数理生物学演習

第14回　機械学習と深層学習
(CIFAR-10を用いた画像認識・分類の実装)

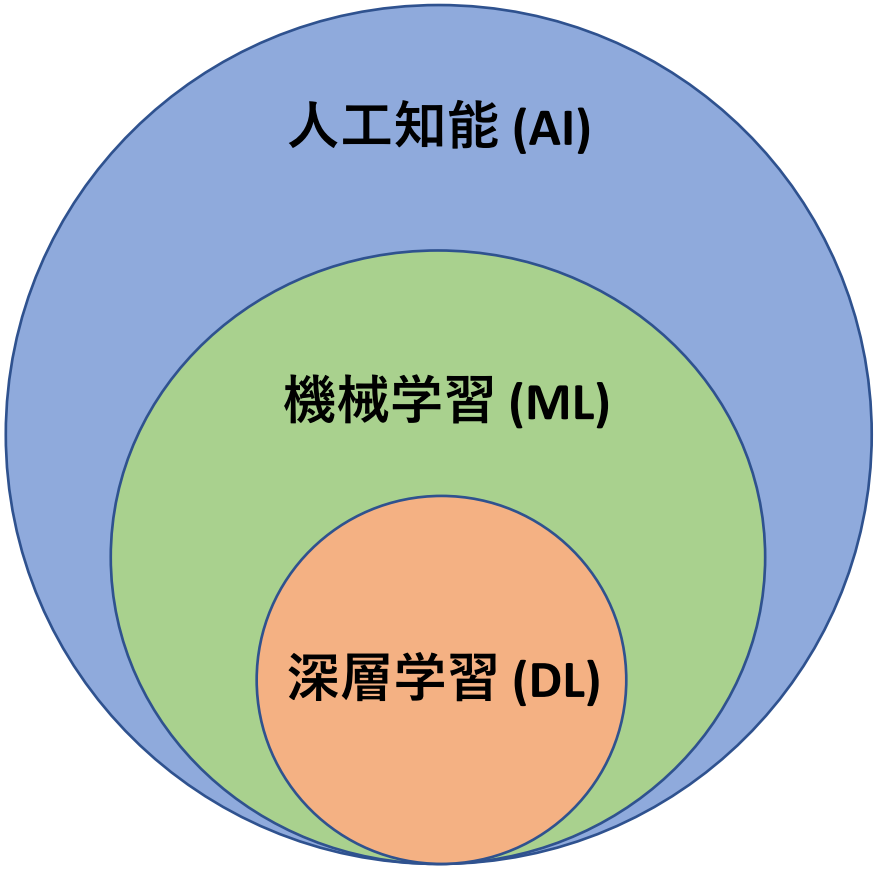
染野大輝
someno@morphometrics.jp
九州大学システム生命科学府
数理生物学研究室

本日の内容

1. 機械学習と深層学習の理解
2. ニューラルネットワークの理解
3. 畳み込みニューラルネットワークの理解
4. 画像認識・分類の実装

1. 機械学習と深層学習の理解

人工知能, 機械学習, 深層学習



- **人工知能 (Artificial Intelligence)**

人間の知能を模倣した技術

- **機械学習 (Machine Learning)**

与えられたデータに対して, そのパターンを見つけ, その問題を解決する技術

- **深層学習 (Deep Learning)**

機械学習の一部で, 人間の脳構造を模したニューラルネットワークを用いて, 与えられた高度なタスクをこなす技術

機械学習 (Machine Learning)

- **教師あり学習**

予測データと教師データ (正解データ)を見比べてモデルの学習を行う手法

例) 回帰：面積や駅からの距離から家賃を予測

　　分類：スパムメールかどうかの判断

- **教師なし学習**

教師データは無く, 与えられたパターンを分析してタスクをこなす手法

例) クラスタリング：k-means法　　※教師あり学習の「分類」と混同しないように！

　　次元削減：PCA

- **強化学習**

報酬が最大となるように行動を試行錯誤する手法

例) 自動運転, 将棋AI

深層学習 (Deep Learning)

機械学習の一部で, ニューラルネットワークを用いて高度なタスクをこなす。
特に, 大量の教師データを用いた教師あり学習の発展が目覚ましい。

画像認識・分類

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (n.d.). ImageNet Classification with Deep Convolutional Neural Networks.

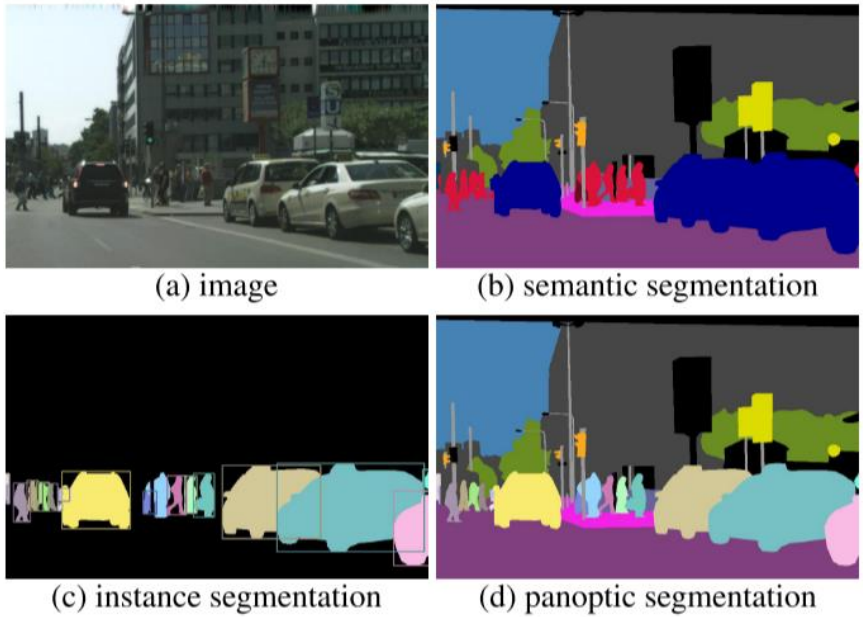
自然言語処理

DeepL

https://en.wikipedia.org/wiki/DeepL_Translator

https://ja.wikipedia.org/wiki/ChatGPT

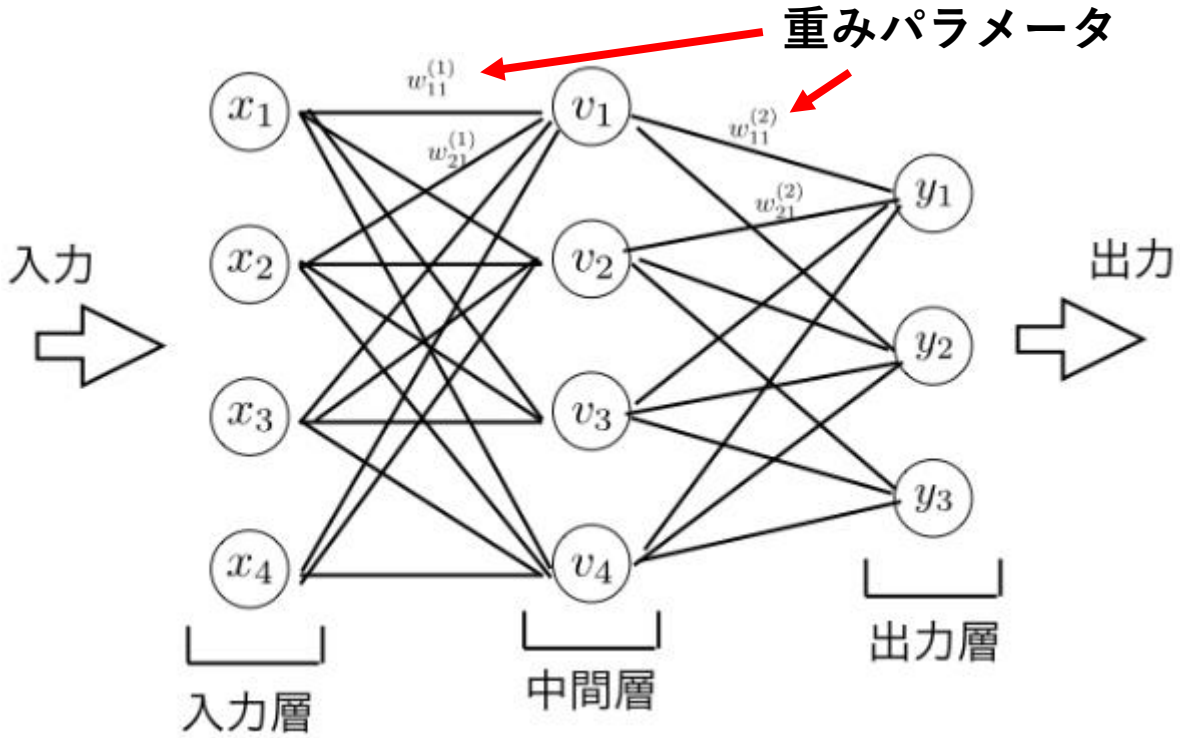
領域分割



Kirillov, A., He, K., Girshick, R., Rother, C., & Dollár, P. (n.d.). *Panoptic Segmentation*.

