知能プログラミング演習 I

# 第 3 回: 深層ニューラルネットワーク

梅津 佑太
2 号館 404A: umezu.yuta@nitech.ac.jp

# 課題のダウンロード

前回作ったディレクトリに移動して今日の課題のダウンロードと解凍

**step1:** `cd ./DLL`

**step2:** `wget http://www-als.ics.nitech.ac.jp/~umezu/Lec3.zip`

**step3:** `unzip Lec3.zip`

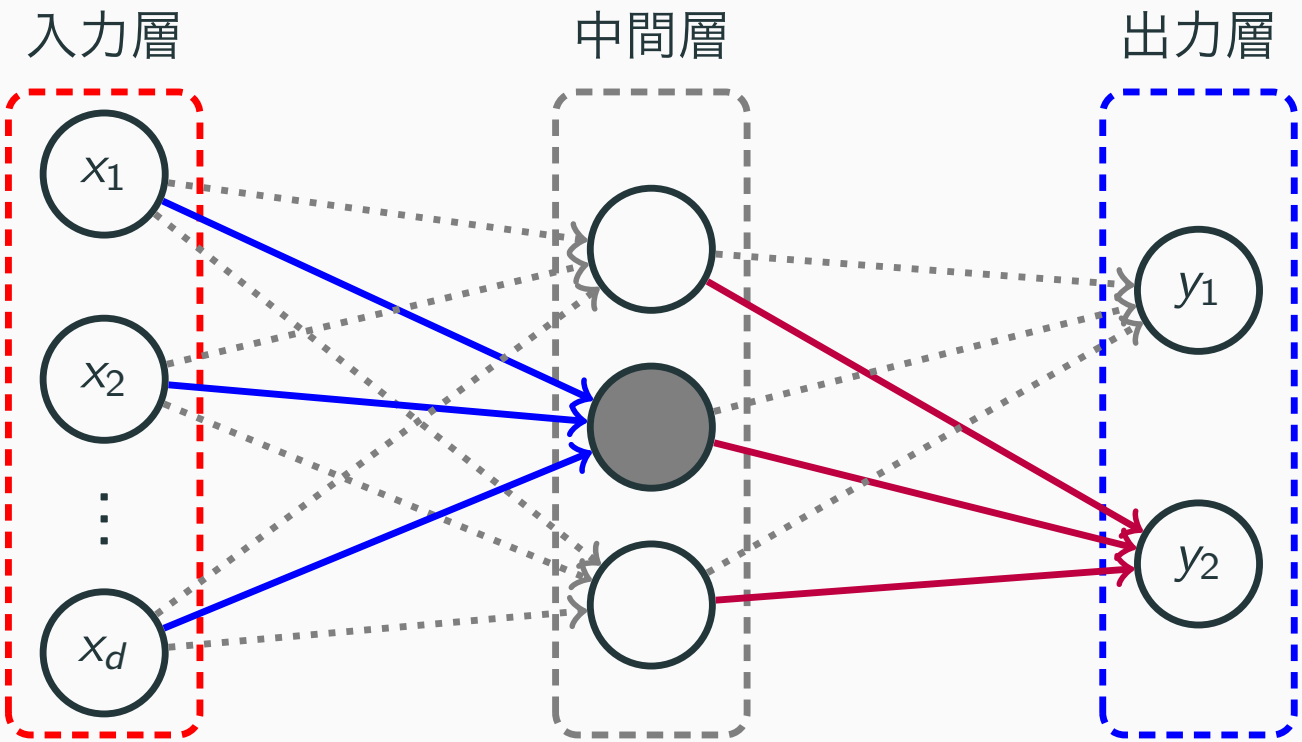
- ✓ まだ DLL のフォルダを作ってない人は, step1 の前に
  `mkdir -p DLL`
  でフォルダを作成する

講義ノート更新しました.

# 今日の講義内容

1. 深層ニューラルネットワークの学習

# 前回の復習: (3 層) ニューラルネットワーク




- パーセプトロンの合成によって少し複雑なモデルを記述
  - ✓ 各層のユニットがパーセプトロンの出力を表現している
  - ✓ すべての矢線には, 学習すべき重みがかかっている
  - ✓ 各ユニットを表現するための活性化関数は同じである必要なない

# 前回の復習: 活性化関数

- 入力層から出力層への活性化関数
  - ✓ ReLU[1]: $f(x) = \max\{0, x\}$
  - ✓ シグモイド関数: $f(x) = 1/(1 + e^{-x})$
  - ✓ ハイパボリックタンジェント: $f(x) = \tanh x = (e^{x} - e^{-x})/(e^{x} + e^{-x})$

- 中間層から出力層への活性化関数
  - ✓ 恒等写像: 回帰問題で用いられる

$$g(x) = x$$

  - ✓ ソフトマックス関数: 多クラス分類問題で用いられるもので, 出力されたベクトルは, $\boldsymbol{x}$ の所属確率を表す

$$g(\boldsymbol{x}) = \frac{1}{\sum_{k=1}^{m} e^{x_k}} (e^{x_1}, \ldots, e^{x_m})^{\top}$$

