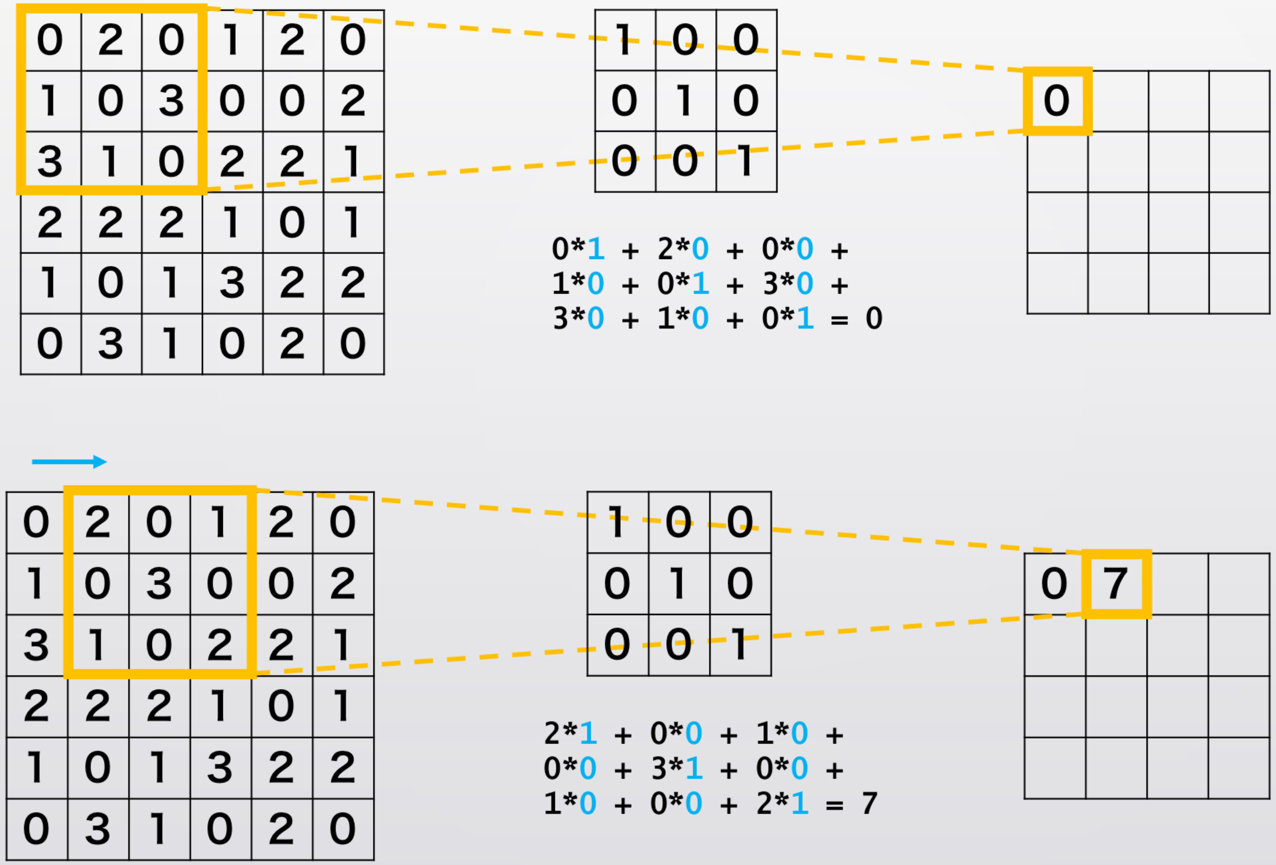
データマイニング
Data Mining

14: ニューラルネットワーク② Neural Network

土居 裕和 Hirokazu Doi
長岡技術科学大学 Nagaoka University of Technology

2次元の畳み込み 2D-Convolution



https://axa.biopapyrus.jp/deep-learning/cnn/convolution.html

画像フィルター Image Filter

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

6 x 6

*

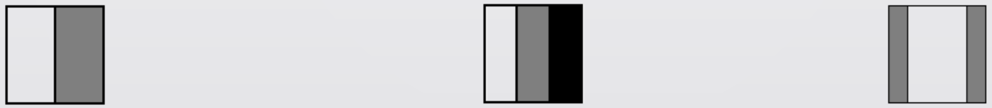
1	0	-1
1	0	-1
1	0	-1

3 x 3

=

-0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

4 x 4



https://datahacker.rs/edge-detection/

中央のフィルターで畳み込むと、縦方向のエッジが強調される

When an image is convoluted with a filter in the center, vertical edges in the image is extracted

画像フィルター Image Filter

1	0	-1
1	0	-1
1	0	-1

Vertical edges

-1	-1	-1
0	0	0
1	1	1

Horizontal edges

https://datahacker.rs
/edge-detection/

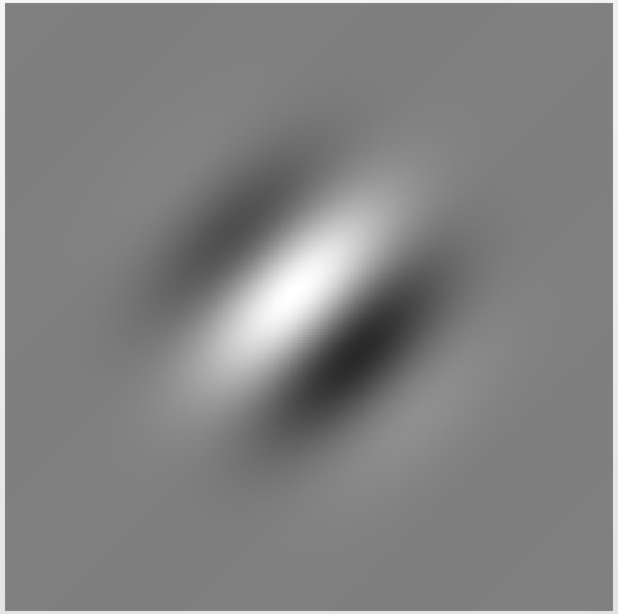
ガボールフィルター Gabor Filter

$$g(x,y) = Kexp\left(-\frac{x'^2 + y'^2}{2}\right) exp\{i(2\pi f x' + P)\}$$

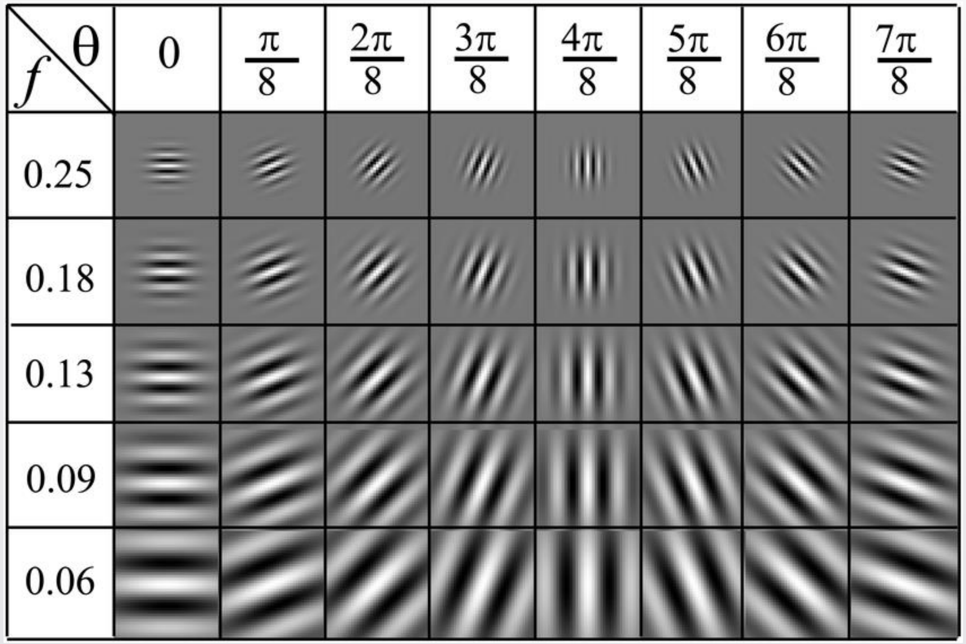
$$(x', y') = (x, y)\begin{pmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{pmatrix}\begin{pmatrix} \frac{1}{\sigma_x} & 0 \\ 0 & \frac{1}{\sigma_y} \end{pmatrix}$$

座標回転した後、正規化

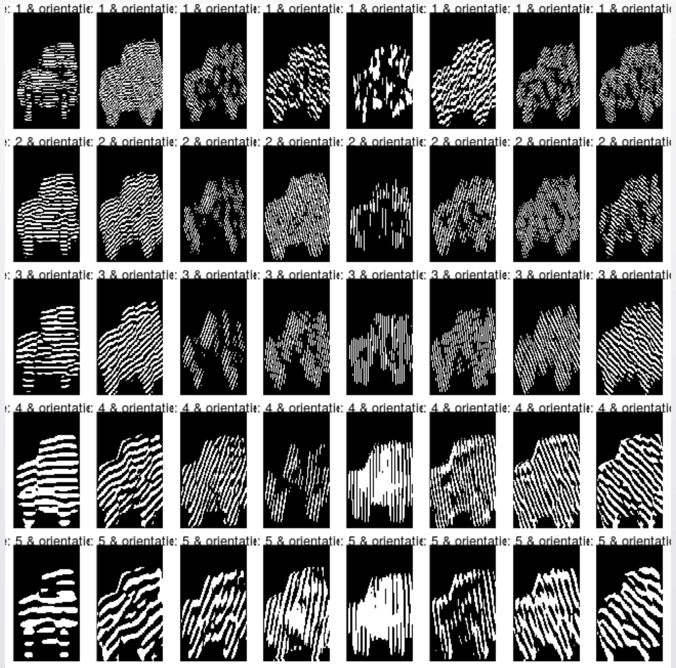
Coordinate rotation and then scaling



ガボールフィルター Gabor Filter



Soans et al, 2016



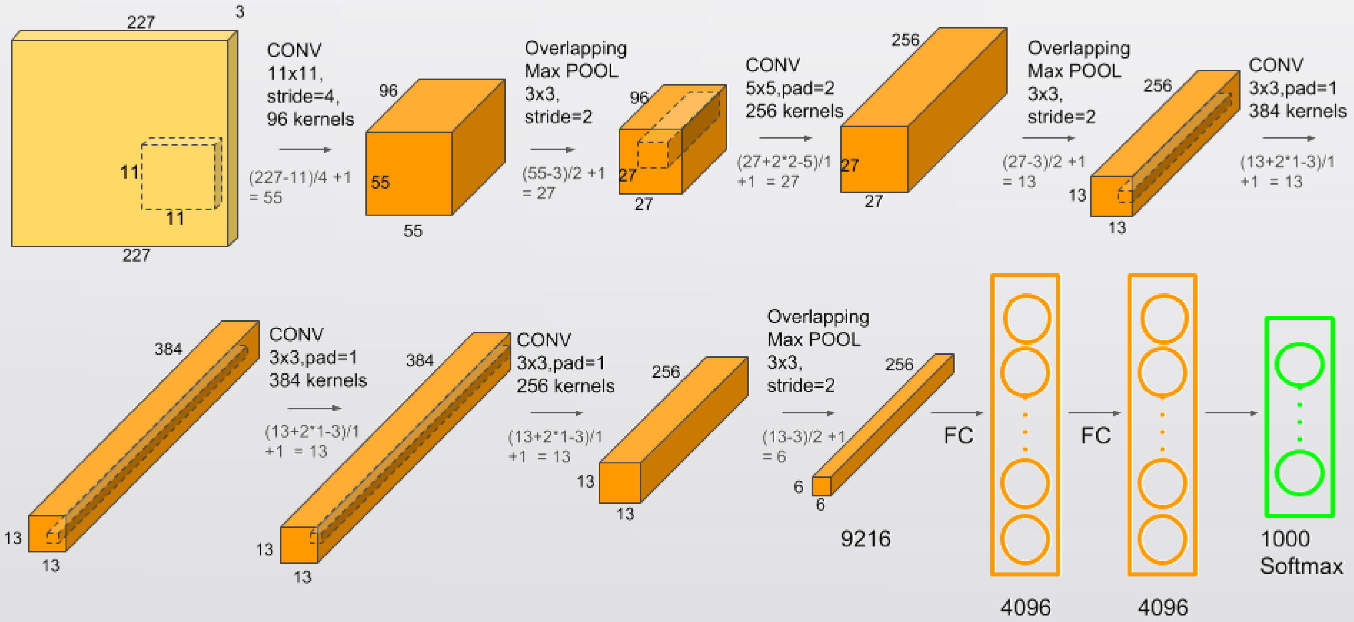
https://cran.r-project.org/web/packages/OpenImageR/vignettes/Gabor_Feature_Extraction.html

