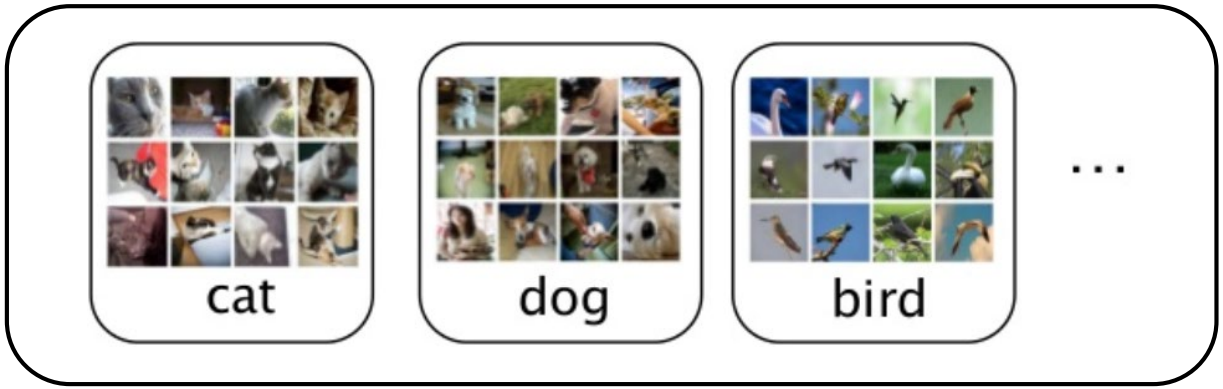
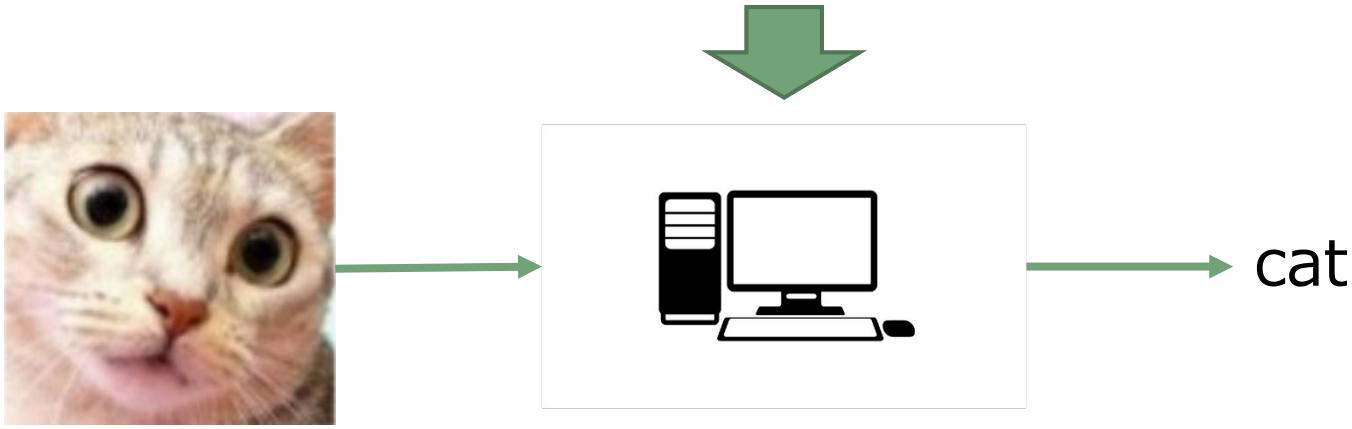
機械学習とディープラーニングの基礎

