7章: 畳み込みニューラルネットワーク

石田研 M1 宮尾

畳み込みニューラルネットワーク (CNN)

- 「畳み込み層」や「プーリング層」などの幾つかの特徴的な機能を持った層を積み上げることで構成されるニューラルネットワーク
- 特に画像認識の分野で優れた性能を発揮する

CNNの構造

- これまで見てきたニューラルネットワーク

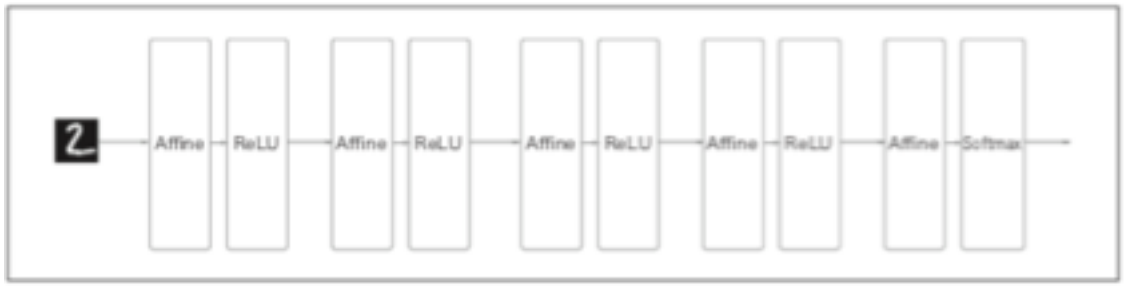


図7-1　全結合層（Affine レイヤ）によるネットワークの例

- 畳み込みニューラルネットワーク（CNN）

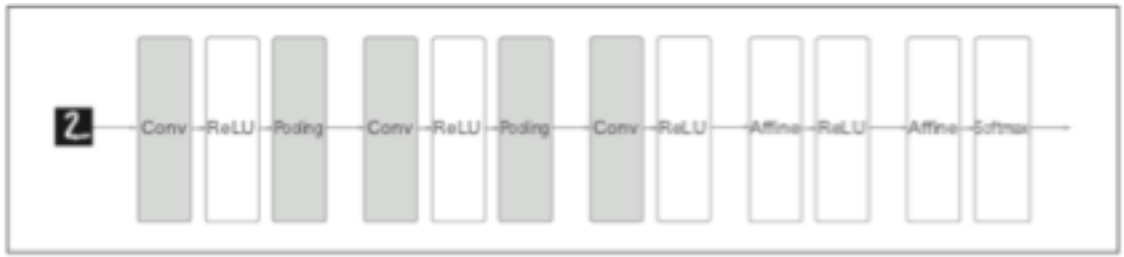


図7-2　CNN によるネットワークの例：Convolution レイヤと Pooling レイヤが新たに加わる（それぞれ背景が灰色の矩形で描画）

全結合層(Affineレイヤ)の問題点

- データの構造が無視されてしまう
  - 例えば画像データの場合、28ピクセル×28ピクセルの入力画像があったとき、全結合層ではそれを一列に並べて784個のデータとして扱う。
  - 画像データの元の形状(縦・横・チャンネル方向の３次元構造)には汲み取るべき本質的なパターンが潜んでいる可能性がある。

