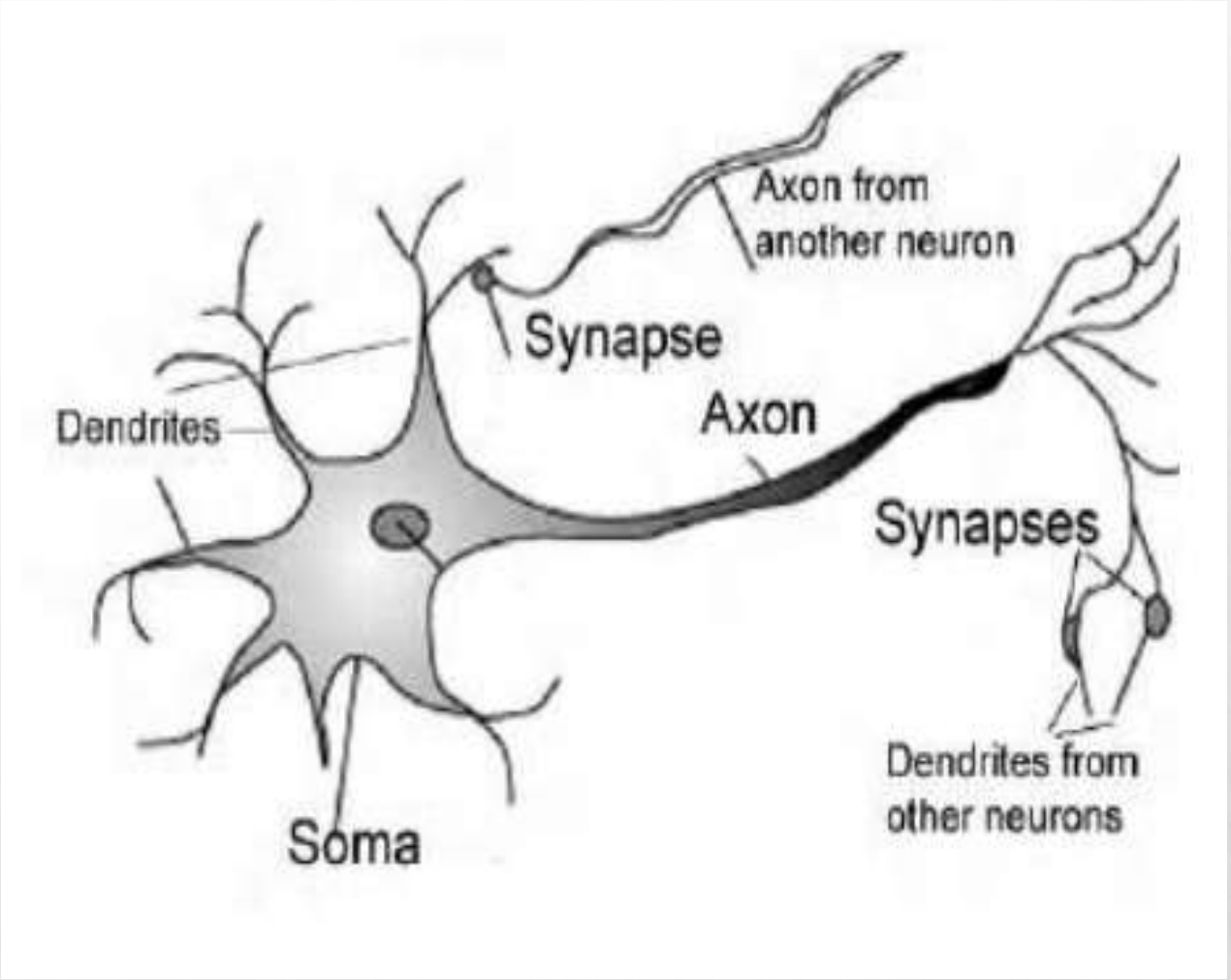
CONVOLUTIONAL NEURAL NETWORK

Neural Network & Deep Learning

Algoritma yang mengimitasi kemampuan otak manusia (*biological neuron*) untuk mengenali pola tertentu dan dapat melakukan proses pembelajaran.