パターン認識と機械学習

教師あり
学習データ
画像＋ラベル



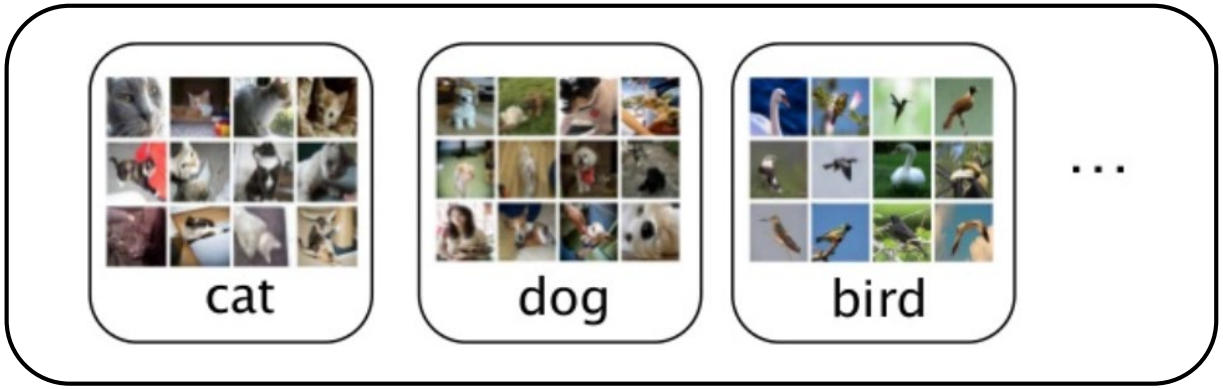


未知のデータのラベルを正しく認識するよう学習させる

どのように学習させる？

機械学習の基本概念

教師あり
学習データ
画像
＋
クラスラベル



基本的には比較
画像の近さでラベルを決定

何を持って近いとするか？
画像は画素の集合、画素の類似度を利用？
位置が少しずれるだけでも画素の類似度は変わる

→ クラス分類に有用な画像特徴を利用

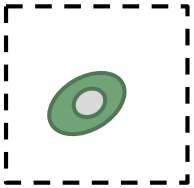
データが数百万あったら、全部1つ1つ比較？

→ 特徴空間を予め分けておく

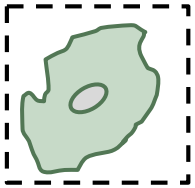
従来の機械学習：特徴量

Hand-made 特徴量：問題によって適切な特徴量を設計

良い細胞　　悪い細胞



or



の2つに分類する問題

見た目の特徴	小さい 丸っぽい 色が濃い	大きい 凸凹している 色が薄い

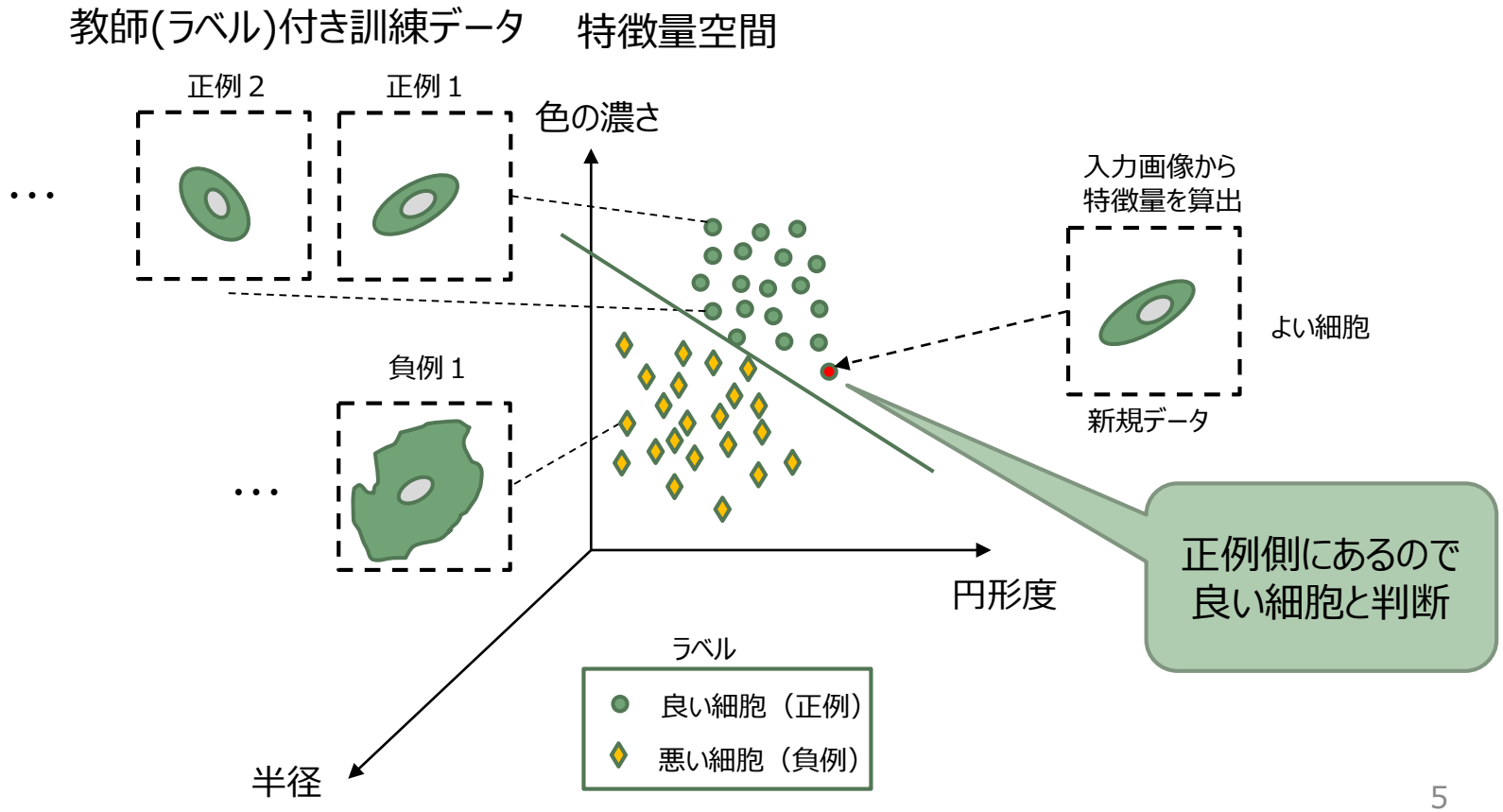
2つの違いを表現する数値特徴量を設計

特徴量	20 0.8 0.9	半径 円形度 緑色の濃さ	30 0.5 0.6

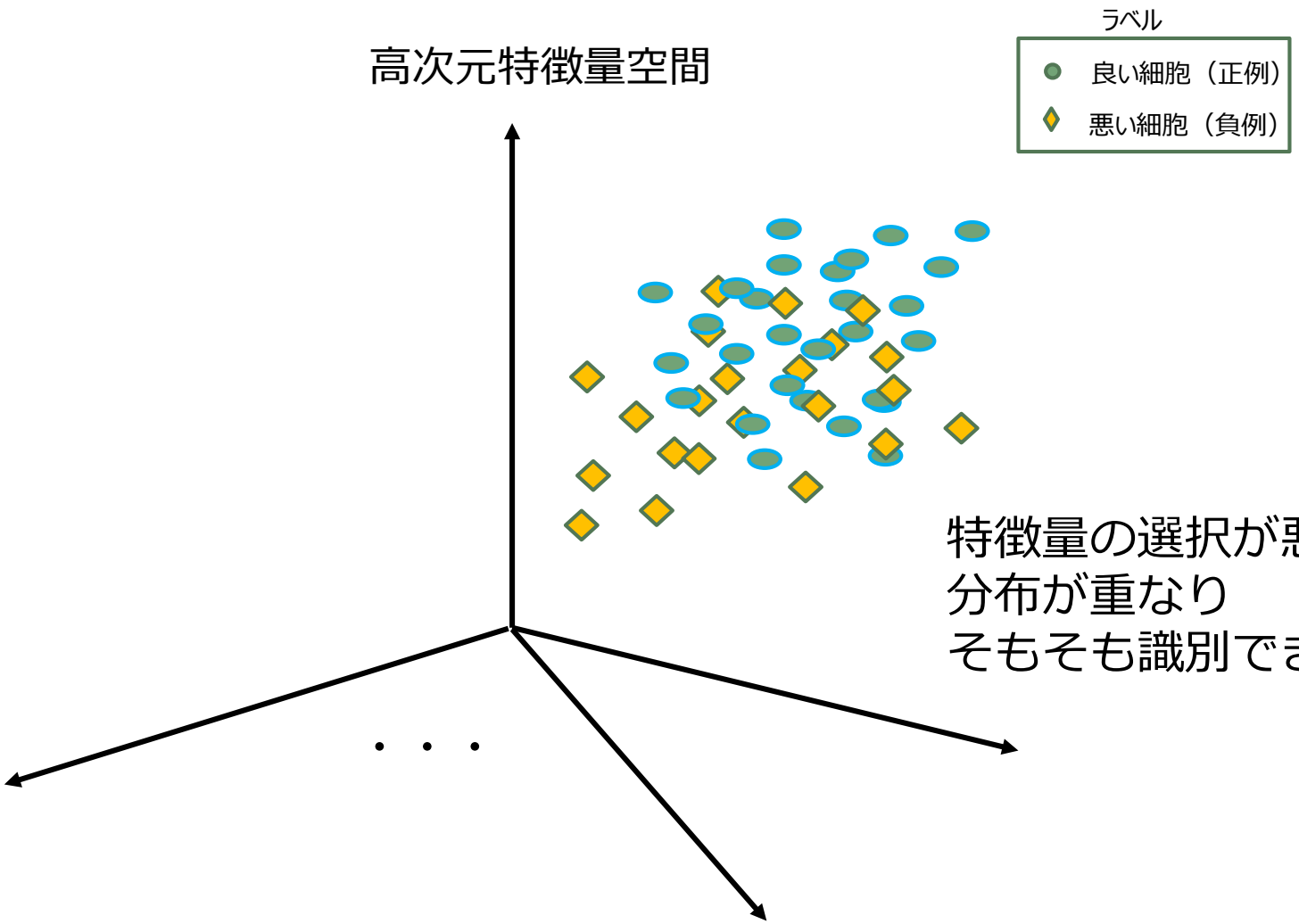
従来の機械学習：特徴量空間

学習
1．訓練データ１つ１つの画像の特徴量を算出
2．特徴量空間にマッピング
3．識別局面を決定

テスト（推定）
1．入力データの特徴量を算出
2．特徴量空間にマッピング
3．識別局面を用いて識別



従来の機械学習：特徴量選択の課題



特徴量の選択が悪いと、
分布が重なり
そもそも識別できない。

従来の機械学習

特徴量抽出

Intensities　PCA　SIFT

Haar-like　HOG　Texton　LBP

Bag of features　Fisher-vector

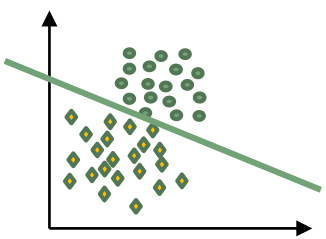
対象を分類するための
一般的な特徴量の提案

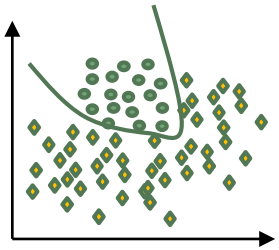
識別面の推定

SVM　判別分析　グラフィカルモデル

Random forest　adaboost　ベイズ推定

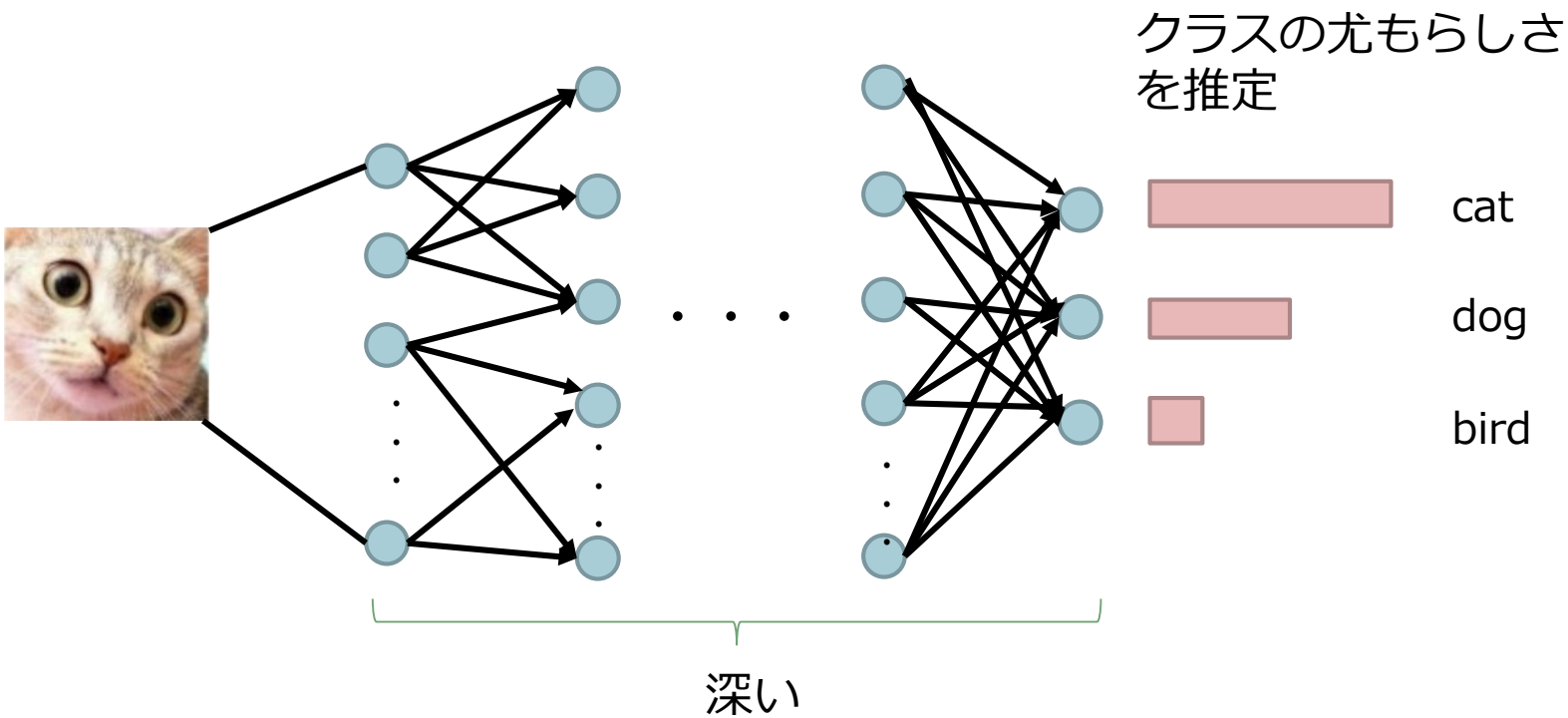
特徴量空間の中で
分布をどのように分離するか





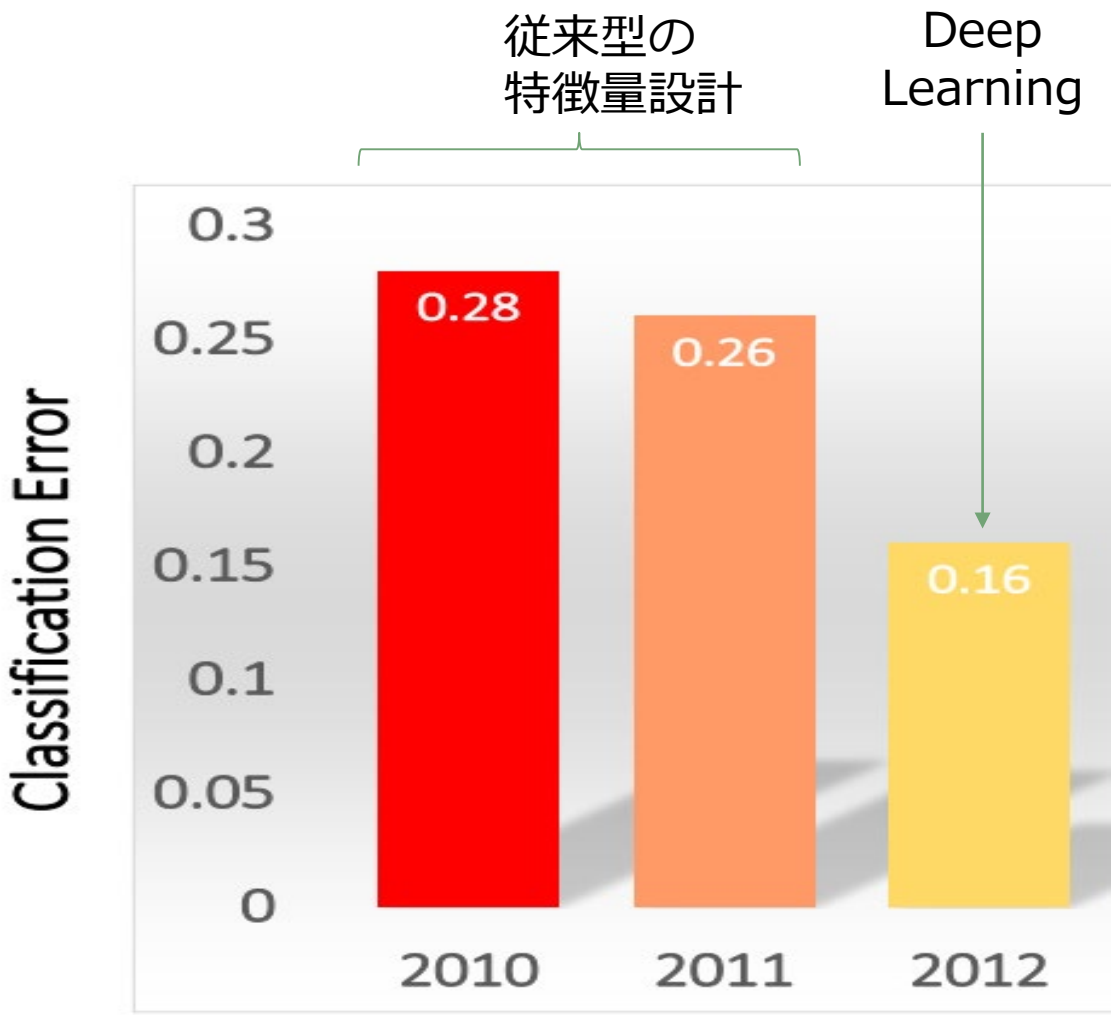
Deep Learning（　深層学習）

- ニューラルネットワークを用いたパターン認識技術の総称
  - 脳(神経細胞)の働きを模倣した学習アルゴリズム
- 特徴
  - 学習データから特徴量抽出も自動で学習
  - 深く大規模な構造



