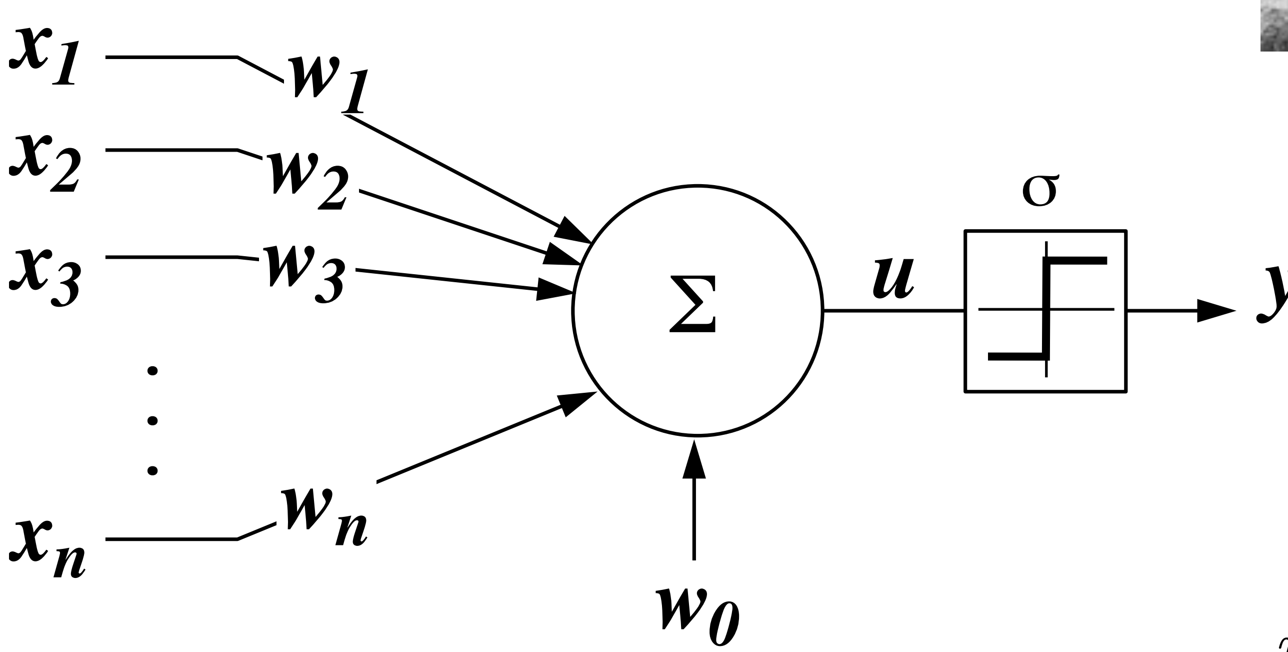
Supervised learning

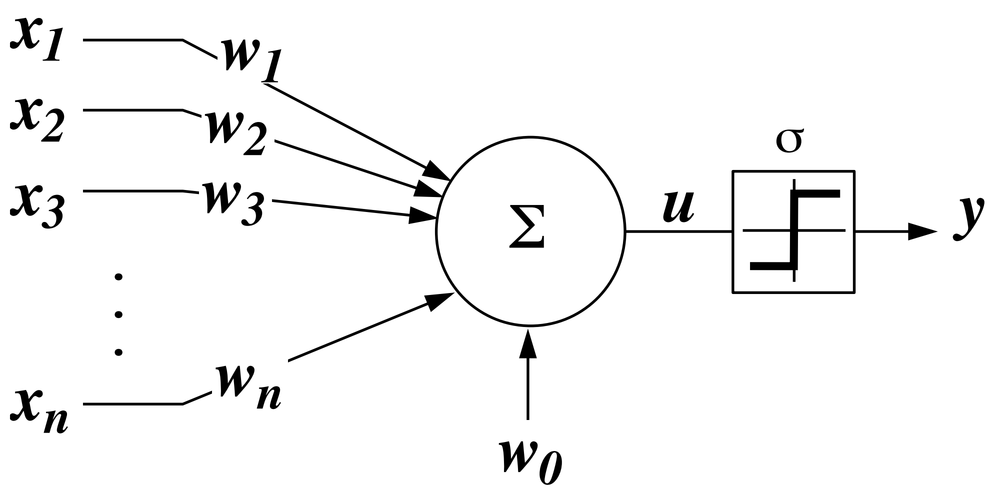
Perceptron model
(Rosenblatt, ca. 1960)