2. ニューラルネットワークの理解

ニューラルネットワーク

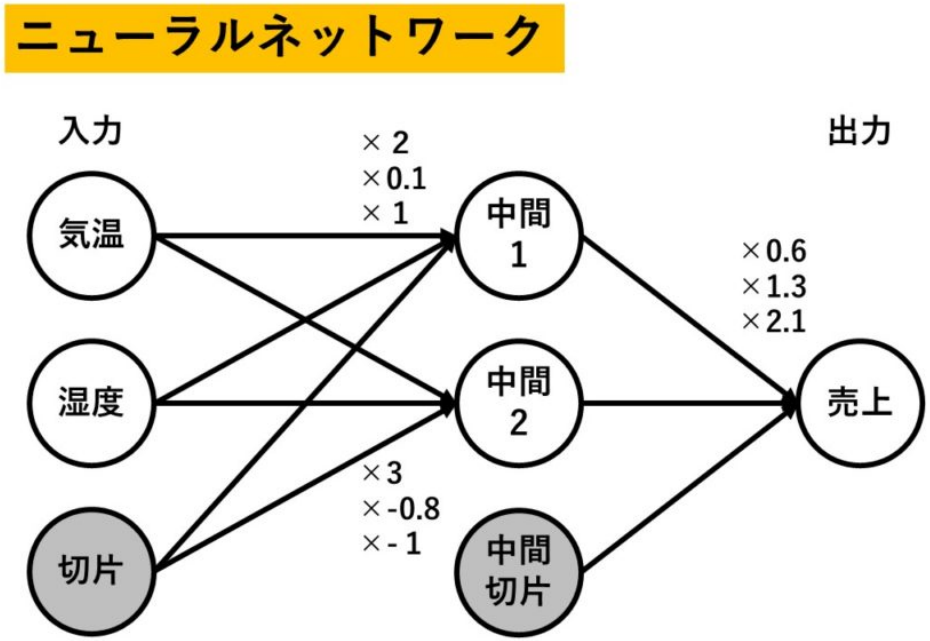


https://atmarkit.itmedia.co.jp/ait/articles/1811/20/news012.html

人間の脳が持つ神経細胞 (ニューロン)を模した構造を持ち,
最適な重みパラメータ (w)を見つけるように学習していく。

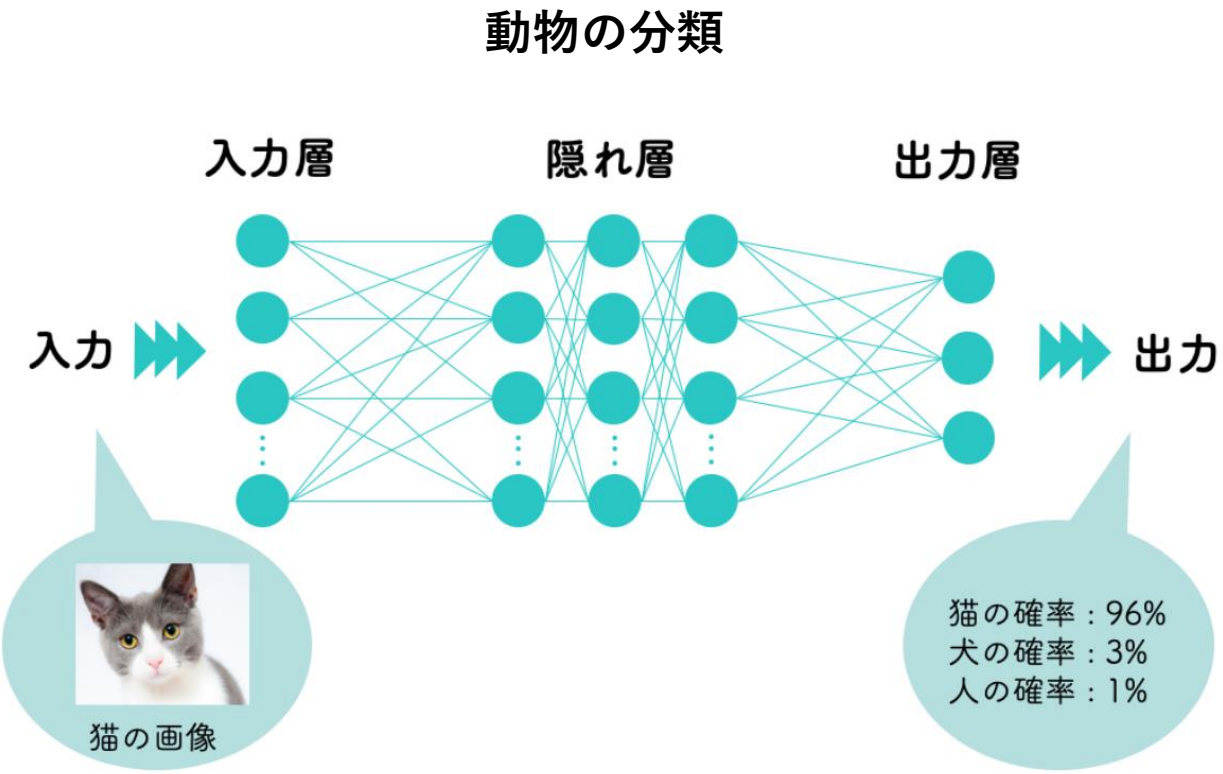
ニューラルネットワークの例 (回帰)

ビールの売上予測



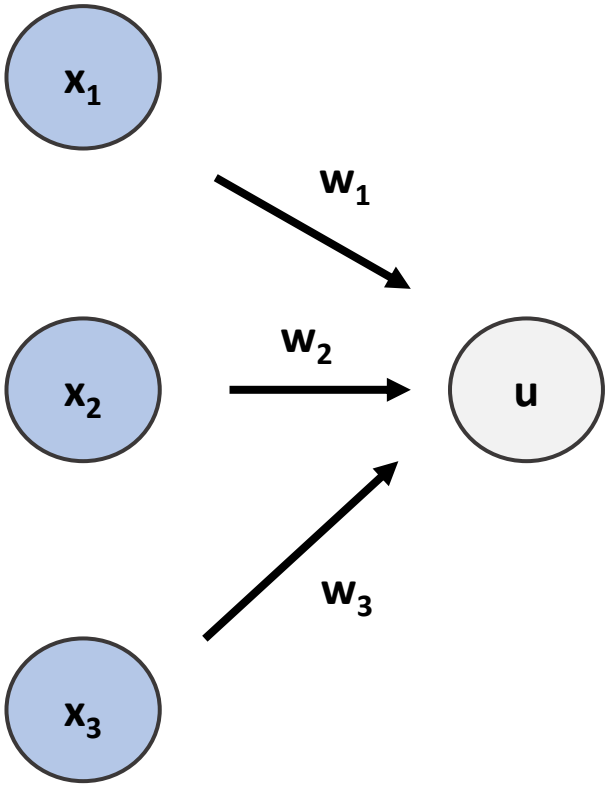
https://logics-of-blue.com/time-series-analysis-by-nnet/

ニューラルネットワークの例 (分類)



