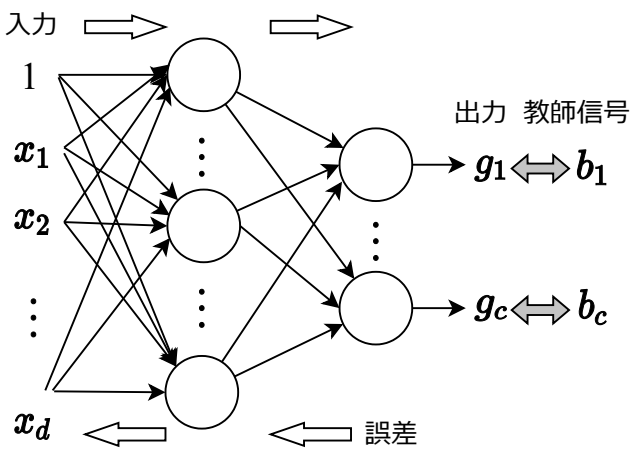
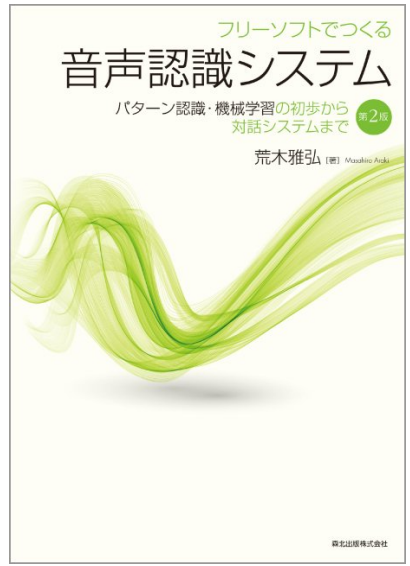
7. 限界は破れるか(2) – ニューラルネットワーク –



- 7.1 ニューラルネットワークの構成
- 7.2 誤差逆伝播法による学習
- 7.3 ディープニューラルネットワーク



- 荒木雅弘:『フリーソフトでつくる音声認識システム(第2版)』(森北出版, 2017年)
- スライドとJupyter notebook
- サポートページ

7.1 ニューラルネットワークの構成（1/7）

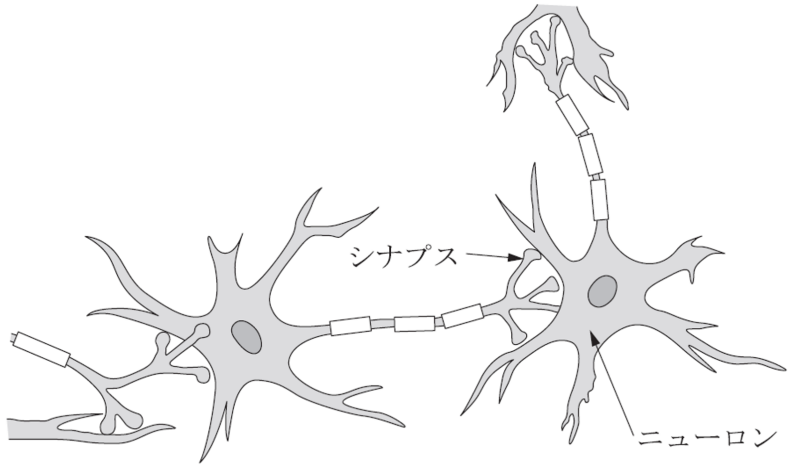
- 誤差評価に基づく学習の限界
  - 複雑な境界となるデータの学習においては、線形識別面では性能向上に限界がある
  - 誤差評価に基づく方法で非線形識別面の学習は可能だが、どのような非線形関数にするかを事前に設計する必要がある
- 誤差の小さい任意形の識別面を学習することはできないか

  → ニューラルネットワーク
- ネットワークの構造を工夫して、特徴抽出も学習の対象とすることはできないか

  → ディープニューラルネットワーク

7.1 ニューラルネットワークの構成（2/7）

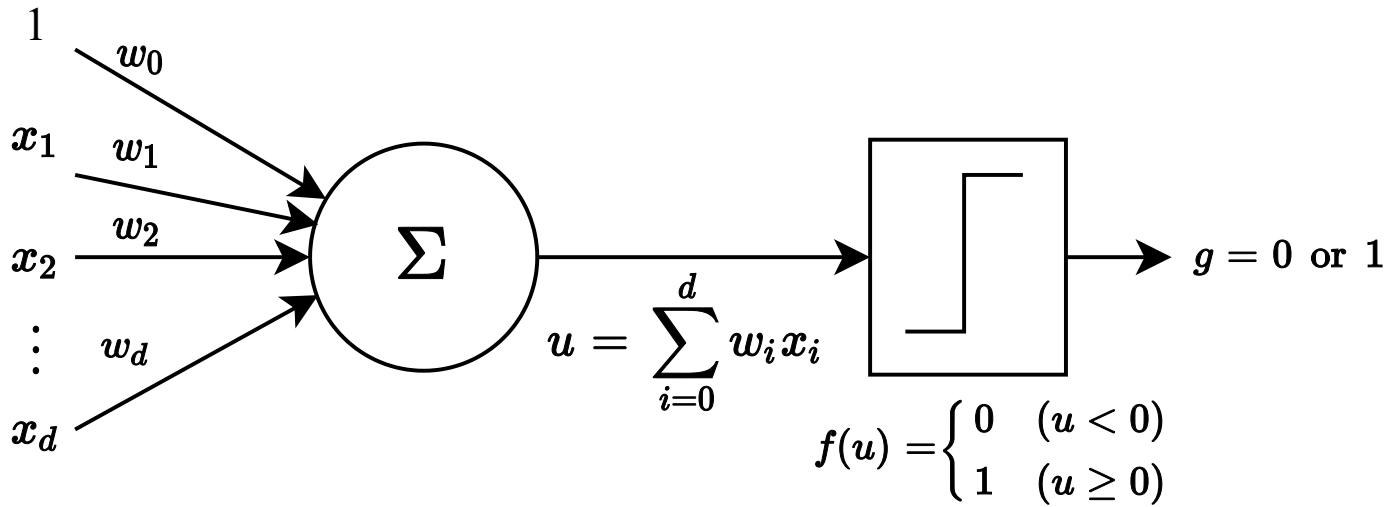
- ニューラルネットワークの基本的なアイディア
  - ニューロンとよばれる神経細胞がシナプス結合を介して信号を伝達する神経細胞の計算メカニズムをモデル化
  - 入力された電気信号の重み付き和の値によって、各ニューロンの活性／非活性が決まる



