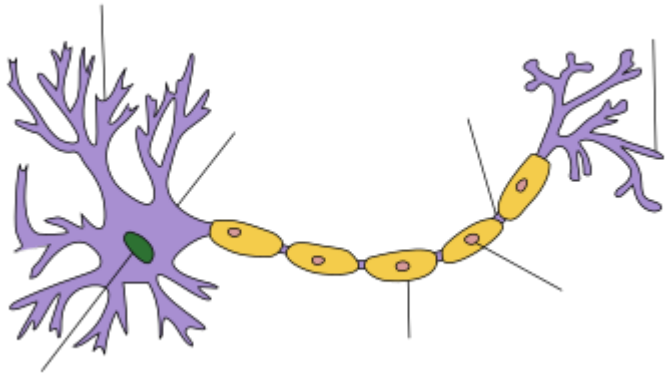
科学エキスパート講座

岡野浩三

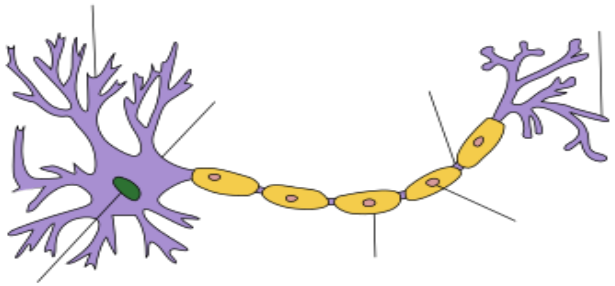
深層学習

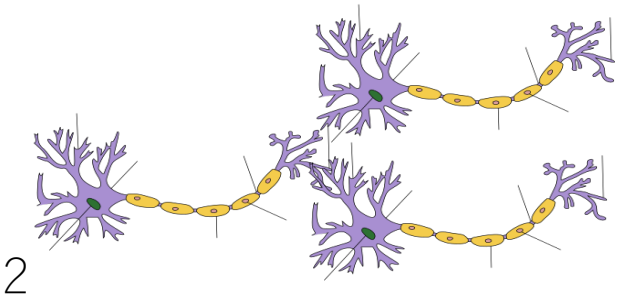
神経細胞

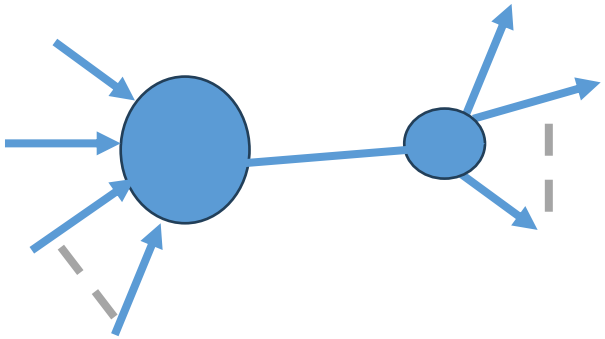


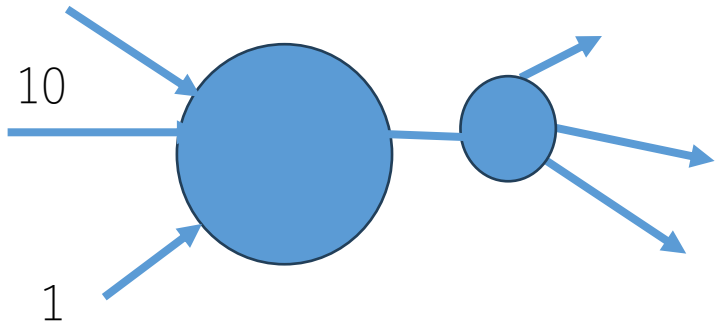
From Wikipedia

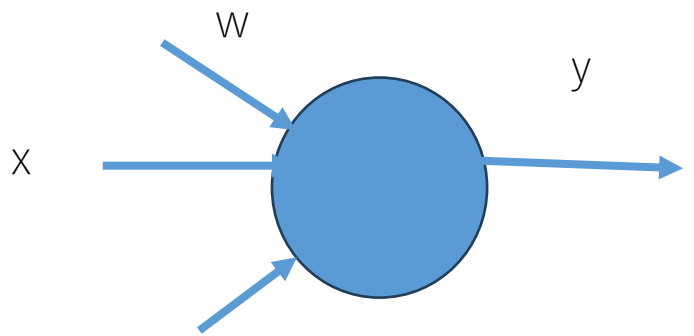
- 神経細胞は死なない（一生そのまま変わらない）
- 非再生系組織の細胞
- 細胞の結合が変わる
- 脳の学習とは神経細胞のつながり方を変えていく作業











- 脳の学習とは神経細胞のつながり方を変えていく作業
  - 枝の重みを変ええ行く作業
  - 重み　大きい：つながりが強い
  - 　　小さい：つながりがよわい

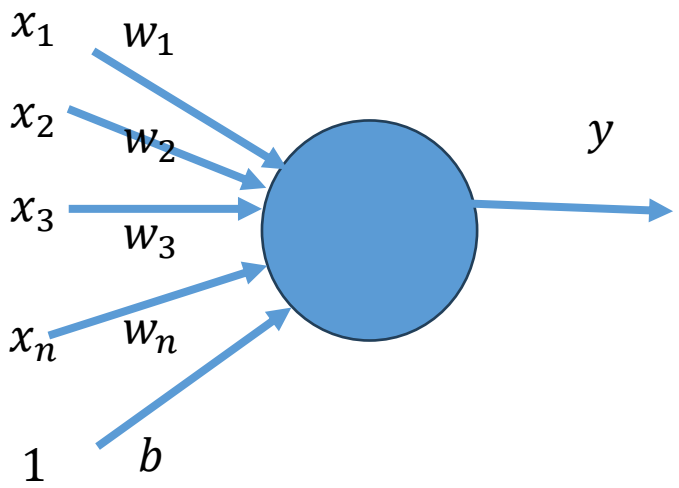
ニューロンの計算

- 信号の伝達の式は
  $a = b + w_1x_1 + w_2x_2 + \cdots$
  のようになる.
- 活性化関数
  入力信号の総和がどのように
  活性化するかを決定する役割がある.

