵﾓﾐｺｳｼﾝｹｲｽｳa : 0

ｶﾞｸｼｭｳﾃﾞｰﾀ . ﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ: b:exor.in
ｷｮｳｼﾃﾞｰﾀ.ﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ : b:exor.out
ｶﾞｸｼｭｳｶｲｽｳｼﾞｮｳｹﾞﾝﾉｾｯﾃｲﾖｵｺﾅｯﾃｸﾀﾞｻｲ : -1
ｷｿﾞﾝﾉｵﾓﾐﾃﾞｰﾀﾖﾂｶｲﾏｽｶ? (y/n) : n

ｵﾓﾐﾉｼｭｳｾｲﾎｳﾎｳ
1.ｲｯｶﾂｼｭｳｾｲﾎｳ
2.ﾂｲｼﾞｼｭｳｾｲﾎｳ
ﾄﾞﾁﾗﾆｼﾏｽｶ? : 1

ｴﾗｰﾖﾌｧｲﾙﾆｼｭﾂﾘｮｸｼﾏｽｶ? (y/n) : y
ｼｭﾂﾘｮｸｴﾗｰﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ : b:exor.err

ｶﾞｸｼｭｳｶｲｽｳ = 1389
ｺﾞｻ(1) = 0.009985

ｶﾞｸｼｭｳｹｯｶﾖｼｭﾂﾘｮｸｼﾏｽｶ? (y/n) : y

ﾆｭｳﾘｮｸ :0.0000.000
ｼｭﾂﾘｮｸ : 0.128
ｺﾞｻ :0.016

ﾆｭｳﾘｮｸ :0.0001.000
ｼｭﾂﾘｮｸ : 0.881
ｺﾞｻ :0.014

ﾆｭｳﾘｮｸ :1.0000.000
ｼｭﾂﾘｮｸ : 0.943
ｺﾞｻ :0.003

ﾆｭｳﾘｮｸ :1.0001.000
ｼｭﾂﾘｮｸ : 0.078
ｺﾞｻ :0.006
```

表4.2

```
*** ﾈｯﾄﾜｰｸﾊﾟﾗﾒｰﾀｰﾉｾｯﾃｲ ***
ｶｸﾚﾝｿｳﾉｶｽﾞ (max:5) : 1
ﾆｭｳﾘｮｸｿｳﾉﾕﾆｯﾄﾉｶｽﾞ : 2
ｶｸﾚﾝｿｳ 1 ｿｳﾒﾉﾕﾆｯﾄﾉｶｽﾞ : 2
ｼｭﾂﾘｮｸｿｳﾉﾕﾆｯﾄﾉｶｽﾞ:1
ｵﾓﾐｺｳｼﾝｹｲｽｳn : 0.5
ｵﾓﾐｺｳｼﾝｹｲｽｳa : 0

ｶﾞｸｼｭｳﾃﾞｰﾀ . ﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ: B:EXOR.IN
ｷｮｳｼﾃﾞｰﾀ.ﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ : EXOR.OUT
EXOR.OUT ｶﾞｵｰﾌﾟﾝﾃﾞｷﾏｾﾝﾃﾞｼﾀ。
ﾓｳｲﾁﾄﾞﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ : -1
-1 ｶﾞｵｰﾌﾟﾝﾃﾞｷﾏｾﾝﾃﾞｼﾀ。
ﾓｳｲﾁﾄﾞﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ : B:exor.out
ｶﾞｸｼｭｳｶｲｽｳｼﾞｮｳｹﾞﾝﾉｾｯﾃｲﾖｵｺﾅｯﾃｸﾀﾞｻｲ : -1
ｷｿﾞﾝﾉｵﾓﾐﾃﾞｰﾀﾖﾂｶｲﾏｽｶ? (y/n) : n

ｵﾓﾐﾉｼｭｳｾｲﾎｳﾎｳ
1.ｲｯｶﾂｼｭｳｾｲﾎｳ
2.ﾂｲｼﾞｼｭｳｾｲﾎｳ
ﾄﾞﾁﾗﾆｼﾏｽｶ? : 1

ｴﾗｰﾖﾌｧｲﾙﾆｼｭﾂﾘｮｸｼﾏｽｶ? (y/n) : y
ｼｭﾂﾘｮｸｴﾗｰﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ : b:exor.err

