

知の探究

5. AI・機械学習

堀田 敬介

神経細胞とニューラルネットワーク

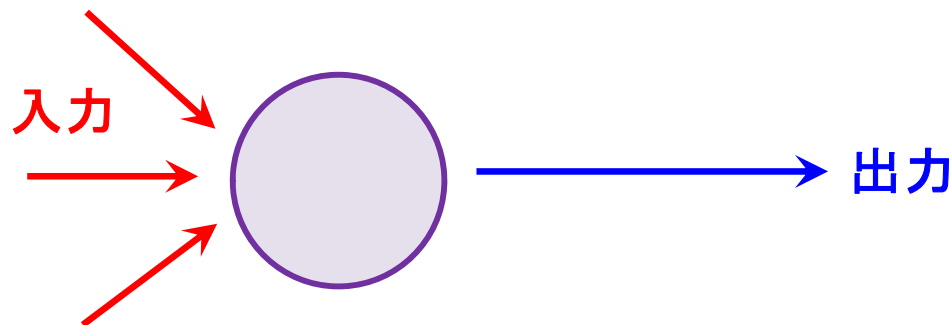
- 神経細胞 neuron



1. 樹状突起に, 他の神経細胞とのシナプス synapse が多数ある
2. シナプス部位で情報を受け取る(**入力**)
3. 細胞体内で活動電位を発生させ電気信号に変換, 軸索を通り軸索末端へ
4. 軸索末端が神経伝達物質を放出(次の神経細胞へ**出力**)