https://ainow.ai/2019/08/06/174245/

単純パーセプトロン

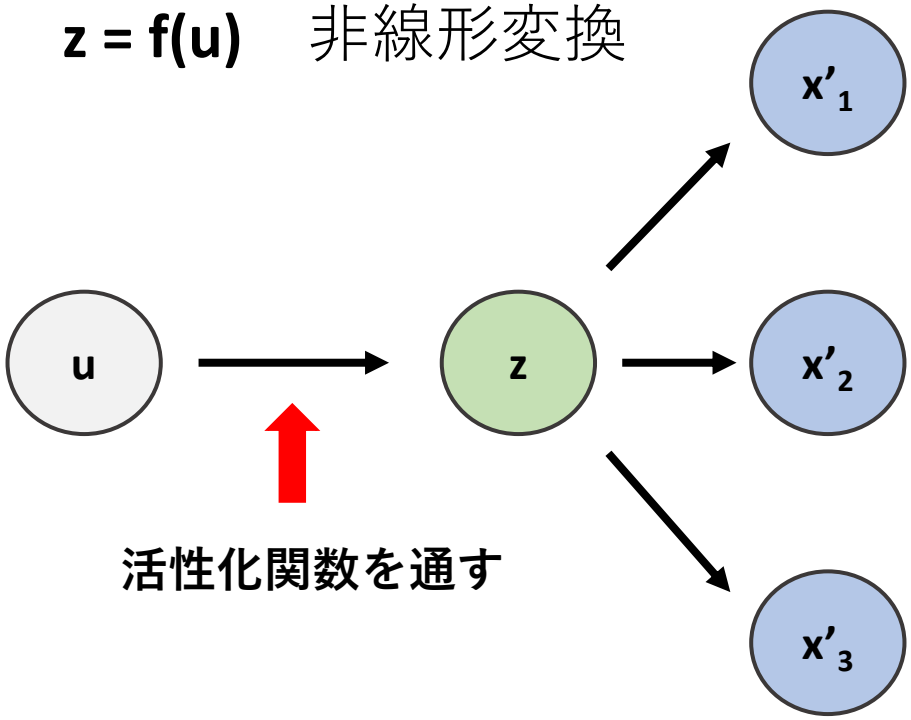


線形結合

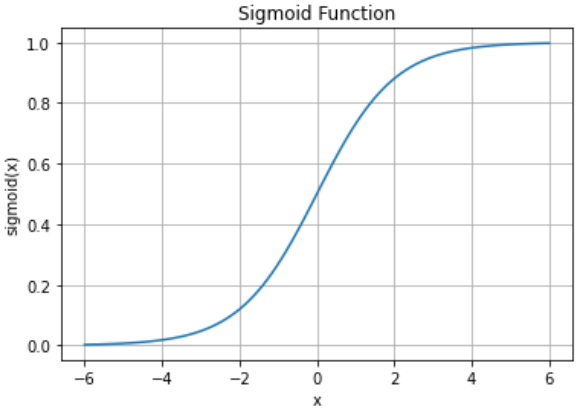
$$u = x_1w_1 + x_2w_2 + x_3w_3$$

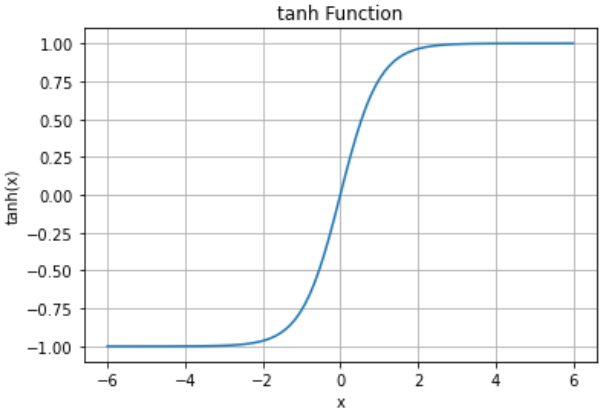
活性化関数

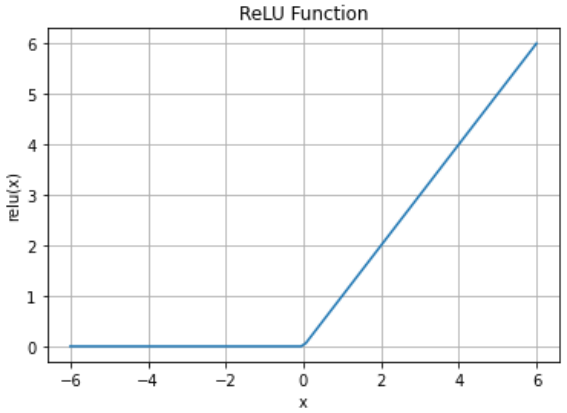
z = f(u)　非線形変換



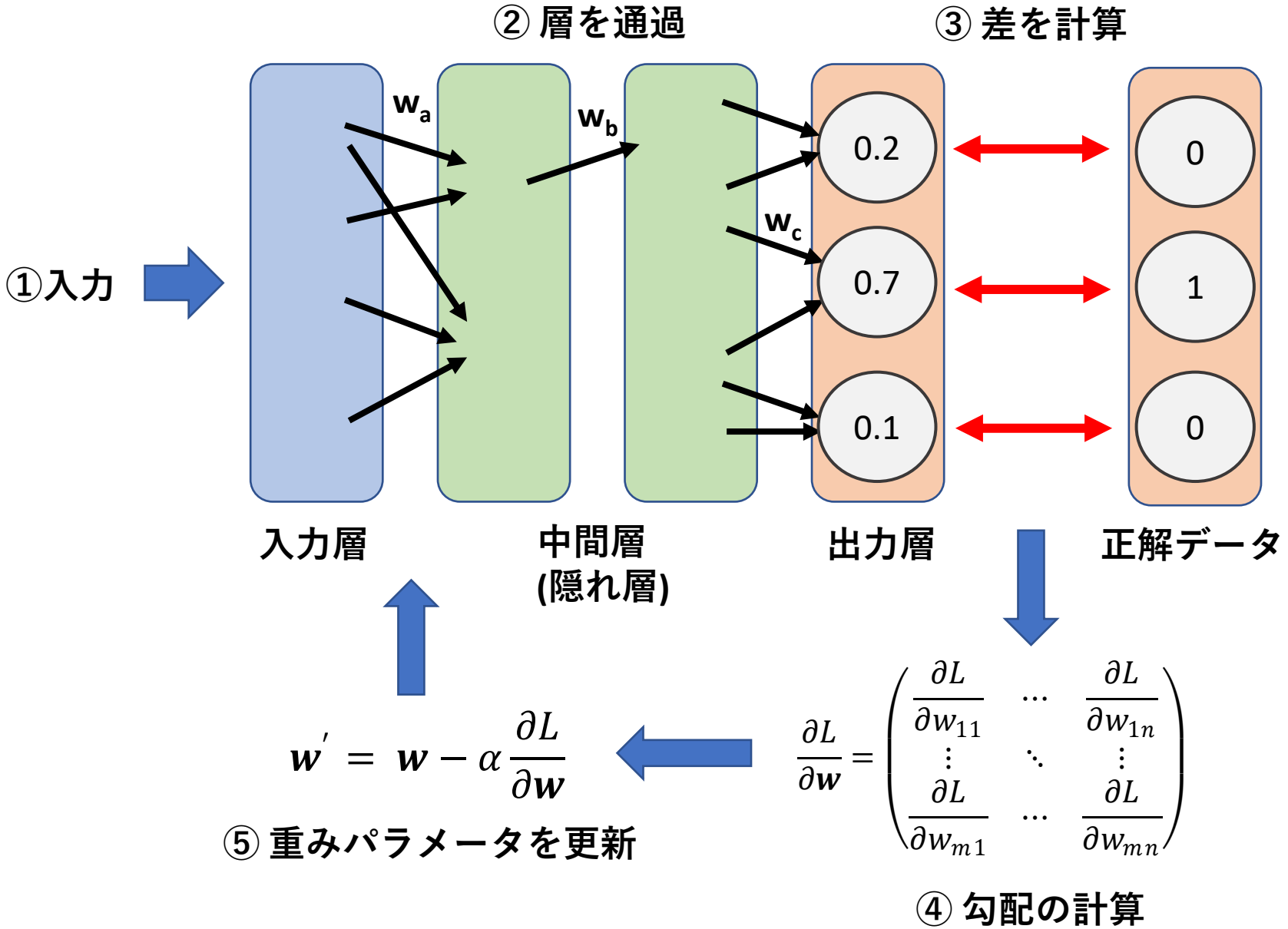
zは次の層に対する入力データとなる







ニューラルネットワークの学習の流れ (多クラス分類)



ニューラルネットワークの学習の流れ

今回の講義で
一番伝えたいこと

ニューラルネットワークでの
学習の一番の目的は予測値と
正解データの差をできるだけ
小さくすること。

そのためには更新を繰り返して
最適な重みパラメータを見つけ
る必要がある。

① 入力データを受け取る

② 入力層～中間層～出力層
を通って, 予測値を出力

③ 予測値と正解データの差を計算

④ 重みパラメータの勾配を計算

⑤ 重みパラメータを更新

①～⑤の作業を繰り返す

③ 予測値と正解データの差を計算

回帰問題

平均二乗誤差

$$L = \frac{1}{N}\sum_{i=1}^{N}(y_i - t_i)^2$$

y = 予測値
t = 正解データ

Nはデータの数
(※ ミニバッチの大きさ)

分類問題

交差エントロピー誤差 (多クラス分類)

$$L = -\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K} t_k \log y_k$$

y = 予測値
t = 正解データ

Nはデータの数
(※ ミニバッチの大きさ)

Kはクラスの数

④ 重みパラメータの勾配を計算 (理解が難しい)

$$\frac{\partial L}{\partial \boldsymbol{W}} = \begin{pmatrix} \frac{\partial L}{\partial w_{11}} & \cdots & \frac{\partial L}{\partial w_{1n}} \\ \vdots & \ddots & \vdots \\ \frac{\partial L}{\partial w_{m1}} & \cdots & \frac{\partial L}{\partial w_{mn}} \end{pmatrix}$$

損失を計算したら, 左のようにニューラルネットワーク内の重みパラメータ (+ バイアス ***b***)の勾配を求める。

勾配を求める時には**誤差逆伝播法**が使われる。

出力層に近い方の重みパラメータから勾配を求めていく。

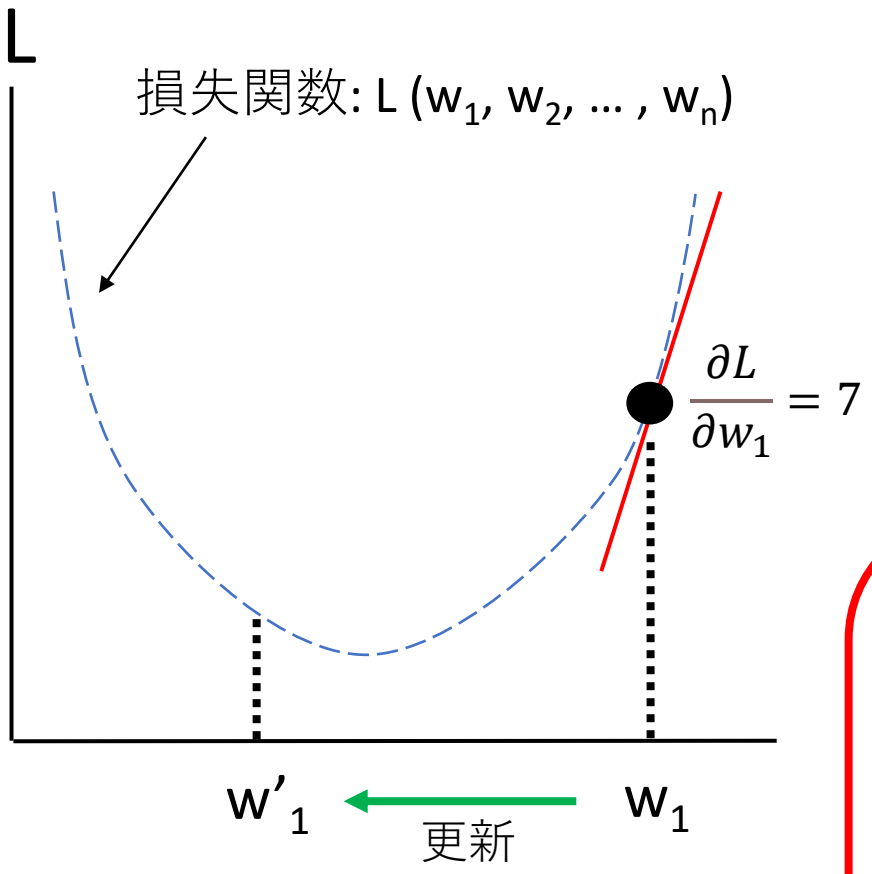
ざっくりとしたイメージ：
$\mathbf{w_n}$の勾配はn+1層目以降の勾配の情報を使った連鎖律によって求めることができる。