異なる向きと空間周波数をもった特徴の抽出に適している

Suitable for extraction of features with different orientations and spatial frequencies

畳み込みニューラルネットワーク CNN
Convolutional Neural Network

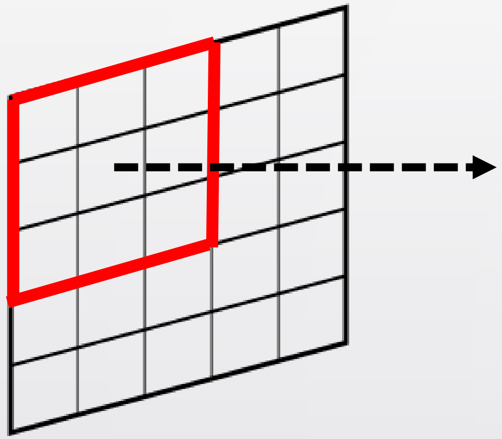
Alex Net



Layer		Feature Map	Size	Kernel Size	Stride	Activation
Input	Image	1	227x227x3	-	-	-
1	Convolution	96	55 x 55 x 96	11x11	4	relu
	Max Pooling	96	27 x 27 x 96	3x3	2	relu
2	Convolution	256	27 x 27 x 256	5x5	1	relu
	Max Pooling	256	13 x 13 x 256	3x3	2	relu
3	Convolution	384	13 x 13 x 384	3x3	1	relu
4	Convolution	384	13 x 13 x 384	3x3	1	relu
5	Convolution	256	13 x 13 x 256	3x3	1	relu
	Max Pooling	256	6 x 6 x 256	3x3	2	relu
6	FC	-	9216	-	-	relu
7	FC	-	4096	-	-	relu
8	FC	-	4096	-	-	relu
Output	FC	-	1000	-	-	Softmax

https://medium.com/@siddheshb008/alexnet-architecture-explained-b6240c528bd5

畳み込み層 Convolution Layer



i番目の赤枠内の画素値のベクトル $\boldsymbol{x_i}$

Vector of pixels within i-th red square

									1

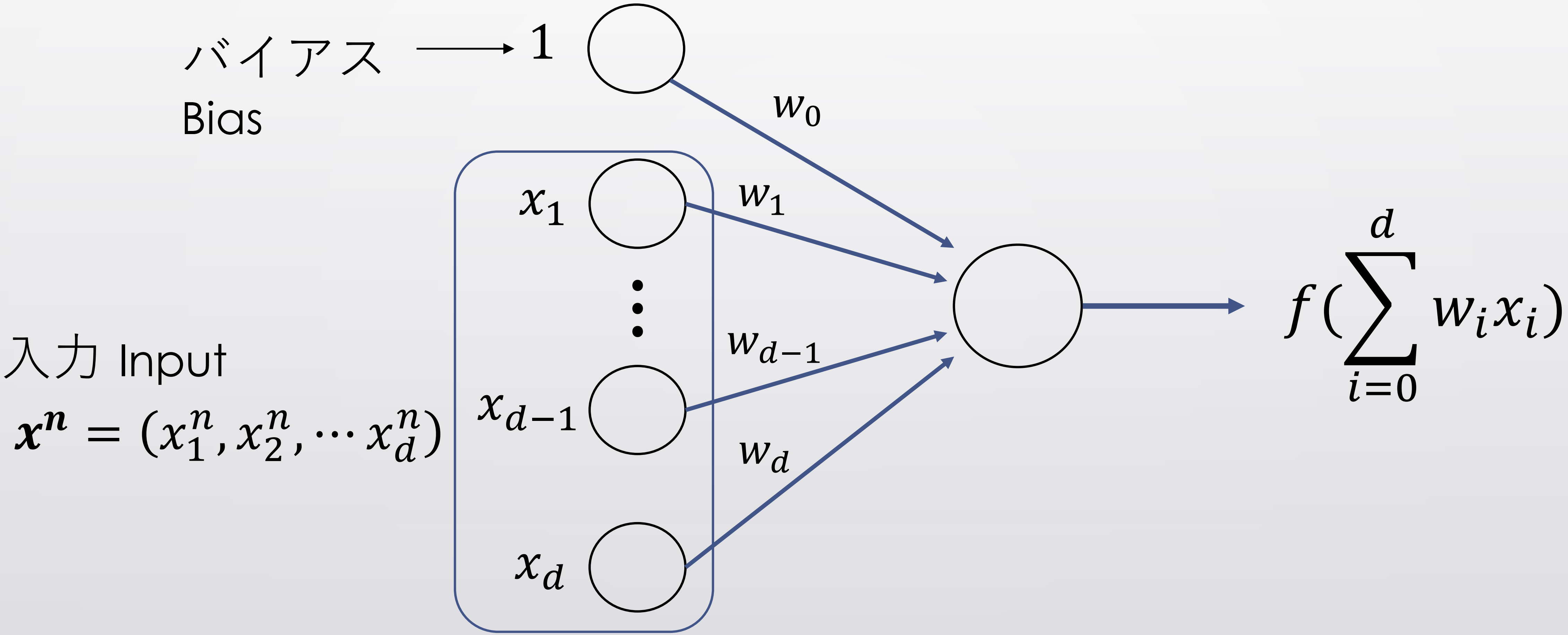
j番目のフィルターの重みベクトル$\boldsymbol{w_j}$

Weight vector of j-th filter

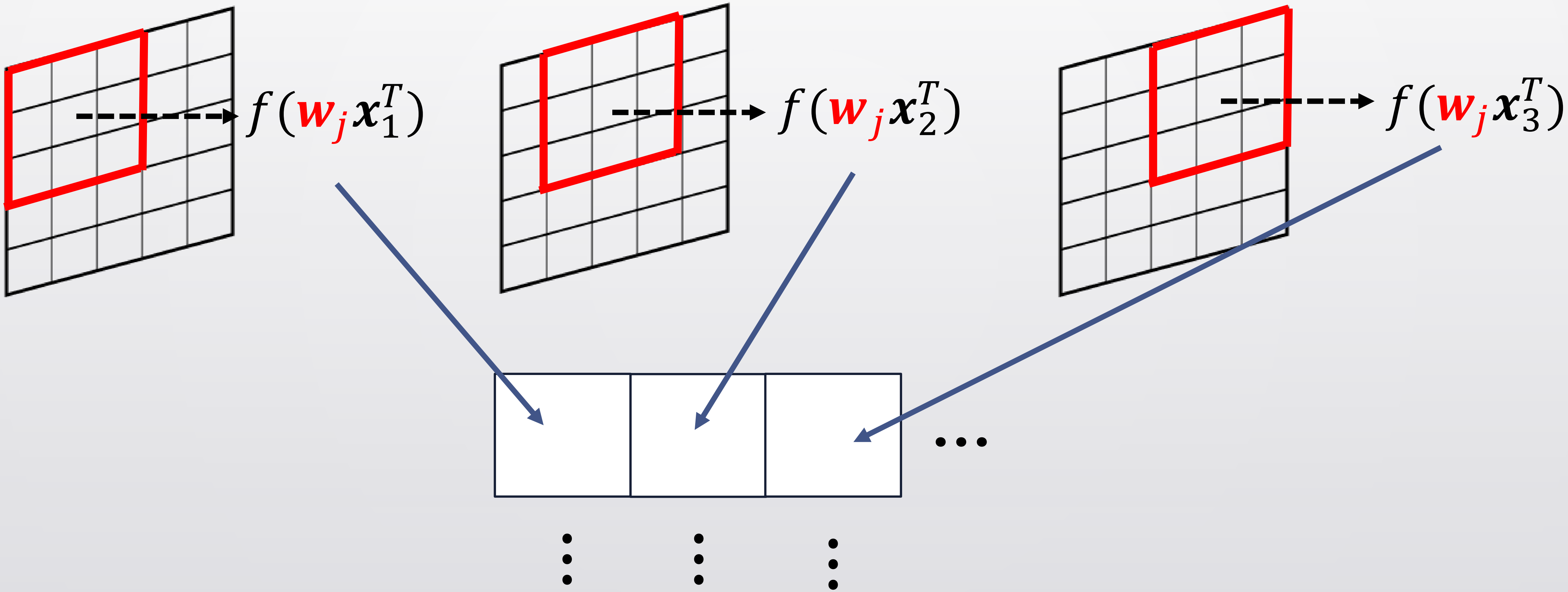
									b_j

$$Activation = f\left(\boldsymbol{w}_j \boldsymbol{x}_i^T\right)$$

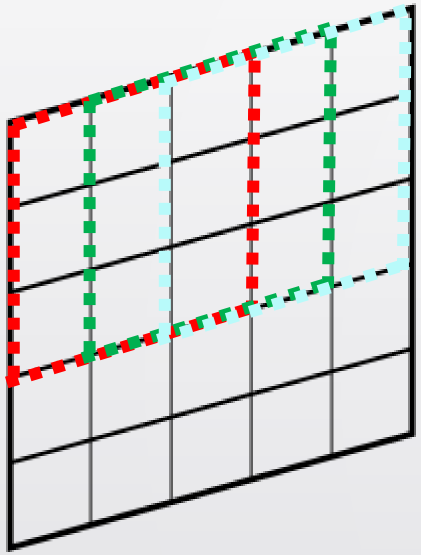
パーセプトロン Perceptron



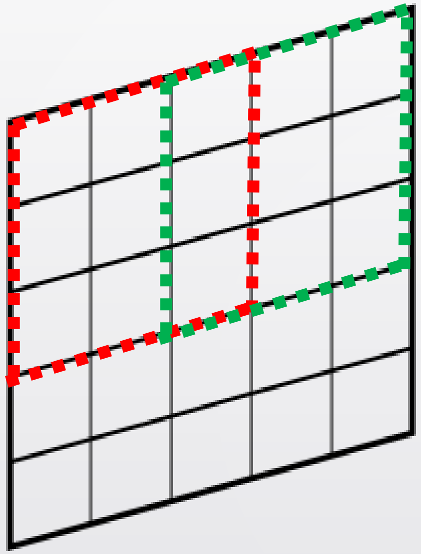
畳み込み層 Convolution Layer



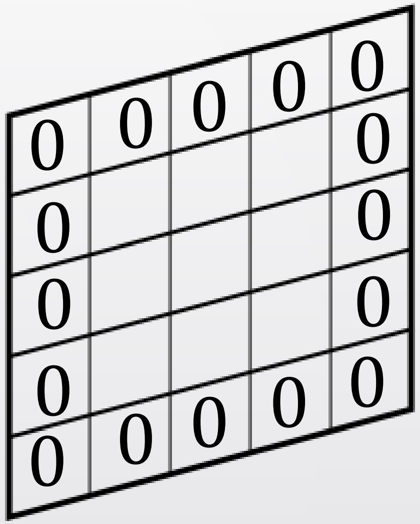
パディングとストライド Padding and Stride



$Stride = 1$



$Stride = 2$



ストライドが大きいと畳み込みの結果出力される画像が小さくなる

Larger stride makes size of output image smaller

出力画像の周りを数値で埋めることで、画像サイズを復元する

Restore image size by adding numerical values around output image

パディングとストライド Padding and Stride

H, W: 入力画像のサイズ
Size of input image

O_h, O_w: 出力画像のサイズ
Size of output image

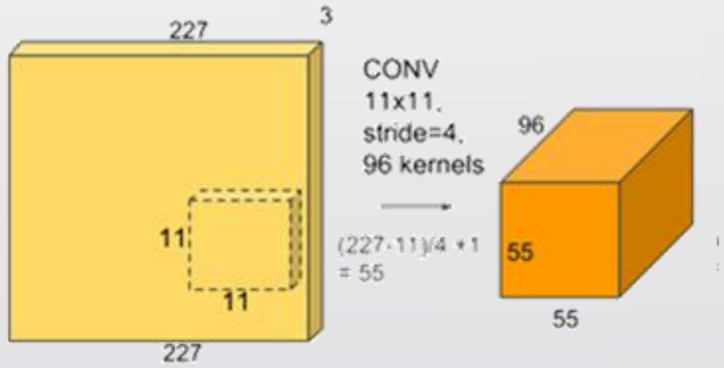
F_h, F_w: フィルターのサイズ
Size of output image