Deep Learning による精度改善

Deep Learning により10%以上精度向上！

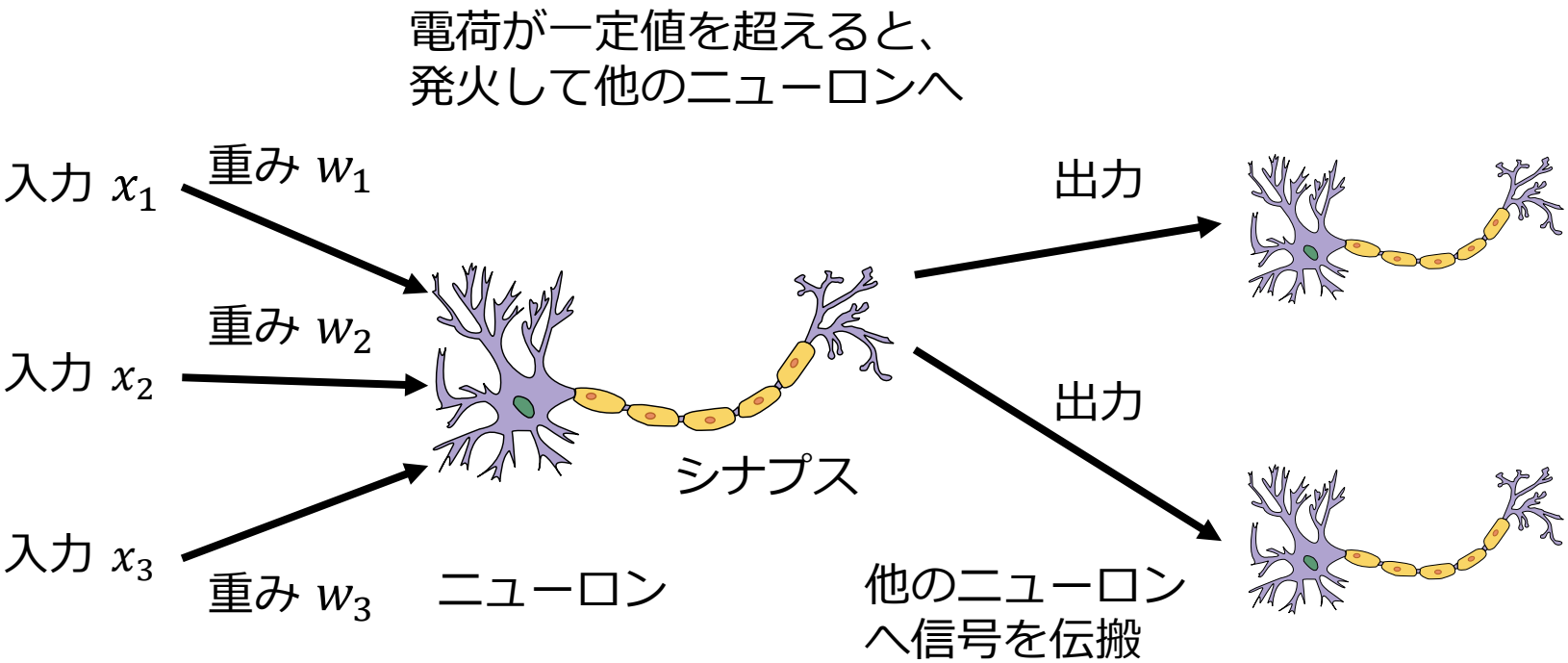


http://image-net.org/challenges/talks_2017/ILSVRC2017_overview.pdf

https://commons.wikimedia.org/

ニューロンの計算モデル

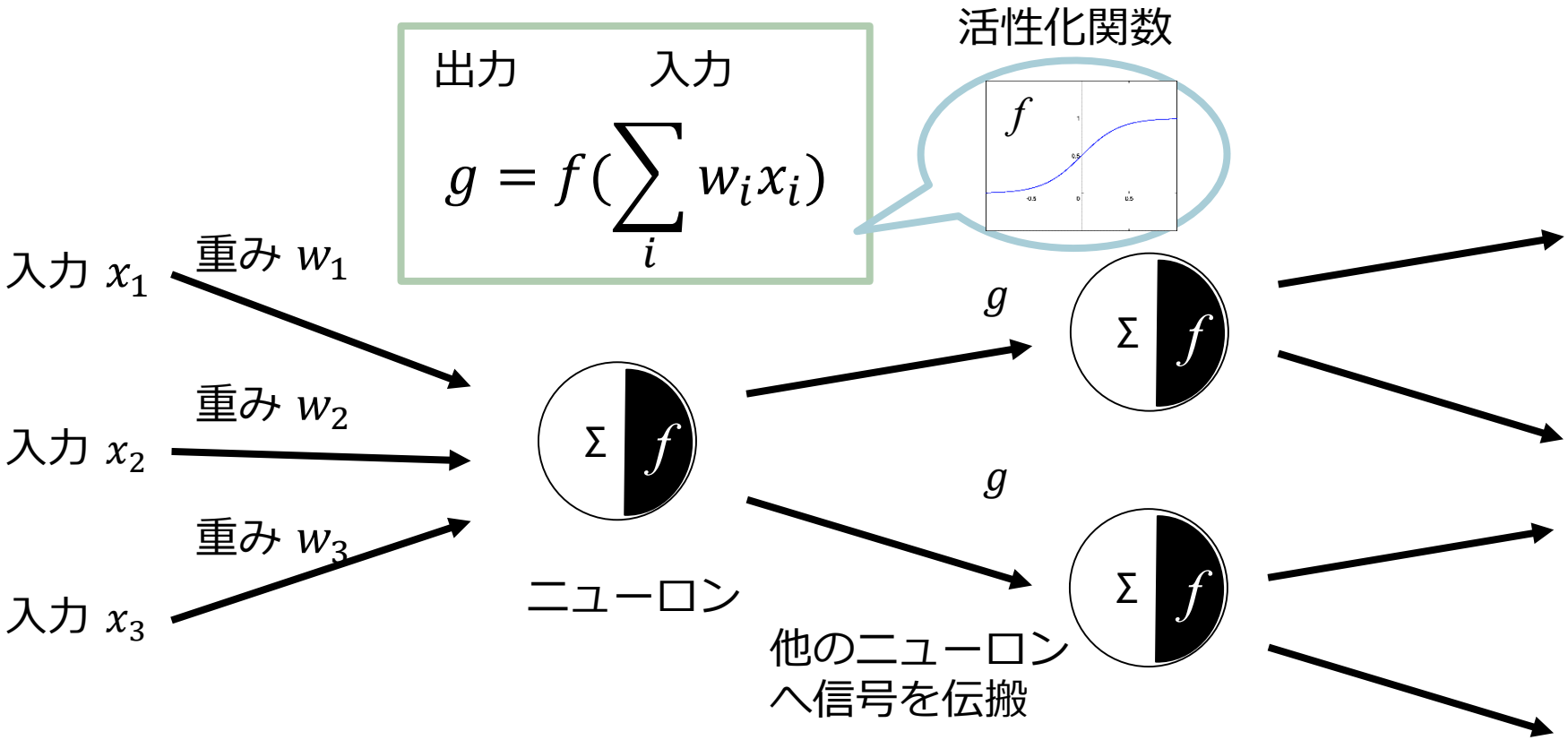
- 脳神経系を模した数学モデル
- 入力の重み付け和が閾値を超えると、シグナルを発生する単純な構造を持ったニューロンのネットワーク