ｶﾞｸｼｭｳｶｲｽｳ = 957
ｺﾞｻ(1) = 0.009966

ｶﾞｸｼｭｳｹｯｶﾖｼｭﾂﾘｮｸｼﾏｽｶ? (y/n) : y

ﾆｭｳﾘｮｸ :0.0000.000
ｼｭﾂﾘｮｸ : 0.107
ｺﾞｻ :0.011

ﾆｭｳﾘｮｸ :0.0001.000
ｼｭﾂﾘｮｸ : 0.915
ｺﾞｻ :0.007

ﾆｭｳﾘｮｸ :1.0000.000
ｼｭﾂﾘｮｸ : 0.914
ｺﾞｻ :0.007

ﾆｭｳﾘｮｸ :1.0001.000
ｼｭﾂﾘｮｸ : 0.117
ｺﾞｻ :0.014
```

表4.3

```
*** ﾈｯﾄﾜｰｸﾊﾟﾗﾒｰﾀｰﾉｾｯﾃｲ ***
ｶｸﾚﾝｿｳﾉｶｽﾞ (max:5) : 1
ﾆｭｳﾘｮｸｿｳﾉﾕﾆｯﾄﾉｶｽﾞ : 2
ｶｸﾚﾝｿｳ 1 ｿｳﾒﾉﾕﾆｯﾄﾉｶｽﾞ : 2
ｼｭﾂﾘｮｸｿｳﾉﾕﾆｯﾄﾉｶｽﾞ:1
ｵﾓﾐｺｳｼﾝｹｲｽｳn : 0.6
ｵﾓﾐｺｳｼﾝｹｲｽｳa : 0

ｶﾞｸｼｭｳﾃﾞｰﾀ . ﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ: b:exor.in
ｷｮｳｼﾃﾞｰﾀ.ﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ : b:exor.out
ｶﾞｸｼｭｳｶｲｽｳｼﾞｮｳｹﾞﾝﾉｾｯﾃｲﾖｵｺﾅｯﾃｸﾀﾞｻｲ : 1000
ｷｿﾞﾝﾉｵﾓﾐﾃﾞｰﾀﾖﾂｶｲﾏｽｶ? (y/n) : n

ｵﾓﾐﾉｼｭｳｾｲﾎｳﾎｳ
1.ｲｯｶﾂｼｭｳｾｲﾎｳ
2.ﾂｲｼﾞｼｭｳｾｲﾎｳ
ﾄﾞﾁﾗﾆｼﾏｽｶ? : 1

ｴﾗｰﾖﾌｧｲﾙﾆｼｭﾂﾘｮｸｼﾏｽｶ? (y/n) : y
ｼｭﾂﾘｮｸｴﾗｰﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ : b:exor.err

ｶﾞｸｼｭｳｶｲｽｳ = 1000
ｺﾞｻ(1) = 0.176685

ｶﾞｸｼｭｳｹｯｶﾖｼｭﾂﾘｮｸｼﾏｽｶ? (y/n) : y

ﾆｭｳﾘｮｸ :0.0000.000
ｼｭﾂﾘｮｸ : 0.360
ｺﾞｻ :0.130

ﾆｭｳﾘｮｸ :0.0001.000
ｼｭﾂﾘｮｸ : 0.325
ｺﾞｻ :0.456

ﾆｭｳﾘｮｸ :1.0000.000
ｼｭﾂﾘｮｸ : 0.908
ｺﾞｻ :0.008

ﾆｭｳﾘｮｸ :1.0001.000
ｼｭﾂﾘｮｸ : 0.336
ｺﾞｻ :0.113
```

表4.4

```
*** ﾈｯﾄﾜｰｸﾊﾟﾗﾒｰﾀｰﾉｾｯﾃｲ ***
ｶｸﾚﾝｿｳﾉｶｽﾞ (max:5) : 1
ﾆｭｳﾘｮｸｿｳﾉﾕﾆｯﾄﾉｶｽﾞ : 2
ｶｸﾚﾝｿｳ 1 ｿｳﾒﾉﾕﾆｯﾄﾉｶｽﾞ : 2
ｼｭﾂﾘｮｸｿｳﾉﾕﾆｯﾄﾉｶｽﾞ:1
ｵﾓﾐｺｳｼﾝｹｲｽｳn : 0.6
ｵﾓﾐｺｳｼﾝｹｲｽｳa : 0

ｶﾞｸｼｭｳﾃﾞｰﾀ . ﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ: b:exor.in
ｷｮｳｼﾃﾞｰﾀ.ﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ : b:exor.out
ｶﾞｸｼｭｳｶｲｽｳｼﾞｮｳｹﾞﾝﾉｾｯﾃｲﾖｵｺﾅｯﾃｸﾀﾞｻｲ : 1500
ｷｿﾞﾝﾉｵﾓﾐﾃﾞｰﾀﾖﾂｶｲﾏｽｶ? (y/n) : n

ｵﾓﾐﾉｼｭｳｾｲﾎｳﾎｳ
1.ｲｯｶﾂｼｭｳｾｲﾎｳ
2.ﾂｲｼﾞｼｭｳｾｲﾎｳ
ﾄﾞﾁﾗﾆｼﾏｽｶ? : 1

ｴﾗｰﾖﾌｧｲﾙﾆｼｭﾂﾘｮｸｼﾏｽｶ? (y/n) : y
ｼｭﾂﾘｮｸｴﾗｰﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ : b:exor.err