$$u = w_0 + \sum_{i=1}^{n} w_i \, x_i$$

$$y = \sigma(u)$$

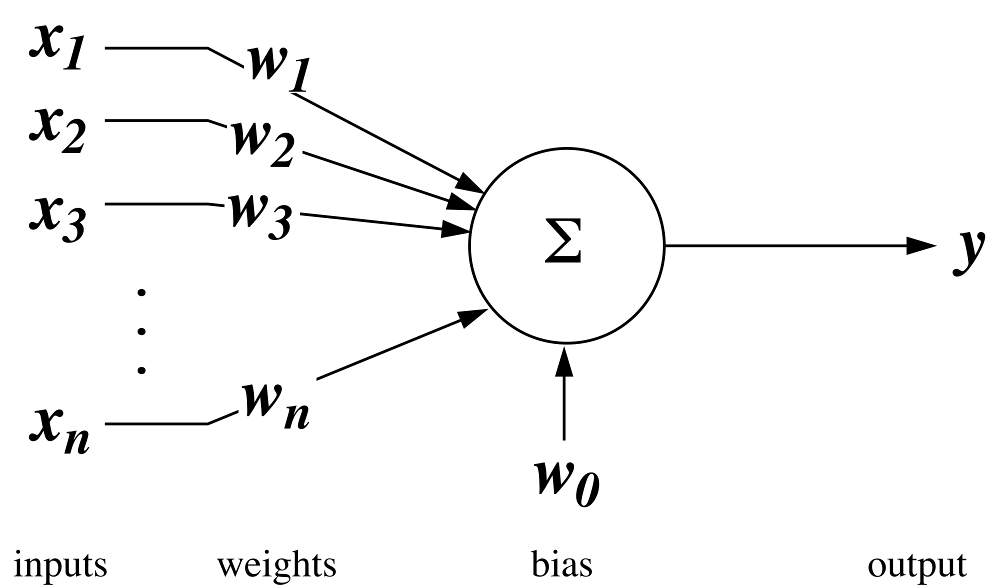
Perceptron learning rule
(Rosenblatt 1962)



$$\Delta w_k = \begin{cases} 2\eta \, T^{(\alpha)} \, x_k^{(\alpha)} & y^{(\alpha)} \neq T^{(\alpha)} \\ 0 & \text{otherwise} \end{cases}$$

$$= \eta \left(T^{(\alpha)} - y^{(\alpha)} \right) x_k$$

Linear neuron learning rule
(Widrow & Hoff 1960)