https://commons.wikimedia.org/

ニューロンの計算モデル

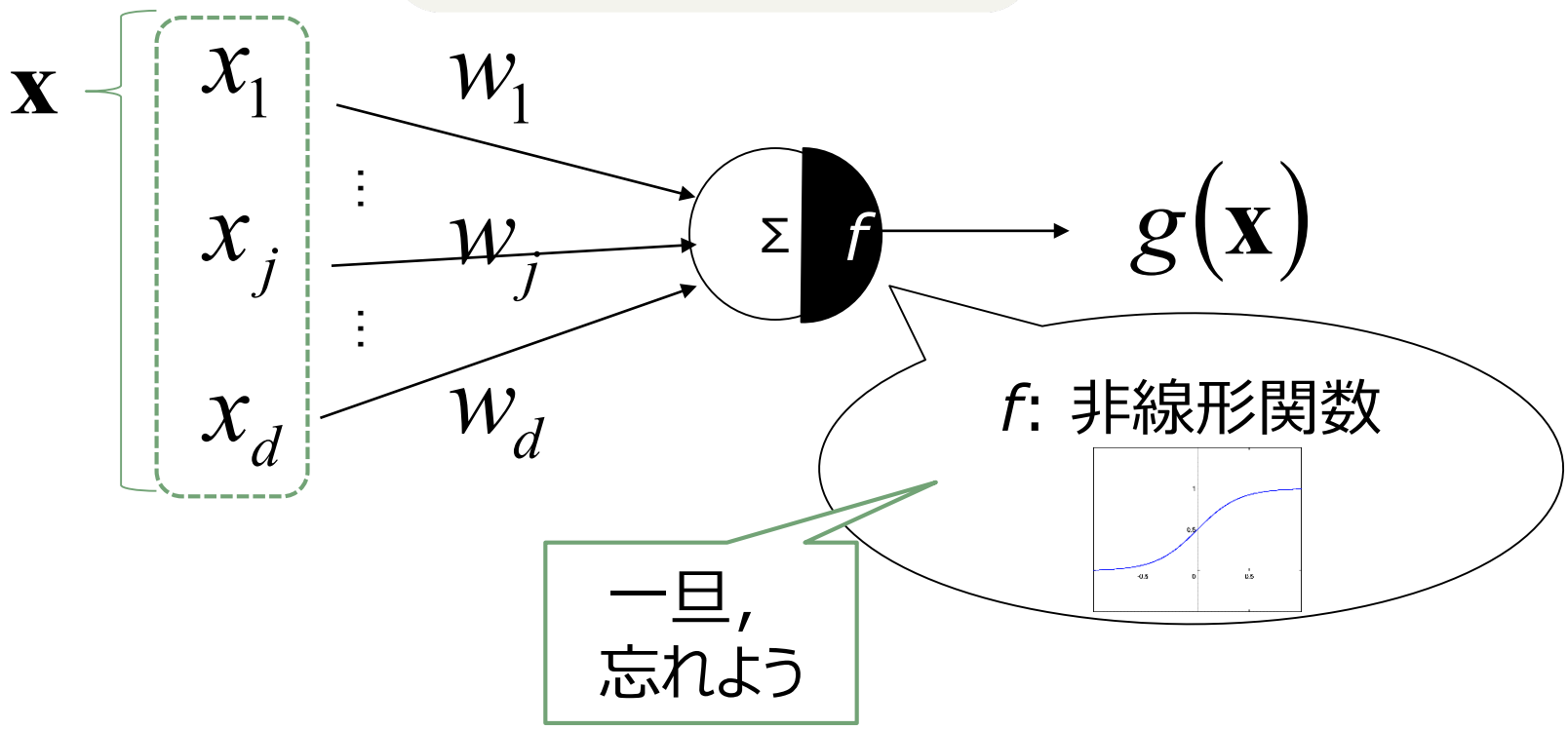
- 脳神経系を模した数学モデル
- 入力の重み付け和が閾値を超えると、シグナルを発生する単純な構造を持ったニューロンのネットワーク



キチッとしたいけど，面倒くさそうなものは考えたくない

九大
内田教授提供

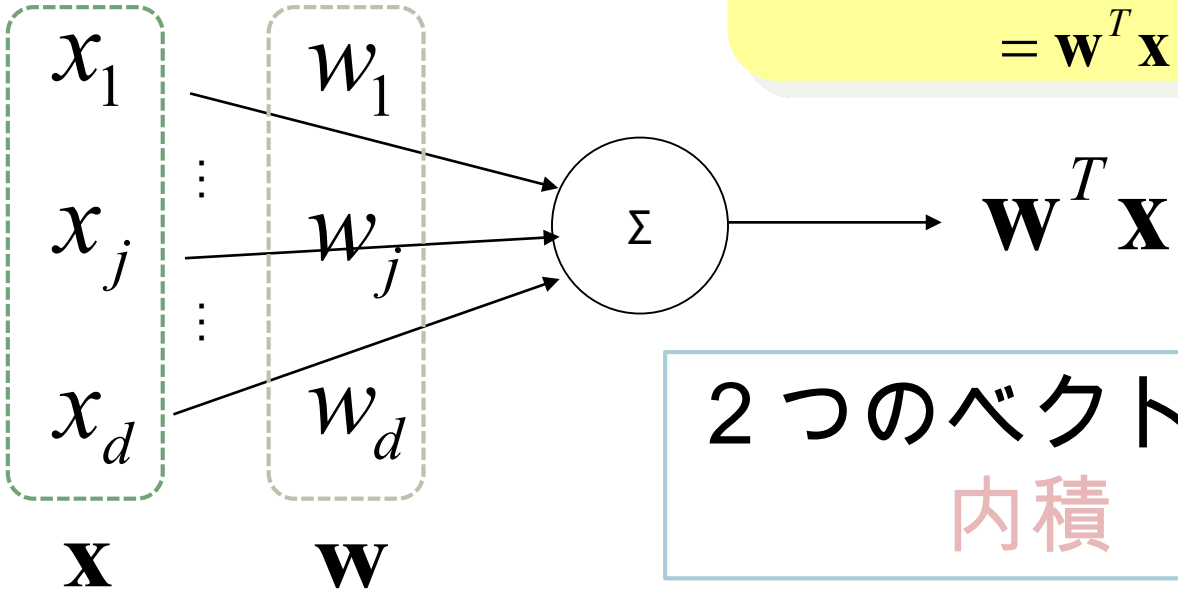
$$g(\mathbf{x}) = f\left(\sum_{j=1}^{d} w_j x_j\right) = f\left(\mathbf{w}^T \mathbf{x}\right)$$



ずいぶん簡単になった.