7.1 ニューラルネットワークの構成（3/7）

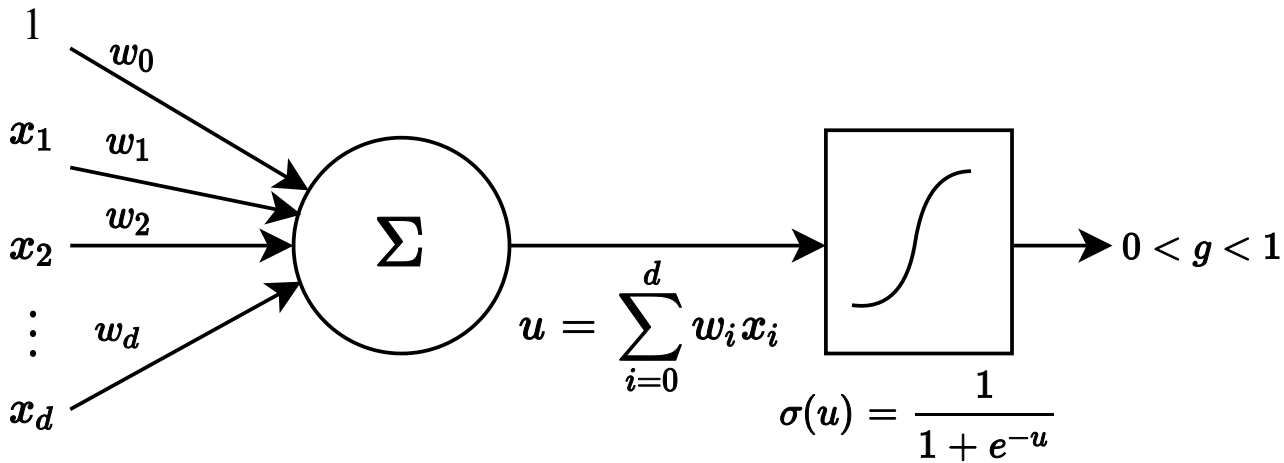
- 閾値論理ユニットによるニューロンのモデル化
  - $\boldsymbol{w}^T\boldsymbol{x} = 0$ という特徴空間上の識別面を表現（$\boldsymbol{w}$, $\boldsymbol{x}$ は $d+1$ 次元）

    → パーセプトロン（データが線形分離可能なときのみ学習可能）



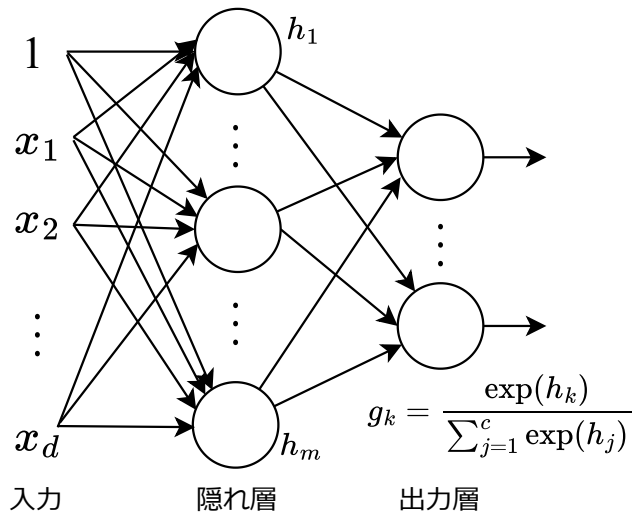
7.1 ニューラルネットワークの構成（4/7）

- 活性化関数を閾値関数からシグモイド関数に差し替え
  - 入力の重み付き和を0～1の値に変換
    - 入力が正例である確率とみなせる
  - データが線形分離不可能でも勾配降下法により誤差最小の線形識別面が学習可能



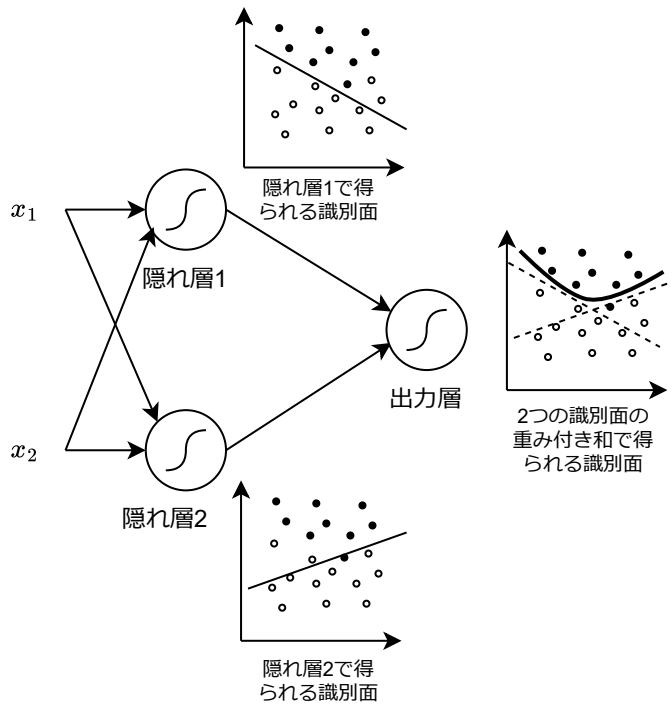
7.1 ニューラルネットワークの構成（5/7）

- フィードフォワード型ニューラルネットワーク
  - 入力 $\rightarrow$ 隠れ層 $\rightarrow$ 出力層とユニットを階層状に結合することで非線形識別面を実現
  - 多クラス識別の出力層には活性化関数として softmax 関数を用いる
    - 隠れ層の出力の重み付き和を、大小の順序を変えずに0～1、かつすべて足して1に変換



