Modul Praktikum Komunikasi Data

Happy Nugroho, S.T., M.T.

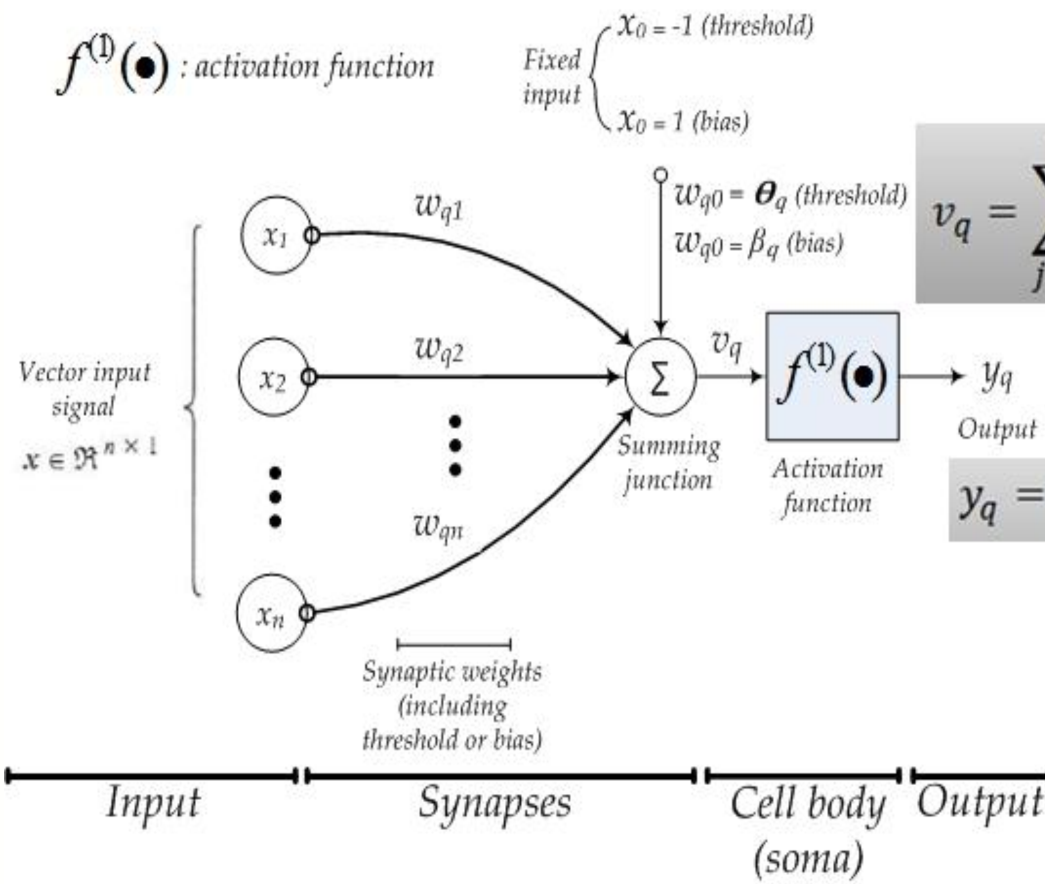
SEMESTER III

PRODI TEKNIK ELEKTRO

FAKULTAS TEKNIK

UNIVERSITAS MULAWARMAN SAMARINDA

General ANNs Basic Models



$$v_q = \sum_{j=0}^{n} w_{qj} x_j$$

$$y_q = f(v_q)$$

Alternative model

- u_q is a linier combiner of input (x_j) and synaptic weight (w_{qj})

$$u_q = \sum_{j=1}^{n} w_{qj} x_j = w_q^T x = x^T w_q$$

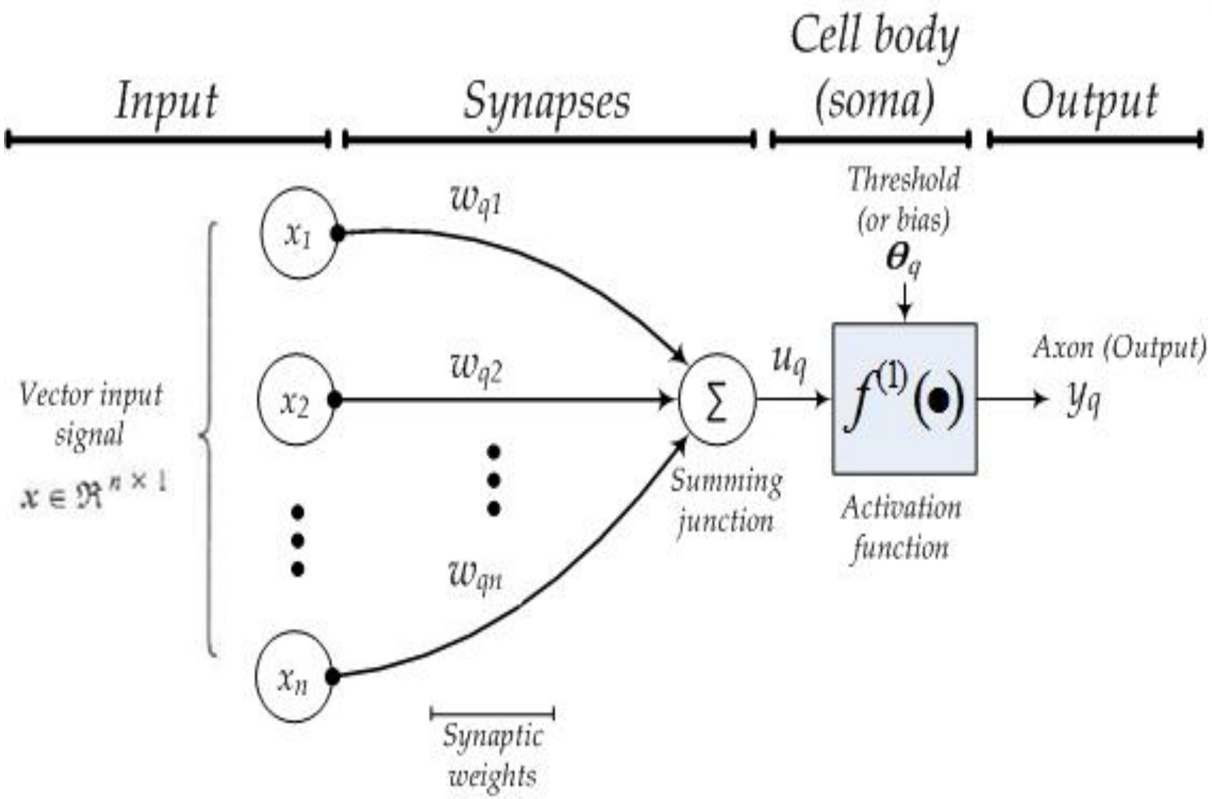
- Activation potensial

$$v_q = u_q - \theta_q$$

- Output of activation function:

$$y_q = f(v_q) = f(\sum_{j=1}^{n} w_{qj} x_j - \theta_q)$$

General ANNs Basic Models



$f^{(1)}(\bullet)$: activation function

Nonlinear model of an ANN

- u_q *is a linier combiner of input* (x_j) *and synaptic weight* (w_{qj})

$$u_q = \sum_{j=1}^{n} w_{qj} x_j = w_q^T x = x^T w_q$$

- *Activation potensial*

$$v_q = u_q - \theta_q$$

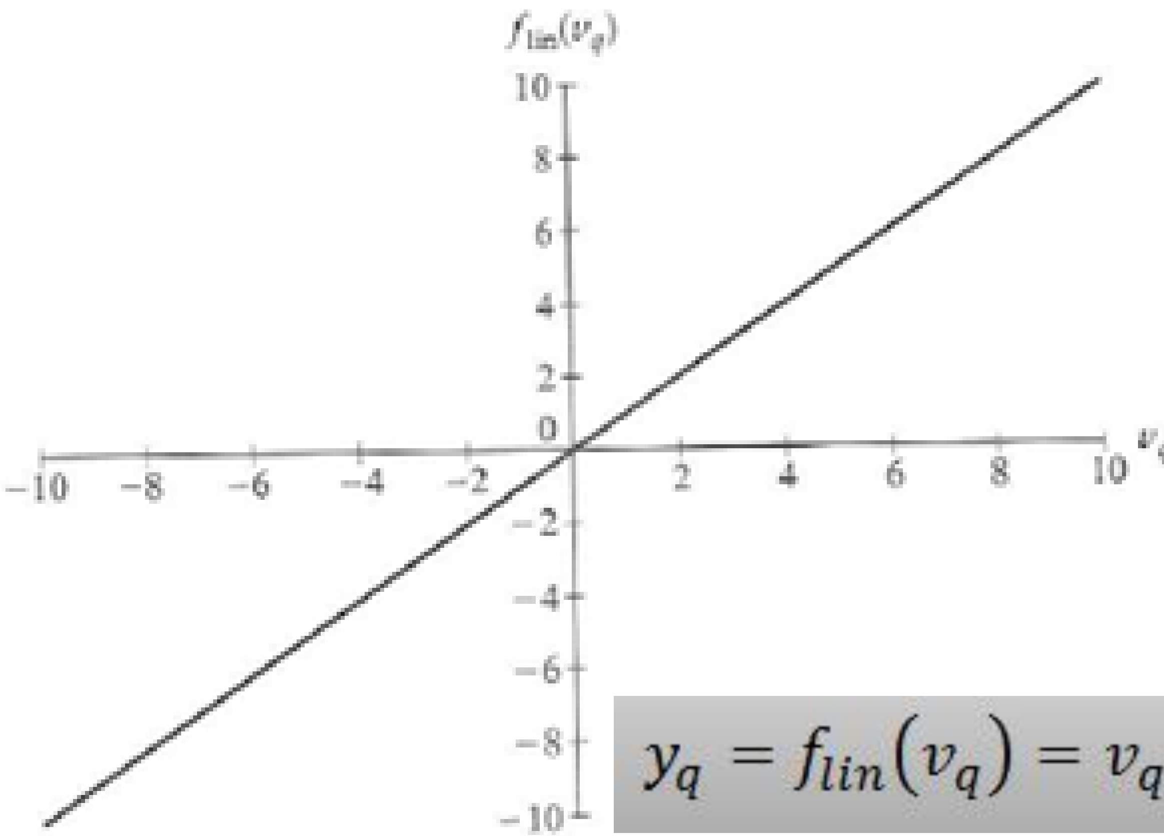
- *Output of activation function:*

$$y_q = f(v_q) = f(\sum_{j=1}^{n} w_{qj} x_j - \theta_q)$$

Activation Functions

- *Activation function can be a linear or nonlinear function.*
- *Selection of activation function depends on particular problem to be solved.*