Neuron

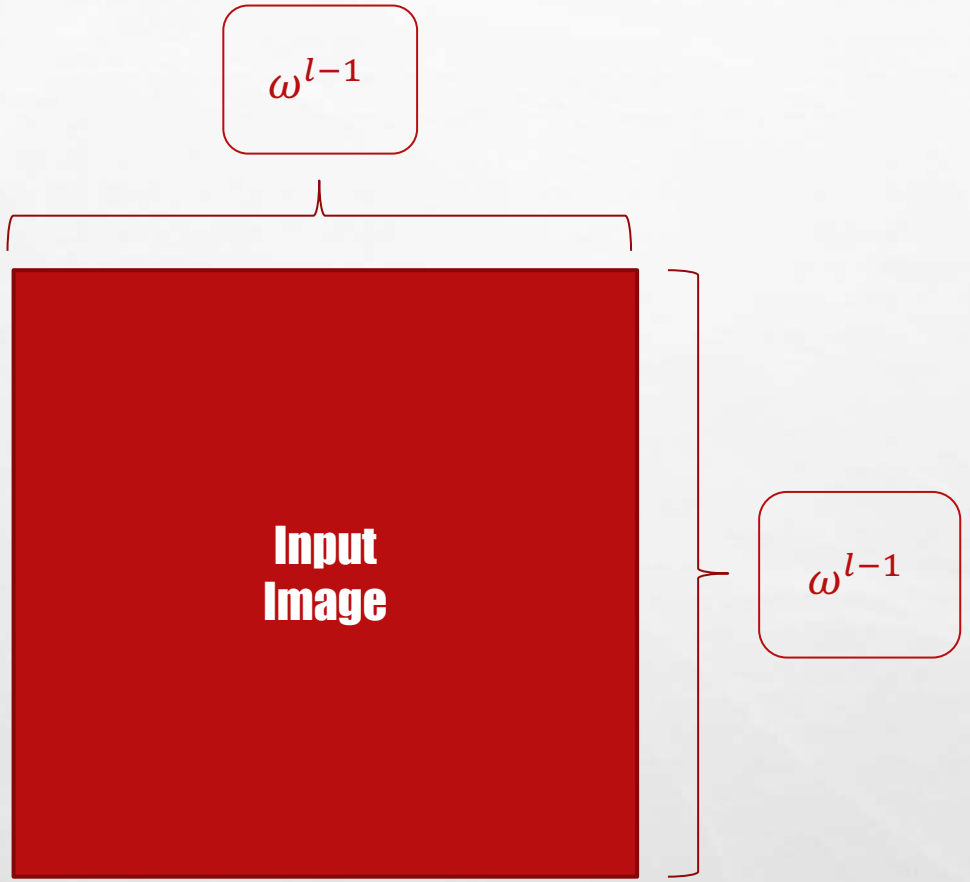
Unit Cerdas yang Memiliki Kemampuan Untuk Belajar



