AI 勉強会（第 1 回）

木更津工業高等専門学校
大枝真一

2017 年 9 月 19 日

1　はじめに

ニューラルネットワークは，脳の神経回路網をモデル化した機械学習の手法である．近年，深層学習（Deep Learning）の登場により再びニューラルネットワークが注目を集めている．深層学習は，画像認識，音声認識，自然言語処理など様々な分野で取り入れられており，Google 社をはじめ，多くの企業で利用，研究がされている．本資料では，ニューラルネットワークの基本的な原理，学習方法について記述する．

2　基本構造

ニューラルネットワークは，複数のニューロンとニューロン同士を結合するシナプスから構成される．シナプスは各結合の強さを表す結合重みを持っている．ニューラルネットワークの図を図 1 に示す．

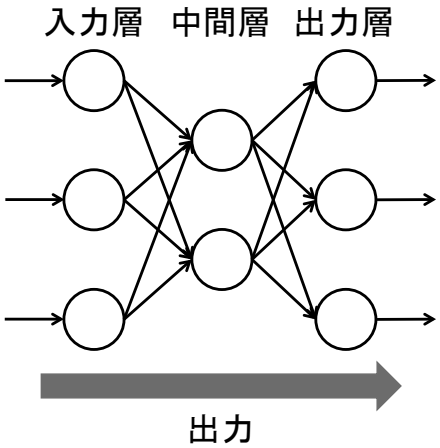

図 1　ニューラルネットワーク

ニューラルネットワークは，入力層，中間層，出力層の多層構造になっており，各ニューロンは前の層のニューロンの出力を入力として受け取り，計算を行ったあと，次のニューロンへと出力する．各ニューロンの出力が次々と伝播していき，出力層のニューロンからネットワークの出力を得る．ニューロンの構造を図 2 に示す．

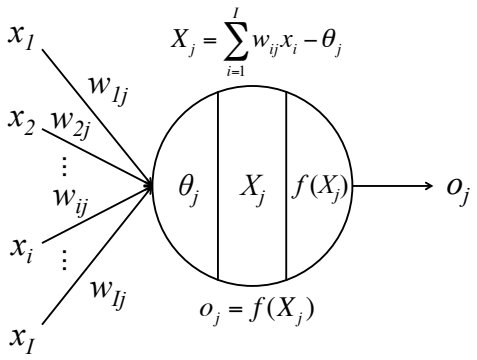

図 2　ニューロンのモデル

ニューロンは複数の入力を受け取り，1 つの値を出力する．ある層のニューロン j は，その前の層の I 個のニューロンから入力 x を受け取り，重み付き和を計算する．その重み付き和からニューロン j のバイアス θ_j を引いた値を活性化関数の引数としてその値を出力する．これらを式で表すと以下のようになる．