[1] Rectified Linear Unit

# 前回の復習: 誤差関数

教師データ[2] $y_1, \ldots, y_m$ とモデルの出力 $g(\boldsymbol{v}_1^\top \boldsymbol{z}), \ldots, g(\boldsymbol{v}_m^\top \boldsymbol{z})$ に対して[3],

- $\ell_2$-誤差: 回帰問題 ($g$ : 恒等写像)

$$E(W, V) = \sum_{j=1}^{m} (y_j - g(\boldsymbol{v}_j^\top \boldsymbol{z}))^2$$

- クロスエントロピー: 分類問題 ($g$ : ソフトマックス関数)

$$E(W, V) = -\sum_{i=1}^{m} y_j \log g(\boldsymbol{v}_j^\top \boldsymbol{z})$$

✓ 特に, $m = 2$ で $y \in \{0, 1\}$ の場合は

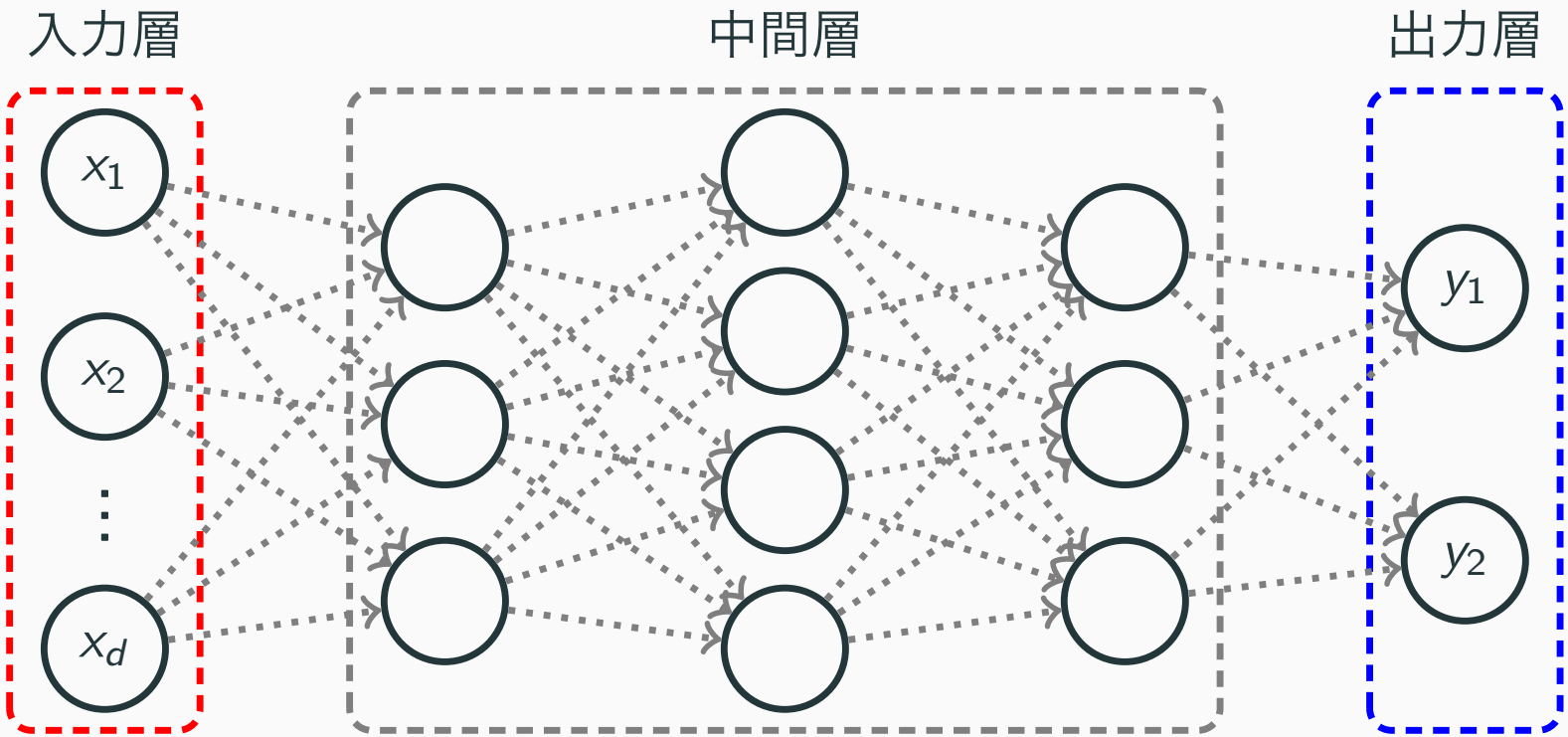
$$E(\boldsymbol{w}) = -y \log f(\boldsymbol{w}^\top \boldsymbol{x}) - (1 - y) \log(1 - f(\boldsymbol{w}^\top \boldsymbol{x}))$$

