

Convolutional Neural Network (CNN)

Deep-people #5

前回のおさらい

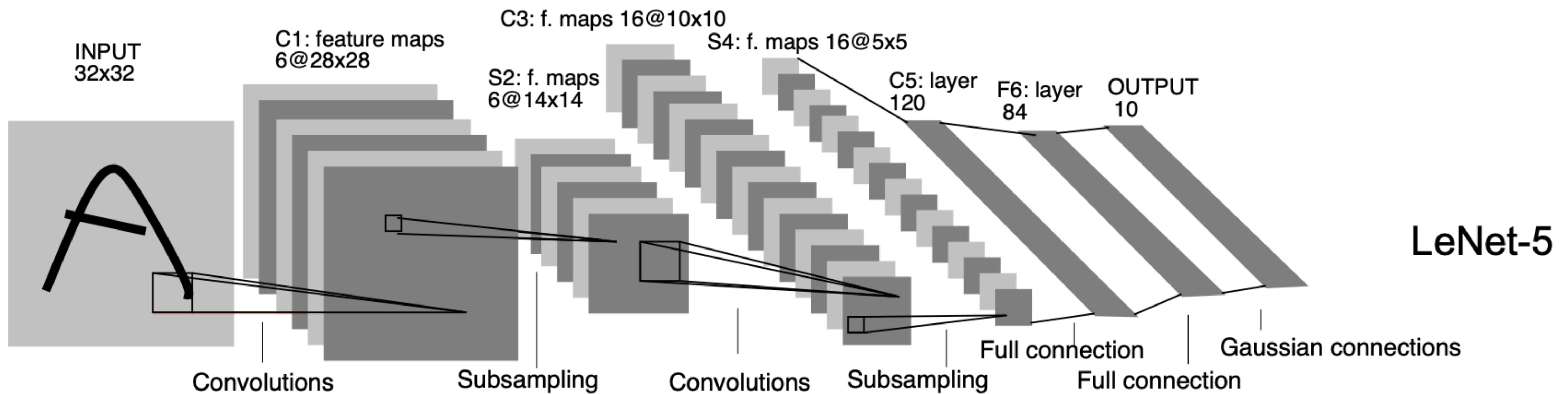
- 深層学習のまとめ
 - パーセプトロン -> ニューラルネットワークのしくみ
 - 活性化関数
 - 学習 (勾配降下法, 誤差逆伝播 etc)
 - PyTorchの基礎

今日の参考資料

- Juhanの資料
- [https://mac.kaist.ac.kr/~juhan/gct634/Slides/\[week5-1\]%20convolutional%20neural%20network.pdf](https://mac.kaist.ac.kr/~juhan/gct634/Slides/[week5-1]%20convolutional%20neural%20network.pdf)
- Convolutional Neural Networks cheatsheet
- <https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks>

今回のお話

- 置み込みニューラルネットワーク (以降, CNN)
- 画像処理で成功を遂げたニューラルネットワーク



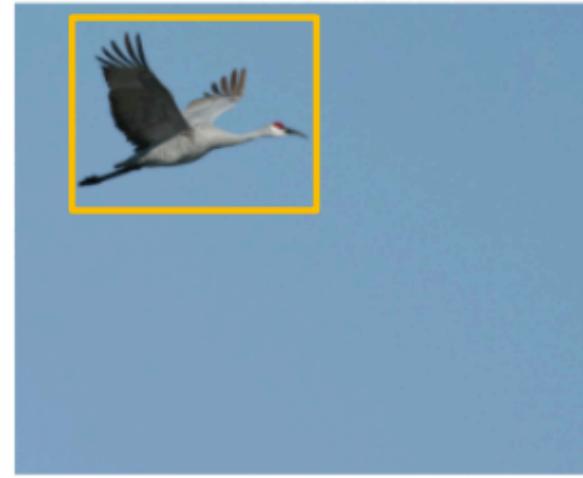
画像データや音データなどのセンサーデータにNNは非効率的

5

- 画像・音のデータはいわゆる信号, 高い次元を持っている
 - 画像 : $256 * 256$
 - 音 : 3秒のメルスペクトログラムで $128 * 300$
- NN (線形結合 + 活性化関数) を適用すると, パラメータ数が非常に多くなる
 - 隠れ層のサイズが 256 とすると, 256×256 のカラー画像なら, 5000 万もパラメータが必要
 - このままだと過学習するのは火を見るより明らか
- なんとかいい方法でパラメータを減らせないだろうか?