ｶﾞｸｼｭｳｶｲｽｳ = 1500
ｺﾞｻ(1) = 0.130751

ｶﾞｸｼｭｳｹｯｶﾖｼｭﾂﾘｮｸｼﾏｽｶ? (y/n) : y

ﾆｭｳﾘｮｸ :0.0000.000
ｼｭﾂﾘｮｸ : 0.508
ｺﾞｻ :0.258

ﾆｭｳﾘｮｸ :0.0001.000
ｼｭﾂﾘｮｸ : 0.490
ｺﾞｻ :0.260

ﾆｭｳﾘｮｸ :1.0000.000
ｼｭﾂﾘｮｸ : 0.955
ｺﾞｻ :0.002

ﾆｭｳﾘｮｸ :1.0001.000
ｼｭﾂﾘｮｸ : 0.052
ｺﾞｻ :0.003
```

$\partial E/\partial w_{ij}^{(m)} = \partial E_1/\partial w_{ij}^{(m)} + \partial E_2/\partial w_{ij}^{(m)} + \partial E_3/\partial w_{ij}^{(m)} + \partial E_4/\partial w_{ij}^{(m)}$　(3-3)

とする。(3-3) 式の右辺の各項は、方法1で述べたようにして計算可能であるから、各学習データにわたって総和をとってから、(3-1) 式を用いて重み係数の更新を行う（一括更新)。

ここで問題となるのが、どちらの更新方法を用いても (3-1) 式における学習率 $\eta$ は、定数であり、$\eta$ を大きくすると、収束速度は、勿論速くなるが、大き過ぎると発散してしまうということである。従来、この $\eta$ に関しては、経験的に値を決定していた。系統的な選択法は、確率されていないのが現状である。

## 4. 排他的論理和を例とした、バックプロパゲーション学習の実行

計算機シュミレーションを行うために、文献 (4) のC言語で書かれたプログラムを参考とし、実際に作成した。ただし、隠れ層の数は、後に、結果の解析を容易とするために1層とした。更新法としては、一括更新法を主に使用した。学習率（重み更新係数）が、0.3、0.5、0.6（学習回数1000回)、0.6（学習回数1500回）の場合を表4.1、表4.2、表4.3、表4.4に示す。プログラムは表4.5に示す。

表4.5

```
/* ********************************************************** */
/*                                                            */
/*                                                            */
/*                        .bp.c.                              */
/*                   programmed by I.TAGUTI                   */
/* ********************************************************** */
#include <string.h>
#include <math.h>
#include <stdio.h>
#include <process.h>

#define YES 1
#define NO  0
#define EXIT 0
#define AGAIN 1

int junbi(int mode);
void omomi( int sou );
void neu_cal( int sou, int larn );
void execute( int sou, int mm );
void data_set( int i, int sou );
void bprun( int sou );
void out_cal( int sou, int mm );
void add_ew( int sou, int patern );
void w_renew( int sou );
float ran( void );
void wf_write( int sou );
int chknum( int nu , int np );
int hityn( void );
int mode_sel( void );
FILE  *openfile( char *mode, int sw );

/* ********************************************************** */
/*                                                            */
/*                                                            */
/*                  k(k>50)                                   */
/* mode[m+2].d[k].x[m+2][k].                                  */
```

```
/* w[m-1][n][n].dw[m-1][n][n].ew[m-1][n][n].                    */
/* tohenkousuru.                                                 */
/*                                                                */
/* ************************************************************** */
int icount, node[3];
float eta, alphar, d[4], ind[8], oud[4], err[2];
float x[3][4], w[4][2][2], dw[3][2][2], ew[4][2][2];

/* ************************************************************ */
/*  ﾒｲﾝﾙｰﾁﾝ                                                      */
/* ************************************************************ */
void main(void)
{
       FILE *fi, *fo;
       int larn, sou, i, j, mode;
       float dat;

