

# パターン認識と学習

## 深層学習(2)

管理工学科

篠沢佳久

# 資料の内容

- 深層学習(2)
  - 畳み込みニューラルネットワーク\*(2)
  - 主な畳み込みニューラルネットワーク
  - 畳み込みニューラルネットワークの応用
    - 物体検出(Detection)
    - Semantic Segmentation
    - 画像生成

---

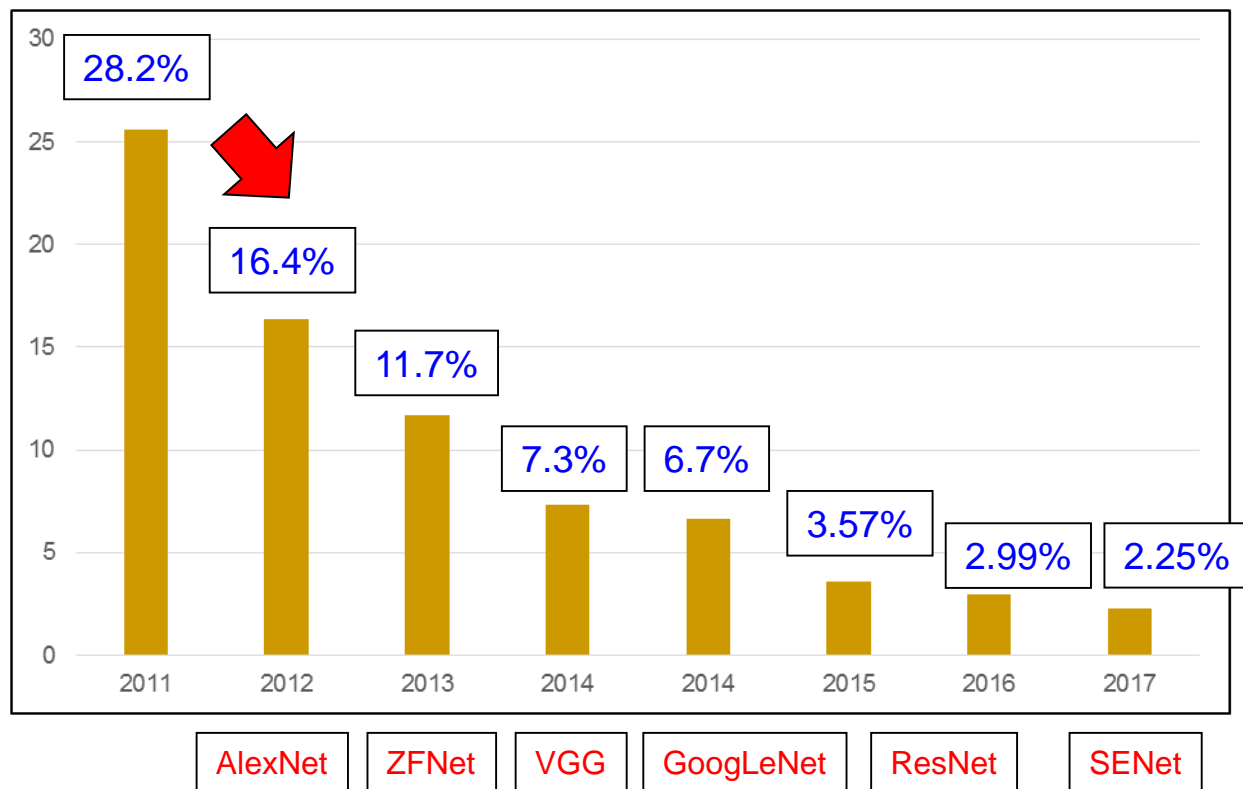
\*資料では畳み込みネットワークもしくはCNNと略して表記します

# 主な畳み込みニューラルネットワーク

ILSVRC

# 主な畳み込みニューラルネットワーク

## ■ ILSVRC\*におけるエラー率の向上



\*ILSVRC (ImageNet Large Scale Visual Recognition Challenge)

# Image Net

(<http://www.image-net.org/>)

クラス数 21,841  
画像数 14,197,122

**Ornamental**  
Any plant grown for its beauty or ornamental value

0 pictures 70.62% Popularity Percentile Wordnet IDs


ImageNet 2011 Fall Release (32326) ^

- plant, flora, plant life (4486)
  - phytoplankton (2)
  - microflora (0)
  - crop (9)
  - endemic (0)
  - holophyte (0)
  - non-flowering plant (0)
  - plantlet (0)
  - wilding (141)
  - ornamental (1)
    - flowering maple (0)
  - pot plant (0)
  - acrogen (0)
  - apomict (0)
  - aquatic (0)
  - cryptogam (1)
  - annual (0)
  - biennial (0)
  - perennial (1)
  - escape (0)
  - hygrophyte (0)
  - neophyte (0)
  - embryo (0)
  - monocarp, monocarpic plant, r
  - sporophyte (0)
  - gametophyte (2)
  - houseplant (12)
  - garden plant (1)
  - vascular plant, tracheophyte (4
  - poisonous plant (31)
  - air plant, epiphyte, aerophyte, e

Treemap Visualization Images of the Synset Downloads

ImageNet 2011 Fall Release > Plant, flora, plant life > Ornamental

Flowering



# AlexNet (A. Krizhevsky, 2012) ①

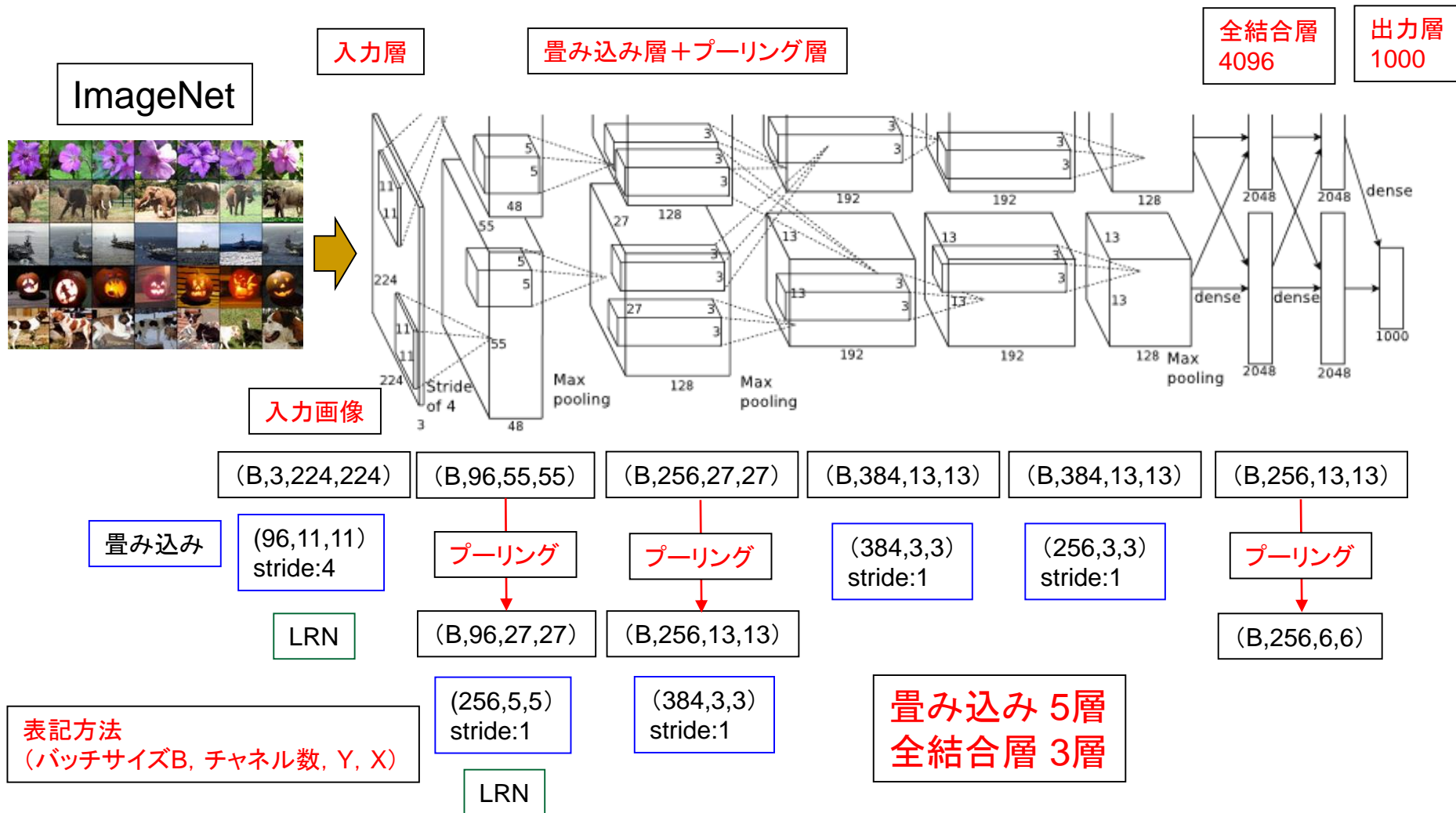


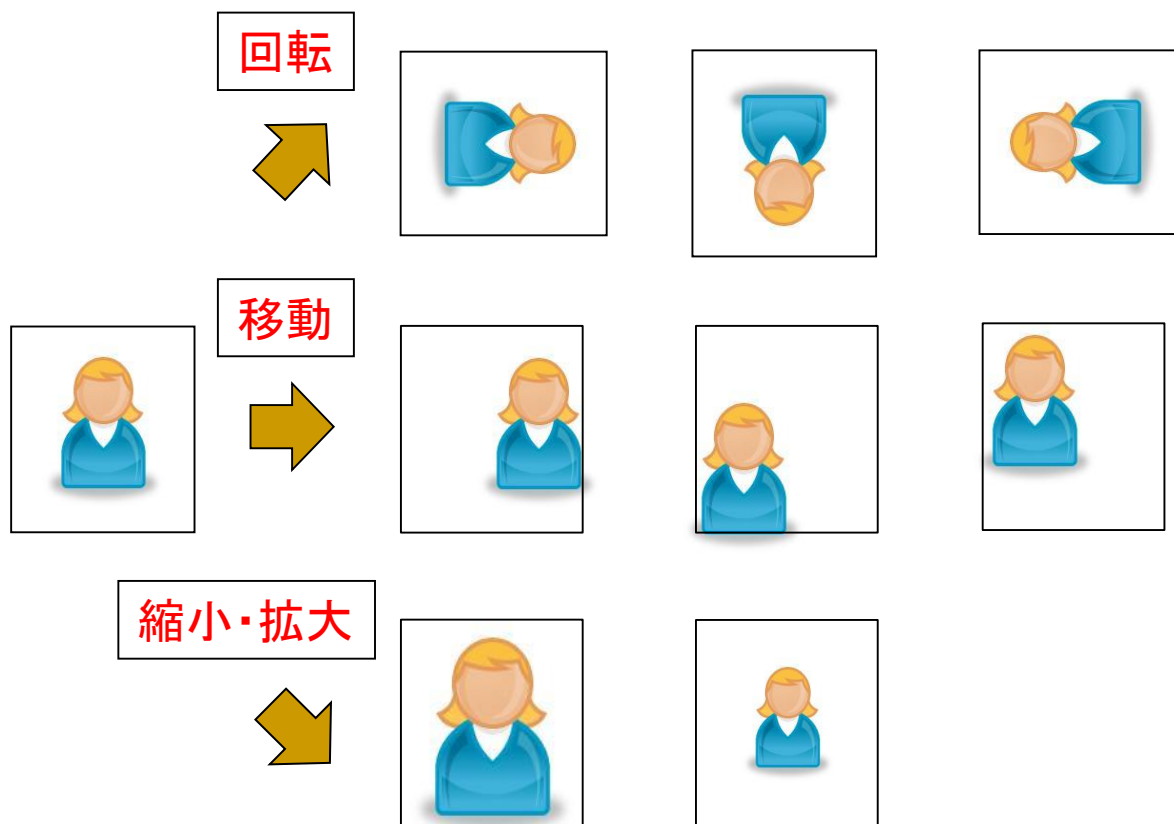
図: A. Krizhevsky, ImageNet Classification with Deep Convolutional Neural Networks, Advances in neural information processing systems, 2012

# AlexNetの工夫①

- ReLU関数
- データ拡張
  - $256 \times 256$ の大きさの画像から $224 \times 224$ の大きさの画像を切り出す
  - 水平反転
- ドロップアウト
- Overlapping Pooling
  - 領域をかぶらせてプーリングを行なう

# データ拡張

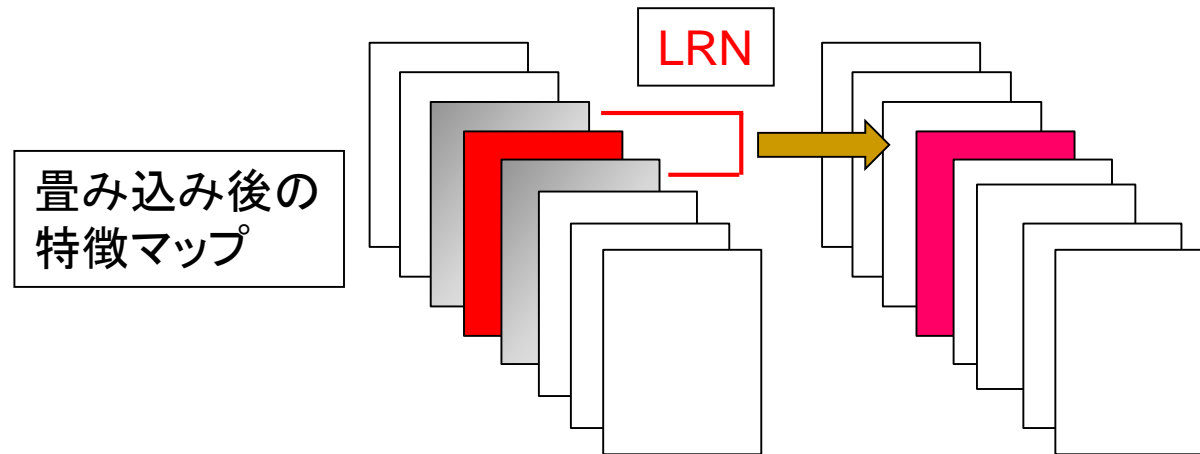
- 学習データに人工的な操作を施す
  - 移動, 回転, 拡大, 縮小



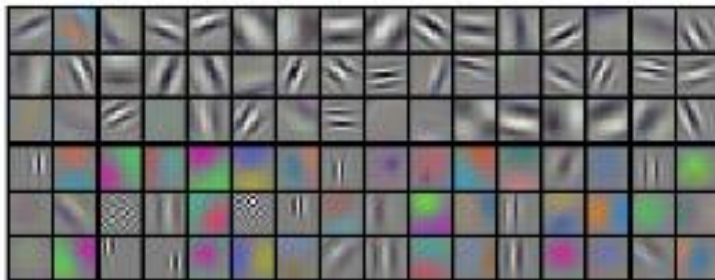


# AlexNetの工夫②

- 局所応答正規化 (Local Response Normalization, LRN)



- 学習後のフィルター

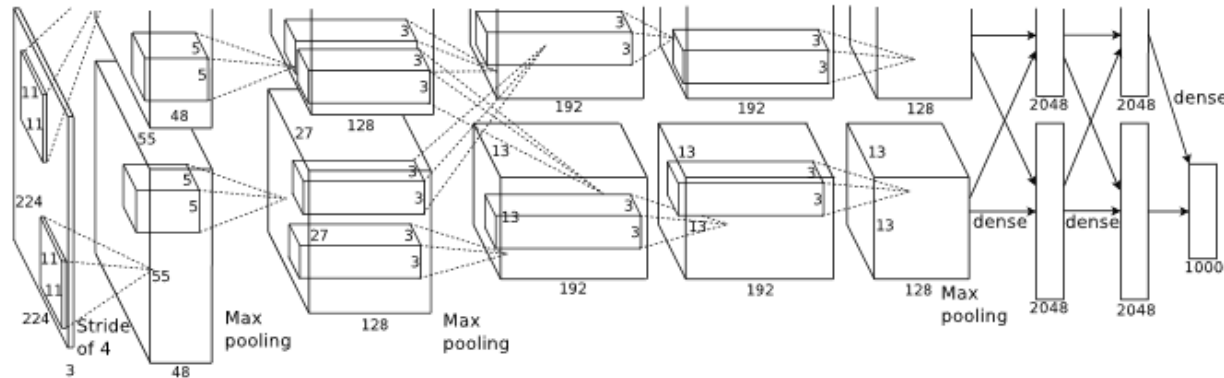


(11 × 11)  
3チャンネル  
96枚

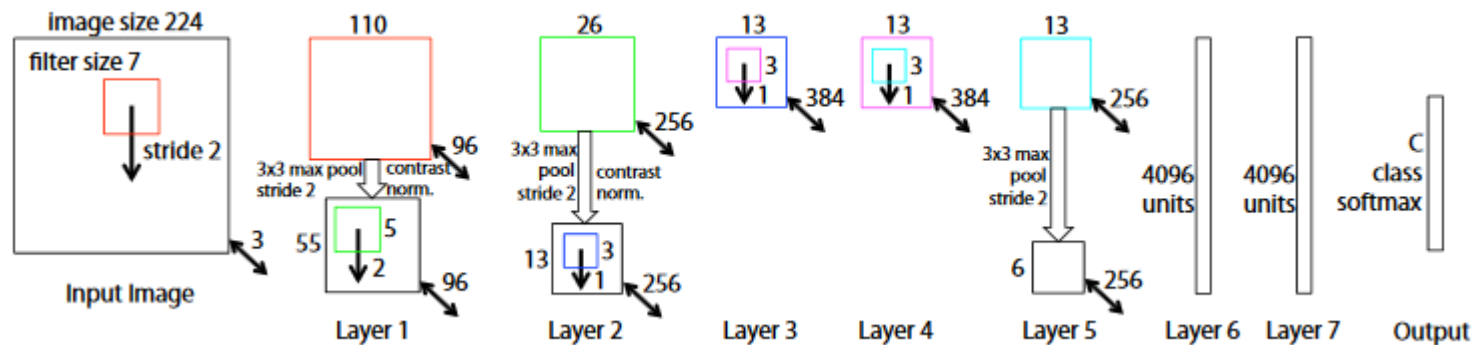
# ZFNet (Matthew D. Zeiler and Rob Fergus)