- 畳み込み層(Convolutionレイヤ)では入力データの形状を維持したまま次の層にデータを出力するため、画像などの形状を有したデータを正しく理解できる。

畳み込み演算

- 入力データに対してフィルターを適用

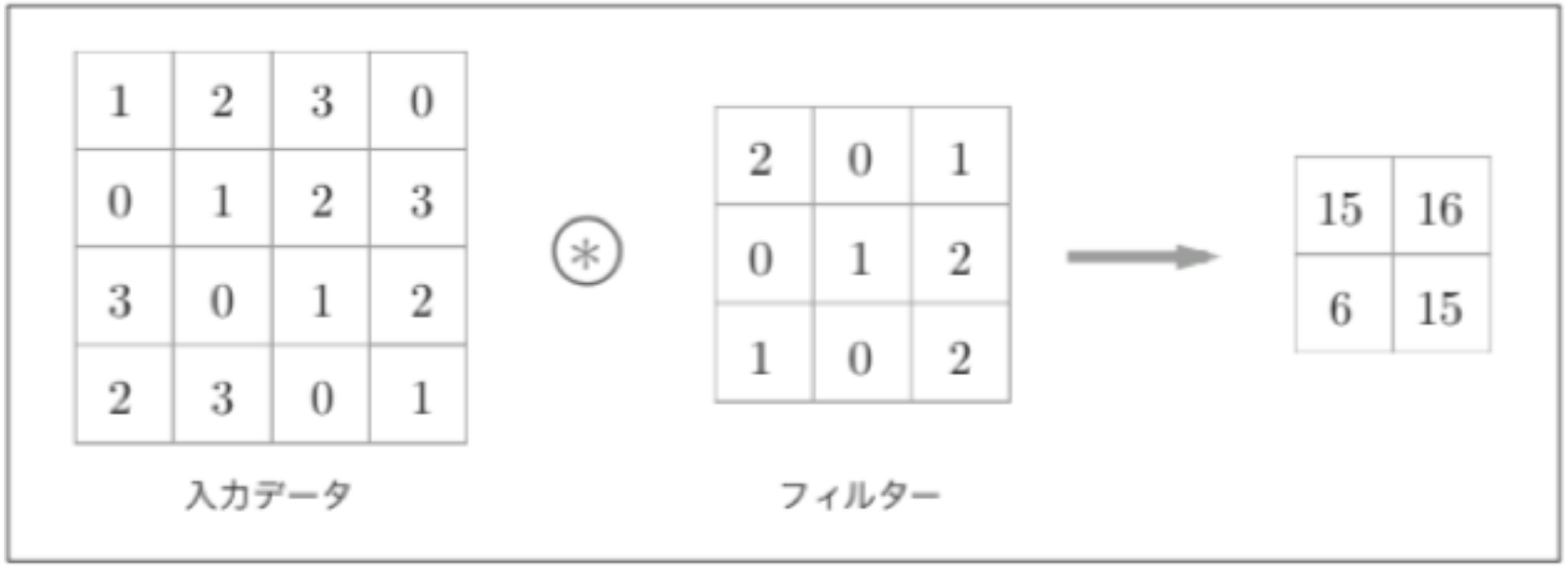


図7-3　畳み込み演算の例：畳み込み演算を「⊛」記号で表記

畳み込み演算の計算

- 入力データに対して、フィルターのウィンドウ（右図のグレー部）を一定の間隔でス ライドさせながら適用していく
- それぞれの場所で、フィルター の要素と入力の対応する要素を乗算しその和を求める
- 結果を出力の対応する場所へ格納

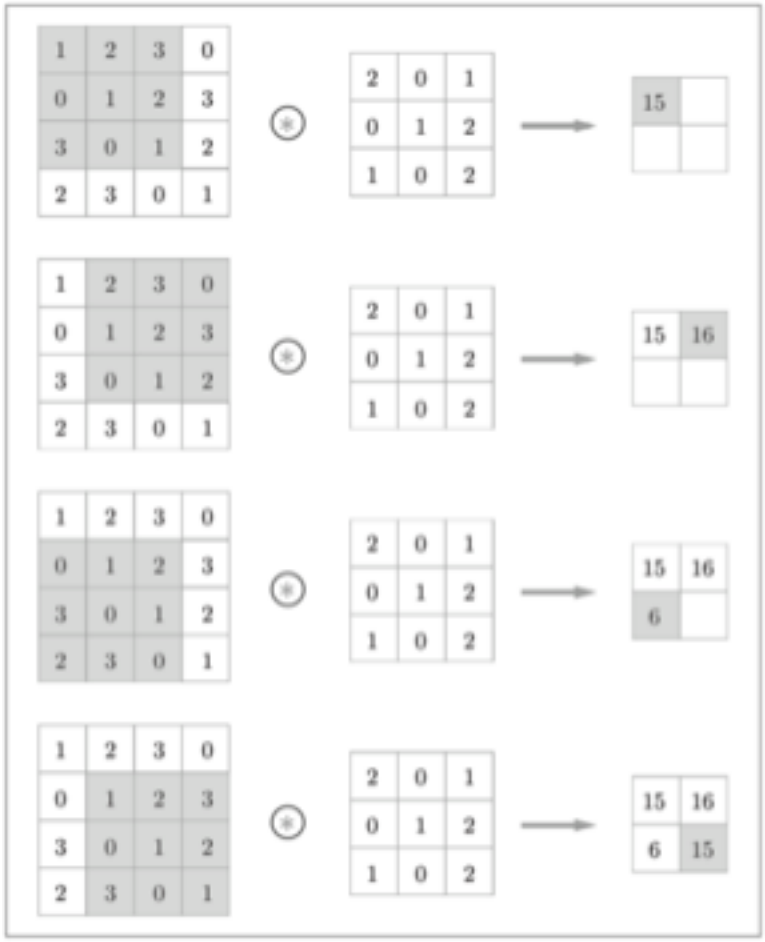


図7-4　畳み込み演算の計算手順

畳み込み演算のバイアス

- CNNではフィルターのパラメータが重みに対応
- バイアスはフィルター適用後のデータに加算
- バイアスの値は1つ（1×1）であり、同じ値がフィルター適用後のすべての要素に加算される。