       while( 1 ) {
mode=mode_sel();                          /*gakushu.jikou.shuryounosentaku */
if ( mode==3 ) break;
printf("¥x1b*" );                      /* ﾃｷｽﾄｶﾞﾒﾝﾉｸﾘｱ */
sou = junbi( mode );                  /* ﾆｭｰﾗﾙﾈｯﾄﾉｼｮｷｾｯﾃｲ */
if ( mode == 1 )
     printf( "¥nｶｸｼｭｳﾃﾞｰﾀ . ﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ: " );
     if ( mode == 2 )
 printf( "¥nﾃﾞｰﾀ.ﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ : " ) ;
    /* ﾊﾟﾀｰﾝﾖﾐｺﾑ */
fi = openfile( "rt", AGAIN );
fscanf( fi, "%d", &icount );              /* ﾊﾟﾀｰﾝｽｳﾉﾖﾐｺﾐ */
for ( i = 0; i < icount; i++ )  {
     for ( j = 0; j < node[0]; j++) {
  fscanf( fi, "%f", &dat );
  ind[node[0]*i+j] = dat;
     }
 }
 fclose( fi );
 if ( mode != 2 ) {
     printf( "ｷｮｳｼﾃﾞｰﾀ.ﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ : ") ;
      /* ｷｮｳｼﾃﾞｰﾀﾉﾖﾐｺﾐ */
     fo = openfile( "rt", AGAIN );
     for ( i = 0; i < icount; i++ ) {
 for ( j = 0; j < node[sou-1]; j++ ) {
 fscanf( fo, "%f", &dat );
 oud[node[sou-1]*i+j] = dat;
      }
}
fclose( fo ) ;
printf( "ｶﾞｸｼｭｳｶｲｽｳｼﾞｮｳｹﾞﾝﾉｾｯﾃｲﾜｵｺﾅｯﾃｸﾀﾞｻｲ : " );
scanf( "%d", &larn );
}
omomi( sou );
if ( mode == 1 ) neu_cal( sou, larn );          /* ｶﾞｸｼｭｳ   */
if ( mode == 2 ) execute( sou,2 );              /* ｼﾞｯｺｳ    */
    }
}
/* ************************************************** */
/*  ﾆｭｰﾗﾙﾈｯﾄﾉｼｮｷｾｯﾃｲ                                  */
/* ************************************************** */
int junbi( int mode )
{
       int n, i, j, sou;

       printf( "¥n*** ﾈｯﾄﾜｰｸﾊﾟﾗﾒｰﾀｰﾉｾｯﾃｲ ***¥n" );
       printf( "ｶｸﾚﾝｿｳﾉｶｽﾞ (max:5) : " );      /* ｶｸﾚﾝｿｳﾉｶｽﾞﾉﾆｭｳﾘｮｸ */
       scanf( "%d", &sou );
       sou = sou + 2;
       printf( "ﾆｭｳﾘｮｸｿｳﾉﾕﾆｯﾄﾉｶｽﾞ : " );        /* ｶｸｿｳﾉPEｽｳﾉﾆｭｳﾘｮｸ */
```

```
      scanf( "%d", &node[0] );
      if ( sou != 2 ) {
  if ( sou > 7 ) sou = 7;
  for ( i = 1; i < sou -1; i++ ) {
       printf( "ｶｸﾚﾝｿｳ %d ｿｳﾒﾉﾕﾆｯﾄﾉｶｽﾞ : ", i );
       scanf( "%d", &node[i] );
   }
}
printf( "ｼｭﾂﾘｮｸｿｳﾉﾕﾆｯﾄﾉｶｽﾞ:" );
scanf( "%d", &node[sou-1] );
if ( mode == 1) {
printf( "ｵﾓﾐｺｳｼﾝｹｲｽｳn  : " );        /* ｵﾓﾐｺｳｼﾝｹｲｽｳnﾉﾆｭｳﾘｮｸ  */
scanf( "%f", &eta );

printf( "ｵﾓﾐｺｳｼﾝｹｲｽｳa : " );        /* ｵﾓﾐｺｳｼﾝｹｲｽｳaﾉﾆｭｳﾘｮｸ  */
scanf( "%f", &alphar );
}
for ( n = 0; n < sou -1; n++ ) {
    for ( i = 1; i <= node[n+1]; i++ ) {
for ( j = 0; j <= node[n]; j++ ) {
    dw[n][i][j] = (float)0.0;
    }
 }
      }
      return( sou );
}
/* ********************************************************* */
/*  ｵﾓﾐﾉｼｮｷｶｵﾖﾋﾞﾖﾐｺﾐ                                        */
/* ********************************************************* */
void omomi( int sou )
{
    FILE *wfi;
    char dumy[20];
    int i, j, n;