---

[2] 正解のラベルや観測した実数値

[3] 各 $k$ に対して, $g(\boldsymbol{v}_k^\top \boldsymbol{z}) = g(v_{k0} + v_{k1} f(\boldsymbol{w}_1^\top \boldsymbol{x}) + \cdots + v_{kq} f(\boldsymbol{w}_q^\top \boldsymbol{x}))$

# ディープニューラルネットワーク




- 中間層を増やす (深くする) ことで, より複雑なモデルを表現
  - ✓ 深層学習では, 入力層と出力層をつなぐ矢線上のすべての重みを学習
  - ✓ 層が深ければ深いほどパラメータ数は増加
    - 上のダイアグラムで $d = 1024$ とした場合, パラメータ数は 3102 個 orz

# 実社会における深層学習

2016年10月18日 07時00分 公開

**自動運転技術：**

## ディープニューラルネットワークで自動運転向け画像認識、デンソーと東芝が共同開発

**デンソーと東芝は、自動運転や高度運転支援の画像認識システムに向けた人工知能技術を共同開発する。東芝は人工知能技術を搭載した画像認識プロセッサーを2018年にサンプル出荷する目標だ。デンソーは2020年以降の実用化を目指す。**

[齊藤由希，MONOist]

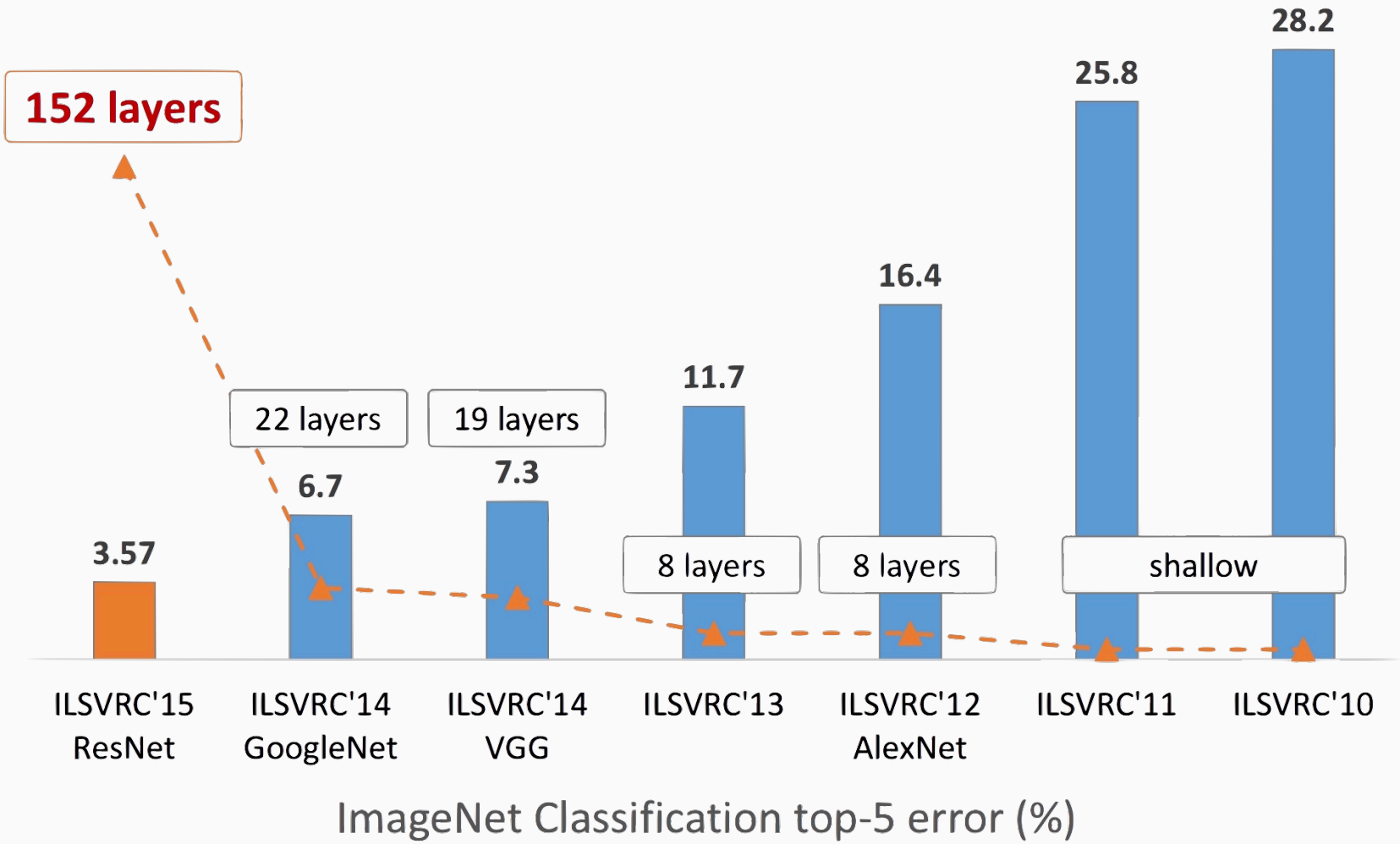
印刷／PDF　通知　見る　ツイート　52 シェア　52 いいね！　0 B! Bookmark　9 Pocket　G+