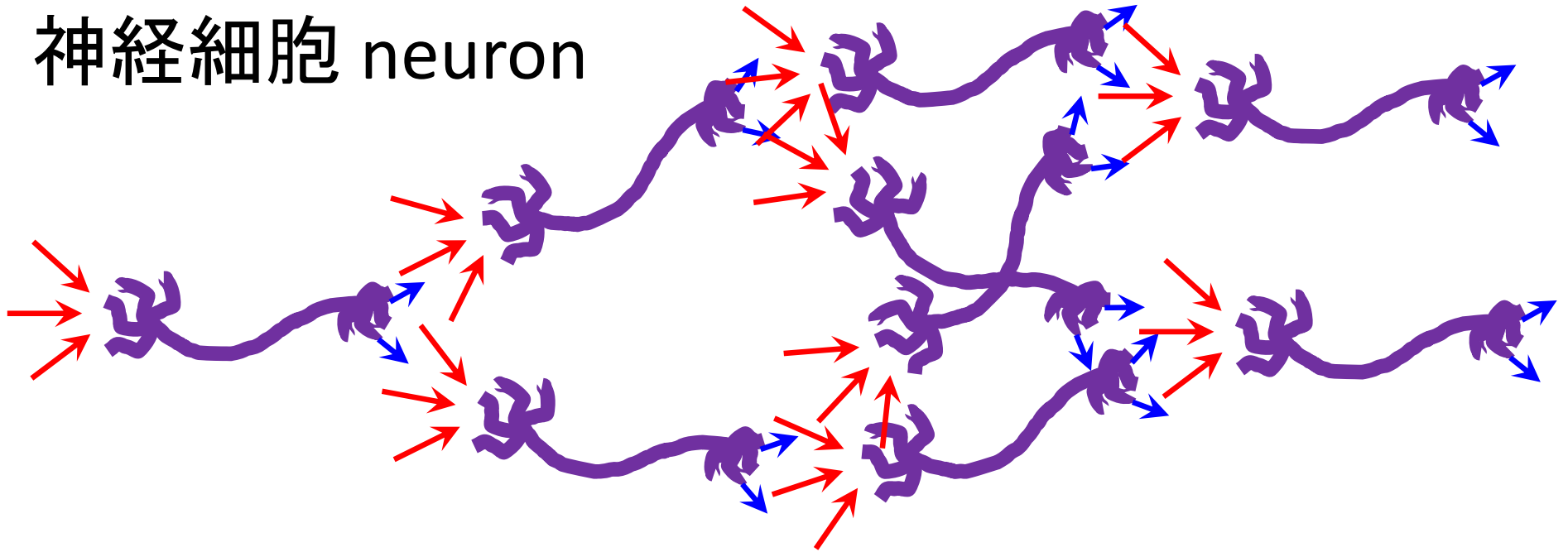
- ニューラルネットワーク neural network

- 神経細胞をグラフ $G = (V, E)$ でモデル化

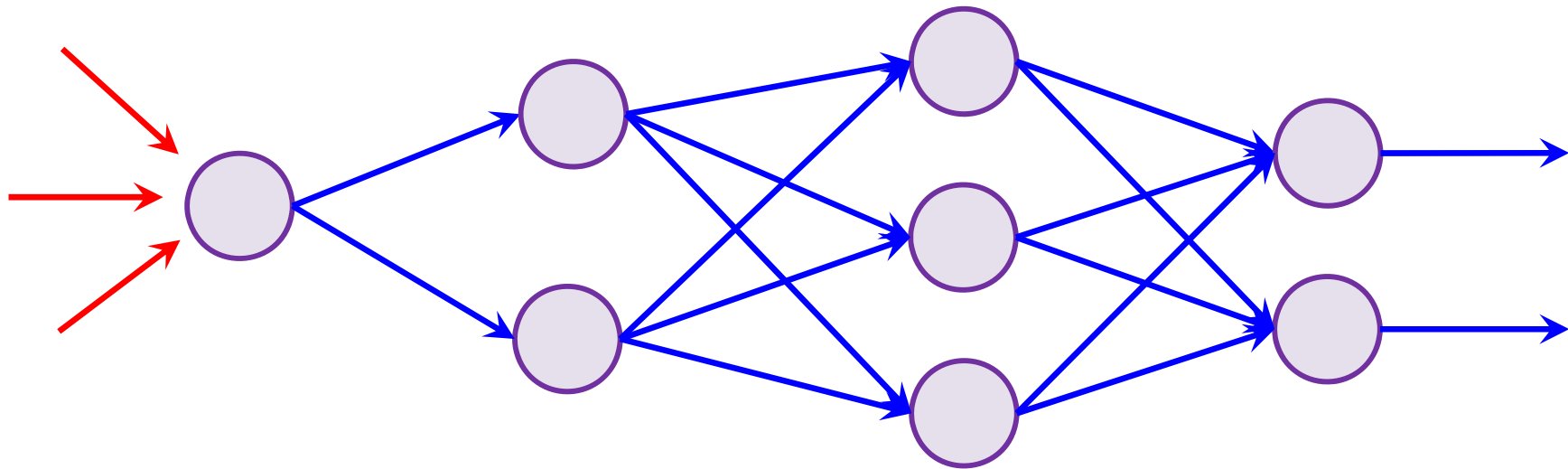


神経細胞とニューラルネットワーク

- 神経細胞 neuron

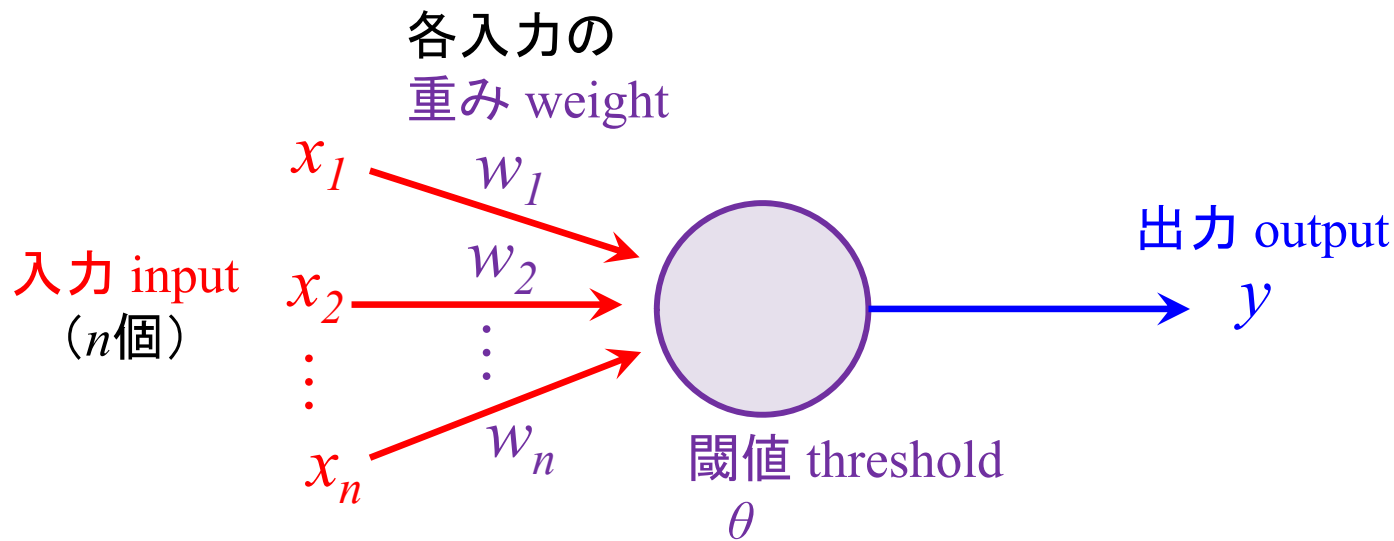


- ニューラルネットワーク neural network



神経細胞とニューラルネットワーク

- 基本ユニットと入力から出力までの伝達(計算)



※出力値は1つだが、次の複数のユニットに伝達される

$$\begin{aligned} s &= w_1x_1 + w_2x_2 + \dots + w_nx_n - \theta \\ &= \mathbf{w}^T \mathbf{x} \end{aligned}$$

$$y = a(s)$$

※ a は活性化関数 activation function

$$\text{ただし, } \mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \\ \theta \end{pmatrix}, \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \\ -1 \end{pmatrix}$$