局所性

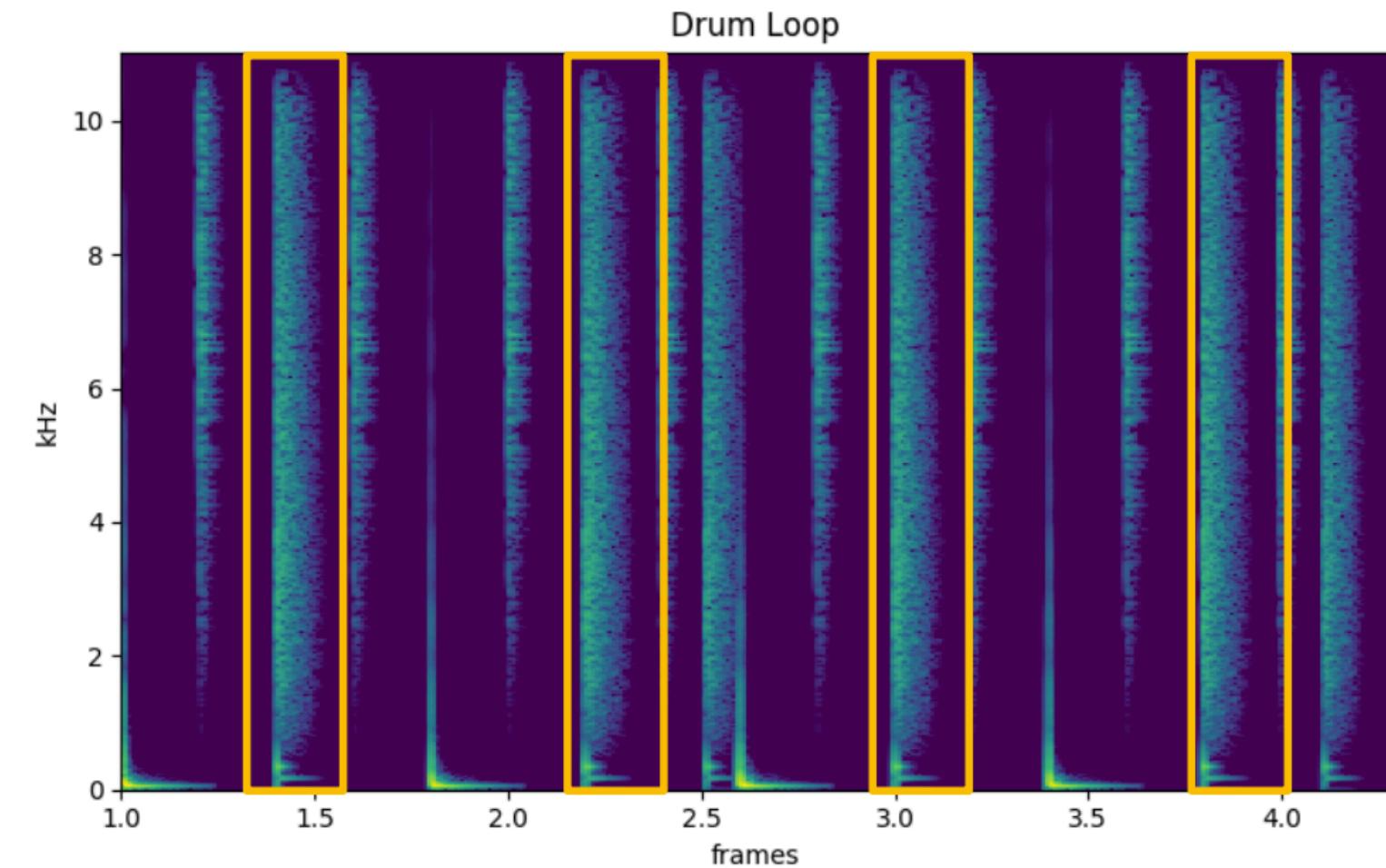
対象となる物体は空間的にまとまる



- ・鳥がいる領域は画像全体には散らばらない
- ・どこにいても鳥は鳥と認識できる

位置不变性

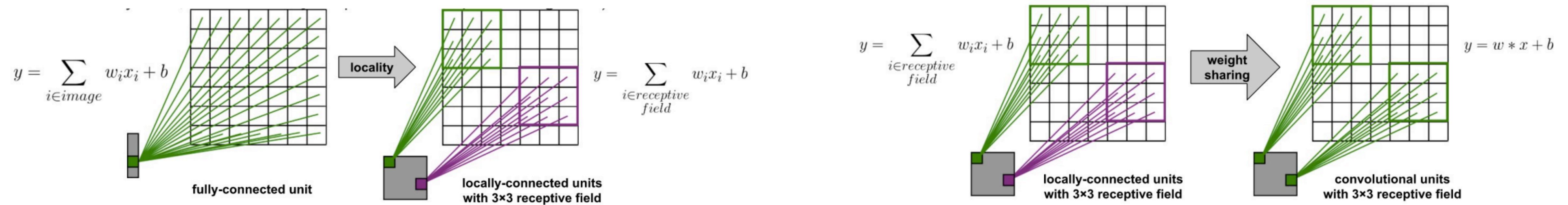
対象は位置に独立：
どこにあろうが対象 자체が変わることはない



- ・ドラム音は特定時間、周波数帯域にまとまりを見せる
- ・どの時間にならそうが音響的性質は不变

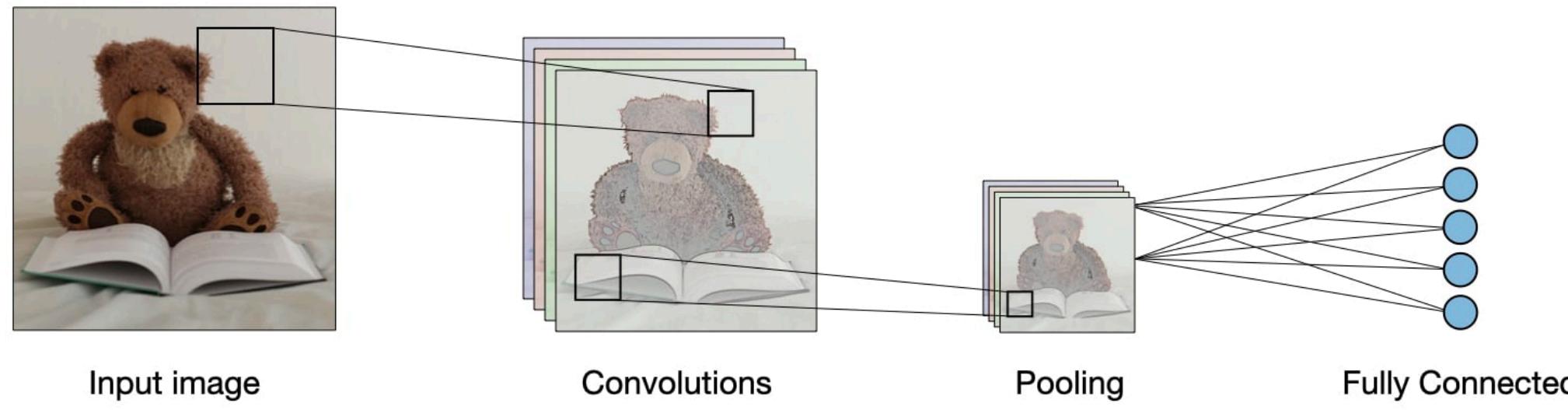
局所的な情報に強く着目する！

- 一定の着目領域（受容野）のみに対して線形演算を行うようにする
- さらに、重みを位置で不变に（共有）すればさらにパラメータが減る！
- これが畳み込み演算と同等に！



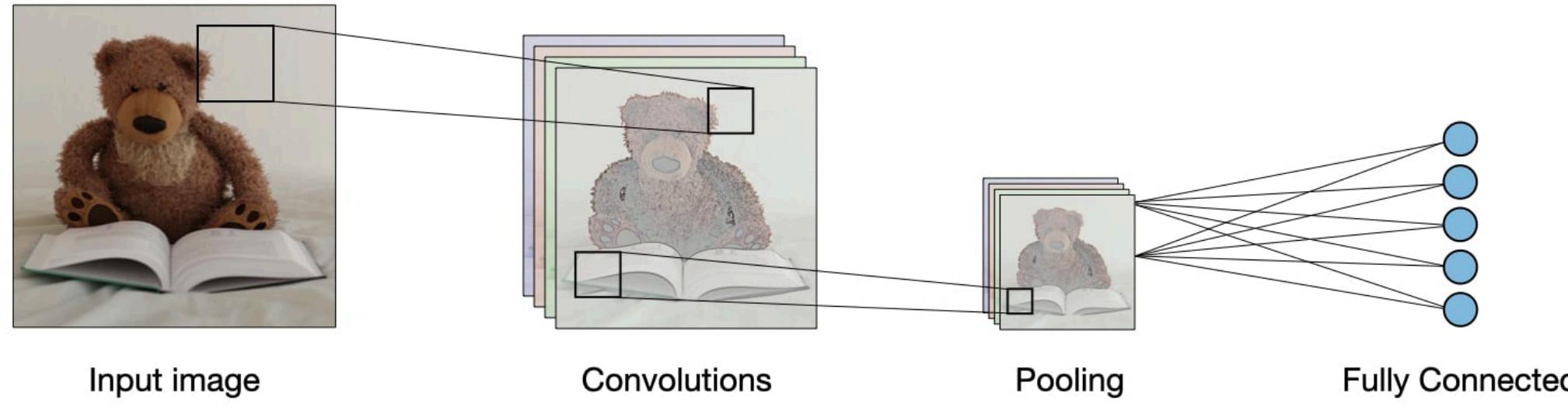
CNNの持つ機構について

8



- **Convolution層**
 - 置み込み計算
 - ストライド
 - パディング
- **Pooling層**
 - ダウンサンプリングの方法
- **全結合層**
 - CNN -> 特徴抽出, 全結合層を識別器に

CNNの持つ機構について



- **Convolution層**

- 署み込み計算
- ストライド
- パディング

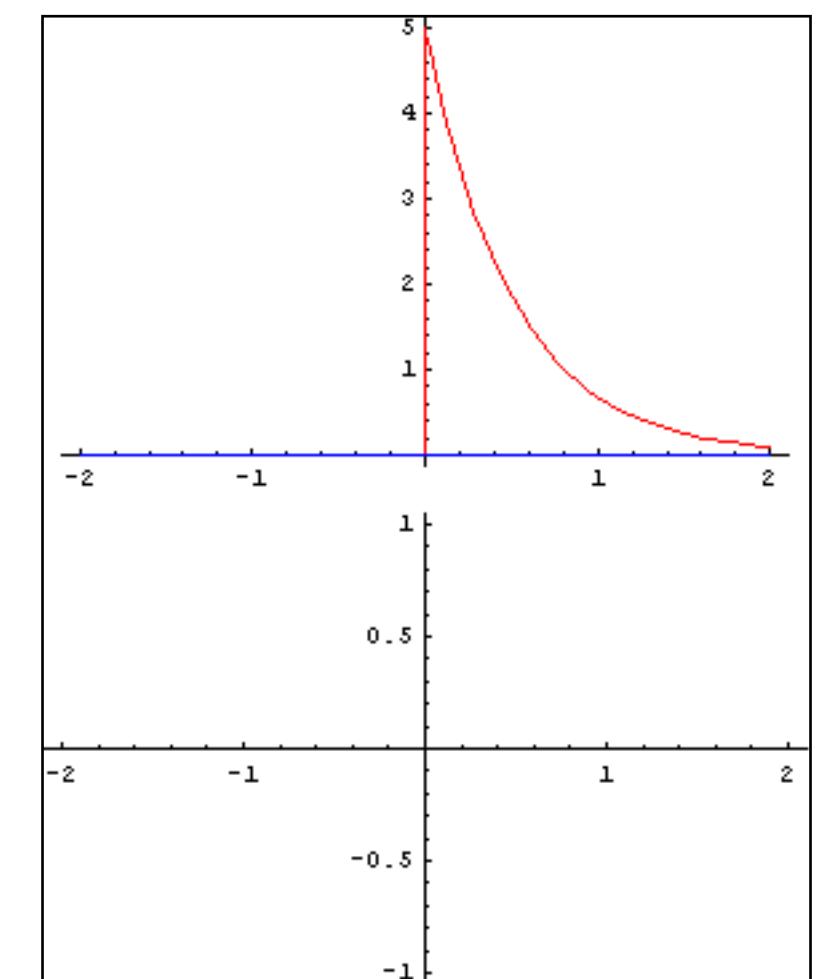
- **Pooling層**

- ダウンサンプリングの方法
- **全結合層**
- CNN -> 特徴抽出, 全結合層を識別器に

- ・ 疋み込みとは
 - ・ 2つの関数を、時間ずらしながら内積を計算する
 - ・ <https://ja.wikipedia.org/wiki/%E7%95%B3%E3%81%BF%E8%BE%BC%E3%81%BF>

連続 $(f * g)(t) = \int f(\tau)g(t - \tau) d\tau$
ずらし時間

$$\text{離散} \quad (f * g)(m) = \sum_n f(n) g(m - n)$$



畳み込み層 (Convolutional layer)

11

- 入力データに対し「カーネル」を畳み込み,
特徴マップを出力する
 - カーネルで局所性の情報を得る

入力	カーネル	出力																							
<table border="1"><tr><td>7</td><td>2</td><td>3</td><td>3</td><td>8</td></tr><tr><td>4</td><td>5</td><td>3</td><td>8</td><td>4</td></tr><tr><td>3</td><td>3</td><td>2</td><td>8</td><td>4</td></tr><tr><td>2</td><td>8</td><td>7</td><td>2</td><td>7</td></tr><tr><td>5</td><td>4</td><td>4</td><td>5</td><td>4</td></tr></table>	7	2	3	3	8	4	5	3	8	4	3	3	2	8	4	2	8	7	2	7	5	4	4	5	4
7	2	3	3	8																					
4	5	3	8	4																					
3	3	2	8	4																					
2	8	7	2	7																					
5	4	4	5	4																					

 | | | | |---|---|----| | 1 | 0 | -1 | | 1 | 0 | -1 | | 1 | 0 | -1 | | | | | | |---|--|--| | 6 | | | | | | | | | | | || * | | = |
| | | |

$7 \times 1 + 4 \times 1 + 3 \times 1 + 2 \times 0 + 5 \times 0 + 3 \times 0 + 3 \times -1 + 3 \times -1 + 2 \times -1 = 6$

- 各ピクセル値とカーネルを乗算
 - > 掛け算した部分の和を取る
 - > ずらす・・・の繰り返し

<https://medium.datadriveninvestor.com/convolutional-neural-networks-3b241a5da51e>
より