$$z = \theta_1 a_1 + \theta_2 a_2 + \cdots + \theta_n a_n + b$$

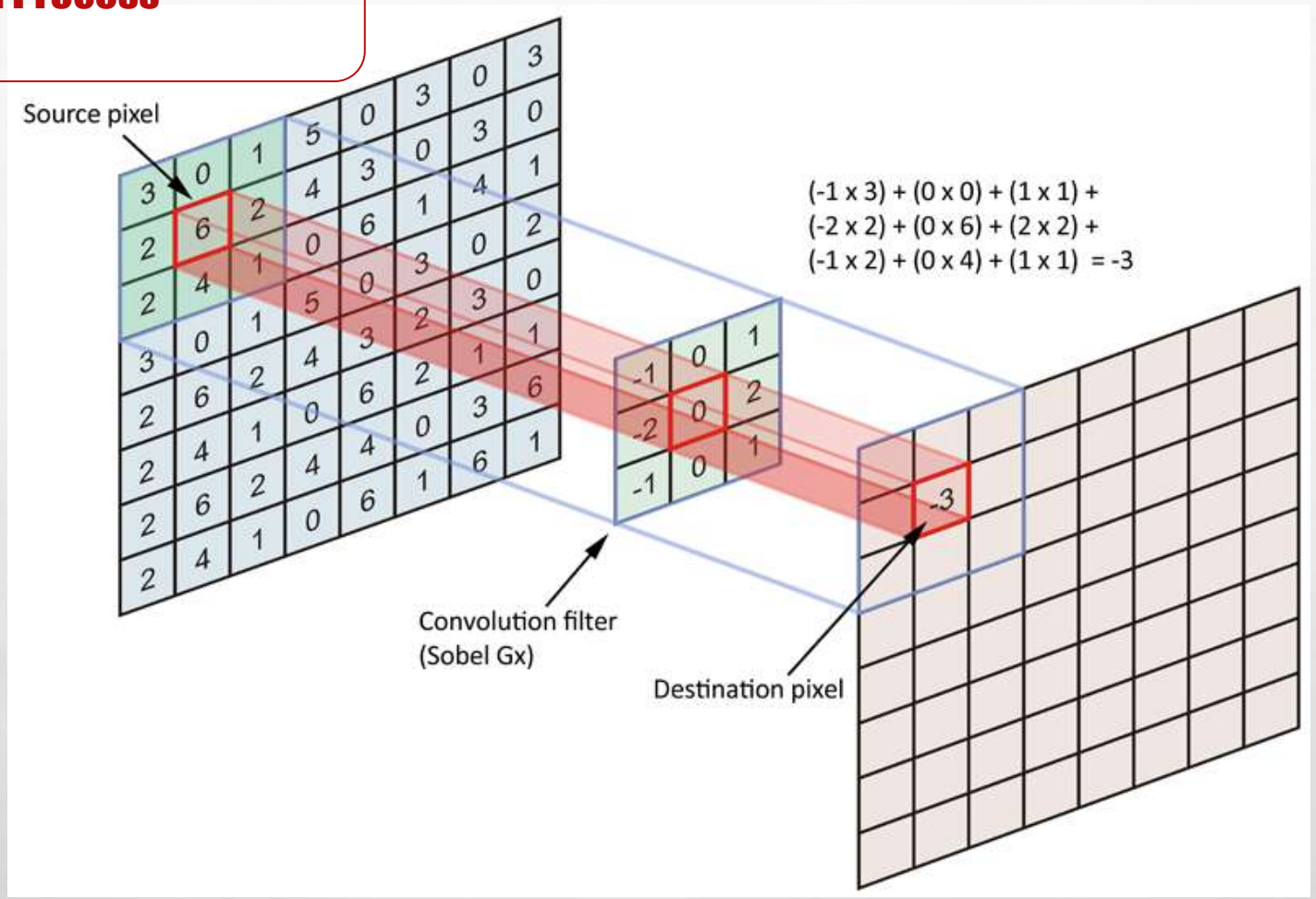
Convolutional Neural Network

1. Local receptive fields
2. Shared weights
3. Spatial subsampling

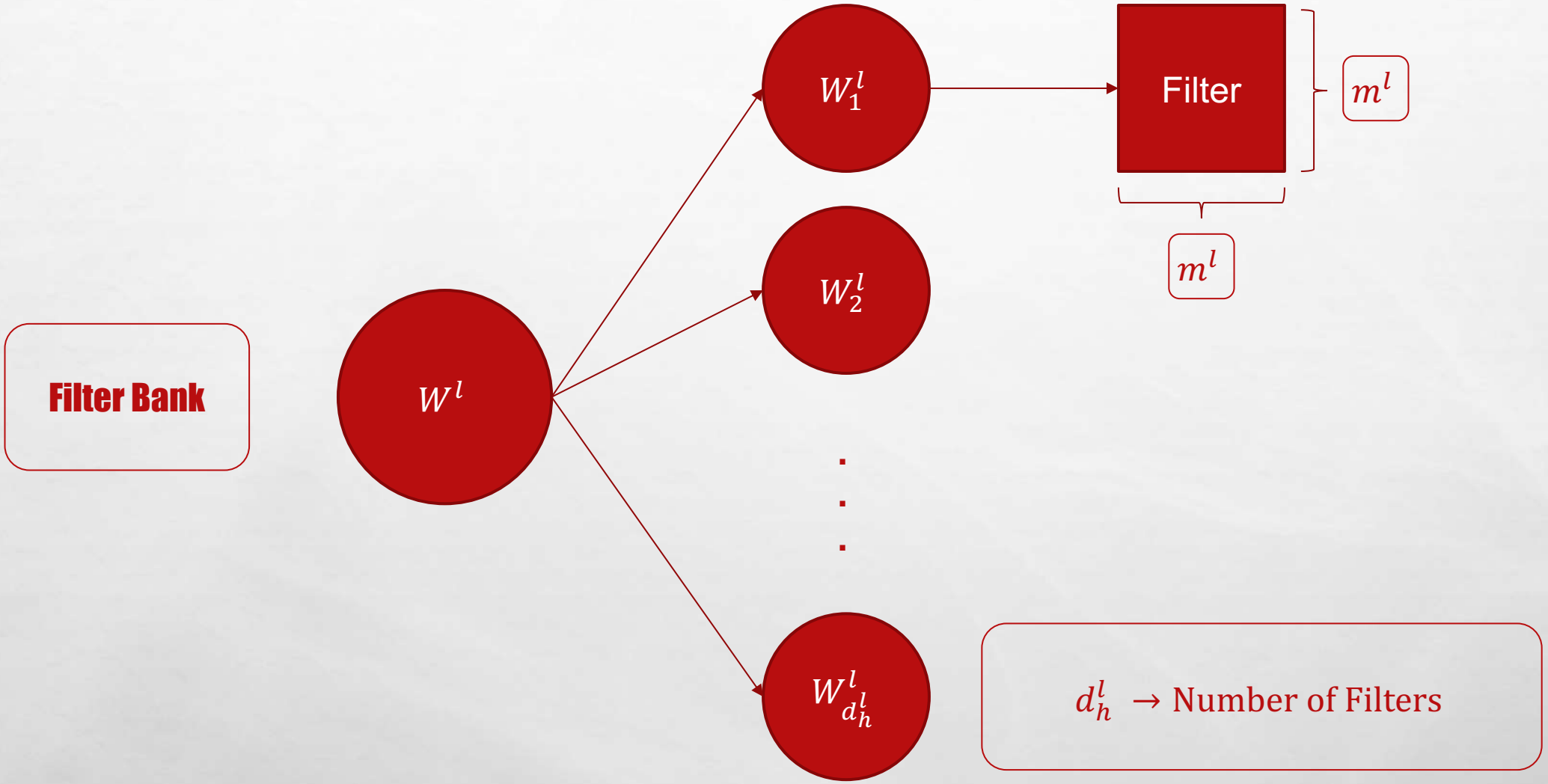


INPUT

Convolution Process



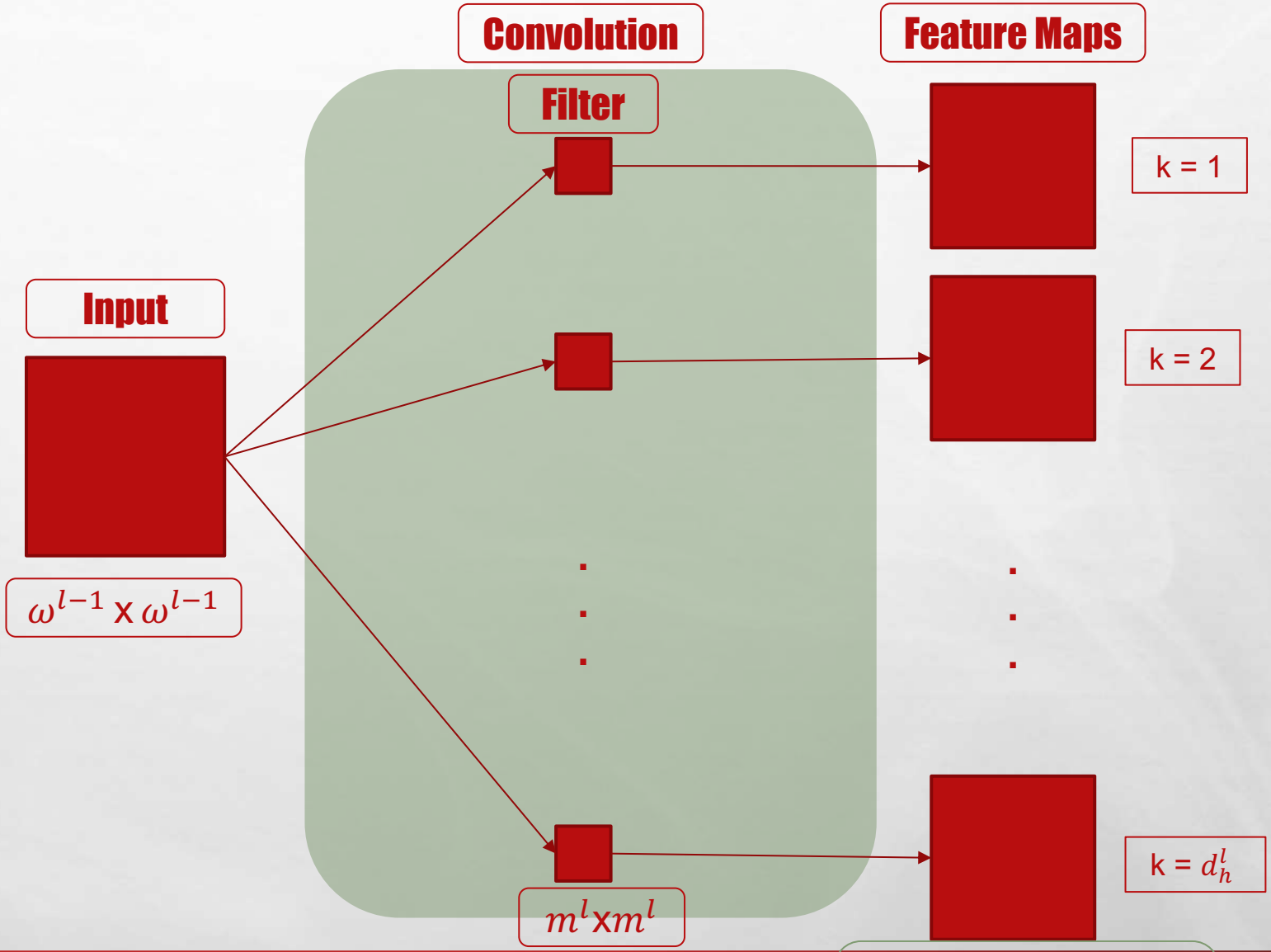
CONVOLUTIONAL LAYER



CONVOLUTIONAL LAYER

Feature Maps

$$g_k^l = I_p^{l-1} * W_k^l$$