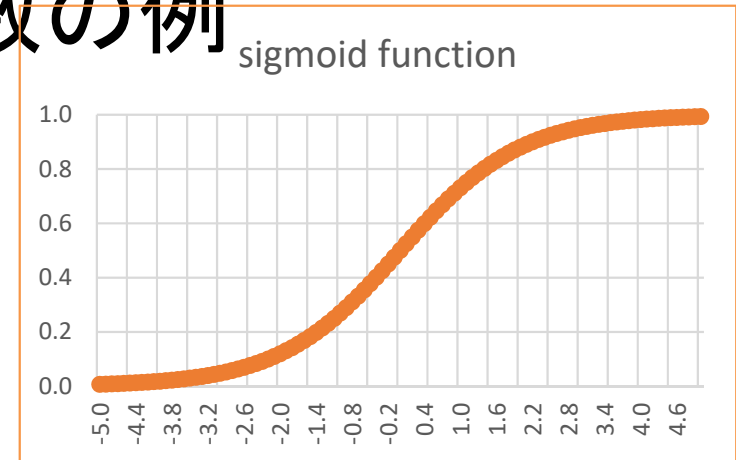
※重みと閾値 w は各ユニットがもつ固有の値

神経細胞とニューラルネットワーク

- 活性化関数として使われる関数の例

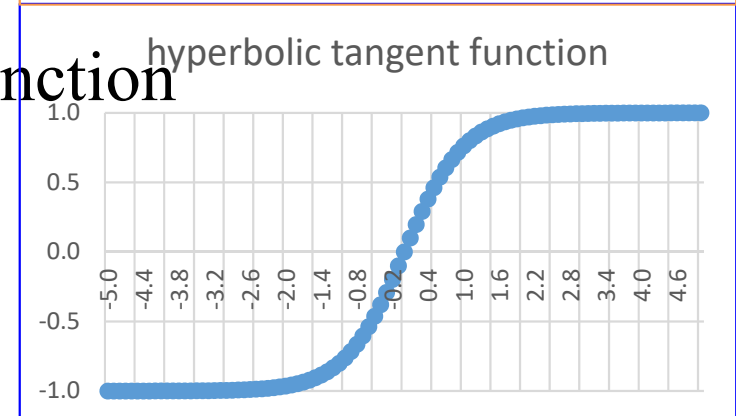
- シグモイド関数 sigmoid function

$$\sigma(s) = \frac{1}{1 + e^{-s}} \in (0, 1)$$



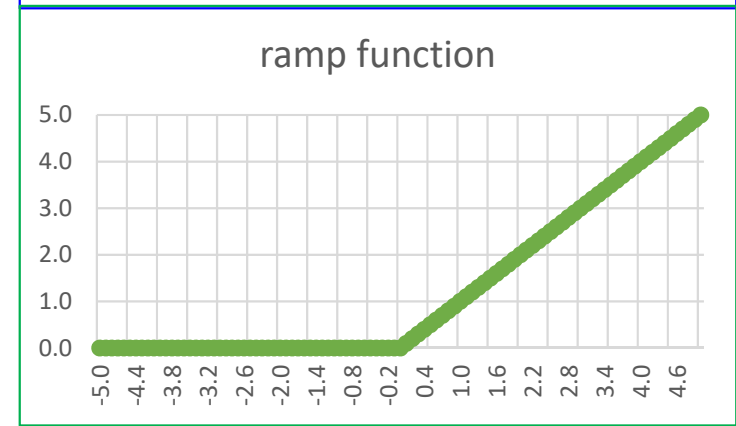
- 双曲線正接関数 hyperbolic tangent function

$$\tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}} \in (-1, 1)$$



- ランプ関数 ReLU ramp function

$$\text{ReLU}(s) = \begin{cases} s & (s \geq 0) \\ 0 & (s < 0) \end{cases}$$



- etc.

ニューラルネットワークと機械学習

✓ 例: 4 × 3画素の画像識別

- 各画素は 0 or 1 の2値とする
- 画像例

0	1	1
1	0	0
1	0	0
0	1	1

1	1	1
0	1	0
0	1	0
0	1	0

- 入力(12個)となる

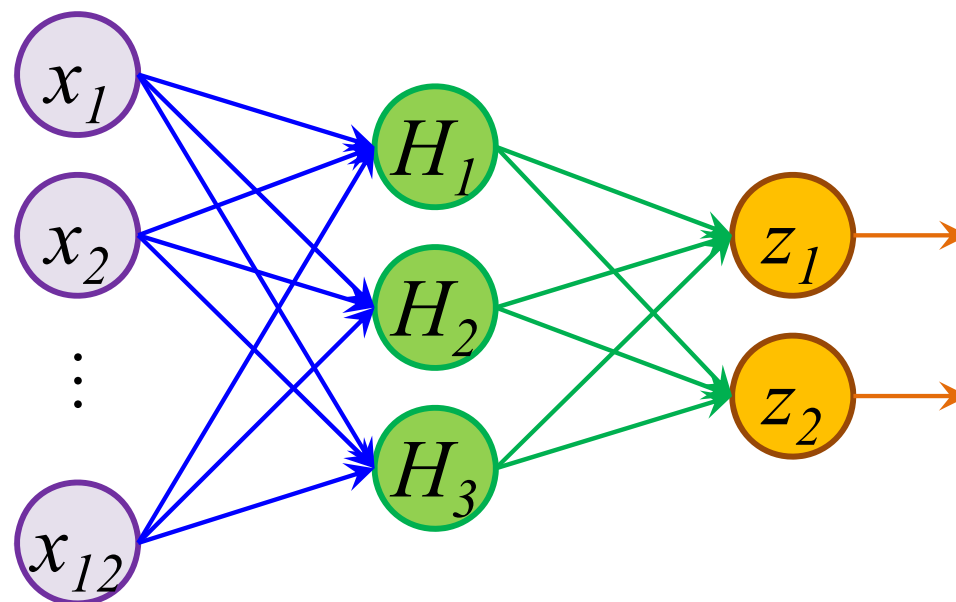
x_1	x_2	x_3
x_4	x_5	x_6
x_7	x_8	x_9
x_{10}	x_{11}	x_{12}



入力層
input layer

隠れ層
hidden layer

出力層
output layer



※ネットワークを多層にしたのが
深層学習 Deep Learning

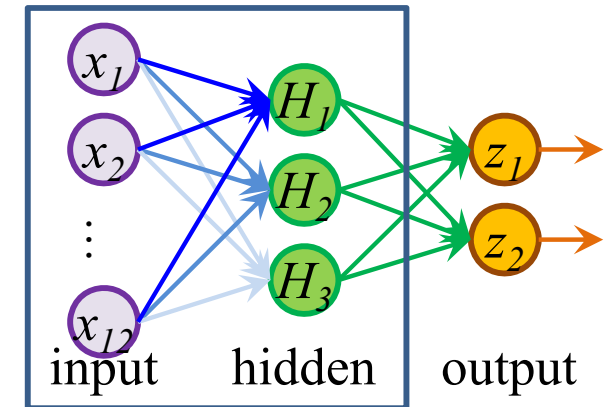
＜深層学習 Deep Learning の例＞

- 畳み込みニューラルネットワーク
CNN; Convolutional Neural Network
- 回帰型ニューラルネットワーク
RNN; Recurrent Neural Network
- 深層強化学習 DQN; Deep Q-Network

ニューラルネットワークと機械学習

➤ *input layer* → *hidden layer*

$$\begin{cases} s_1^H = w_{11}^H x_1 + \cdots + w_{12,1}^H x_{12} - \theta_1^H = (\mathbf{w}_1^H)^T \mathbf{x} \\ s_2^H = w_{12}^H x_1 + \cdots + w_{12,2}^H x_{12} - \theta_2^H = (\mathbf{w}_2^H)^T \mathbf{x} \\ s_3^H = w_{13}^H x_1 + \cdots + w_{12,3}^H x_{12} - \theta_3^H = (\mathbf{w}_3^H)^T \mathbf{x} \end{cases}$$



$$\text{ただし, } \mathbf{w}_i^H = \begin{pmatrix} w_{1i}^H \\ w_{2i}^H \\ \vdots \\ w_{12,i}^H \\ \theta_i^H \end{pmatrix}, \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{12} \\ -1 \end{pmatrix}$$