$$y = h(a)$$

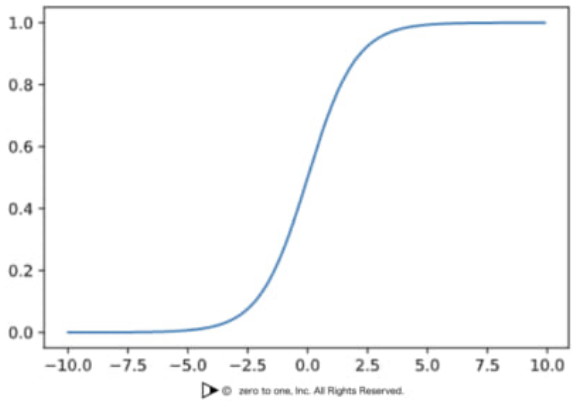
- シグモイド関数

$$h(x) = \frac{1}{1 + \exp(-x)}$$



シグモイド関数

$$f(x) = \frac{1}{1 + e^{-ax}} \quad (a > 0)$$



ニューロンによる論理関数の表現

- 2 入力and
- $y = h(-1.5 + x_1 + x_2)$　両方1でないと≈ 1にならない

- 2 入力or
- $y = h(-0.2 + x_1 + x_2)$片方1になると≈ 1になる

- not
- $y = h(-x - 0.2)$
- xorは 1つのニューロンでは表現できない
- しかし複数のニューロンを組み合わせると表現できる

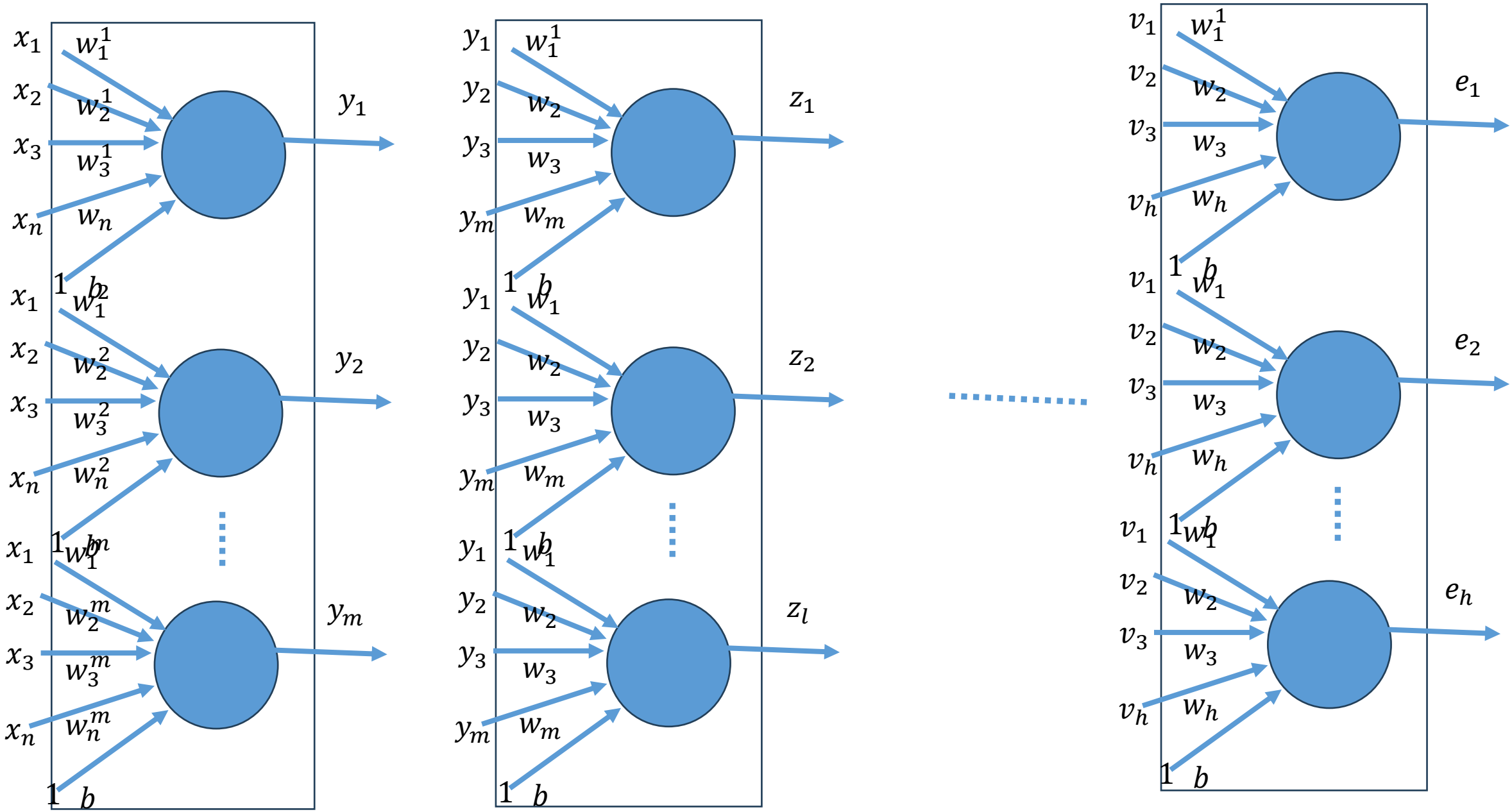
ニューロンによる論理関数の表現

- 任意の組み合わせ回路はand,or,notの組み合わせ（関数合成）で表現可能

- ニューロンの合成で「任意の組み合わせ回路」が実現できる！

- ニューラルネットワークの研究は「重みの学習」に進んでいく
  - ニューロンが人間の脳の神経細胞の「模倣」であるならば
  - 人間の学習も模倣できるはず？

DeepNN


x_1 x_2 x_3 x_n 1
w_1^1 w_2^1 w_3^1 w_n b
y_1
w_1^2 w_2^2 w_3^2 w_n^2 b
y_2
w_1^m w_2^m w_3^m w_n^m b
y_m
y_1 y_2 y_3 y_m 1
w_1^1 w_2 w_3 w_m b
z_1
w_1 w_2 w_3 w_m b
z_2
w_1 w_2 w_3 w_m b
z_l
v_1 v_2 v_3 v_h 1
w_1^1 w_2 w_3 w_h b
e_1
w_1 w_2 w_3 w_h b
e_2
w_1 w_2 w_3 w_h b
e_h

3次元以上の変数の集まり

- ${}^{k}w_{j}^{i}$

- k層目のi番目のニューロンのj番目の信号の重み
- ３次元の配列　テンソル
  - 0次元　スカラー
  - 1次元　ベクトル
  - 2次元　行列
- Cf. Tensor Flow