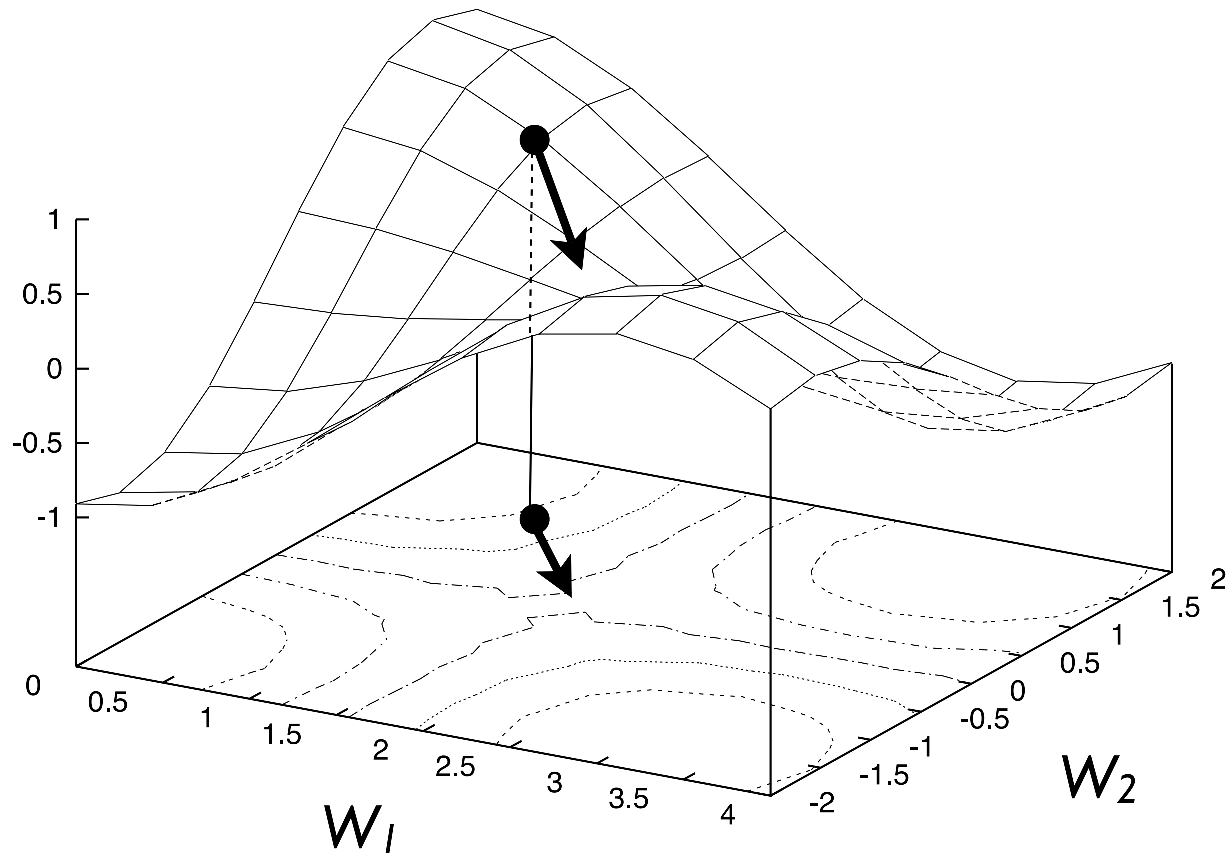


Objective function ⟶ Learning rule

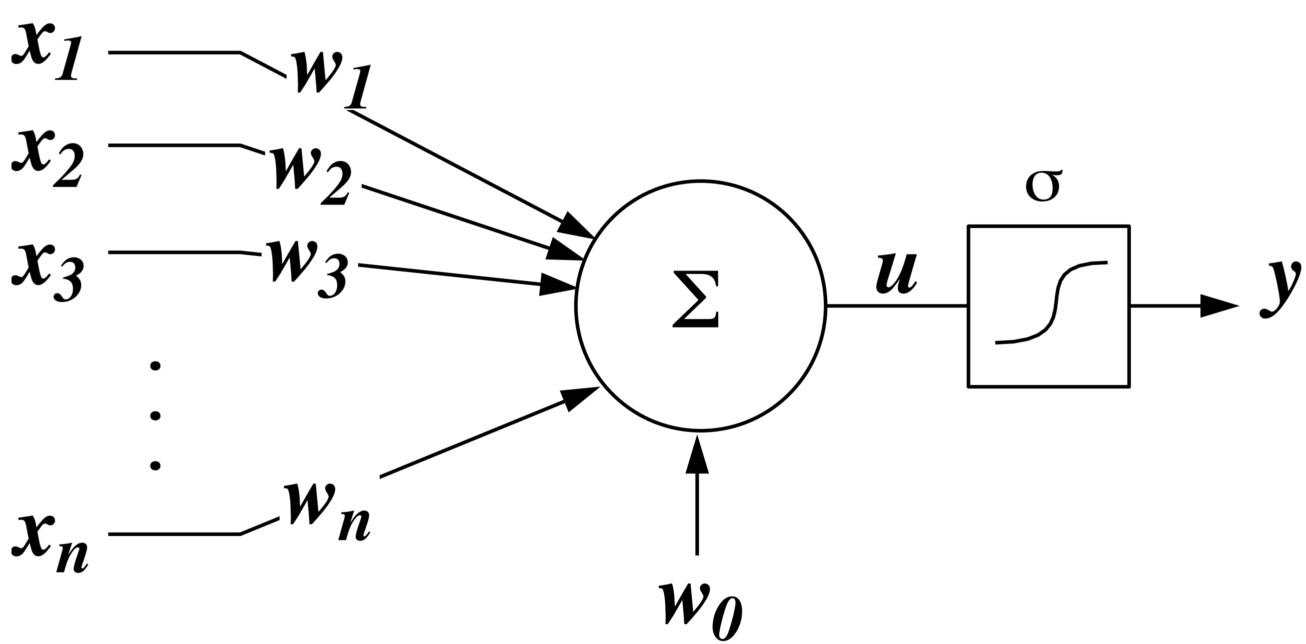
$$E = \frac{1}{2}\sum_{\alpha}\left[T^{(\alpha)} - y^{(\alpha)}\right]^2$$