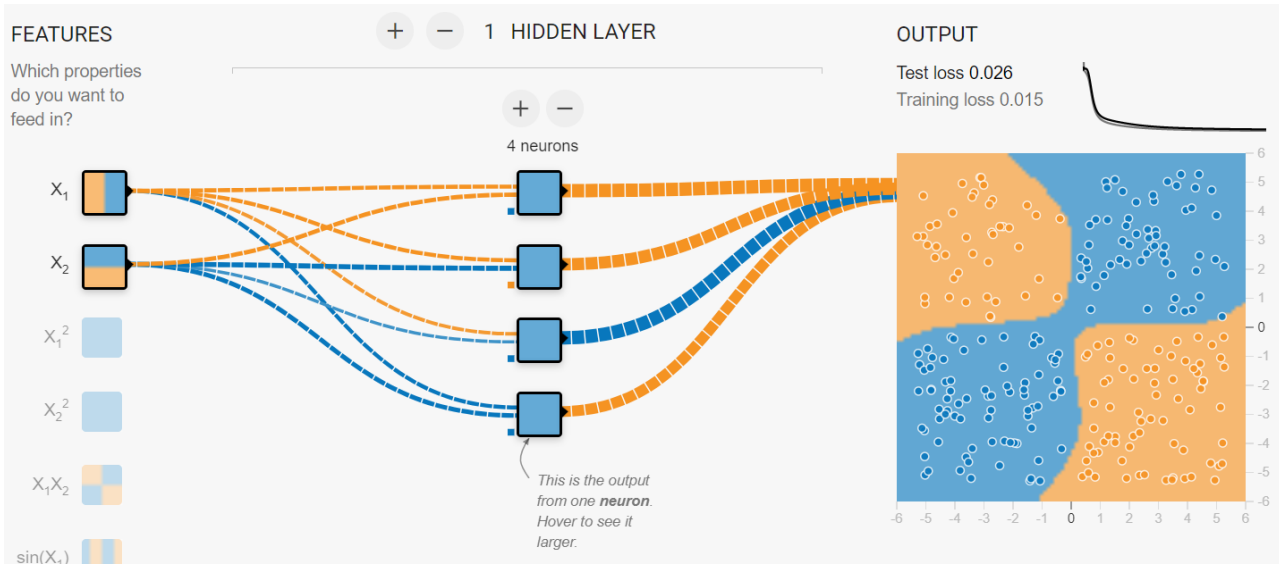
7.1 ニューラルネットワークの構成（6/7）

- フィードフォワード型ニューラルネットワークによる非線形識別面の実現
  - 線形識別関数の重み付き和で非線形識別関数を近似



7.1 ニューラルネットワークの構成（7/7）

- ニューラルネットワークによる非線形識別面の実現
  - 隠れ層のユニット数を増やすと、より複雑な非線形識別面が実現可能

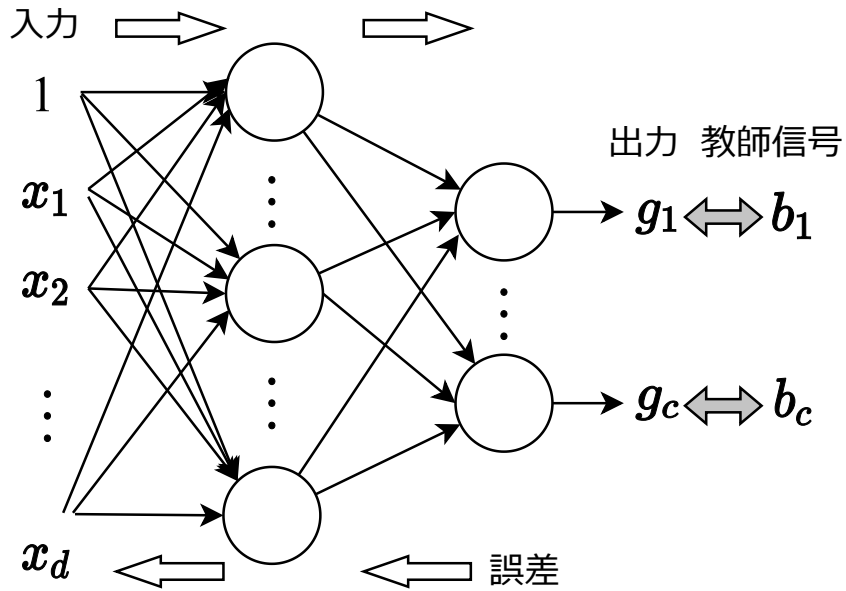


  - デモ：https://playground.tensorflow.org/
    - HIDDEM LAYER:1, neurons:4, Activation:Sigmoid, DATA:Exclusive or, Noise:0

7.2 誤差逆伝播法による学習

7.2.1 誤差逆伝播法の名前の由来

- 出力と教師信号の差から計算された誤差が、出力層から入力側へ逆方向に伝わる



7.2.2 結合重みの調整アルゴリズム (1/4)