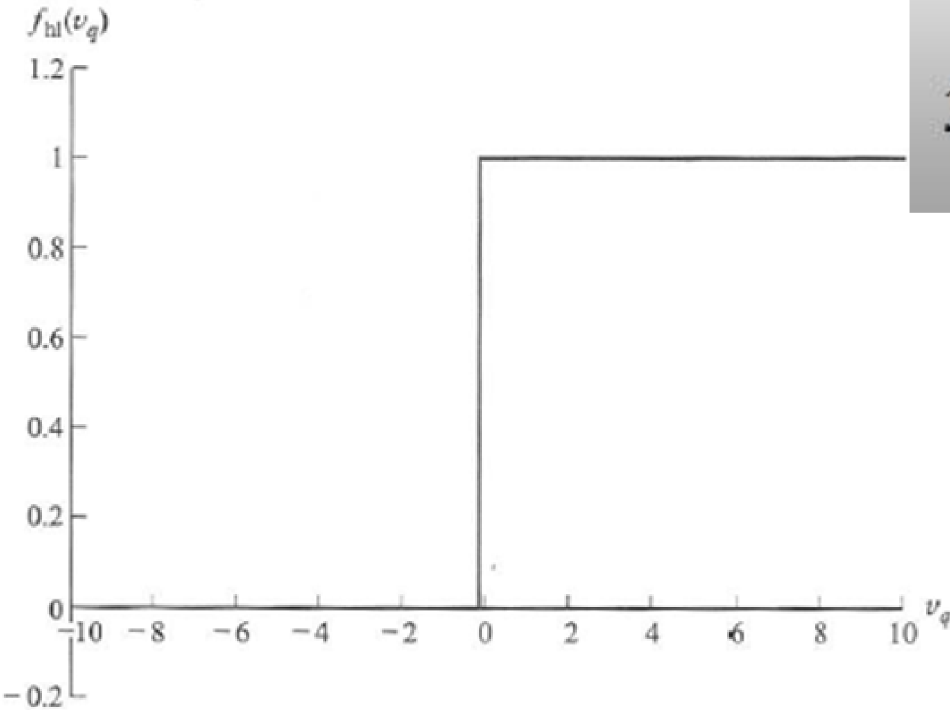
A. Linear (Identity) Function



$$y_q = f_{lin}(v_q) = v_q$$

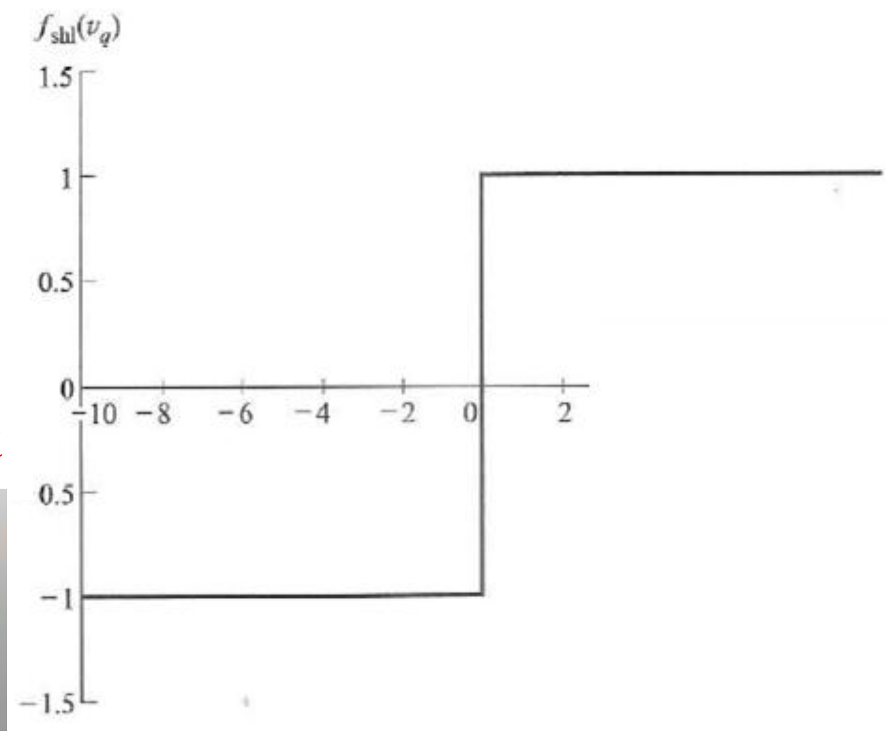
B1. Hard Limiter Function

$$y_q = f_{hl}(v_q) = \begin{cases} 0 \ if \ v_q < 0 \\ 1 \ if \ v_q \geq 0 \end{cases}$$



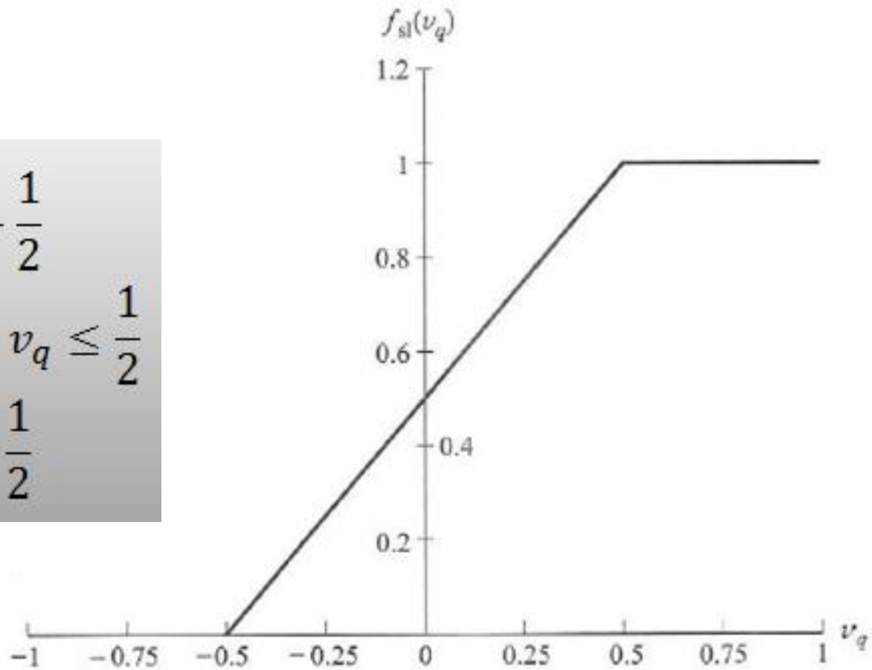
B2. Symmetric Hard Limiter Function

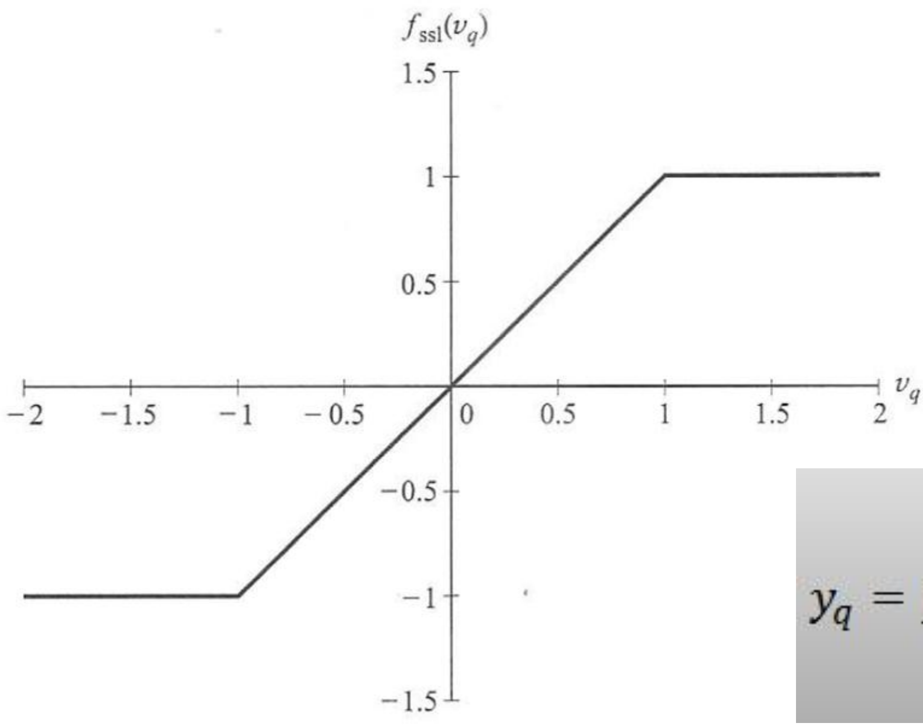
$$y_q = f_{shl}(v_q) = \begin{cases} -1 \ if \ v_q < 0 \\ 0 \ if \ v_q = 0 \\ 1 \ if \ v_q > 0 \end{cases}$$