P: パディングの幅 Width of Padding

S: ストライド Stride

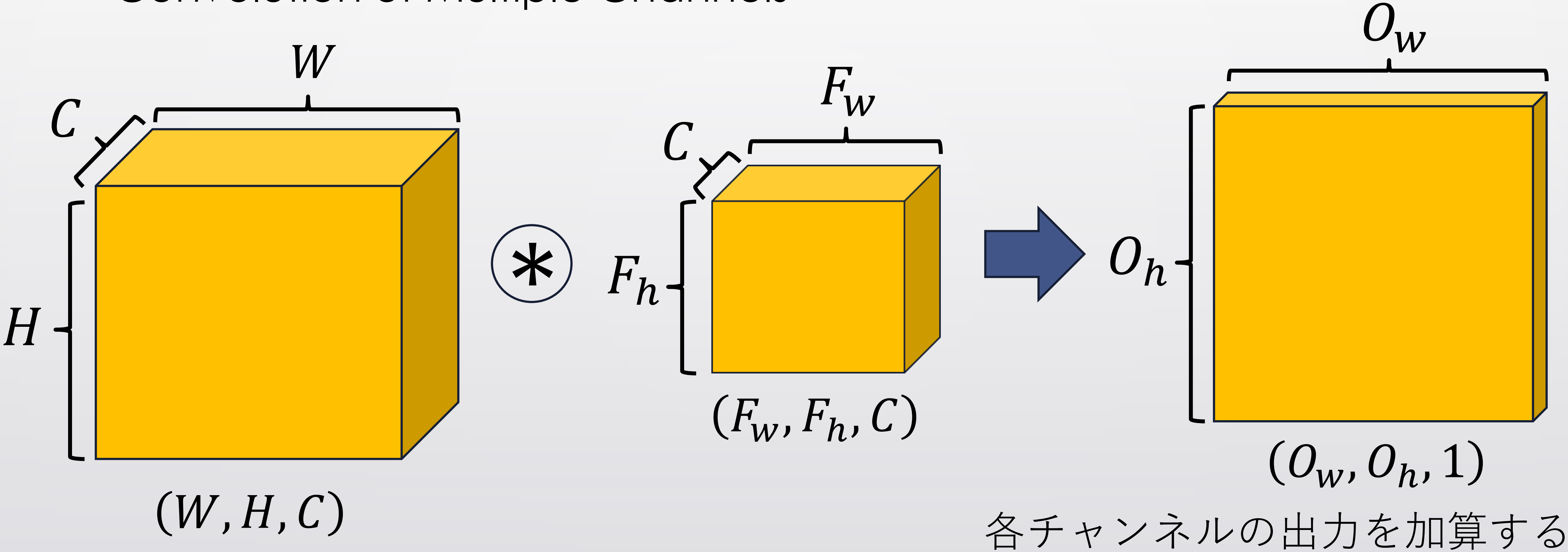
$$O_w = \frac{W+2P-F_w}{S} + 1$$

$$O_h = \frac{H+2P-F_h}{S} + 1$$



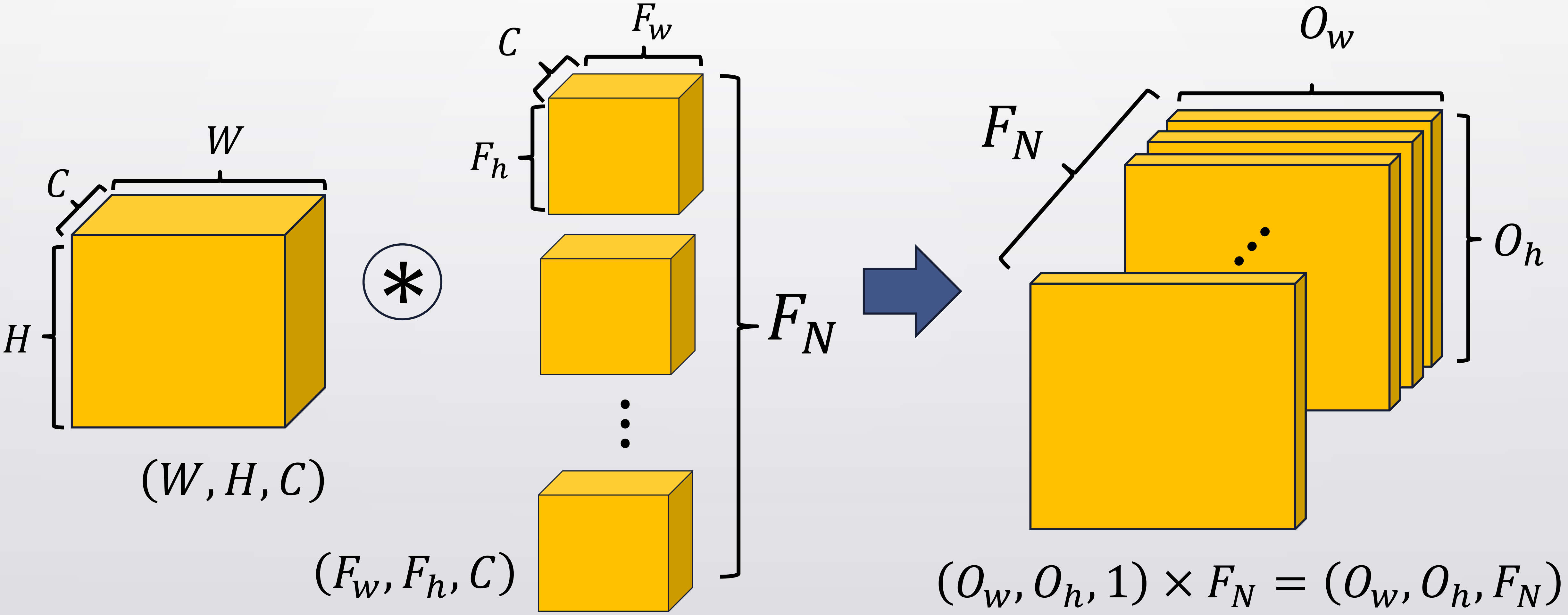
AlexNetでは96枚のフィルターを使用
96 filters are used in the first convolution layer in Alex Net

複数チャンネルの畳み込み
Convolution of Multiple Channels

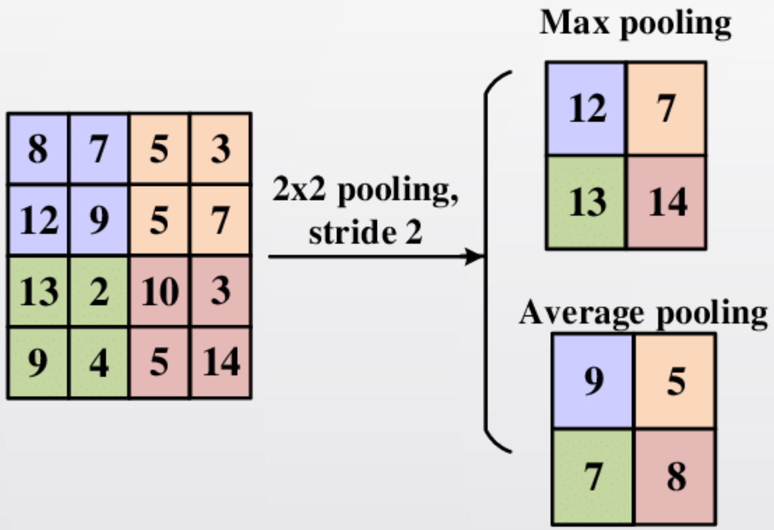


各チャンネルの出力を加算する
Output is summed across channels

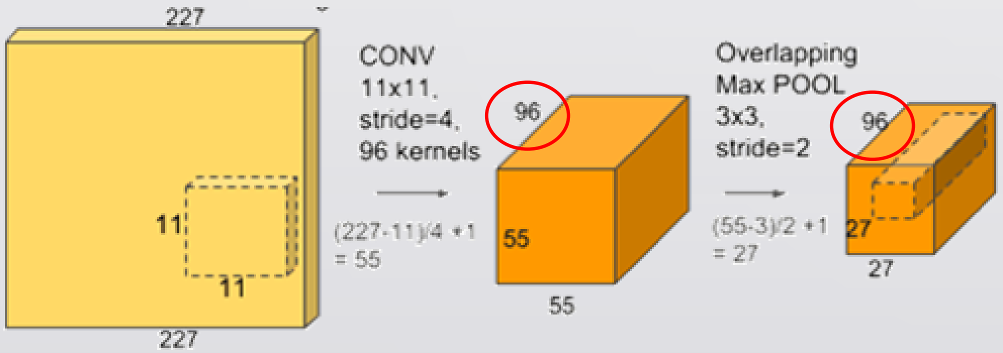
複数チャンネルの畳み込み Convolution of Multiple Channels



プーリング層 Pooling Layer



Yingge et al, 2020



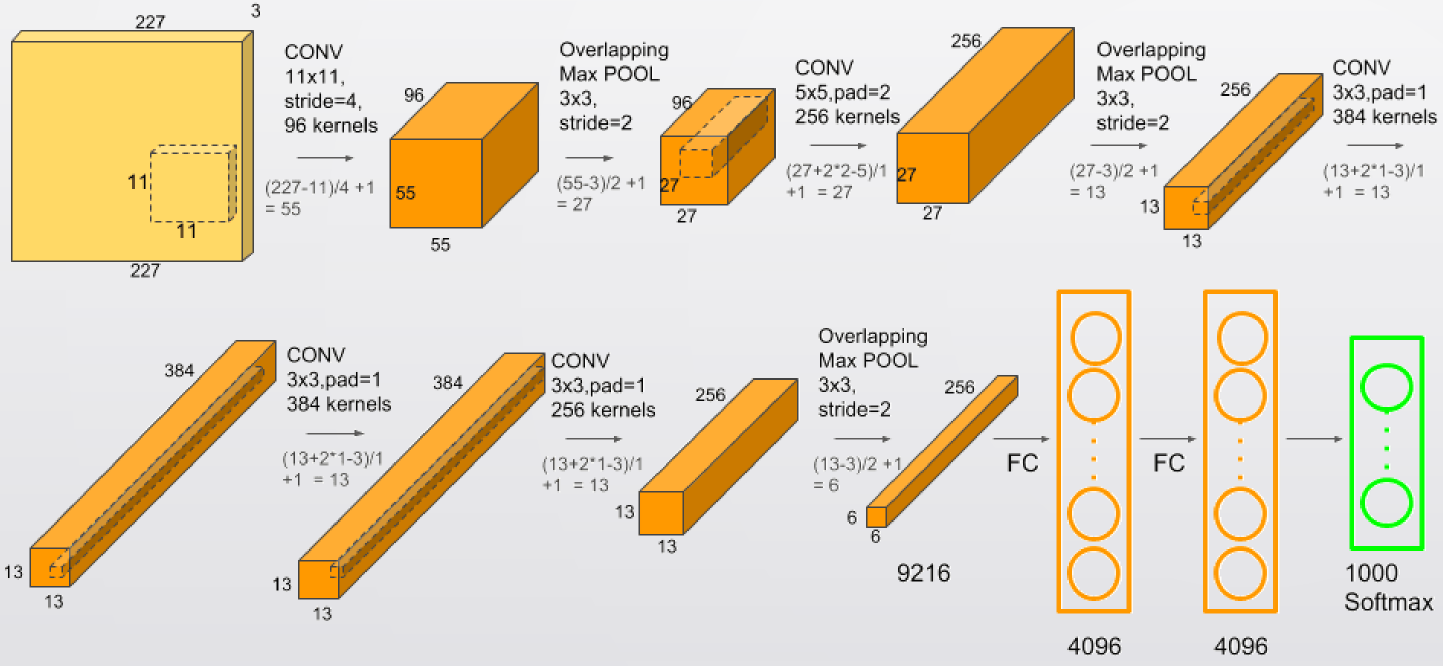
プーリングで位置への依存性を低減させる

Dependency on location within an image is mitigated by pooling

プーリングでも出力画像サイズは小さくなるので、パディングを行うことがある

Padding is often used to avoid shrinkage of image by pooling

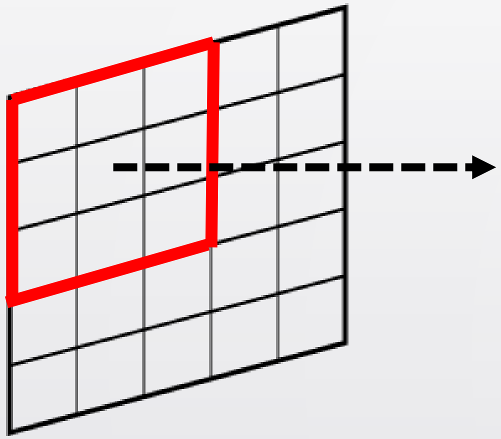
Alex Net



畳み込み層とプーリング層の後に、全結合層を配置する

Fully connected layers are located after repetition of convolution and pooling layers