## ■ AlexNetの改良

AlexNet



ZFNet



# VGG (Visual Geometry Group) ①

表記方法

(バッチサイズB, チャンネル数, Y, X)

VGG19

(19層)

(B,3,224,224)

入力層

maxプーリング  
2×2, スライド2

畳み込み処理後の活性化関数  
ReLU

畳み込み層

畳み込み層

プーリング層

畳み込み層

畳み込み層

プーリング層

畳み込み層

畳み込み層

畳み込み層

畳み込み層

プーリング層

畳み込み層

畳み込み層

畳み込み層

畳み込み層

プーリング層

(64,3,3)  
stride:1

(128,3,3)  
stride:1

(256,3,3)  
stride:1

(512,3,3)  
stride:1

(B,64,224,224)

(B,128,112,112)

(B,256,56,56)

(B,512,28,28)

(B,64,112,112)

(B,128,56,56)

(B,256,28,28)

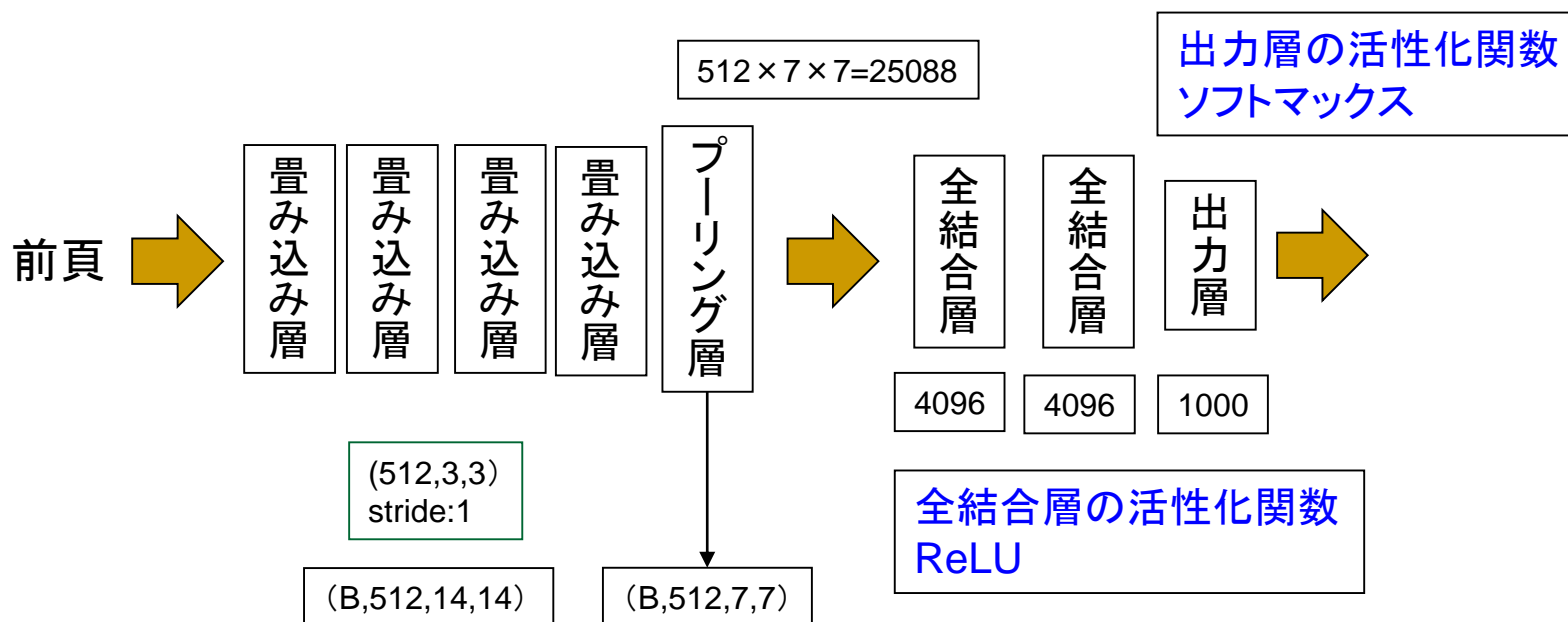
(B,512,14,14)

次頁

# VGG (Visual Geometry Group) ②

表記方法

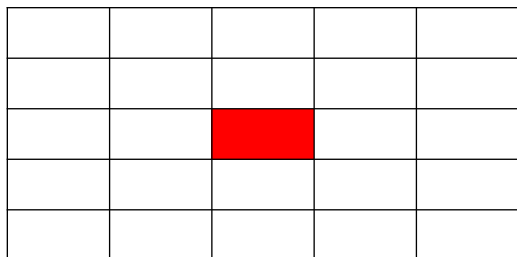
(バッチサイズB, チャンネル数, Y, X)



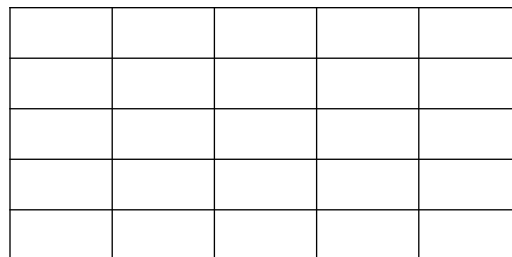
\*ILSVRC2014の時は16層 (VGG16)

# VGGの工夫①

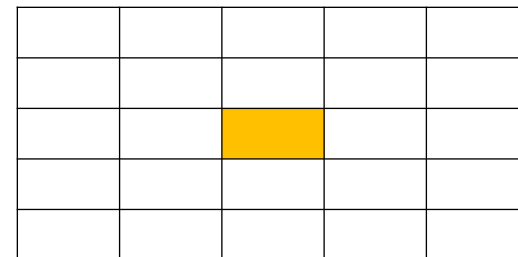
画像



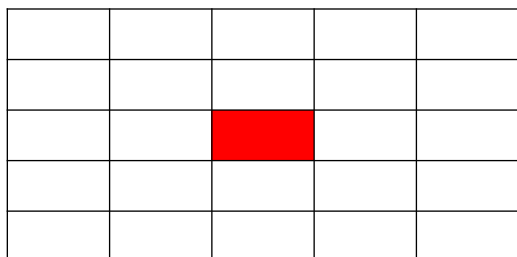
フィルター(5×5)



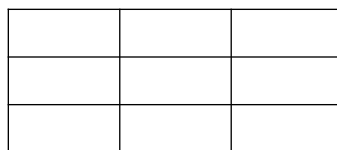
畳み込み



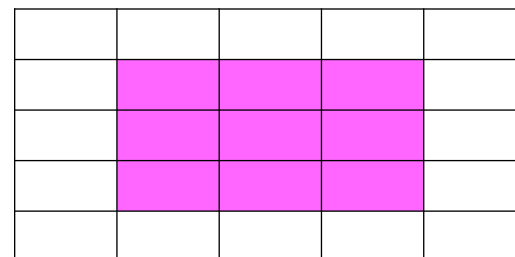
画像



フィルター(3×3)

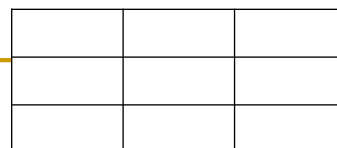
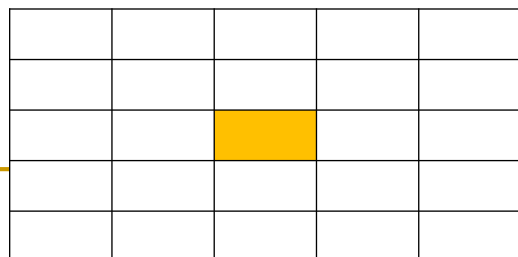


畳み込み



畳み込み

フィルター(3×3)



# VGGの工夫②

VGG-A(11層)  
から多層化

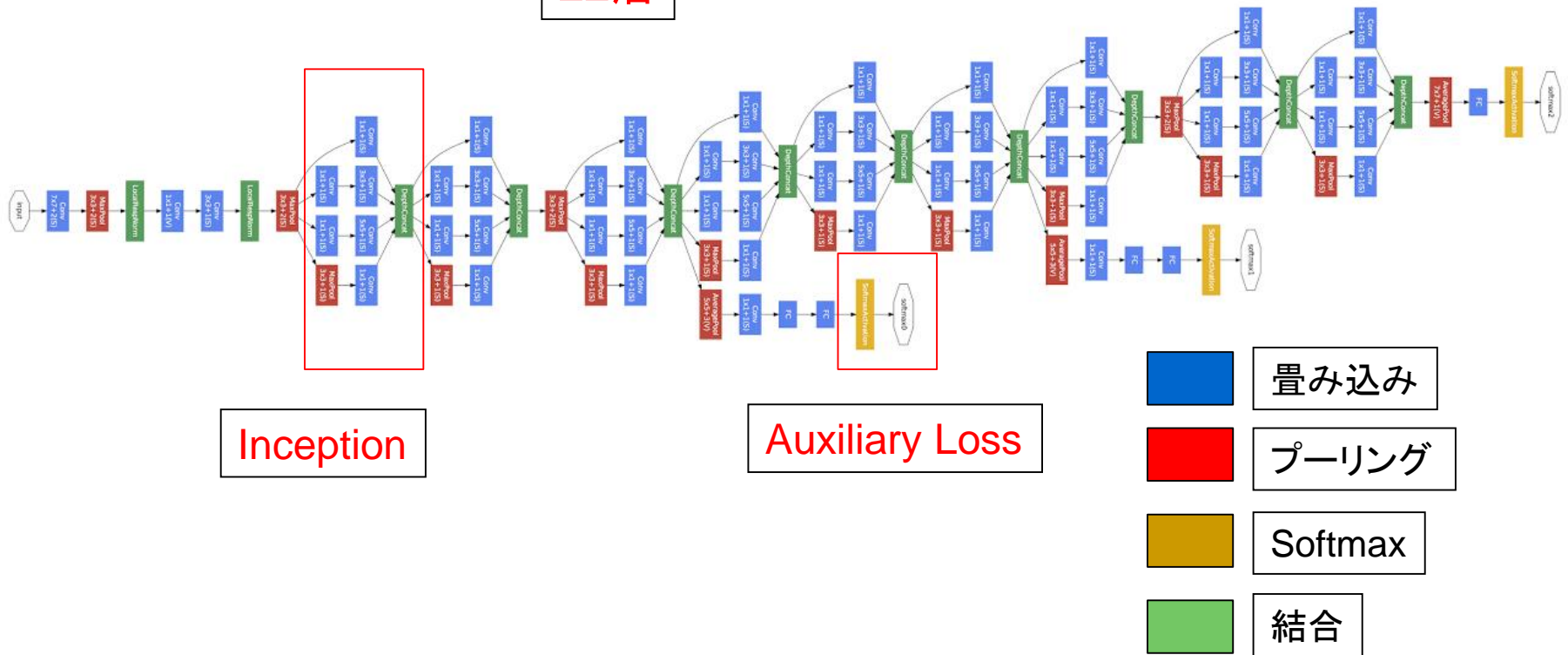
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 LRN	conv3-64 <b>conv3-64</b>	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 <b>conv3-128</b>	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 <b>conv1-256</b>	conv3-256 conv3-256 <b>conv3-256</b>	conv3-256 conv3-256 conv3-256 <b>conv3-256</b>
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 <b>conv1-512</b>	conv3-512 conv3-512 <b>conv3-512</b>	conv3-512 conv3-512 conv3-512 <b>conv3-512</b>
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 <b>conv1-512</b>	conv3-512 conv3-512 <b>conv3-512</b>	conv3-512 conv3-512 conv3-512 <b>conv3-512</b>
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

表: K.Simonyan, A.Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, 2015

# GoogLeNet (C. Szegedy, 2014)

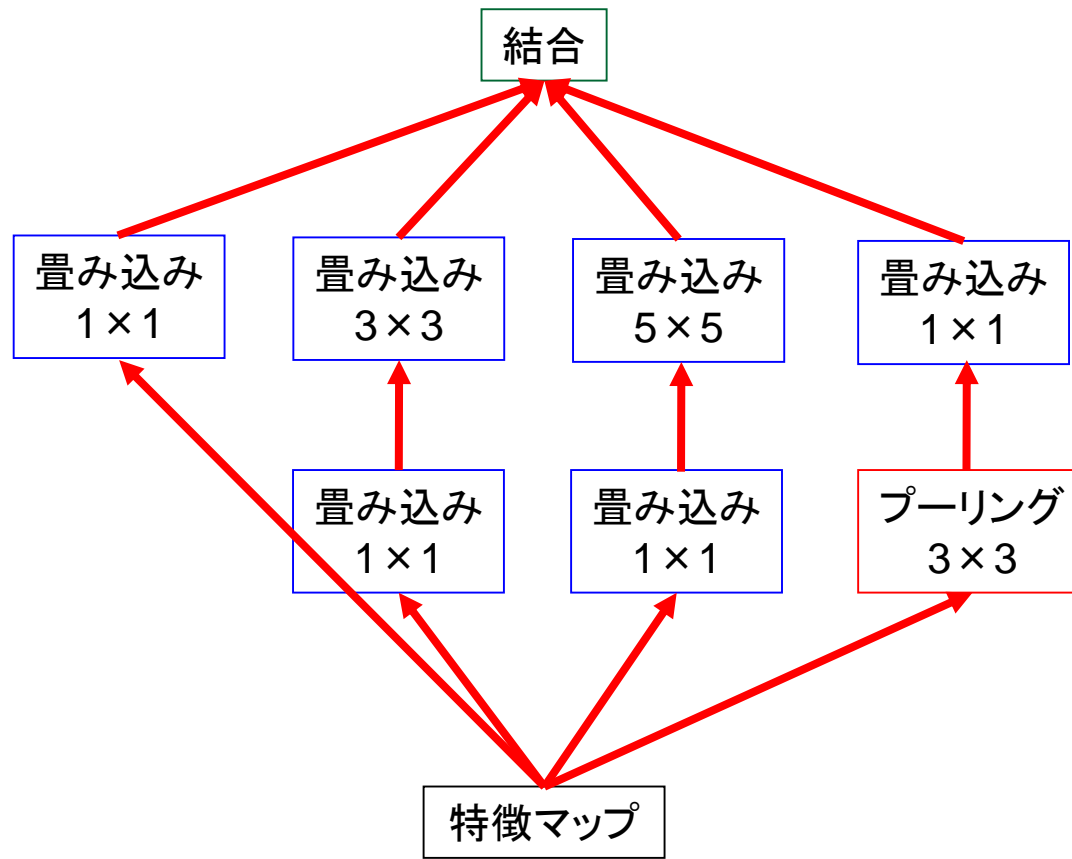
## ■ GooLeNet (Inception-v3)

22層



# GoogLeNetの工夫①

## ■ Inception



特徴マップ  
(B, C, Y, X)

フィルター  
(M, 1, 1)