畳み込み演算による空間的特徴抽出

12

- カーネルの重みwの組み合わせによって、様々な特徴を捉えられる



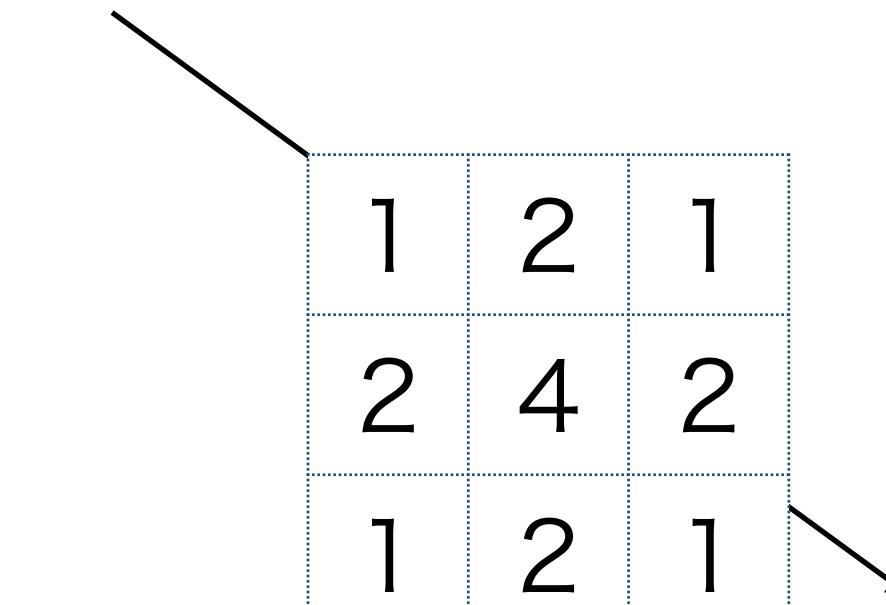
0	1	0
1	-4	1
0	1	0



輪郭

- 上：ラプラシアンフィルタ
- 下：ガウシアンフィルタ

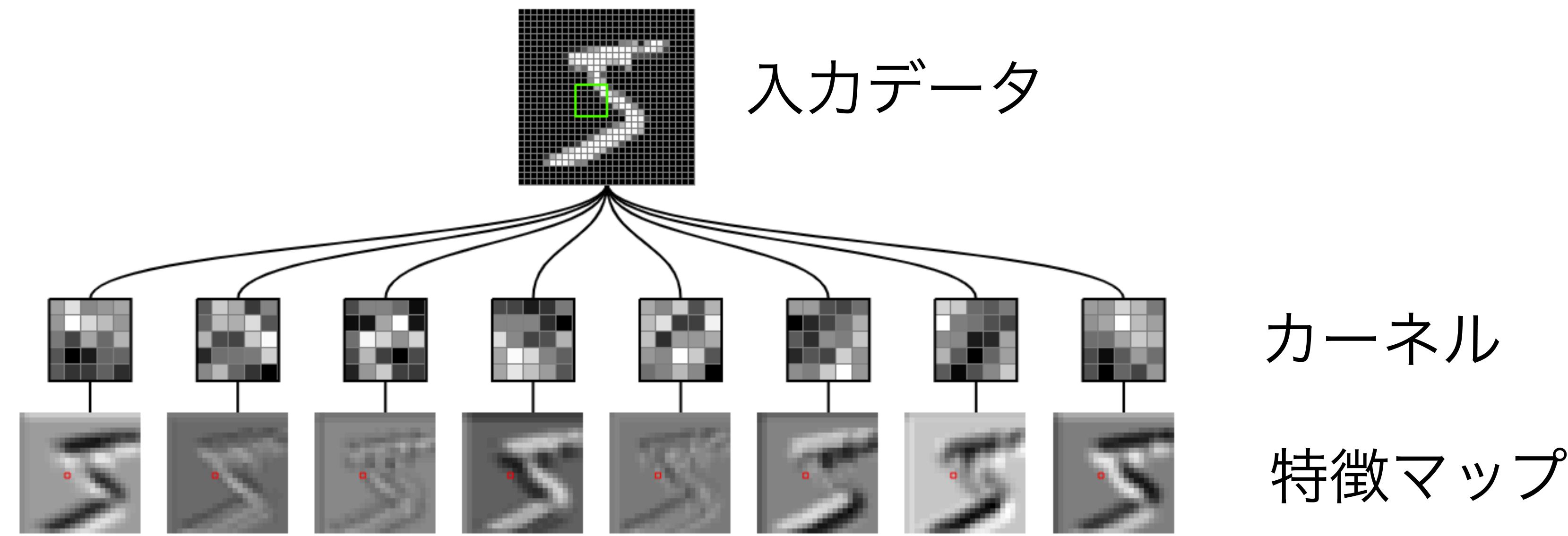
1	2	1
2	4	2
1	2	1



ぼかし

<https://algorithm.joho.info/image-processing/laplacian-filter/>
<https://algorithm.joho.info/image-processing/gaussian-filter/>

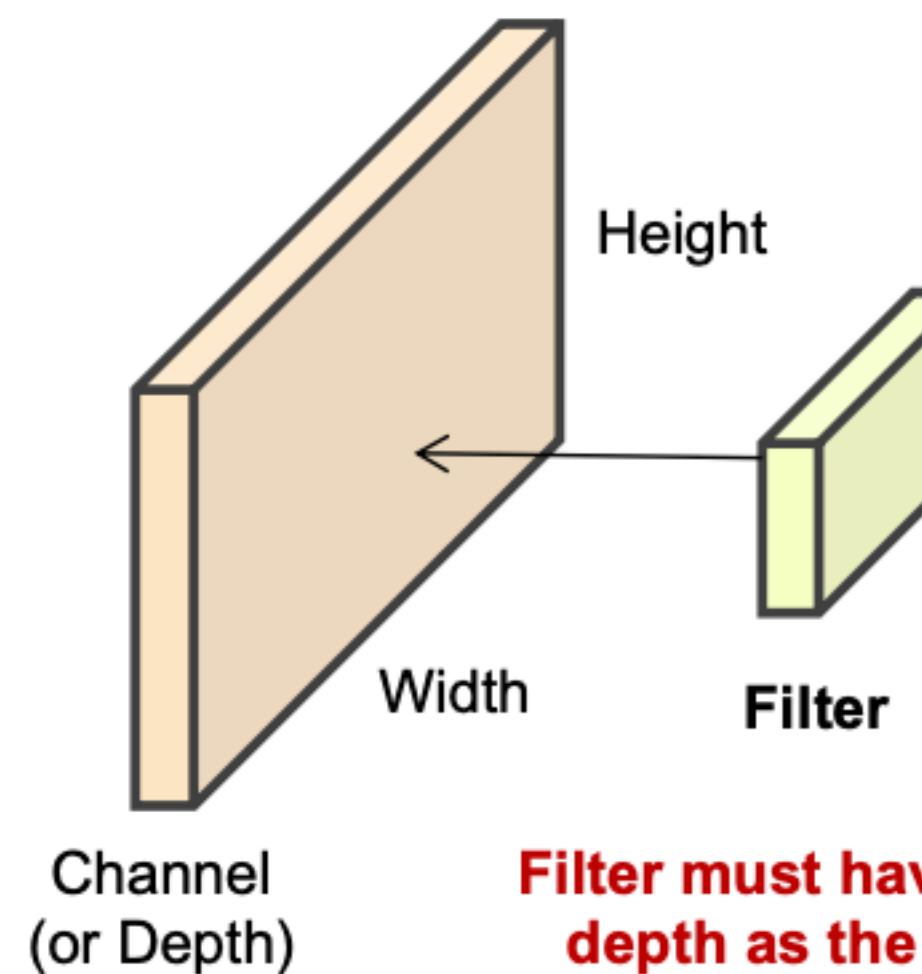
- カーネルを複数持たせ、複数の特徴量（特徴マップ）を得る
- カーネルの値を学習させて適応的な学習を実現する



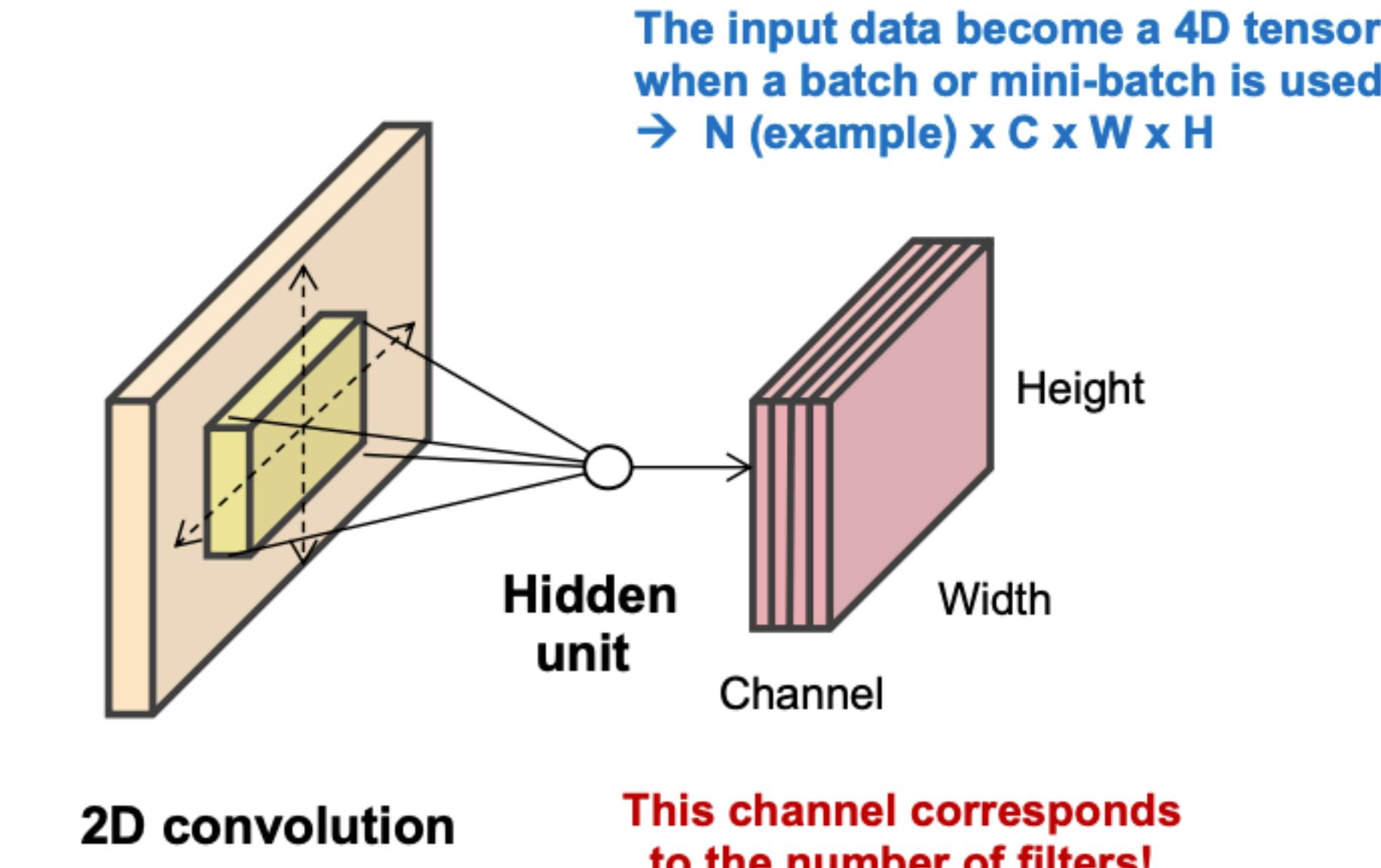
<https://ml4a.github.io/ml4a/jp/convnets/>