C1. Piecewise Linear Function (Saturating Function)

$$y_q = f_{sl}(v_q) = \begin{cases} 0 \; if \; v_q < -\frac{1}{2} \\ v_q + \frac{1}{2} \; if \; -\frac{1}{2} \le v_q \le \frac{1}{2} \\ 1 \; if \; v_q > \frac{1}{2} \end{cases}$$



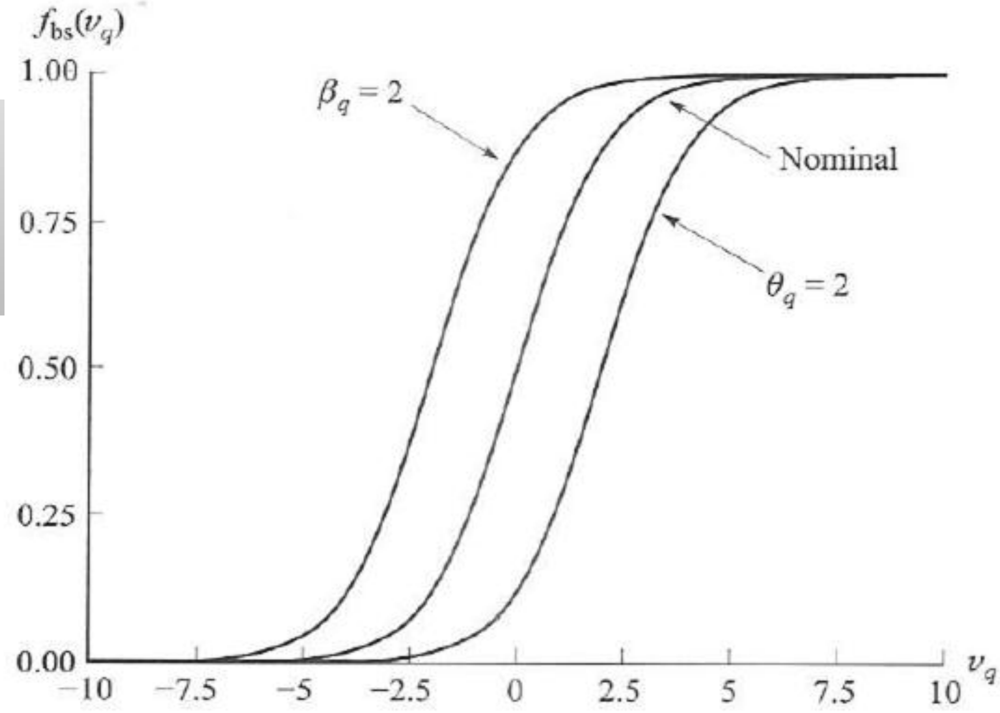


C2. Symmetric Piecewise Linear Function

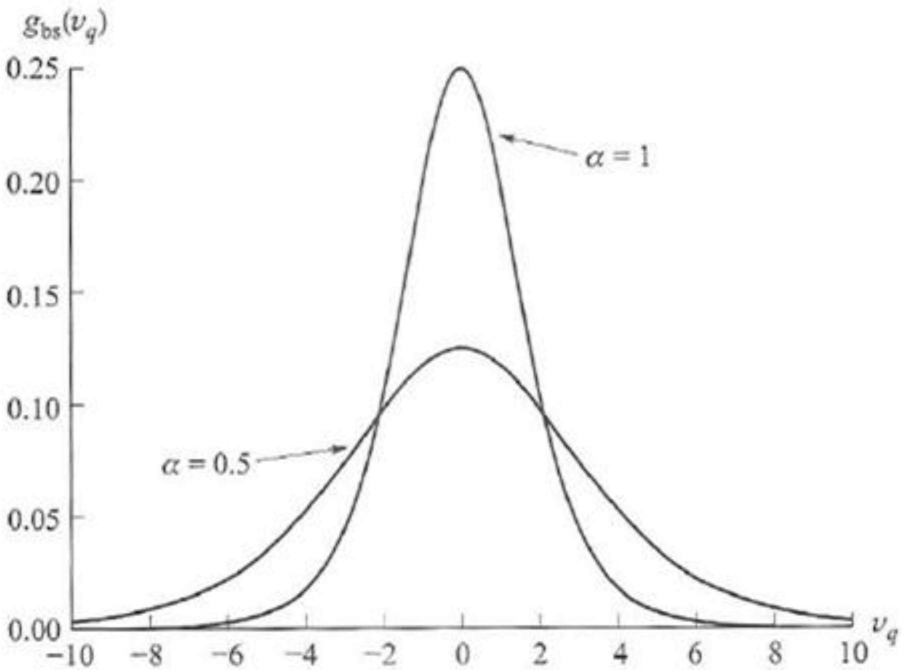
$$y_q = f_{ssl}(v_q) = \begin{cases} -1 \; if \; v_q < -1 \\ v_q \; if \; -1 \le v_q \le 1 \\ 1 \; if \; v_q > 1 \end{cases}$$

D1. Binary Sigmoid Function

$$y_q = f_{bs}(v_q) = \frac{1}{1 + e^{-\alpha v_q}}$$



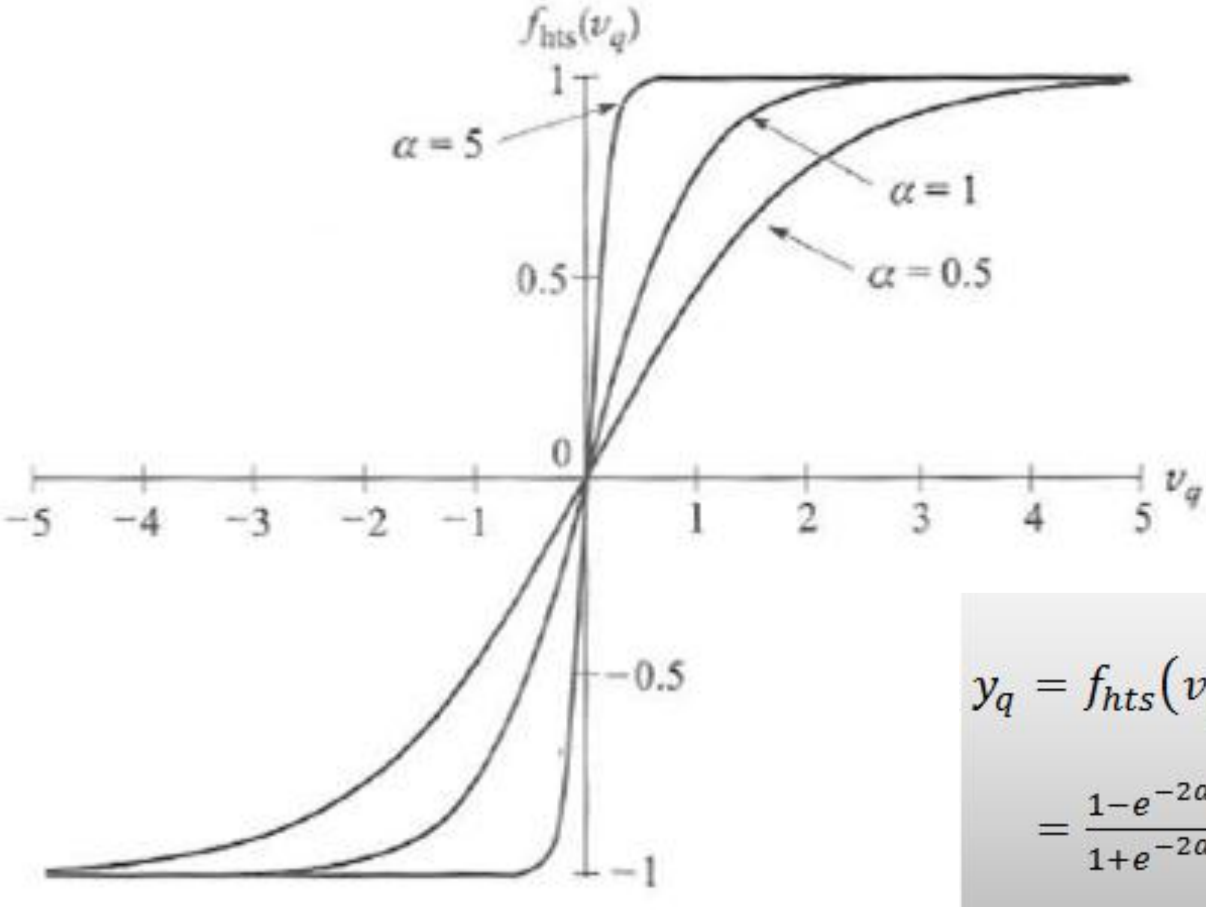
Derivative of binary sigmoid function



$$g_{bs}(v_q) = \frac{df_{bs}(v_q)}{dv_q} = \frac{\alpha e^{-\alpha v_q}}{(1+e^{-\alpha v_q})^2}$$