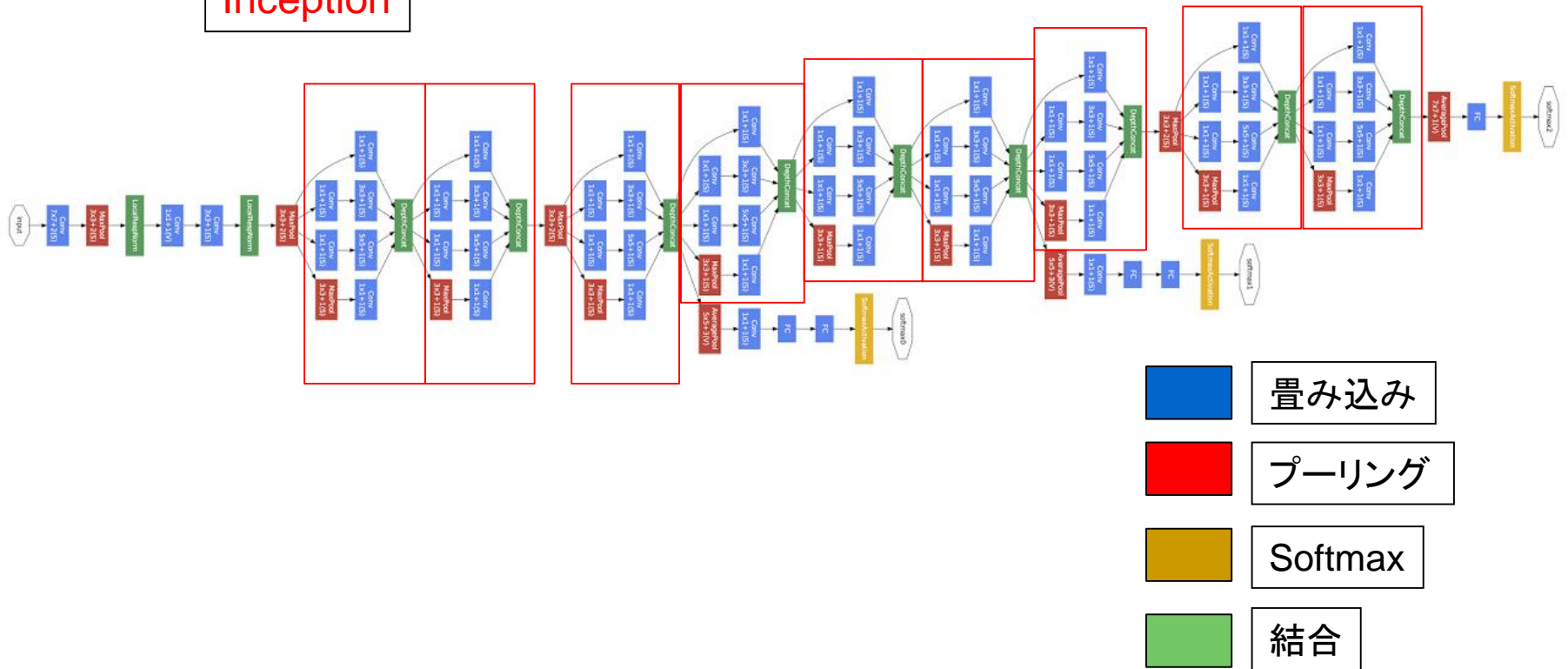
新たな特徴マップ  
(B, M, Y, X)

**C > M の場合**  
特徴マップを削減



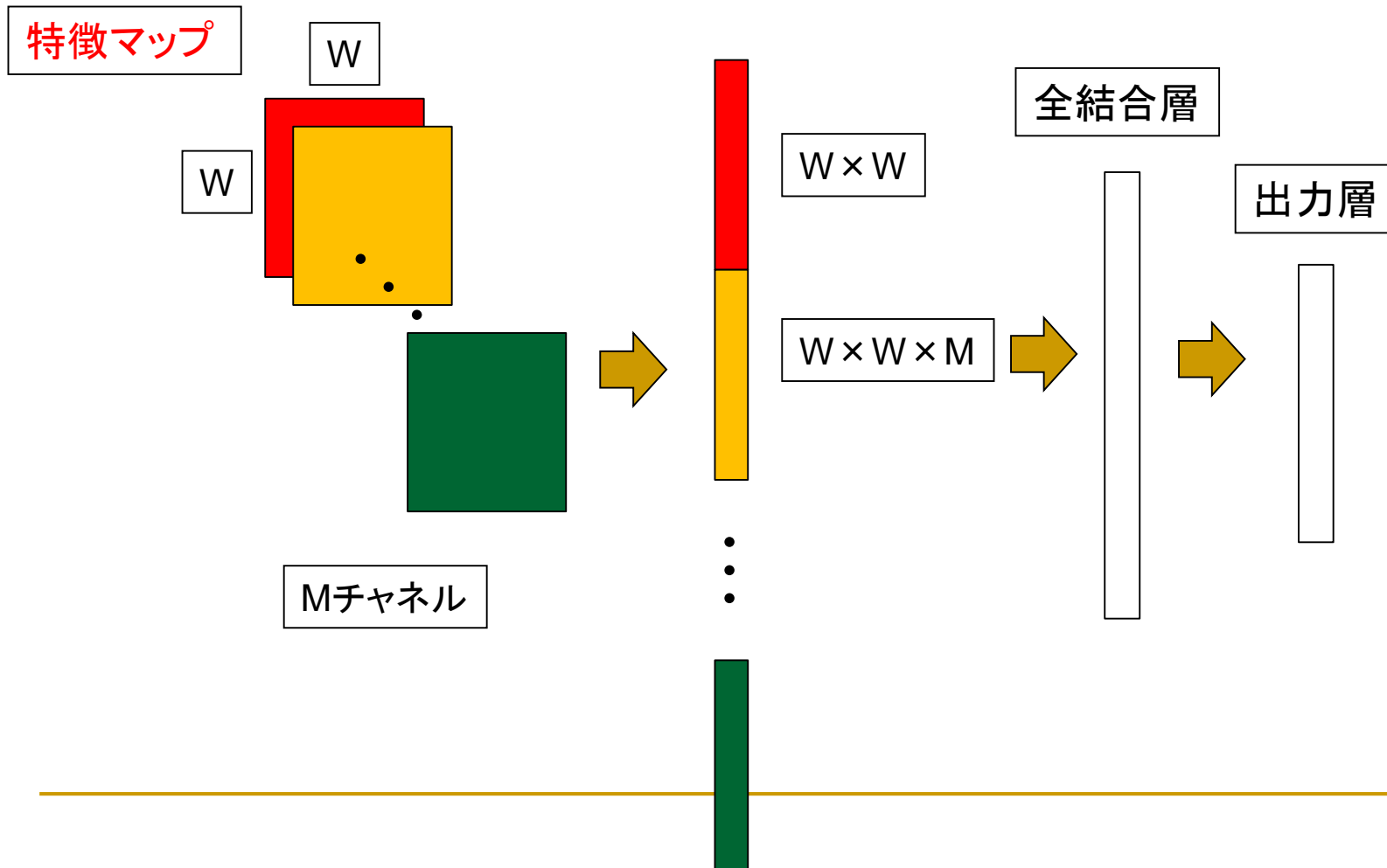
# GoogLeNetの工夫②

## Inception



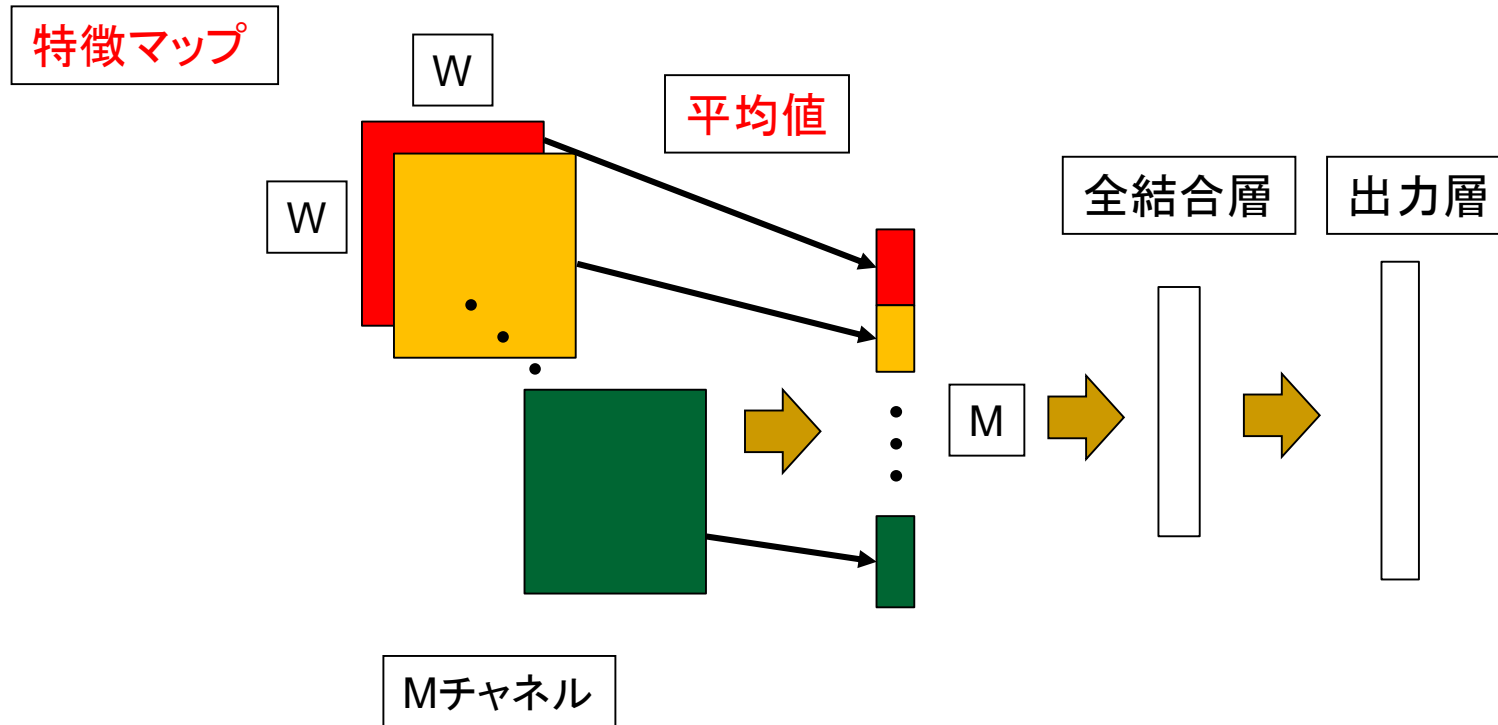
# GoogLeNetの工夫④

## ■ 全結合層



# GoogLeNetの工夫⑤

## ■ Global Average Pooling (GAP)



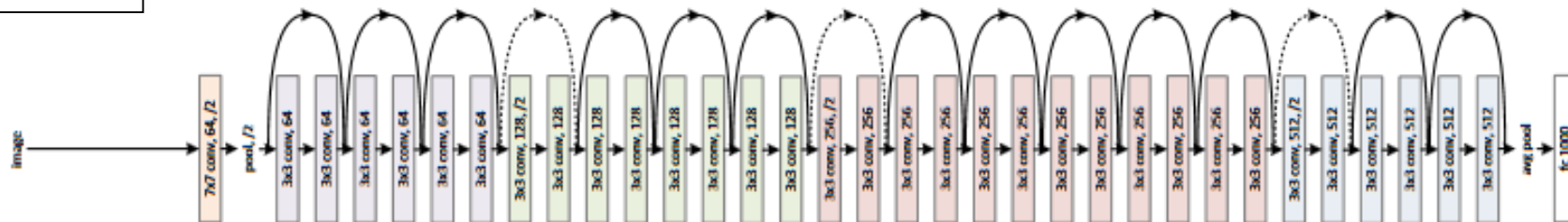
# ResNet (Microsoft Research Asia, 2015)

ResNet

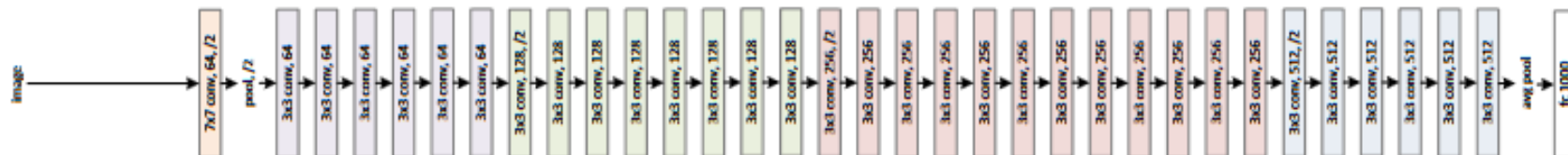
shortcut

34層  
(ILSVRC2015では152層)

34-layer residual



34-layer plain



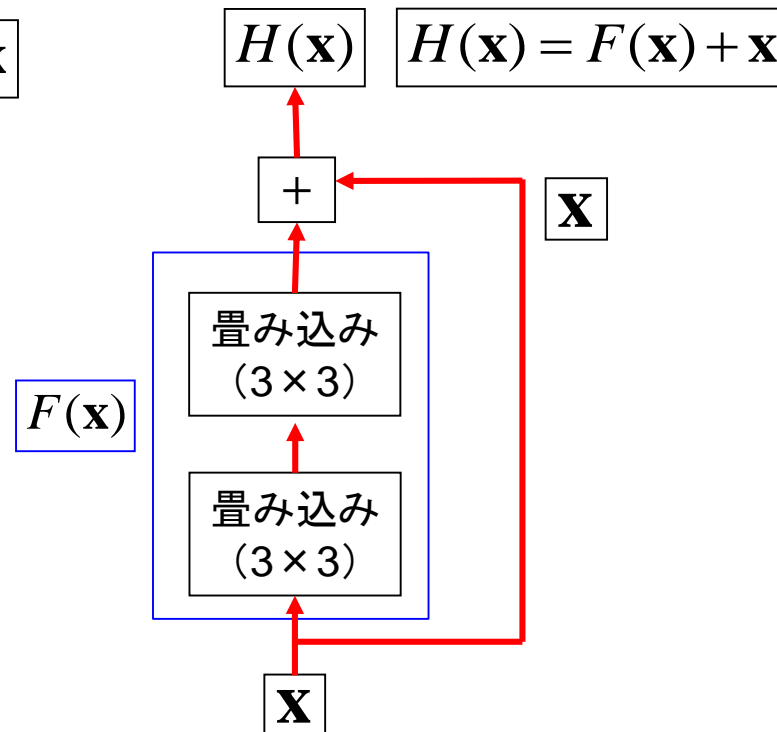
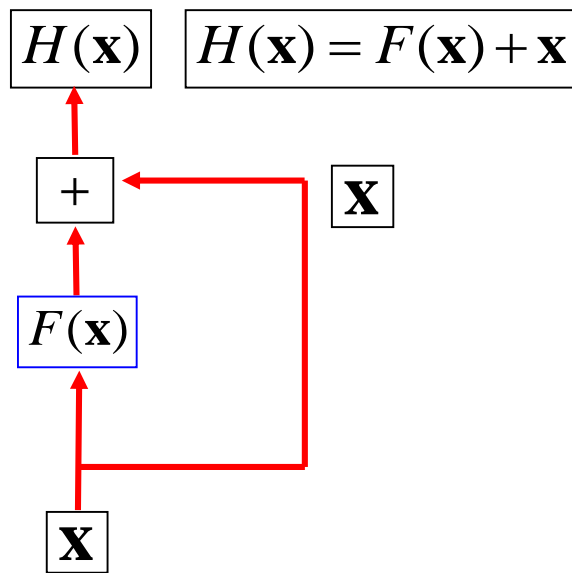
VGG-19



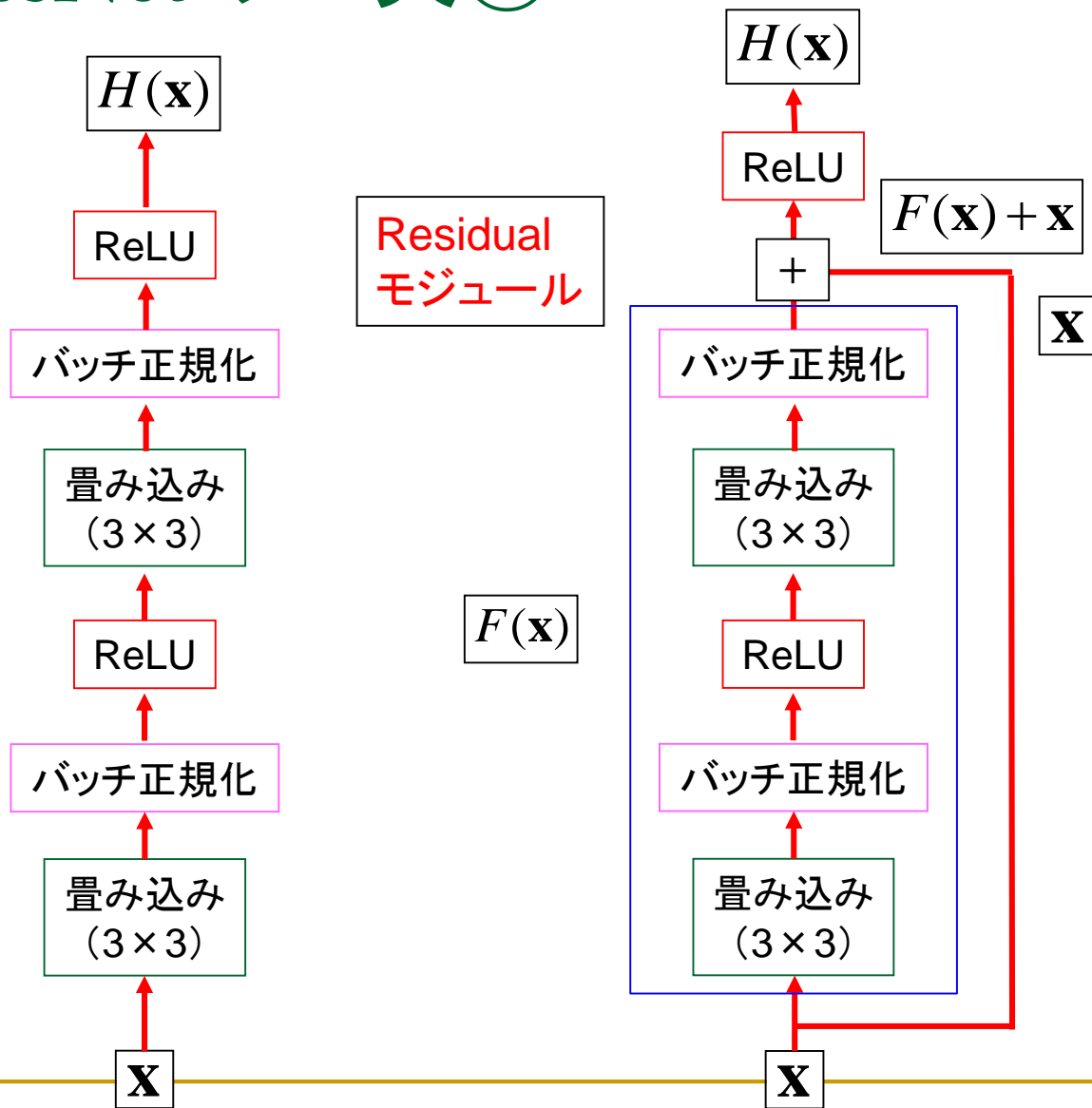
☒ : K.He, Deep Residual Learning for Image Recognition, 2015