$(i = 1, 2, 3)$



この値を求めたい(決めたい)！
人間がこの値を設定するのではなく、
機械に自ら設定させる(＝機械学習)

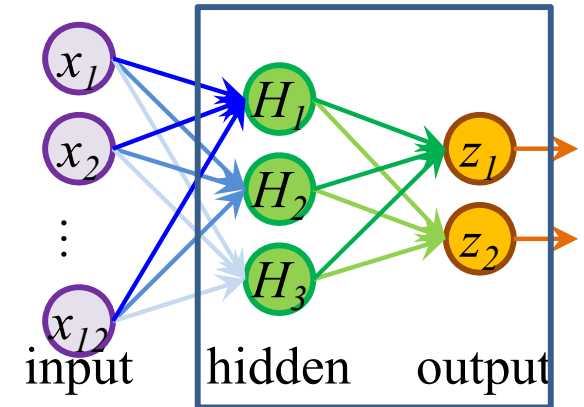
$$\begin{cases} h_1 = a(s_1^H) \\ h_2 = a(s_2^H) \\ h_3 = a(s_3^H) \end{cases}$$

※ a は活性化関数 activation function

ニューラルネットワークと機械学習

➤ *hidden layer* → *output layer*

$$\begin{cases} s_1^o = w_{11}^o h_1 + w_{21}^o h_2 + w_{31}^o h_3 - \theta_1^o = (\mathbf{w}_1^o)^T \mathbf{h} \\ s_2^o = w_{12}^o h_1 + w_{22}^o h_2 + w_{32}^o h_3 - \theta_2^o = (\mathbf{w}_2^o)^T \mathbf{h} \end{cases}$$



$$\text{ただし, } \mathbf{w}_j^o = \begin{pmatrix} w_{1j}^o \\ w_{2j}^o \\ w_{3j}^o \\ \theta_j^o \end{pmatrix}, \mathbf{h} = \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ -1 \end{pmatrix}$$

$$(j = 1, 2)$$



この値を求めたい(決めたい)！
人間がこの値を設定するのではなく、
機械に自ら設定させる(＝機械学習)

※ a は活性化関数 activation function

ニューラルネットワークと機械学習

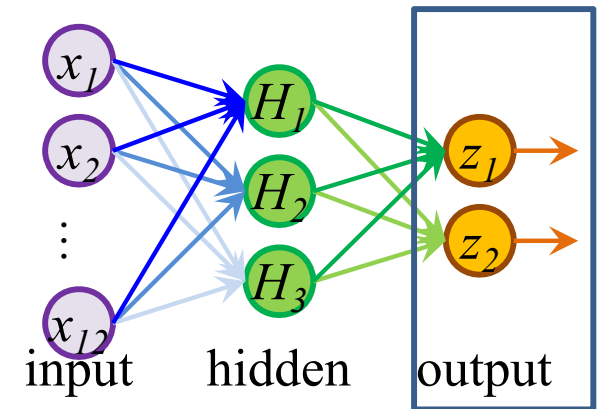
➤ *output layer* → *results*

- 例: 4 × 3画素の画像識別
 - z_1 ... 文字「C」に反応
 - z_2 ... 文字「T」に反応

$\begin{cases} z_1 = 1 \\ z_2 = 0 \end{cases}$ → 画像は「C」と結論する

$\begin{cases} z_1 = 0 \\ z_2 = 1 \end{cases}$ → 画像は「T」と結論する

$\begin{cases} z_1 = 0 \\ z_2 = 0 \end{cases}$ → 画像は「C」でも「T」でもない結論する



➤ 誤差 *error*

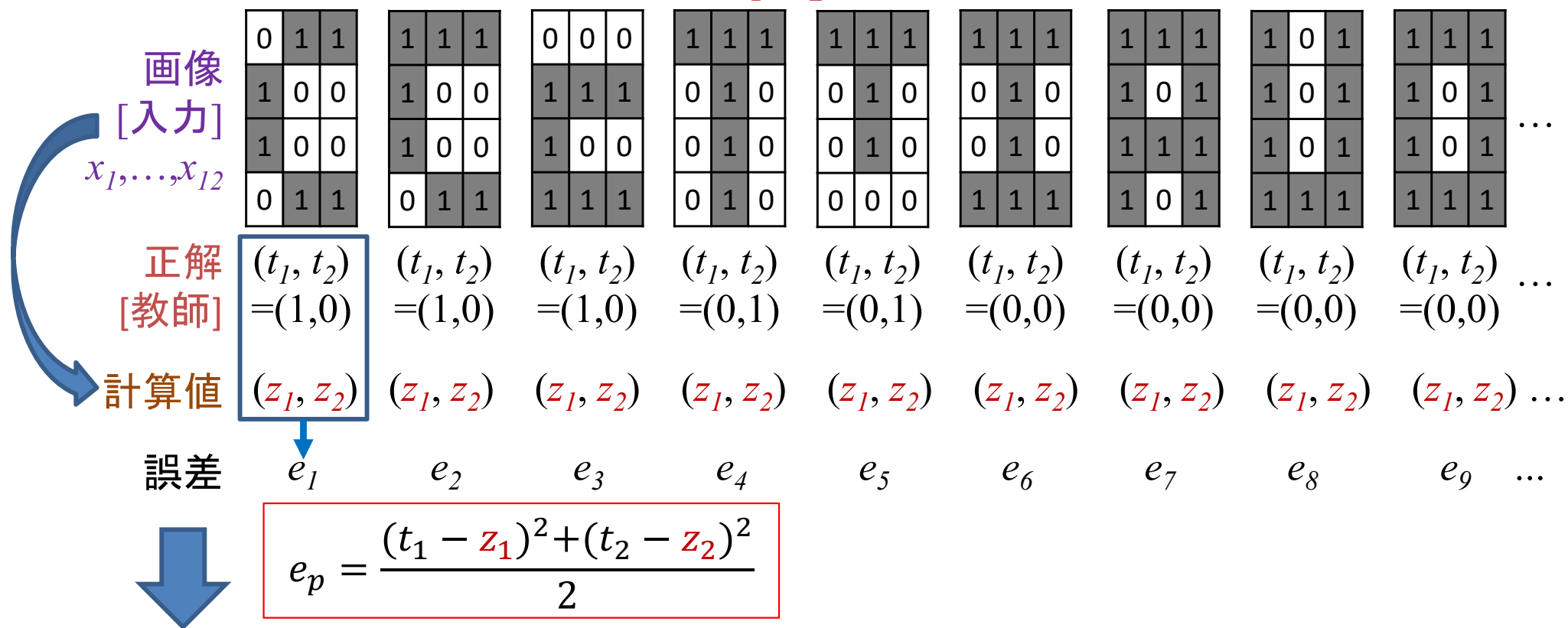
$$e = \frac{(t_1 - z_1)^2 + (t_2 - z_2)^2}{2}$$