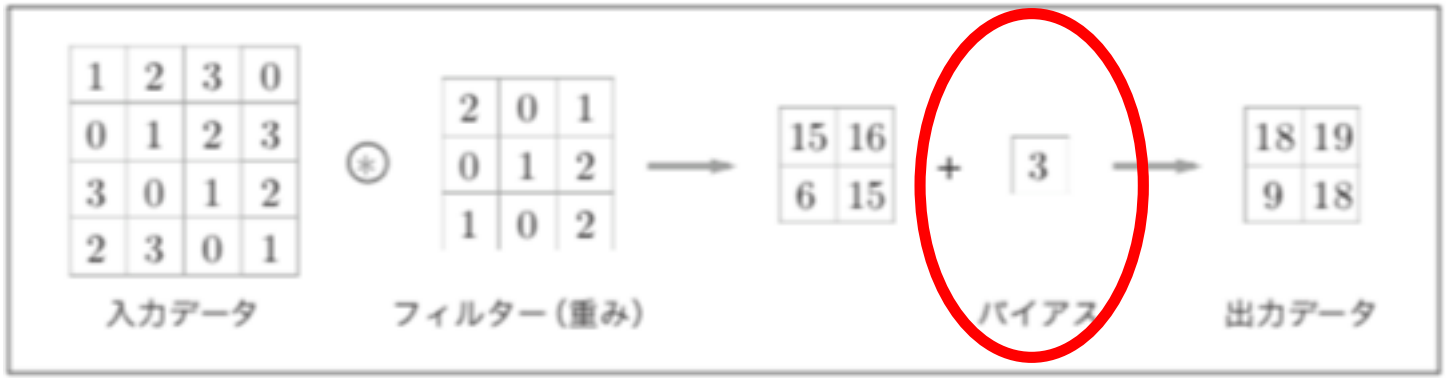


図7-5　畳み込み演算のバイアス：フィルターの適用後の要素に固定の値（バイアス）を加算する

パディング

- 出力サイズを調整するため、畳み込み層の処理を行う前に入力データの周囲に例えば0などの固定のデータを埋めることがある
- 下図の例では、本来入力データが(4, 4)のサイズに(3, 3)のフィルターを適用すると出力サイズは(2, 2)となるが、幅1のパディングを適用することで出力サイズを(4, 4)としてデータのサイズを保っている。