# ResNetの工夫①

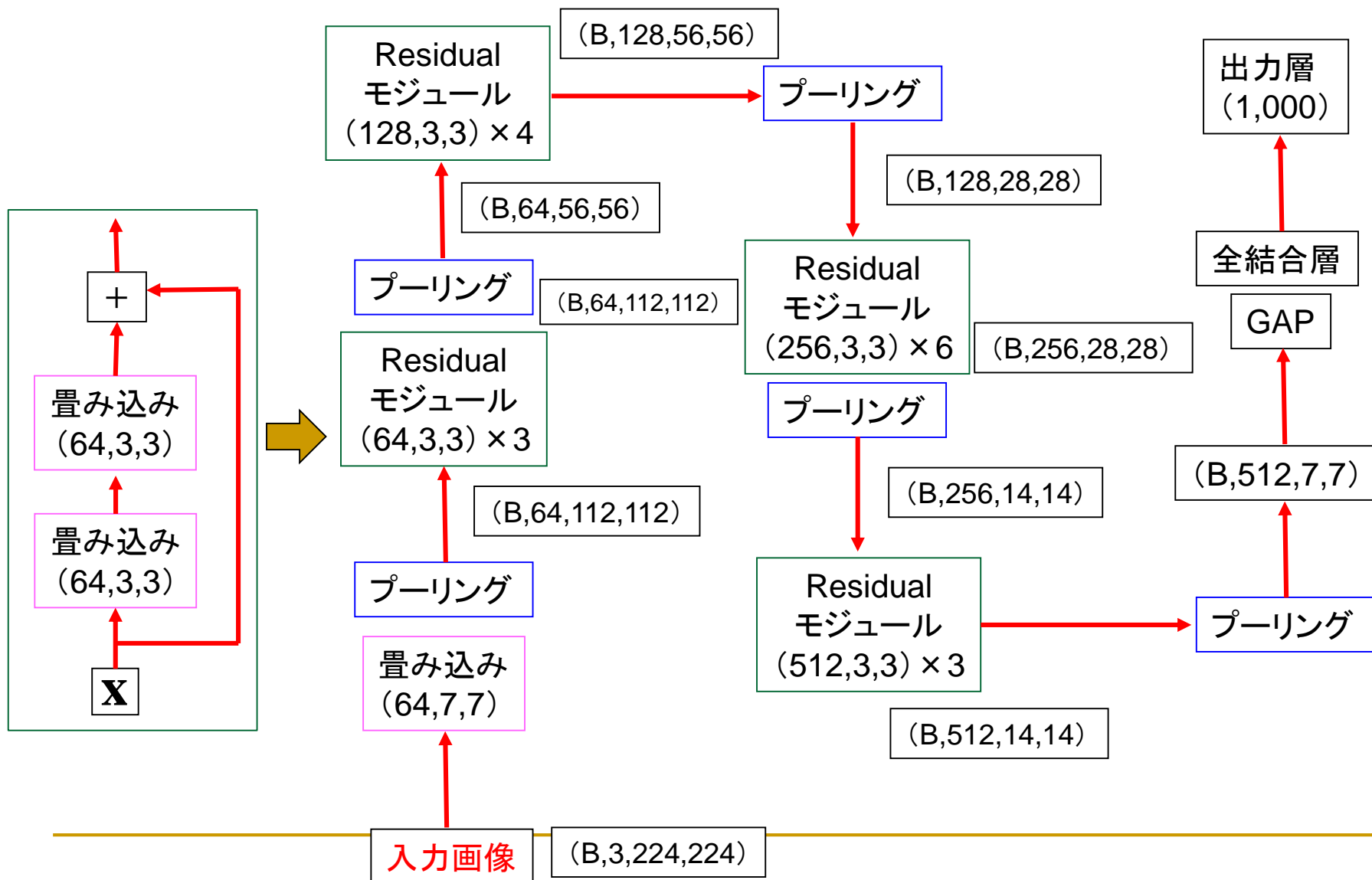
Residual  
モジュール



## ResNetの工夫②



# ResNet(34層)



# Batch Normalization

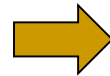
ミニバッチ

$$B = \{x_1, x_2, \dots, x_m\}$$

平均

$$\mu_B = \frac{1}{m} \sum_{i=1}^m x_i$$

正規化



$$\hat{x}_i \leftarrow \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$$

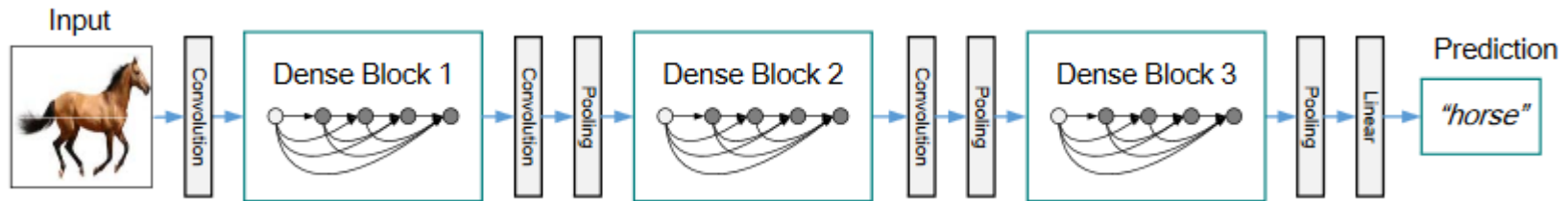
標準偏差

$$\sigma_B = \frac{1}{m} \sum_{i=1}^m (x_i - \mu_B)^2$$

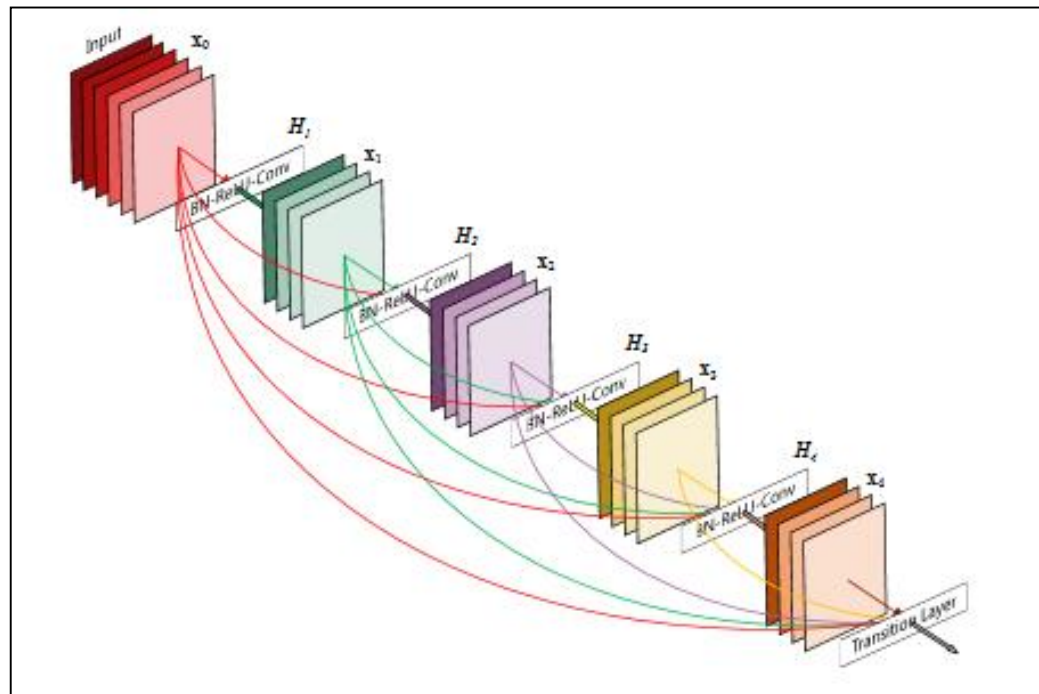
$$y_i \leftarrow \gamma \hat{x}_i + \beta$$



# Densely Connected Convolutional Networks (DenseNet)



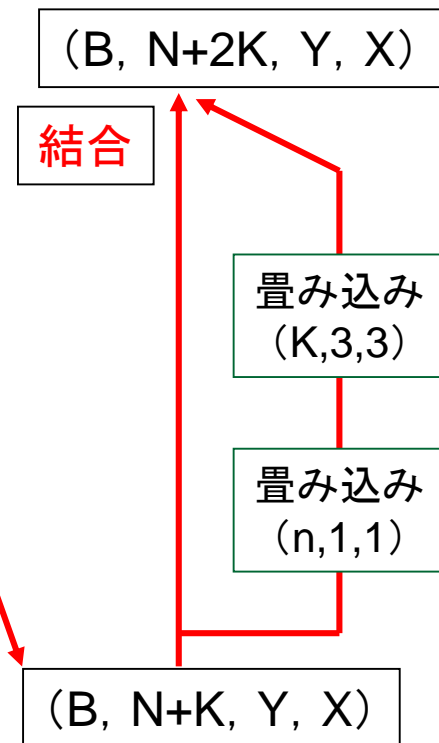
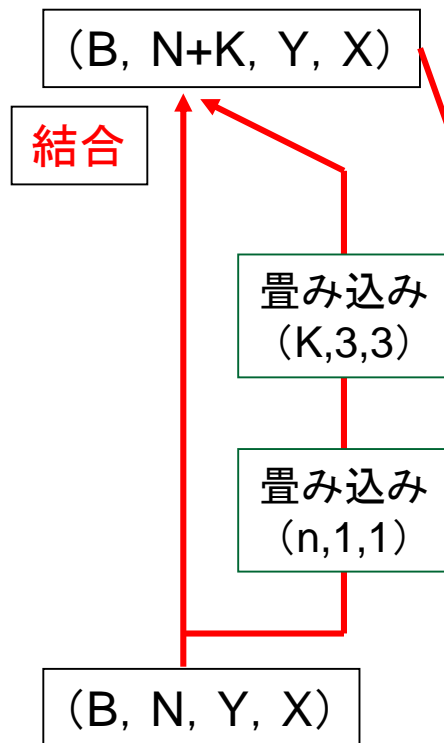
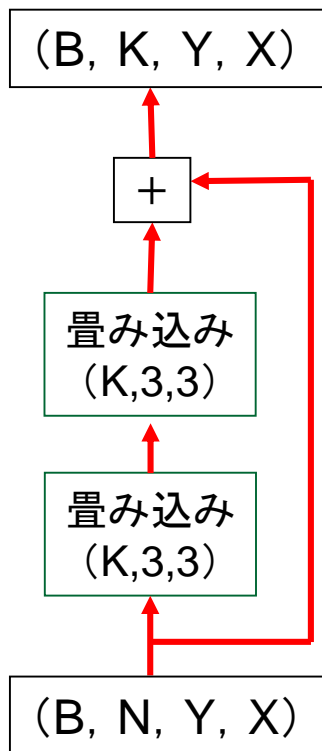
## Dense Block



# Denseブロック

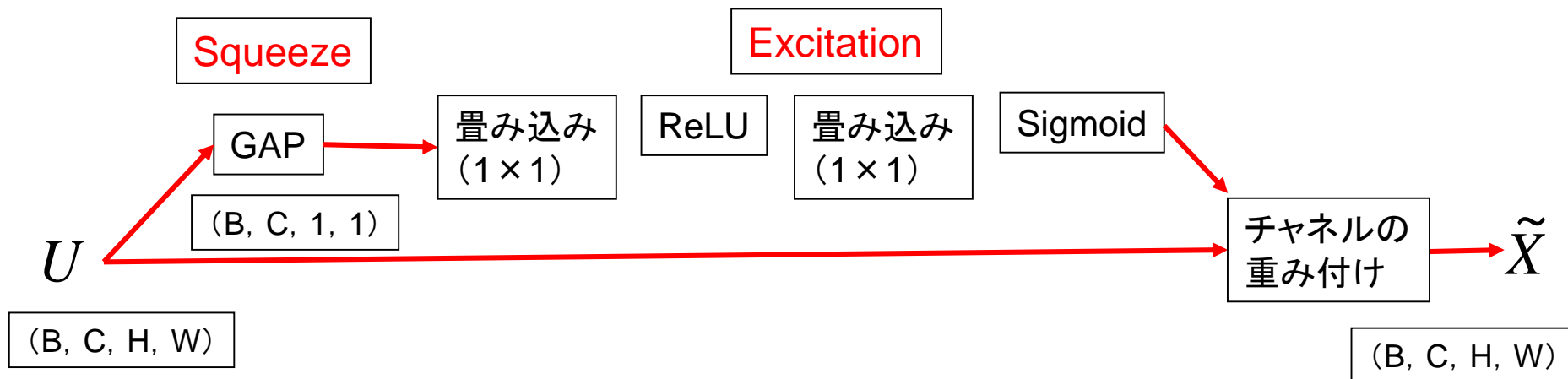
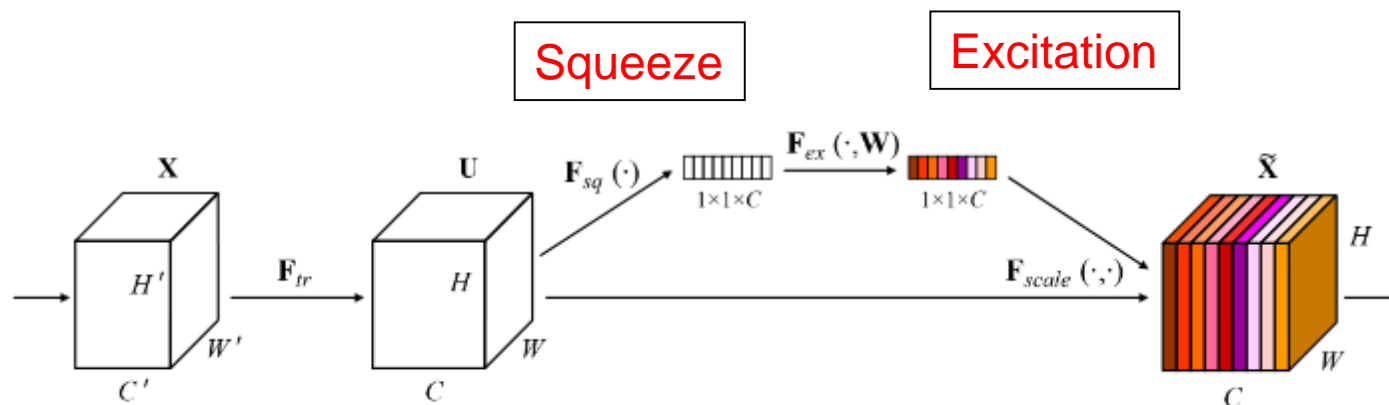
Residual  
モジュール

Dense  
ブロック



# SEnets (Squeeze-and-Excitation Networks)

## SE (Squeeze-and-Excitation) ブロック



# CNNを利用する際に考えるべきこと

- アーキテクチャー
- ハイパーパラメータ
- 計算時間の削減

# 畳み込みネットワークの学習①

- 設定すべきネットワークの構造に関する手法, パラメータ
  - アーキテクチャ, 層の構成, 層の総数
  - 畳み込み層
    - フィルターのサイズ, フィルター数, ストライド, パディング
  - プーリング層
    - 種類(最大プーリング, 平均プーリング, Lpプーリング)
    - サイズ, ストライド, パディング
  - 全結合層
    - ニューロン数
  - 活性化関数
    - シグモイド関数, ReLU関数, 恒等関数