畳み込み層 Convolution Layer



i番目の赤枠内の画素値のベクトル $\boldsymbol{x_i}$

Vector of pixels within i-th red square

									1

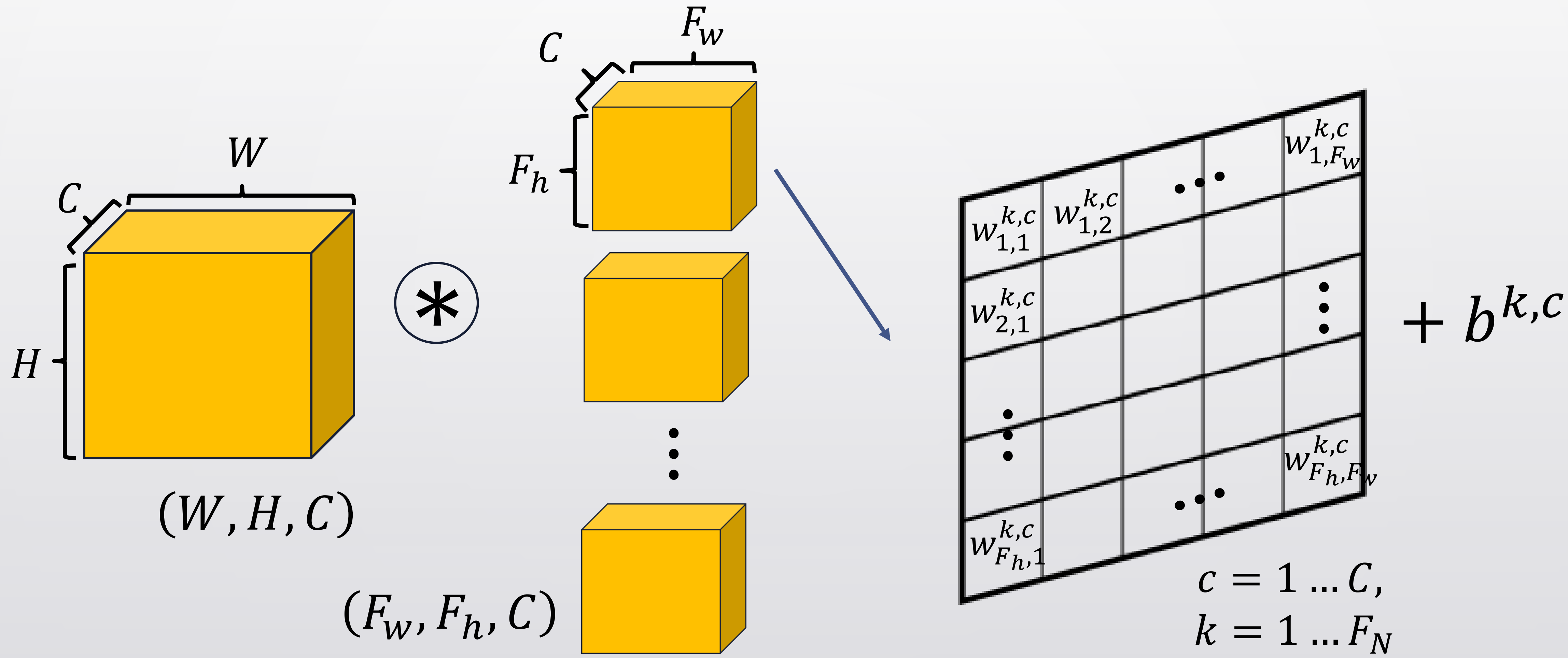
j番目のフィルターの重みベクトル$\boldsymbol{w_j}$

Weight vector of j-th filter

									b_j

$$Activation \ = f\left(\boldsymbol{w}_j \boldsymbol{x}_i^T\right)$$

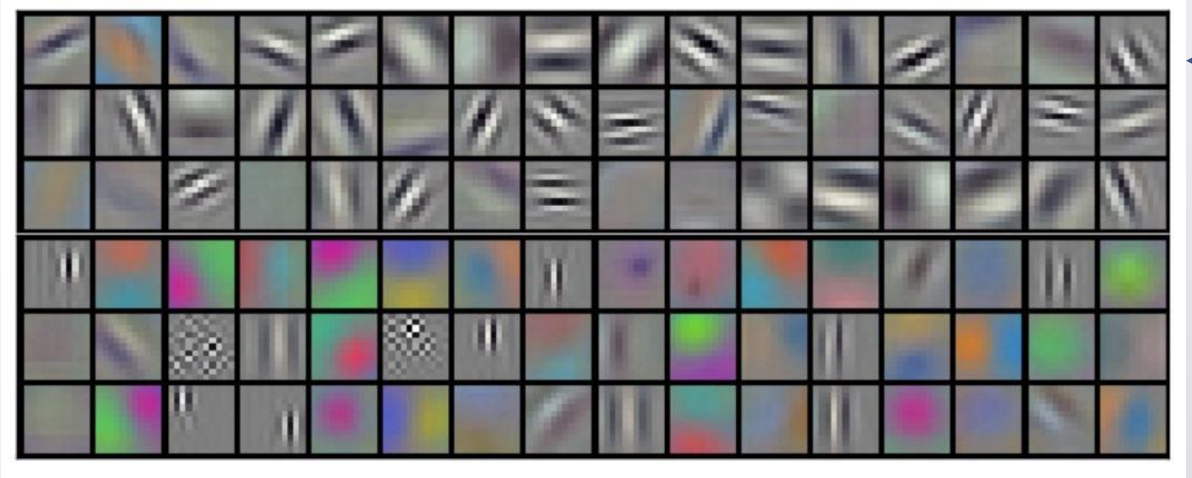
フィルターの学習 Acquisition of Filters

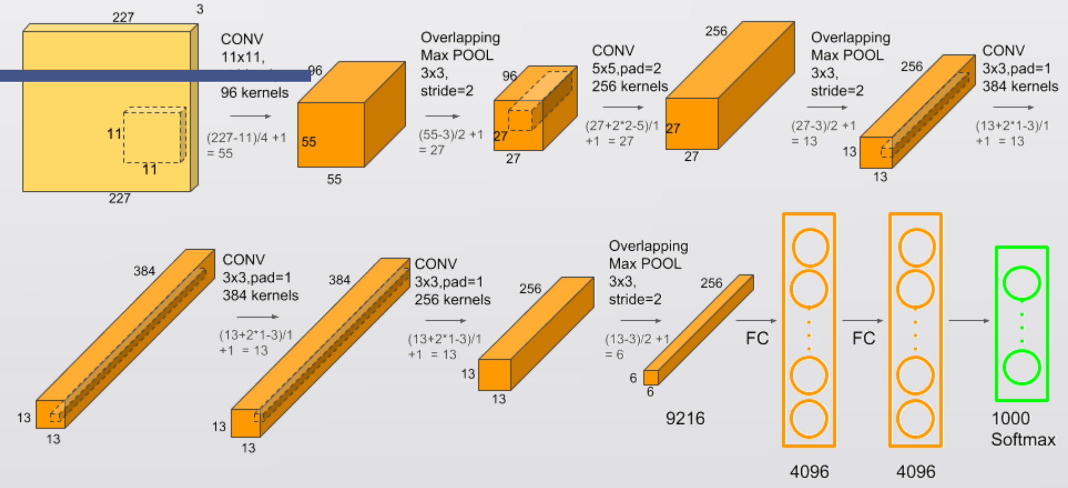


フィルターの学習 Acquisition of Filters

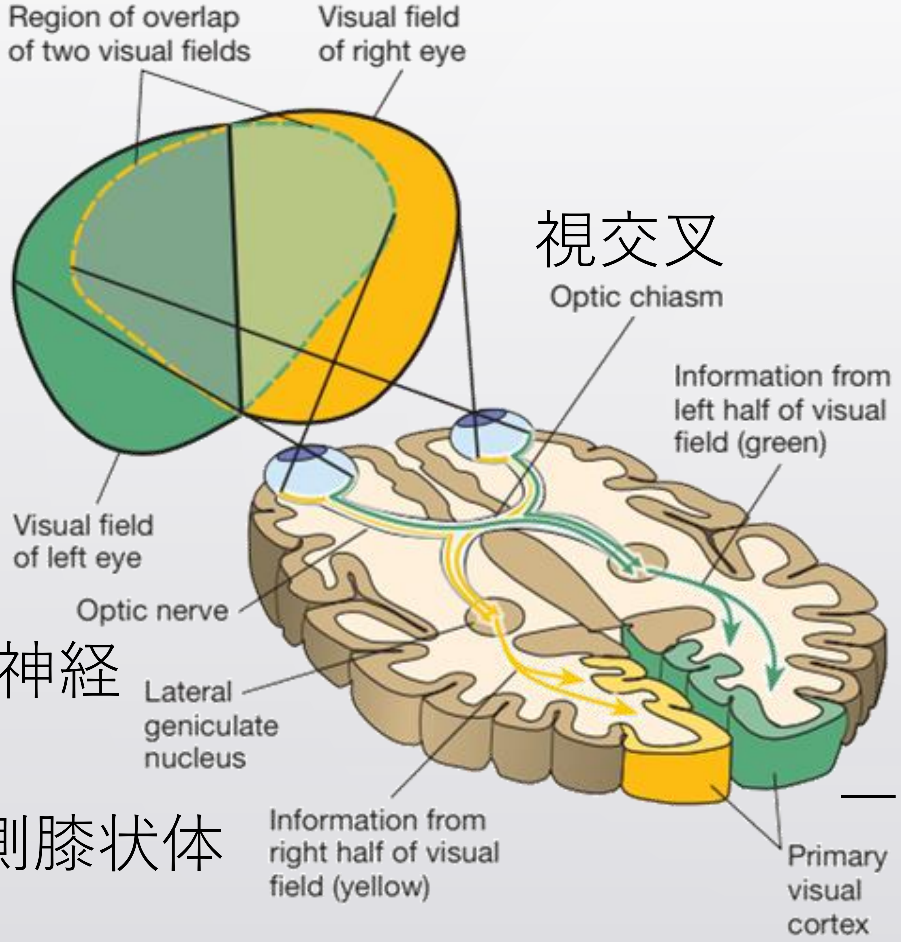
誤差逆伝搬法による学習の結果、Alex Netの第一の畳み込み層で
ガボールフィルターが獲得された

Weight updating by backpropagation led to acquisition of Gabor filter at the first convolution layer of Alex Net





視覚伝導路 Visual Pathway



視交叉

視神経

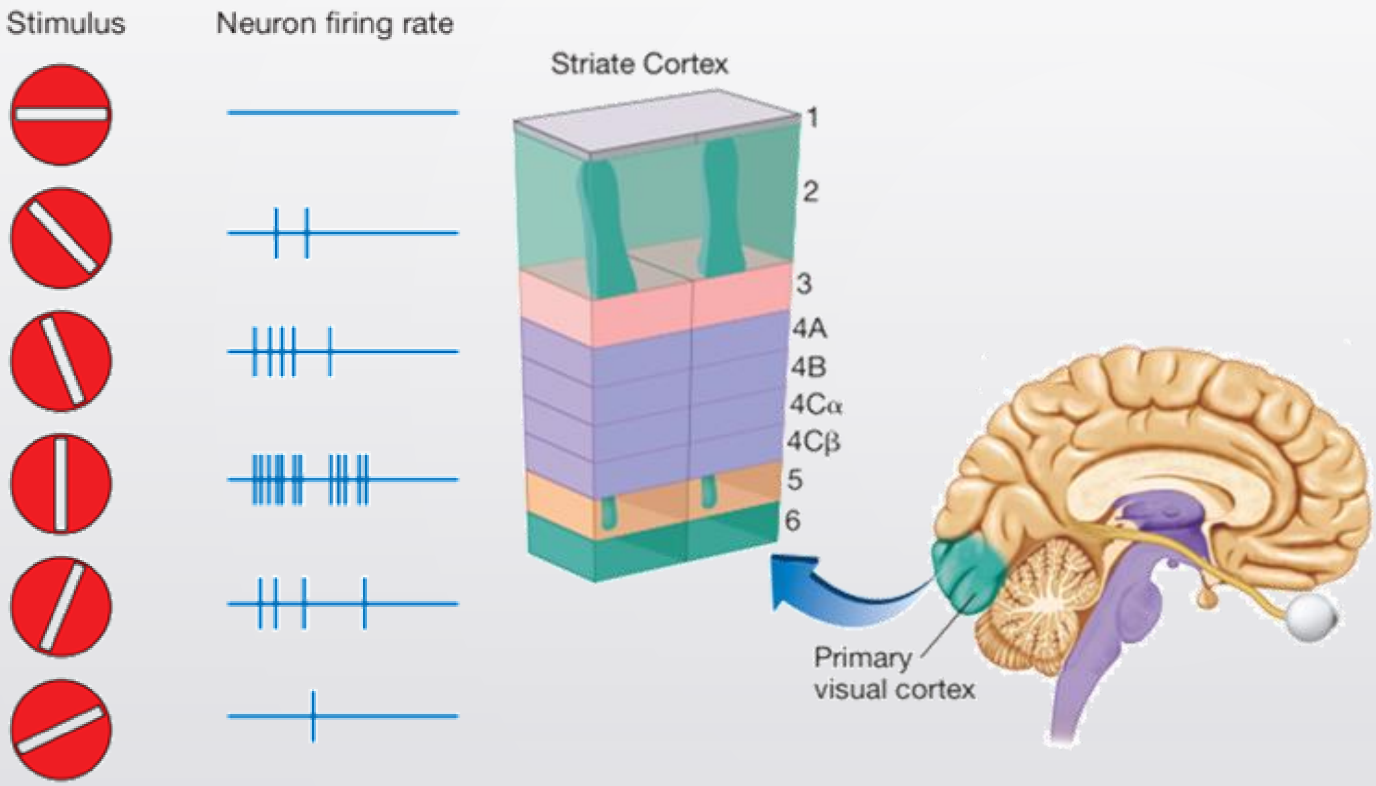
外側膝状体

一次視覚野

外側膝状体では、視覚像は画素の集まりとして表象される

Visual image is represented as a collection of pixels at lateral geniculate nucleus