この誤差が最小になるように、機械に学習させる

ニューラルネットワークと機械学習

➤ 誤差 *error* 評価で機械学習 (教師あり学習の場合)

– 画像[入力]と正解[教師] (t_1, t_2) の組を多数準備する



$$e_p = \frac{(t_1 - z_1)^2 + (t_2 - z_2)^2}{2}$$

誤差の総和 $E = e_1 + e_2 + e_3 + e_4 + e_5 + e_6 + e_7 + e_8 + e_9 \dots$

この誤差 (の総和) が最小になるように, ネットワークの各ユニットがもつ固有の値 [パラメータ] (w_i^H ($i = 1, 2, 3$), w_j^O ($j = 1, 2$)) を機械に学習・設定させる

機械学習（誤差の総和最小化）

➤ 誤差総和 E の最小化

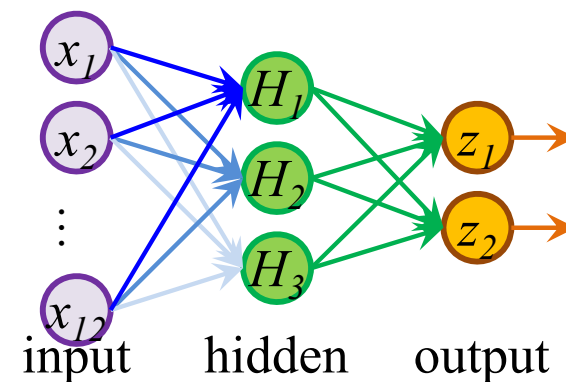
$$E = e_1 + e_2 + \dots$$

$$e_p = \frac{(t_1 - z_1)^2 + (t_2 - z_2)^2}{2}$$

$$\begin{cases} z_1 = a(s_1^o) \\ z_2 = a(s_2^o) \end{cases} \begin{cases} s_1^o = (\mathbf{w}_1^o)^T \mathbf{h} \\ s_2^o = (\mathbf{w}_2^o)^T \mathbf{h} \end{cases}$$

$$\text{ただし, } \mathbf{w}_j^o = \begin{pmatrix} w_{1j}^o \\ w_{2j}^o \\ w_{3j}^o \\ \theta_j^o \end{pmatrix}, \mathbf{h} = \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ -1 \end{pmatrix} \quad (j = 1, 2)$$

$$\begin{cases} h_1 = a(s_1^H) \\ h_2 = a(s_2^H) \\ h_3 = a(s_3^H) \end{cases} \begin{cases} s_1^H = (\mathbf{w}_1^H)^T \mathbf{x} \\ s_2^H = (\mathbf{w}_2^H)^T \mathbf{x} \\ s_3^H = (\mathbf{w}_3^H)^T \mathbf{x} \end{cases} \text{ただし, } \mathbf{w}_i^H = \begin{pmatrix} w_{1i}^H \\ w_{2i}^H \\ \vdots \\ w_{12,i}^H \\ \theta_i^H \end{pmatrix}, \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{12} \\ -1 \end{pmatrix} \quad (i = 1, 2, 3)$$



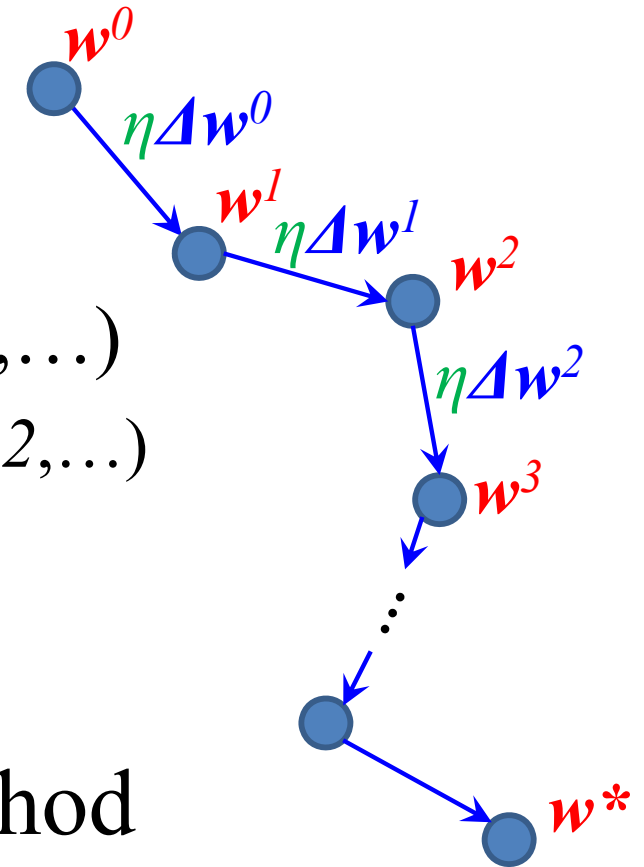
➡ E を最小とするパラメータ(変数) $\mathbf{w}_j^o$ ($j = 1, 2$), $\mathbf{w}_i^H$ ($i = 1, 2, 3$) の値を求めたい
 変数の数: (入力数+1) × (隠れ層数) + (隠れ層数+1) × (出力数) = $13 \times 3 + 4 \times 2 = 47$