$$\begin{aligned}
\Delta w_k &= -\eta\,\frac{\partial E}{\partial w_k} \\
&= \eta \sum_{\alpha} \delta^{(\alpha)}\, x_k^{(\alpha)} \\
\delta^{(\alpha)} &= T^{(\alpha)} - y^{(\alpha)}
\end{aligned}$$

Gradient descent in weight space

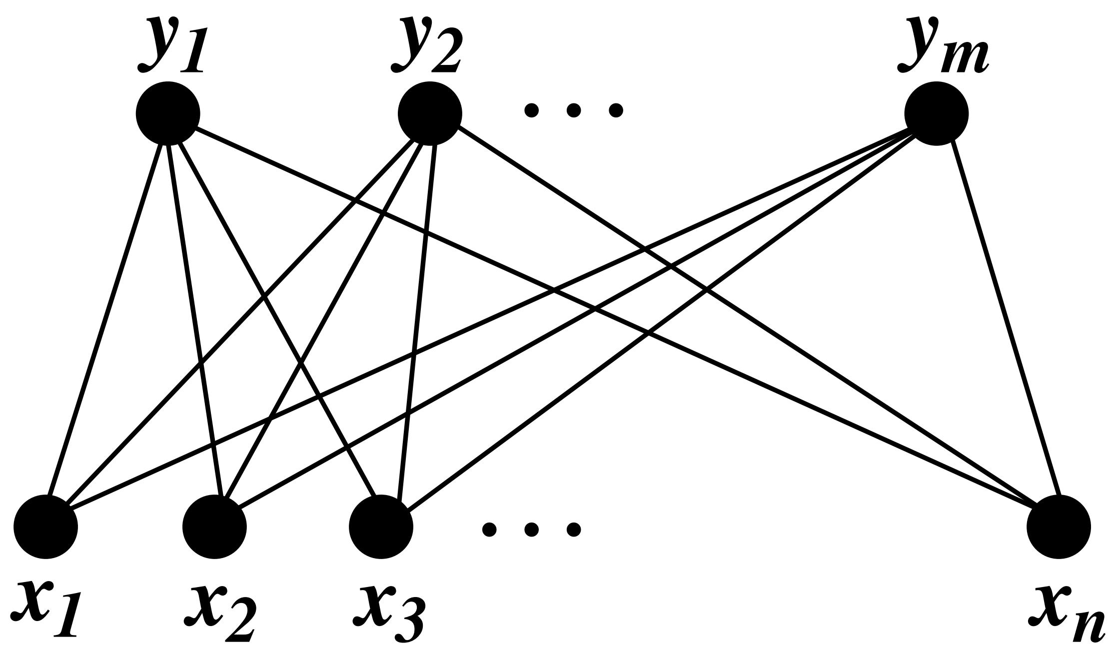


Linear neuron with output non-linearity



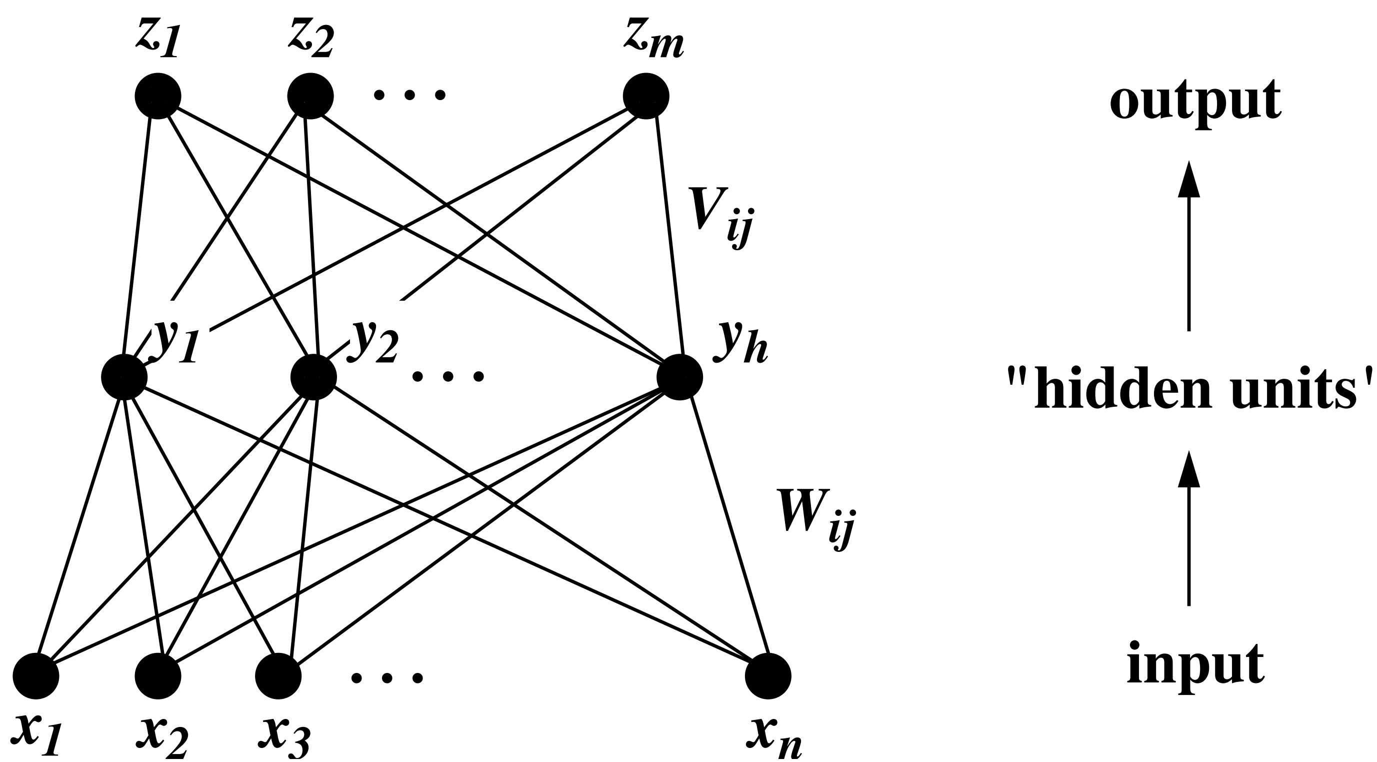
$$y = \sigma(u) \equiv \frac{1}{1 + e^{-\beta\, u}}$$