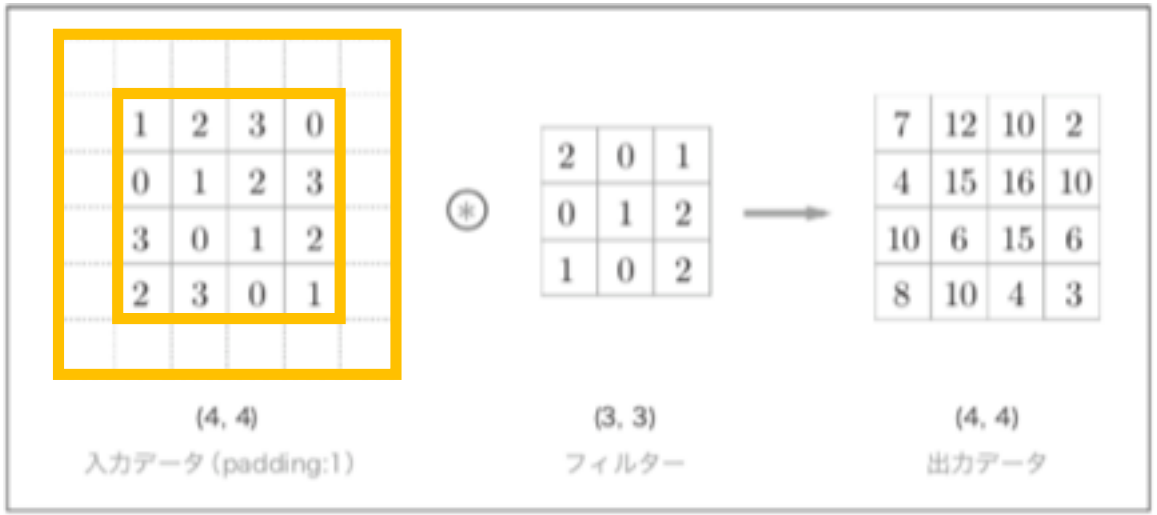


図7-6　畳み込み演算のパディング処理：入力データの周囲に 0 を埋める（図ではパディングを破線で表し、中身の「0」の記載は省略する）

ストライド

- フィルターを適用する位置の間隔をストライドと言う
- 下図の例では、入力サイズが (7, 7) のデータに対して、ストライドが 2 のフィルターを適用することで、出力サイズが(5, 5)から(3, 3)になっている

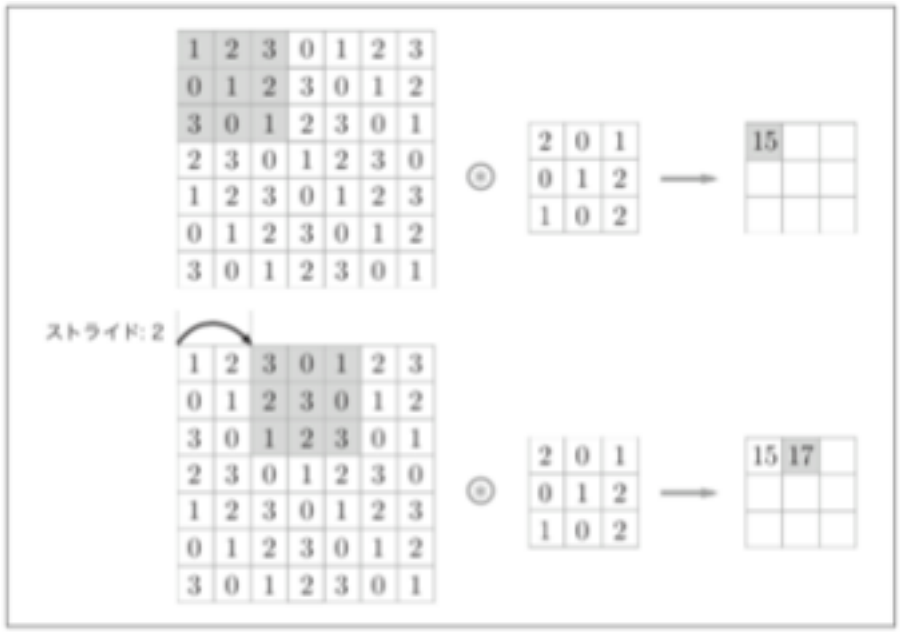


図7-7　ストライドが２の畳み込み演算の例

出力サイズの計算

- 出力サイズはストライドを大きくすると小さく、パディングを大きくすれると大きくなる
- この関係性を定式化すると以下のようになる

$$OH = \frac{H + 2P - FH}{S} + 1$$

$$OW = \frac{W + 2P - FW}{S} + 1$$

入力サイズ：(H, W)　フィルターサイズ：(FH, FW)

出力サイズ：(OH, OW)

パディング：P　ストライド：S

例：出力サイズの計算

入力サイズ:(28, 31)、パディング:2、ストライド:3、フィルターサイズ: (5, 5)の時、OH,OWを計算せよ

注意点

- 計算式の$\frac{H+2P-FH}{S}$と$\frac{W+2P-FW}{S}$が割り切れるようにそれぞれの値を設定しなければならない
- 出力サイズが割り切れない場合（結果が小数の場合）は、エラーを出力する、最も近い整数に丸めるなどの対応が必要

3 次元データの畳み込み演算

- 画像の場合、縦・横方向に加えてチャンネル方向も合わせた 3 次元のデータを扱う必要がある
- チャンネル方向に複数の特徴マップがある場合、チャンネルごとに入力データとフィルターの畳み込み演算を行い、それらの結果を加算してひとつの出力を得る