- 特定のデータ $\boldsymbol{x}_p$ に対する二乗誤差

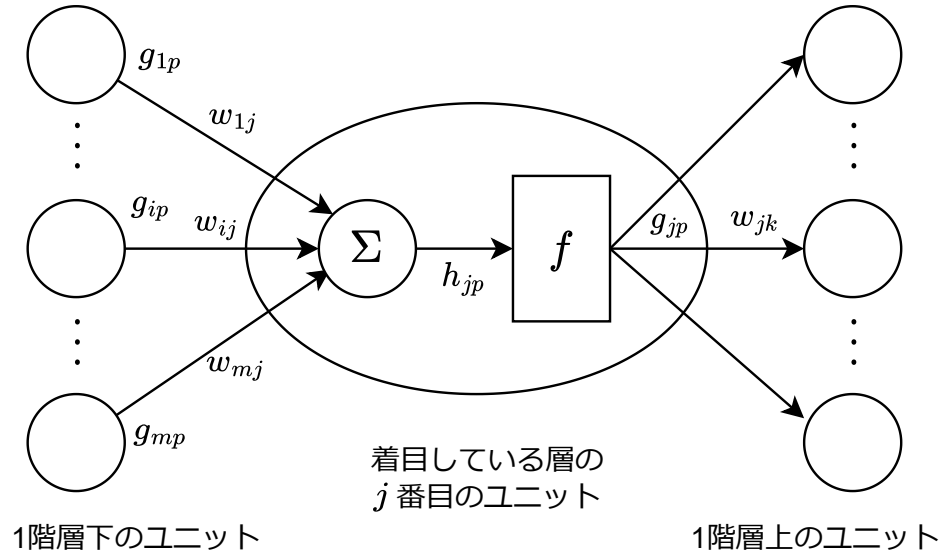
$$J(\boldsymbol{w}) \equiv \frac{1}{2}\sum_{i=1}^{c}(g_i(\boldsymbol{x}_p) - b_i)^2$$

- 勾配降下法による誤差 J の最小化
  - $\boldsymbol{w}$ を J の勾配の逆方向へ一定量だけ動かすことを繰り返して最適解へ収束させる
    - ρ：学習係数

$$\boldsymbol{w}' \leftarrow \boldsymbol{w} - \rho\frac{\partial J(\boldsymbol{w})}{\partial \boldsymbol{w}}$$

7.2.2 結合重みの調整アルゴリズム（2/4）

- 学習対象の重み
  - ある階層の j 番目のユニットと1階層下の i 番目のユニットとの重み w_{ij} の調整を考える



7.2.2 結合重みの調整アルゴリズム（3/4）

- ユニット j の内部の計算
  - 学習データ $\boldsymbol{x}_p$ が入力されたときのユニット j への入力の重み付き和
    - w_{ij}：ユニット i からユニット j への結合重み、g_{ip}：ユニット i の出力

$$h_{jp} = \sum_i w_{ij} g_{ip}$$

  - ユニット j の出力
    - f：活性化関数

$$g_{jp} = f(h_{jp})$$

7.2.2 結合重みの調整アルゴリズム (4/4)

- 学習データ $\boldsymbol{x}_p$ が入力されたときの出力層における誤差の定義
  - g_{lp}：出力層の l 番目のユニットの出力、b_{lp}：クラス l の教師信号

$$J_p \equiv \frac{1}{2}\sum_l (g_{lp} - b_{lp})^2$$

- 重み w_{ij} の調整式

$$w'_{ij} \leftarrow w_{ij} - \rho \frac{\partial J_p}{\partial w_{ij}}$$

7.2.3 調整量を求める (1/4)

- 調整量の計算
  - 合成関数の微分の公式を使う

$$\frac{\partial J_p}{\partial w_{ij}} = \frac{\partial J_p}{\partial h_{jp}} \frac{\partial h_{jp}}{\partial w_{ij}}$$

  - 右辺第2項は $\frac{\partial h_{jp}}{\partial w_{ij}} = \frac{\partial \sum_i w_{ij} g_{ip}}{\partial w_{ij}} = g_{ip}$
  - 右辺第1項を ε_{jp} とおいて、合成関数の微分の公式を適用

$$\varepsilon_{jp} = \frac{\partial J_p}{\partial h_{jp}} = \frac{\partial J_p}{\partial g_{jp}} \cdot \frac{\partial g_{jp}}{\partial h_{jp}} = \frac{\partial J_p}{\partial g_{jp}} \cdot f'(h_{jp})$$

7.2.3 調整量を求める (2/4)

- $\frac{\partial J_p}{\partial g_{jp}}$ の計算
  - ユニット j が出力層の場合

$$\frac{\partial J_p}{\partial g_{jp}} = g_{jp} - b_{jp}$$

  - ユニット j が中間層の場合
    - 誤差 J_p への影響は、1階層上のユニット k への影響の和

$$\frac{\partial J_p}{\partial g_{jp}} = \sum_k \frac{\partial J_p}{\partial h_{kp}} \cdot \frac{\partial h_{kp}}{\partial g_{jp}} = \sum_k \varepsilon_{kp} w_{jk}$$

- 活性化関数の微分
  - シグモイド関数の場合: $f'(h_{jp}) = f(h_{jp})(1 - f(h_{jp})) = g_{jp}(1 - g_{jp})$

7.2.3 調整量を求める (3/4)

- ここまでの計算のまとめ
  - 誤差の変化量

$$\varepsilon_{jp} = \begin{cases} (g_{jp} - b_{jp}) g_{jp} (1 - g_{jp}) & \text{出力層} \\ (\sum_k \varepsilon_{kp} w_{jk}) g_{jp} (1 - g_{jp}) & \text{隠れ層} \end{cases}$$