　デンソーと東芝は2016年10月17日、自動運転や高度運転支援の画像認識システム向けの人工知能（AI）技術を共同開発すると発表した。両社で車載用プロセッサーに実装可能なDNN（Deep Neural Network、ディープニューラルネットワーク）を開発し、GPUを搭載するシステムよりも低消費電力な画像処理を実現する。

http://monoist.atmarkit.co.jp/mn/articles/1610/18/news021.html

# 実社会における深層学習

# 研究としての深層学習




http://kaiminghe.com/ilsvrc15/ilsvrc2015_deep_residual_learning_kaiminghe.pdf

# なぜ層を深くするのか?

- 3 層ニューラルネットワークでも, 中間層のユニット数 $q \to \infty$ で, 任意の関数を任意の精度で近似できる

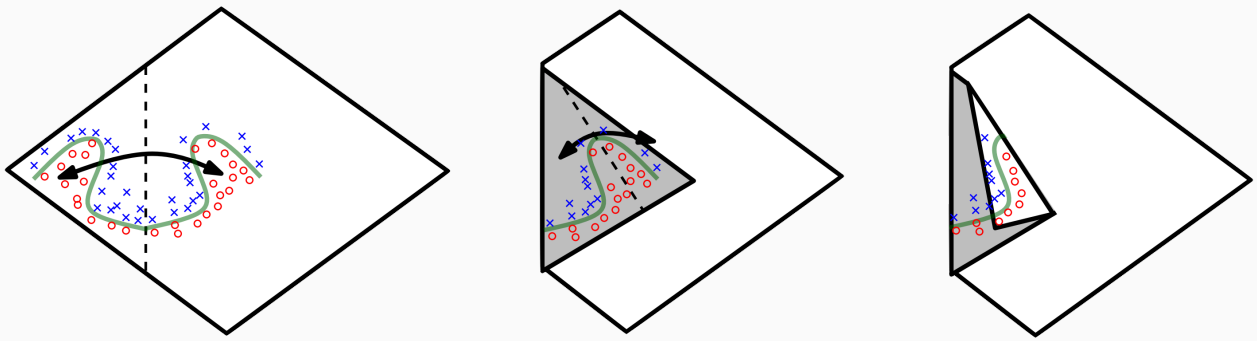
  疑問: じゃあ, なんで深い方がいいの?

  答え: 深さに対して 表現力が指数的に増大 するから

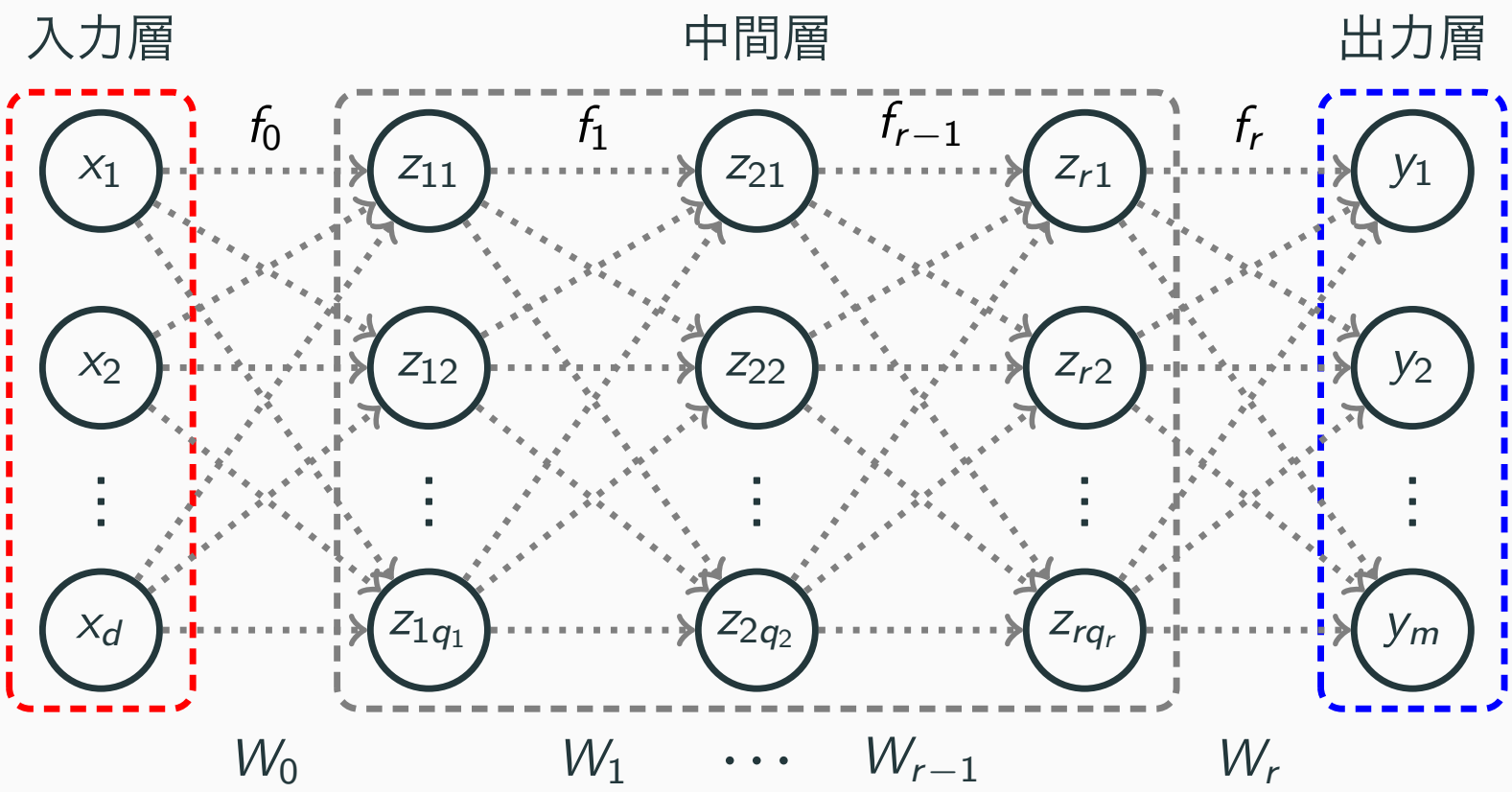
- ニューラルネットの表現力: 領域を何個の多面体に分割できるか
  - ✓ $q$ ($\geq d$): 中間層の幅 (共通), $r$: 層の数. ReLU による領域の分割数は