$$= \alpha f_{bs}(v_q)[1 - f_{bs}(v_q)]$$

D2. Hyperbolic Tangent Sigmoid (Binary Sigmoid) Function



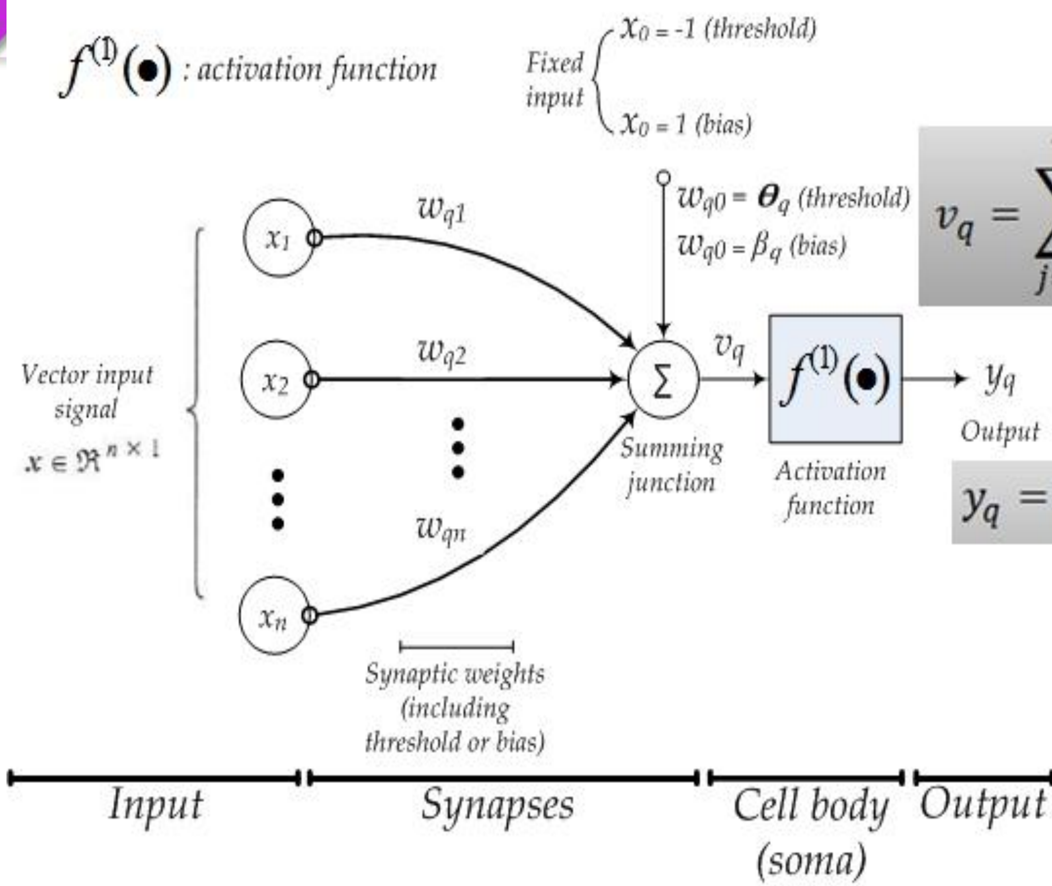
$$y_q = f_{hts}(v_q) = \tanh(\alpha v_q) = \frac{e^{\alpha v_q} - e^{-\alpha v_q}}{e^{\alpha v_q} + e^{-\alpha v_q}}$$

$$= \frac{1-e^{-2\alpha v_q}}{1+e^{-2\alpha v_q}}$$

Derivative of hyperbolic tangent sigmoid function

$$g_{hts}(v_q) = \frac{d f_{hts}(v_q)}{d v_q} = \alpha[1 + f_{hts}(v_q)][1 - f_{hts}(v_q)]$$

General ANNs Basic Models



$$v_q = \sum_{j=0}^{n} w_{qj} x_j$$

$$y_q = f(v_q)$$

Alternative model

- u_q is a linier combiner of input (x_j) and synaptic weight (w_{qj})

$$u_q = \sum_{j=1}^{n} w_{qj} x_j = w_q^T x = x^T w_q$$

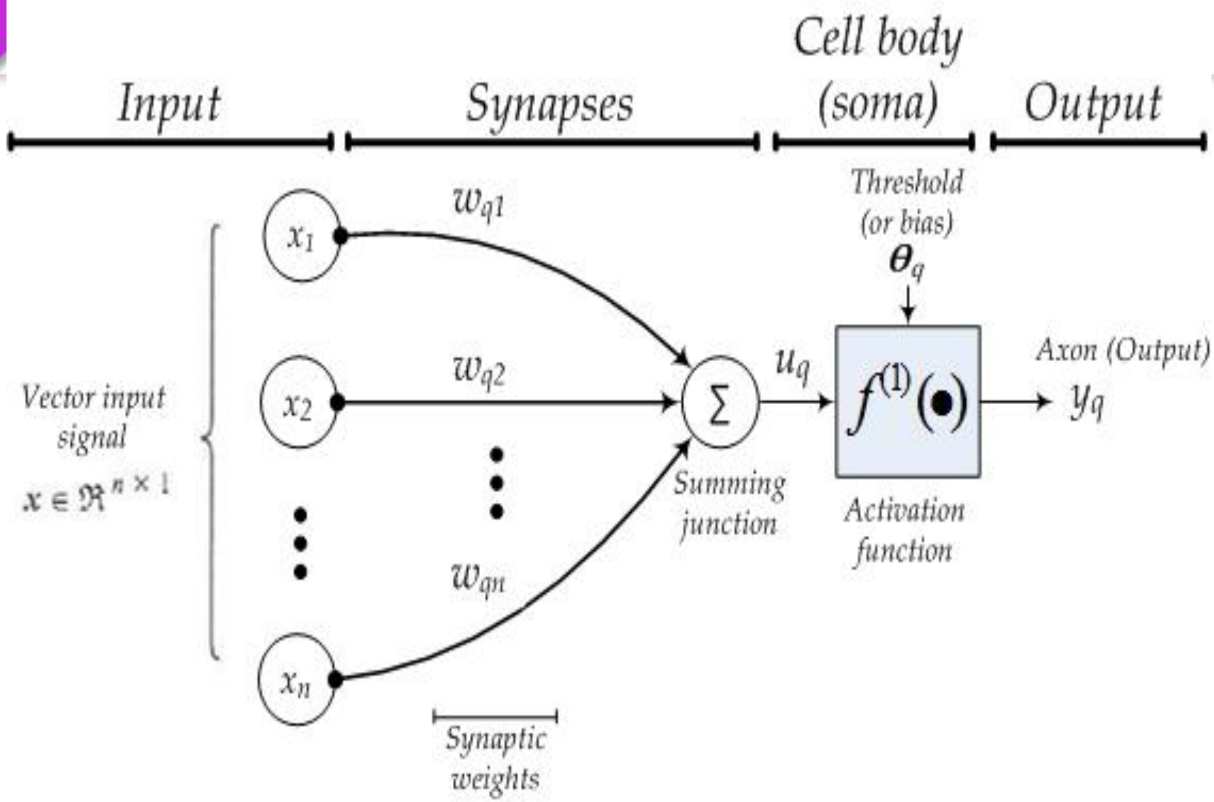
- Activation potensial

$$v_q = u_q - \theta_q$$

- Output of activation function:

$$y_q = f(v_q) = f\left(\sum_{j=1}^{n} w_{qj} x_j - \theta_q\right)$$

General ANNs Basic Models



$f^{(1)}(\bullet)$: activation function

Nonlinear model of an ANN

- u_q is a linier combiner of input (x_j) and synaptic weight (w_{qj})