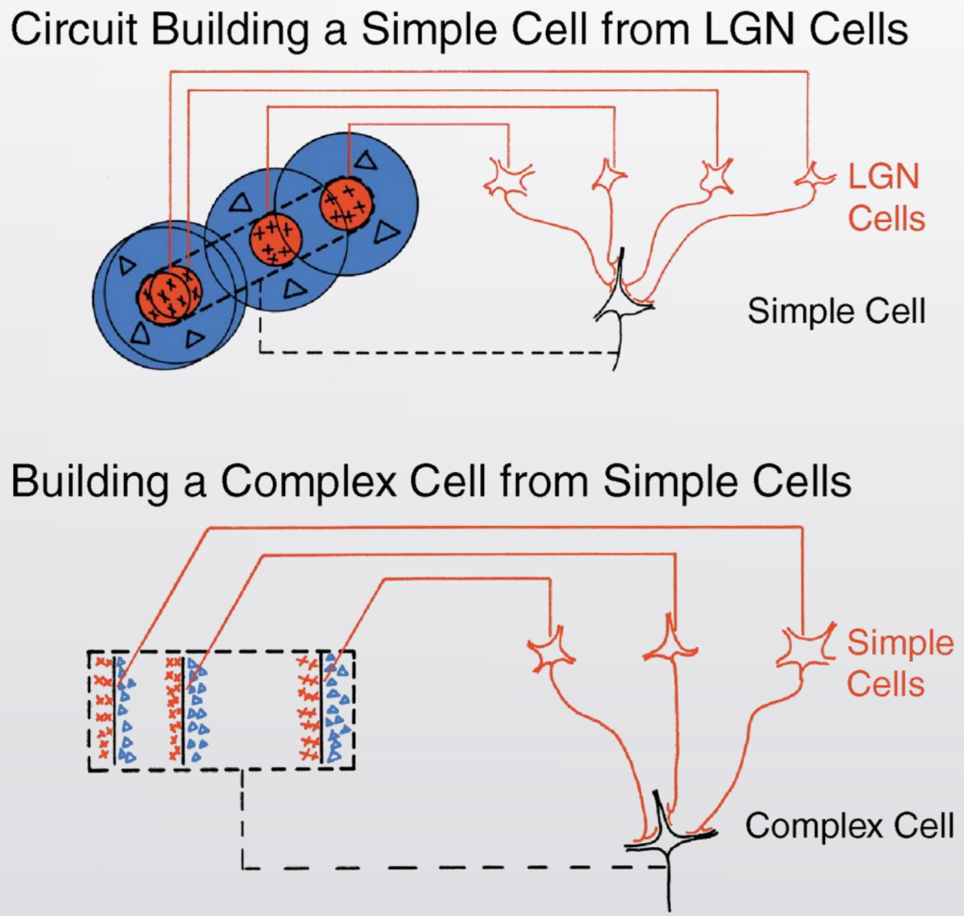
単純・複雑細胞 Simple and Complex Cells at V1



一次視覚野には、受容野内の色・傾きなどの単純な知覚特徴量に反応する細胞がある

Primary visual cortex contains neurons responsive to low-order perceptual features such as color and orientation within corresponding receptive field

単純・複雑細胞 Simple and Complex Cells at V1

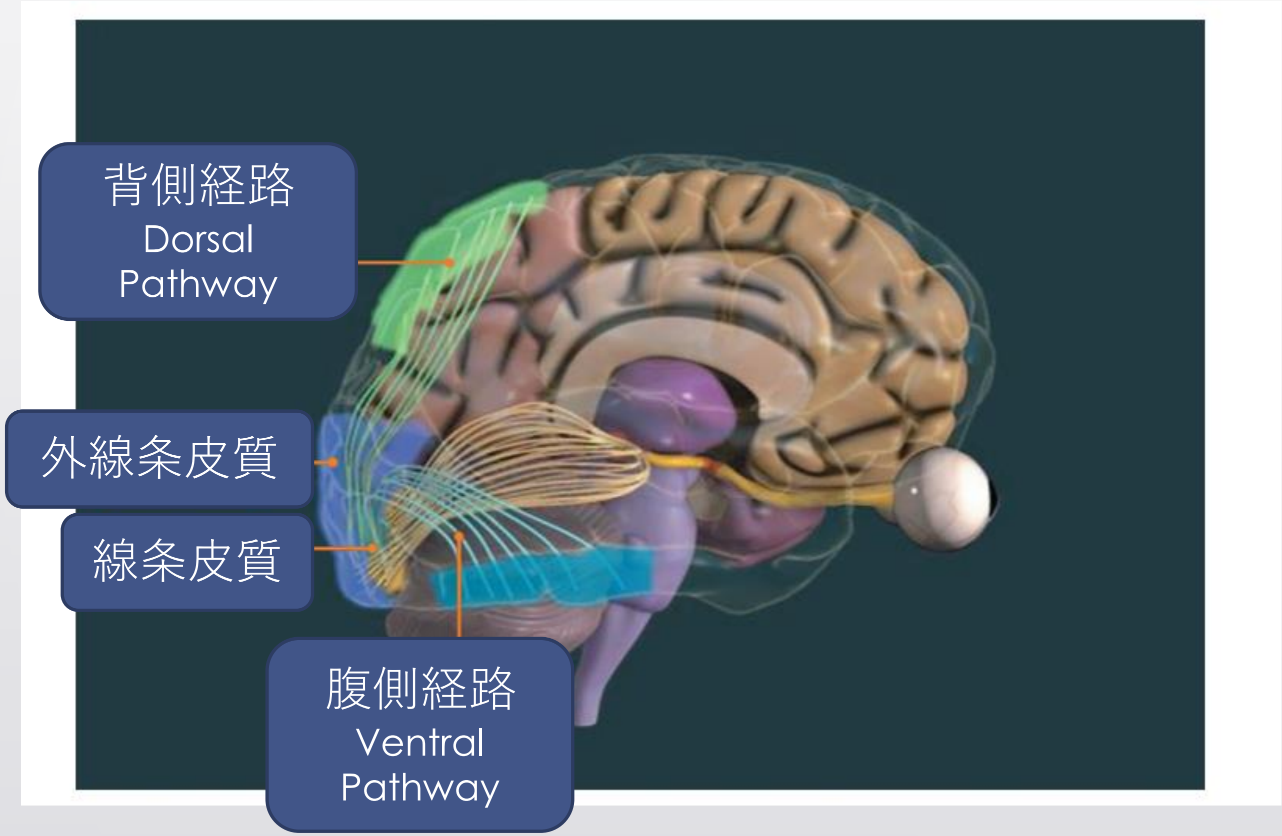


広域の情報を統合することで、
複雑な画像情報が計算される

Complex visual information is computed by integrating features from broad receptive field

Callaway, 2001

高次視覚野 Higher-Order Visual Regions



視覚情報処理の下流で、物体カテゴリーが識別される

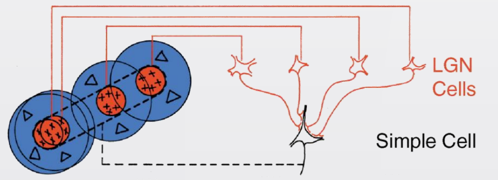
Object category is classified at the downstream of visual processing

CNNと脳の類似性 Similarity between CNN and Brain

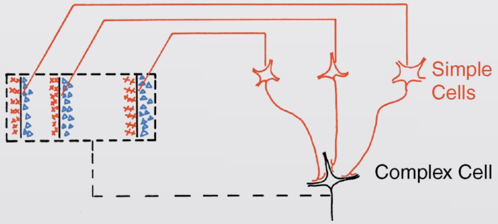
プーリングの効果で受容野が広くなる

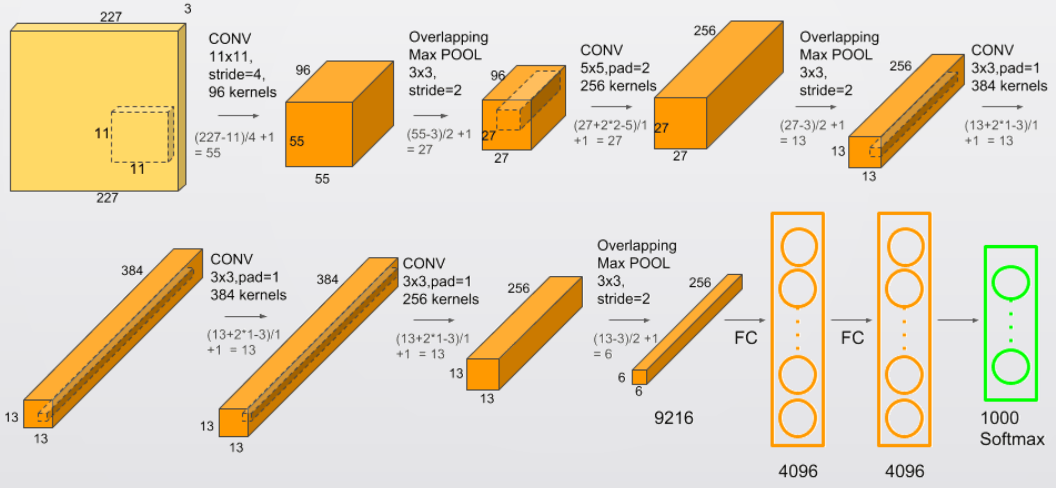
Receptive field gets broader as a result of pooling

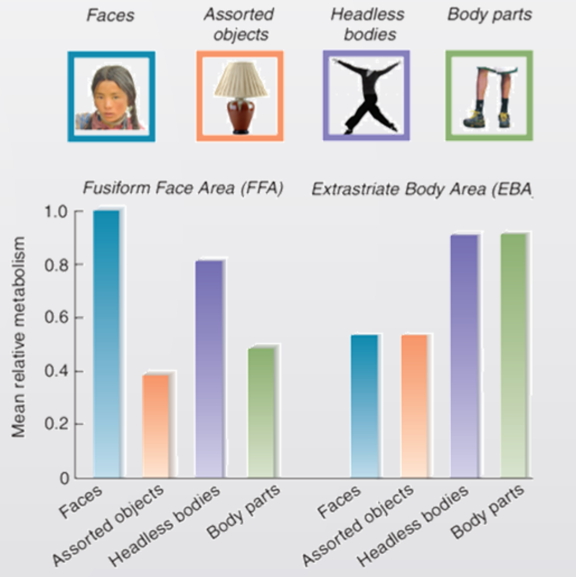
Circuit Building a Simple Cell from LGN Cells