$$X_j = \sum_{i=1}^{I} w_{ij} x_i - \theta_j \tag{1}$$

$$o_j = f(X_j) \tag{2}$$

活性化関数には，以下のシグモイド関数や双曲線正接関数などが使われる．

$$f(x) = \frac{1}{1 + \exp(-\epsilon x)} \tag{3}$$

$$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{4}$$

シグモイド関数と双曲線正接関数のグラフは図 3 のような形となる．

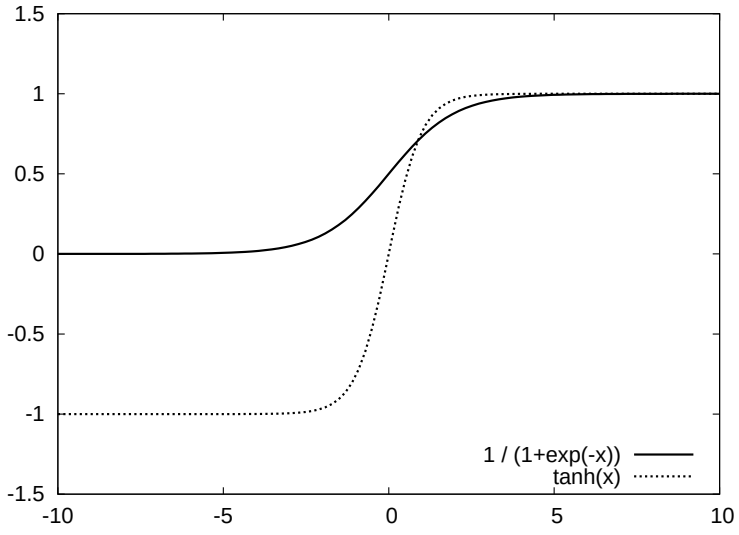

図 3　シグモイド関数と双曲線正接関数のグラフ

近年，これらの活性化関数の代わりに正規化線形関数が利用されることが多くなっている．正規化線形関数の式とグラフを以下に示す．

$$f(x) = \max(0, x) \tag{5}$$

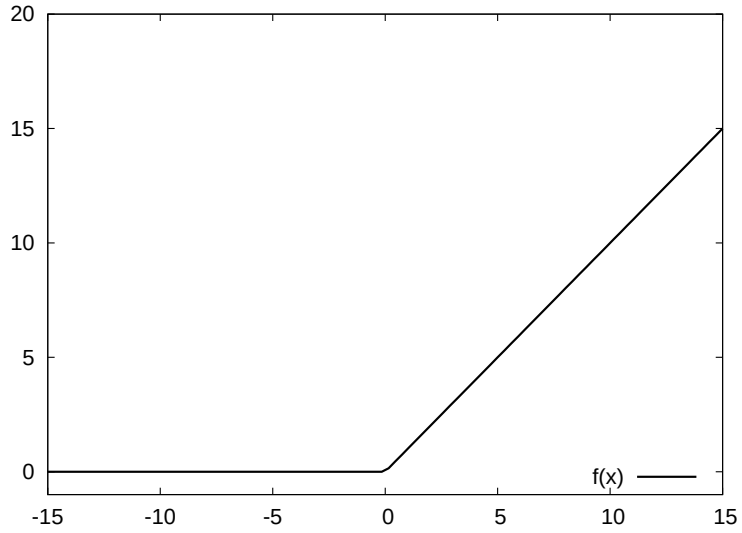

図 4　正規化線形関数のグラフ

なお，正規化線形関数を活性化関数とするニューロンのことを ReLU（Rectified Linear Unit）という．以上の計算を入力層側のニューロンから順に行う．ただし，入力層のニューロンはこれらの計算を行わず，入力をそのまま出力する．この前向きの計算により，入力層への入力に対し，中間層を経て，出力層から出力を得ることができる．

3　学習の目的

ニューラルネットワークは，理想の出力に近いほど性能が高いといえる．各出力ニューロン k に対応する教師信号 t_k が与えられたとき，各入力パターンに対する誤差 E_p と P 個の入力パターンの誤差の総和 E_{all} は以下の式で表すことができる．

$$E_p = \frac{1}{2}\sum_{k=1}^{K}(o_k - t_k)^2 \tag{6}$$

$$E_{all} = \sum_{p=1}^{P} E_p \tag{7}$$

ニューラルネットワークの学習は，この誤差総和 E_{all} を最小化し，入力に対して適切な出力が得られるように各結合重みを調整することである．

4　仮想ニューロン

ニューラルネットワークの出力を最適化するには，各結合重みと閾値を調整すれば良いが，それぞれを修正するのは手間がかかる．そこで閾値を重みとして扱うために仮想ニューロンを導入する．入力層，中間層，出力層のニューロン数がそれぞれ I 個，J 個，K 個のニューラルネットワークに仮想ニューロンを導入した形を図 5 に示す．

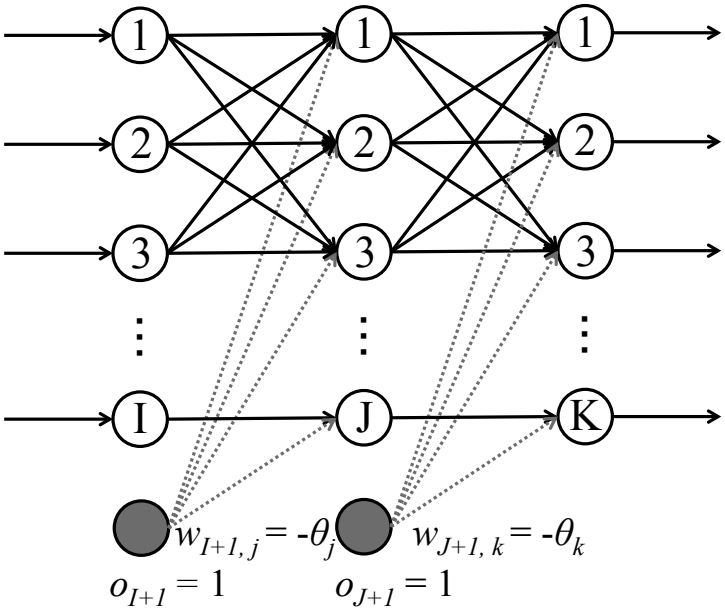

図 5　仮想ニューロン

仮想ニューロンは出力層を除く，各層のニューロンの最後に追加する．図 5 のニューラルネットワークでは，$I+1$ 番目，$J+1$ 番目に追加している．各仮想ニューロンは次の層の全ニューロンと結合しており，結合先のニューロンの閾値を結合重みとしている．また，各仮想ニューロンは常に 1 を出力している．これにより，式 (1) は以下のように置き換えられる．