➡ 解析的に解く(求める)のは困難なので, 反復数値計算(勾配法)で求める

機械学習（誤差の総和最小化）

➤ 反復数値計算（勾配法）

- 初期解 w^0 から最適解 w^* へ
- 解の更新 $w^{k+1} = w^k + \eta \Delta w^k$ ($k = 0, 1, 2, \dots$)
 - ✓ 反復 k 回目の降下方向 Δw^k ($k = 0, 1, 2, \dots$)
 - ✓ 反復サイズ (step size) η



➤ 最急降下法 steepest descent method

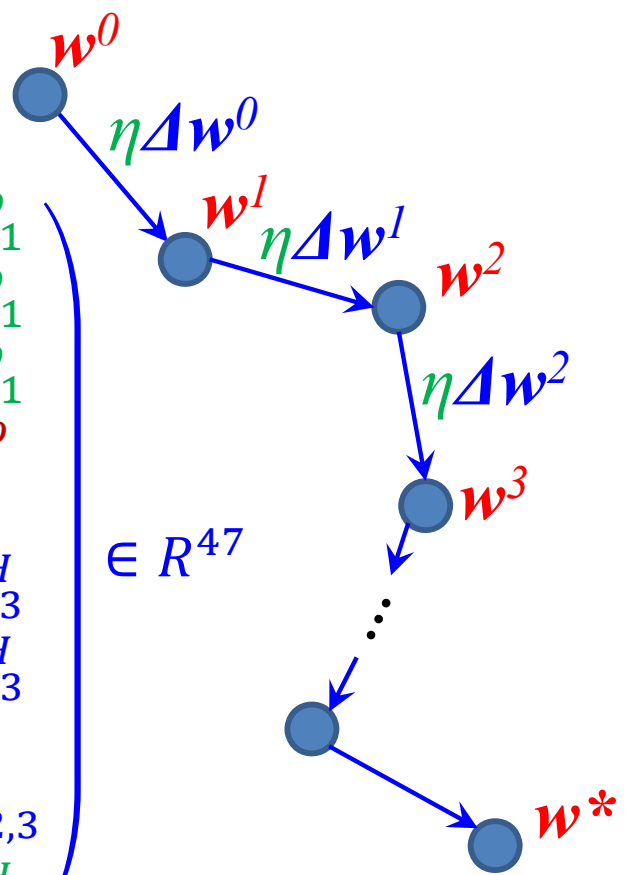
- 降下方向 Δw に最急降下方向を使う手法
- 関数 $f(w)$ の最急降下方向は $-\nabla f(w) = - \begin{pmatrix} \frac{\partial f}{\partial w_1} \\ \vdots \\ \frac{\partial f}{\partial w_n} \end{pmatrix}$

※最急降下法は勾配法的一种で、最も簡便な手法

※勾配法のstep sizeは直線探索 (Armijo method等) で求める

機械学習（誤差の総和最小化）

➤ 例題の解 $\mathbf{w}$ と降下方向 $\Delta \mathbf{w}$

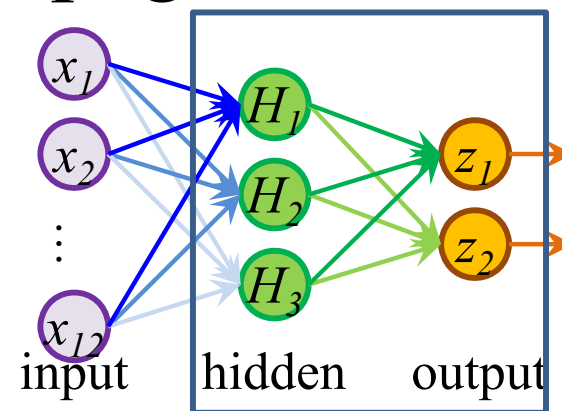
$$\mathbf{w} = \begin{pmatrix} \mathbf{w}_1^O \\ \mathbf{w}_2^O \\ \mathbf{w}_1^H \\ \mathbf{w}_2^H \\ \mathbf{w}_3^H \end{pmatrix} = \begin{pmatrix} w_{11}^O \\ w_{21}^O \\ w_{31}^O \\ \theta_1^O \\ \vdots \\ w_{13}^H \\ w_{23}^H \\ w_{12,3}^H \\ \vdots \\ \theta_3^H \end{pmatrix} \in R^{47}, \Delta \mathbf{w} = \begin{pmatrix} \Delta \mathbf{w}_1^O \\ \Delta \mathbf{w}_2^O \\ \Delta \mathbf{w}_1^H \\ \Delta \mathbf{w}_2^H \\ \Delta \mathbf{w}_3^H \end{pmatrix} = \begin{pmatrix} \Delta w_{11}^O \\ \Delta w_{21}^O \\ \Delta w_{31}^O \\ \Delta \theta_1^O \\ \vdots \\ \Delta w_{13}^H \\ \Delta w_{23}^H \\ \Delta w_{12,3}^H \\ \vdots \\ \Delta \theta_3^H \end{pmatrix} \in R^{47}$$


➤ 例題の誤差総和 E の最急降下方向

$$\Delta \mathbf{w} = -\nabla E(\mathbf{w}) = -\begin{pmatrix} \vdots \\ \frac{\partial E}{\partial w_l} \\ \vdots \end{pmatrix}, \quad \frac{\partial E}{\partial w_l} = \frac{\partial e_1}{\partial w_l} + \frac{\partial e_2}{\partial w_l} + \dots$$