* 入力データとフィルターのチャンネル数は同じでなくてはならない

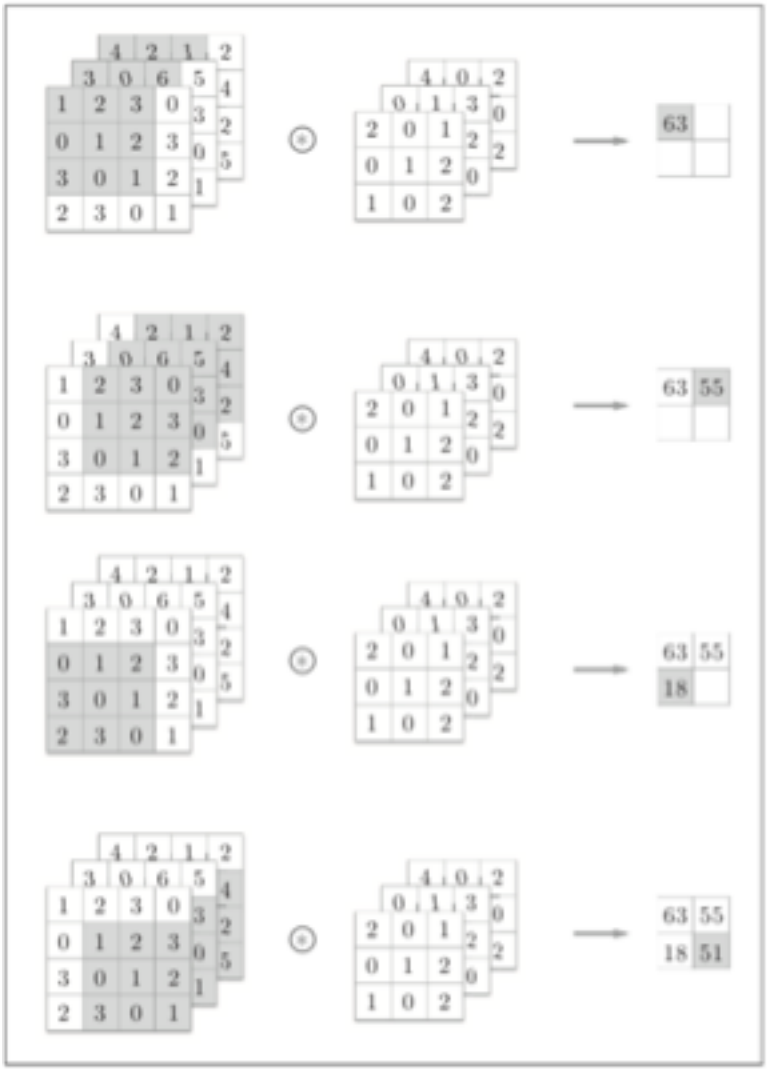


図7-9　3 次元データに対する畳み込み演算の計算手順

ブロックで考える

- 3 次元の畳み込み演算はデータやフィルターを直方体のブロックで考えると分かりやすい
- チャンネル数 C、高さ H、横幅 W のデータの形状は(C, H, W)と書く