  - 重みの修正式

$$w'_{ij} \leftarrow \begin{cases} w_{ij} - \rho (g_{jp} - b_{jp}) g_{jp} (1 - g_{jp}) g_{ip} & \text{出力層} \\ w_{ij} - \rho (\sum_k \varepsilon_{kp} w_{jk}) g_{jp} (1 - g_{jp}) g_{ip} & \text{隠れ層} \end{cases}$$

7.2.3 調整量を求める (4/4)

- 誤差逆伝播法(確率的勾配降下法)
  1. ネットワークの重みを小さな初期値に設定
  2. 全データに対する学習を1エポックとし、エポック数だけ以下繰り返し
     1. データの順番をシャッフル
     2. 個々のデータ $\boldsymbol{x}_p$ に対して
        1. ネットワークの出力を計算
        2. 出力層から順に誤差項 ε の計算

$$\varepsilon_{jp} = \begin{cases} (g_{jp} - b_{jp}) g_{jp}(1 - g_{jp}) & \text{出力層} \\ (\sum_k \varepsilon_{kp} w_{jk}) g_{jp}(1 - g_{jp}) & \text{隠れ層} \end{cases}$$

        3. 重みの更新

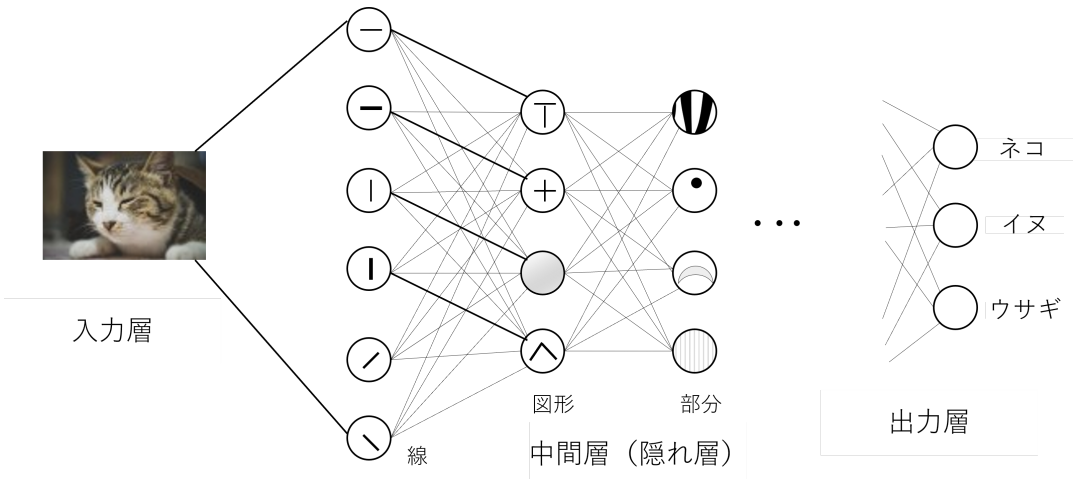
$$w'_{ij} \leftarrow \begin{cases} w_{ij} - \rho(g_{jp} - b_{jp}) g_{jp}(1 - g_{jp}) g_{ip} & \text{出力層} \\ w_{ij} - \rho(\sum_k \varepsilon_{kp} w_{jk}) g_{jp}(1 - g_{jp}) g_{ip} & \text{隠れ層} \end{cases}$$

7.2.4 過学習に気をつけよう

- ニューラルネットワークの学習の特徴
  - 過学習
    - ニューラルネットワークは非線形識別面を学習することができるので、学習データに対する誤識別率を限りなく 0 に近づけることができる
    - そのような識別面は、未知データに対して誤識別率が高いことが多い
    - 学習データに特化しすぎた識別面が学習される現象を**過学習**とよぶ
  - 学習の不安定性
    - 一般に誤差関数が複雑な形となり、極小値が複数存在して、極小値 $\neq$ 最小値となることが多い
    - どのような極小値で学習が停まるかは初期値に依存するので、初期値によっては学習がうまくいかないことがある

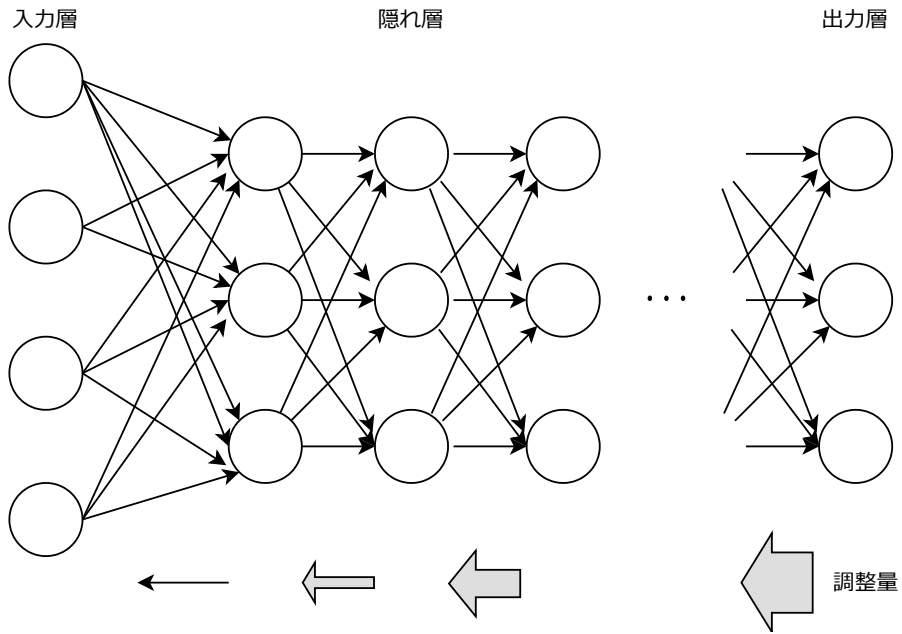
7.3 ディープニューラルネットワーク

- ディープニューラルネットワーク
  - ニューラルネットワークの隠れ層を多層にすることで性能の向上を目指したもの
  - 多階層にすることで、特徴抽出の処理も学習の対象とすることができる
  - 問題に特化したネットワーク構造を導入することもできる