九大
内田教授提供

$$g(\mathbf{x}) = \sum_{j=1}^{d} w_j x_j$$
$$= \mathbf{w}^T \mathbf{x}$$

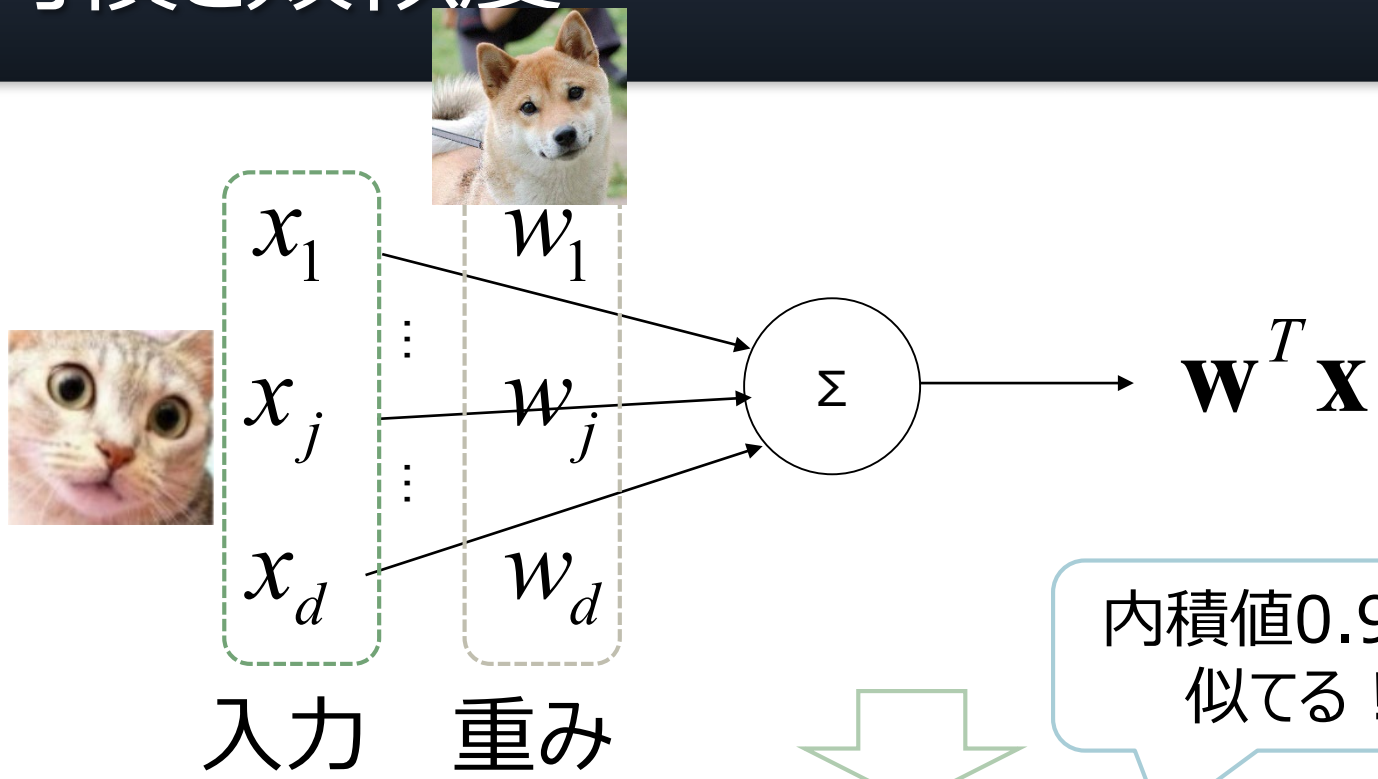


入力　重み

２つのベクトルの
内積

内積と類似度

九大
内田教授提供



内積値0.9なら
似てる！

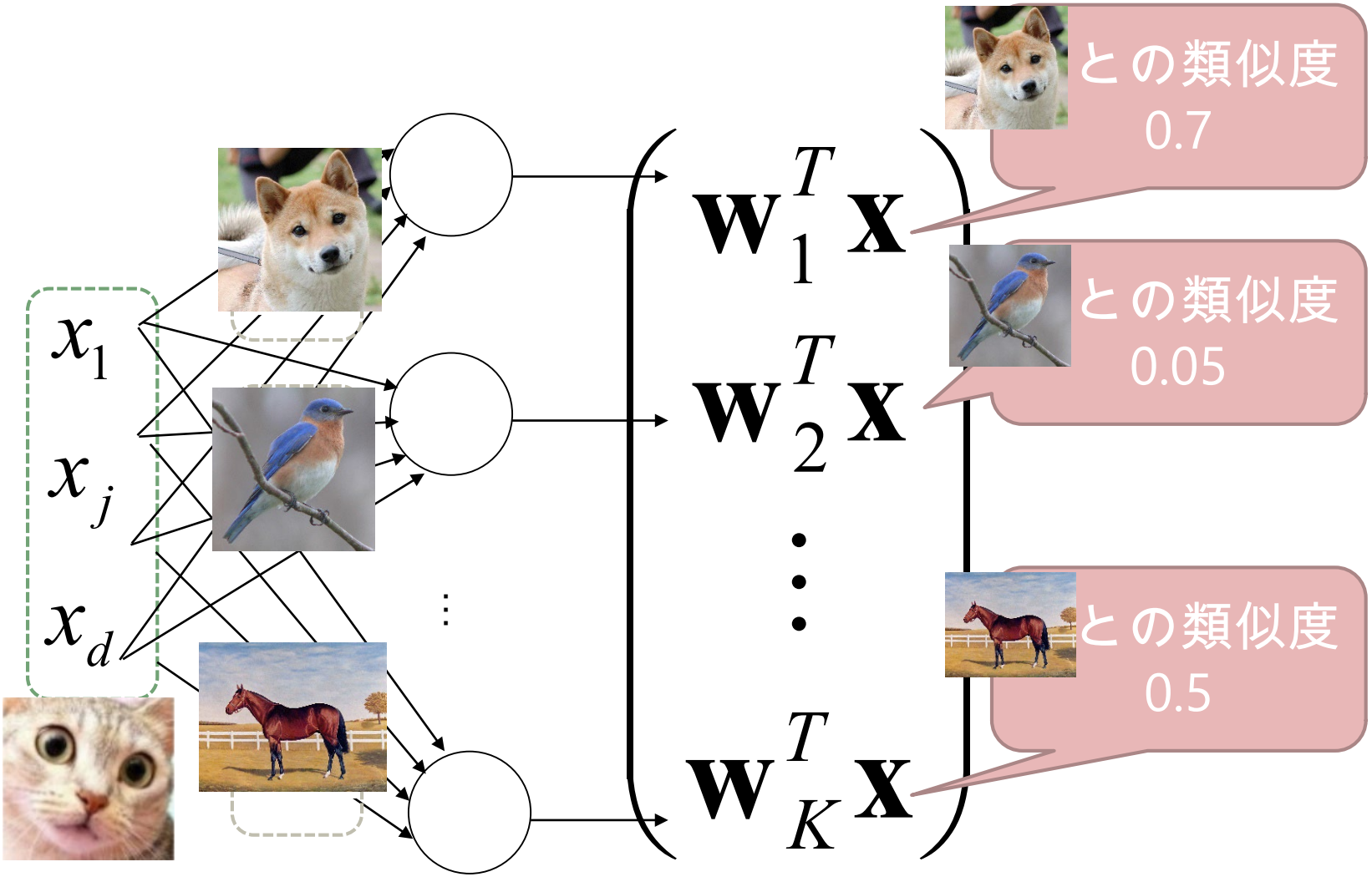
似たベクトルの内積値は大きくなる

→内積は入力**X** と重み**W**の類似度を表現

内積値0.02なら
似てない

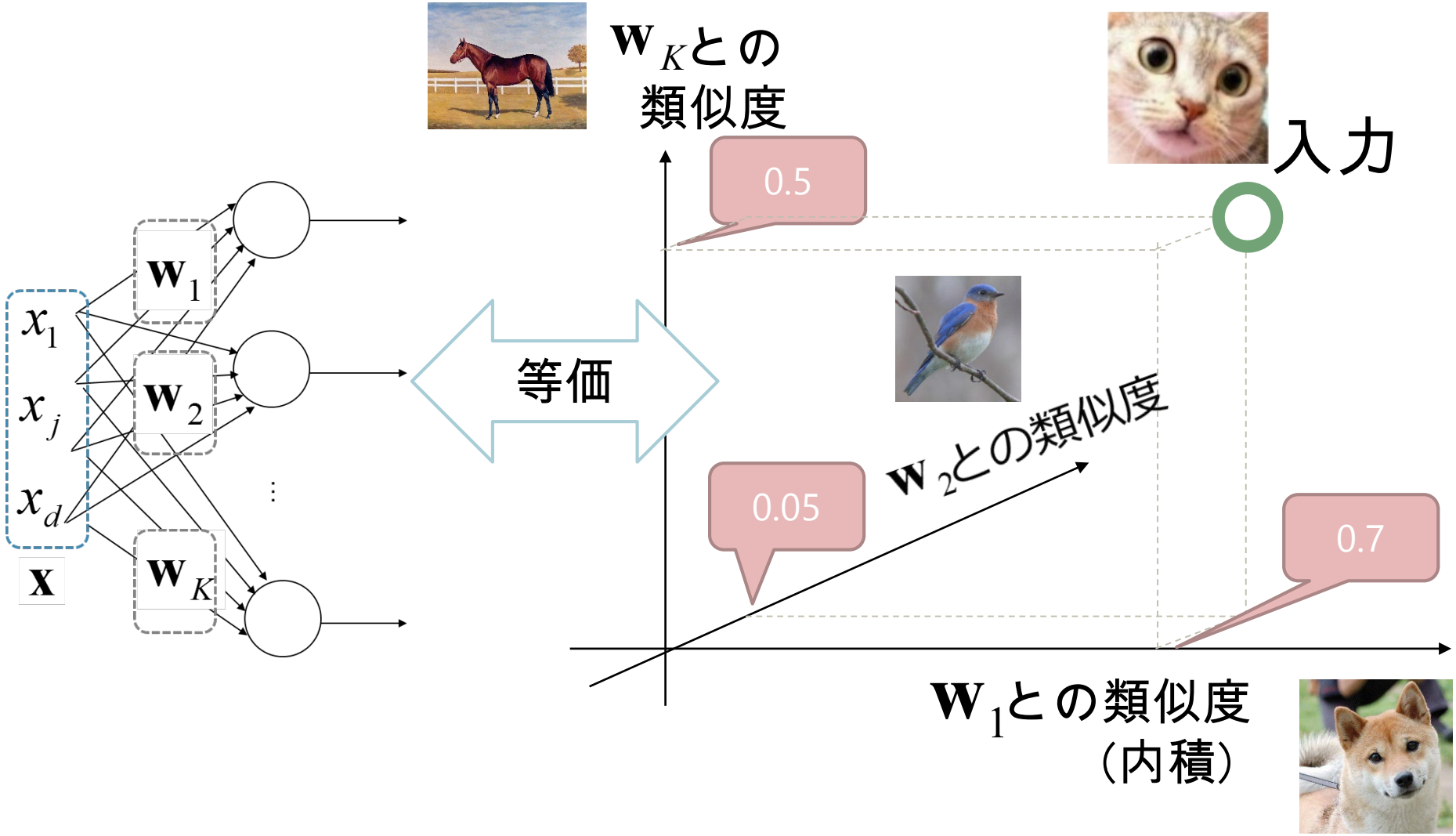
ということは，K個のニューロンがあればK個の類似度が…

九大
内田教授提供



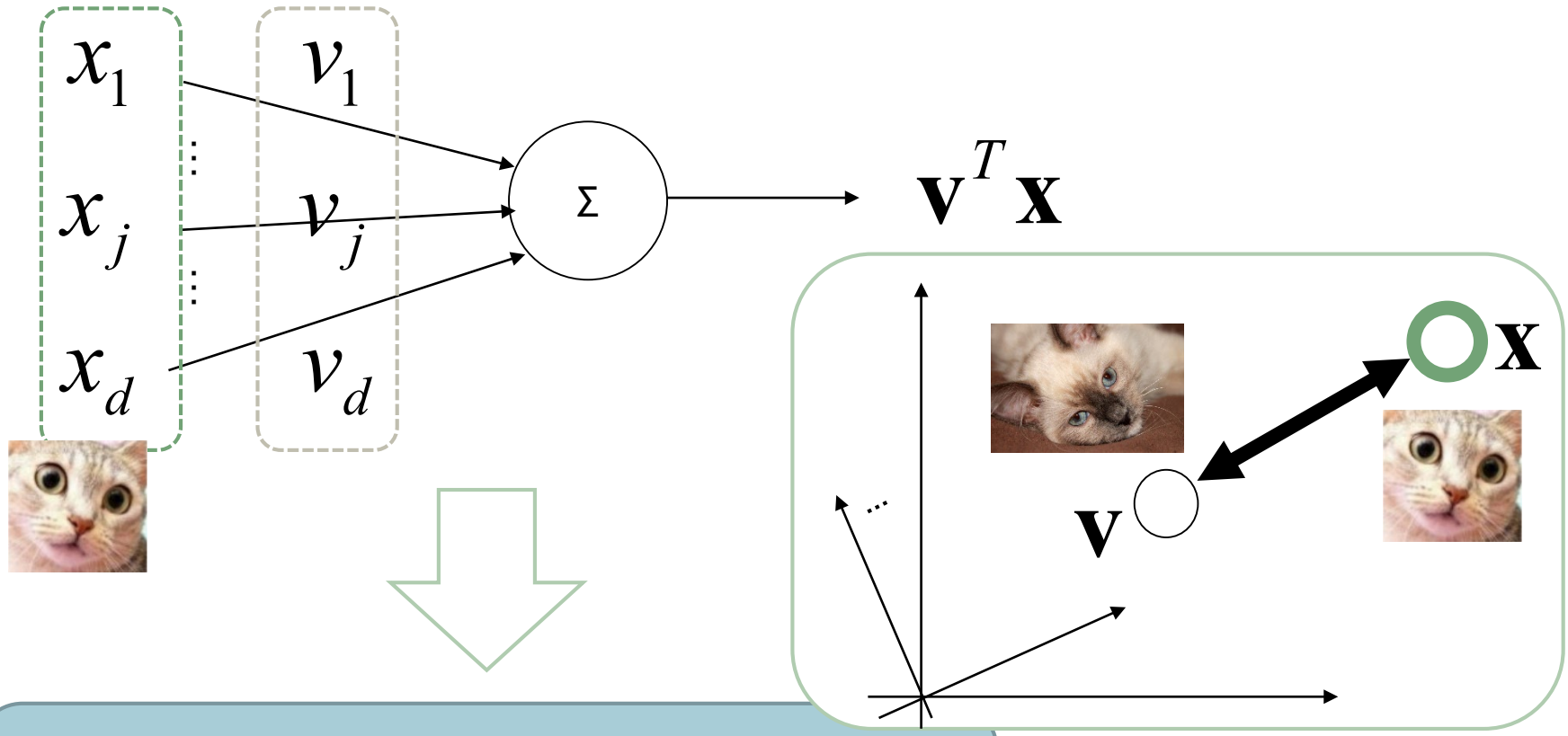