※ 誤差逆伝播法の説明は複雑で時間が掛かるので今回は説明を割愛させていただきます。

気になる方は右の書籍が分かりやすいのでぜひ読んでみて下さい。

ゼロから作るDeep Learning
—Pythonで学ぶディープラーニングの理論と実装
斎藤 康毅　著

⑤ 重みパラメータを更新



勾配降下法という手法を用いて重みパラメータが更新される。

損失関数Lは多次元の関数だが, 分かりやすいように左図では1つの重みパラメータw_1だけに着目する。

ニューラルネットワークでの学習の一番の目的は予測値と正解データの損失差をできるだけ小さくすること。

↓

そのためには損失差が小さくなる重みパラメータを見つける必要がある。

↓

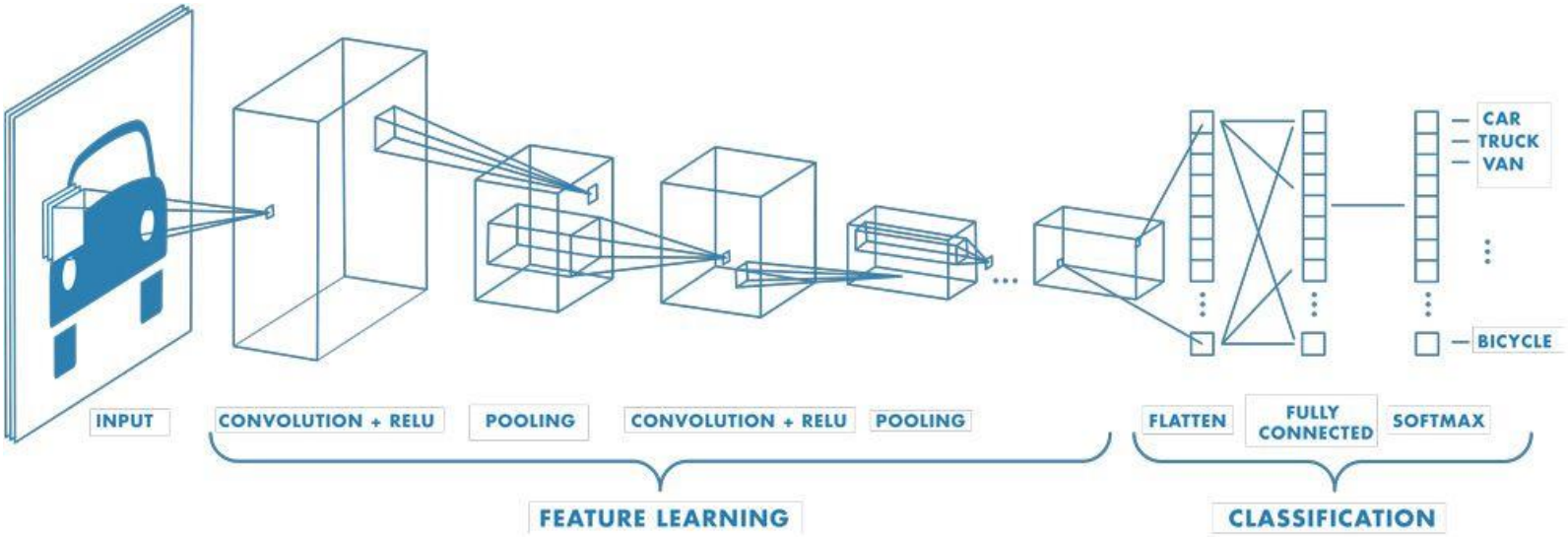
最適な重みパラメータの探索には勾配計算が役に立つ。

$$w_1' = w_1 - \alpha \frac{\partial L}{\partial w_1} \quad (\alpha > 0)$$

- 勾配 > 0のとき, 左側にw_1が移動
- 勾配 < 0のとき, 右側にw_1が移動

3. 畳み込みニューラルネットワークの理解

畳み込みニューラルネットワーク (CNN)



https://jp.mathworks.com/discovery/convolutional-neural-network-matlab.html?ref=blog.paperspace.com

画像を入力データとする場合, 普通のニューラルネットワークでは1次元にデータが加工されてしまうが, CNNではそのまま層に流すことが出来る。

画像に写る位置情報を失わずに学習を進めることが出来る！

CNNの構造の例

畳み込み層 → 活性化関数 → プーリング層 → 畳み込み層 → 活性化関数 → プーリング層 → 畳み込み層 → 活性化関数 → プーリング層 →

畳み込み層 → 活性化関数 → プーリング層 → 全結合層 → 全結合層 → 全結合層 → Sigmoid関数 or Softmax関数 → 出力

1次元に変換

全結合層は上で見た普通の
ニューラルネットワークと同じ構造

**前半の方の層では画像の抽象的な部位を,
後半の方の層ではきめ細かい部位の特徴量
(特徴マップ)抽出を行う。**

畳み込み層 (2次元の場合)

畳み込み演算

カーネル (フィルター)がニューラルネットワークでの**重みパラメータ**の役割を果たす。

入力データ (5x5)

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

*

カーネル (3x3)

1	2	3
4	5	6
7	8	9

?		

1x1+2x2+3x3+6x4+7x5+8x6+11x7+12x8+13x9 = **411**

入力データ (5x5)

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

*

カーネル (3x3)

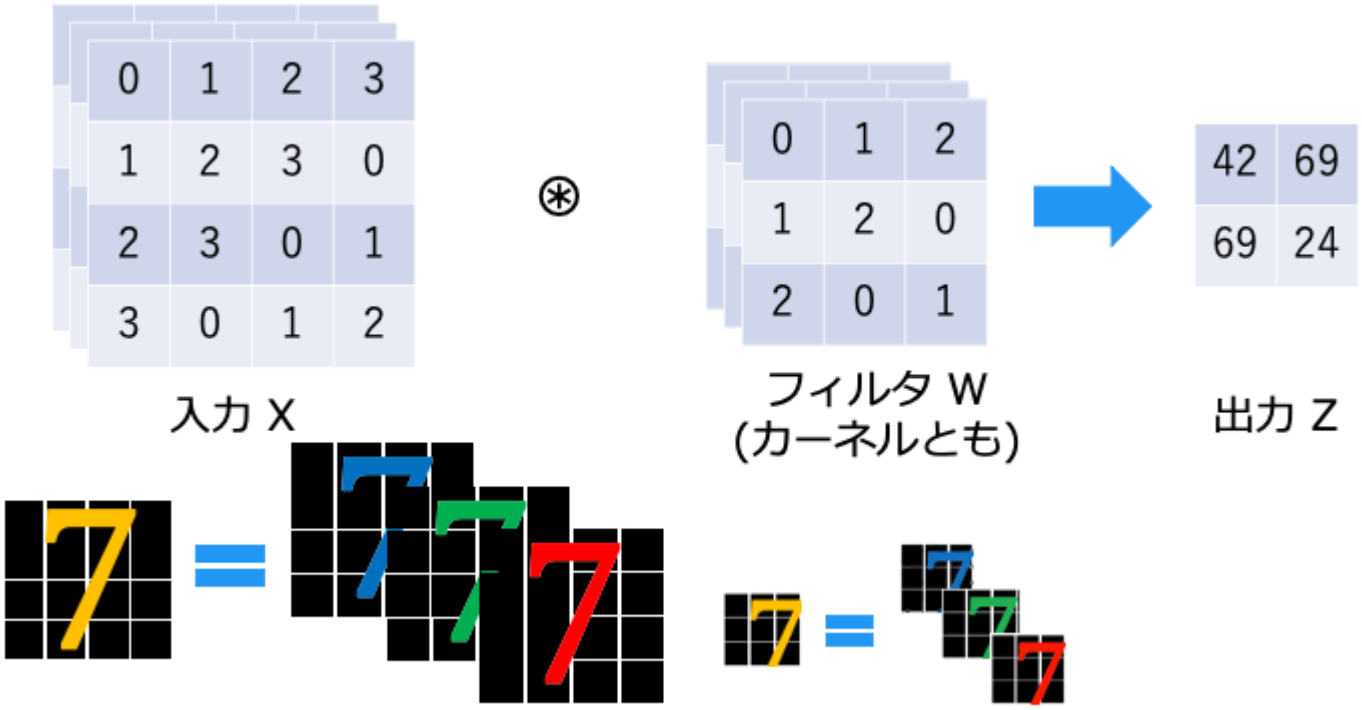
1	2	3
4	5	6
7	8	9

411	?	

https://data-analytics.fun/2021/11/23/understanding-convolution/

畳み込み層 (3次元の場合)