$$u_q = \sum_{j=1}^{n} w_{qj} x_j = w_q^T x = x^T w_q$$

- Activation potensial

$$v_q = u_q - \theta_q$$

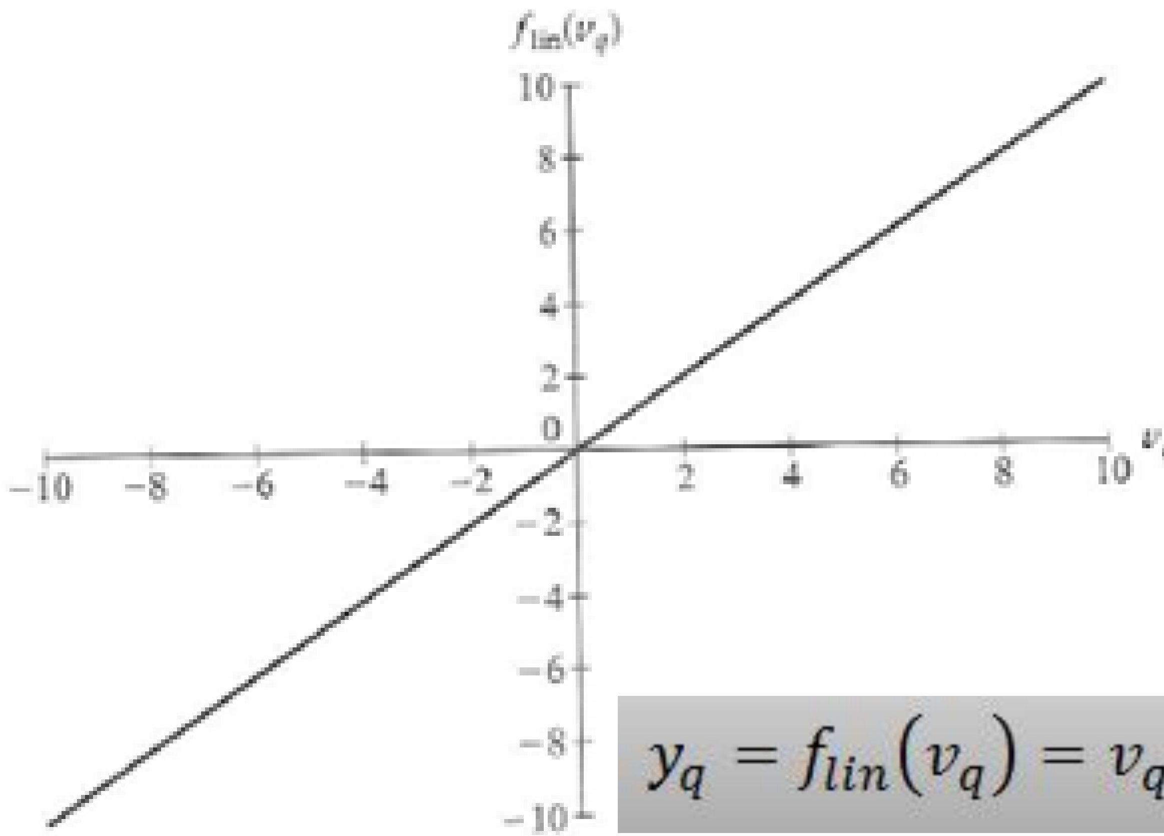
- Output of activation function:

$$y_q = f(v_q) = f(\sum_{j=1}^{n} w_{qj} x_j - \theta_q)$$

Activation Functions

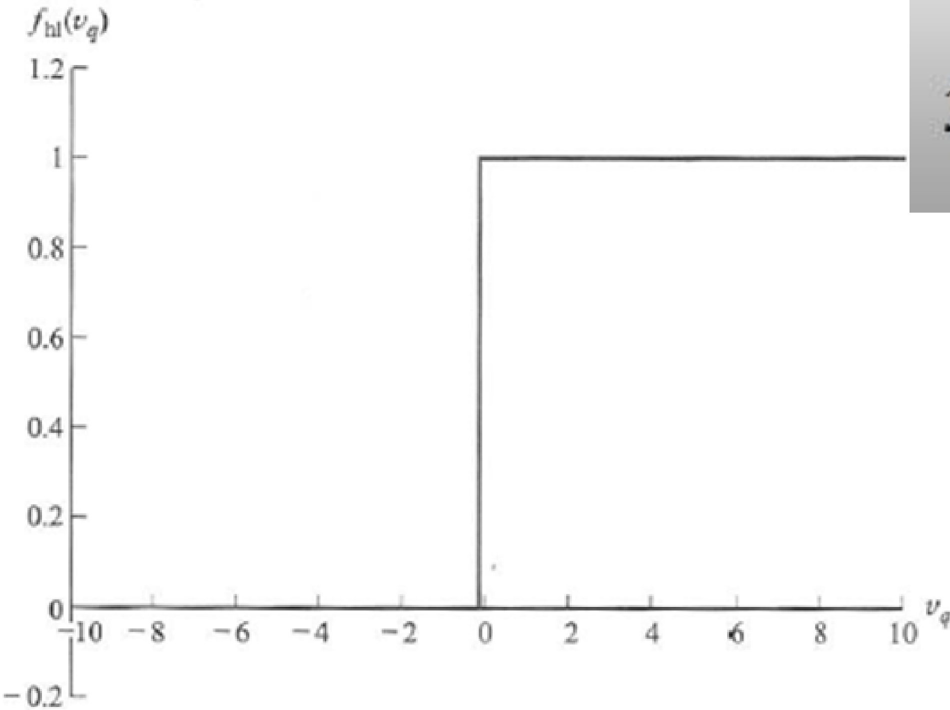
- Activation function can be a linear or nonlinear function.
- Selection of activation function depends on particular problem to be solved.

A. Linear (Identity) Function



$$y_q = f_{lin}(v_q) = v_q$$

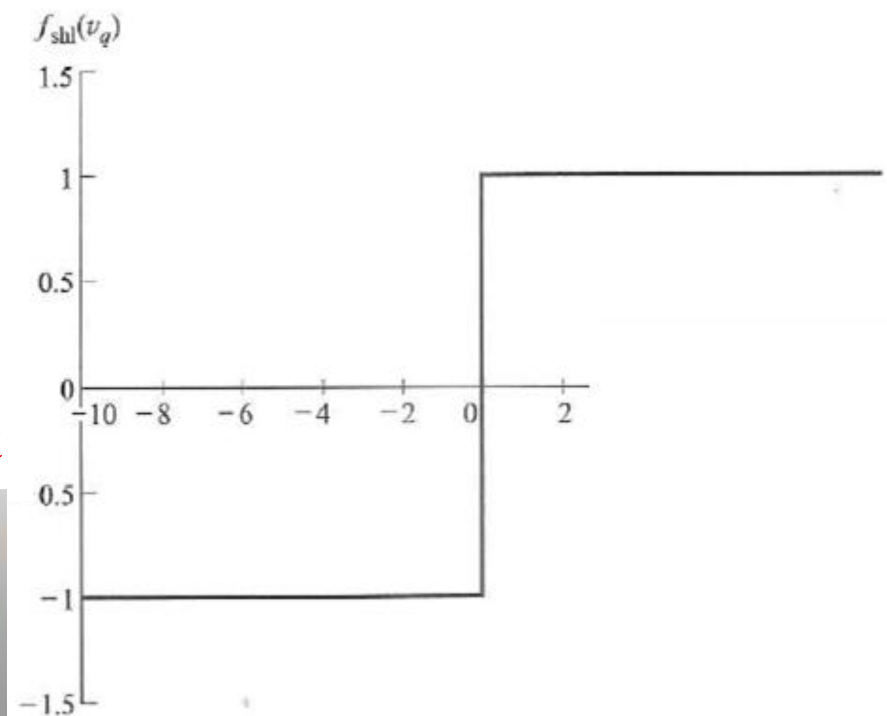
B1. Hard Limiter Function



$$y_q = f_{hl}(v_q) = \begin{cases} 0 \ if \ v_q < 0 \\ 1 \ if \ v_q \geq 0 \end{cases}$$