- 2次元畳み込みの場合、入力はChannel, Height, Widthの3次元
 - Channel数：入力がカラー画像：RGBで3, スペクトログラム：1
 - ミニバッチ学習を適用する場合は (Batch, Channel, Height, Width) の4次元
- 出力Channel数は適用させるカーネルの枚数に一致

○ Channel (C): R, G, B



Filter must have the same depth as the input has

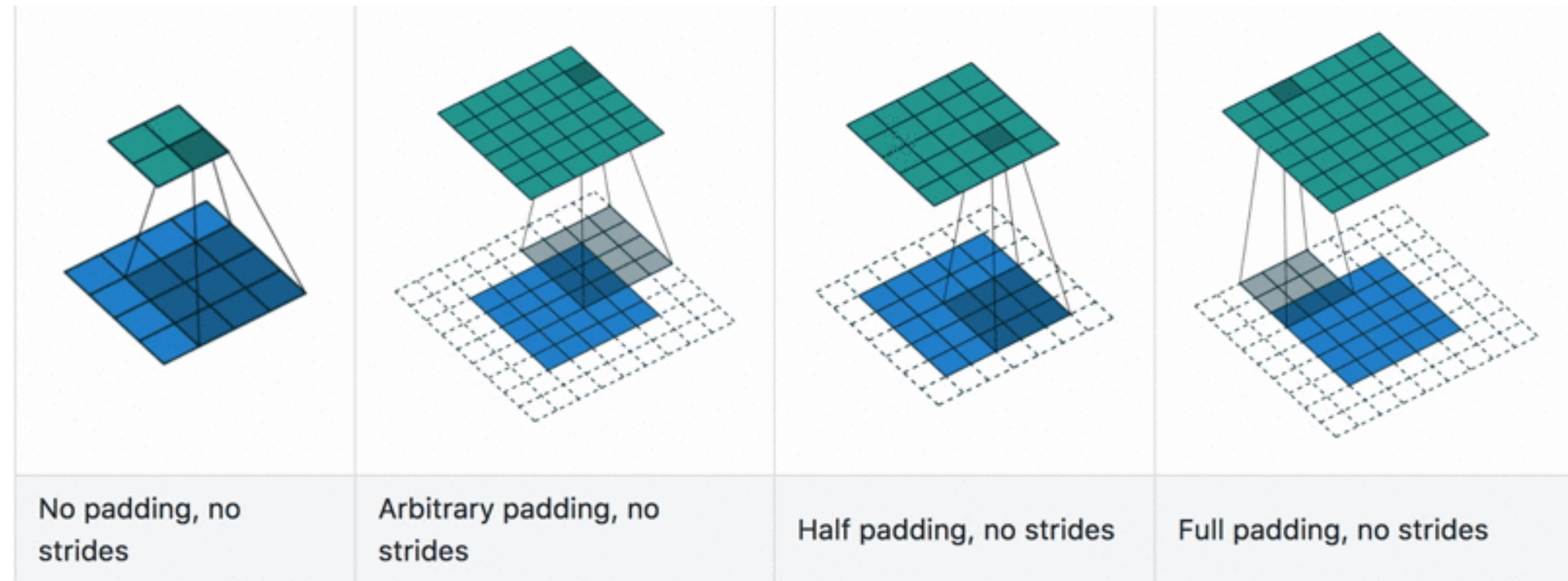


This channel corresponds to the number of filters!

ストライド (Stride) とパディング (Padding)

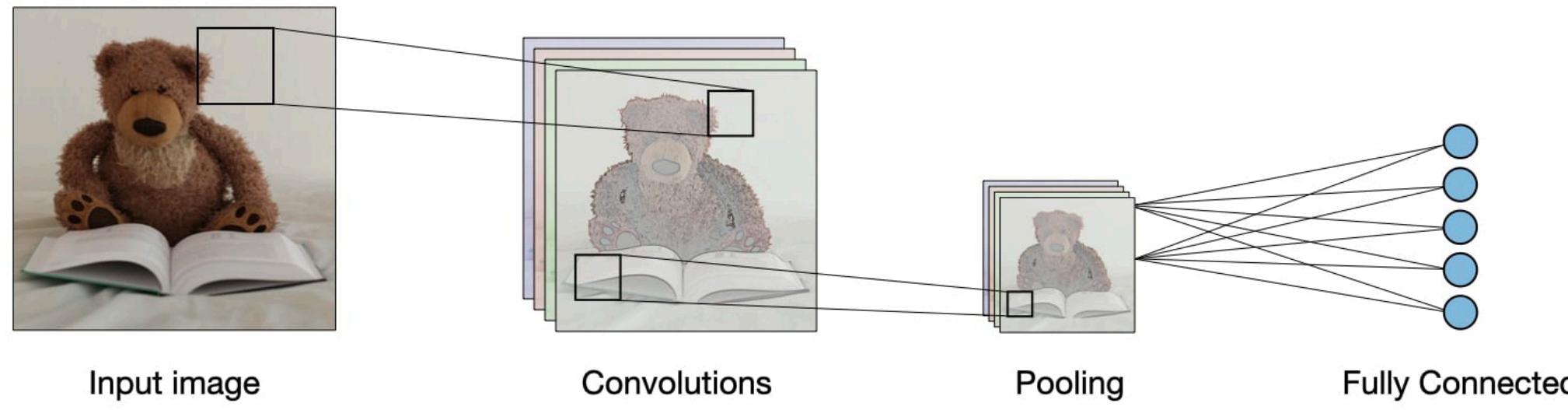
15

- ストライド：カーネルをずらす幅
- パディング：入力サイズを大きくするため、周りに0の新しいセルで囲む



CNNの持つ機構について

16



- **Convolution層**

- 置み込み計算
- ストライド
- パディング

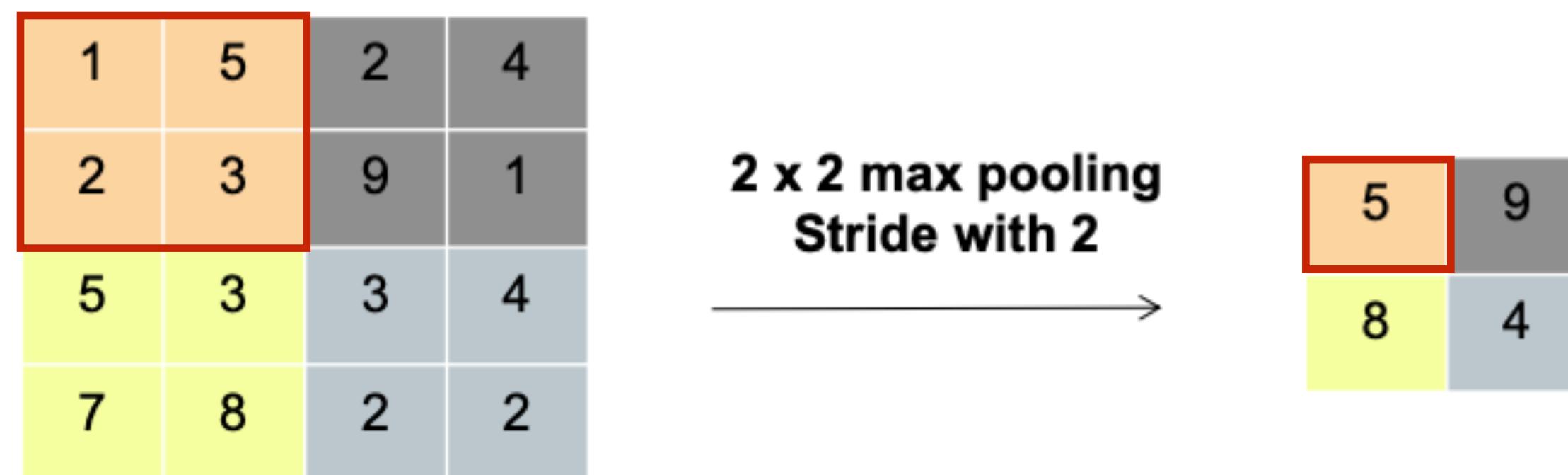
- **Pooling層**

- ダウンサンプリングの方法
- **全結合層**
- CNN -> 特徴抽出, 全結合層を識別器に

プーリング (Pooling)