機械学習（誤差逆伝播法backpropagation）

➤ 出力層の勾配[(重み3+閾値1)×2=8個]



$$e_p = \frac{(t_1 - z_1)^2 + (t_2 - z_2)^2}{2}$$

$$\begin{cases} z_1 = a(s_1^o) \\ z_2 = a(s_2^o) \end{cases}$$

$$\rightarrow \begin{cases} \frac{\partial e_p}{\partial z_1} = (z_1 - t_1) \\ \frac{\partial e_p}{\partial z_2} = (z_2 - t_2) \end{cases} \rightarrow \begin{cases} \frac{\partial z_1}{\partial s_1^o} = a'(s_1^o) \\ \frac{\partial z_2}{\partial s_2^o} = a'(s_2^o) \end{cases}$$

$$\begin{cases} s_1^o = w_{11}^o h_1 + w_{21}^o h_2 + w_{31}^o h_3 - \theta_1^o \\ s_2^o = w_{12}^o h_1 + w_{22}^o h_2 + w_{32}^o h_3 - \theta_2^o \end{cases}$$

$$\rightarrow \begin{cases} \frac{\partial s_1^o}{\partial w_{11}^o} = h_1, \frac{\partial s_1^o}{\partial w_{21}^o} = h_2, \frac{\partial s_1^o}{\partial w_{31}^o} = h_3, \frac{\partial s_1^o}{\partial \theta_1^o} = -1 \\ \frac{\partial s_2^o}{\partial w_{12}^o} = h_1, \frac{\partial s_2^o}{\partial w_{22}^o} = h_2, \frac{\partial s_2^o}{\partial w_{32}^o} = h_3, \frac{\partial s_2^o}{\partial \theta_2^o} = -1 \end{cases}$$

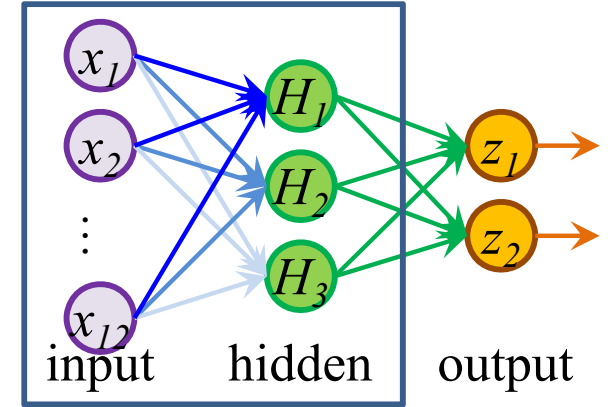
$$\begin{cases} \frac{\partial e_p}{\partial w_{11}^o} = \frac{\partial e_p}{\partial z_1} \frac{\partial z_1}{\partial s_1^o} \frac{\partial s_1^o}{\partial w_{11}^o} = (z_1 - t_1) a'(s_1^o) h_1 \\ \frac{\partial e_p}{\partial w_{21}^o} = \frac{\partial e_p}{\partial z_1} \frac{\partial z_1}{\partial s_1^o} \frac{\partial s_1^o}{\partial w_{21}^o} = (z_1 - t_1) a'(s_1^o) h_2 \\ \frac{\partial e_p}{\partial w_{31}^o} = \frac{\partial e_p}{\partial z_1} \frac{\partial z_1}{\partial s_1^o} \frac{\partial s_1^o}{\partial w_{31}^o} = (z_1 - t_1) a'(s_1^o) h_3 \\ \frac{\partial e_p}{\partial \theta_1^o} = \frac{\partial e_p}{\partial z_1} \frac{\partial z_1}{\partial s_1^o} \frac{\partial s_1^o}{\partial \theta_1^o} = -(z_1 - t_1) a'(s_1^o) \\ \frac{\partial e_p}{\partial w_{12}^o} = \frac{\partial e_p}{\partial z_2} \frac{\partial z_2}{\partial s_2^o} \frac{\partial s_2^o}{\partial w_{12}^o} = (z_2 - t_2) a'(s_2^o) h_1 \\ \frac{\partial e_p}{\partial w_{22}^o} = \frac{\partial e_p}{\partial z_2} \frac{\partial z_2}{\partial s_2^o} \frac{\partial s_2^o}{\partial w_{22}^o} = (z_2 - t_2) a'(s_2^o) h_2 \\ \frac{\partial e_p}{\partial w_{32}^o} = \frac{\partial e_p}{\partial z_2} \frac{\partial z_2}{\partial s_2^o} \frac{\partial s_2^o}{\partial w_{32}^o} = (z_2 - t_2) a'(s_2^o) h_3 \\ \frac{\partial e_p}{\partial \theta_2^o} = \frac{\partial e_p}{\partial z_2} \frac{\partial z_2}{\partial s_2^o} \frac{\partial s_2^o}{\partial \theta_2^o} = -(z_2 - t_2) a'(s_2^o) \end{cases}$$

機械学習（誤差逆伝播法backpropagation）

➤ 隠れ層の勾配[(重み12+閾値1)×3=39個]

✓ $i=1(H_1), 2(H_2), 3(H_3)$ の場合の各13個

$$\left\{ \begin{aligned} \frac{\partial e_p}{\partial w_{1i}^H} &= \left(\frac{\partial e_p}{\partial z_1} \frac{\partial z_1}{\partial s_1^o} \frac{\partial s_1^o}{\partial h_i} + \frac{\partial e_p}{\partial z_2} \frac{\partial z_2}{\partial s_2^o} \frac{\partial s_2^o}{\partial h_i} \right) \frac{\partial h_i}{\partial s_i^H} \frac{\partial s_i^H}{\partial w_{1i}^H} \\ &= ((z_1 - t_1)a'(s_1^o)w_{i1}^o + (z_2 - t_2)a'(s_2^o)w_{i2}^o)a'(s_i^H)x_1 \\ &\quad \vdots \\ \frac{\partial e_p}{\partial w_{12,i}^H} &= \left(\frac{\partial e_p}{\partial z_1} \frac{\partial z_1}{\partial s_1^o} \frac{\partial s_1^o}{\partial h_i} + \frac{\partial e_p}{\partial z_2} \frac{\partial z_2}{\partial s_2^o} \frac{\partial s_2^o}{\partial h_i} \right) \frac{\partial h_i}{\partial s_i^H} \frac{\partial s_i^H}{\partial w_{12,i}^H} \\ &= ((z_1 - t_1)a'(s_1^o)w_{i1}^o + (z_2 - t_2)a'(s_2^o)w_{i2}^o)a'(s_i^H)x_{12} \\ \frac{\partial e_p}{\partial \theta_i^H} &= \left(\frac{\partial e_p}{\partial z_1} \frac{\partial z_1}{\partial s_1^o} \frac{\partial s_1^o}{\partial h_i} + \frac{\partial e_p}{\partial z_2} \frac{\partial z_2}{\partial s_2^o} \frac{\partial s_2^o}{\partial h_i} \right) \frac{\partial h_i}{\partial s_i^H} \frac{\partial s_i^H}{\partial \theta_i^H} \\ &= -((z_1 - t_1)a'(s_1^o)w_{i1}^o + (z_2 - t_2)a'(s_2^o)w_{i2}^o)a'(s_i^H) \end{aligned} \right. \quad (i = 1, 2, 3)$$