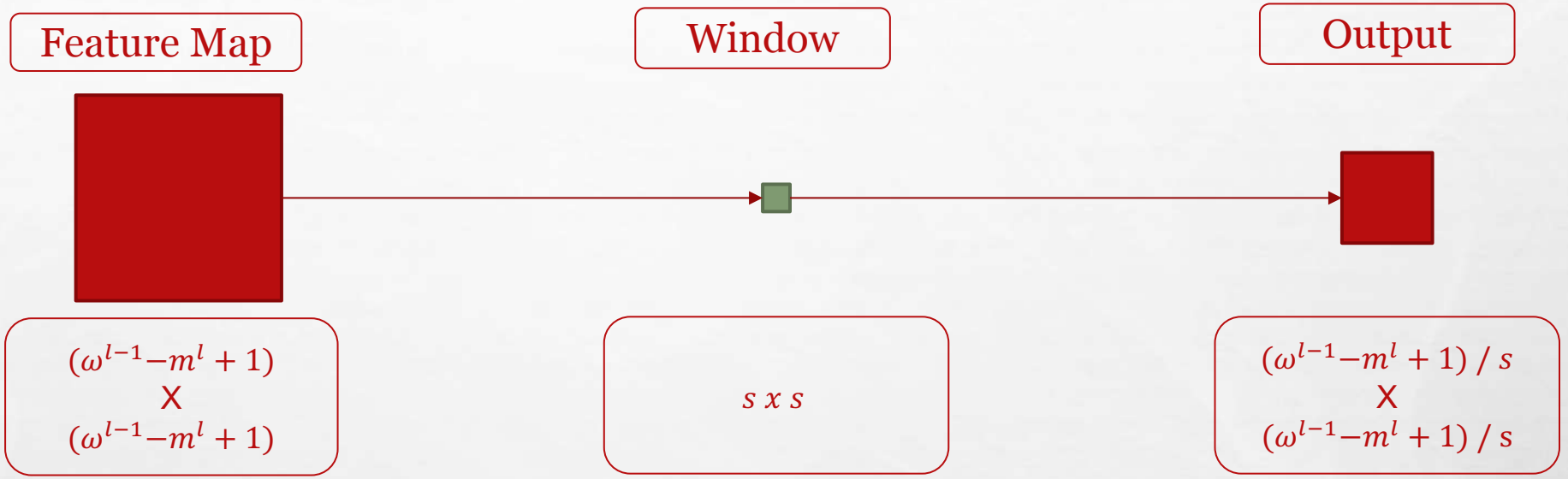
CONVOLUTIONAL LAYER

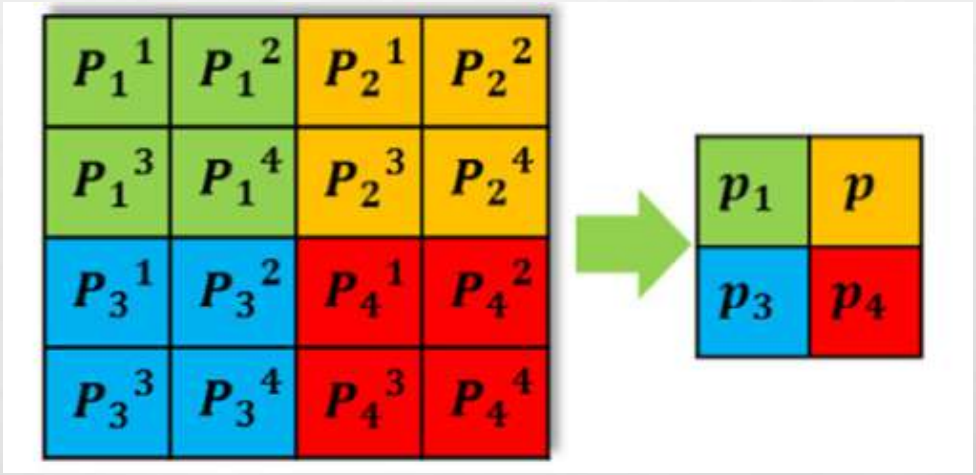
MAX-POOLING LAYER

Max Pooling

Reduce Resolution

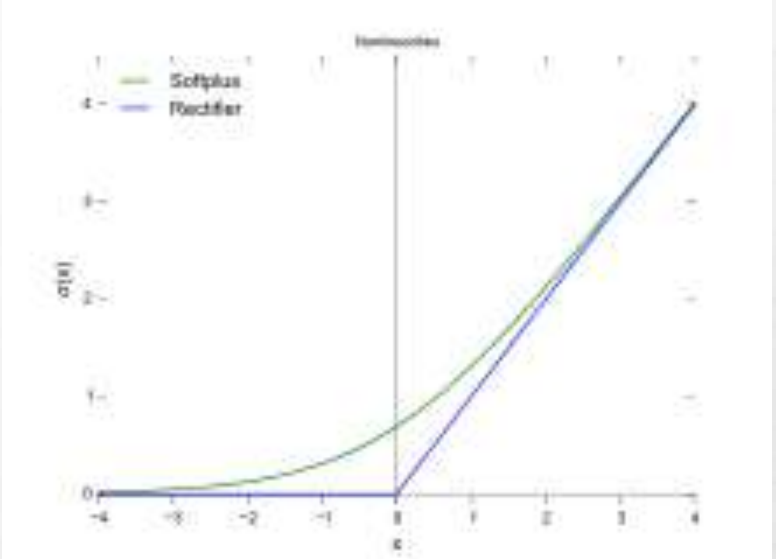
Spatial Invariance

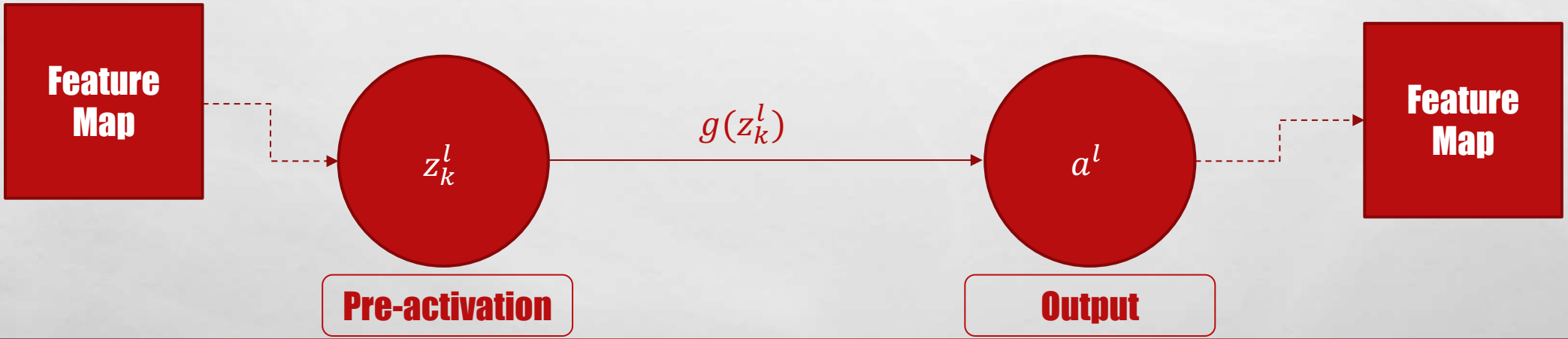




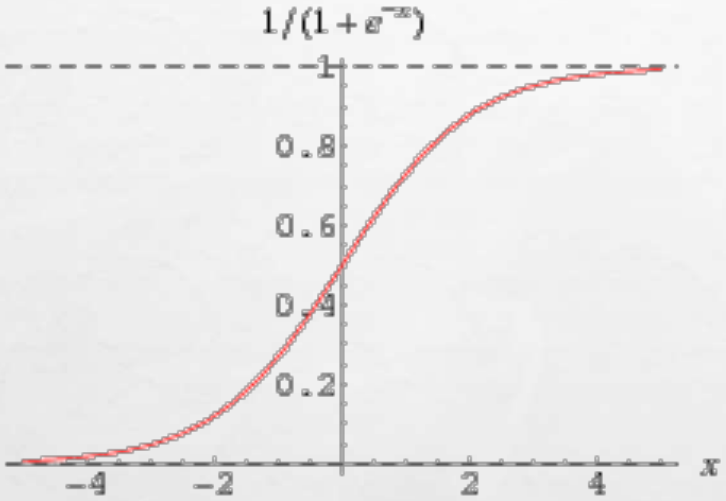
Activation Function: Relu

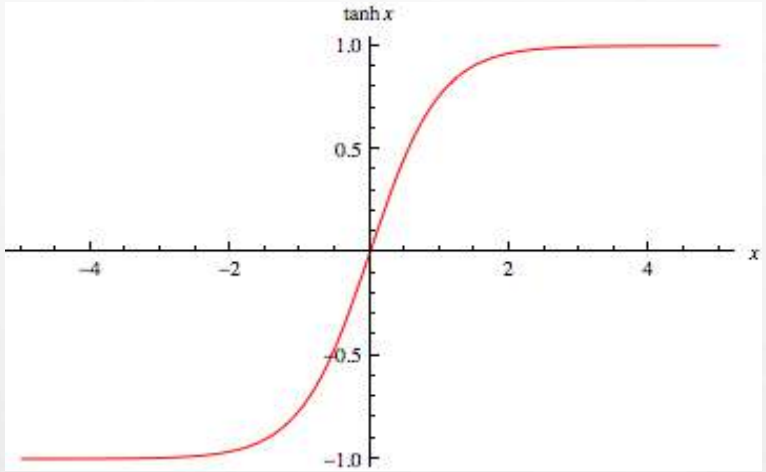
$$x^l = f(g_k^l) = \max(0, x)$$