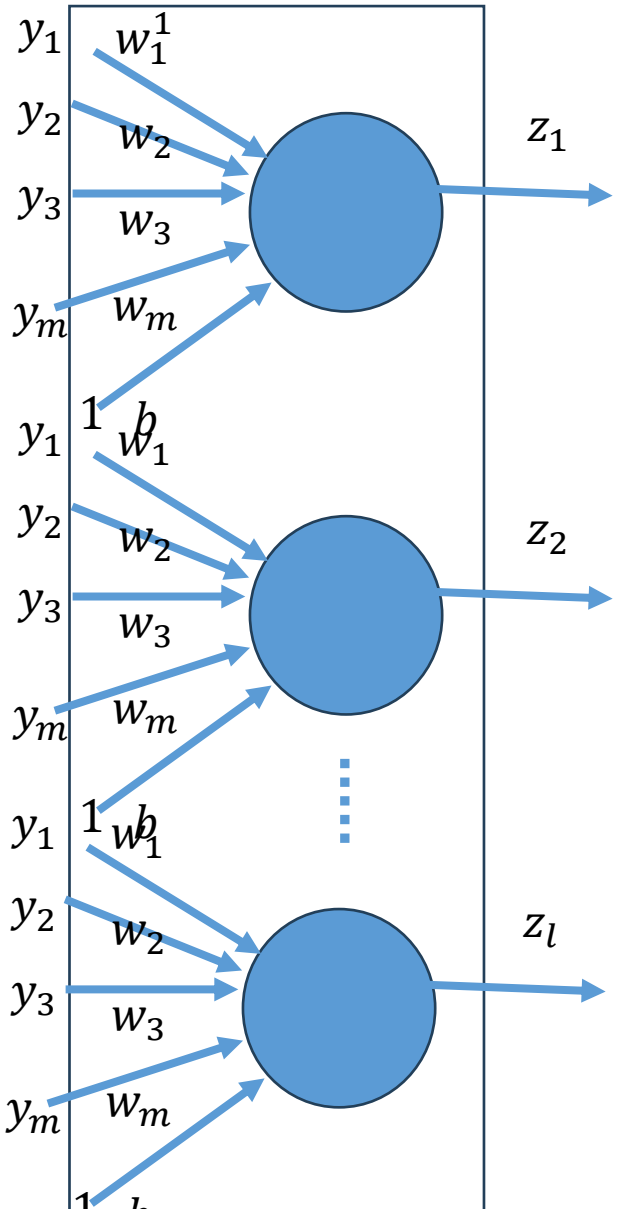
学習とは

- x（刺激）、y（理想的な反応）を大量に与えて
- Wを漸進的に求めていく（確定していく）作業

- 学習後　新たなxを与えると　（ふさわしい）yが出力される

- 人間の脳の学習過程に類似
  - ＧＰＵの発達と
  - 学習アルゴリズム（逆伝播アルゴリズム）の発明

逆伝搬アルゴリズム



$$\boldsymbol{z = F(y; w)}$$

$$\boldsymbol{E(F(y^t; w) - z^t)}$$

$\boldsymbol{z^t}$: $\boldsymbol{y^t}$に対する正しい値

$\boldsymbol{E}$ (): 誤差関数

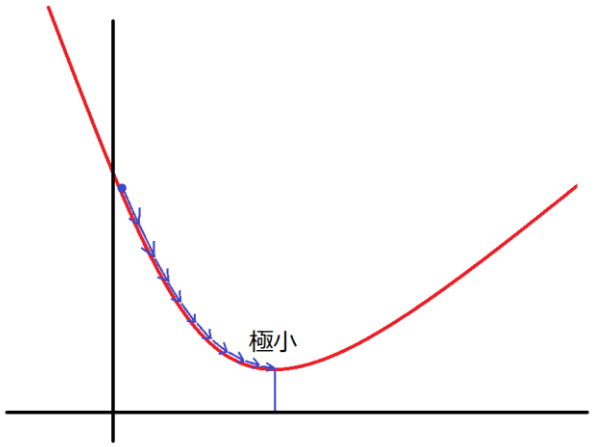
$\boldsymbol{E}$を最小にする$\boldsymbol{w}$を求めたい

大量の$\boldsymbol{z^t}$ と $\boldsymbol{y^t}$ の組み合わせから$\boldsymbol{w}$を求めていく
この計算を後ろの層から前段の層に向けて少しづつ$\boldsymbol{w}$を変化させながら計算してく

SGD