$$\Omega\left(\left(\frac{q}{d}\right)^{r-1}\sum_{j=0}^{q}\binom{q}{j}\right) \approx \Omega\left(\left(\frac{q}{d}\right)^{(r-1)d} q^d\right)$$

- 対称性の高い関数は, 多層にすることで得をする



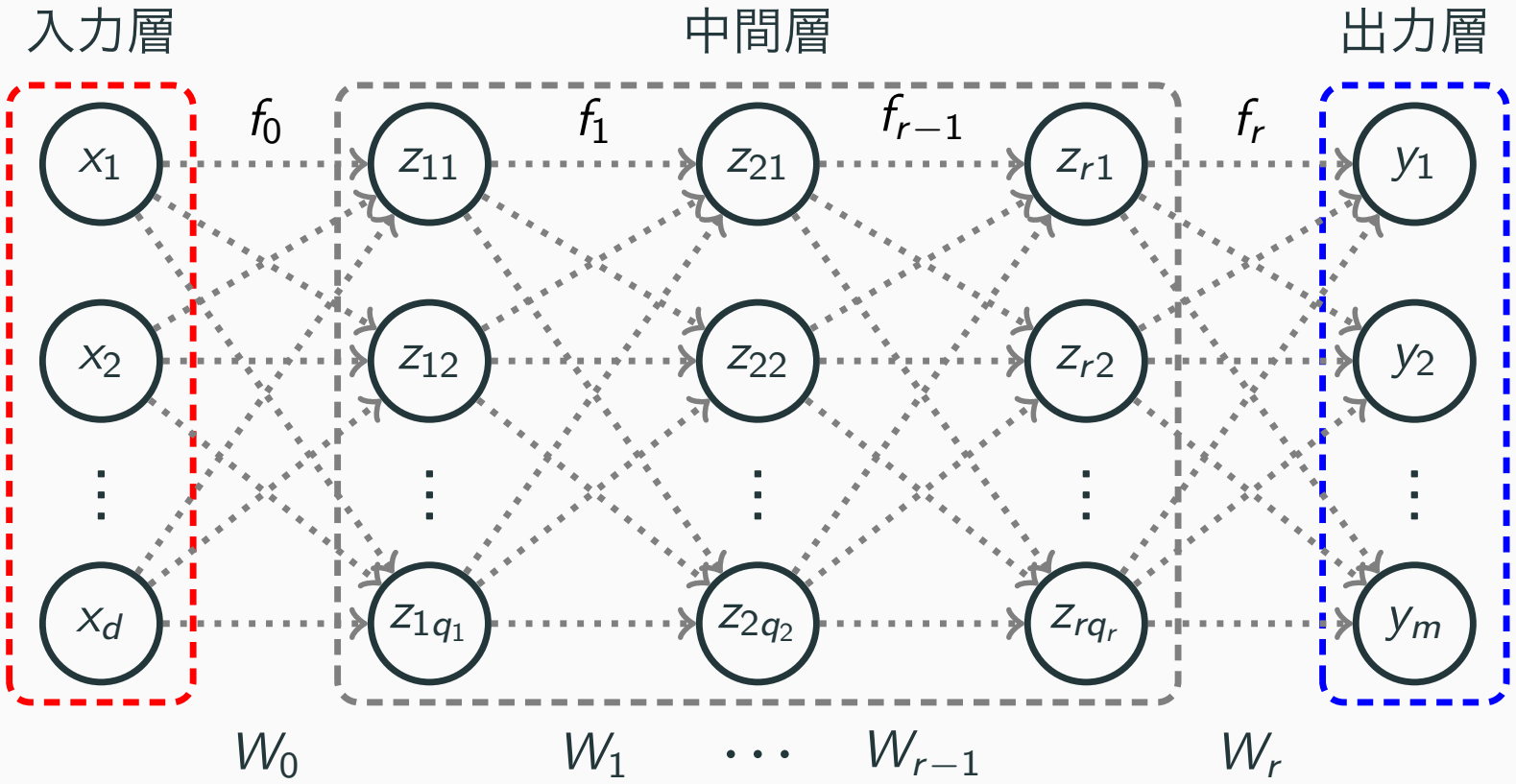
# ディープニューラルネットワークにおける誤差逆伝播法




- 最適化すべきパラメータ: $q_0 = d, q_{r+1} = m$ として,

$$W_k = (\boldsymbol{w}_{k1}, \ldots, \boldsymbol{w}_{kq_{k+1}})^\top \in \mathbb{R}^{q_{k+1} \times (q_k+1)}, \ k = 0, 1, \ldots, r$$

# 誤差逆伝播法




- モデルの関数表現: $k-1$ 層から $k$ 層への活性化関数を $f_k$ として,

$$\boldsymbol{z}_1 = f_0(W_0\boldsymbol{x})\Big(= \big(f_0(\boldsymbol{w}_{01}^\top \boldsymbol{x}), \ldots, f_0(\boldsymbol{w}_{0q_1}^\top \boldsymbol{x})\big)^\top\Big),$$

$$\boldsymbol{z}_k = f_{k-1}(W_{k-1}\boldsymbol{z}_{k-1}), \qquad k = 2, \ldots, r,$$

$$\boldsymbol{y} = f_r(W_r\boldsymbol{z}_r)$$

# 誤差関数とその微分

切片項も考慮して, $(1, \boldsymbol{z}_j^\top)^\top$ を改めて $\boldsymbol{z}_j$ とする.

- 回帰問題: 最後の中間層から出力層への活性化関数を恒等写像 $f_r(W_r \boldsymbol{z}_r) = W_r \boldsymbol{z}_r$ とし, 誤差関数は最小二乗誤差を用いる:

$$E(W_0, \ldots, W_r) = \frac{1}{2} \|\boldsymbol{y} - W_r \boldsymbol{z}_r\|^2 = \frac{1}{2} \sum_{j=1}^{m} (y_j - \boldsymbol{w}_{rj}^\top \boldsymbol{z}_r)^2$$