Single-layer network



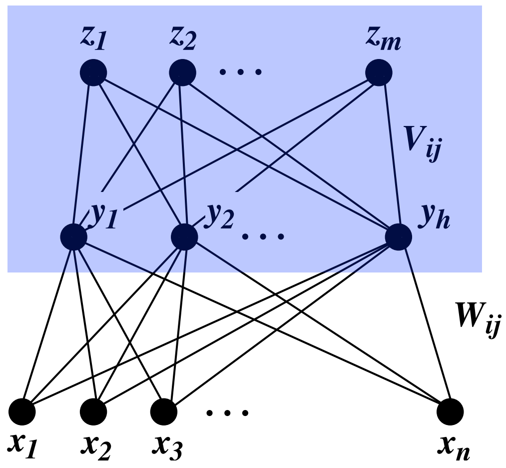
$$y_i = \sigma(\sum_j W_{ij}\, x_j)$$

Two-layer network



$$
\begin{aligned}
z_i &= \sigma(\sum_j V_{ij} y_j) \\
y_i &= \sigma(\sum_j W_{ij} x_j)
\end{aligned}
$$

Learning rule for output layer



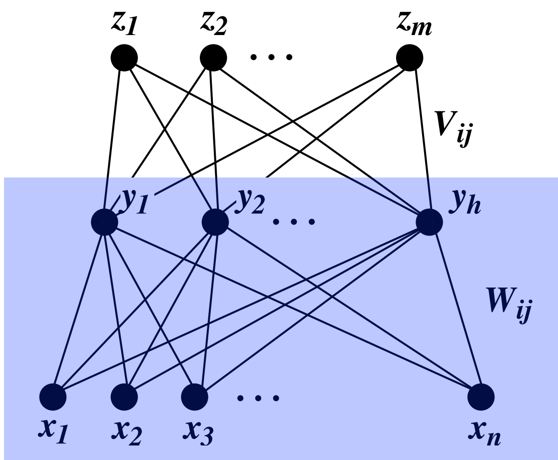
$$E^{(\alpha)} = \frac{1}{2} \sum_i \left[T_i^{(\alpha)} - z_i(\mathbf{x}^{(\alpha)}) \right]^2$$

$$\begin{aligned} \Delta V_{ij} &= -\eta \frac{\partial E}{\partial V_{ij}} \\ &= [T_i - z_i(\mathbf{x})] \frac{\partial z_i(\mathbf{x})}{\partial V_{ij}} \\ &= [T_i - z_i(\mathbf{x})]\, \sigma'(u_{z_i})\, y_j \\ &= \boxed{\delta_{z_i}\, y_j} \end{aligned}$$

where

$$\begin{aligned} \delta_{z_i} &= [T_i - z_i(\mathbf{x})]\, \sigma'(u_{z_i}) \\ u_{z_i} &= \sum_j V_{ij}\, y_j \end{aligned}$$

Learning rule for hidden layer



$$\Delta W_{kl} = -\eta \frac{\partial E}{\partial W_{kl}} = \eta \sum_i [T_i - z_i(\mathbf{x})] \frac{\partial z_i(\mathbf{x})}{\partial W_{kl}} .$$

$$\frac{\partial z_i(\mathbf{x})}{\partial W_{kl}} = \frac{\partial z_i(\mathbf{x})}{\partial y_k} \frac{\partial y_k}{\partial W_{kl}}$$

$$\Delta W_{kl} = \eta \sum_i [T_i - z_i(\mathbf{x})] \, \sigma'(u_{z_i}) V_{ik} \, \sigma'(u_{y_k}) \, x_l = \boxed{\eta \, \delta_{y_k} \, x_l}$$

where $\delta_{y_k} = \sigma'(u_{y_k}) \sum_i \delta_{z_i} V_{ik}$ ← back-propagation of error

Second-order methods

$$E(\mathbf{w}_0 + \Delta\mathbf{w}) \approx E(\mathbf{w}_0) + \Delta\mathbf{w}^T \, \nabla E + \frac{1}{2}\Delta\mathbf{w}^T \, \mathbf{H} \, \Delta\mathbf{w}$$