    printf( "ｷｿﾞﾝﾉｵﾓﾐﾃﾞｰﾀﾖﾂｶｲﾏｽｶ? (y/n) : " );
    if (hityn() == YES ) {
       printf( "ｵﾓﾐﾌｧｲﾙﾒｲ : " );       /* ｵﾓﾐﾌｧｲﾙﾒｲﾉﾆｭｳﾘｮｸ */
       wfi = openfile( "rt", AGAIN );
       for ( n= 0 ; n < sou-1 ; n++ ) {
   fscanf( wfi, "%s", dumy );
   for ( i = 1 ; i <= node[n+1] ; i++ ){
       for ( j = 0 ; j <= node[n] ; j++ ) {
   fscanf( wfi, "%f", &w[n][i][j] );/* ｵﾓﾐﾃﾞｰﾀﾉﾖﾐｺﾐ */
}
     }
  }
  fclose( wfi );
    } else {
for ( n = 0 ; n < sou-1 ; n++ )  {
    for ( i = 1 ; i <= node[n+1] ; i++ )  {
for ( j = 0 ; j <= node[n] ; j++ )  {
    w[n][i][j] = 2. * ( ran() - 0.5 ); /* ｵﾓﾐﾃﾞｰﾀﾉｼｮｷｶ */
       }
   }
}
     }
}
/* ********************************************************************* */
/*  ｶﾞｸｼｭｳ                                                               */
/* ********************************************************************* */
void neu_cal( int sou, int larn )
{
    FILE *fe;
    int yn, wmode, i, j, kai, inext;
    float averr, outerr[20];
```

```
    kai = 1;
    wmode = w_ren_mode();                    /* ｵﾓﾐｼｭｳｾｲﾎｳﾉｾﾝﾀｸ */
    printf( "¥nｴﾗｰﾖﾌｧｲﾙﾆｼｭﾂﾘｮｸｼﾏｽｶ? (y/n) : " );
    if ( ( yn = hityn() ) == YES ) {
printf( "ｼｭﾂﾘｮｸｴﾗｰﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ  : " );
fe = openfile( "wt", AGAIN );
    }
    printf( "¥x1b*" );                       /* ﾃｷｽﾄｶﾞﾒﾝﾉｸﾘｱ */
    printf( "¥n" );

    while( 1 ) {
       inext = 0;
       averr = (float)0.0;
       for ( i = 0; i < node[sou-1]; i++ )          /* ｼｭﾂﾘｮｸｴﾗｰﾉｼｮｷｶ */
 outerr[i] = (float)0.0;
       for ( i =  0; i < icount; i++ ) {
 data_set( i, sou ) ;                    /* ﾃﾞｰﾀﾉｾｯﾄ */
 bprun( sou );
 for ( j = 0; j < node[sou-1]; j++ )
     outerr[j] += err[j];
 if ( wmode == 2 ) w_renew( sou );    /* ｵﾓﾐﾉｺｳｼﾝ (ﾂｲｼﾞｼｭｳｾｲ) */
 if ( wmode == 1 ) add_ew( sou, i ); /* ｵﾓﾐﾃﾞｰﾀﾉﾍﾝﾋﾞﾌﾞﾝﾉﾜ */
    }
    if ( wmode == 1 ) w_renew( sou );         /* ｵﾓﾐﾉｺｳｼﾝ (ｲｯｶﾂｼｭｳｾｲ) */
    printf( "¥x1b[1;1H" );                    /* ｶｰｿﾙｲﾁﾉｲﾄﾞｳ */
    printf( "ｶﾞｸｼｭｳｶｲｽｳ = %d¥n", kai );

    for ( i = 0; i < node[sou-1]; i++ ) {
outerr[i] /= (float)icount;
printf( "ｺﾞｻ(%d) = %f¥n", i+1, outerr[i] );
if ( outerr[i] <= 0.01 ) inext++;       /* ｶﾞｸｼｭｳﾉｼｭｳﾘｮｳｼﾞｮｳｹﾝ */
averr += outerr[i];
    }
    averr /= node[sou-1];
    if ( yn == YES ) fprintf( fe, "%d %f¥n", kai, averr );
    if ( inext == node[sou-1] ) break;
    kai++;
    if (larn > 0 && larn < kai ) break;
  }
  if ( yn == YES ) fclose( fe );
  printf( "¥nｶﾞｸｼｭｳｹｯｶﾖｼｭﾂﾘｮｸｼﾏｽｶ? (y/n) : " );
  if ( hityn() == YES ) execute( sou, 1 );
  wf_write( sou );              /* ｴﾗﾚﾀｵﾓﾐﾃﾞｰﾀﾉﾌｧｲﾙﾍﾉｼｭﾂﾘｮｸ */
}
/* ************************************************ */
/*  ｼﾞｯｺｳ                                           */
/* ************************************************ */
void execute( int sou, int mm )
{
    int i, j;

    for ( i = 0; i < icount; i++ ) {
data_set( i, sou );              /* ﾃﾞｰﾀﾉｾｯﾄ */
out_cal( sou, mm );                 /* ｼｭﾂﾘｮｸﾉｹｲｻﾝ */
printf( "¥nﾆｭｳﾘｮｸ :" );
for ( j = 1; j <= node[0]; j++ ) {
     printf( "%1.3f", x[0][j] );
}
printf( "¥nｼｭﾂﾘｮｸ :" );
for ( j = 1; j <= node[sou-1]; j++ ) {
    printf( " %1.3f", x[sou-1][j] );
}
if ( mm == 1 ) {
   printf( "¥nｺﾞｻ :" );
   for ( j = 0; j < node[sou-1]; j++ ) {
printf( "%1.3f", err[j] );
   }
```