入力データのチャネル数 = 3　　**カーネルのチャネル数 = 3**



https://hogetech.info/machine-learning/deep-learning/dl-basic5

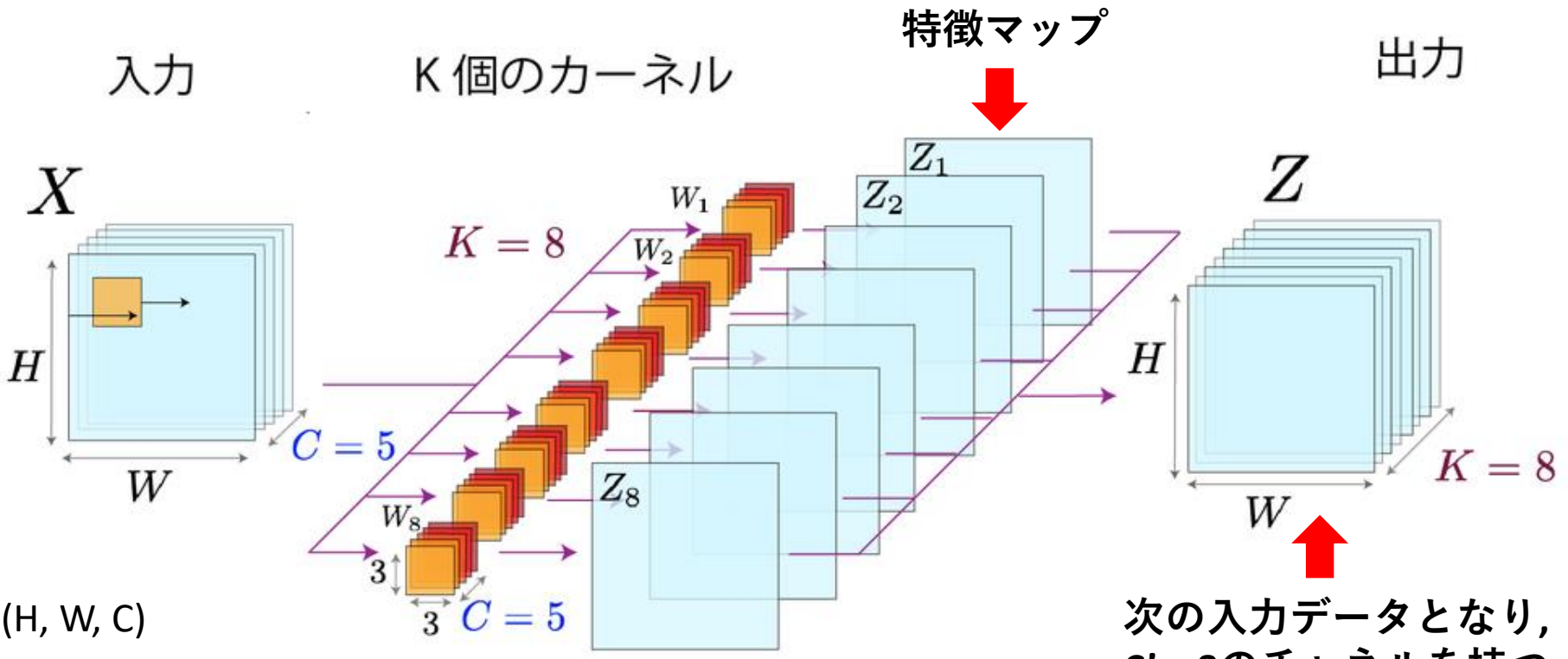
入力データのチャネル数とカーネルが持つチャネル数は同じ (C = 3)！

入力データのi番目のチャネルとカーネルのi番目のチャネルで畳み込み演算を行う。
その後, 得られた値をチャネル同士で合算する。

※ バイアスも考慮する必要がある。

畳み込み層の全体像

入力データのチャネル数とカーネルが持つチャネル数は同じ (C = 5)！



(H, W, C)

= (縦, 横, チャネル数)

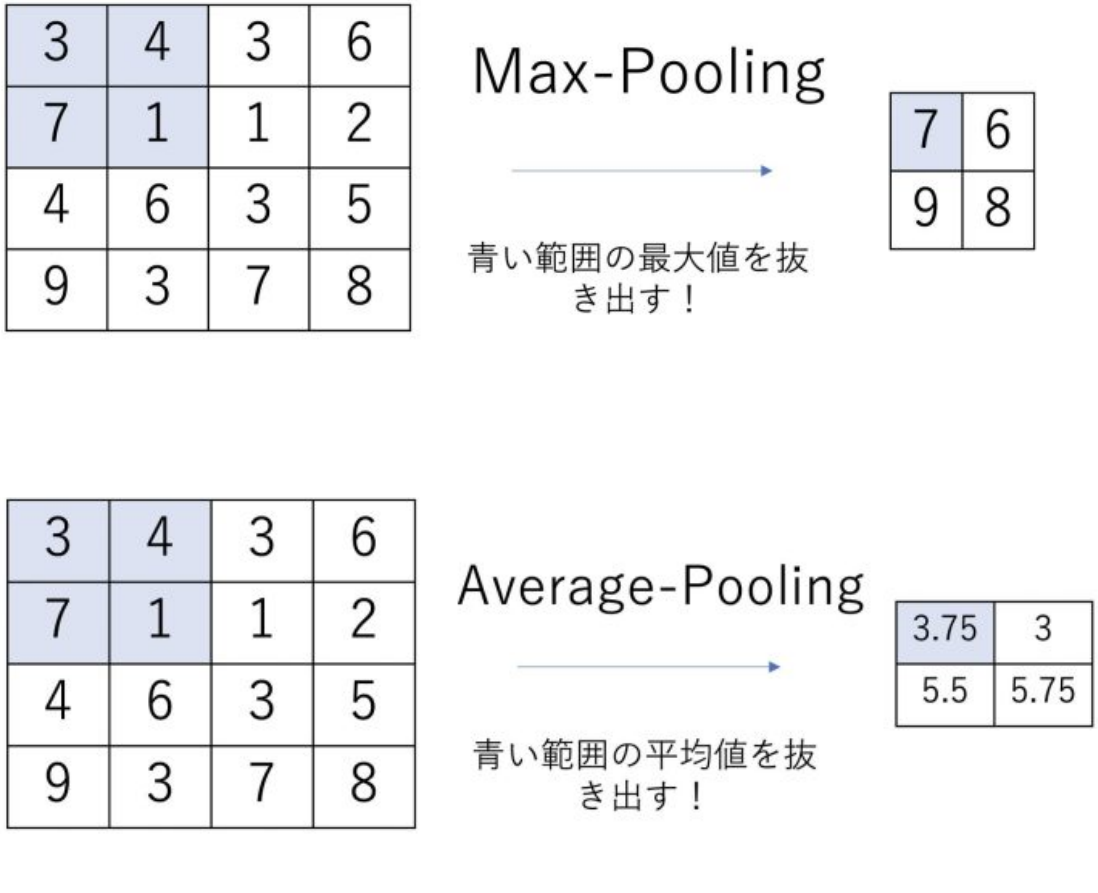
次の入力データとなり，C’ = 8のチャネルを持つ。

https://cvml-expertguide.net/terms/dl/layers/convolution-layer/

カーネルの数と出力データの特徴マップの数は同じ (K = 8)！

チャネル数Cのカーネルを複数個 (K)用意することで複数の特徴マップを獲得することができる。

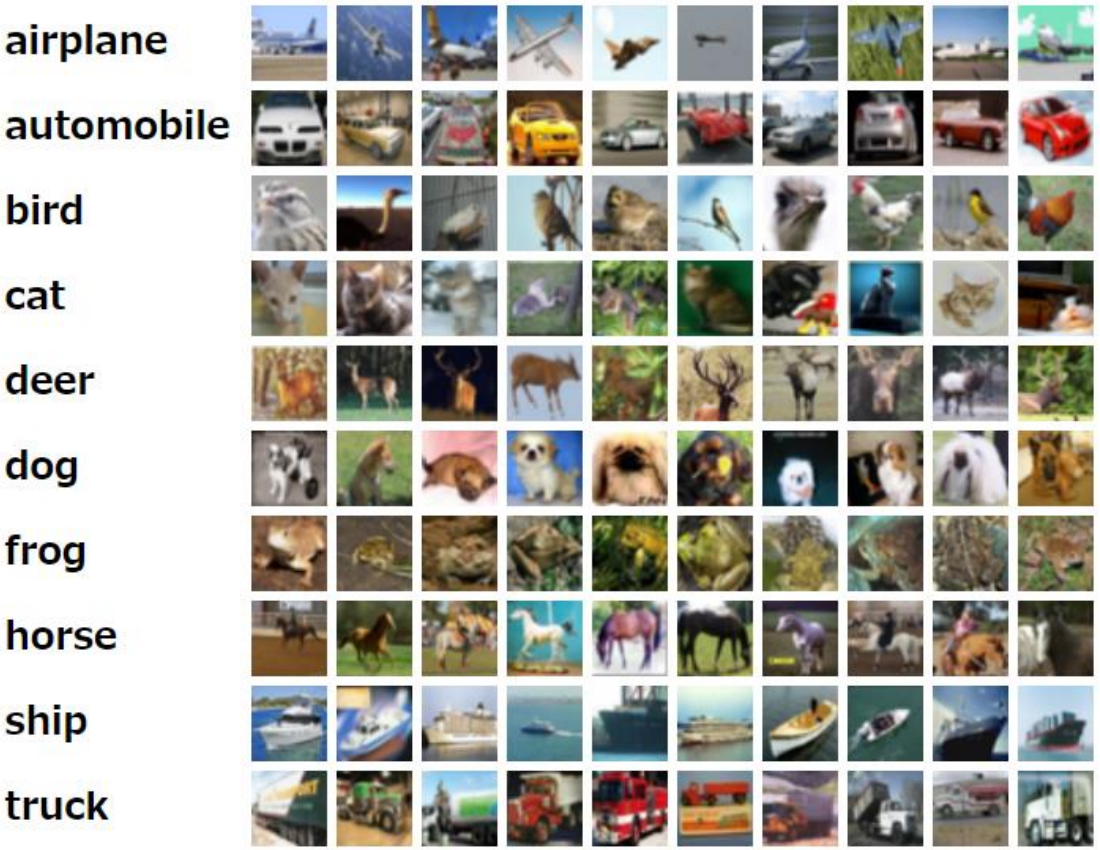
プーリング層



https://prog-you.com/pooling/

4. 画像認識・分類の実装

今回使う画像データセット



https://www.cs.toronto.edu/~kriz/cifar.html

The CIFAR-10 dataset

10クラスのRGB画像 (32x32x3のサイズ)を格納した60,000枚のデータセット (1クラスに6,000枚)