# 畳み込みネットワークの学習②

- 学習に関する手法, パラメータ
  - 学習方法
    - モーメント法, AdaGrad, RMSProp, AdaDelta, Adam
  - ミニバッチの大きさ
    - SGD, ミニバッチ, バッチ
  - 学習係数
  - エポック数, 早期停止
  - 結合係数の初期化
    - LeCunの初期化, Xavierの初期化
  - 正則化
  - ドロップアウト

# 畳み込みネットワークの学習③

- データに関する手法, パラメータ
  - データの正規化
    - 標準化, 無相関化, 白色化
  - データ拡張

# 畳み込みネットワークの応用



# 畳み込みネットワークの応用

- 物体検出 (Detection)
  - R-CNN (Regions with CNN feature)
  - YOLO (You Only Look Once)
- Semantic Segmentation
  - SegNet
  - U-Net
- 画像生成
  - 敵対的生成ネットワーク (Generative Adversarial Network)

# 物体認識①

Categorization  
Classification  
(分類)



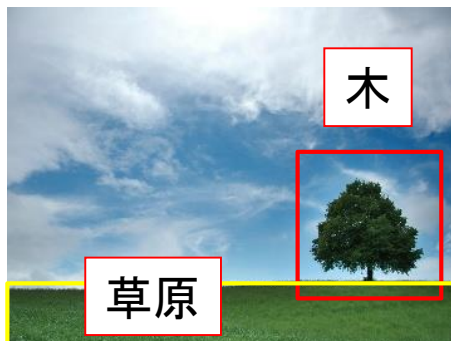
木

草原

空

雲

Detection  
(物体検出)



木

草原

Semantic Segmentation



木, 草原



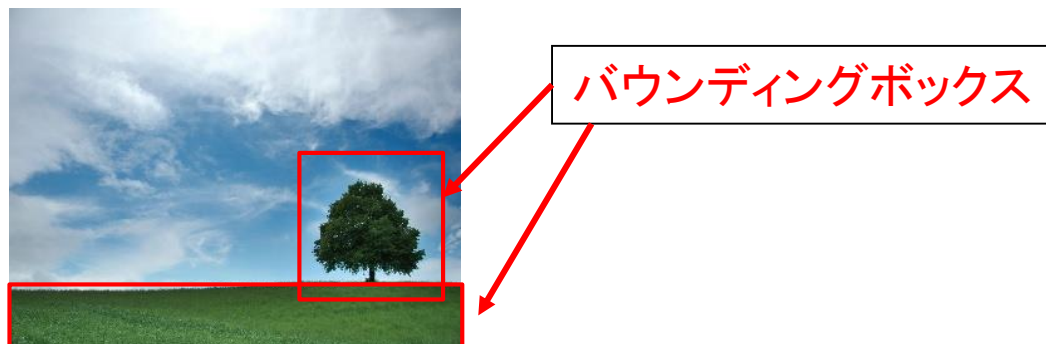
空



雲

# 物体認識②

① 物体領域の候補はどこに存在するのか  
(物体領域の候補の検出)

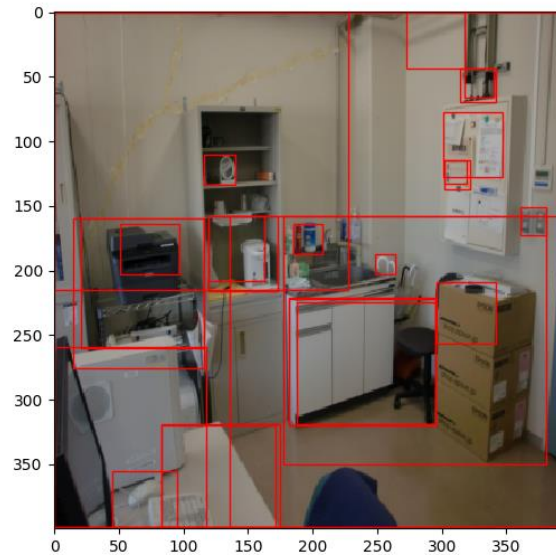


② 物体領域には何が存在するのか(クラス分類)

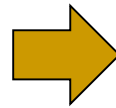
- ①②の処理を同時に行なう

# 物体認識③

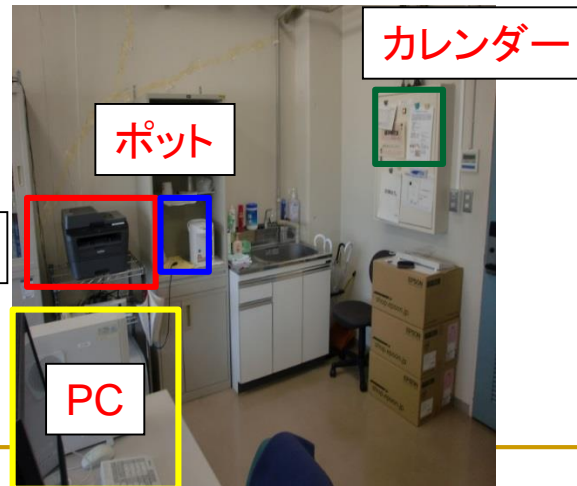
## ① 物体領域の候補の検出



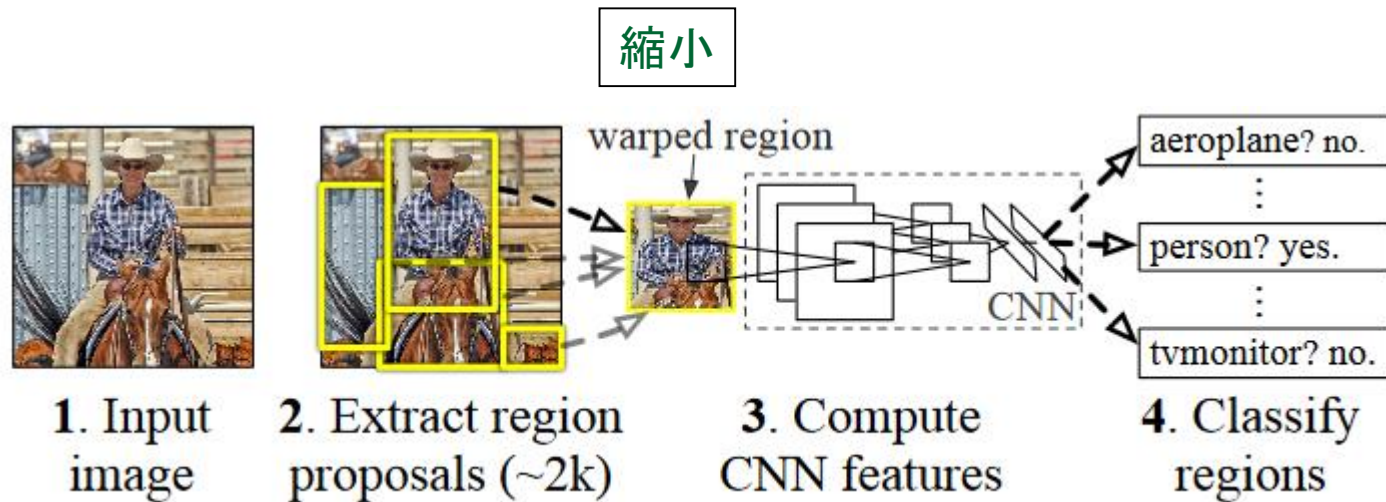
## ② クラス分類



プリンター

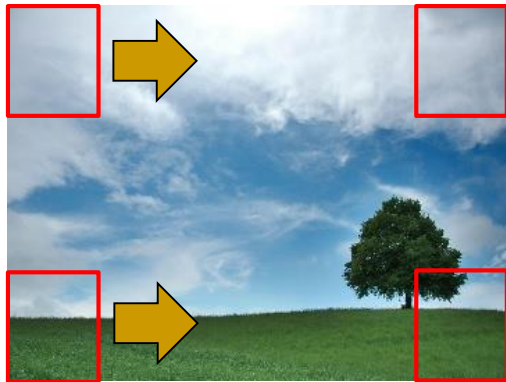


# R-CNN (Regions with CNN feature)



# 物体領域の候補の検出

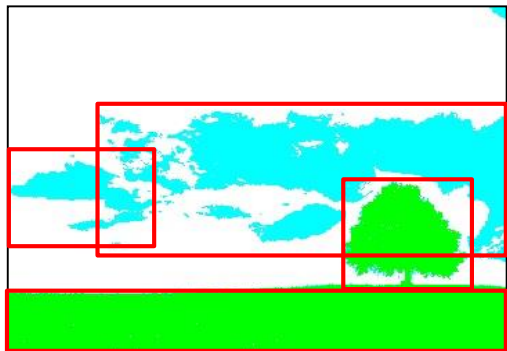
## ■ スライディングウィンドウ法



ウィンドウ  
(物体領域の候補)

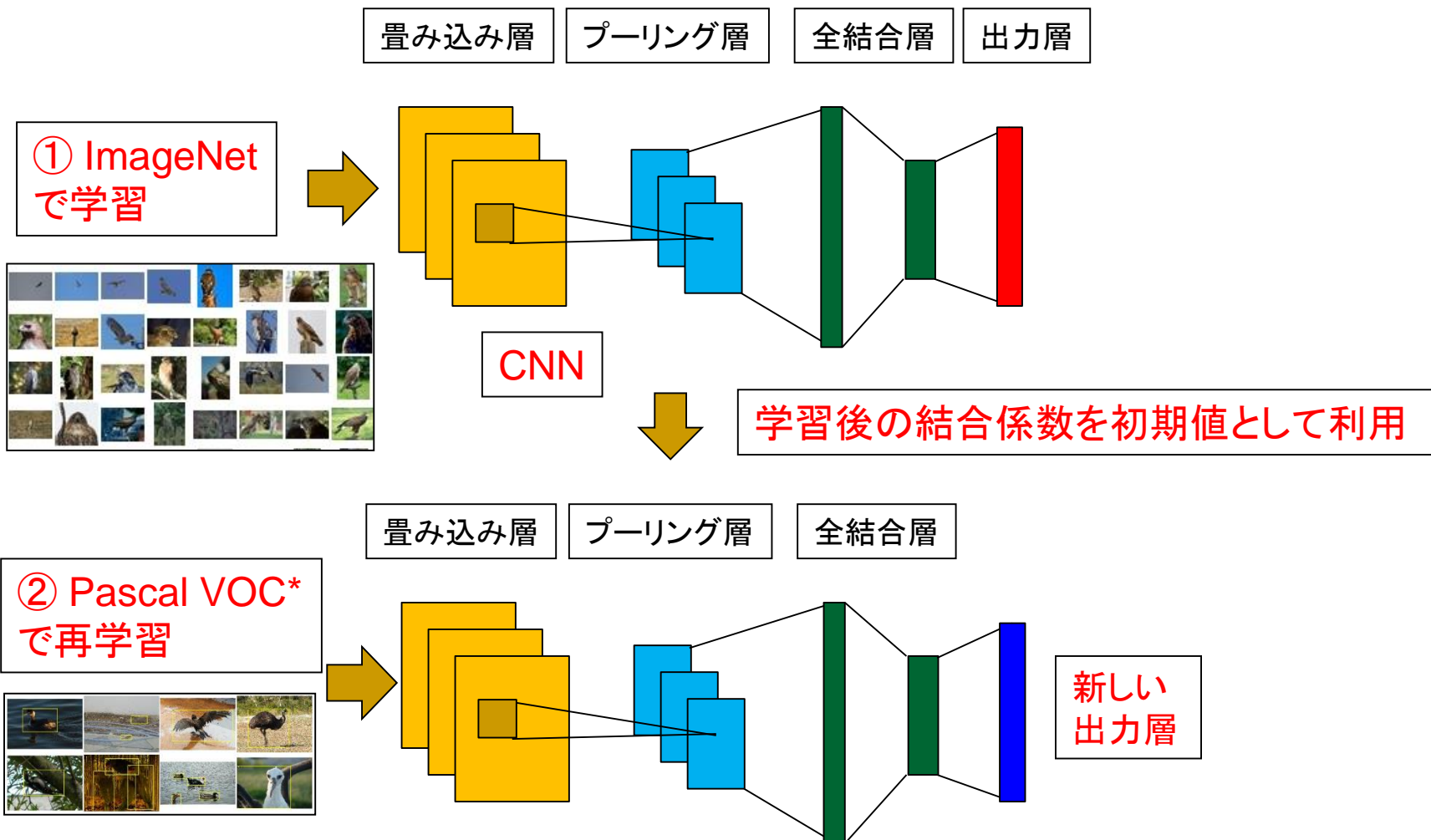
一定の大きさのウィンドウ  
をずらしながら、ウィンドウ  
中の物体を認識

## ■ 選択的検索法 (Selective Search)



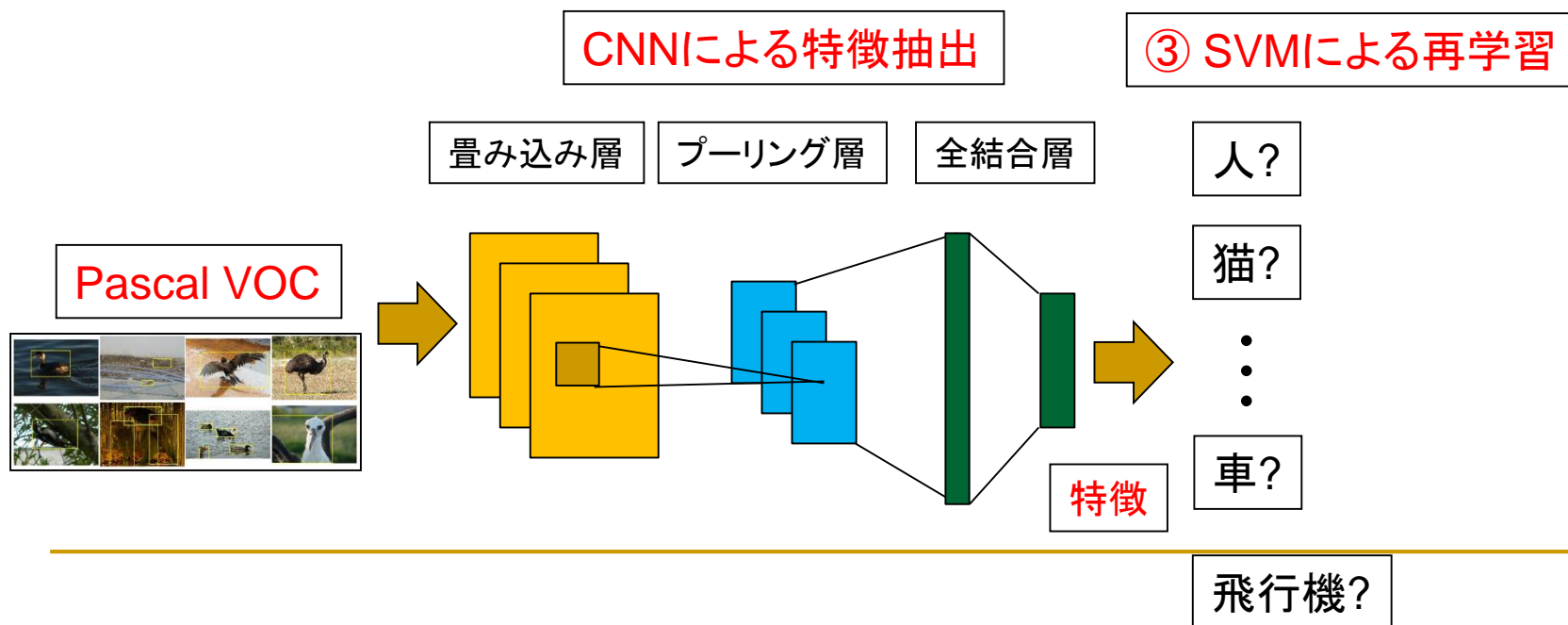
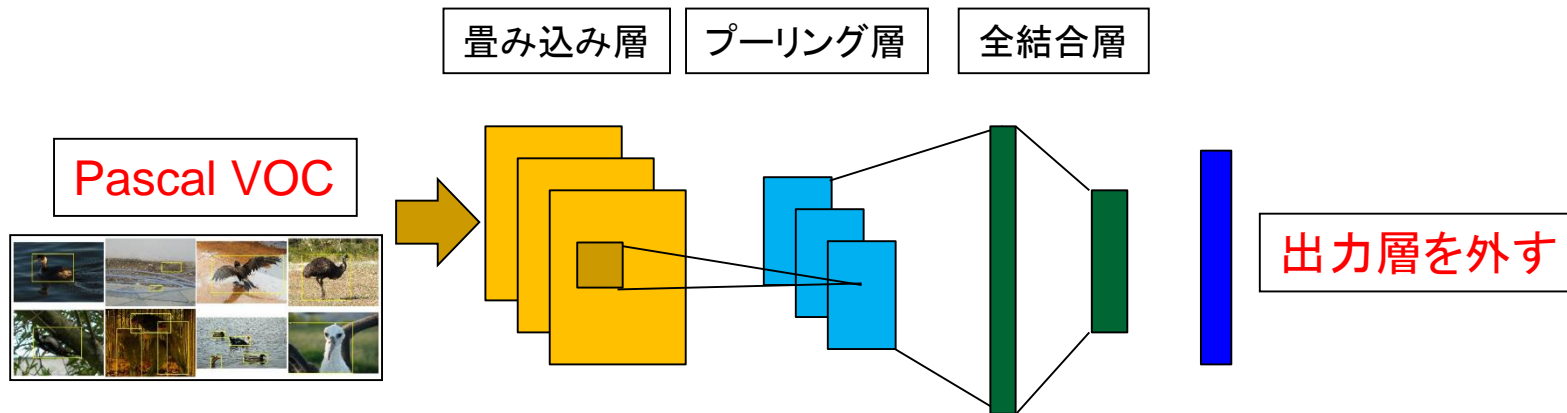
特徴(例えば色)より、同じ領域  
と考えられる候補を検出し、認識