```
}
printf( "¥n" );
    }
    printf( "¥n" );
}

/* ****************************************** */
/*  ﾃﾞｰﾀﾉｾｯﾄ                                  */
/* ****************************************** */
void data_set( int patern, int sou )
{
     int i, j, n;

     for ( n = 0 ; n < sou ; n++ ) {
 for ( j = 0 ; j <= node[n] ; j++ ) {
     x[n][j] = (float)0.0;              /* ｼﾞｮｳﾀｲﾉｼｮｷｶ */
 }
     }
     for ( i = 0; i < node[0]; i++ )
 x[0][i+1] = ind[node[0]*patern+i];
     for ( i = 0; i < node[sou-1]; i++ )
 d[i] = oud[node[sou-1]*patern+i];
}
/* ********************************************** */
/*  ﾊﾞｯｸﾌﾟﾛﾊﾟｹﾞｰｼｮﾝｱﾙｺﾞﾘｽﾞﾑ                          */
/* ********************************************** */
void bprun( int sou )
{
     int n , i, j;
/* ************************************************** */
/*  ｲｶﾉﾊｲﾚﾂﾆﾖﾘﾈｯﾄﾜｰｸﾉｷﾎﾞｶﾞｾｲｹﾞﾝｻﾚﾙ                      */
/*  ﾁｭｳｶﾝｿｳﾉｶｽﾞ m(m>5),ｶｸｿｳﾉﾕﾆｯﾄｽｳｼﾞｮｳｹﾞﾝ n(n>20),   */
/*  ｶﾞｸｼｭｳﾊﾟﾀｰﾝｽｳ k(k>50) ﾄｽﾙﾆﾜ,                      */
/*  ex[m-1][n],ey[m-1][n]                             */
/*  ﾄﾍﾝｺｳｽﾙ.                                          */
/* ************************************************** */
    float ex[6][20], ey[6][20];

    out_cal( sou, 1 );                    /* ｼｭﾂﾘｮｸﾉｹｲｻﾝ */
    n = sou - 2;
    for ( i = 1 ; i <= node[n+1] ; i++ ) {
ey[n][i] = x[n+1][i] - d[i-1];
ex[n][i] = ey[n][i] * x[n+1][i] * ( 1. - x[n+1][i] );
for ( j = 0 ; j <= node[n] ; j++ ) {
    ew[n][i][j] = ex[n][i] * x[n][j];
}
    }
    if ( sou > 2 ) {
       for ( n = sou - 3 ; n >= 0 ; n-- ) {
   for ( j = 0 ; j <= node[n+1] ; j++ ) {
       ey[n][j] = (float)0.0;
       for ( i = 1 ; i <= node[n+2] ; i++ ) {
   ey[n][j] += ( w[n+1][i][j] * ex[n+1][i] );
       }
       ex[n][j] = ey[n][j] * x[n+1][j] * ( 1. - x[n+1][j] );
  }
       }
       for ( n = sou - 3 ; n >= 0 ; n-- ) {
   for ( i = 1 ; i <= node[n+1] ; i++ ) {
for ( j = 0 ; j <= node[n] ; j++ ) {
    ew[n][i][j] = ex[n][i] * x[n][j];
}
     }
  }
       }
    }
```

```
/* ************************************************ */
/*  ｼｭﾂﾘｮｸﾉｹｲｻﾝ                                     */
/* ************************************************ */
void out_cal( int sou, int mm )
{
    int n , i , j;
    for ( n = 0; n < sou - 1; n++ )
x[n][0] = 1.0;                    /* ｼｷｲﾁﾌﾞﾝ */
    for ( n = 0; n < sou-1; n++ ) {
for ( i = 1; i <= node[n+1]; i++ ) {
    for ( j = 0; j <= node[n]; j++ ) {
x[n+1][i] += w[n][i][j] * x[n][j];
    }
    x[n+]][i] = (float)( 1. / (1. + exp( (double)(-x[n+1][i]) ) ) );
}
     }
     if ( mm == 1 ) {
for ( i = 1; i <= node[sou-1]; i++ ) {        /* ｴﾗｰﾉｹｲｻﾝ */
    err[i-1] = ( x[sou-1][i] - d[i-1] ) * ( x[sou-1][i] - d[i-1] );
 }
      }
}
/* ************************************************************ */
/*  ｵﾓﾐﾃﾞｰﾀﾉﾋﾞﾌﾞﾝﾜﾉｶｽﾞ                                        */
/* ************************************************************ */
void add_ew( int sou, int patern )