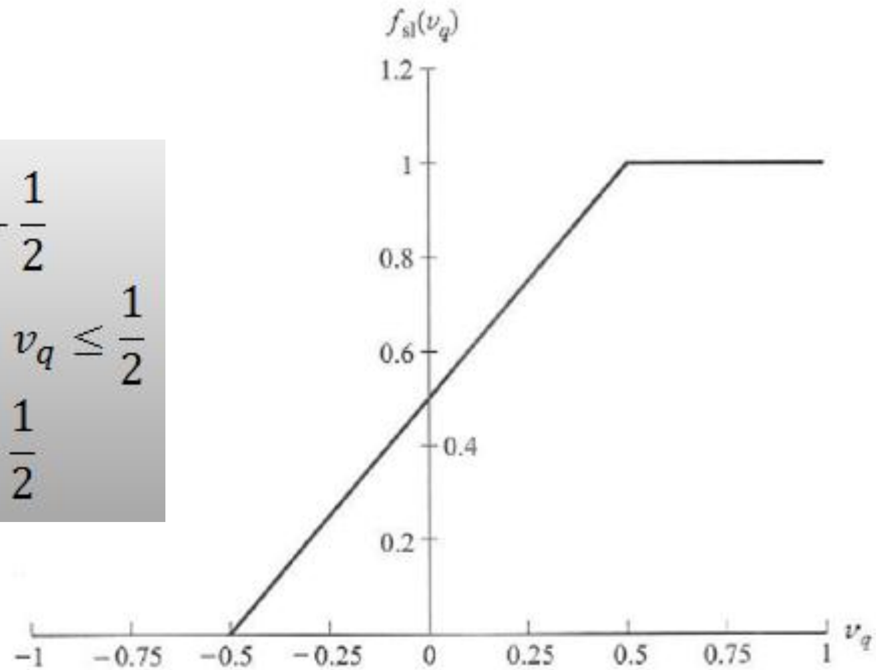
B2. Symmetric Hard Limiter Function

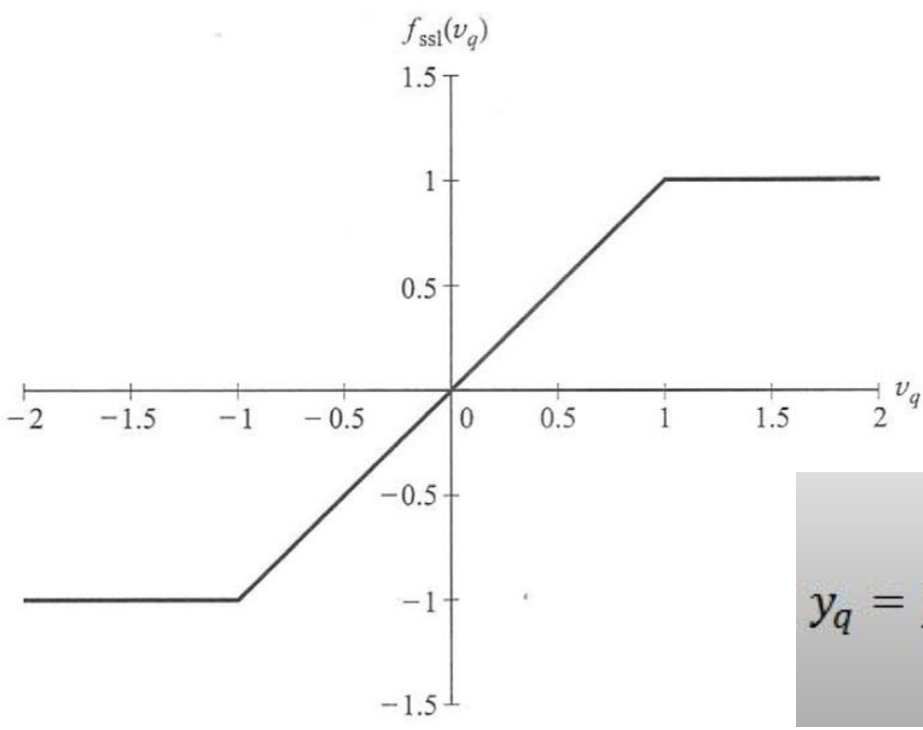
$$y_q = f_{shl}(v_q) = \begin{cases} -1 \ if \ v_q < 0 \\ 0 \ if \ v_q = 0 \\ 1 \ if \ v_q > 0 \end{cases}$$



C1. Piecewise Linear Function (Saturating Function)

$$y_q = f_{sl}(v_q) = \begin{cases} 0 \ if \ v_q < -\frac{1}{2} \\ v_q + \frac{1}{2} \ if \ -\frac{1}{2} \le v_q \le \frac{1}{2} \\ 1 \ if \ v_q > \frac{1}{2} \end{cases}$$



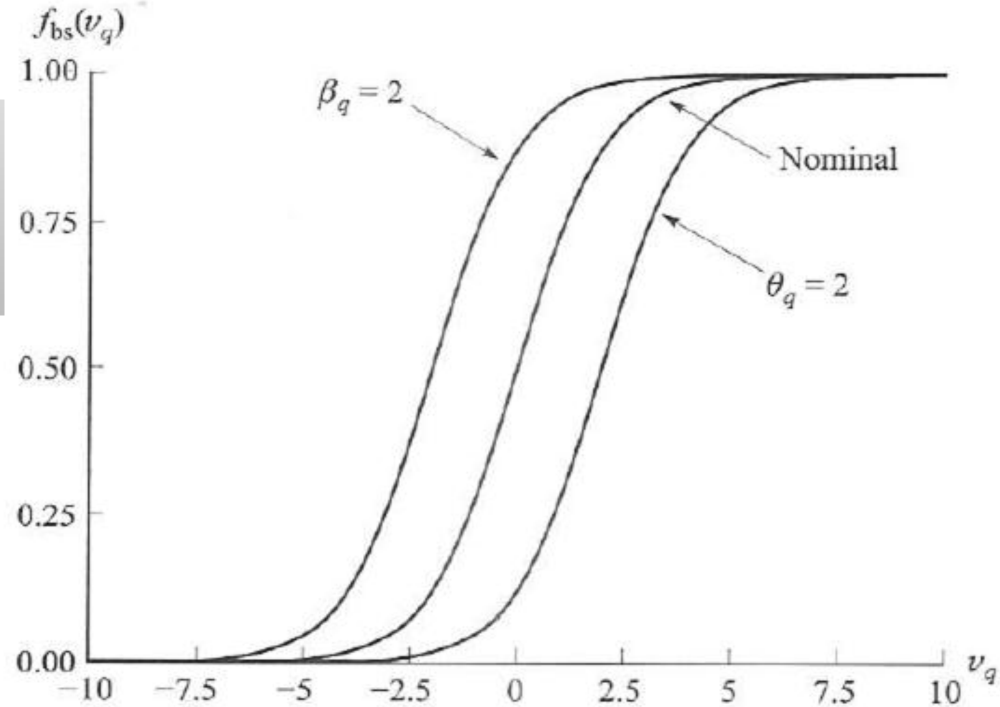


C2. Symmetric Piecewise Linear Function

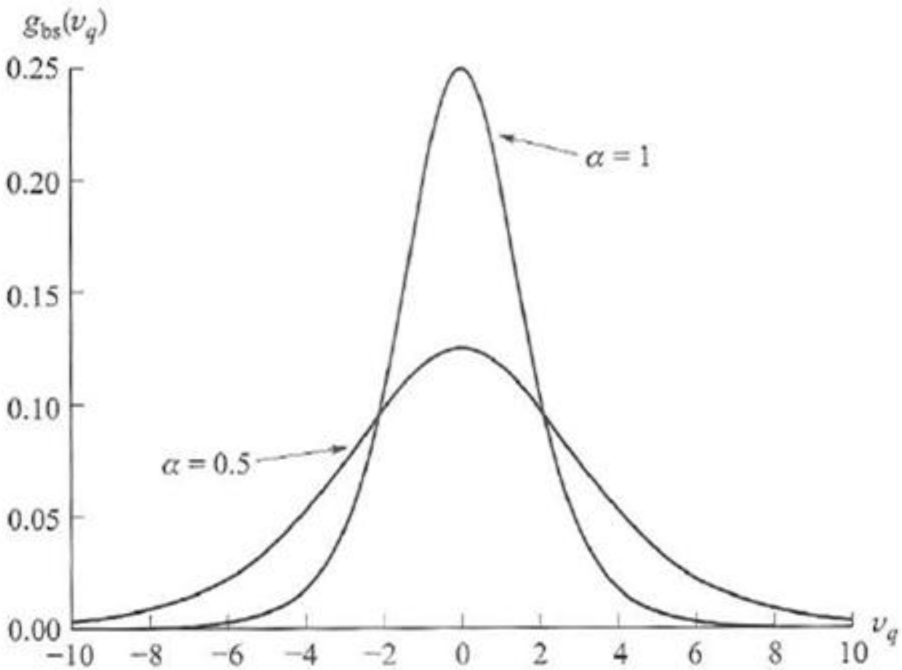
$$y_q = f_{ssl}(v_q) = \begin{cases} -1 \ if \ v_q < -1 \\ v_q \ if \ -1 \le v_q \le 1 \\ 1 \ if \ v_q > 1 \end{cases}$$

D1. Binary Sigmoid Function

$$y_q = f_{bs}(v_q) = \frac{1}{1 + e^{-\alpha v_q}}$$



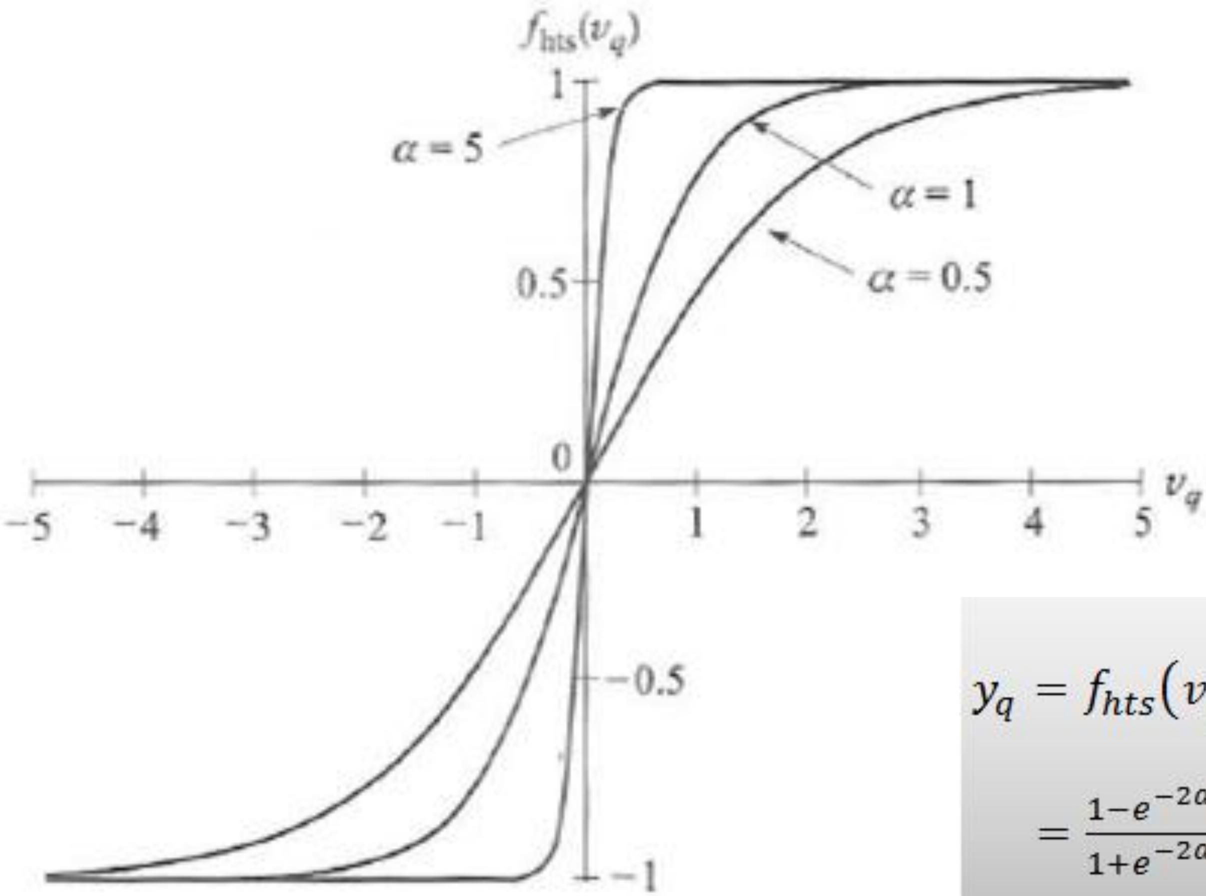
Derivative of binary sigmoid function



$$g_{bs}(v_q) = \frac{df_{bs}(v_q)}{dv_q} = \frac{\alpha e^{-\alpha v_q}}{(1+e^{-\alpha v_q})^2}$$