# 特徴抽出(転移学習)



\*<http://host.robots.ox.ac.uk/pascal/VOC/>

# SVMによる再学習



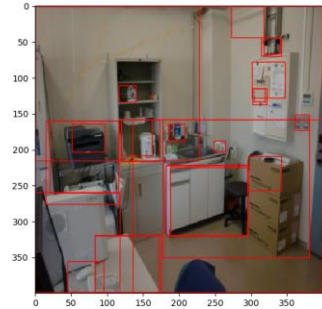


# R-CNNによる物体検出

入力画像



Selective Search  
による候補の検出



CNNによる特徴抽出

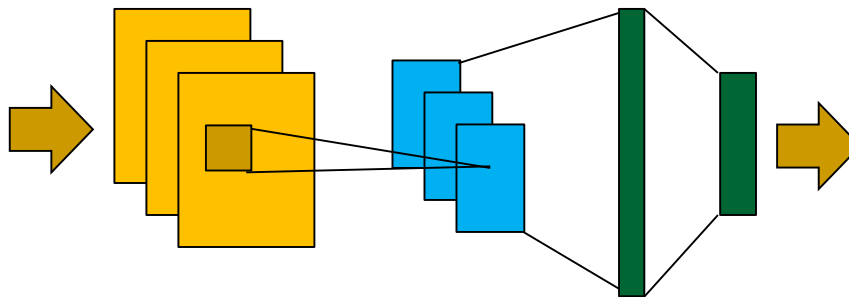


畳み込み層

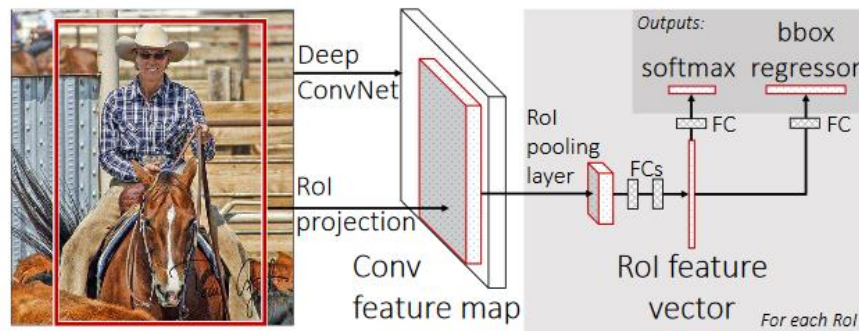
プーリング層

全結合層

SVMによるクラス分類



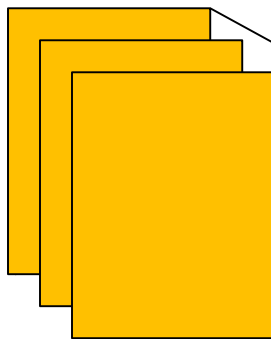
# Fast R-CNN



入力画像



CNN



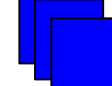
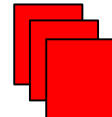
特徴マップ



Selective Search

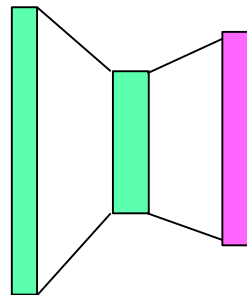


RoI (Region of Interest)  
プーリング



全結合層

出力層



クラス分類

バウンディング  
ボックスの位置  
(x,y,w,h)

# Faster R-CNN

入力画像



CNN

全結合層

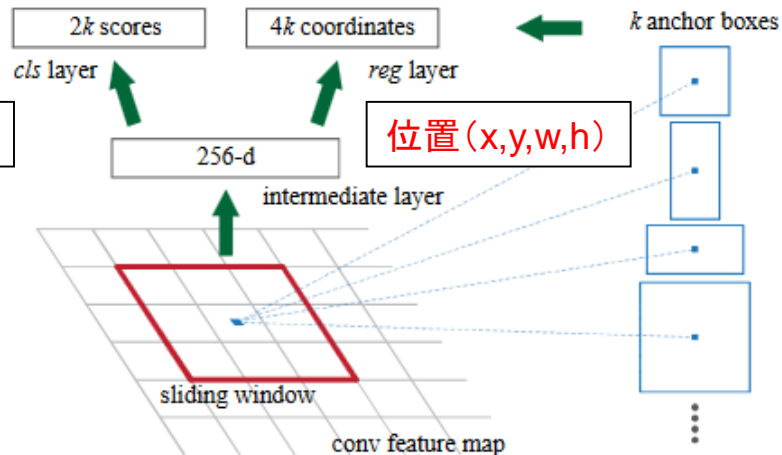
出力層

クラス分類

バウンディング  
ボックスの位置  
( $x, y, w, h$ )

Region Proposal Network

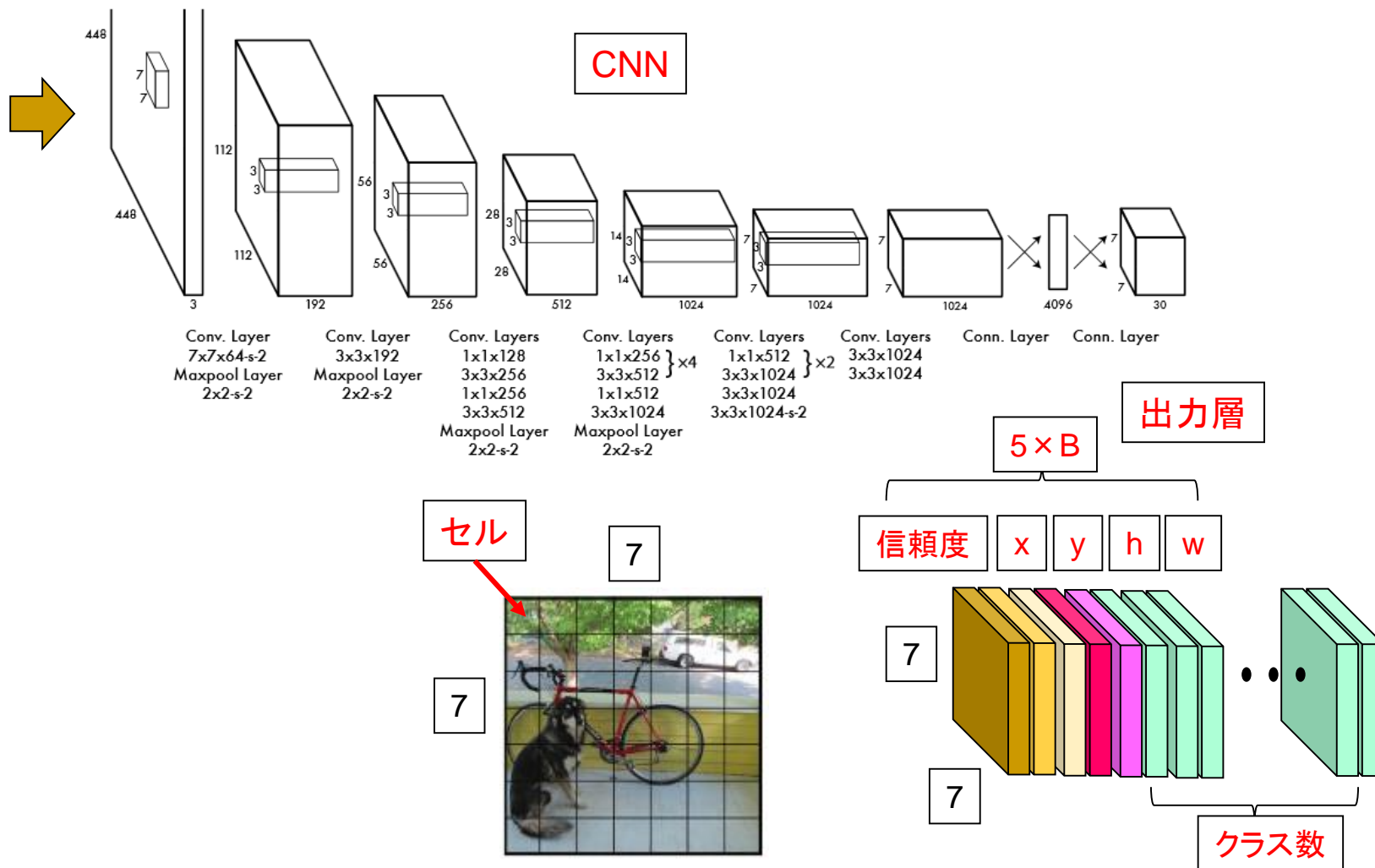
物体か



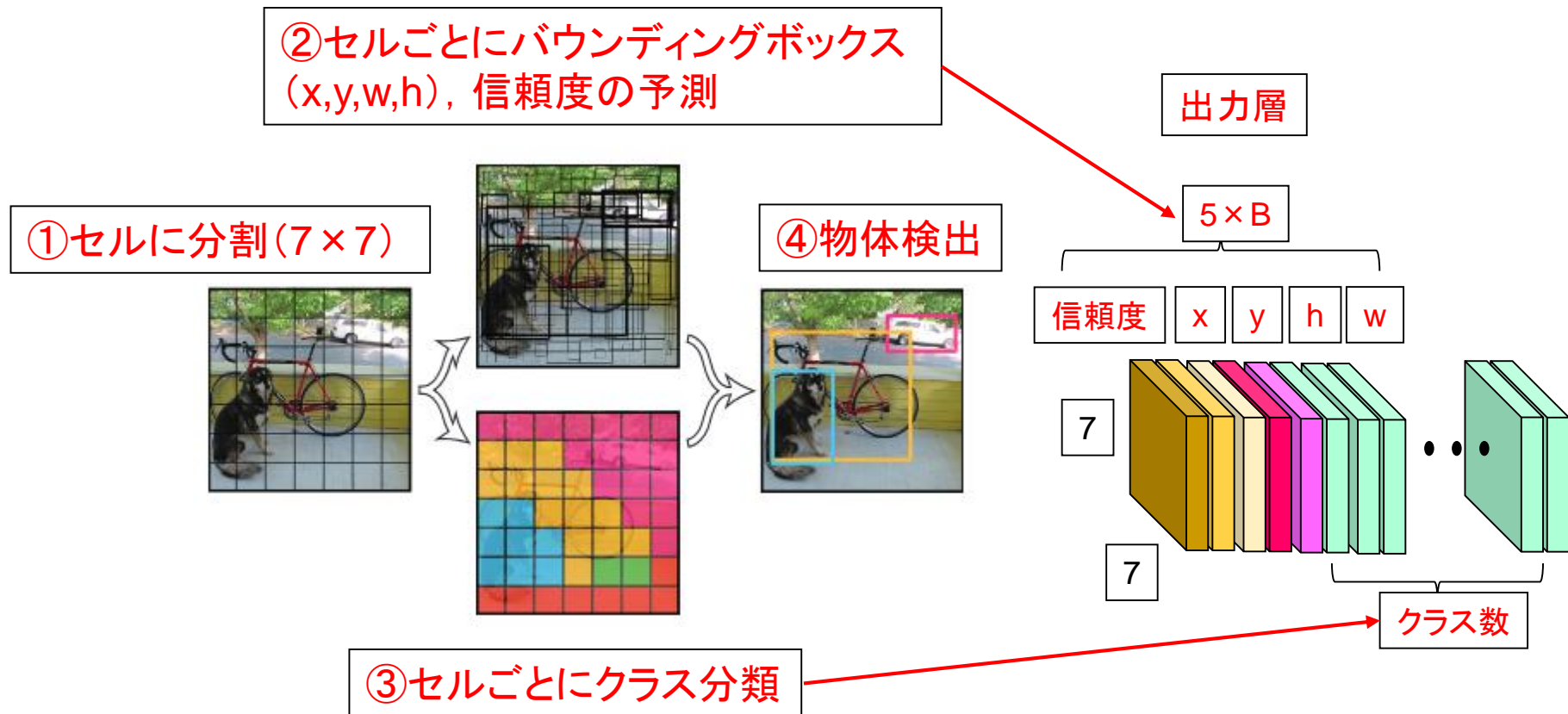
RoI (Region of Interest)  
プーリング

k個のアンカー

# YOLO (You Only Look Once)



# YOLO (You Only Look Once)

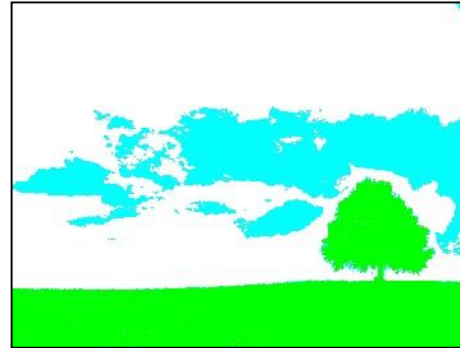
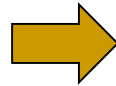


# Semantic Segmentation

## Semantic Segmentation



入力画像



出力画像



木, 草原



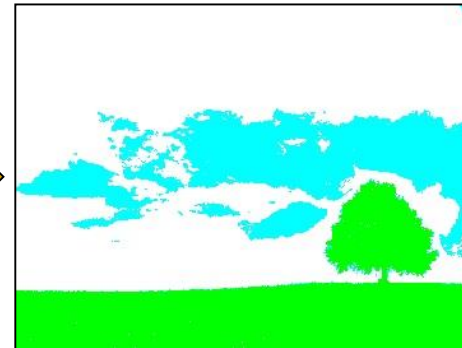
空



雲

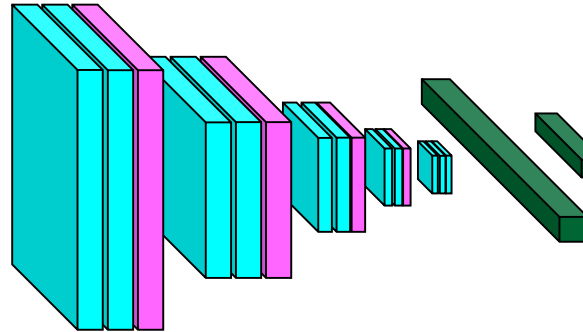
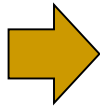


ニューラル  
ネットワーク



# エンコーダ・デコーダ

## 畳み込みネットワーク(CNN)



畳み込み層

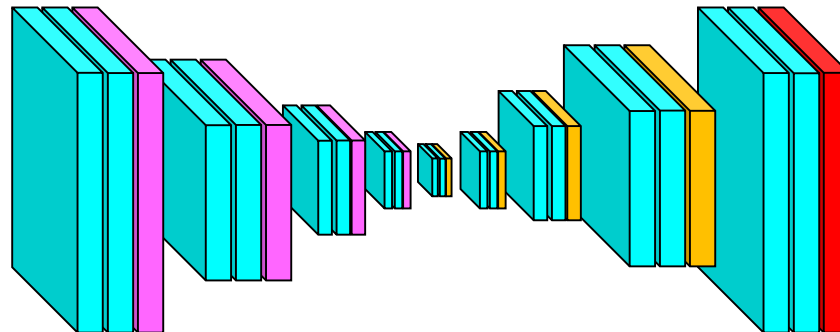
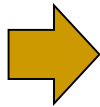
プーリング層