そう，K個のニューロンがあれば K次元の(類似度)特徴が出せる

九大
内田教授提供



実は内積には別の機能も！

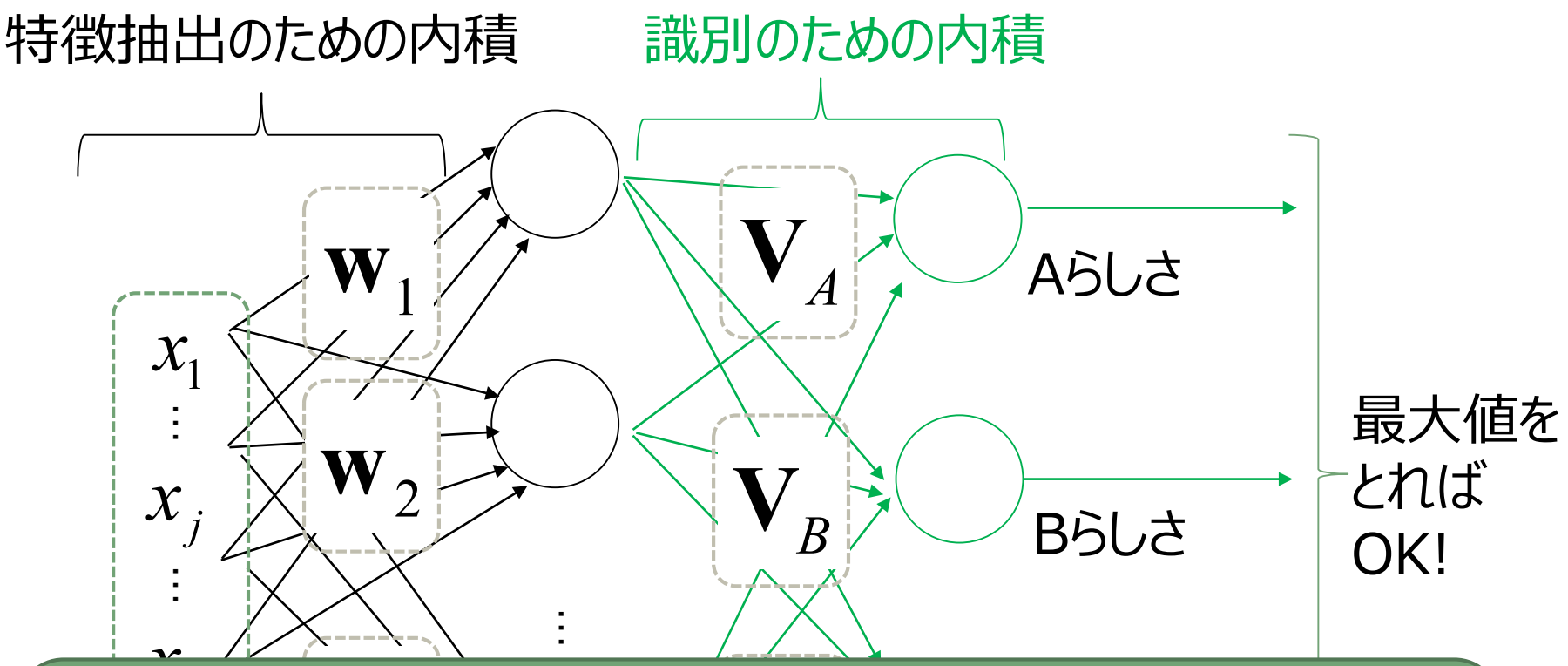
九大
内田教授提供



内積＝類似度なので
識別のための類似度にも使える

ニューラルネットワーク，これで完成！

九大
内田教授提供

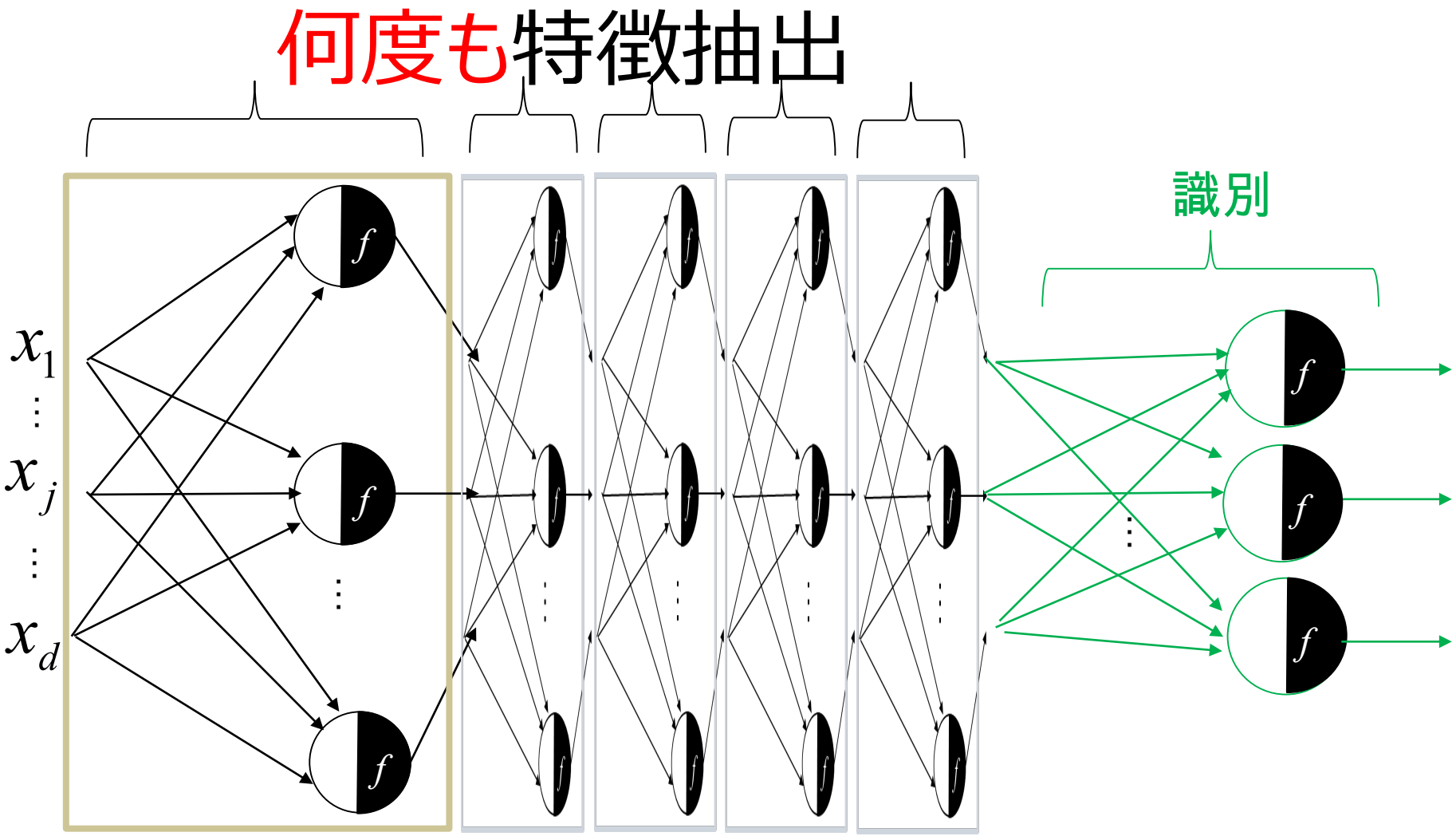


内積だけで特徴抽出と識別の両方を実現！

→ニューロンだけで脳が多機能性を実現するのと似ている

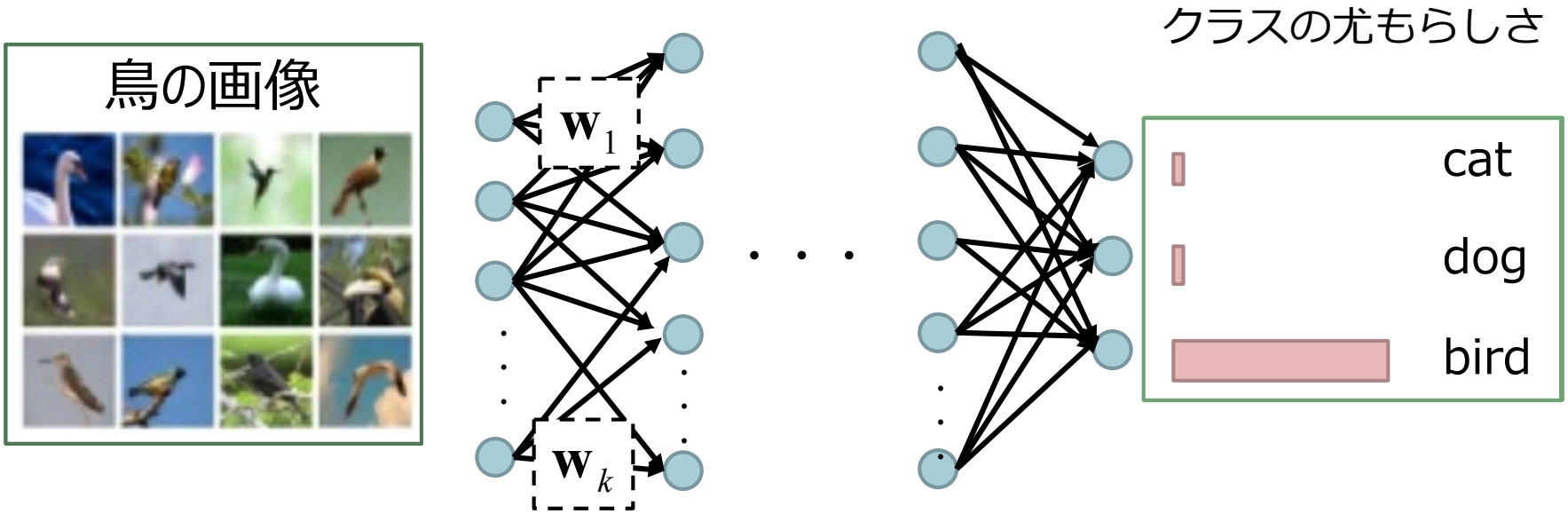
ディープニューラルネットワーク

九大
内田教授提供



ディープニューラルネットワークの学習

- たくさんの学習データで、
  - 猫の画像を入れると「cat」
  - 犬の画像を入れると「dog」
  - 鳥の画像を入れると「bird」
- となるように、重みを決定