Building a Complex Cell from Simple Cells



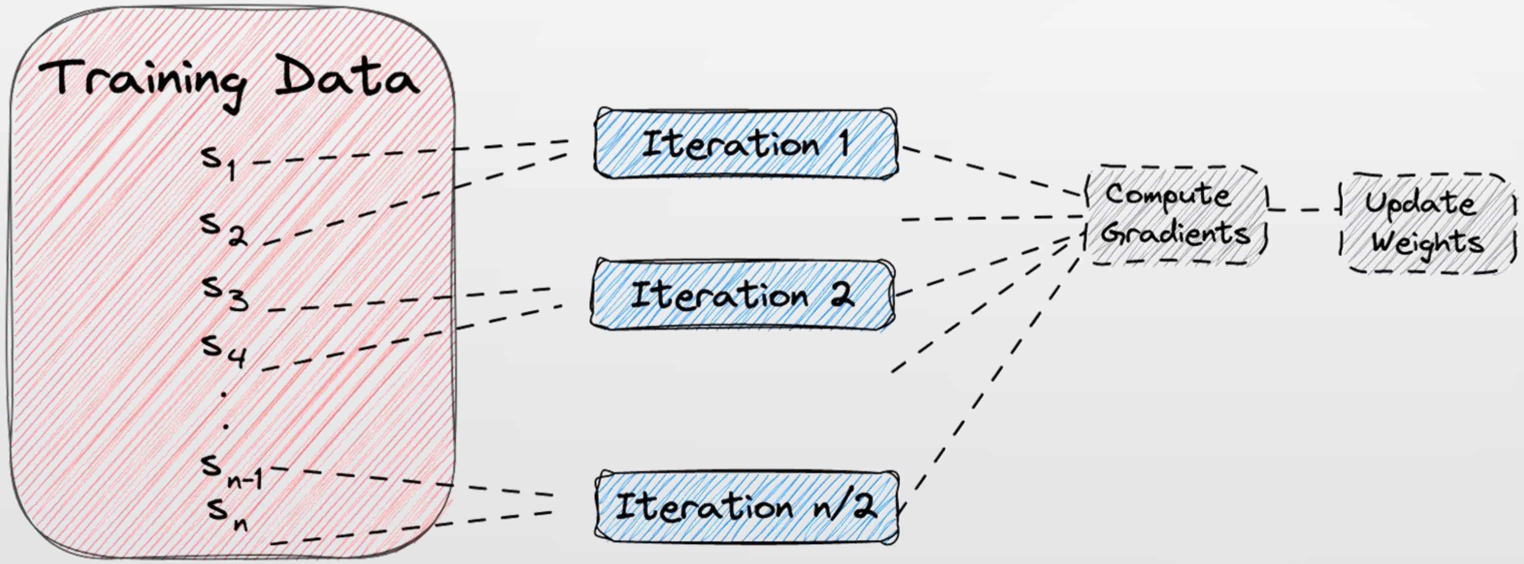




畳み込み層で複雑な情報を表現する

More complex information is represented at convolution layers

バッチ学習 Batch Learning



https://www.baeldung.com/cs/epoch-vs-batch-vs-mini-batch

ランダムに選択したn個のデータをまとめて$batch\ size = n$のミニバッチを作る

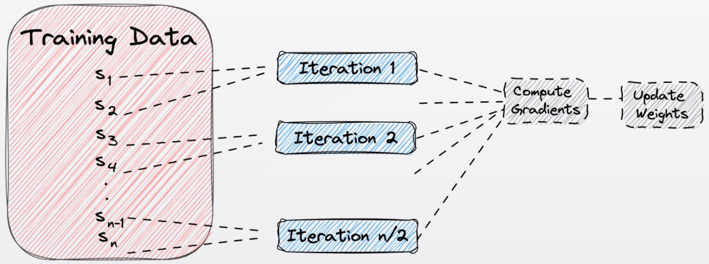
Create mini-batches with $batch\ size = n$ from randomly-selected n data

各ミニバッチのデータで重みを更新する

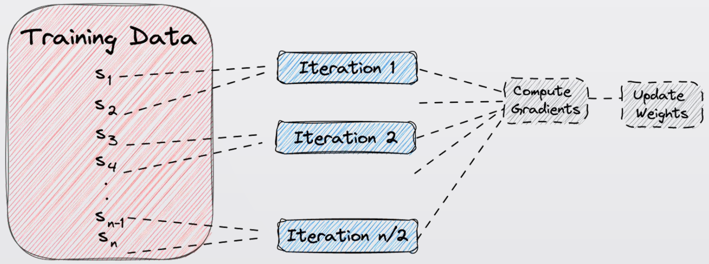
Update weights based on data from single minibatch in each interation

エポック Epoch

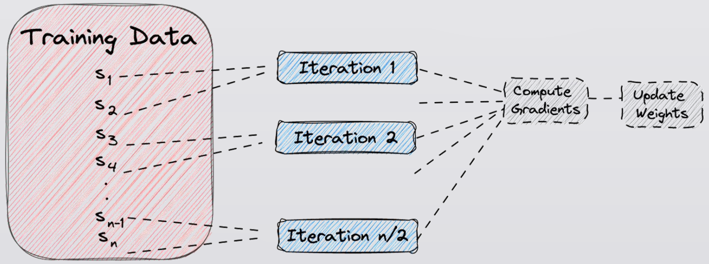
エポック1
Epoch 1



エポック2
Epoch 2



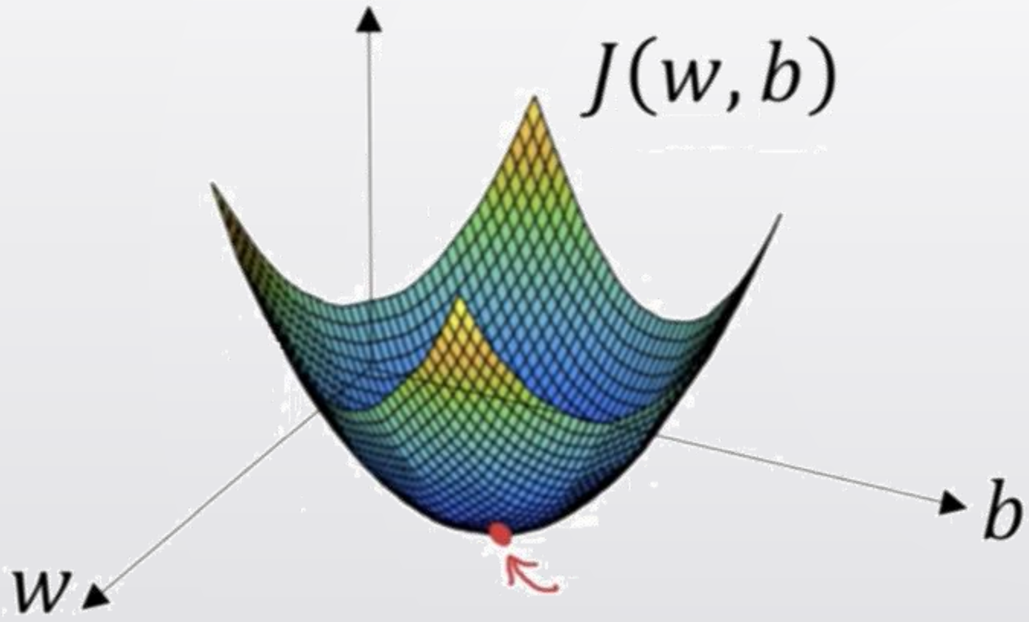
エポック3
Epoch 3



各エポックでは、ミニバッチすべてを用いた重みの更新を一巡する

A cycle of interactions using all the mini-batches is completed in single epoch

勾配降下 Gradient Descent



https://www.linkedin.com/pulse/gradient-descent-its-applications-deep-learning-chirag-subramanian

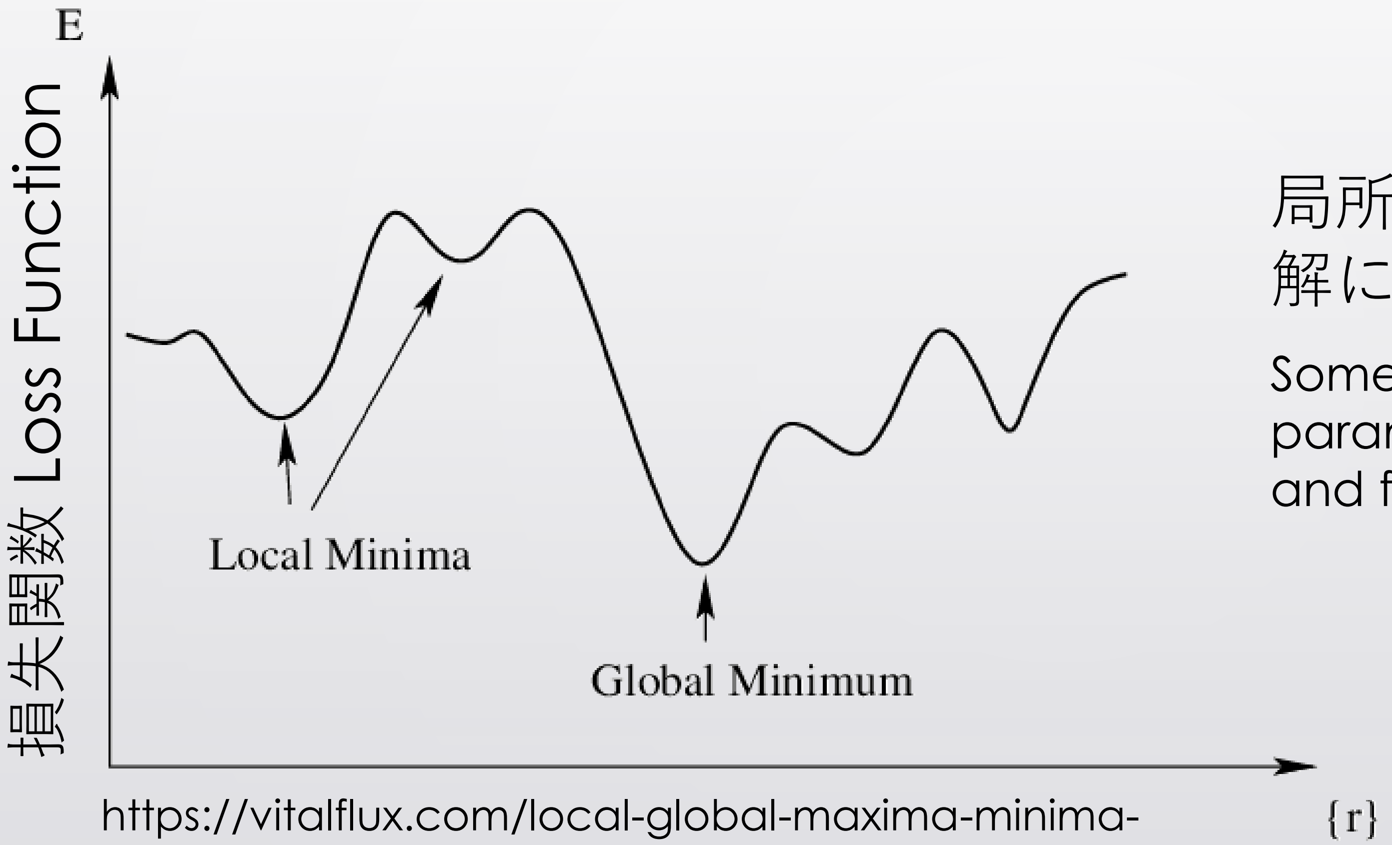
$$-\nabla J = \left(-\frac{\partial J}{\partial w}, -\frac{\partial J}{\partial b}\right)$$

$$w = w - \eta \frac{\partial J}{\partial w}$$

$-\nabla J$ の方向に変数を変化させることで、関数Jの値を最も素早く減少させることが出来る

The output value of function$E$decreases most rapidly along the direction of $-\nabla J$