全結合層

ソフトマックス層

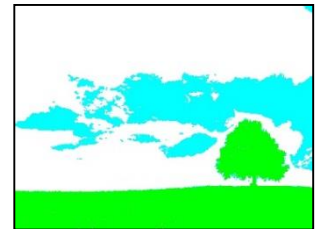
クラス分類

## Full Convolutional Network (FCN)



アンプーリング層

アップサンプリング層

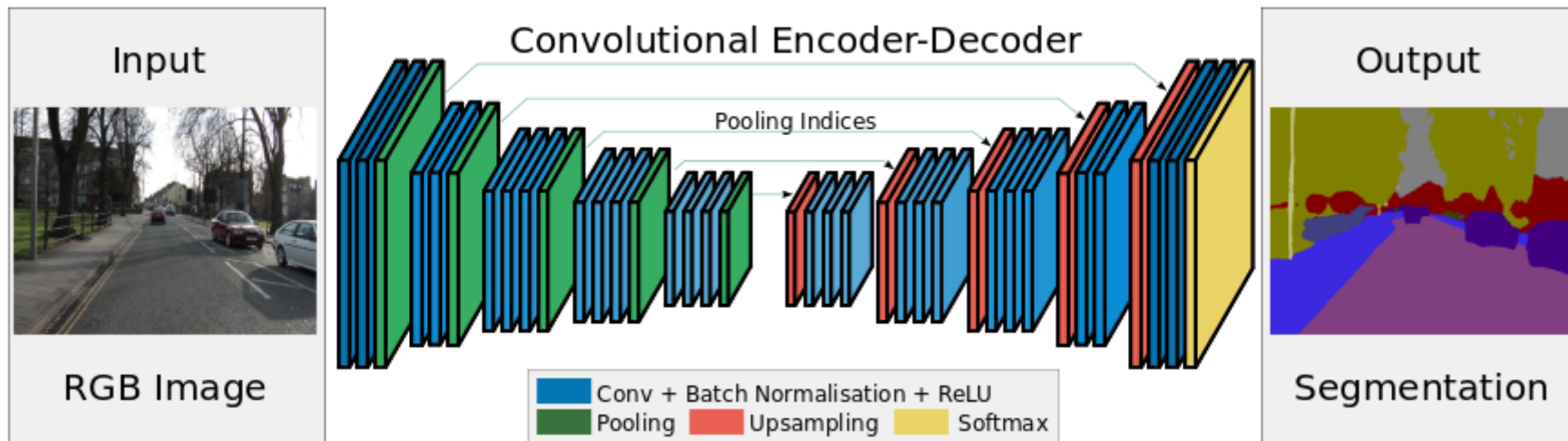


エンコーダ(畳み込み)

デコーダ(逆畳み込み)

# Segnet

エンコーダ  
VGG16の畳み込みネットワーク



プーリング

12	34	5	11
7	9	16	31
24	16	39	0
45	21	33	10



34	31
45	39

pooling Indices

アップサンプリング

12	4
16	21



0	12	0	0
0	0	0	4
0	0	21	0
16	0	0	0



# U-Net

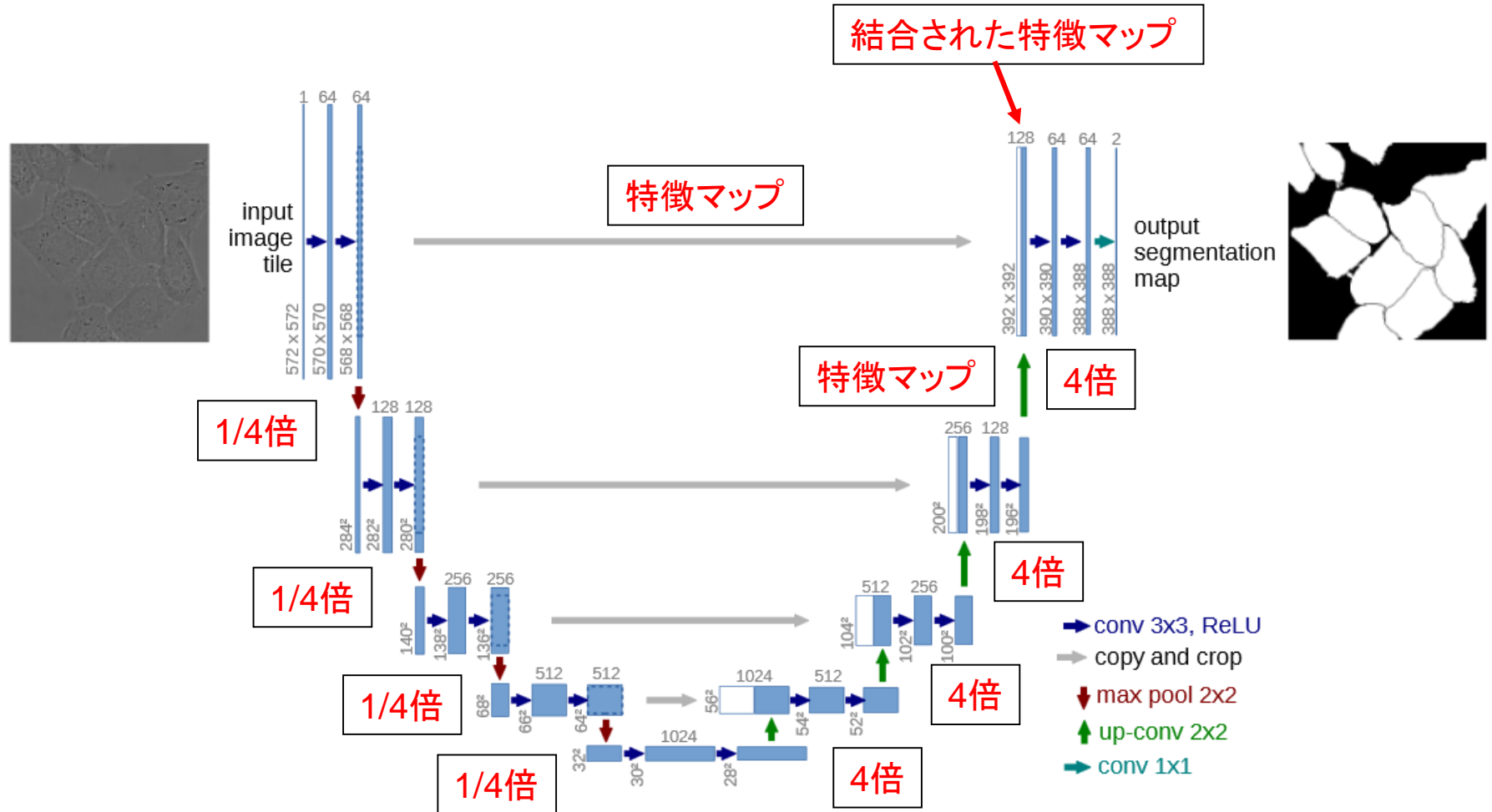
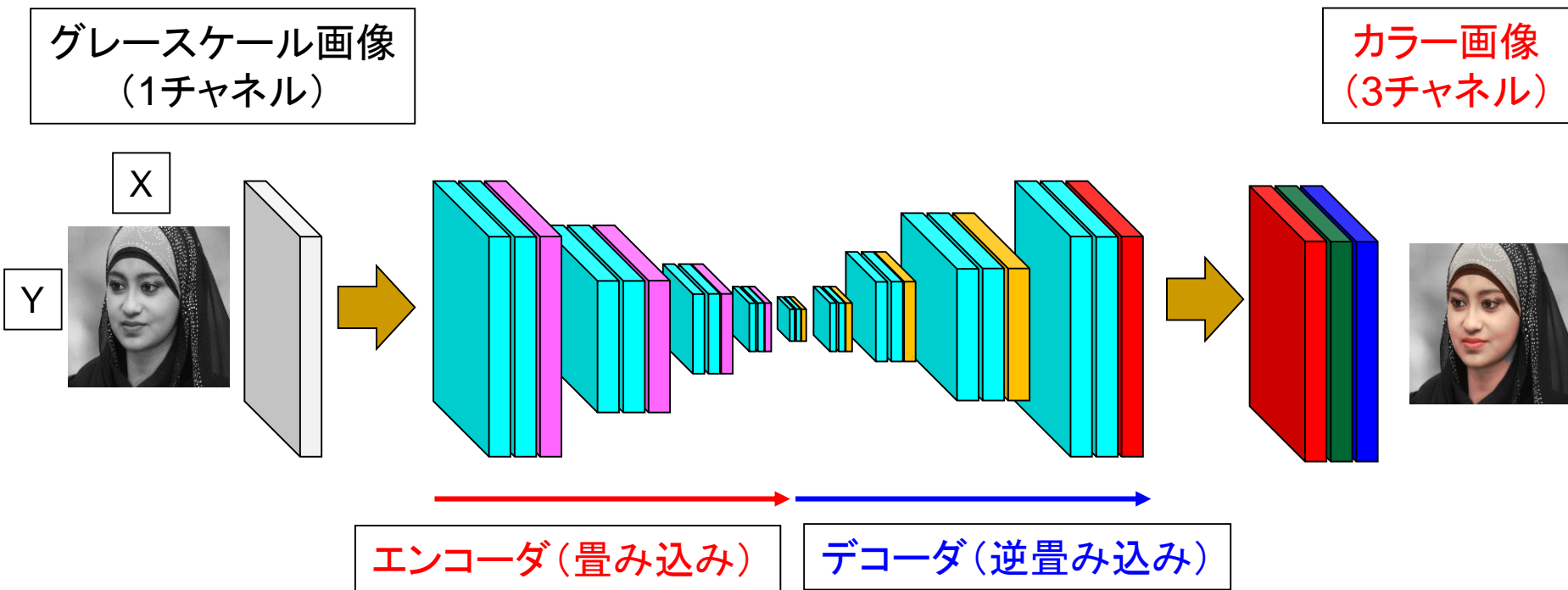


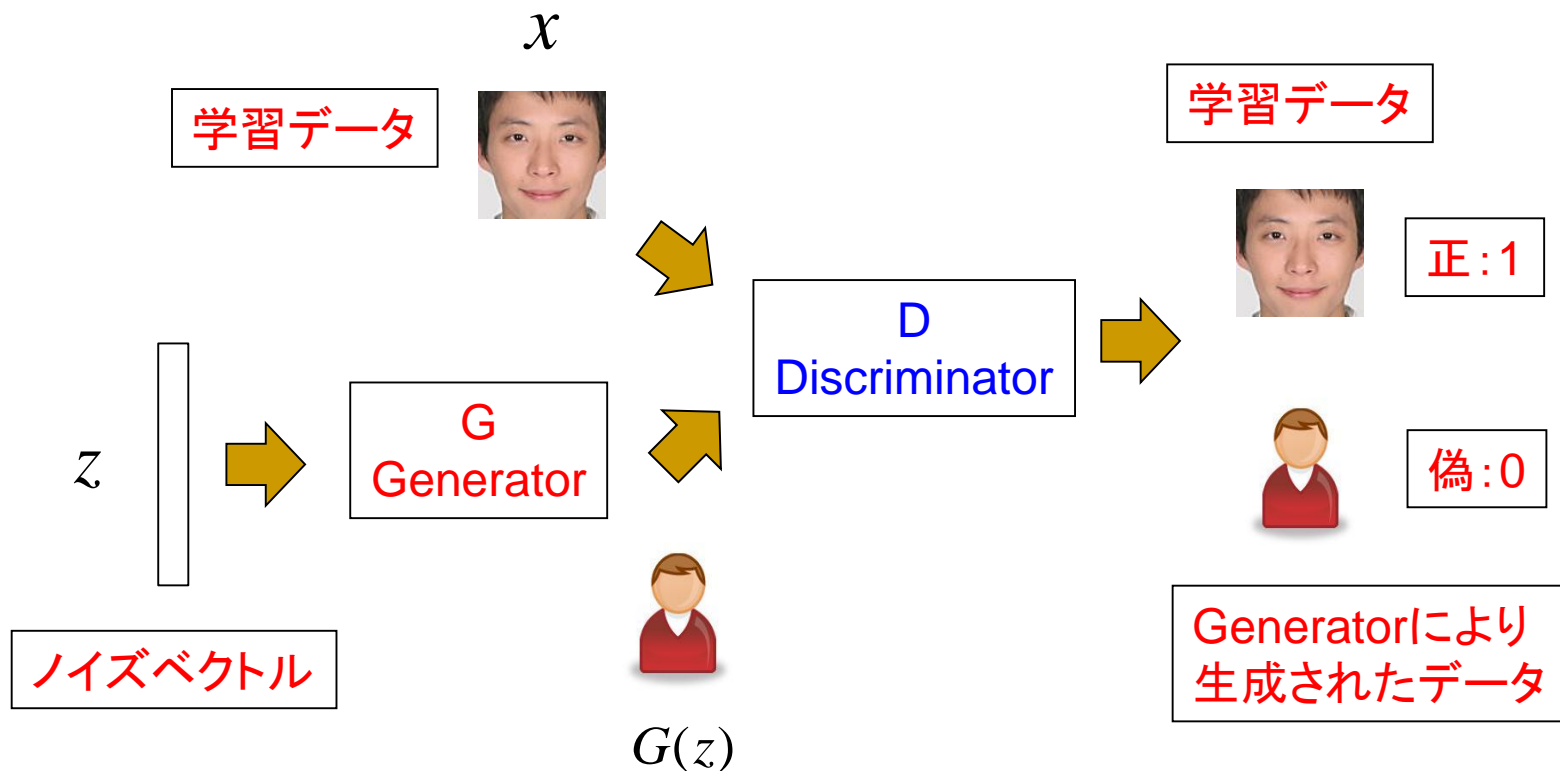
図: Olaf Ronneberger: U-Net: Convolutional Networks for Biomedical Image Segmentation, 2015

# グレースケール画像のカラー化



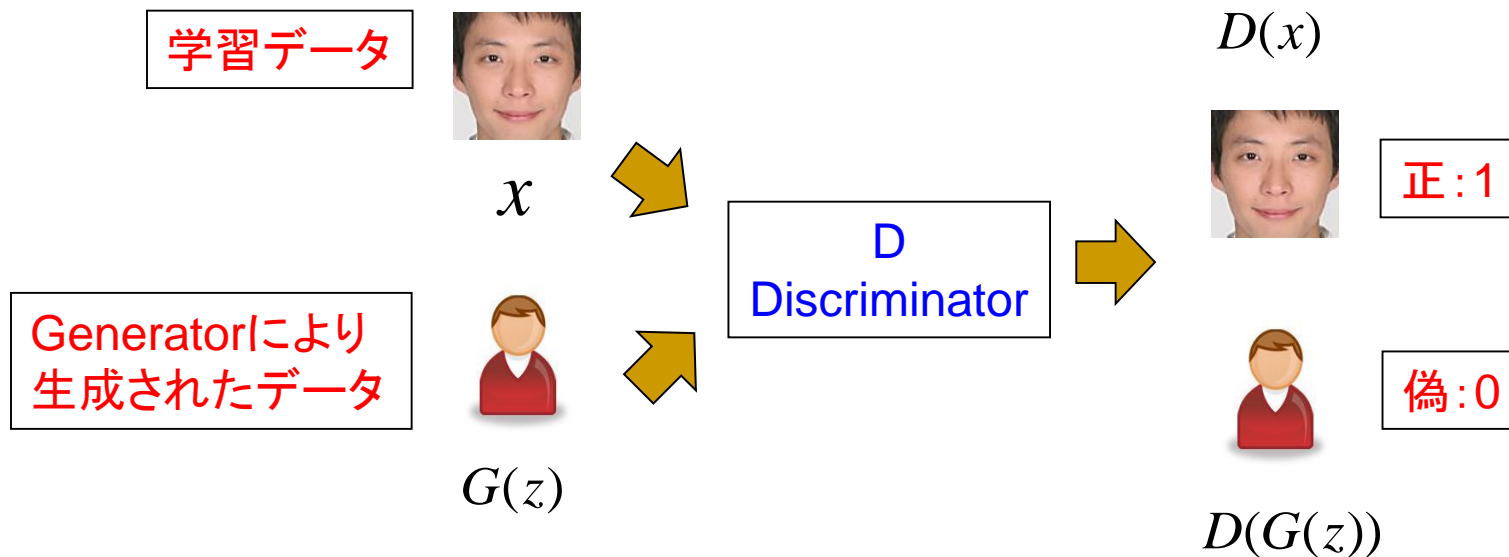
# 敵対的生成ネットワーク

## (GAN: Generative Adversarial Network)



# Discriminator

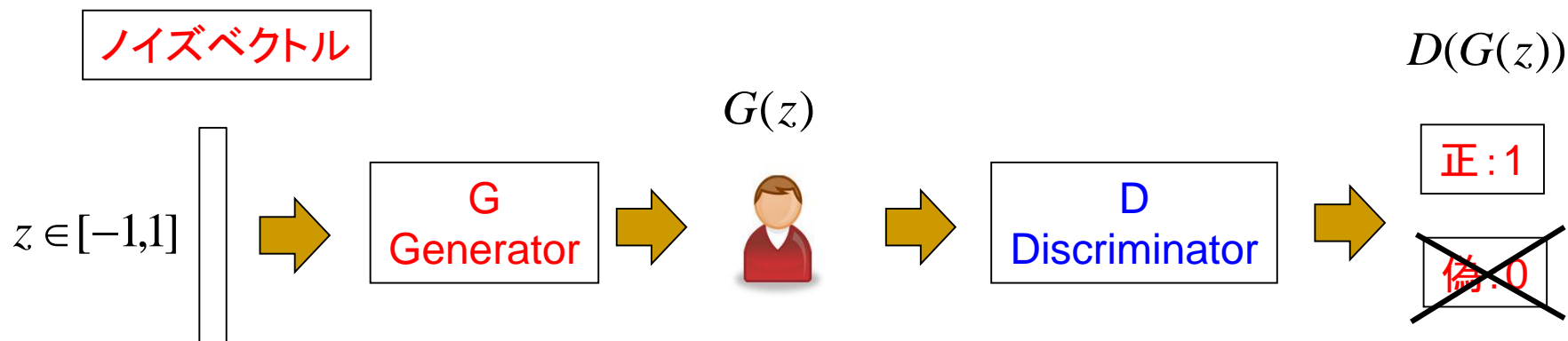
- Discriminatorは、学習データを正、Generatorにより生成されたデータを偽と判定するように学習



$$L_D = \frac{1}{N} \sum_{i=1}^N [\log D(x_i) + \log(1 - D(G(z_i)))]$$

# Generator

- Generatorは、学習データと同じ確率分布となるデータを生成するように学習
- Discriminatorに誤認識させる(騙す)ように学習  
→Discriminatorからの出力 $D(G(x))$ を1



$$L_G = \frac{1}{N} \sum_{i=1}^N \log(1 - D(G(z_i)))$$

# GANの学習方法

Discriminatorは正しく  
判別するように学習

DiscriminatorがGeneratorに  
よる生成データを偽と判別

$$\min_G \max_D V(D, G) = E_{x \sim P_{data}(x)} [\log D(x)] + E_{z \sim P(z)} [\log(1 - D(G(z)))]$$