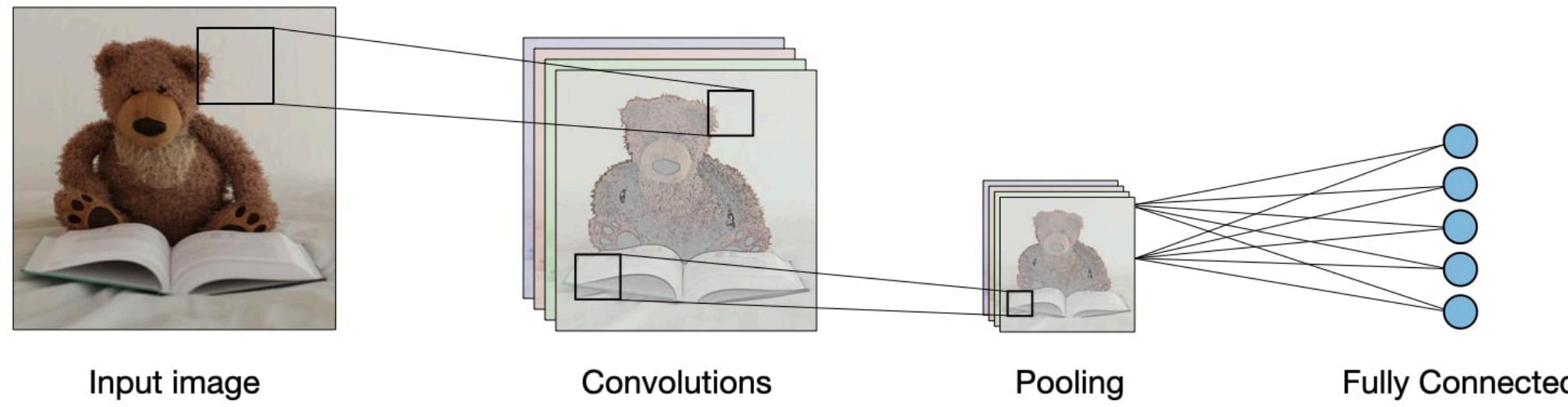
17

- 置み込みして得られた特徴マップをダウンサンプル
- 次元圧縮の役割を果たす→位置不变性の獲得に重要とされている
- Max pooling : 領域の最大値を採用
- Average pooling : 領域の2乗平均を利用



CNNの持つ機構について

18

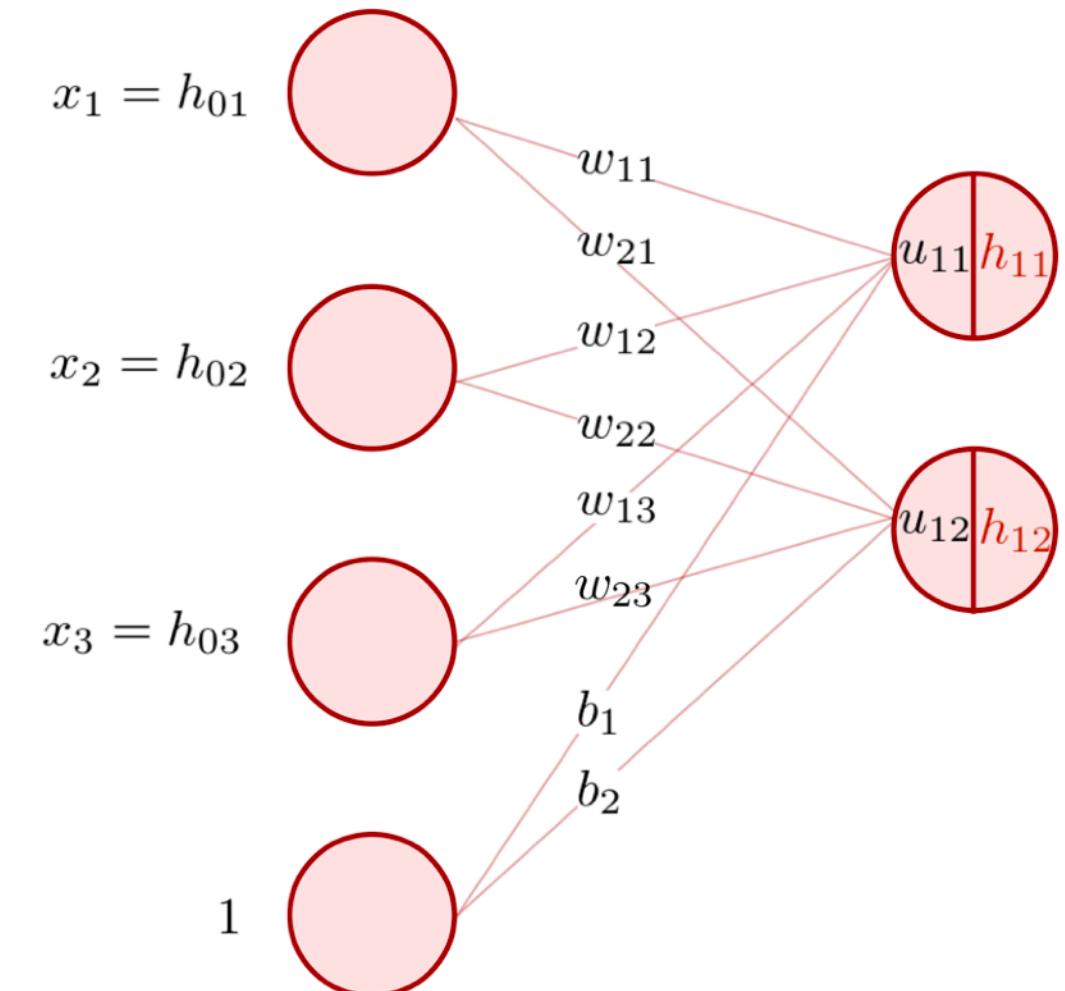


- **Convolution層**
 - 置み込み計算
 - ストライド
 - パディング
- **Pooling層**
 - ダウンサンプリングの方法
- **全結合層**
 - CNN -> 特徴抽出, 全結合層を識別器に

全結合層 (Fully-connected layer)

19

- 各セルが次の層のセルと全て繋がっている層
 - みんなが前回学んだアフィン変換を行う層のこと
 - もちろん活性化関数つき
 - Dense層とも
- いわゆる出力へ変換するための層
 - 所望の出力の次元に揃える
 - 分類問題：クラス数（値はクラスの確率）
 - 回帰問題：1（値は推定回帰値）etc...

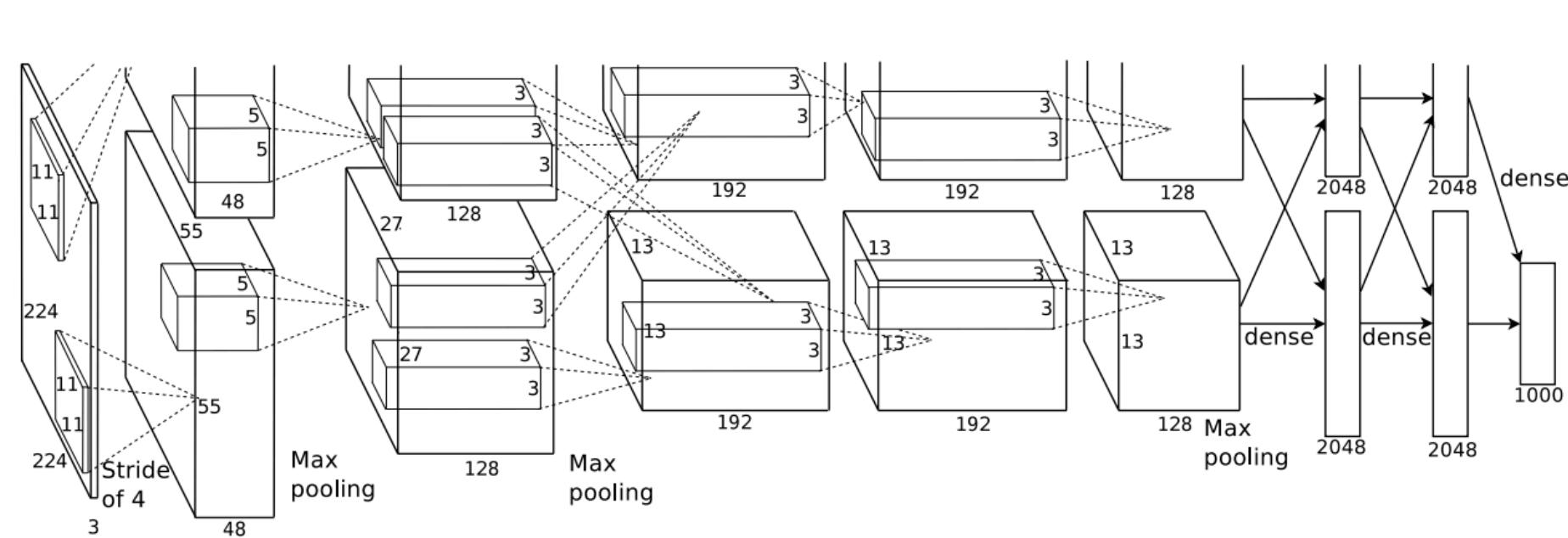


https://tutorials.chainer.org/ja/13_Basics_of_Neural_Networks.html

実際のモデルについて

画像分野での有名モデルその1

21

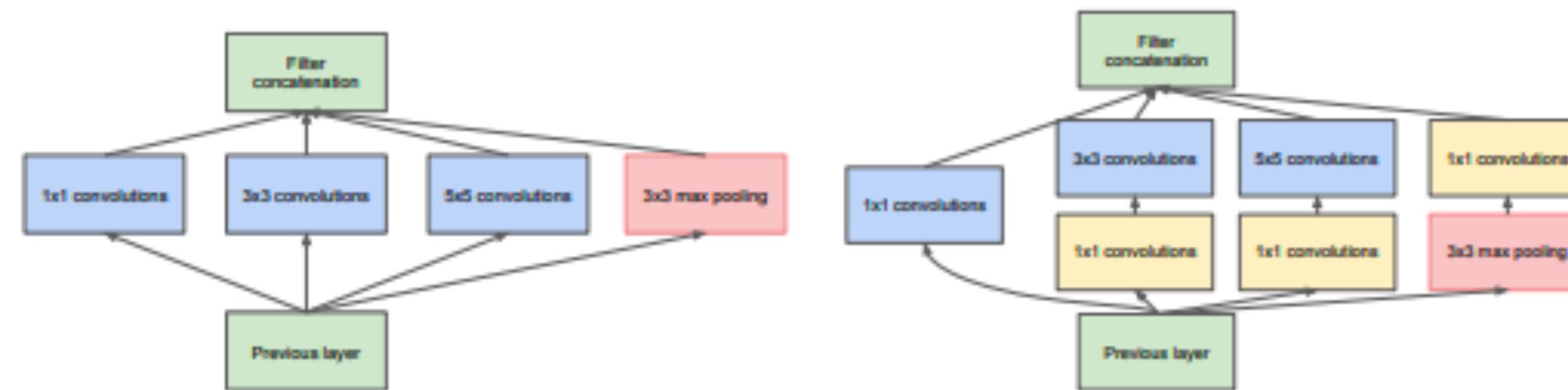


ImageNet classification with deep convolutional neural networks, Alex Krizhevsky, Ilya Sutskever, Geoffrey E Hinton, NIPS, 2012