$$= \alpha f_{bs}(v_q)[1 - f_{bs}(v_q)]$$

D2. Hyperbolic Tangent Sigmoid (Binary Sigmoid) Function



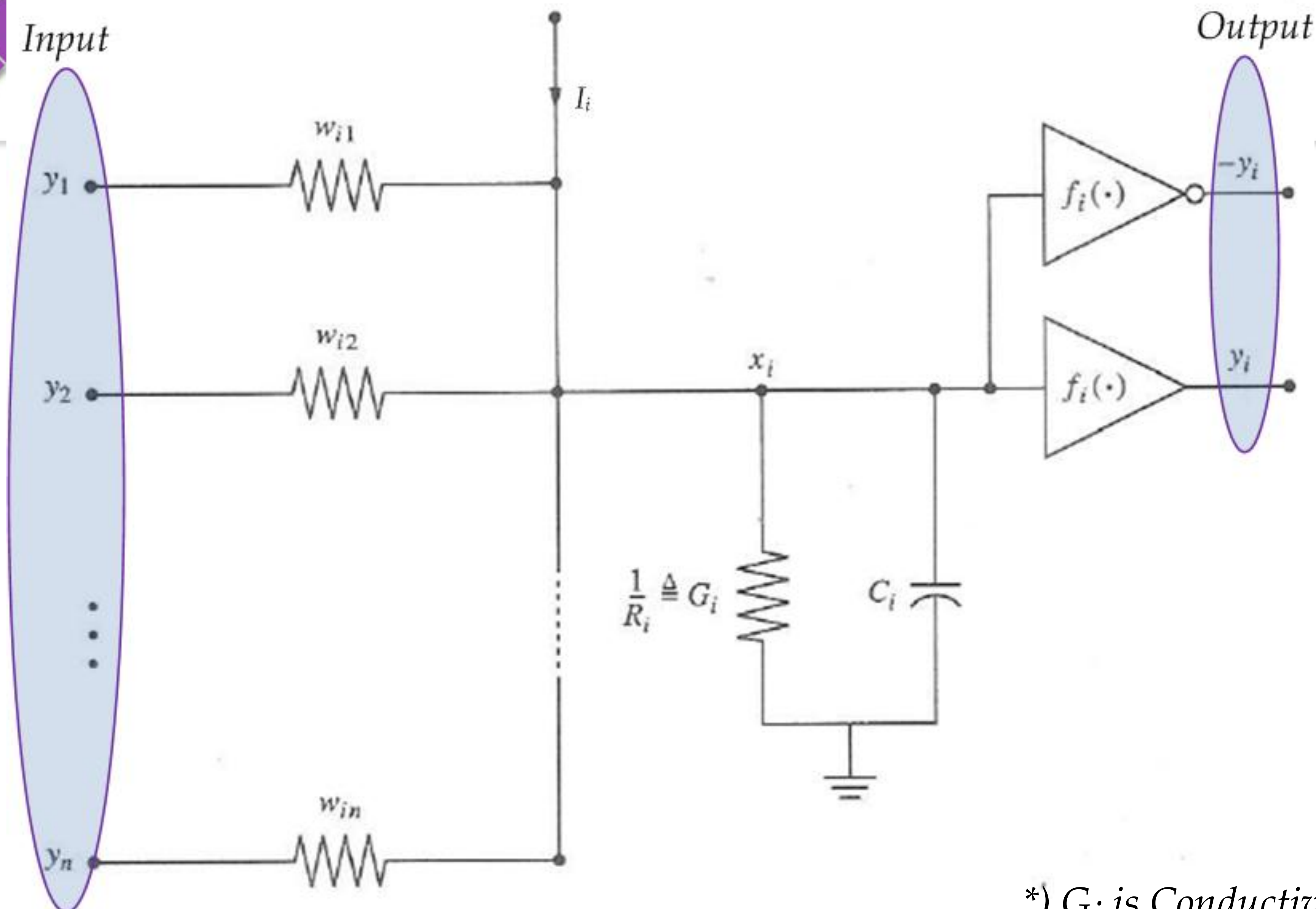
$$y_q = f_{hts}(v_q) = \tanh(\alpha v_q) = \frac{e^{\alpha v_q} - e^{-\alpha v_q}}{e^{\alpha v_q} + e^{-\alpha v_q}}$$

$$= \frac{1-e^{-2\alpha v_q}}{1+e^{-2\alpha v_q}}$$

Derivative of hyperbolic tangent sigmoid function

$$g_{hts}(v_q) = \frac{df_{hts}(v_q)}{dv_q} = \alpha[1 + f_{hts}(v_q)][1 - f_{hts}(v_q)]$$

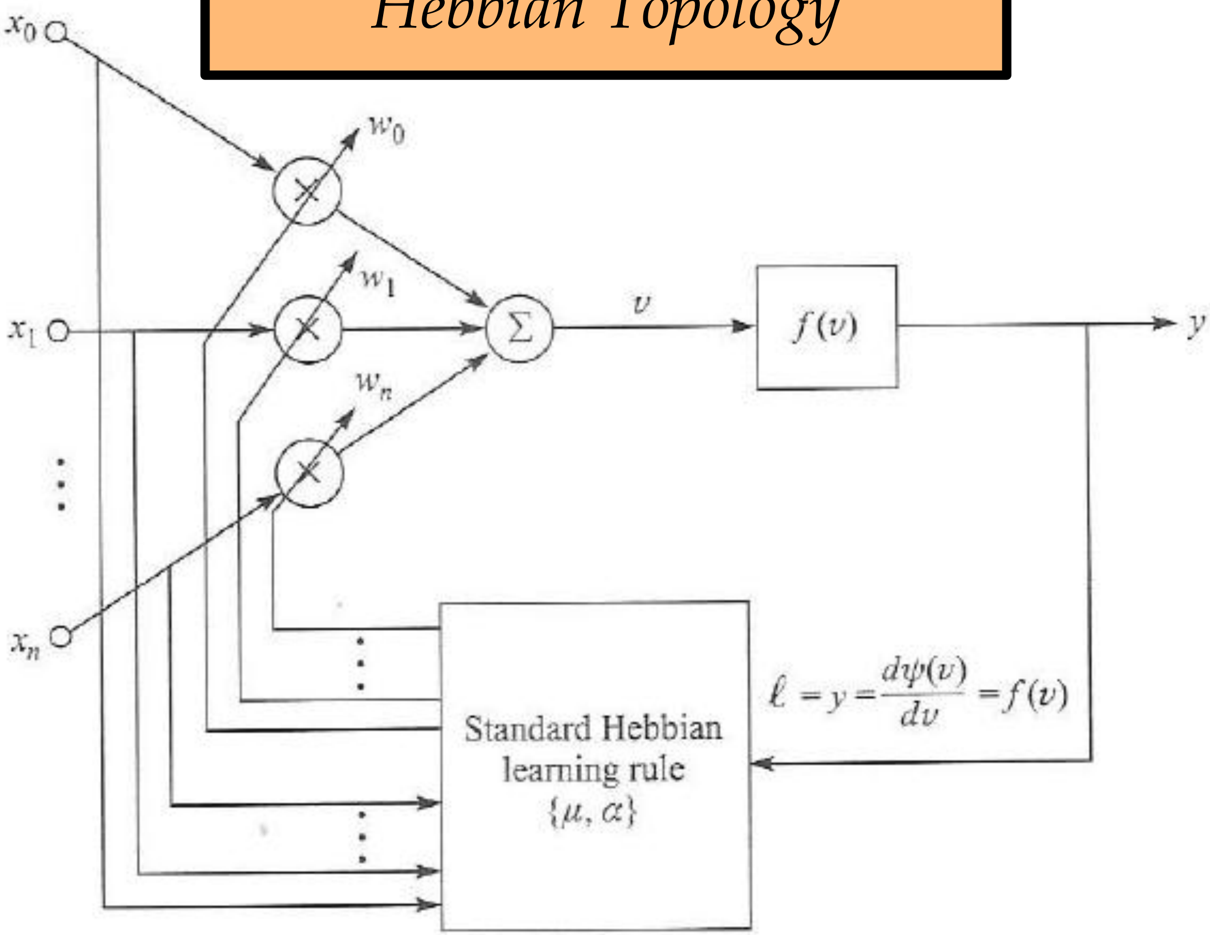
Electrical circuit model of neuron



*) G_i is Conductivity

Hebbian Learning (Hebb)