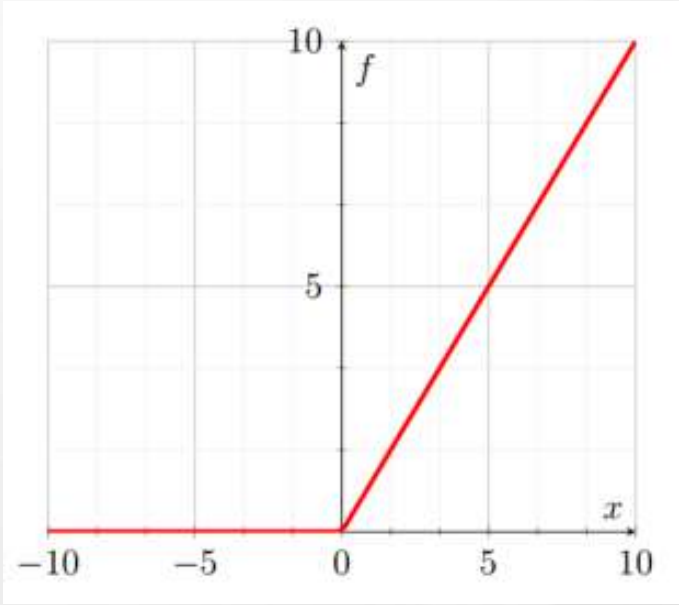




ACTIVATION FUNCTION







Sigmoid

$$g(z) = \frac{1}{1 + e^{-z}}$$

Tanh

$$g(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

Relu

$$g(z) = \max(0, z)$$

ACTIVATION FUNCTION

Loss function: Gap antara prediction dan actual

$$J\left(a^{(L)}, y\right) = a^{(L)} - y$$

Loss function Regresi

Mean Absolute Error	$MAE\left(a^{(L)}, y\right) = \frac{1}{n}\sum_{i=1}^{n}\lvert a_i^L - y\rvert$	(2.13)