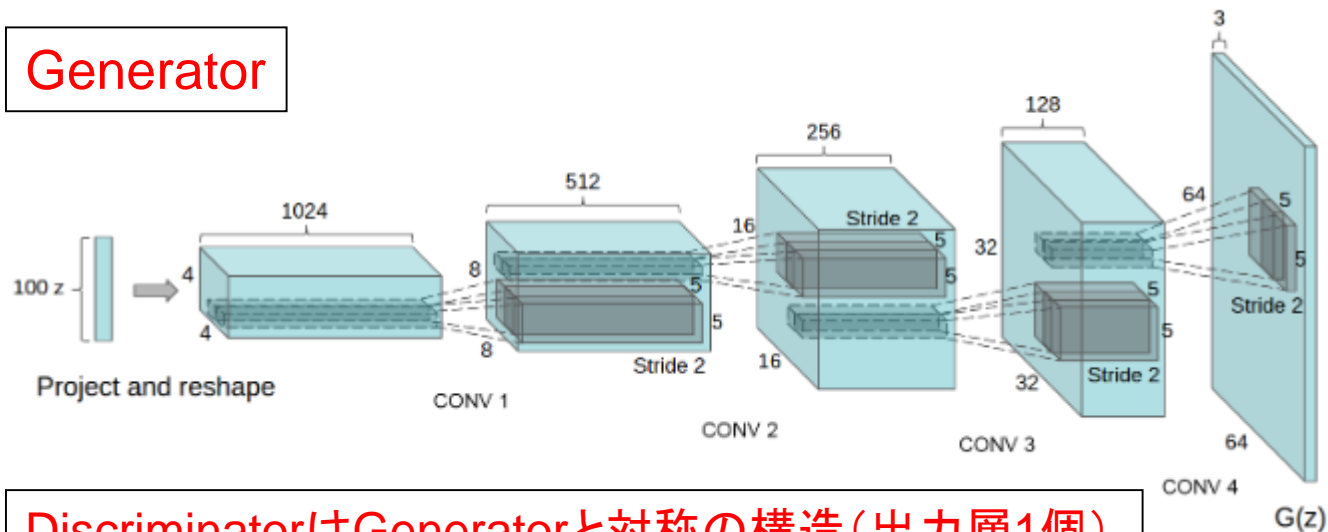
GeneratorはDiscriminator  
を誤判別させるように学習

Discriminatorが学習  
データを正と判別

- Discriminator → Generator → Discriminator → Generator . . .  
と交互に学習

# Deep Convolutional GAN (DCGAN)

## Generator



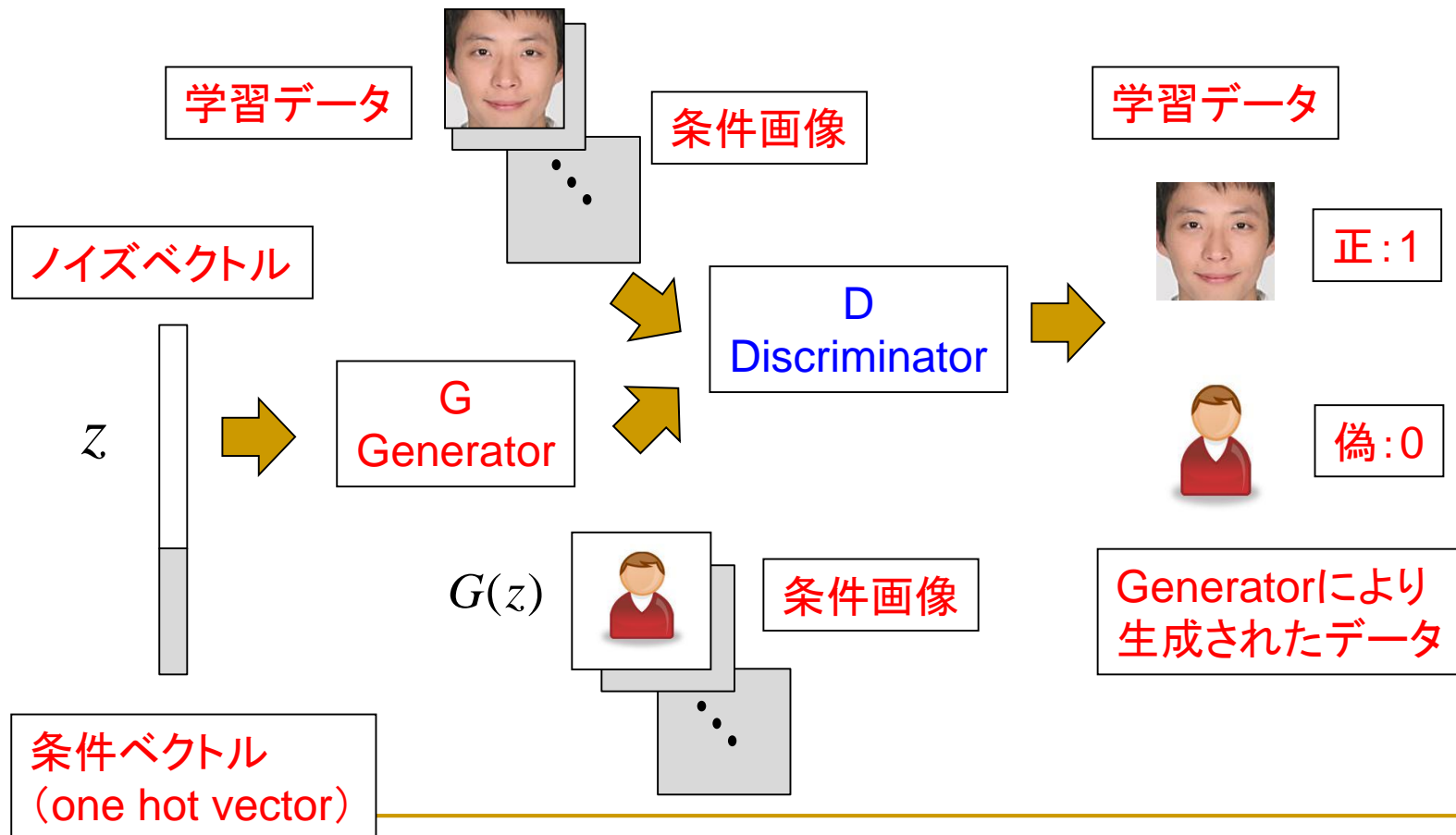
DiscriminatorはGeneratorと対称の構造(出力層1個)

## 生成された画像



# Conditional GAN (CGAN)

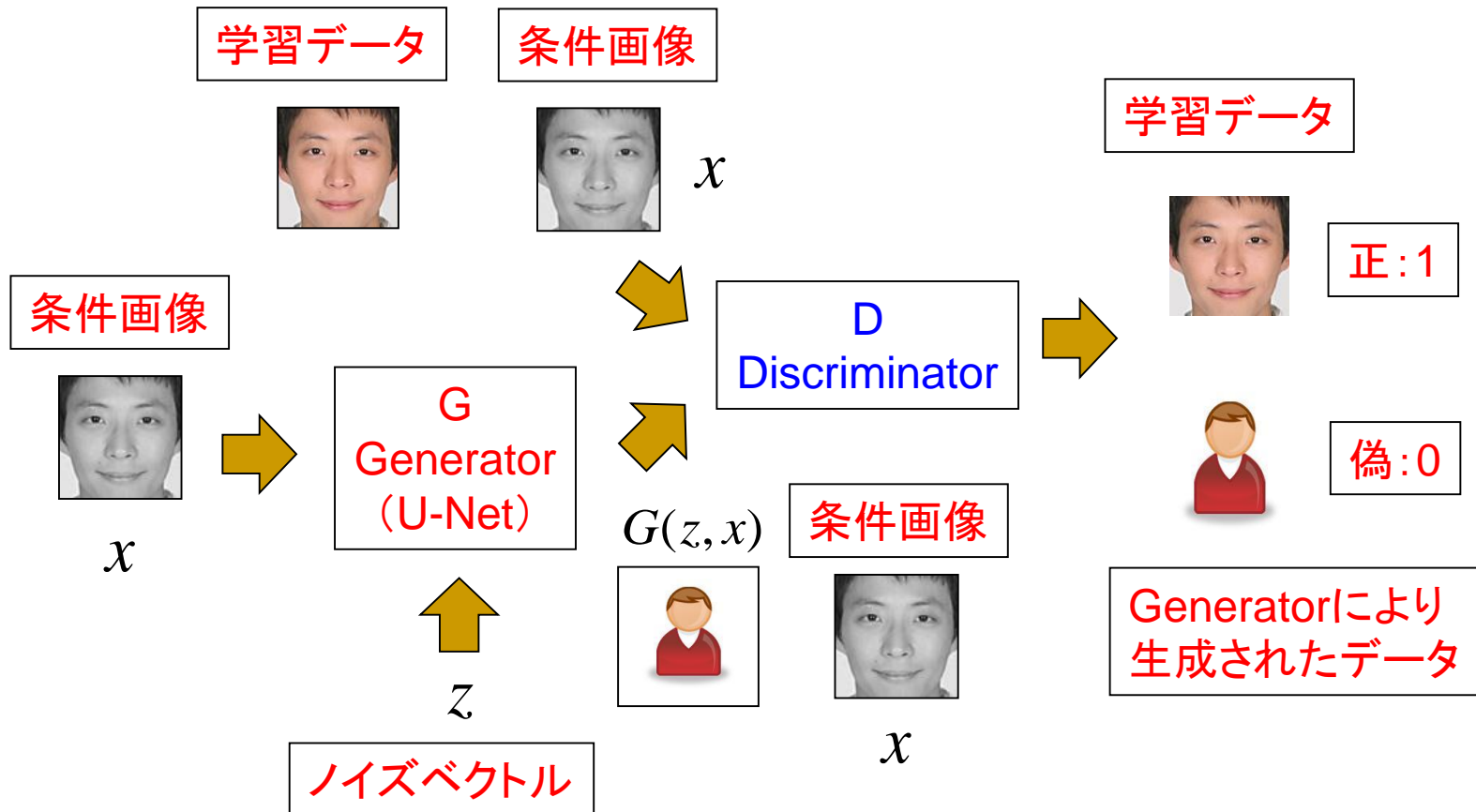
$$\min_G \max_D V(D, G) = E_{x \sim P_{data}(x)} [\log D(x | y)] + E_{z \sim P(z)} [\log(1 - D(G(z | y)))]$$



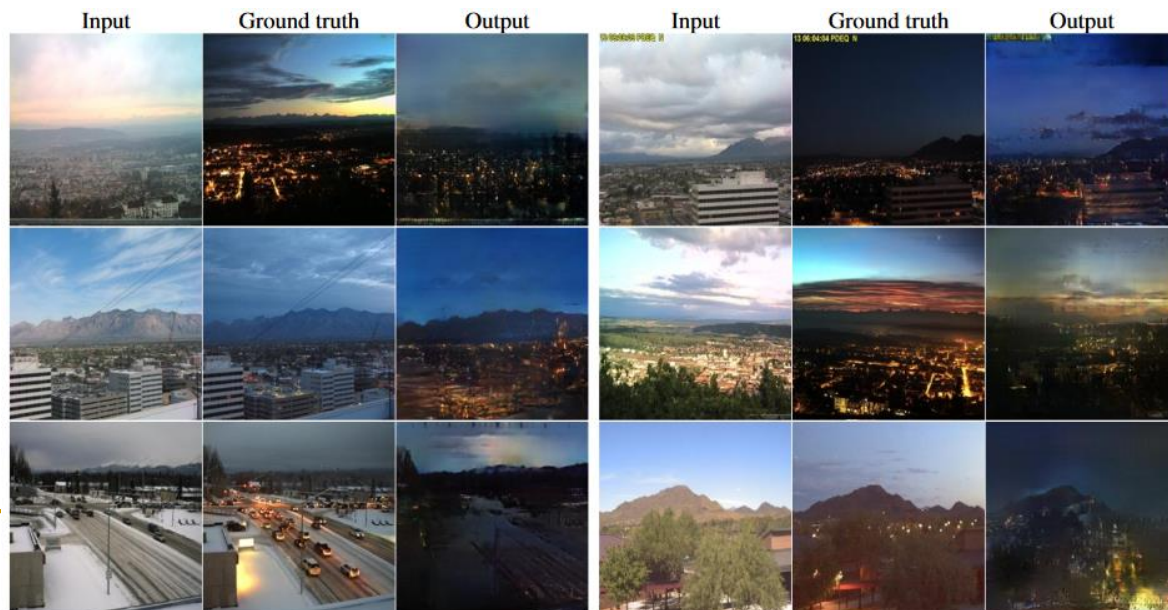
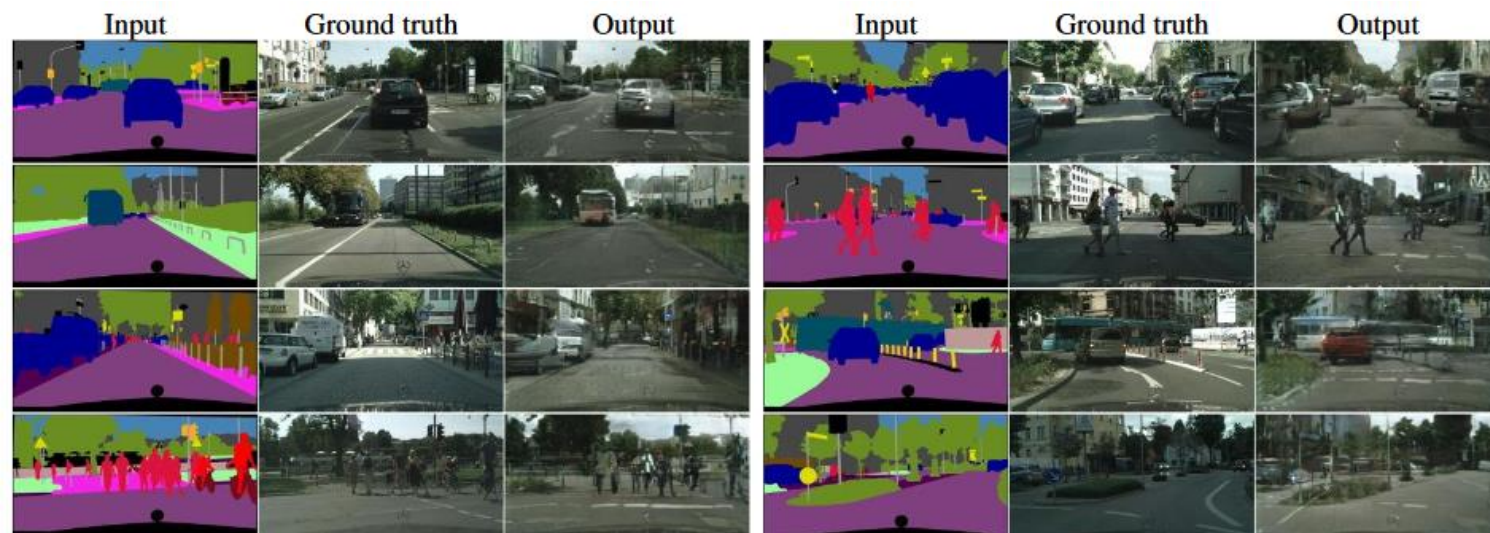


# Pix2Pix

## ■ グレースケール画像のカラー化



# Pix2Pixによる生成画像



# 畳み込みニューラルネットワークのまとめ

- 現在, さまざまなアイデア, モデルが日々提案されています.
- ぜひとも, いろいろと調べて, 動かしてみてください.

# (本日の)参考文献

- J.デイホフ:ニューラルネットワークアーキテクチャ入門, 森北出版(1992)
- P.D.Wasserman:ニューラル・コンピューティング, 理論と実際, 森北出版(1993)
- C.M.ビショップ:パターン認識と機械学習(上), シュプリンガー・ジャパン(2007)
- 岡谷貴之:深層学習, 講談社(2015)
- 瀧雅人:これならわかる深層学習入門, 講談社(2017)
- 原田達也:画像認識, 講談社(2017)