局所最適と全体最適 Local and Global Minimum



https://vitalflux.com/local-global-maxima-minima-explained-examples/

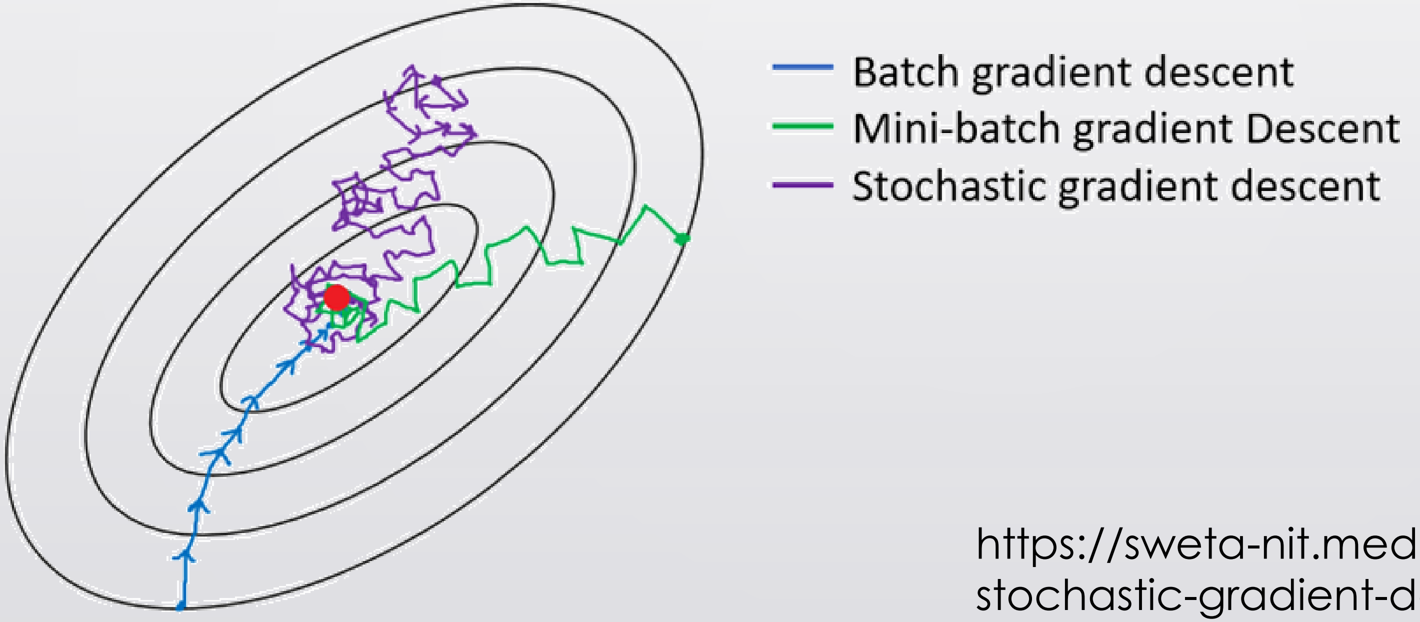
局所最適解にはまって、全体最適解にたどり着かない場合がある

Some times the search for the optimal parameters is caught by local minimum and fails to reach the global minimum

確率的最急降下法 Stochastic Gradient Descent (SGD)

ランダムに選択したデータセットによる重み更新の繰り返しで、局所最適解に陥るのを防ぐ

Avoid local minimum by weight updating using a set of randomly-selected data

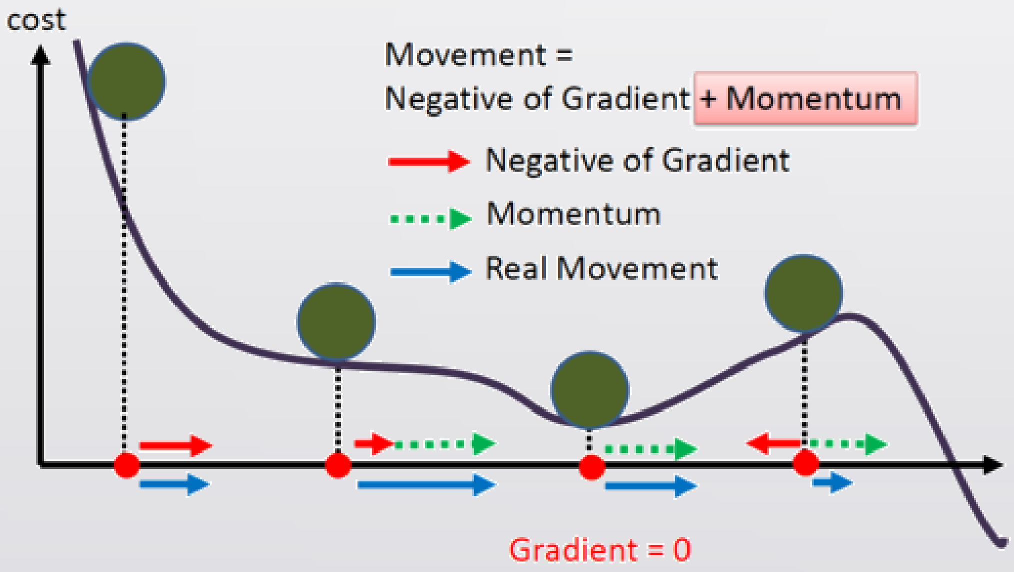


https://sweta-nit.medium.com/batch-mini-batch-and-stochastic-gradient-descent-e9bc4cacd461

モメンタム Momentum

$$w \leftarrow w - \eta \nabla E$$

$$w \leftarrow w + m$$
$$m \leftarrow \alpha m - \eta \nabla E$$



https://medium.com/analytics-vidhya/momentum-rmsprop-and-adam-optimizer-5769721b4b19

前の更新と同じ方向に勾配降下を行わせる慣性項

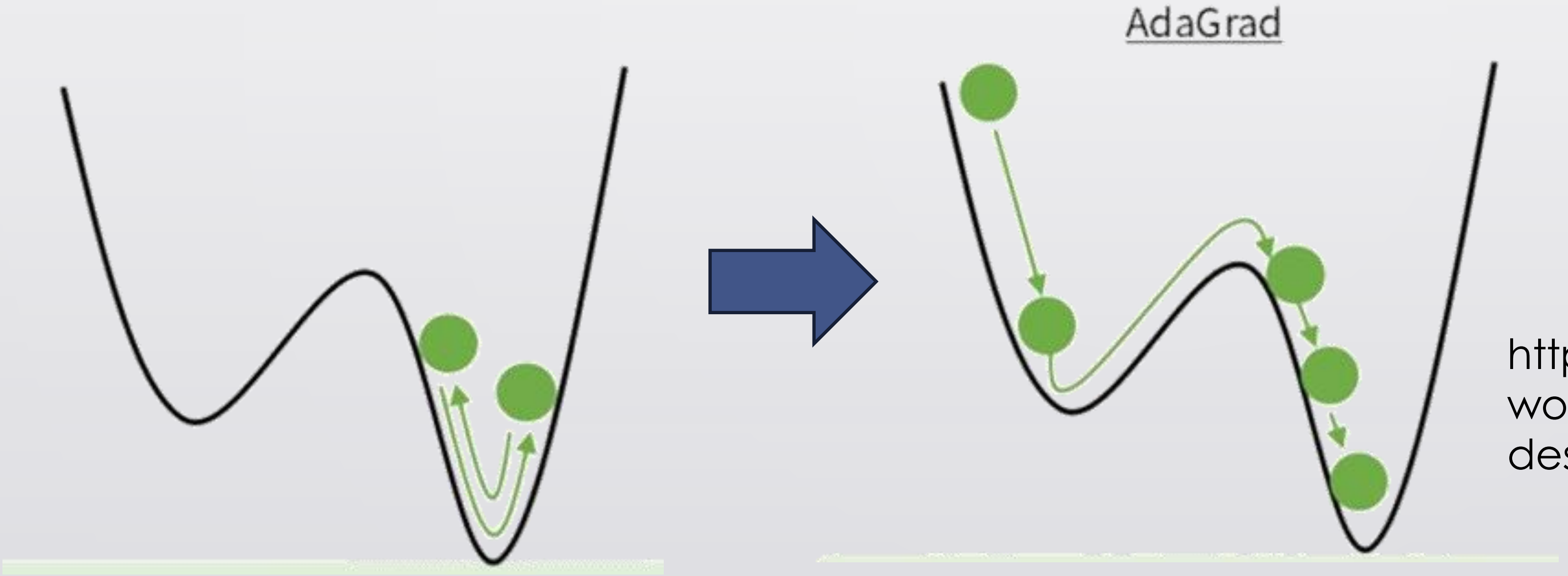
Momentum to induce gradient descent in the same direction as the preceding step

AdaGrad

$$h \leftarrow h + (\nabla E)^2 \qquad w \leftarrow w - \frac{\eta}{\underline{\sqrt{h+\varepsilon}}}\nabla E$$

ステップ毎にhが大きくなるので、学習率が減衰する

Learning rate decays as h increases at each step



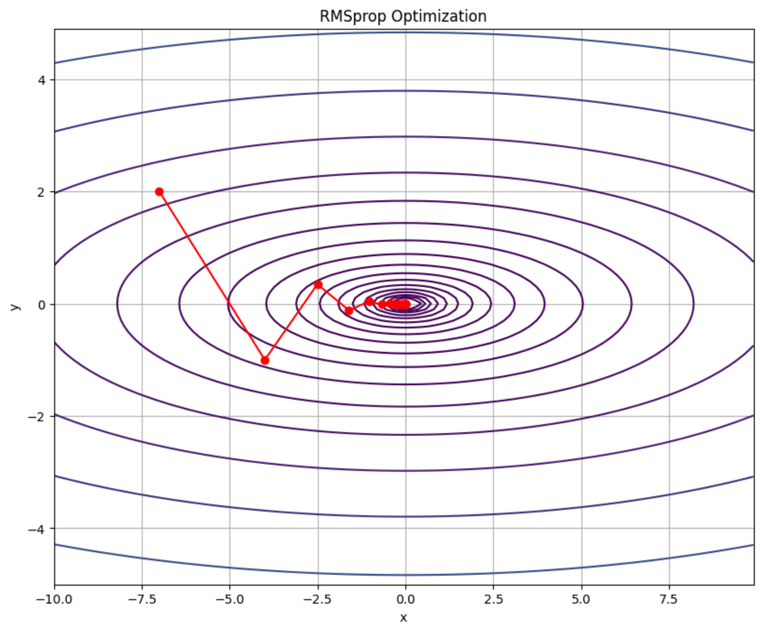
https://zero2one.jp/ai-word/problems-in-gradient-descent-methods/

RMSProp

$$h \leftarrow \beta h + (1 - \beta)(\nabla E)^2 \qquad w \leftarrow w - \frac{\eta}{\sqrt{h + \varepsilon}} \nabla E$$

勾配の二乗の移動平均を取り、学習率の急激な変化を抑制

Avoid abrupt change of learning rate by computing moving average of squared gradient



https://tt-tsukumochi.com/archives/8866

Adam (Adaptive Moment)

モメンタム Momentum

$$m \leftarrow \alpha m + (1-\alpha)\nabla E$$

$$w \leftarrow w - \eta \frac{\hat{m}}{\sqrt{\hat{h}+\varepsilon}}$$

$$\hat{m} \leftarrow \frac{m}{1-\alpha^t}$$

t: ステップ数
Step Number

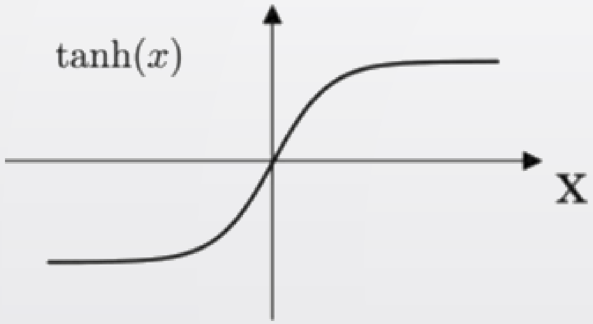
$$h \leftarrow \beta h + (1-\beta)(\nabla E)^2$$

$$\hat{h} \leftarrow \frac{h}{1-\beta^t}$$

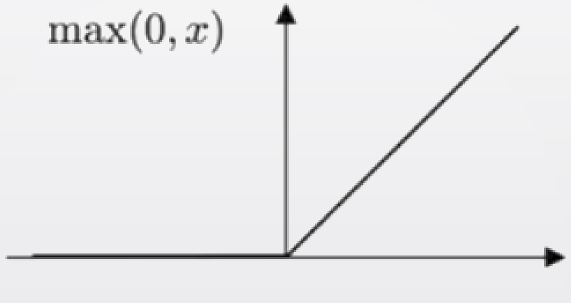
RMSProp

活性化関数 Activation Function

Tanh

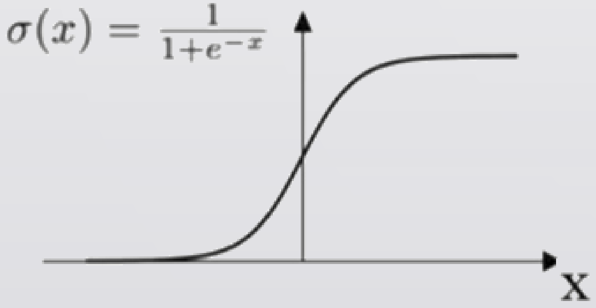


ReLU

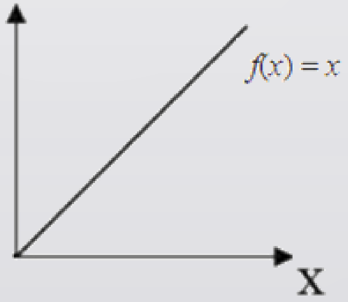


$$f(x) = \begin{cases} x \ (x \geq 0) \\ 0 \ (x < 0) \end{cases}$$

Sigmoid



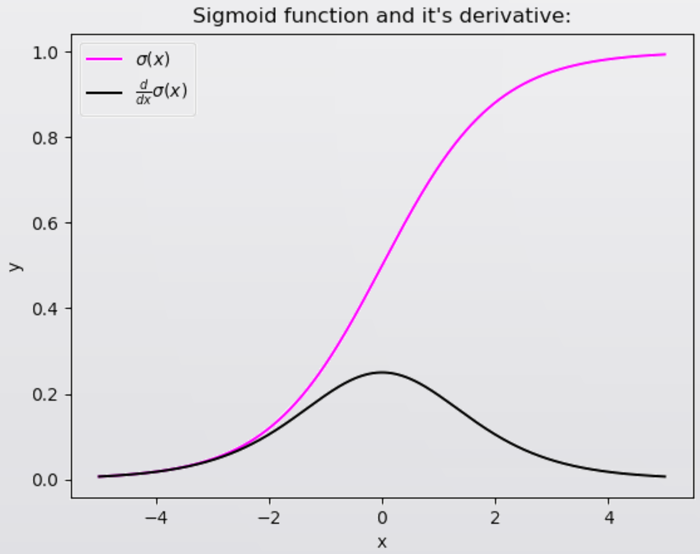
Linear



https://machine-learning.paperspace.com/wiki/activation-function

勾配消失問題 Vanishing Gradient Problem

$$\frac{\partial v_k^{(2)}}{\partial w_{ji}^{(1)}} = = x_i^n w_{kj}^{(2)} f'\left(v_j^{(1)}\right)$$