{
       int n, i, j;
/* ********************************************************************** */
/* ｲｶﾉﾊｲﾚﾂﾆﾖﾘﾈｯﾄﾜｰｸﾉｷﾎﾞｶﾞｾｲｹﾞﾝｻﾚﾙ.                                          */
/* ﾁｭｳｶﾝｿｳﾉｶｽﾞ m(m>5).ｶｸｿｳﾉﾕﾆｯﾄｽｳｼﾞｮｳｹﾞﾝ n(n>20),                          */
/* ｶﾞｸｼｭｳﾊﾟﾀｰﾝｽｳ k(k>50) ﾄｽﾙﾆﾜ.                                              */
/* ew1[m-1][n] ﾄﾍﾝｺｳｽﾙ.                                                      */
/* ********************************************************************** */
    float ew1[6][20][20];

    if ( patern == 0 ) {
for ( n = 0; n < sou - 1; n++ ) {
    for ( i = 1; i <= node[n+1]; i++ ) {
for ( j = 0; j <= node[n]; j++ ) {
    ew1[n][i][j] = (float)0.0;
 }
     }
 }
    }
    for ( n = 0; n < sou - 1; n++ ) {
for ( i = 1; i <= node[n+1]; i++ ) {
    for ( j = 0; j <= node[n]; j++ ) {
ew1[n][i][j] += ew[n][i][j];
     }
 }
     }
     if (patern == 3 ) {
 for ( n = 0; n < sou - 1; n++ ) {
     for ( i = 1; i <= node[n+1]; i++ ) {
 for ( j = 0; j <= node[n]; j++ ) {
     ew[n][i][j] =ew1[n][i][j];
 }
     }
  }
       }
}
 /* ********************************** */
 /* ｵﾓﾐﾉｺｳｼﾝ                           */
 /* ********************************** */
 void w_renew( int sou )
```

```
 {
     int n, i, j;

     for ( n = sou - 2; n >= 0; n-- ) {
 for ( i = 1; i <= node[n+1]; i++ ) {
     for ( j = 0; j <= node[n]; j++ ) {
 dw[n][i][j] = - eta * ew[n][i][j] + alphar * dw[n][i][j];
 w[n][i][j] += dw[n][i][j];
       }
     }
   }
}
/* ************************************************** */
/* ﾗﾝｽｳﾉﾊｯｾｲ                                          */
/* ************************************************** */
float ran( void )
{
    static long ir =5L;
    float ranran;

    ir = 163L * ir + 656329L;
    ir = ir % 12518383L;
    ranran = (float)ir / 12518383.0;
    return( ranran );
}
/* ********************************************************** */
/*  ｵﾓﾐﾃﾞｰﾀﾉﾌｧｲﾙﾍﾉｼｭﾂﾘｮｸ                                    */
/* ********************************************************** */
void wf_write( int sou )
{
    FILE *wfo;
    int i, j, n;

    printf( "¥nｼｭﾂﾘｮｸｵﾓﾐﾌｧｲﾙﾒｲﾜﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ    :");
    wfo = openfile( "wt", AGAIN );
    for ( n = 0 ; n < sou-1 ; n++ ) {
fprintf( wfo, "***%d->%d***¥n", n+1, n+2 );
for (i = 1 ; i <= node[n+1] ; i++ ) {
    for ( j = 0 ; j <= node[n] ; j++ ) {
fprintf( wfo , "%f¥n" , w[n][i][j] );
    }
}
    }
    fclose( wfo );
}
/* ********************************** */
/* ｷｰﾆｭｳﾘｮｸｽｳｼﾞﾉｶｸﾆﾝ                   */
/* ********************************** */
int chknum( int nu , int np )
{
    int num;

    while( 1 ) {
scanf( "%d" , &num );
if ( num >= nu && num <= np ) break;
rewind( stdin );
   }
   return( num );
}
/* ************************************* */
/*  y ・ nｷｰﾉｼｭﾄｸ                        */
/* ************************************* */
int hityn( void )
{
     int c;

     for( ; ; ) {
```

```
c = getchar();
if ( c == 'y'|| c == 'Y' ) return( YES );
if ( c == 'n'|| c == 'N' ) return( NO );
     }
}
/* ********************************************************** */
/*  ｶｸｼｭｳ･ｼｯｺｳ･ｼｭｳﾘｮｳﾉｾﾝﾀｸ                                    */
/* ********************************************************** */
int mode_sel( void )
{
    int mode;

    printf( "¥n1.ｶｸｼｭｳ" );
    printf( "¥n2.ｼｯｺｳ" );
    printf( "¥n3.ｼｭｳﾘｮｳ" );
    printf( "¥nﾓｰﾄﾞ(1-3)ﾖｾﾝﾀｸｼﾃｸﾀﾞｻｲ : " );
    mode = chknum( 1, 3 );
    return( mode );
}
/* ********************************************** */
/*  ｵﾓﾐｼｭｳｾｲﾎｳﾉｾﾝﾀｸ                                */
/* ********************************************** */
int w_ren_mode( void )
{
      int wmode;