AlexNet

2012年，層を深くして性能向上

画像分類のImagenet challengeにて2位以下に圧勝



(a) Inception module, naïve version

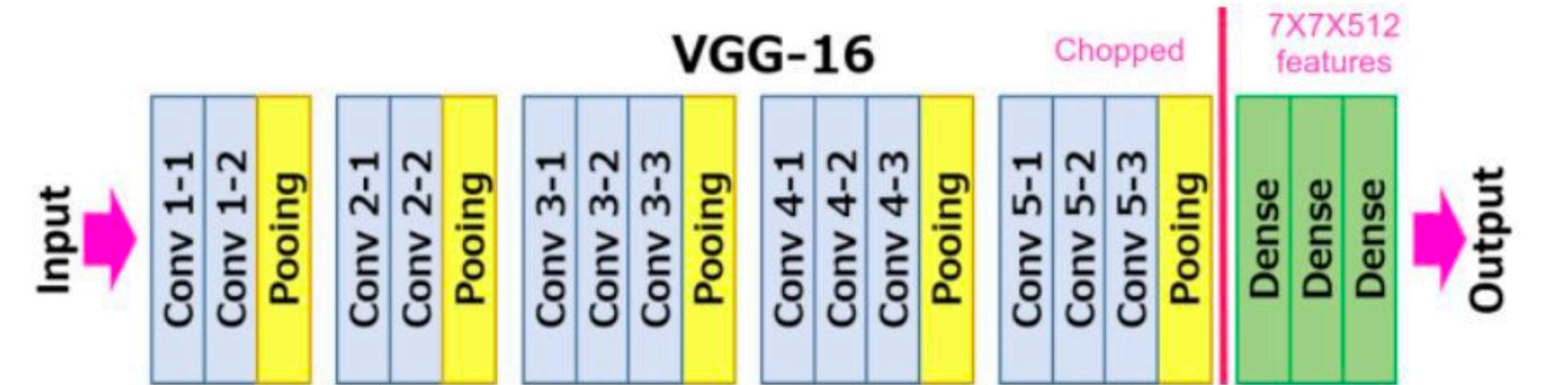
(b) Inception module with dimension reductions

Going Deeper with Convolutions, Szegedy et al., CVPR, 2015

GoogLeNet

2015年、より少ないパラメータで効率的に計算を行う

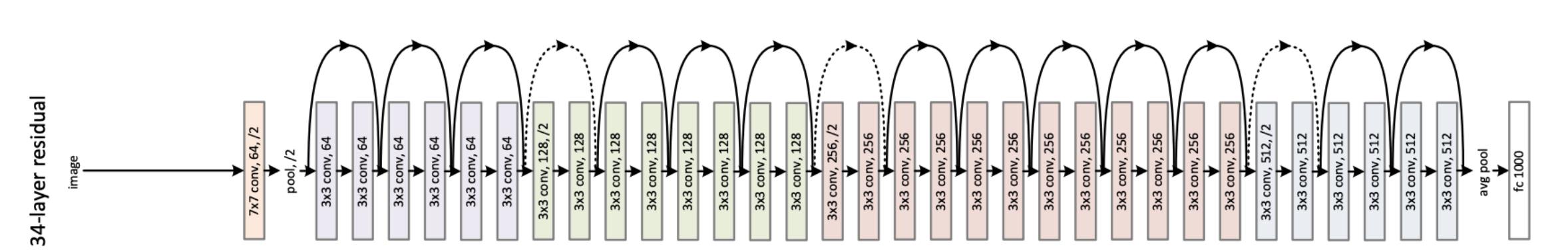
Inception機構を導入：サイズの小さい畳み込み or プーリング
を入力の一部に適用しパラメータを削減



Very Deep Convolutional Networks for Large-Scale Image Recognition, Karen Simonyan, Andrew Zisserman, ICLR, 2015

VGG

2015年, カーネルサイズを小さく (3×3) に, さらに深くすることによって精度向上多くの研究で最もシンプルなモデルとして広く用いられる



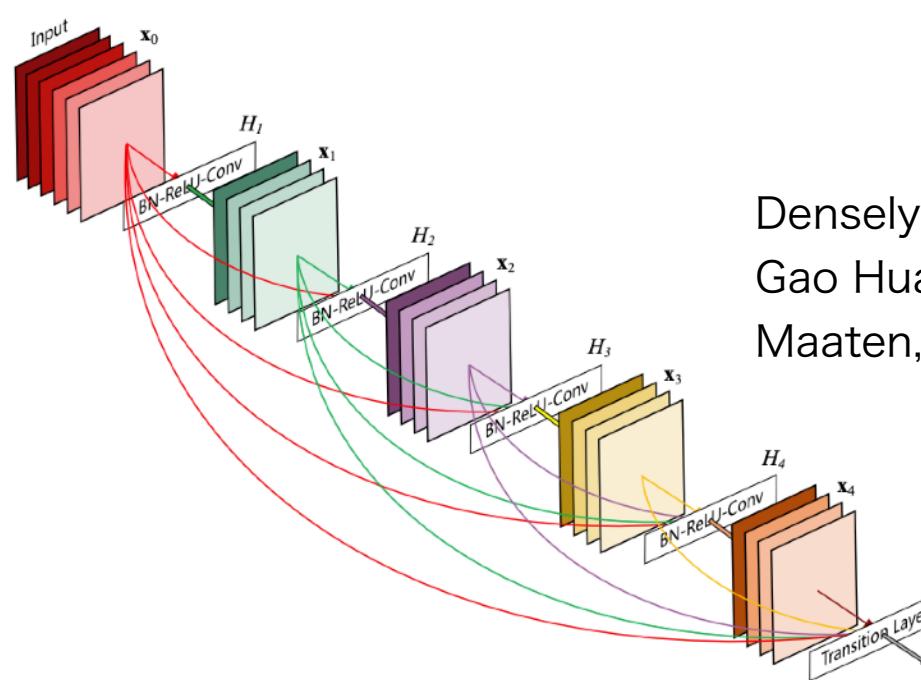
Deep Residual Learning for Image Recognition, Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, CVPR, 2016

ResNet

Skip-connection（層の出力に入力を足す）を導入し、層を深くしても勾配消失しないようにした

画像分野での有名モデル その2

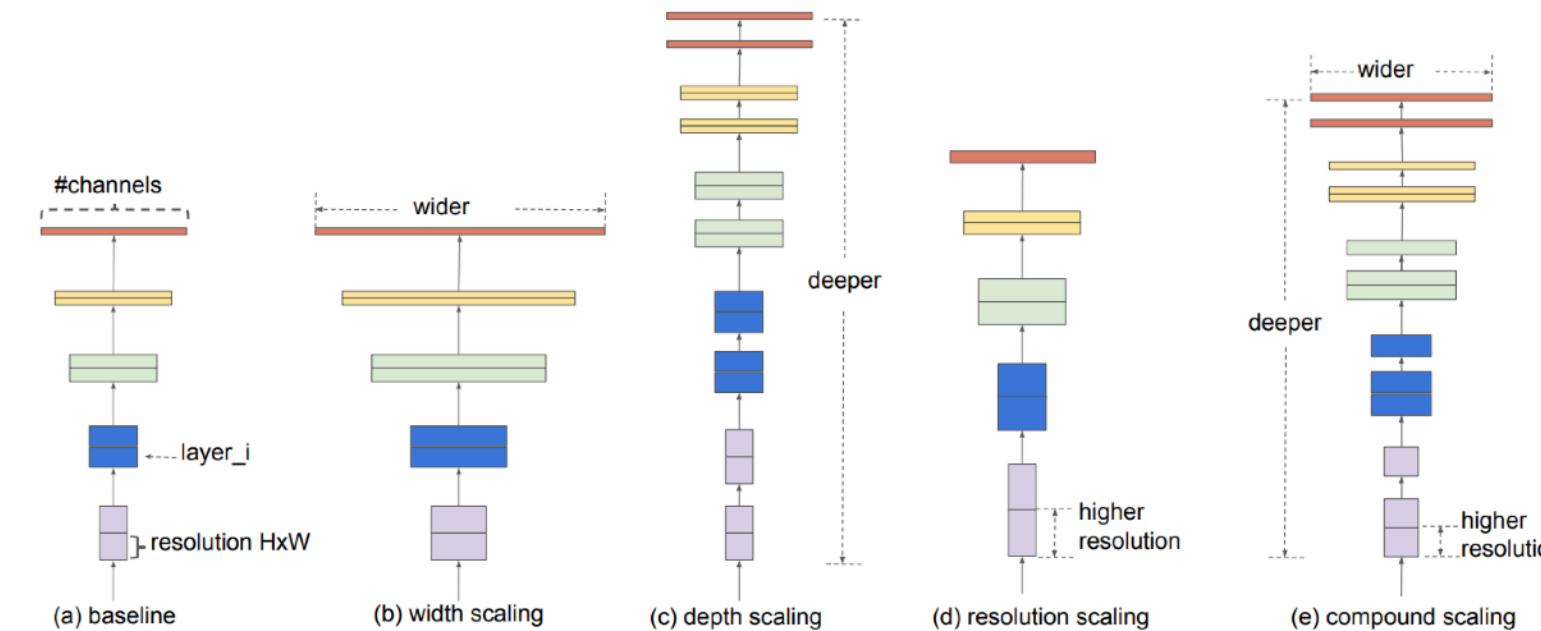
22



Densely Connected Convolutional Networks,
Gao Huang, Zhuang Liu, Laurens van der
Maaten, Kilian Q. Weinberger, CVPR, 2017