実装の大まかな流れ

今回皆さんにはCNNモデルを用いた画像認識・分類を体験してもらいます。

① CIFAR-10のデータセットをダウンロードして, 訓練用, 検証用, テスト用に分割する。
訓練用：40,000枚, 検証用：10,000枚, テスト用：10,000枚

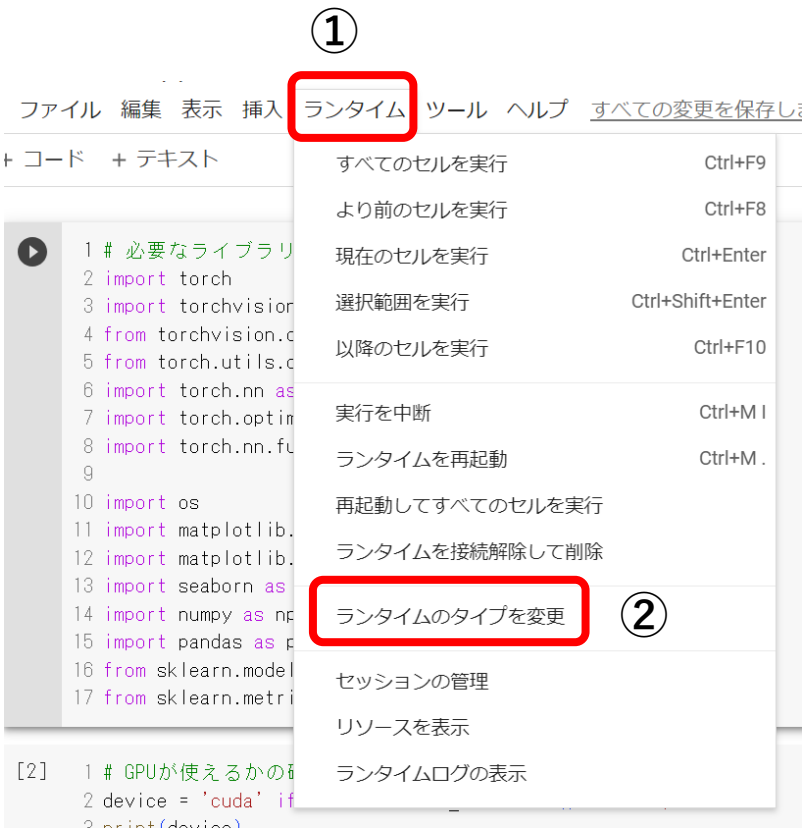
② CNNモデル, 損失関数, 最適化アルゴリズム (勾配降下法)などを定義する。

③ 訓練用, 検証用データを用いてCNNモデルを学習する。

④ テスト用データと③で得られた学習済みCNNモデルを用いて推論
(正解データと予測値がどれほど一致しているかの確認)を行う。

GPUの設定

ノートブックを開いたら, 以下のようにGPUが使えるように設定して下さい。




```
# GPUが使えるかの確認
device = 'cuda' if torch.cuda.is_available() else 'cpu'
print(device)
```

```
cuda
```

← “cuda”であることを確認

ノートブックのダウンロード

復習用に使いたい方はダウンロードしておいて下さい。

データ拡張

学習を行うに当たって, データの水増し処理をすることができる (動的水増し処理)。

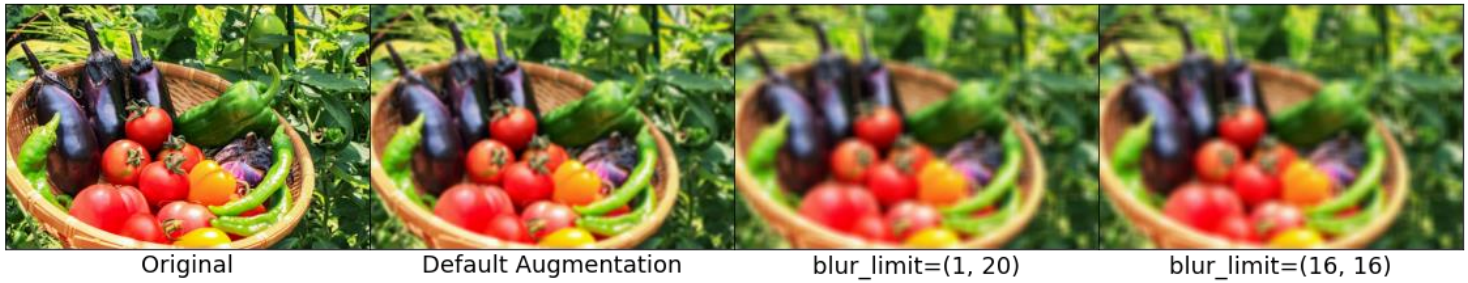
※ 実際にデータの数が増えるということではない。

データの拡張を行うことで, **過学習**をある程度抑えることが期待できる。

VerticalFlip

HorizontalFlip

Blur

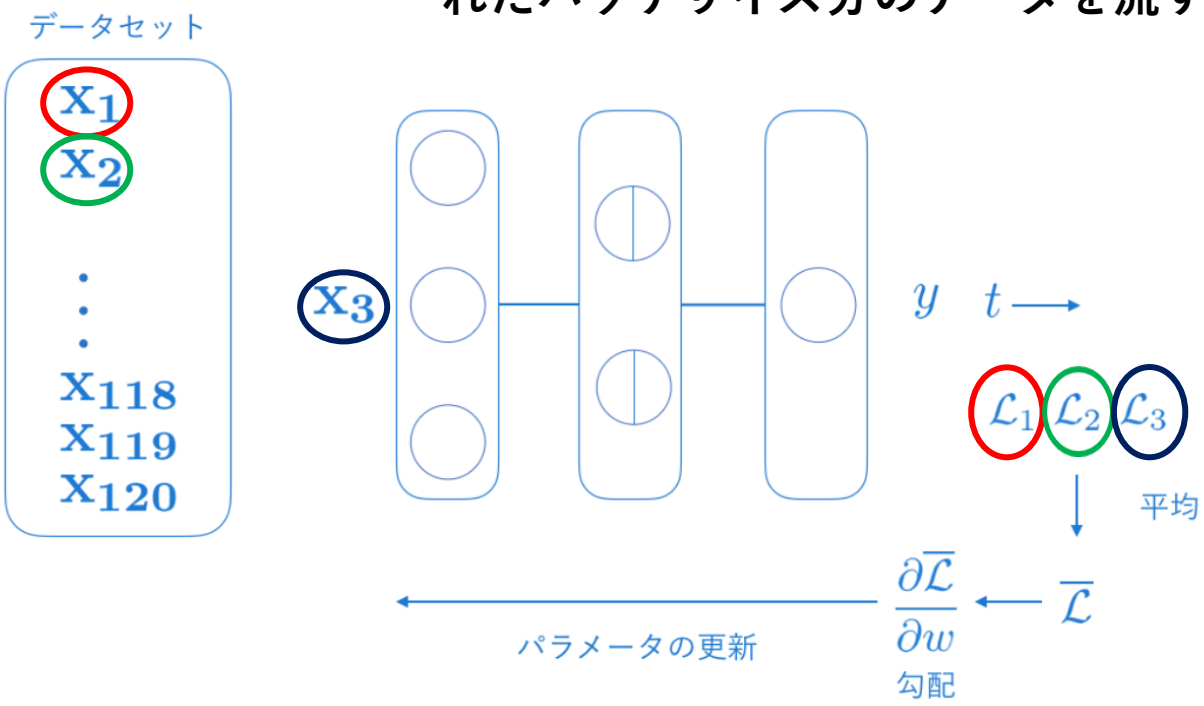


https://qiita.com/kurilab/items/b69e1be8d0224ae139ad

ミニバッチ学習

バッチサイズ＝3 の場合

1イテレーション (重みパラメータを1回更新する)ごとに決められたバッチサイズ分のデータを流す。



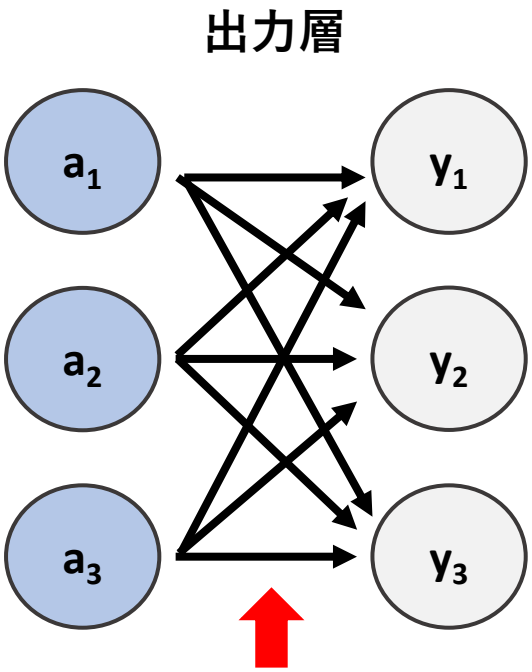
x_1を流した時の損失**L_1**,
x_2を流した時の損失**L_2**,
x_3を流した時の損失**L_3**,
3つの**L**を平均して,
それを勾配計算に用いる。

https://free.kikagaku.ai/tutorial/basic_of_deep_learning/learn/neural_network_basic_backward