      printf( "¥nｵﾓﾐﾉｼｭｳｾｲﾎｳﾎｳ" );
      printf( "¥n1.ｲｯｶﾂｼｭｳｾｲﾎｳ" );
      printf( "¥n2.ﾂｲｼﾞｼｭｳｾｲﾎｳ" );
      printf( "¥nﾄﾞﾁﾗﾆｼﾏｽｶ? : " );
      wmode = chknum( 1, 2 );
      return( wmode );
}
/* ********************************************* */
/*  ﾌｧｲﾙﾉｵｰﾌﾟﾝ                                    */
/* ********************************************* */
FILE *openfile( char *mode, int sw )
{
     char fn[25];
     FILE *fp;

     while( 1 ) {
  scanf( "%s", fn );
  if ( ( fp = fopen( fn , mode ) ) == NULL ) {
     printf( "%s ｶﾞｵｰﾌﾟﾝﾃﾞｷﾏｾﾝﾃﾞｼﾀ。¥n", fn );
     if ( sw == EXIT ) exit( 1 );
     printf( "ﾓｳｲﾁﾄﾞﾌｧｲﾙﾒｲﾖﾆｭｳﾘｮｸｼﾃｸﾀﾞｻｲ : " );
 } else {
      break;
 }
     }
     return( fp );
}
```

表4.1では、学習率を0.3とし、学習回数は、1389回行った場合、入力（0.0、0.0）に対して答えが0.128となった。更に、（0.0、1.0）に対して、0.881となり、1.0に近い値となった。また、（1.0、0.0）に対しては、0.943を得た。この結果は、非常に1.0に近い値である。（1.0、1.0）の入力に対しては、0.078という出力がでた。これらの4個の出力は、排他的論理和の計算を満足する結果となった。

誤差評価方法はここでは詳しく述べないが、各学習データーに対する誤差は各教師データ（1.0または0.0）と実際の出力の差の二乗が個々の誤

差である。全体誤差はそれらの平均をとったものである。学習は、全体誤差が0.01未満となったときに終了するようにプログラム化されている。表4.1に対して、誤差について詳しく見ると、(0.0、0.0) に対しては、0.016、(0.0、1.0) に対しては、0.014、(1.0、0.0) に対しては、0.0003、(1.0、1.0) に対しては、0.006となった。その平均が0.009985である。

学習率を0.5としたときの結果は表4.2に示すが、平均誤差は、0.009966となっており、良好な結果であった。しかし、表4.3および表4.4に示すように学習率を0.6にした場合には、学習回数を1000回、1500回と増加しても良い結果は得られなかった。

## 5. 学習率の問題点について

4章で述べたように、学習率$\eta$は、経験的に定めた定数である。$\eta$は小さい値にすれば、学習回数を増加させることになり、時間がかかる。逆に$\eta$を大きくすれば、収束しないで、発散する結果となってしまう。研究目標としては、速い収束に対する$\eta$の決定法の確立である。文献[2]によれば、個々の結合係数$w_{ij}^{(m)}$に対して可変な学習率を決定し、より効果的な更新を行うためにHesse行列を求めることであると報告されている。また、この計算は、非常にむずかしいとも報告されている。この問題は、今後の課題である。

## 6. おわりに

本研究では、ニューラルネットワークの重み係数の更新方法の物理的な意味と、更新に対する問題点を考察した。特に、バックプロパゲーション則の場合出力層に対する偏微分式の一部は、最後に計算される入力層の偏微分式まで影響を与え、結局すべての偏微分式に影響を与える事がわかった。後半ではおもに、学習率の決定に関する問題点について述べた。実際に計算機シュミレーションを行って、問題点を指摘した。

## 7. 謝辞

本研究に対して、ご指導いただいた、東京工業大学美多勉教授、特別研究員東南大学忻副教授、劉延年副教授、並びにＵＮＩＸ関係でいろいろお世話になった種子田昭彦院生や他の院生に感謝いたします。

## 8. 参考文献

1）八名和夫　鈴木義武共著：ニューロコンピューティング、海文堂、p44-p49、1992年

2）Robert Hecht Nielsen：Neurocomputing、p124-p137、1990年、Addison Wesley

3）松井伸之　石見憲一：しきい値ゆらぎをもつニューロンモデルを用いた階層型ニューラルネットワーク、電気学会論文誌C、114巻11号、p1208-p1213、1994年

4）八名和夫　鈴木義武共著：ニューロ情報処理技術、海文堂、p73-p85、1992年