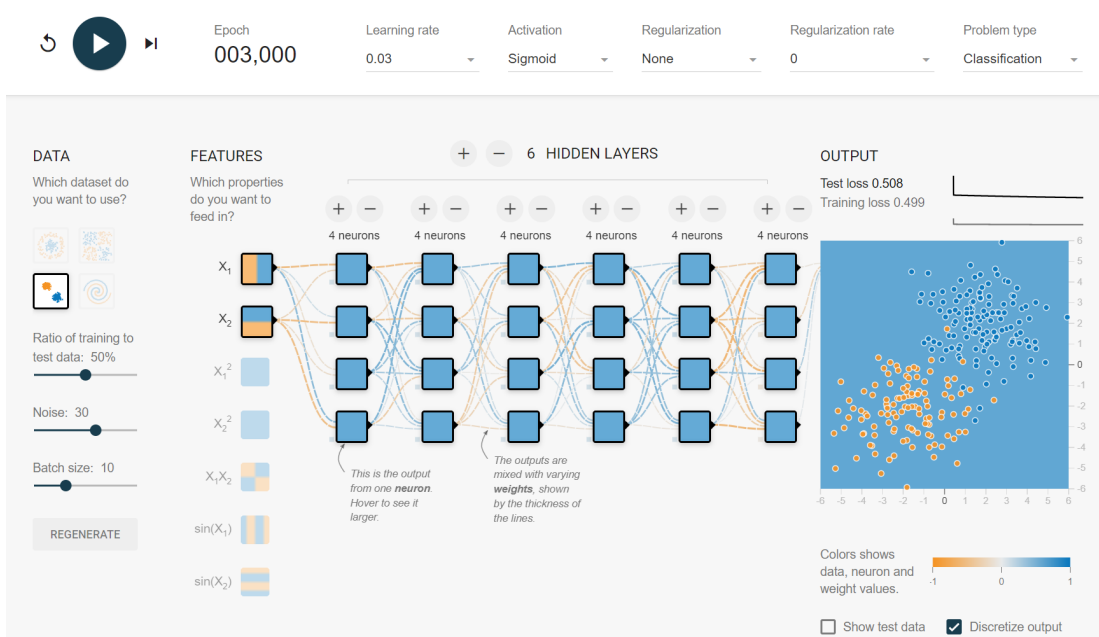
7.3.1 勾配消失問題とは

- 多階層における誤差逆伝播法の問題点
  - 入力層に近づくにつれて調整量が小さくなる**勾配消失**が生じる



7.3.1 勾配消失問題とは

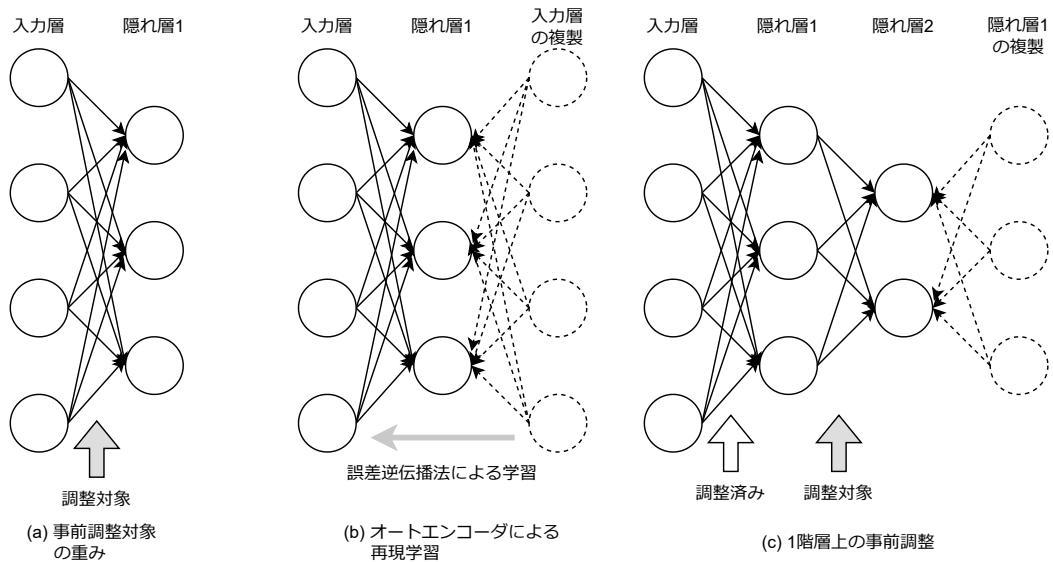
- 勾配消失のシミュレーション



  - デモ: https://playground.tensorflow.org/
    - HIDDEM LAYER:6, neurons:4, Activation:Sigmoid, DATA:Gaussian, Noise:30

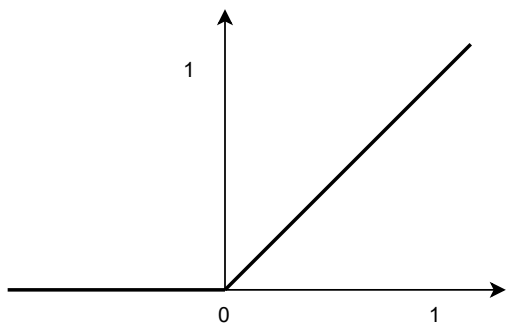
7.3.2 多階層学習における工夫

- 事前学習法
  - 深層学習の初期ではネットワーク全体の学習に先立って、**オートエンコーダ**（誤差逆伝播法による自己の情報の再現）による重みの初期値設定が行われた