バッチ学習：全データセットを一度に流して, 損失差を求めて, 重みパラメータを更新する。

オンライン学習：学習データを1個流して, 損失差を求めて, 重みパラメータを更新する。

Softmax関数

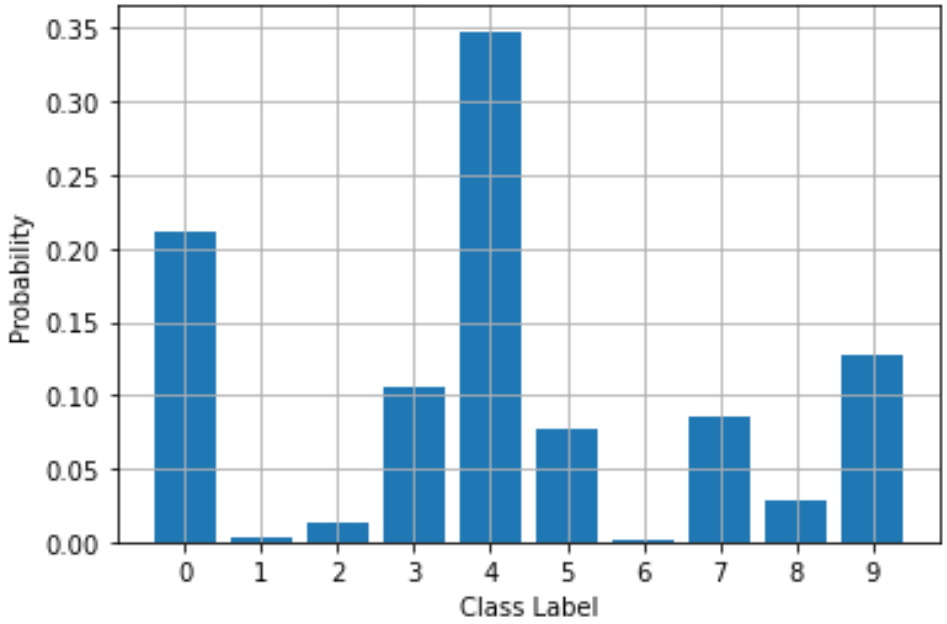


$$y_k = \frac{\exp(a_k)}{\sum_{i=1}^{n} \exp(a_i)}$$

N = 出力の数, k = 番号

例)

出力層の0～9番目の値が
[3.0, -1.5, 0.2, 2.3, 3.5, 2.0, -2.2, 2.1, 1.0, 2.5]
のとき, 以下のように合計が1になるように確率が出力される。



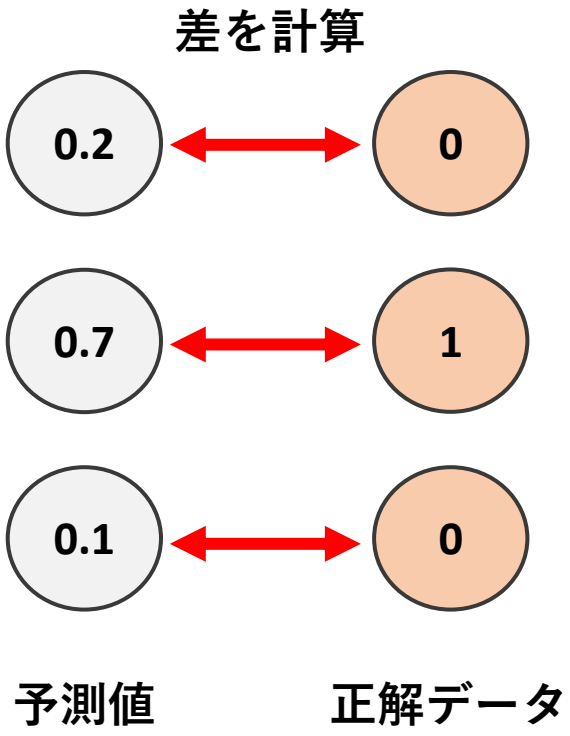
One-hot vectorへの変換

3クラス分類の場合
[イヌ: 0, ネコ: 1, ヒト: 2]

正解がネコ (ラベル番号: 0)の場合,
正解データが (1, 0, 0)に変換される。

正解がネコ (ラベル番号: 1)の場合,
正解データが (0, 1, 0)に変換される。

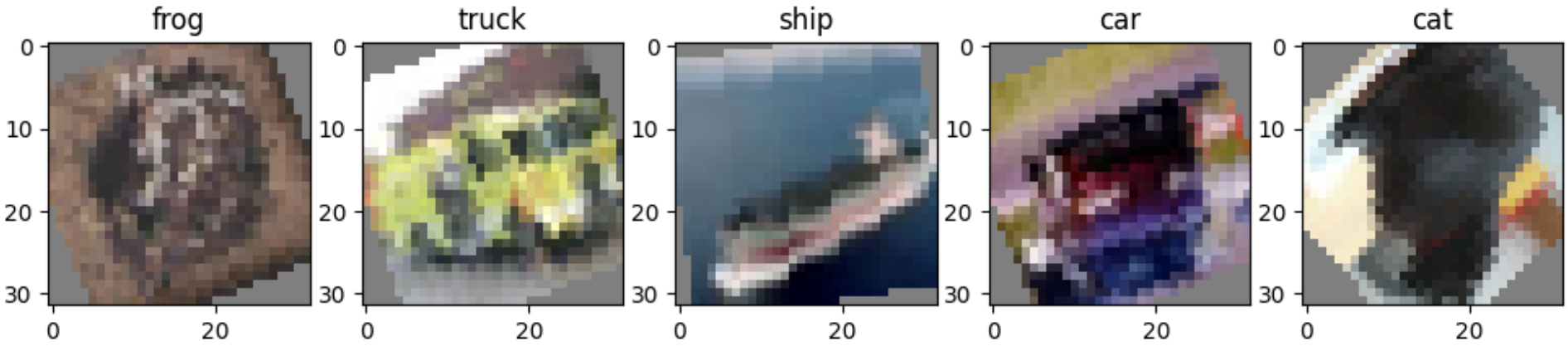
正解がヒト (ラベル番号: 2)の場合,
正解データが (0, 0, 1)に変換される。



交差エントロピー誤差 (多クラス分類)での計算 (ネコが正解のとき)

$$L_1 = -\{\underbrace{(0 * \log 0.2)}_{\text{0になる}} + \underbrace{(1 * \log 0.7)}_{\text{ここだけ計算}} + \underbrace{(0 * \log 0.1)}_{\text{0になる}}\} = 0.357$$