$$
\begin{aligned}
X_j &= \sum_{i=1}^{I} w_{ij} o_i - \theta_j \\
&= \sum_{i=1}^{I} w_{ij} o_i + w_{I+1,j} \\
&= \sum_{i=1}^{I} w_{ij} o_i + w_{I+1,j} \cdot 1 \\
&= \sum_{i=1}^{I} w_{ij} o_i + w_{I+1,j} \cdot o_{I+1} \\
&= \sum_{i=1}^{I+1} w_{ij} o_i
\end{aligned}
\tag{8}
$$

式 (8) より，閾値も重みとして学習により適切に修正が可能となる．

5　最急降下法

ニューラルネットワークの重みを修正するための手法として最急降下法を用いる．最急降下法は，関数 $f(x)$ の傾きから繰り返し x を更新し，極小値を求める反復法である．最急降下法の更新は以下の式で行う．

$$x = x - \eta \frac{\partial f(x)}{\partial x} \tag{9}$$

η は学習係数で $\eta = 0.01$ など小さい値をあらかじめ決めておく．η を大きくするほど学習速度は向上するが大きすぎると収束せず振動や発散をしてしまう．逆に小さいほど，学習は安定して収束するが学習速度が低下してしまう．したがって，η は適切な値の設定が必要となる．

以下の図 6 ような 2 次関数の場合を考えてみる．

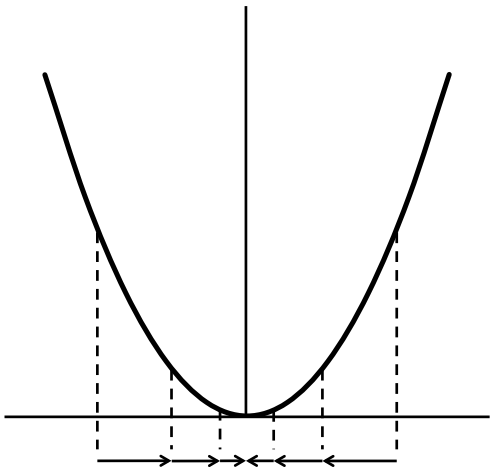

図 6　最急降下法

最初に適当に x の初期値を決める．初期値から偏微分 $\frac{\partial f(x)}{\partial x}$ を求め，式 (9) に代入し，x を更新する．このとき，x が極小値のときよりも大きければ傾きは $\frac{\partial f(x)}{\partial x} > 0$ となるため，x は小さくなり，$f(x)$ は極小値に近づく．それに対して，x が極小値のときよりも小さければ傾きは $\frac{\partial f(x)}{\partial x} < 0$ となるため，x は大きくなり，$f(x)$ はやはり極小値に近づく．このように x がどちらの場合でも更新後 $f(x)$ は極小値に近づく．また，傾きは極小値に近づくほどゆるやかになるため，修正量も小さくなり，η が適切な値の場合は，振動をせずに収束に向かう．このように最急降下法では，着実に極小値には近づくが，関数の形状や初期値によっては，最小値を求めることができない場合がある．

6　誤差逆伝播法

最急降下法を用いて図 7 のようにニューラルネットワークの出力層から入力層に向かって重みの調整を行う手法を誤差逆伝播法（Back Propagation）という．

学習の目的は誤差総和 E_{all} を最小化することだが E_{all} を最小化することは困難であるため，各パターンの誤差 E_p が最小となるように重みを更新する．重みが変化すると出力と誤差も変化する