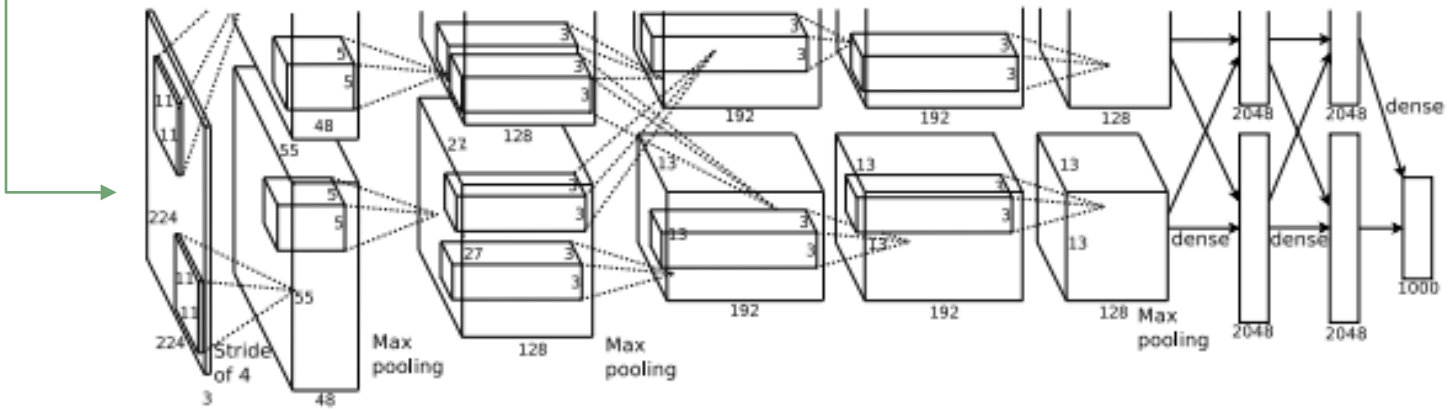
ディープラーニングでできること

クラス分類

A. Krizhevsky et.al., 2012

Classification:入力データのクラスを推定

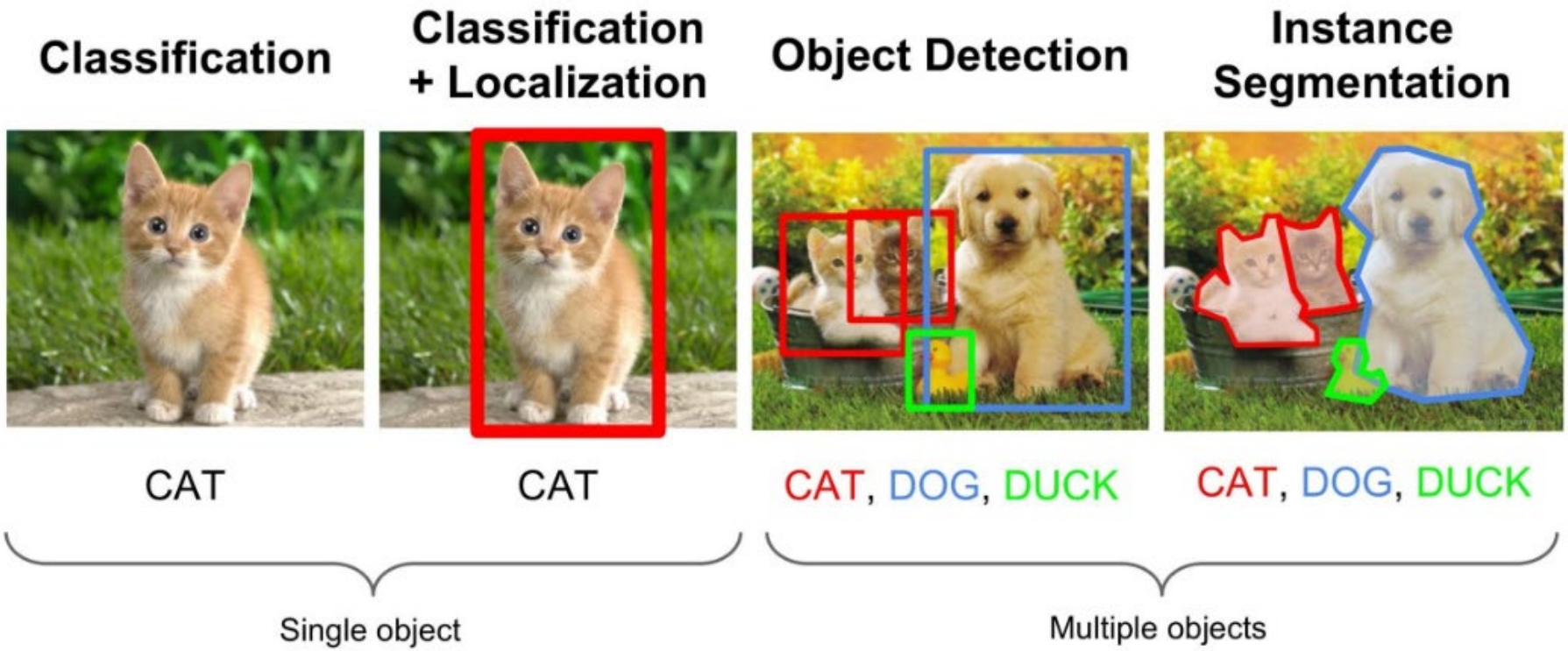
入力



Out:
Leopard
Doc
Cat
Zebra
・・・

認識の基本タスク

- クラス分類（single object）
- クラス分類＋場所の認識（single object）
- オブジェクト検出（Multiple object）
- Segmentation（領域認識）



https://medium.com/comet-app/review-of-deep-learning-algorithms-for-object-detection-c1f3d437b852

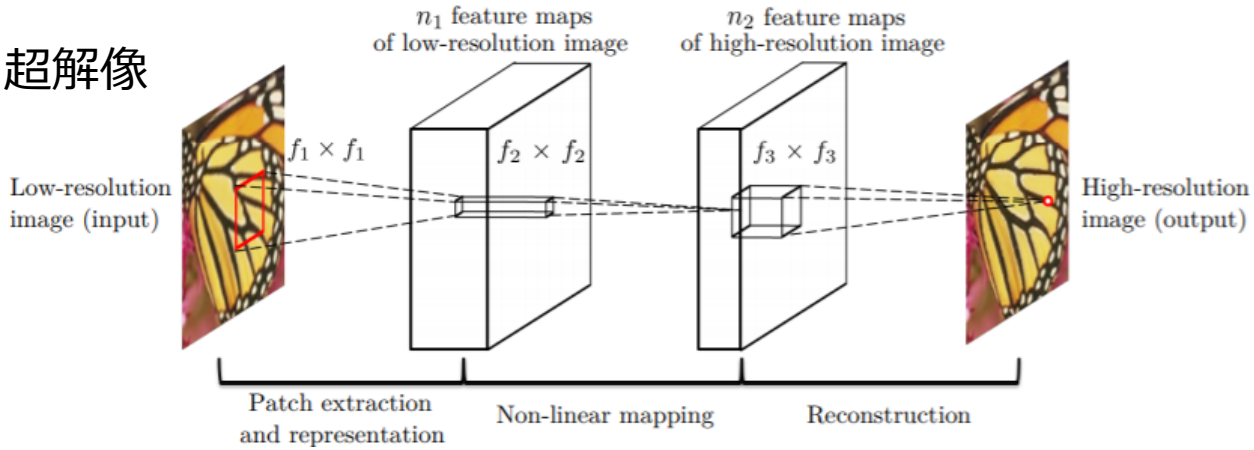
Applications

Image Captioning

The man at bat readies to swing at the pitch while the umpire looks on.

Object Detection from Video
With tracking,
Without tracking

超解像



Google ColabでDeep Learning

九州大学システム情報科学研究院
備瀬　竜馬

keras

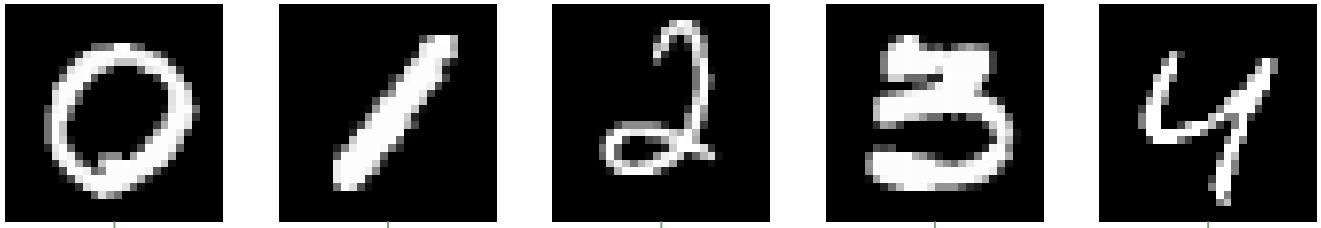
» Python で書かれた高水準のニューラルネットワークライブラリ

» クラスタリング処理をsklearn.clusterのKmeans関数で簡単にできるように、処理を全部自分書くことなく、簡単にライブラリを使って利用できる

MNIST：タスク

» 数字（0～9）の手書き文字の画像を0～9の10クラスに分類する課題
  – 入力：画像(28×28 pixel)
  – 出力：クラス（0～9）

入力画像



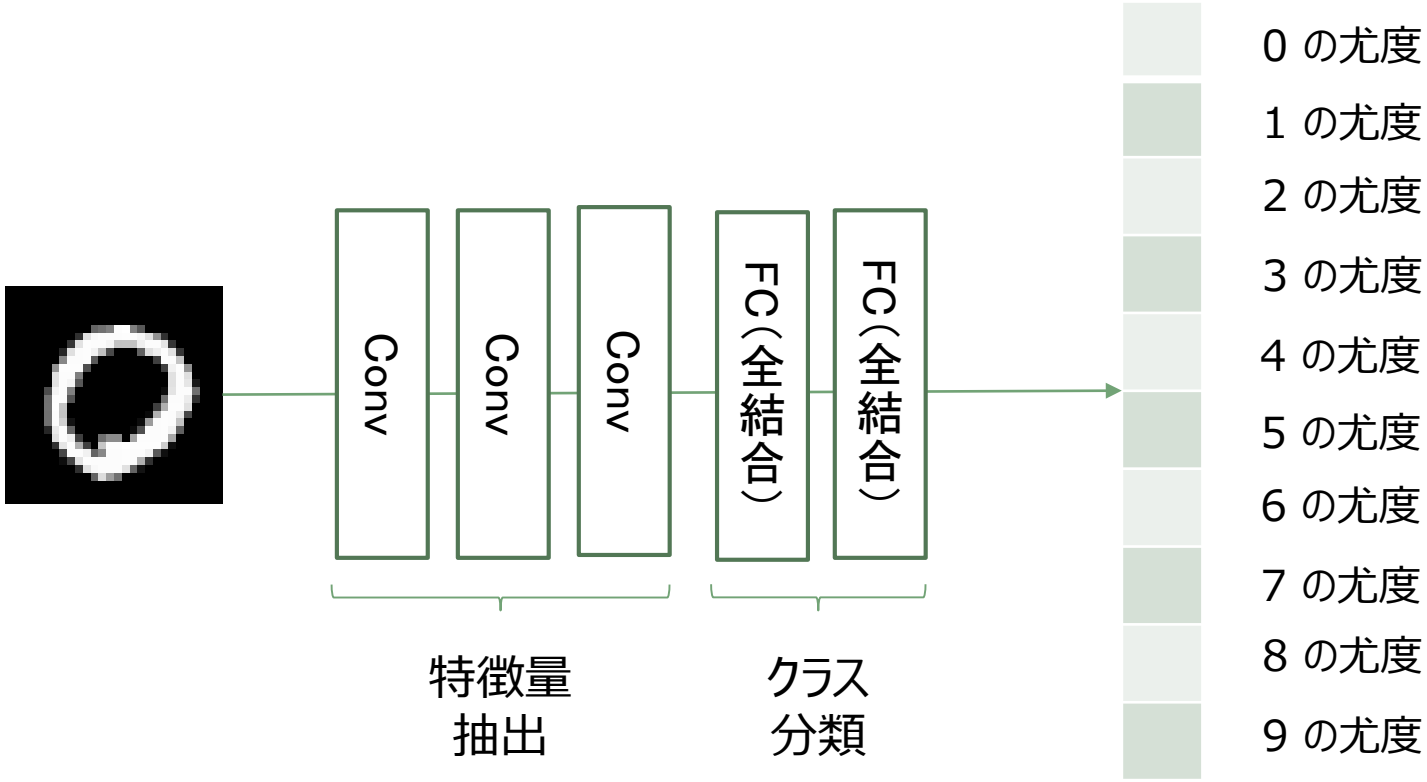
…

出力クラス(0～9)