- 分類問題: 最後の中間層から出力層への活性化関数をソフトマックス関数 $f_r(W_r \boldsymbol{z}_r) = e^{W_r \boldsymbol{z}_r} / (\sum_{i=1}^{m} e^{\boldsymbol{w}_{ri}^\top \boldsymbol{z}_r})$ とし, 誤差関数はクロスエントロピーを用いる:

$$E(W_0, \ldots, W_r) = -\sum_{j=1}^{m} y_j \log f_r(\boldsymbol{w}_{rj}^\top \boldsymbol{z}_r)$$

# $W_r$ に関する誤差関数の微分の導出 I

回帰問題の場合は簡単なので, 分類問題だと思って計算する.

- 準備: ソフトマックス関数の第 $j$ 成分を $\boldsymbol{w}_{rk}$ で微分する.

$$f_r(\boldsymbol{w}_{rj}^\top \boldsymbol{z}_r) = \frac{1}{\sum_{i=1}^{m} e^{\boldsymbol{w}_{ri}^\top \boldsymbol{z}_r}} e^{\boldsymbol{w}_{rj}^\top \boldsymbol{z}_r}$$

より, 分数関数の微分と, $\frac{\partial e^{\boldsymbol{w}_{rj}^\top \boldsymbol{z}_r}}{\partial \boldsymbol{w}_{rk}} = e^{\boldsymbol{w}_{rk}^\top \boldsymbol{z}_r} \boldsymbol{z}_r \ (j = k)$ & $0 \ (j \neq k)$ などを用いて,

$$j = k \Rightarrow \frac{\partial f_r(\boldsymbol{w}_{rk}^\top \boldsymbol{z}_r)}{\partial \boldsymbol{w}_{rk}} =$$

$$j \neq k \Rightarrow \frac{\partial f_r(\boldsymbol{w}_{rj}^\top \boldsymbol{z}_r)}{\partial \boldsymbol{w}_{rk}} =$$