- 一般的に関数の最小値の候補（極値）を見つけるには現時点の傾きをしらべその方向に進めばよい
- 傾き：偏微分
- 確率的に少しずつ進めていく
- 以上の考え方が確率的勾配降下法(Stochastic Gradient Descent)

$$w^{t+1} = w^t - \varepsilon \frac{dE}{dw}$$

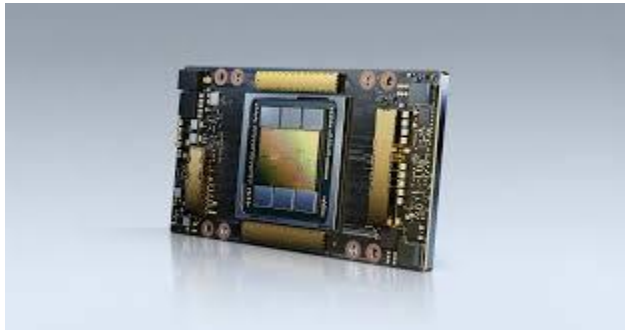


GPU

- 本来はディスプレイする画面を描画するデバイス
- 描画用メモリと描画用専用計算ユニットからなる
  - メモリの並びがディスプレイのドットに対応し、メモリの値がドット色情報をあたえる
- 最近では３DCGを高速に表示する必要性から
  - 大量の計算ユニットを並列に同時に動かす
  - 大量のメモリ空間を計算ユニットが共有する

GPUの機械学習への転用

- 大量のメモリ空間を計算ユニットが共有する
- ニューラルネットワークの計算にピッタリ
  - 大量のニューロンの計算を高速に並列実行
  - 重みやニューロンの結線情報はメモリに保存
- 最近のグラフィックカードの発展は機械学習の実行の容易さに直結している
- グラフィックカード会社は機械学習専用のカードも発売している（大量高速メモリと大量の並列計算ユニッ