0　1　2　3　4

MNIST：ネットワーク構造

» ネットワークの構造



Keras のコーディングの流れ

- データを用意
  - 画像とそのクラスラベル
- ネットワークモデルを構築
- モデルにデータを学習させる
- モデルを評価する

データの用意(1)

```
import keras
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
from keras import backend as K
batch_size = 128
num_classes = 10 # 0〜9までの手書き文字
epochs = 8 # 訓練データを何回繰り返して学習させるのか

img_rows, img_cols = 28, 28 # image size

# 学習データとテストデータに分割したデータ
(x_train, y_train), (x_test, y_test) = mnist.load_data()
```

必要なライブラリ

データの用意(2)

```
# backendがTensorflowとTheanoで配列のshapeが異なるために2パターン記述
print(K.image_data_format())
if K.image_data_format() == 'channels_first':
  # 1次元配列に変換
  x_train = x_train.reshape(s_train.shape[0], 1, img_rows, img_cols)
  x_test = x_test.reshape(x_test.shape[0], 1, img_rows, img_cols)
  input_shape = (1, img_rows, img_cols)
else:
  # 1次元配列に変換
  x_train = x_train.reshape(x_train.shape[0], img_rows, img_cols, 1)
  x_test = x_test.reshape(x_test.shape[0], img_rows, img_cols, 1)
  input_shape = (img_rows, img_cols, 1)

# 入力データ[0, 1]の範囲に正規化
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
# 255で割ったものを新たに変数とする
x_train /= 255
x_test /= 255
```

データの用意(3)

```
print('x_train shape : ', x_train.shape)
print(x_train.shape[0], 'train samples')
print(x_test.shape[0], 'test samples')

# ラベルをOne-Hotベクトルで表現
"""
例えば、サンプルに対するターゲットが「5」の場合次のような形になります。
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
"""
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)

# 学習時間を短くするため、講義用に小さいデータを作成(1000サンプル)
y_train_org = y_train
x_train_org = x_train
N = 1000
inds = range(N)
y_train = y_train[inds][:]
x_train = x_train[inds][:]
print(y_train.shape)
```

ネットワークの構築

```
# CNNネットワークの構築
model = Sequential()
# 3×3のカーネルサイズの2D Convolution layer
model.add(Conv2D(32, kernel_size=(3, 3),
                 activation='relu',
                 input_shape=input_shape)) model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2))) # max pooling layer
model.add(Flatten())
model.add(Dense(128, activation='relu')) # 全結合層
model.add(Dense(num_classes, activation='softmax'))

# 損失関数,最適化関数,評価指標を指定してモデルをコンパイル
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adadelta(),
              metrics=['accuracy'])
```

Model の学習

```
# モデルの学習
model.fit(x_train, y_train,
          batch_size=batch_size,
          epochs=epochs,
          verbose=1,
          validation_data=(x_test, y_test))
```

Modelの概要

モデルの概要を表示

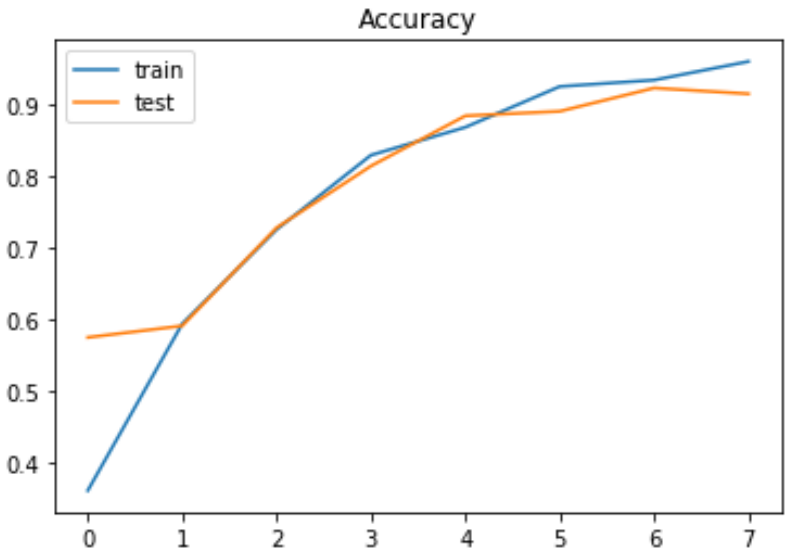
model.summary()

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 26, 26, 32)        320
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 24, 24, 64)        18496
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 22, 22, 64)        36928
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 20, 20, 64)        36928
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 10, 10, 64)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 6400)              0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               819328
_________________________________________________________________
dense_2 (Dense)              (None, 10)                1290
=================================================================
Total params: 913,290
Trainable params: 913,290
Non-trainable params: 0
_________________________________________________________________
```

モデルの評価

```
# モデルの評価
score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])

import matplotlib.pyplot as plt
# 学習をグラフ化（正解率）
acc = model.history.history['acc']
val_acc = model.history.history['val_acc']
```

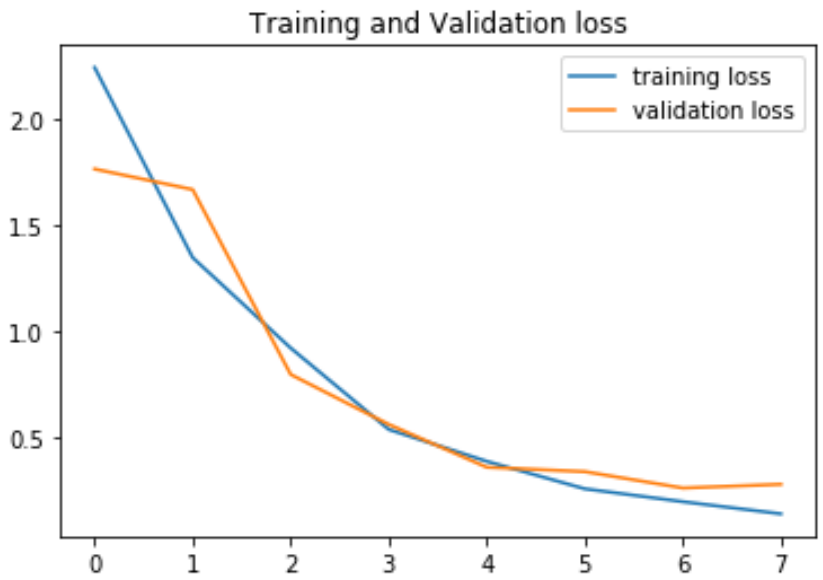


Test loss: 0.2760849190503359

Test accuracy: 0.9158

モデルの評価

```
# Loss Plot
loss = model.history.history['loss']
val_loss = model.history.history['val_loss']
plt.plot(loss ,label = 'training loss')
plt.plot(val_loss, label= 'validation loss')
plt.title('Training and Validation loss')
plt.legend()
plt.show()
```



1サンプルごとにクラス予測

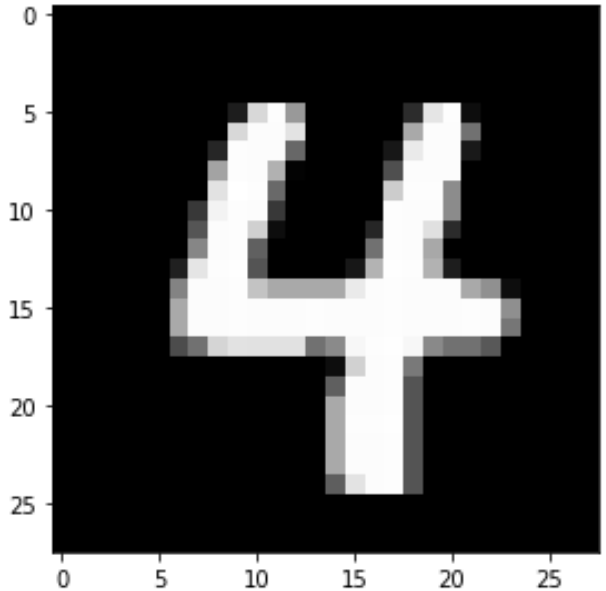
```
# 1つのサンプルデータを用いてテスト
from skimage import io
import numpy as np
import random
# test dataをランダムに取得
ind = np.int16(np.round(random.random()*x_test.shape[0]))
print(ind)
x_sample = x_test[ind]
y_sample = y_test[ind]

# 画像を可視化
x_sample2 = x_sample.reshape(x_sample.shape[0], x_sample.shape[1])
print(y_sample)
io.imshow(x_sample2)

# 予測関数入力用に変形
x_sample = x_sample.reshape(1, img_rows, img_cols, 1)

# ネットワークによる予測
expect = model.predict(x_sample)
print(np.round(expect))
```

ランダムID 6138
正解 [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
予測 [[0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]]



演習問題1

» Conv層を一つずつ変えて、精度の違いを比べよ
  - Conv層：1層
  - Conv層：2層
  - Conv層：3層
  - Conv層：4層
  - Conv層：5層
  - Conv層：6層

演習問題2

» クラスを奇数・偶数の2ラベル認識問題として学習し、精度評価を行え。

» クラス１：１、３、５,7,9

» クラス２：0,2，４、６,8