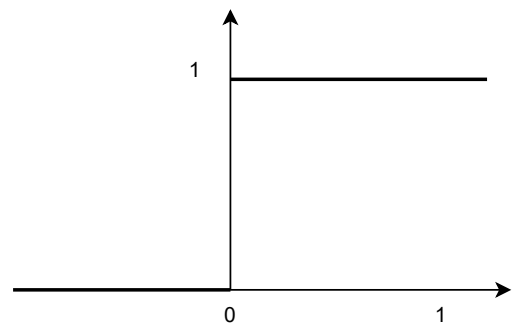


7.3.2 多階層学習における工夫

- 活性化関数の工夫
  - 勾配消失の原因はシグモイド関数の微分が最大でも0.25であることが大きい
  - 微分してもあまり小さな値にならない活性化関数を用いることで事前学習が不要になった
  - 例) ReLU(rectified linear unit) $ReLU(x) = \max(0, x)$
    - 入力が正の場合、微分値が1となり勾配消失が起こりにくい
    - 入力が負の場合、ユニットの出力が0となり学習対象のパラメータ削減に寄与する



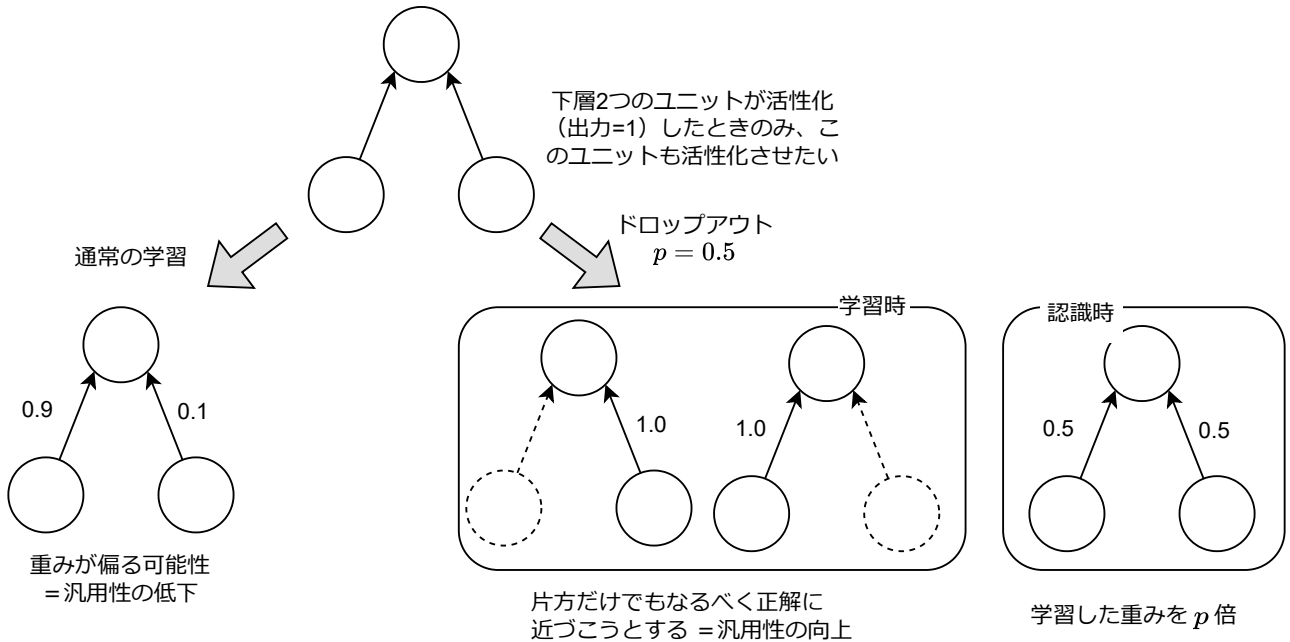
(a) rectified linear 関数



(b) (a) の導関数

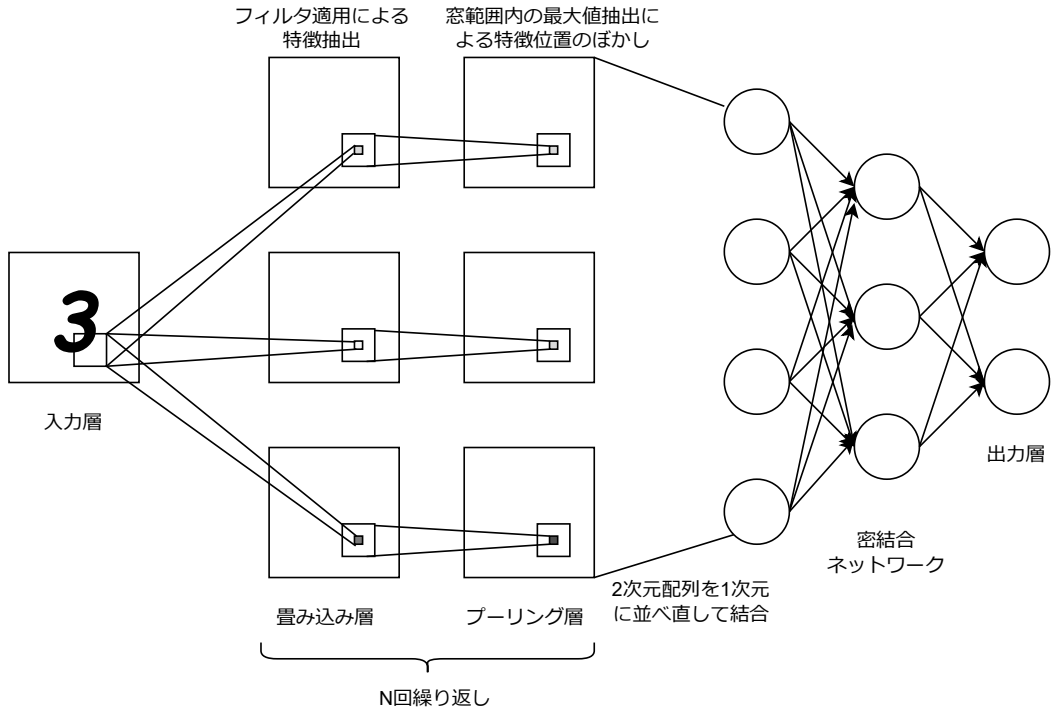
7.3.2 多階層学習における工夫

- 過学習の回避
  - ドロップアウト：ランダムに一定割合のユニットを消して学習を行う



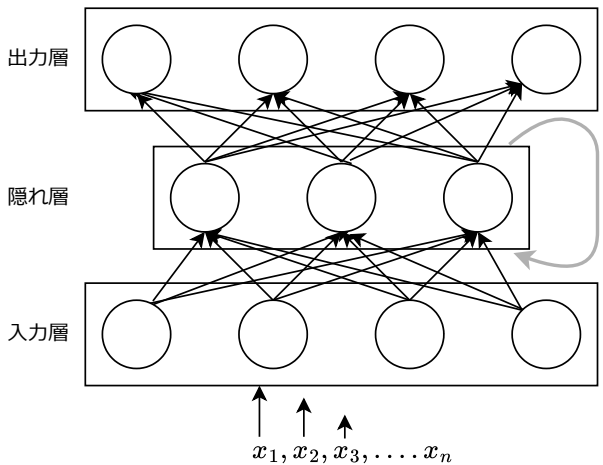
7.3.3 特化した構造をもつニューラルネットワーク（1/3）

- 畳み込みニューラルネットワーク（CNN）: 画像認識に適する
  - 入力画像に対して畳み込みとプーリングを繰り返し、最後は密結合層で識別を行う