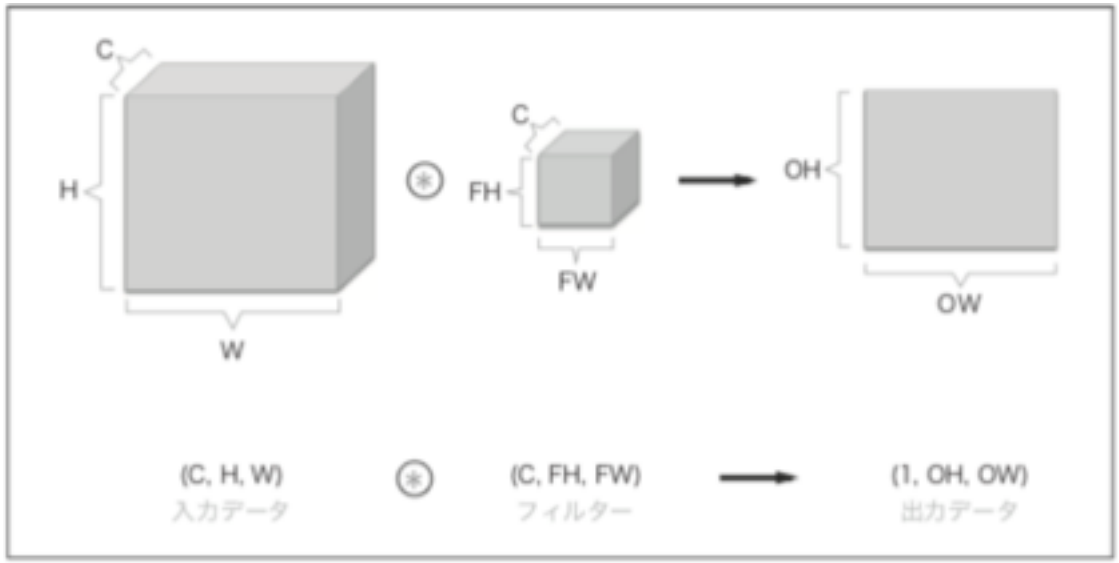


図7-10　畳み込み演算をブロックで考える。ブロックの形状に注意

複数フィルターによる畳み込み演算

- 出力をチャンネル方向にも複数持たせるために複数のフィルター(重み)を用いる
- FN 個のフィルターを適用することで出力のマップも FN 個生成される
- フィルターの重みデータは 4 次元のデータ として、(output_channel, input_channel, height, width) のように書く

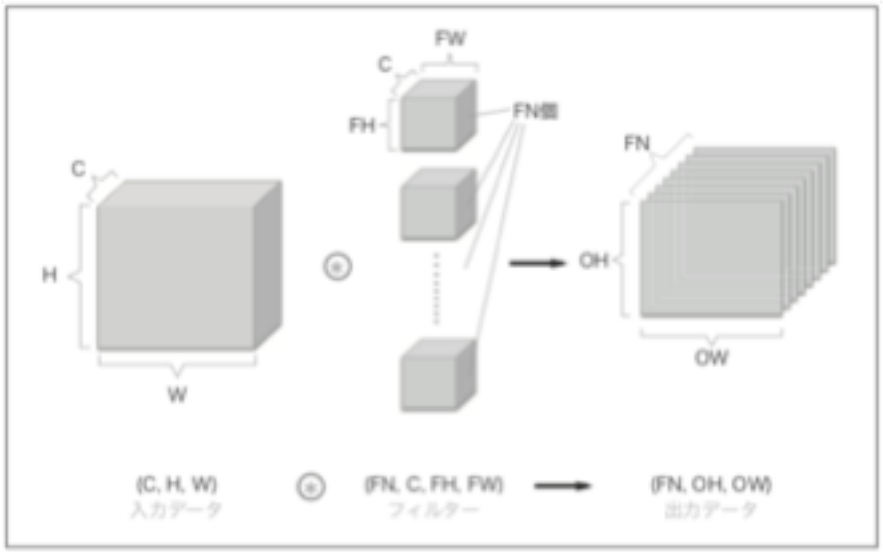
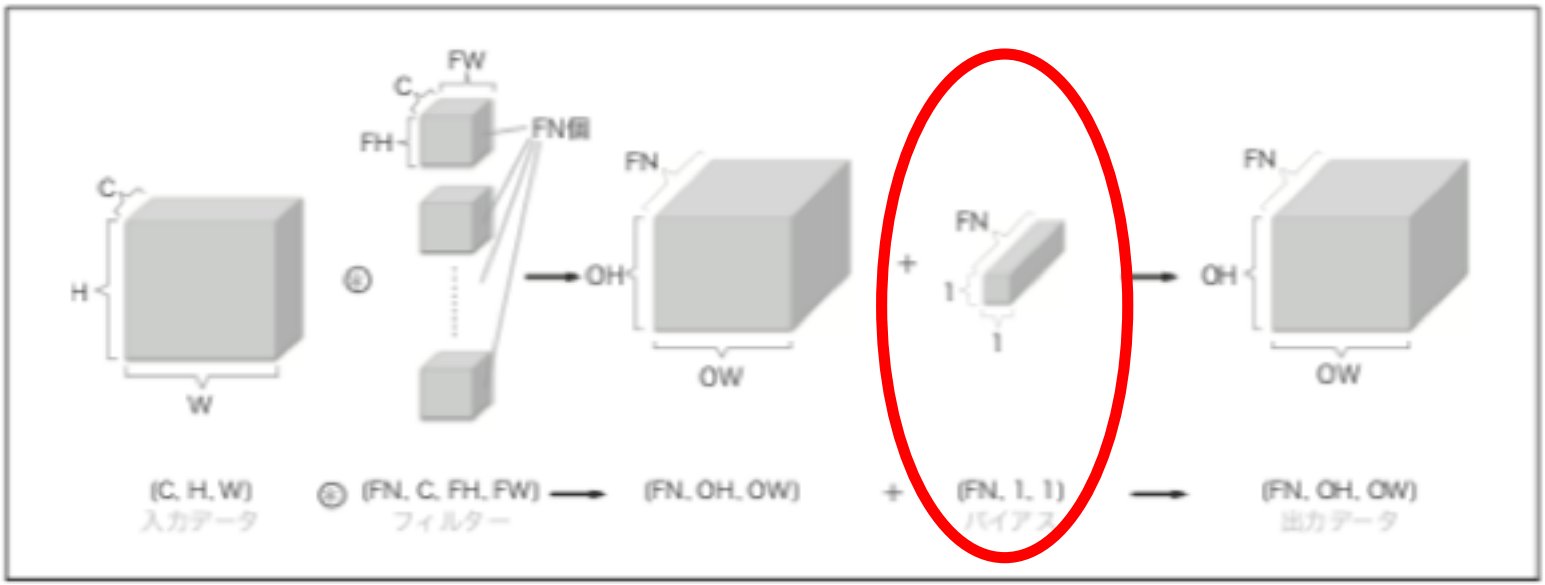