Mean Square Error	$MSE = \frac{1}{n}\sum_{i=1}^{n}(a_i^L - y)^2$	(2.14)
Mean Square Logarithmic Error	$MSLE = \frac{1}{n}\sum_{i=1}^{n}(\log(a_i^L + 1) - \log(y + 1))^2$	(2.15)
L1	$L1 = \sum_{i=1}^{n}\lvert a_i^L - y\rvert$	(2.16)
L2	$L2 = \sum_{i=1}^{n}(a_i^L - y)^2$	(2.17)

Loss function Klasifikasi

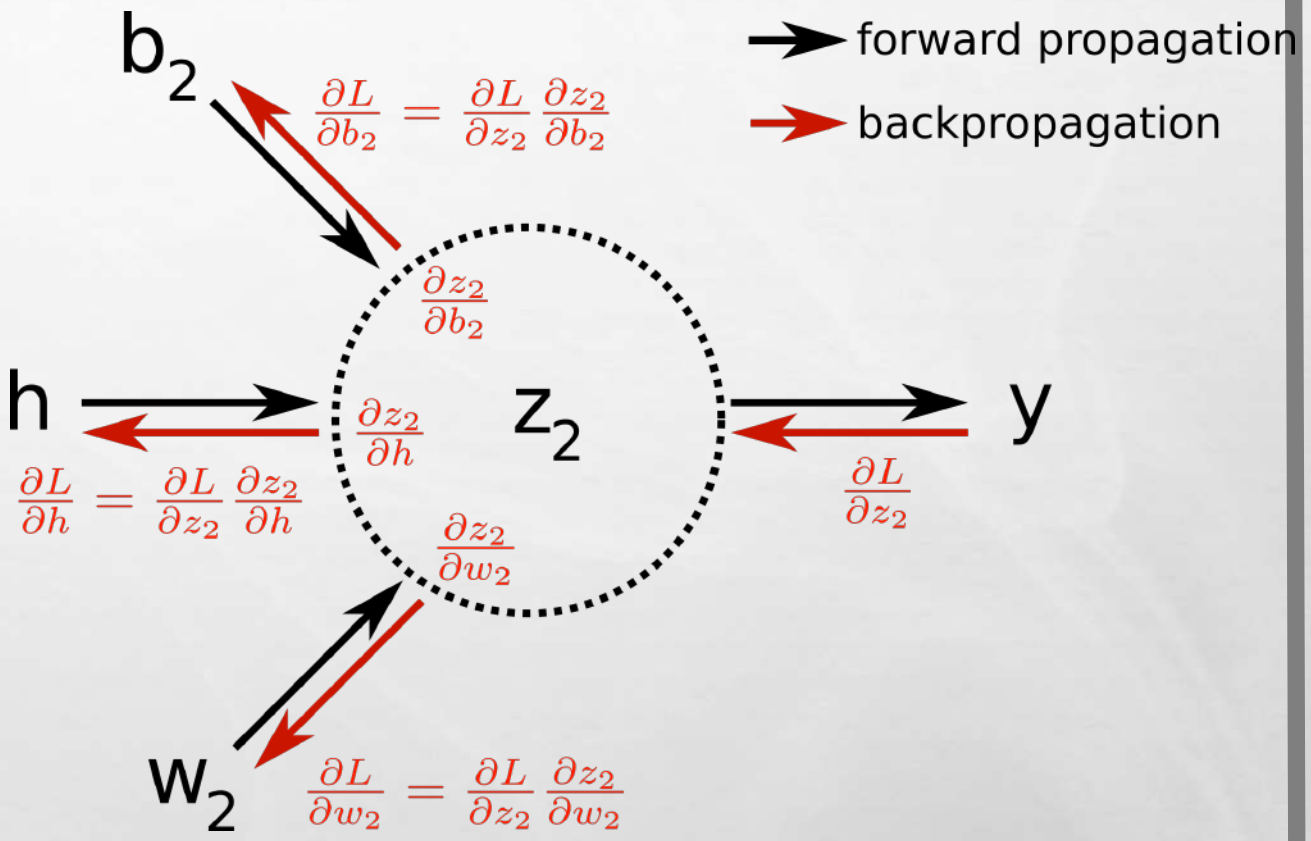
Negative log likelihood	$NLL = -\frac{1}{n}\sum_{i=1}^{n}\log(a_i^L)$	(2.18)
Binary Cross Entropy	$BCE = -\left(y_i \log(a_i^L)\right) + \left((1 - y_i)\log(1 - a_i^L)\right)$	(2.19)
Cross Entropy	$CE = -\sum_{i=1}^{n} y \log(a_i^L))$	(2.20)
Kullback Leibler	$KL = \frac{1}{n}\sum_{i=1}^{n}(y_i \log(y_i)) - \frac{1}{n}\sum_{i=1}^{n} y_i \log(a_i^L))$	(2.21)
Hinge	$HINGE = \frac{1}{n}\sum_{i=1}^{n}\max(0, 1 - y_i \cdot a_i^L)$	(2.22)