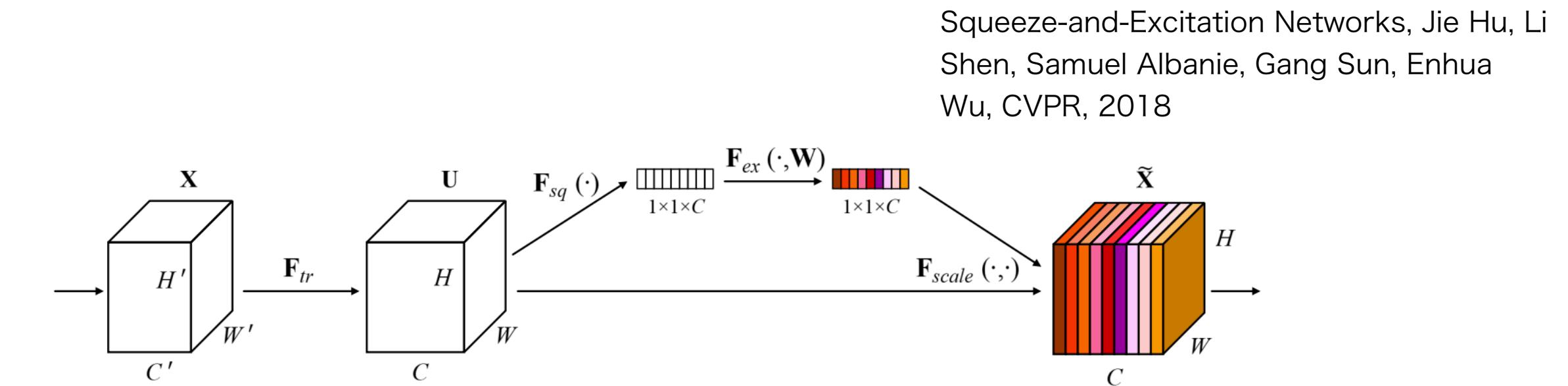
DenseNet

2017年, ResNetにインスピア, 全ての層の中間出力結合 (Concat) し, パラメータ削減



EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks
, M.Tan et al. ICML, 2019

2019年, 真の効率性のためにCNNをどう設計すればいいか?
層の深さ, カーネルの広さ, 入力の解像度等をチューニング
スケールアップ時, 各要素をとある定数で定数倍すればいい



Squeeze-and-Excitation (SENet)

2018年, 特徴マップのうち有用なものに重みをつける
ゲート機構を追加して性能向上

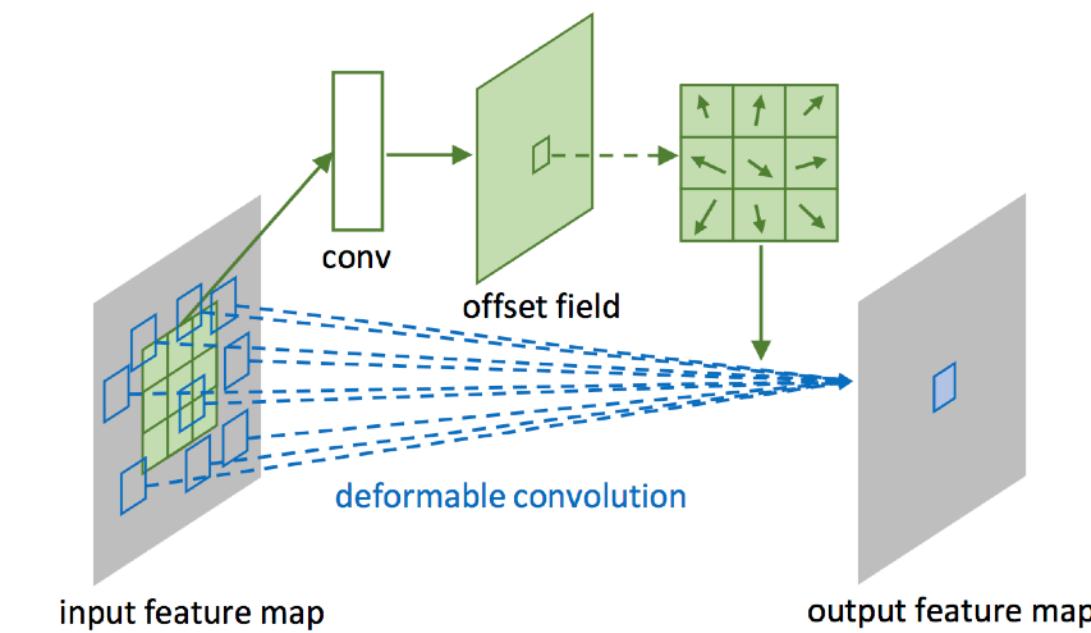


Figure 2: Illustration of 3×3 deformable convolution.