# $W_r$ に関する誤差関数の微分の導出 II

ソフトマックス関数の微分を用いて, 誤差関数 (クロスエントロピー) を $\boldsymbol{w}_{rk}$ で微分すると,

$$\begin{aligned}\frac{\partial E(W_0,\ldots,W_r)}{\partial \boldsymbol{w}_{rk}} &= -\sum_{j=1}^{m} y_j \frac{\partial \log f_r(\boldsymbol{w}_{rj}^\top \boldsymbol{z}_r)}{\partial \boldsymbol{w}_{rk}} \\ &= -\sum_{j=1}^{m} y_j \frac{1}{f_r(\boldsymbol{w}_{rj}^\top \boldsymbol{z}_r)} \frac{\partial f_r(\boldsymbol{w}_{rj}^\top \boldsymbol{z}_r)}{\partial \boldsymbol{w}_{rk}} = (f_r(\boldsymbol{w}_{rk}^\top \boldsymbol{z}_r) - y_k)\boldsymbol{z}_r\end{aligned}$$

となる [4]. $W_r = (\boldsymbol{w}_{r1},\ldots,\boldsymbol{w}_{rm})^\top$ であったから,

$$\begin{aligned}\frac{\partial E(W_0,\ldots,W_r)}{\partial W_r} &= \left(\frac{\partial E(W_0,\ldots,W_r)}{\partial \boldsymbol{w}_{r1}},\ldots,\frac{\partial E(W_0,\ldots,W_r)}{\partial \boldsymbol{w}_{rm}}\right)^\top \\ &= (f_r(W_r\boldsymbol{z}_r) - \boldsymbol{y})\boldsymbol{z}^\top\end{aligned}$$

[4] $\boldsymbol{y}$ は正解ラベルで 1, それ以外は 0 なので, $\boldsymbol{y}$ の要素をすべて足すと 1 になることに注意!

# $W_{r-1}$ に関する誤差関数の微分 I

同様に, 誤差関数を $\boldsymbol{w}_{r-1,k}$ $(k = 1, \ldots, q_r)$ で微分すると,

$$\begin{aligned}
\frac{\partial E(W_0, \ldots, W_r)}{\partial \boldsymbol{w}_{r-1,k}} &= \sum_{j=1}^{m} \frac{\partial E(W_0, \ldots, W_r)}{\partial \boldsymbol{w}_{rj}^\top \boldsymbol{z}_r} \frac{\partial \boldsymbol{w}_{rj}^\top \boldsymbol{z}_r}{\partial \boldsymbol{w}_{r-1,k}} \\
&= \sum_{j=1}^{m} (f_r(\boldsymbol{w}_{rj}^\top \boldsymbol{z}_r) - y_j) \frac{\partial \boldsymbol{w}_{rj}^\top (1, f_{r-1}(W_{r-1}\boldsymbol{z}_{r-1})^\top)}{\partial \boldsymbol{w}_{r-1,k}} \\
&= \sum_{j=1}^{m} (f_r(\boldsymbol{w}_{rj}^\top \boldsymbol{z}_r) - y_j) w_{rjk} \nabla f_{r-1}(\boldsymbol{w}_{r-1,k}^\top \boldsymbol{z}_{r-1}) \boldsymbol{z}_{r-1}
\end{aligned}$$

ここで, $\tilde{\boldsymbol{w}}_{rk} = (w_{r1k}, \ldots, w_{rmk})^\top$ は $W_r$ の $k$ 列目なので,

$$\frac{\partial E(W_0, \ldots, W_r)}{\partial \boldsymbol{w}_{r-1,k}} = f_r(W_r \boldsymbol{z}_r - \boldsymbol{y})^\top \tilde{\boldsymbol{w}}_{rk} \nabla f_{r-1}(\boldsymbol{w}_{r-1,k}^\top \boldsymbol{z}_{r-1}) \boldsymbol{z}_{r-1}$$

# $W_{r-1}$ に関する誤差関数の微分 II

したがって,

$$\begin{aligned}
\frac{\partial E(W_0, \ldots, W_r)}{\partial W_{r-1}} &= \left( \frac{\partial E(W_0, \ldots, W_r)}{\partial \boldsymbol{w}_{r-1,1}}, \ldots, \frac{\partial E(W_0, \ldots, W_r)}{\partial \boldsymbol{w}_{r-1,q_r}} \right)^\top \\
&= \begin{pmatrix} \tilde{\boldsymbol{w}}_{r1}^\top f_r(W_r \boldsymbol{z}_r - \boldsymbol{y}) \nabla f_{r-1}(\boldsymbol{w}_{r-1,1}^\top \boldsymbol{z}_{r-1}) \boldsymbol{z}_{r-1}^\top \\ \vdots \\ \tilde{\boldsymbol{w}}_{rq_r}^\top f_r(W_r \boldsymbol{z}_r - \boldsymbol{y}) \nabla f_{r-1}(\boldsymbol{w}_{r-1,q_r}^\top \boldsymbol{z}_{r-1}) \boldsymbol{z}_{r-1}^\top \end{pmatrix} \\
&= \left[ \tilde{W}_r^\top (f_r(W_r \boldsymbol{z}_r) - \boldsymbol{y}) \odot \nabla f_{r-1}(W_{r-1} \boldsymbol{z}_{r-1}) \right] \boldsymbol{z}_{r-1}^\top
\end{aligned}$$

ただし, $\tilde{W}_r$ は $W_r$ の第 1 列 (切片項に対応する部分) を除いた $q_{r+1} \times q_r$ 次元の行列.