gradient Hessian

This approximation will be minimized when

$$\nabla E + \mathbf{H} \, \Delta\mathbf{w} = 0$$

Thus

$$\Delta\mathbf{w}^* = -\mathbf{H}^{-1} \, \nabla E$$

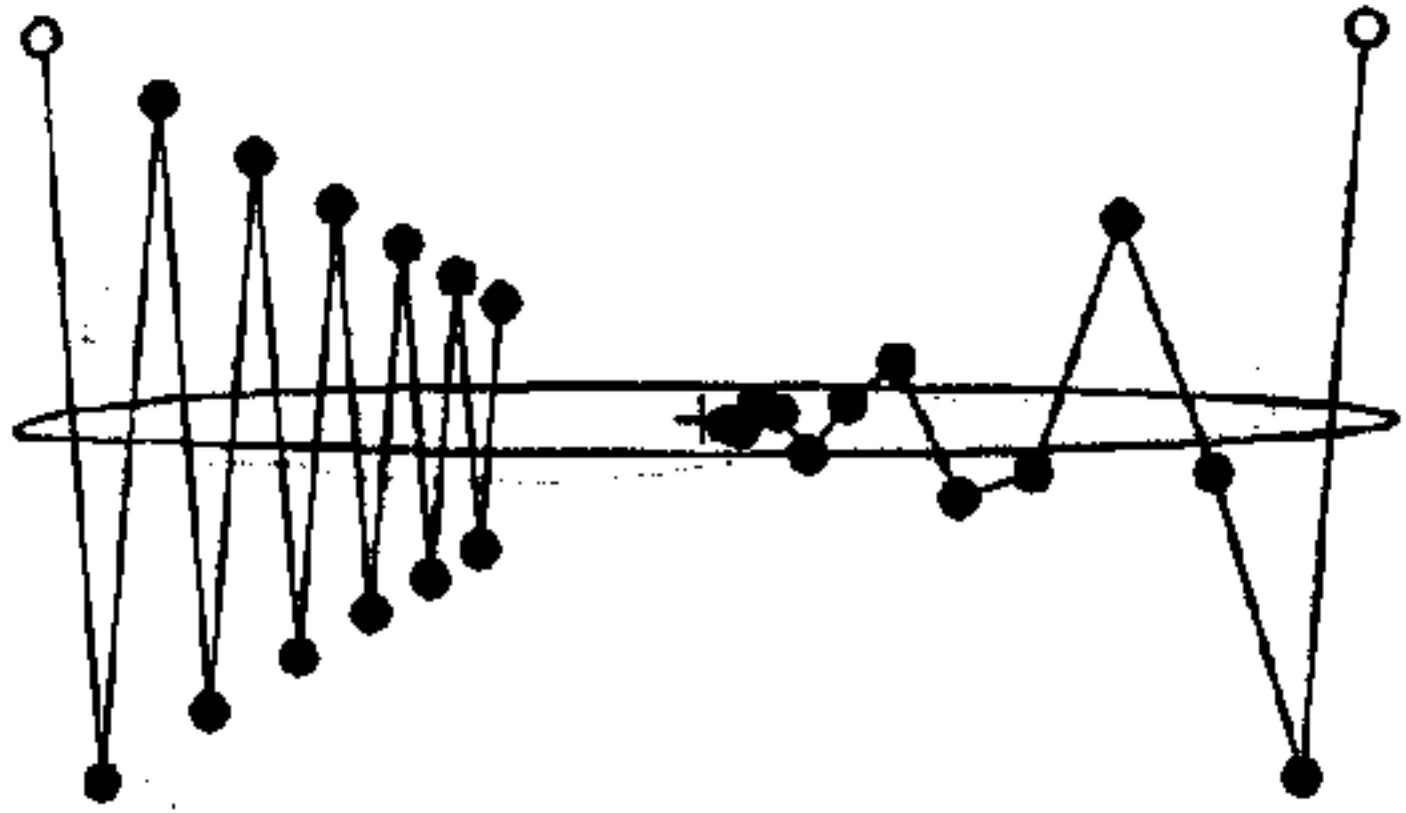
Momentum

$$\Delta w_{kl}(t+1) = -\eta \frac{\partial E}{\partial w_{kl}} + \alpha \, \Delta w_{kl}(t)$$