初期のDNNと応用例

- 初期の深層NN（DNN）は例えば画像分類（画像文字認識）に応用できた
- 入力は画像の画素情報（28 x 28）256の値
- 出力はone-hot-vector （例えば10の要素を持つベクトル）
- 3の画像の正解出力は(0,0,0,1,0,0,0,0,0,0) 3のみが1その他はすべて0のベクトル値
- 実際のDNNの出力はアナログで他の要素も非0になる
- (0,0,0.2,0.7,0,0.1,0,0,0,0)
- 2が0.2の確からしさ、3が0.7の確からしさ、5が0.1の確からしさ
- 最終段はsoftmax関数

Softmax関数

$$\sigma(\boldsymbol{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad \text{for } i = 1, \ldots, K \text{ and } \mathbf{z} = (z_1, \ldots, z_K) \in \mathbb{R}^K$$

確率と解釈できる

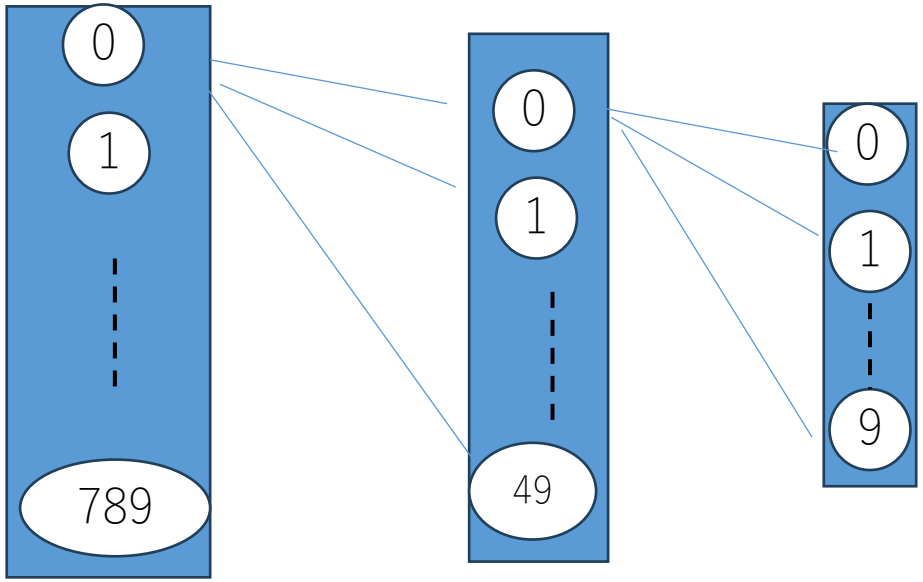
MNIST

- **MNISTデータベース**（Modified National Institute of Standards and Technology database）
- さまざまな画像処理システムの学習に広く使用される手書き数字画像の大規模なデータベース
- 機械学習分野での学習や評価に広く用いられている
  - 60,000枚の訓練用画像
  - 10,000枚の評価用画像

MNISTを判別する簡単なFCNN

- ２層構造
- 1層目　入力　786= 28 * 28 + 2
- 2層目　入力　50
- 出力　10　one-shot

一次元配列にする



学習の実際

- ミニバッチ
  - ひとつづつ学習するのではなく
  - 一度にでも少しずつ、確率的に
- 初期値
  - 初期値は０からスタート
  - とはいえ、０では逆伝搬がうまくいかない（消失問題）
  - ０に近いが０ではない値を確率的に
- エポック
  - 同じ学習過程を何度か繰り返す
  - １通り終わったところを１エポックとみなす

ミニバッチ学習

- **ミニバッチ学習**では、N個の訓練データの中から、n個を取り出して、パラメータ更新を実施
- ニューラルネットワークの学習でよく用いられる
- 取り出した訓練データをミニバッチと呼ぶ
- 取り出すデータ数nをミニバッチサイズと呼ぶ

エポック数

- エポック数：「訓練データを何回繰り返して学習させるか」の数

- 10,000個の訓練データがある場合、10,000個を１回ずつ学習させた場合が1エポック
- 100エポック ＝　各訓練データを100回ずつ学習