# 誤差逆伝播法

より一般に, $u_{kl} = \boldsymbol{w}_{kl}^\top \boldsymbol{z}_k$ $(l = 1, \ldots, q_{k+1})$ とすれば,

$$\begin{aligned}\frac{\partial E(W_0, \ldots, W_r)}{\partial u_{kl}} &= \sum_{j=1}^{q_{k+2}} \frac{\partial E(W_0, \ldots, W_r)}{\partial u_{k+1,j}} \frac{\partial u_{k+1,j}}{\partial u_{kl}} \\ &= \sum_{j=1}^{q_{k+2}} \frac{\partial E(W_0, \ldots, W_r)}{\partial u_{k+1,j}} w_{k+1,jl} \nabla f_k(u_{kl})\end{aligned}$$

なので, $\delta_{kl} = \partial E(W_0, \ldots, W_r)/\partial u_{kl}$ として, 漸化式

$$\delta_{kl} = \sum_{j=1}^{q_{k+2}} \delta_{k+1,j} w_{k+1,jl} \nabla f_k(u_{kl})$$

が得られる. このとき, $\boldsymbol{\delta}_k = (\delta_{k1}, \ldots, \delta_{kq_{k+1}})^\top$ は, $W_{k+1}$ の第 1 列を除いた $q_{k+2} \times q_{k+1}$ 次元の行列を用いて, 以下のように書ける:

$$\boldsymbol{\delta}_k = \tilde{W}_{k+1}^\top \boldsymbol{\delta}_{k+1} \odot \nabla f_k(W_k \boldsymbol{z}_k), \qquad (k = 0, 1, \ldots, r-1)$$

# 誤差逆伝播法

したがって, 第 $k$ 層において, パラメータに関する微分は,

$$\frac{\partial E(W_0, \ldots, W_r)}{\partial \boldsymbol{w}_{kl}} = \frac{\partial E(W_0, \ldots, W_r)}{\partial u_{kl}} \frac{\partial u_{kl}}{\partial \boldsymbol{w}_{kl}} = \delta_{kl} \boldsymbol{z}_k$$

より,

$$\frac{\partial E(W_0, \ldots, W_r)}{\partial W_k} = \begin{pmatrix} \delta_{k1} \boldsymbol{z}_k^\top \\ \vdots \\ \delta_{kq_{k+1}} \boldsymbol{z}_k^\top \end{pmatrix} = \left\{ \tilde{W}_{k+1}^\top \boldsymbol{\delta}_{k+1} \odot \nabla f_k(W_k \boldsymbol{z}_k) \right\} \boldsymbol{z}_k^\top$$

したがって, パラメータの更新式は

$$W_k \leftarrow W_k - \eta \left\{ \tilde{W}_{k+1}^\top \boldsymbol{\delta}_{k+1} \odot \nabla f_k(W_k \boldsymbol{z}_k) \right\} \boldsymbol{z}_k^\top$$

# 疑似コード

エポック内で誤差逆伝播法のアルゴリズムは以下の通り. ただし, $\eta$ は適当な学習率.

1. ランダムにデータを読み込む: $(\boldsymbol{y}, \boldsymbol{x}) \in \{(\boldsymbol{y}_i, \boldsymbol{x}_i) \mid i = 1, \ldots, n\}$
2. 順伝播:
   - 2.1 $\boldsymbol{z}_1 \leftarrow f_0(W_0 \boldsymbol{x})$
   - 2.2 $\boldsymbol{z}_{k+1} \leftarrow f_k(W_k \boldsymbol{z}_k)$ $(k = 1, \ldots, r-1)$
3. 逆伝播:
   - 3.1 $\boldsymbol{\delta}_r \leftarrow f_r(W_r \boldsymbol{z}_r) - \boldsymbol{y}$
   - 3.2 $\boldsymbol{\delta}_k \leftarrow \tilde{W}_{k+1}^{\top} \boldsymbol{\delta}_{k+1} \odot \nabla f_k(W_k \boldsymbol{z}_k)$ $(k = 0, 1, \ldots, r-1;\ \boldsymbol{z}_0 = \boldsymbol{x})$
4. 重みの更新: $W_k \leftarrow W_k - \eta \boldsymbol{\delta}_k \boldsymbol{z}_k^{\top}$