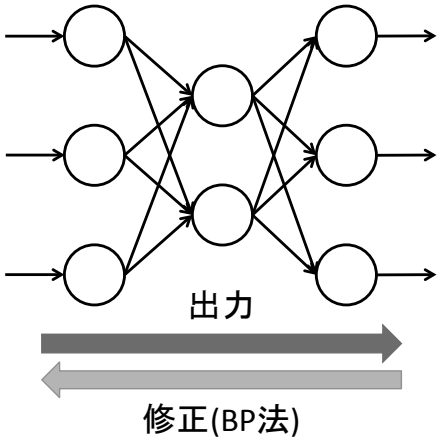

図 7　誤差逆伝播法

ため，E_p は w を引数とした関数 $E_p(w)$ と考えることができる．したがって，$\frac{\partial E_p(w)}{\partial w}$ を求めて，w を更新することで E_p を小さくすることができる．最急降下法を用いた重み w の更新式は以下のようになる．

$$w = w - \eta \frac{\partial E_p}{\partial w} \tag{10}$$

式 (10) を用いて各重みを出力層から入力層に向かって順に更新していく．図 9 のようなニューラルネットワークの場合を考える．

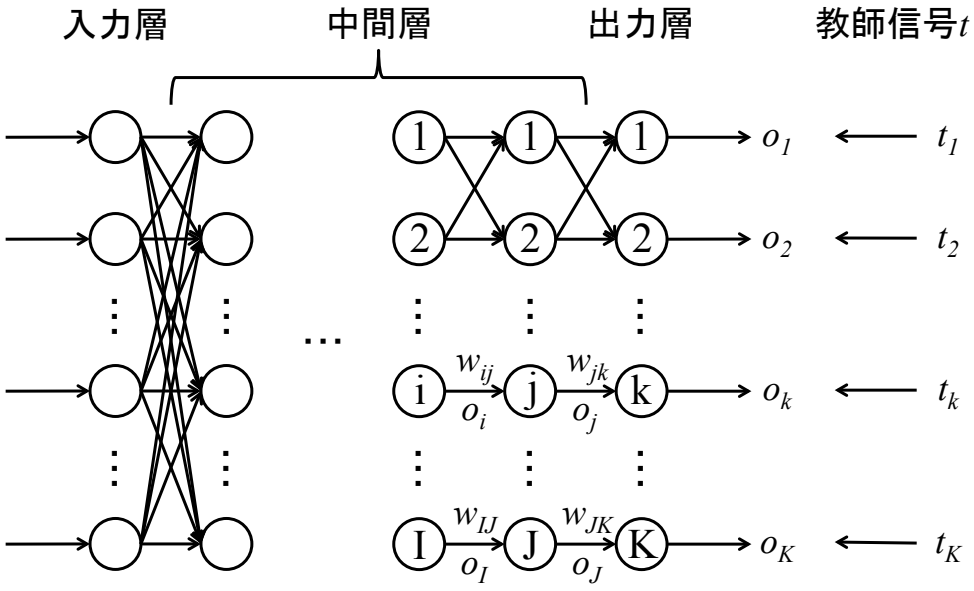

図 8　誤差逆伝播法の手順

最初に中間層-出力層の任意の重み w_{jk} について考える．E_p の式には直接 w_{jk} は存在しないので合成関数の微分を考える．$\frac{\partial E_p}{\partial w_{jk}}$ の合成関数の微分は以下のようになる．

$$\frac{\partial E_p}{\partial w_{jk}} = \frac{\partial E_p}{\partial o_k} \frac{\partial o_k}{\partial X_k} \frac{\partial X_k}{\partial w_{jk}} \tag{11}$$

式 (11) の偏微分をそれぞれ計算すると以下のようになる．

$$\frac{\partial E_p}{\partial o_k} = \frac{1}{2}\{2(o_k - t_k)\} = o_k - t_k \tag{12}$$
$$\frac{\partial o_k}{\partial X_k} = f'(X_k) \tag{13}$$
$$\frac{\partial X_k}{\partial w_{jk}} = o_j \tag{14}$$

以上より，中間層-出力層の重みの更新式は以下のようになる．

$$w_{jk} = w_{jk} - \eta(o_k - t_k)f'(X_k)o_j \tag{15}$$

さらに出力ニューロンに関係する式を δ_k としてまとめると式 (15) は次のように表せる．

$$\delta_k = \frac{\partial E_p}{\partial o_k}\frac{\partial o_k}{\partial X_k} \tag{16}$$
$$= (o_k - t_k)f'(X_k) \tag{17}$$
$$w_{jk} = \quad w_{jk} - \eta\delta_k o_j \tag{18}$$

次に中間層以下の重み w_{ij} について考える．中間層以下の重みの合成関数の微分は以下のようになる．

$$\frac{\partial E_p}{\partial w_{ij}} = \sum_{k=1}^{K} \frac{\partial E_p}{\partial o_k}\frac{\partial o_k}{\partial X_k}\frac{\partial X_k}{\partial o_j}\frac{\partial o_j}{\partial X_j}\frac{\partial X_j}{\partial w_{ij}} \tag{19}$$