Deformable Convolution Networks, J.Dai et al. , ICCV, 2017

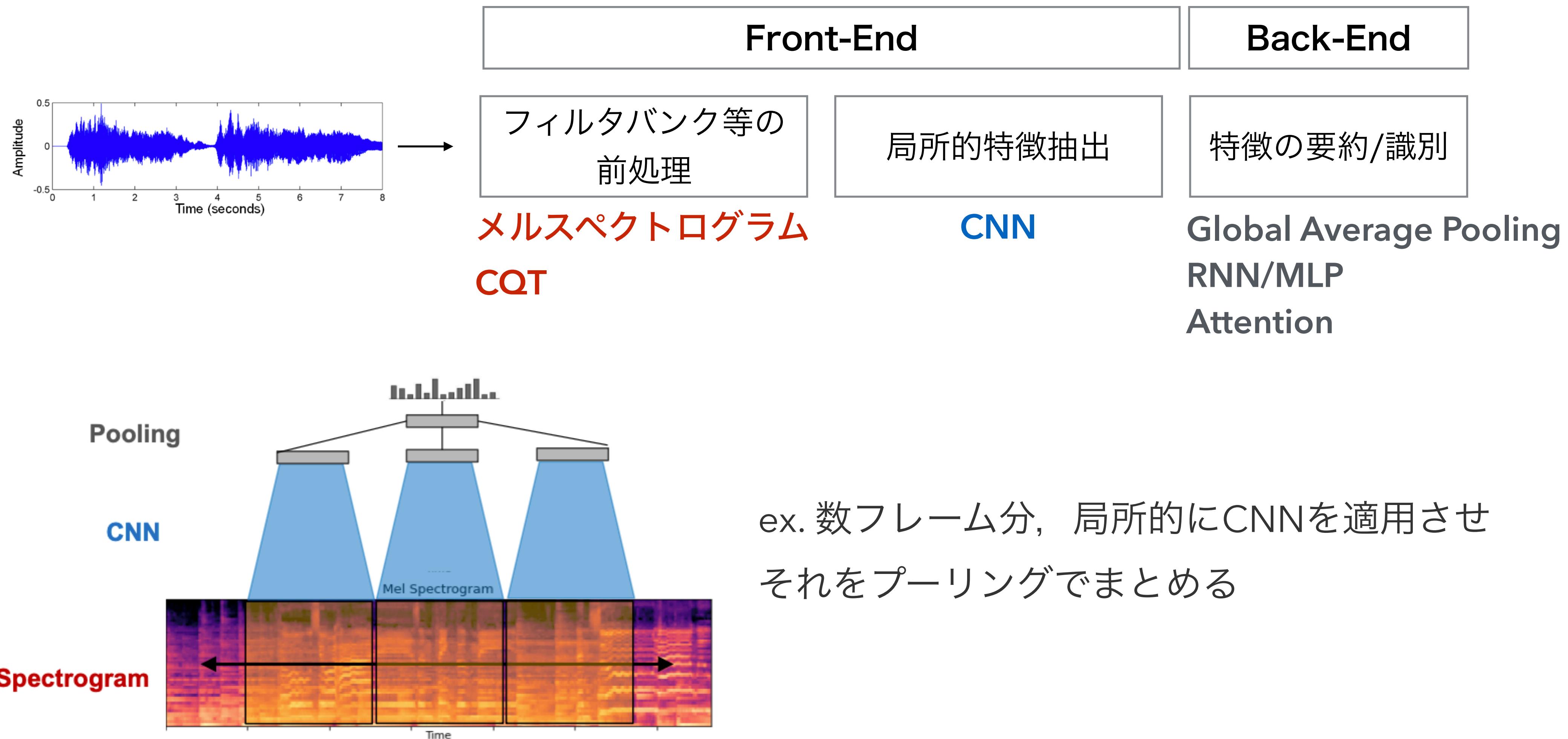
Deformable Convolution

2017年, 物体の形状の多様性に対応するために
カーネルをバラバラにする

音響データにおけるCNNの適用方法

23

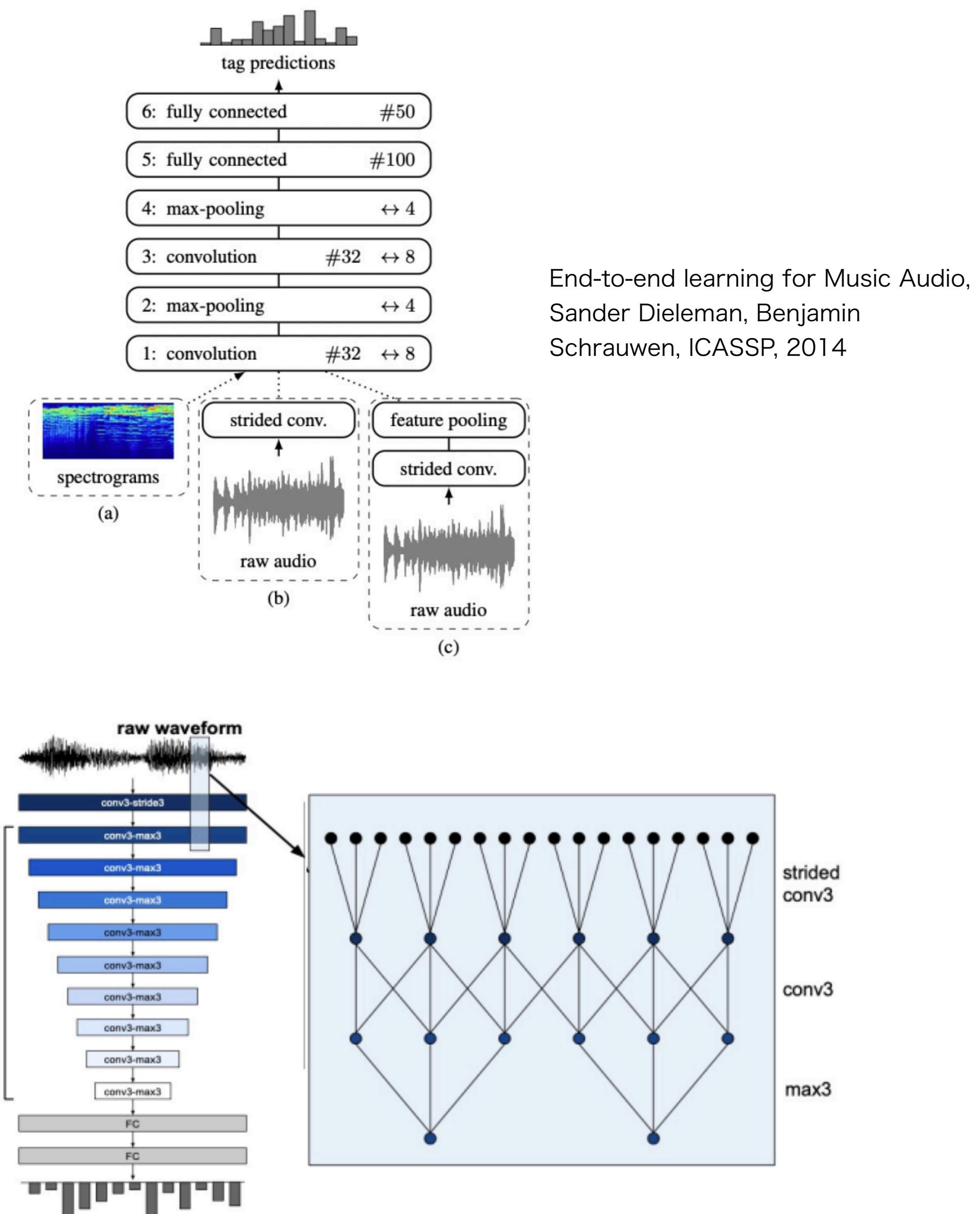
- 音響の識別問題のステップにおける、**特徴抽出**に用いる



音におけるモデル①：波形+1DCNN

24

- 1次元 (1D) CNNを用いて信号波形を入力
- フレームレベルのカーネル次元 (256, 512, 1024)
ではメルスペクトログラムより性能が低かった
- サンプルレベルまで落として、(3, 9) 層を深くすれば性能は向上 (SampleCNN)
- データ数が多いほど波形入力は有効



音におけるモデル② 2D CNN + スペクトログラム

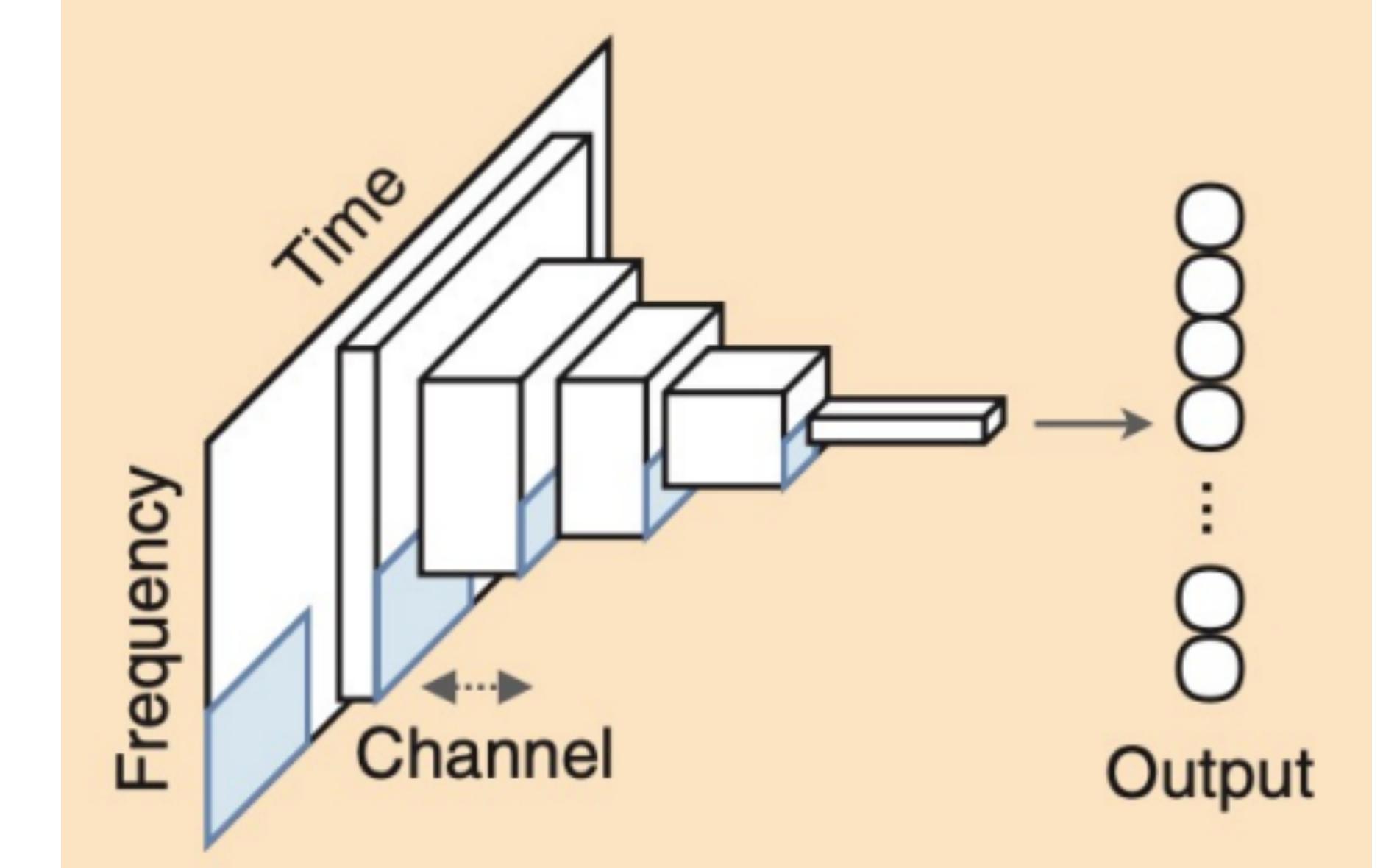
25

- 先ほどまで説明した画像 + 2次元カーネルを持つCNN
(2DCNN) の適用

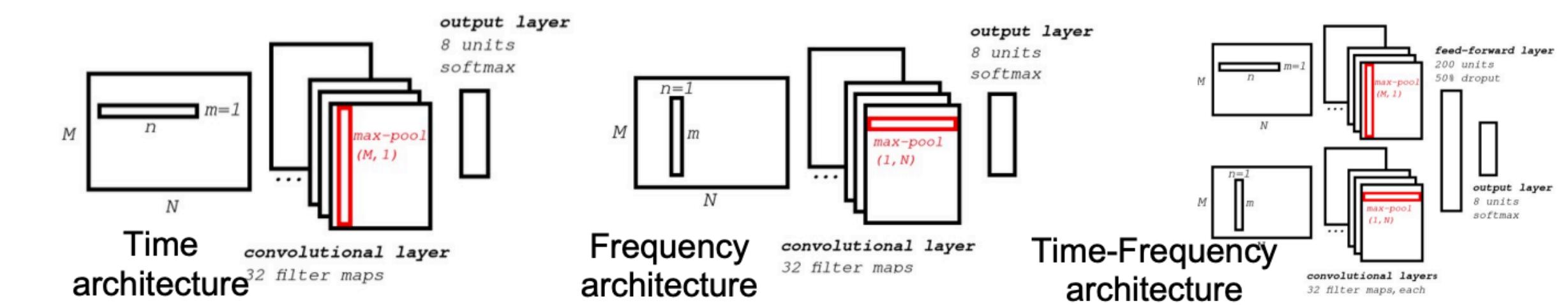
- スペクトログラムを2次元のベクトルとみなし、
1Channel画像とみなし処理

- 多くの場合1DCNNより性能が良くなる

- 縦 (H) を周波数、横 (W) を時間次元とする
- この構造に着目し、正方形でないカーネルを持つ
CNNも提案してきた



Deep Learning for Audio-Based Music Classification and Tagging, Juhua Nam, Keunwoo Choi, Jongpil Lee, Szu-Yu Chou, and Yi-Hsuan Yang, IEEE SPM, 2018



Experimenting with musically motivated convolutional neural networks, Jordi Pons, Thomas Lidy, Xavier Serra, CBMI, 2016

- CNNについて解説
 - 高い次元をもつセンサーデータに対し、局所性と、位置不变性に基づきパラメータ削減したNN
 - 置み込み層、プーリング層、全結合層をもつ
 - 画像処理では様々なモデルが提案されている
 - 音データではスペクトログラム+2DCNNが最も広く用いられている

補足

- **Ground Truthの表現**
 - 単ラベル分類：one-hot表現（5クラスなら [0,0,1,0,0]）
 - 複数ラベル分類：multi-hot表現（5クラスなら [1,0,1,0,1]）
- **出力層の表現**
 - 単ラベル分類：Softmax - クラス間で合計すると1になるようにスケール
 - 複数ラベル分類：Sigmoid - 各クラスでON/OFFとなる確率値を独立に出力。値域は [0,1]
- 単ラベル：音楽分類、歌手識別等、複数ラベル：音楽タグづけ、混合音からの楽器識別等

- 学習済み有名モデル
 - <https://github.com/minzwon/sota-music-tagging-models>
 - <https://github.com/jordipons/musicnn>
 - <https://github.com/tensorflow/models/tree/master/research/audioset/vggish>
 - <https://github.com/marl/openI3>
- 音楽分類に関するデータセット
 - GTZAN: <http://marsyas.info/downloads/datasets.html>
 - Free Music Archive: <https://github.com/mdeff/fma>
 - MagnaTagATune: <https://mirg.city.ac.uk/codeapps/the-magnatagatunedataset>
 - Million Song Dataset: <http://millionsongdataset.com/>