図7-11　複数のフィルターによる畳み込み演算の例

３次元データでのバイアス加算

- バイアスは、1 チャンネルごとにひとつだけデータを持つ
- フィルターの出力 結果の (FN, OH, OW) に対して、チャンネルごとに、同じバイアスの値が加算される



バッチ処理

- 全結合のニューラルネットワークと同様に入力データを一束にまとめたバッチ処理に対応したい
- 各層を流れるデータを4次元のデータ(batch_num, channel, height, width)として格納

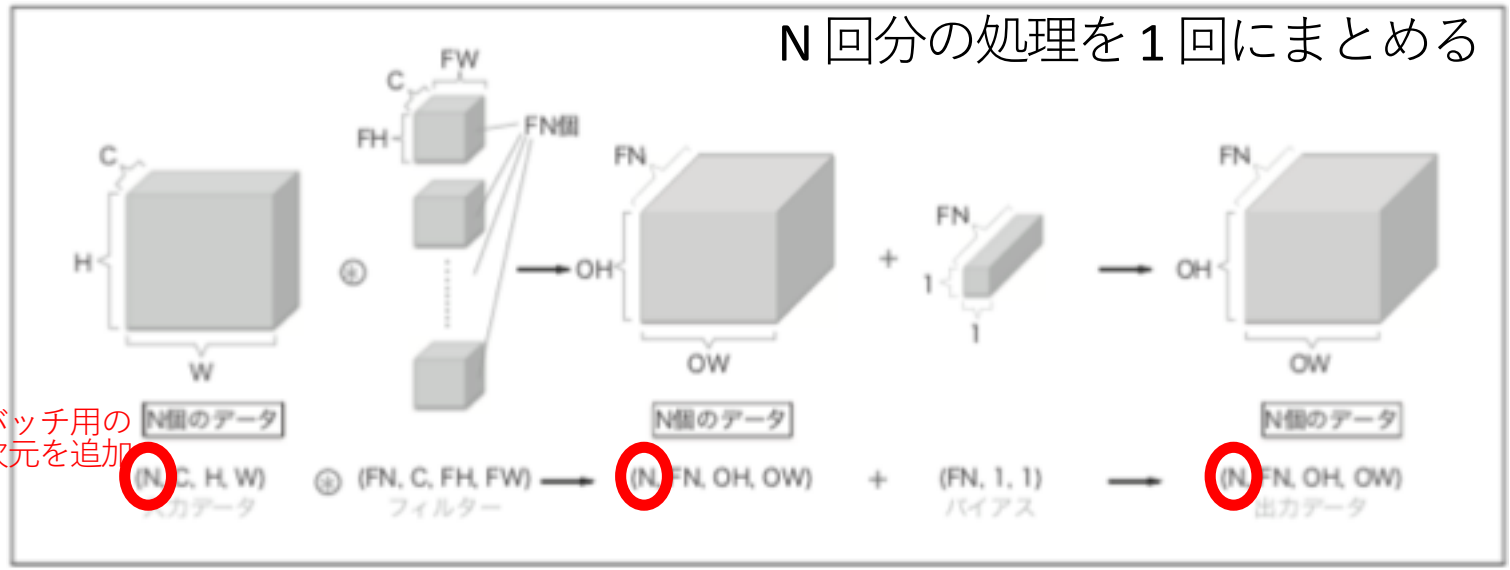


図7-13　畳み込み演算の処理フロー（バッチ処理）

プーリング層（Poolingレイヤ）

- 空間内のある領域を一つに集約するような処理を行うことで縦・横方向の空間を小さくする演算
- 下図は、ある一定の領域で最大値を取る**Maxプーリング**を領域2×2、ストライド2で行った場合の処理手順。
- 一般的にプーリングのウィンドウサイズと、ストライドは同じ値に設定する