式 (19) では，注目ニューロンより上位の層で結合している全てのニューロンの誤差を考慮するために $\sum$ を取っている．$\frac{\partial E_p}{\partial o_k}\frac{\partial o_k}{\partial X_k}$ は式 (17) の δ_k が代入できるので，それ以外の微分について考える．各偏微分は以下のように計算できる．

$$\frac{\partial X_k}{\partial o_j} = w_{jk} \tag{20}$$
$$\frac{\partial o_j}{\partial X_j} = f'(X_j) \tag{21}$$
$$\frac{\partial X_j}{\partial w_{ij}} = o_i \tag{22}$$

以上より，中間層以下の重みの更新式は以下のようになる．

$$\delta_j = \sum_{k=1}^{K} \frac{\partial E_p}{\partial o_k}\frac{\partial o_k}{\partial X_k}\frac{\partial X_k}{\partial o_j}\frac{\partial o_j}{\partial X_j} \tag{23}$$
$$= \sum_{k=1}^{K} \delta_k w_{jk} f'(X_j) \tag{24}$$
$$w_{ij} = w_{ij} - \eta\delta_j o_i \tag{25}$$

以後，同じように繰り返し δ を求めていけば，全ての重みを式 (25) で修正することができる．活性化関数がシグモイド関数の場合，$f'(x)$ は以下のようになる．

$$
\begin{aligned}
f'(x) &= \frac{\epsilon \exp(-\epsilon x)}{(1+\exp(-\epsilon x))^2} \qquad (26)\\
&= \epsilon \cdot \frac{\exp(-\epsilon x)}{1+\exp(-\epsilon x)} \cdot \frac{1}{1+\exp(-\epsilon x)}\\
&= \epsilon \cdot \frac{1+\exp(-\epsilon x)-1}{1+\exp(-\epsilon x)} \cdot \frac{1}{1+\exp(-\epsilon x)}\\
&= \epsilon \cdot \left(1 - \frac{1}{1+\exp(-\epsilon x)}\right) \cdot \frac{1}{1+\exp(-\epsilon x)}\\
&= \epsilon(1-f(x))f(x)
\end{aligned}
$$

$f(X_k) = o_k$ より

$$
f'(X_k) = \epsilon(1-o_k)o_k \qquad (27)
$$

このようにシグモイド関数の場合は，各ニューロンの出力から $f'(X_k)$ を求めることができる．この式を用いると各重みと閾値の更新式は以下のようになる．

- 中間層-出力層

$$
\begin{aligned}
\delta_k &= (o_k - t_k)\epsilon(1-o_k)o_k \qquad (28)\\
w_{jk} &= w_{jk} - \eta\delta_k o_j \qquad (29)\\
\theta_k &= \theta_k - \eta\delta_k \qquad (30)
\end{aligned}
$$

- 中間層以下

$$
\begin{aligned}
\delta_j &= \sum_{k=1}^{K} \delta_k w_{jk}\epsilon(1-o_j)o_j \qquad (31)\\
w_{ij} &= w_{ij} - \eta\delta_j o_i \qquad (32)\\
\theta_j &= \theta_j - \eta\delta_j \qquad (33)
\end{aligned}
$$

付録 A　合成関数の偏微分

合成関数の偏微分法

$z = f(x, y)$ が x, y で全微分可能であり，$x = x(t), y = y(t)$ が t で微分可能ならば，
合成関数 $z = f(x(t), y(t))$ は t 微分可能であり，次の式が成り立つ．

$$\frac{dz}{dt} = \frac{\partial z}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial z}{\partial y}\frac{\partial y}{\partial t} \tag{34}$$

図でイメージすると，z は x と y の関数であり，x と y は t の関数である．したがって，t から x, y への線分と，x から z，y から z への線分が必要となる．

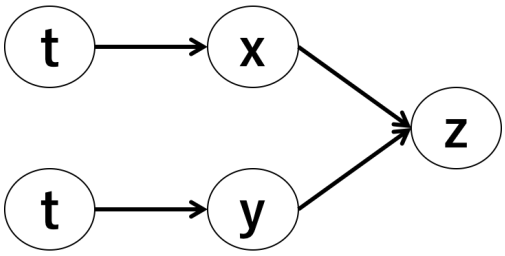

図 9　合成関数の関係