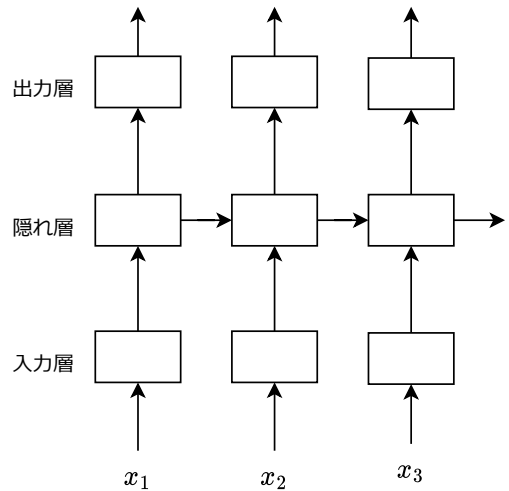


7.3.3 特化した構造をもつニューラルネットワーク（2/3）

- リカレントニューラルネットワーク（RNN）: 時系列信号認識や自然言語処理に適する
  - 隠れ層の出力を次の時刻の入力に結合することで、前方の系列の情報を保持する



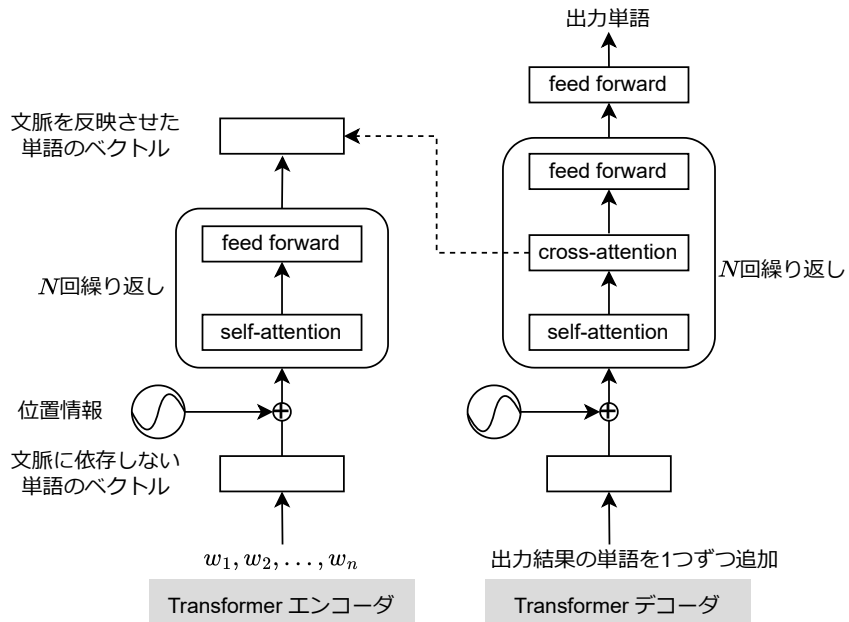
(a) リカレントニューラルネットワーク



(b) 帰還路を時間方向に展開

7.3.3 特化した構造をもつニューラルネットワーク（3/3）

- Transformer: 機械翻訳や言語生成モデルに適する
  - 隠れ層の出力計算に入力の他の部分との関係を考慮するSelf-attentionを用いる
  - BERT, GPT などの事前学習モデルに使われる



まとめ

- ニューラルネットワークの構成
  - ニューラルネットワークは誤差を最小にする確率的勾配降下法の枠組みで非線形識別面を学習できる
- 誤差逆伝播法による学習
  - 多階層のニューラルネットワークでは誤差を出力層から逆に伝えることで学習を行う
- ディープニューラルネットワーク
  - 勾配消失問題などで学習がうまくゆかないことがあったが、現在では様々な工夫により深層学習が可能になっている
  - CNN, RNN, Transformer など対象に特化したネットワーク構造がある
- Jupyter notebook