LOSS FUNCTION

$$\delta^{(l)} = \sum_{i=1}^{n} \left(\delta_i^{(l+1)} \cdot \theta_i^{(l+1)} \right) \cdot g'\left(z^{(l)}\right)$$

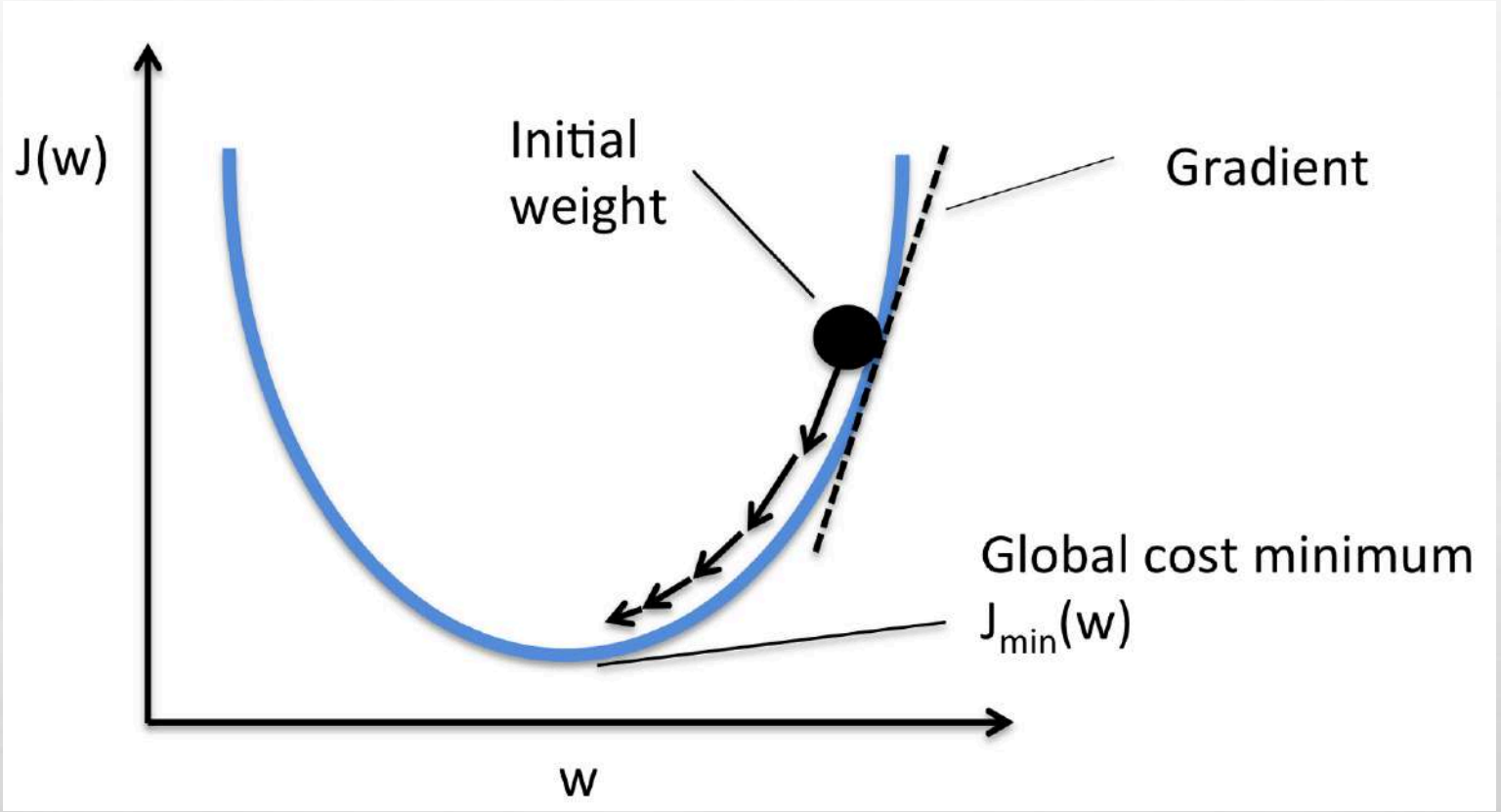
$$\delta_\theta^{(l)} = \delta^{(l)} \cdot a^{(l-1)}$$

$$\delta_b^{(l)} = \delta^{(l)}$$



LEARNING ALGORITHM (BACKPROPAGATION)

$$\theta_{new}^{l} = \theta_{old}^{l} - \gamma \frac{\partial J(a^{(L)}, y)}{\partial \theta}$$



GRADIENT DESCENT

TERIMA KASIH