$$\begin{cases} h_1 = a(s_1^H) \\ h_2 = a(s_2^H) \\ h_3 = a(s_3^H) \end{cases}$$

$$\begin{cases} s_1^H = w_{11}^H x_1 + \dots + w_{12,1}^H x_{12} - \theta_1^H \\ s_2^H = w_{12}^H x_1 + \dots + w_{12,2}^H x_{12} - \theta_2^H \\ s_3^H = w_{13}^H x_1 + \dots + w_{12,3}^H x_{12} - \theta_3^H \end{cases} \rightarrow \begin{cases} \frac{\partial s_1^H}{\partial w_{11}^H} = x_1, \dots, \frac{\partial s_1^H}{\partial w_{12,1}^H} = x_{12}, \frac{\partial s_1^H}{\partial \theta_1^H} = -1 \\ \frac{\partial s_2^H}{\partial w_{12}^H} = x_1, \dots, \frac{\partial s_2^H}{\partial w_{12,2}^H} = x_{12}, \frac{\partial s_2^H}{\partial \theta_2^H} = -1 \\ \frac{\partial s_3^H}{\partial w_{13}^H} = x_1, \dots, \frac{\partial s_3^H}{\partial w_{12,3}^H} = x_{12}, \frac{\partial s_3^H}{\partial \theta_3^H} = -1 \end{cases} \rightarrow \begin{cases} \frac{\partial h_1}{\partial s_1^H} = a'(s_1^H) \\ \frac{\partial h_2}{\partial s_2^H} = a'(s_2^H) \\ \frac{\partial h_3}{\partial s_3^H} = a'(s_3^H) \end{cases}$$

機械学習（誤差逆伝播法backpropagation）

▶ 活性化関数 $a(s)$ の微分 $a'(s)$

- シグモイド関数 sigmoid function

$$\sigma(s) = \frac{1}{1 + e^{-s}}$$

$$\begin{aligned}\frac{\partial \sigma}{\partial s} &= -\frac{-e^{-s}}{(1 + e^{-s})^2} = \frac{1 + e^{-s} - 1}{(1 + e^{-s})^2} \\ &= \frac{1}{1 + e^{-s}} \cdot \left(1 - \frac{1}{1 + e^{-s}}\right) \\ &= \sigma(s)\{1 - \sigma(s)\}\end{aligned}$$

- 双曲線正接関数 hyperbolic tangent function

$$\tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}}$$

$$\begin{aligned}\frac{\partial t}{\partial s} &= \frac{(e^s + e^{-s})^2 - (e^s - e^{-s})^2}{(e^s + e^{-s})^2} \\ &= 1 - \tanh(s)^2\end{aligned}$$

- ランプ関数 ReLU ramp function

$$\text{ReLU}(s) = \begin{cases} s & (s \geq 0) \\ 0 & (s < 0) \end{cases}$$

$$\frac{\partial R}{\partial s} = \begin{cases} 1 & (s > 0) \\ 0 & (s < 0) \end{cases}$$

- etc.

※ $s=0$ で微分不可能 indifferentiable

※凸関数より $s=0$ で劣微分可能 subdifferentiable

機械学習（教師あり学習）

➤ 機械学習のアルゴリズム algorithm

1. 入力と正解の組を充分数準備. 初期解 w^0 を生成
(=各ユニット固有の値の初期値を設定). $k = 0$
2. 入力 $(x_1, \dots, x_{l_2})$ から, 各ユニット固有の値 w^k で出力
 (z_1, z_2) を計算 (※準備した全てについて計算)
3. 正解 (t_1, t_2) と出力 (z_1, z_2) から誤差 E を計算 (※準備した全てについて計算し, 誤差 e_p の和 E を計算)
4. 誤差 E が十分小さい ($E < \varepsilon$) なら終了. そうでなければ, 誤差逆伝播法で勾配 Δw^k を計算
5. 勾配法 (最急降下法) で解を更新 $w^{k+1} = w^k + \eta \Delta w^k$.
 $k = k+1$ として step 2へ

機械学習（教師あり学習）

※[2] 斎藤康毅「ゼロから作るDeep Learning」オンライン
リージャパン(2016) 6章 学習に関するテクニック

➤ 補足

✓ ネットワークの構成

- 層の数をどうするか？ / 各層のノード数をどうするか？

✓ 初期解 w^0 の生成（重要）

- Xavierの初期値 (sigmoid関数/tanh関数)
- Heの初期値 (ReLU関数)

✓ 勾配法の選択 と step size の決め方

- 最急降下法 steepest descent method
- Momentum
- AdaGrad; adaptive gradient
- Adam

✓ 過学習 overfitting への対処

- Weight Decay
- Dropout

参考文献

0	1	1
1	0	0
1	0	0
0	1	1

- [1] 涌井良幸・涌井貞美「Excelでわかる機械学習超入門」
技術評論社(2019)
- [2] 斎藤康毅「ゼロから作るDeep Learning」
オライリージャパン(2016)