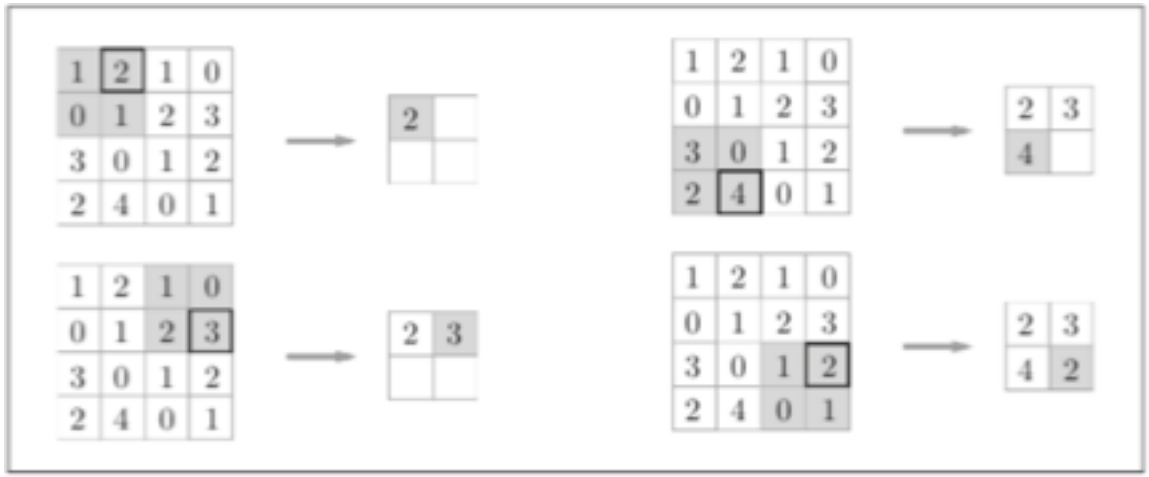


図7-14　Max プーリングの処理手順

プーリング層の特徴

- 学習するパラメータがない
  - 対象領域から最大値を取る（もしくは平均を取る）だけの処理なので、学習すべきパラメータは存在しない
- チャンネル数は変化しない
  - チャンネルごとに独立して計算を行う
- 微小な位置変化に対してロバスト(頑健)
  - 入力データの小さなズレはプーリング が吸収するためプーリングは同じような結果を返す

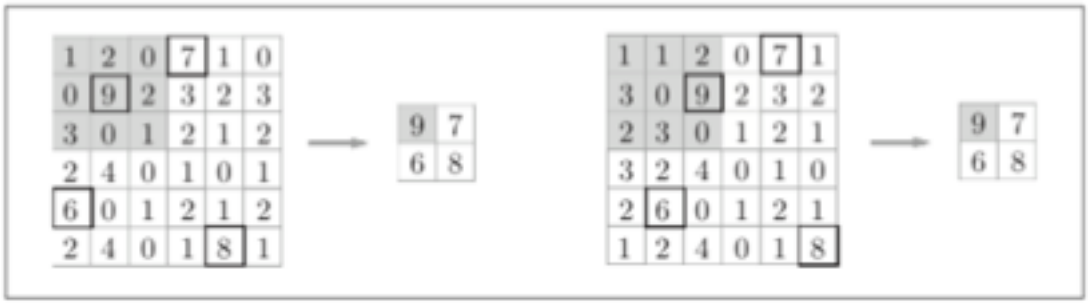


図7-16　入力データが横方向に 1 要素分だけずれた場合でも、出力は同じような結果になる（データによっては同じにならない場合もある）

プーリング層の役割

- 抽出された特徴が平行移動・回転などでも影響を受けにくいようロバスト性を与える
- 画像が猫なのか犬なのか識別できることが重要であり、写っている位置などの重要でない情報は削ぎ落とす

同じカテゴリとして識別したい2つの画像

まとめ

- CNN を構成する基本モジュールである「畳み込み層」と「プーリング層」 について学んだ
- 畳み込み層はパディング・ストライドなどを用いて入力データの形状を維持したまま次の層にデータを出力することができる
- プーリング層では空間内のある領域を一つに集約するような処理を行うことで入力データにロバスト性を与える

次回

- Convolution / Pooling レイヤの実装
- CNN の実装