画像データと正解ラベルの確認

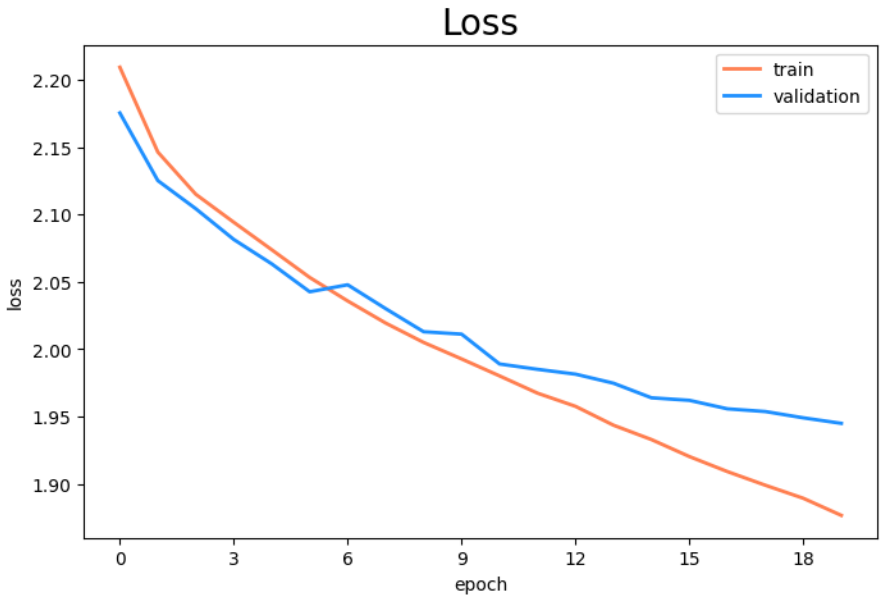


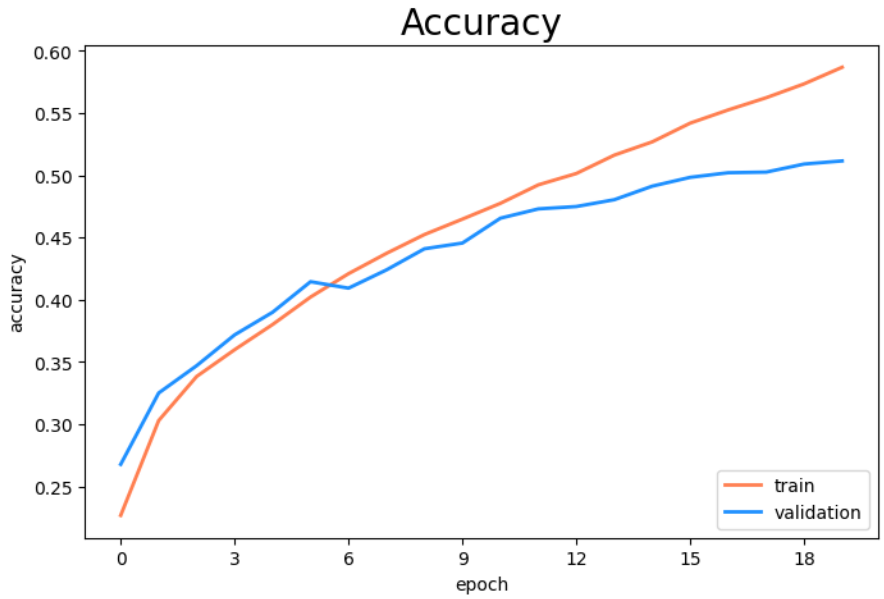
(縦, 横, チャネル数) = (32, 32, 3)のピクセルしかないため, ぼやけて見えてしまうが気にせずに。

学習中の損失差と正解率の推移 (Epoch = 20)

学習が進むごとに損失差が小さくなっている。

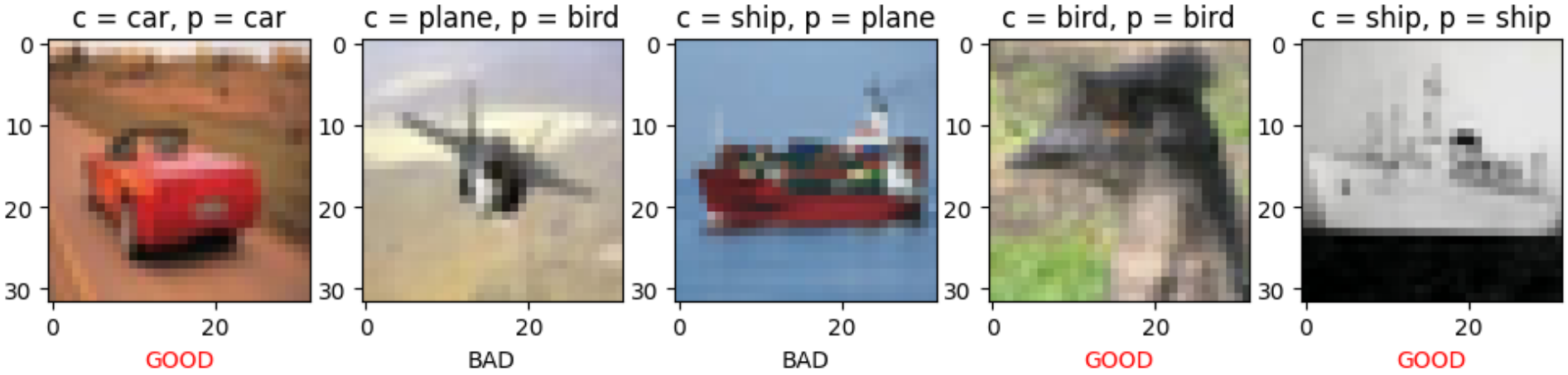
正解率も上がっている。





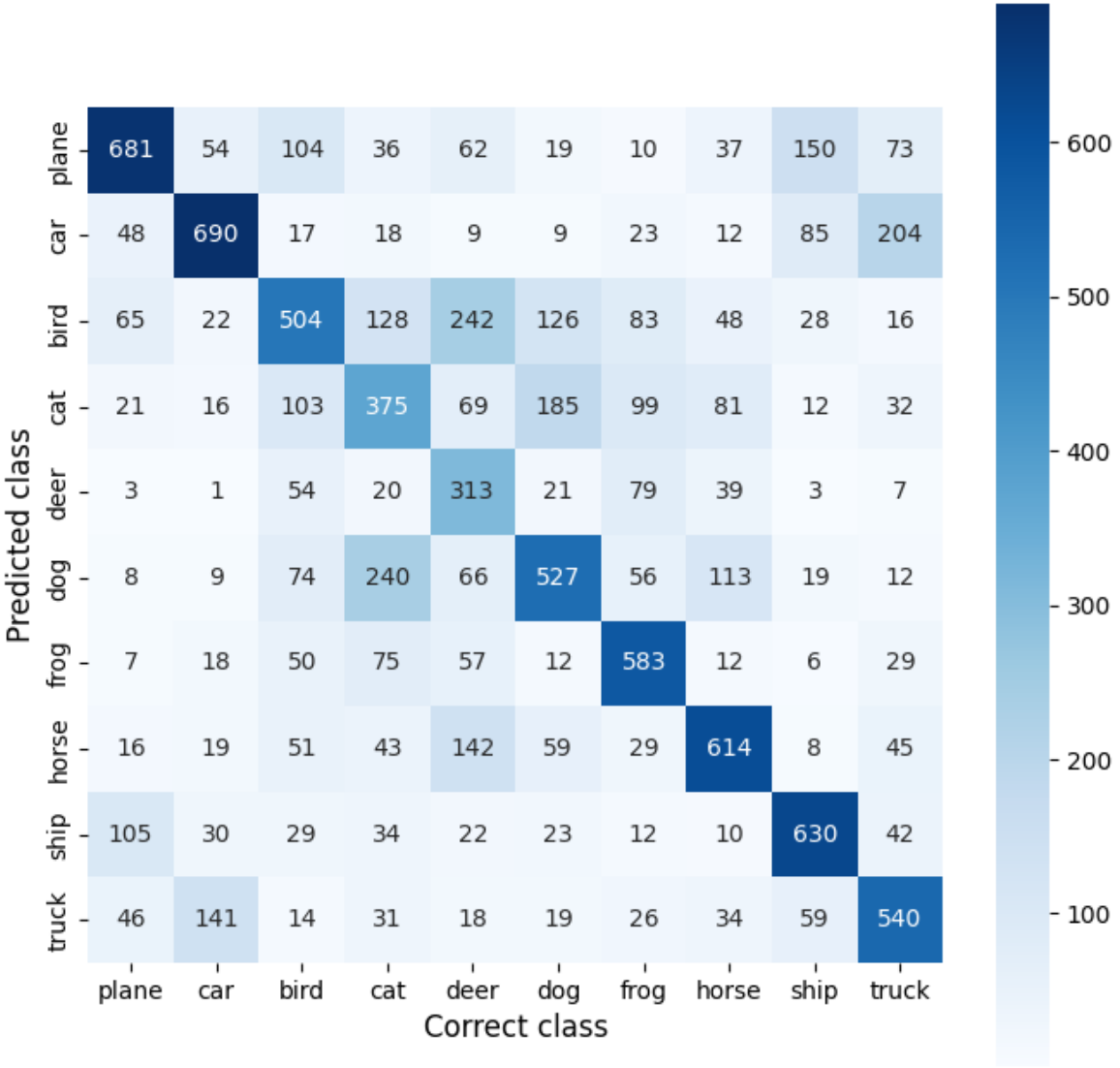
ただし, Epochの数をさらに増やすと過学習が起こることが予想される (Epochが進むにつれてtrainとvalidationの差が広がる)。

正解クラスと予測クラスの比較



c = 正解クラス　p = 予測クラス

Heatmapで表示



区別が難しいと思われるもの

- **正解クラス → 予測クラス**
- 飛行機 → 船
- 車 → トラック
- 鳥 → 飛行機&ネコ
- ネコ → イヌ
- シカ → 鳥
- イヌ → ネコ
- カエル → ネコ
- 馬 → イヌ
- 船 → 飛行機
- トラック → 車

最後に

- ニューラルネットワークを学ぶにあたり, まだ他に大事なこと (Dropout, Batch Normalization, Early Stoppingなど)がありますが, そこは各自で勉強してもらえればと思います。
- 説明やスライドに不足なところや誤りがあるかもしれませんが, その点はご了承下さい。

参考

- ゼロから作るDeep Learning ―Pythonで学ぶディープラーニングの理論と実装　斎藤 康毅 著
- つくりながら学ぶ! PyTorchによる発展ディープラーニング　小川雄太郎 著
- Udemy講座【Hands Onで学ぶ】PyTorchによる深層学習入門　by Tetsuya T
- 最短突破 ディープラーニングG検定(ジェネラリスト) 問題集　高橋光太郎, 落合達也 著
- https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html

本日の課題　ノーマル

1. 質問, 感想, 要望をどうぞ

課題をノートブック(.ipynbファイル)にまとめて，Moodleにて提出すること
ファイル名は[回数，01~15]_[難易度，ノーマル nかハード h].ipynb.例 .14_n.ipynb