各層への入力が 0 から遠いと、シグモイド関数の微分が0に漸近する

If input to each layer deviates from zero, derivative of sigmoid function gets close to zero

重みが更新されなくなる

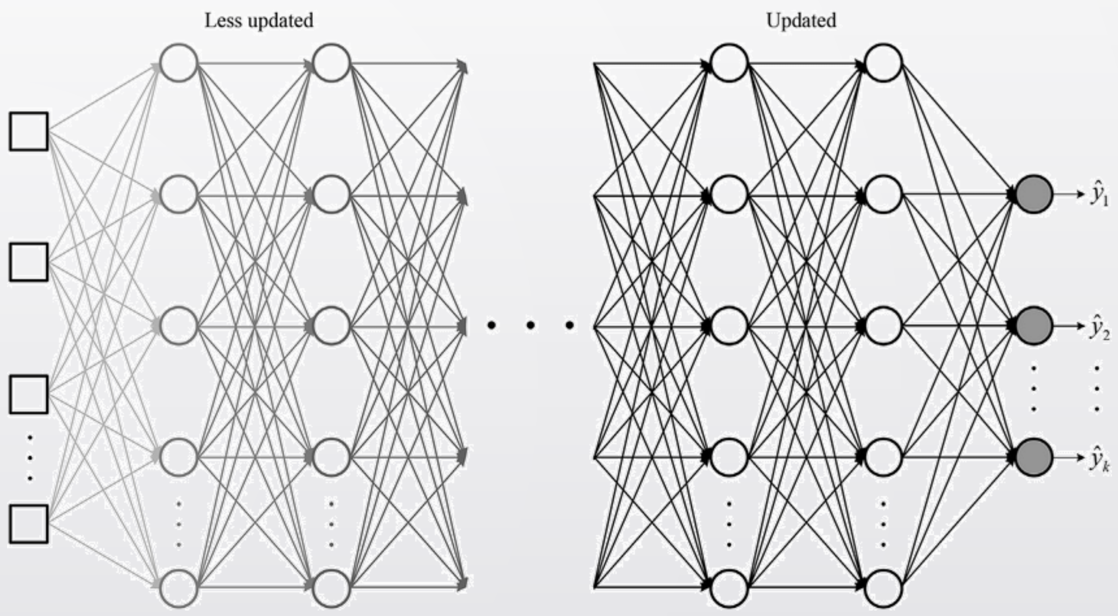
Wight updating gets almost halted

https://www.analyticsvidhya.com/blog/2021/06/the-challenge-of-vanishing-exploding-gradients-in-deep-neural-networks/

勾配消失問題 Vanishing Gradient Problem

$$\delta_k^{(2)} = \left(y_k^{(2)} - t_k\right) f'\left(\sum_{j=0}^{j=M} w_{kj}^{(2)} y_j^{(1)}\right)$$

$$\delta_j^{(1)} = f'\left(v_j^{(1)}\right) \sum_{k=1}^{k=C} w_{kj}^{(2)} \cdot \delta_k^{(2)}$$

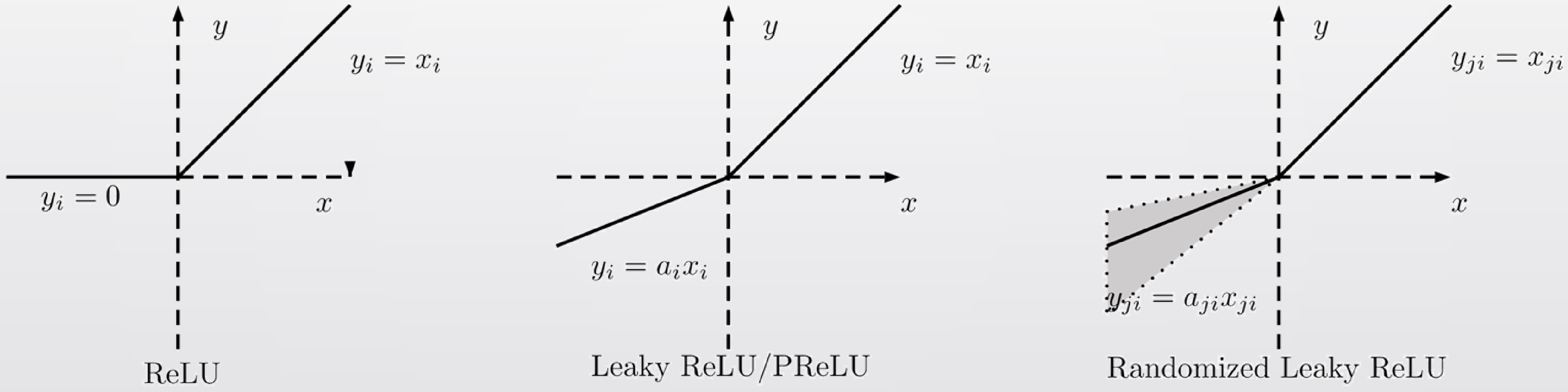


Koo et al, 2018

勾配消失問題は入力層に近い層でより深刻

Vanishing gradient problem is severer in layers closer to the input layer

ReLuとLeaky ReLu
Rectified Linear Unit (ReLu) and Leaky ReLu

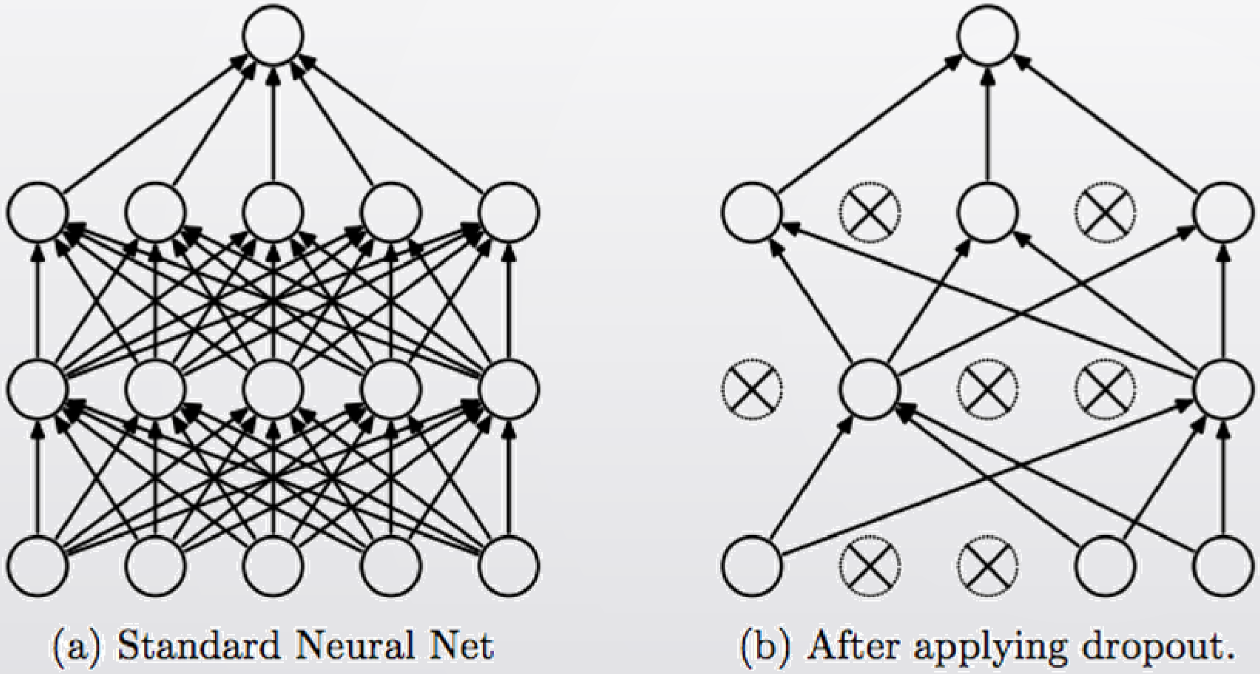


https://paperswithcode.com/method/rrelu

Leaky ReLuでは入力が0以下でも勾配が発生する

In contrast to ReLu, Leaky ReLu retains non-zero gradient for input below zero

ドロップアウト Dropout



ミニバッチごとにネットワークのノードをランダムに取り除く

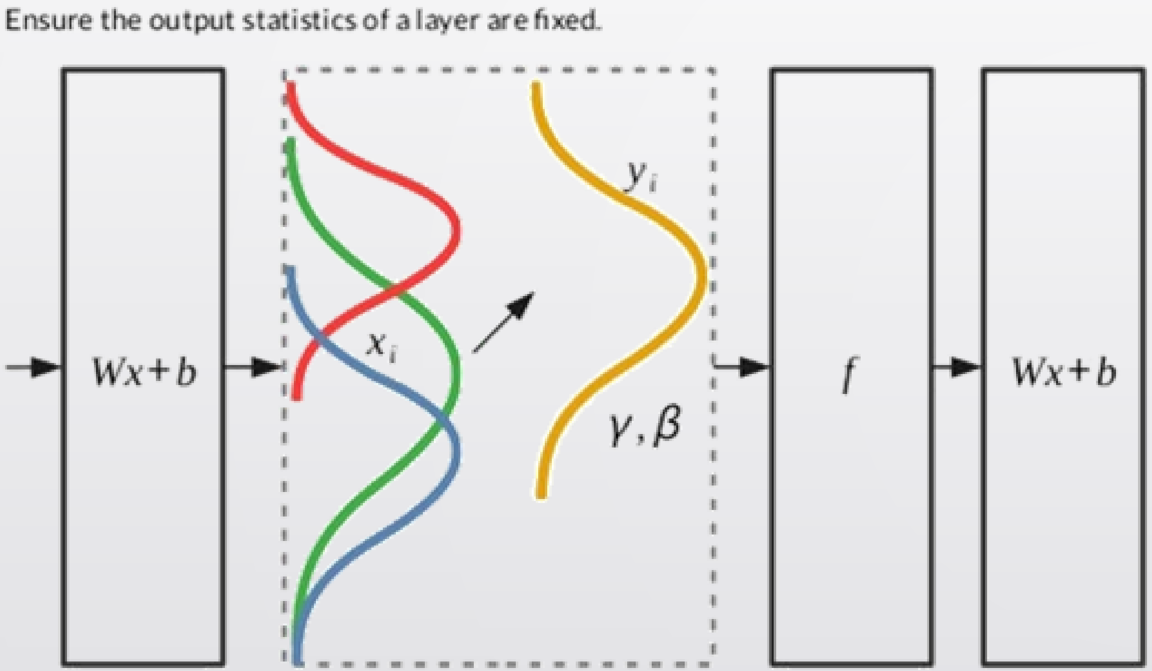
Remove randomly-selected nodes from network for each mini-batch

過学習を抑制する効果がある

Effective in suppressing overfitting

Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting", JMLR 2014

バッチ正規化 Batch Normalization



https://www.srose.biz/wp-content/uploads/2020/08/Deep-Learning-Performance-Part-3-Batch-Normalization-Dropout-Noise.html

バッチ毎に各層への入力を正規化する

Normalize inputs to layers within each single batch

$$y = \gamma \frac{x - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}} + \beta$$

学習の安定化、過学習の抑制、勾配消失問題の緩和に効果がある

Effective in stabilization of learning process, suppression of overfitting and vanishing gradient problem