Hebbian Topology



- Energy function

$$\varepsilon(\omega) = -\Psi(w^T x) + \frac{\alpha}{2}\|w\|_2^2$$

$\Psi(\bullet)$ differentiable function

$\alpha > 0$ is the forgetting factor

- Neuron output:

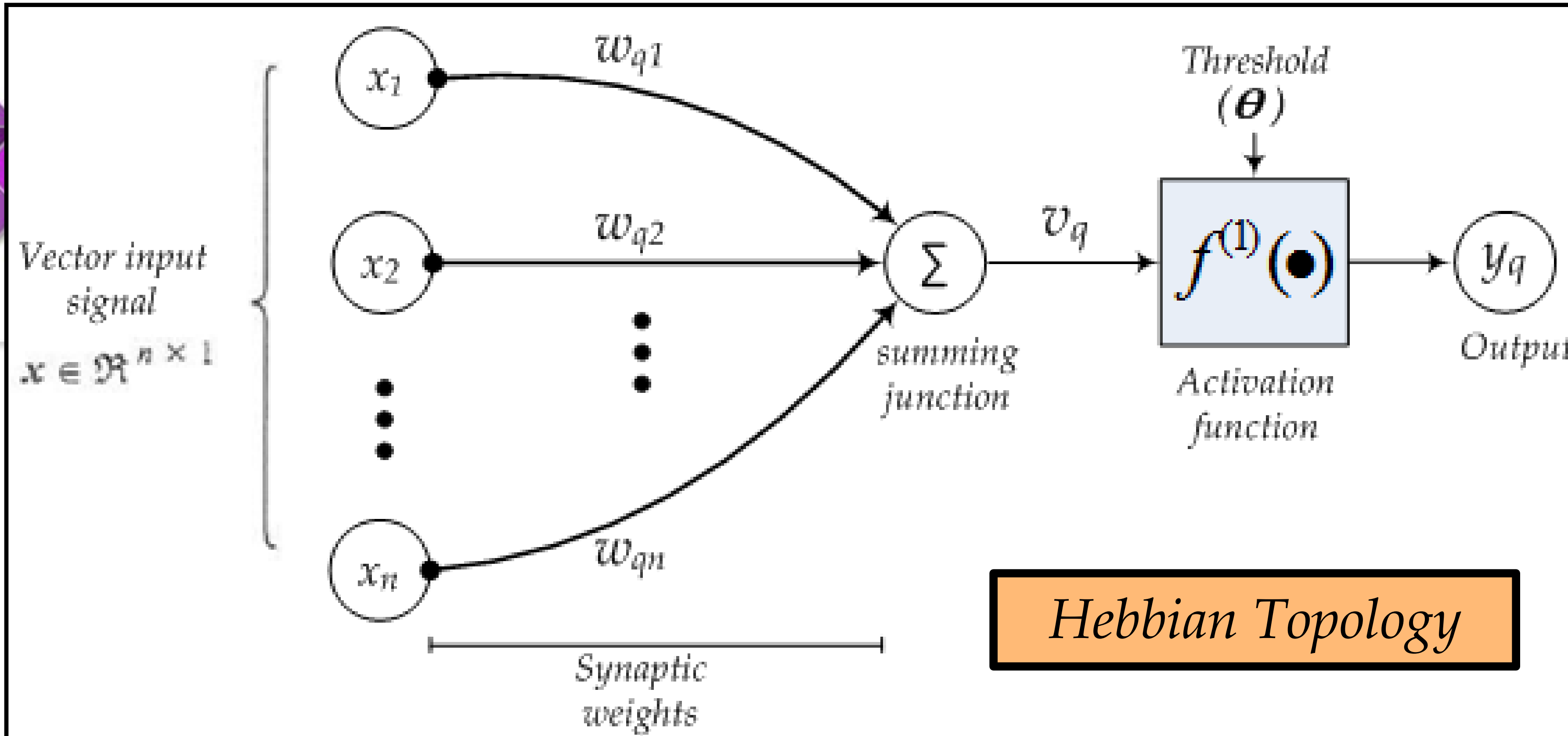
$$y = \frac{d\Psi(v)}{dv} = f(v); v = w^T x$$

- Steepest descent method:

$$\frac{dw}{dt} = -\mu \nabla_w \varepsilon(\omega)$$

μ is learning rate

$$\nabla_w \varepsilon(\omega) = -f(v)\frac{\partial v}{\partial w}\alpha w = -yx + \alpha w$$



Learning Algorithm:

1)Topologi/Arsitektur

*2)Target (**t**)*

3)Initialized weight ($\boldsymbol{w}_{qi}$)

4)net = $\boldsymbol{v}_q$

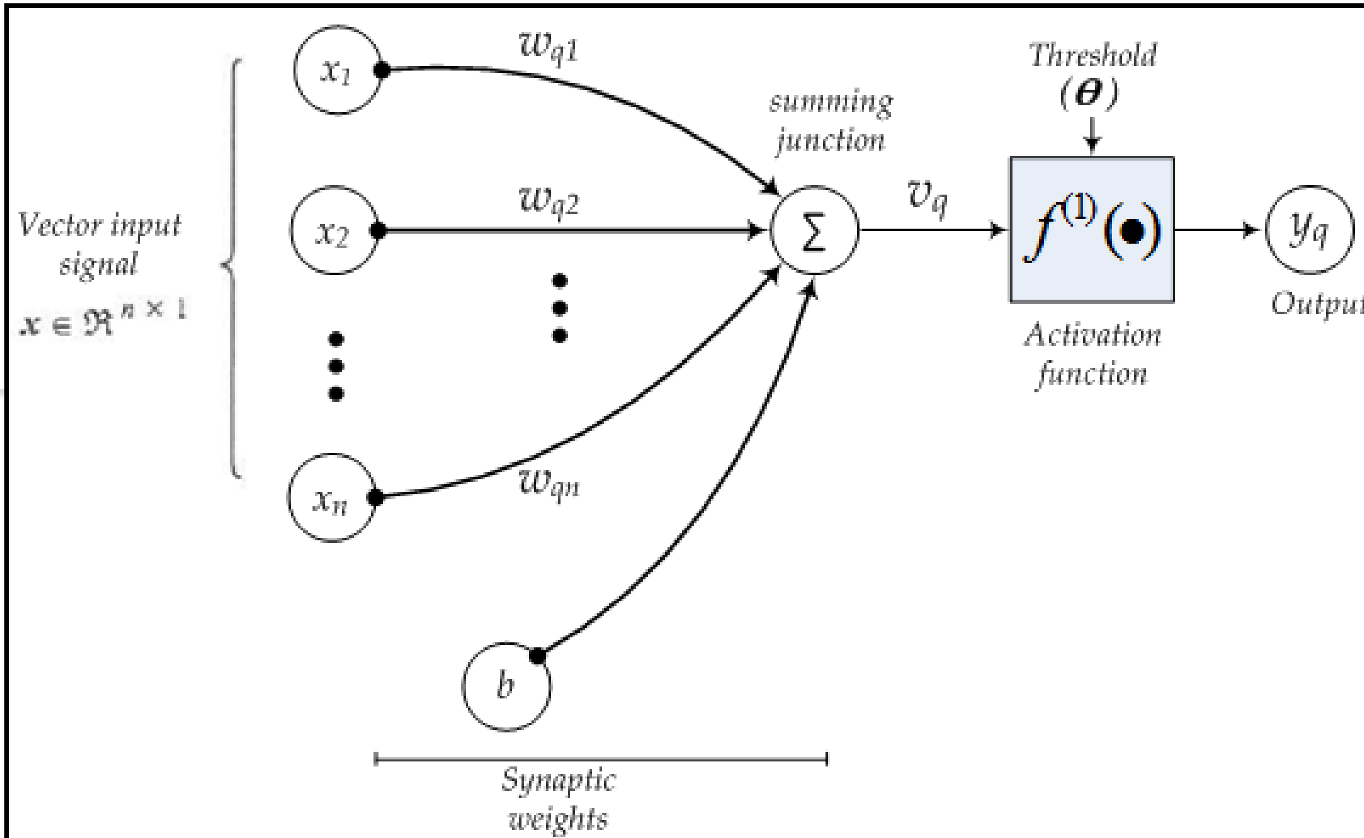
$$net = v_q = \sum_{i=1}^{n} w_{qi} x_i$$

5) Threshold ($\boldsymbol{\theta}$)

6) Output ($\boldsymbol{y}_q$)

$$y_q = f(net) = f(v_q) = \begin{cases} 1; & v_q \geq \theta \\ 0; & v_q < \theta \end{cases}$$

*7) error(**e**) = target(**t**) – output($\boldsymbol{y}_q$)*



Learning Algorithm (with bias):

1)*Topologi/Arsitektur*

2)*Target* (***t***)

3)*Initialized weight* ($\boldsymbol{w_{qi}}$)

4)*net* = $\boldsymbol{v_q}$

$$net = v_q = b + \sum_{i=1}^{n} w_{qi} x_i$$

5) *Threshold* ($\boldsymbol{\theta = 0}$)

6) *Bias* (***b***)

7) *Output* ($\boldsymbol{y_q}$)

$$y_q = f(net) = f(v_q) = \begin{cases} 1; & v_q \geq \theta \\ -1; & v_q < \theta \end{cases}$$

8) *error*(***e***) = *target*(***t***) – *output*($\boldsymbol{y_q}$)