Converges to

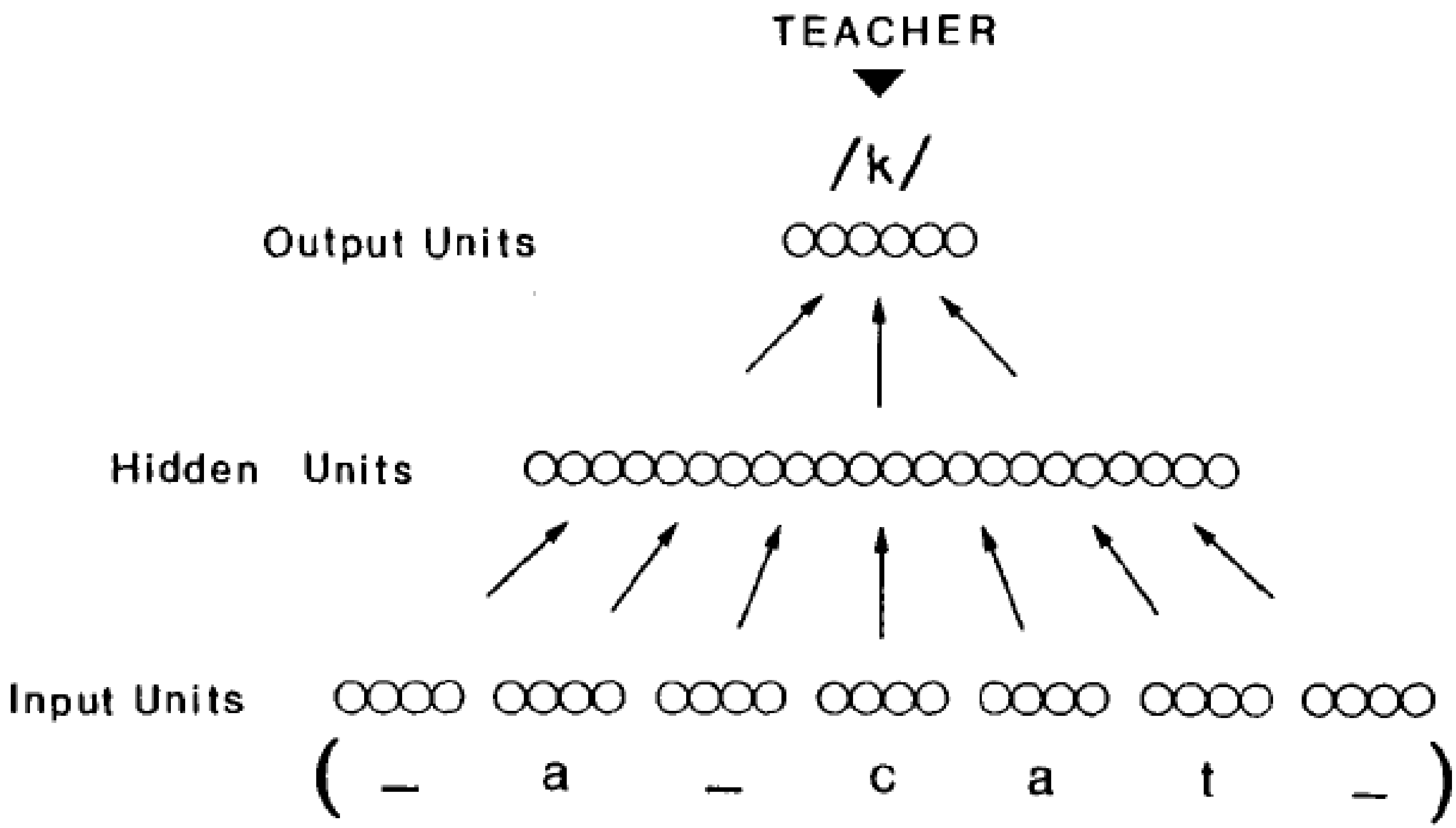
$$\Delta w_{kl} \approx -\frac{\eta}{1-\alpha} \frac{\partial E}{\partial w_{kl}}$$

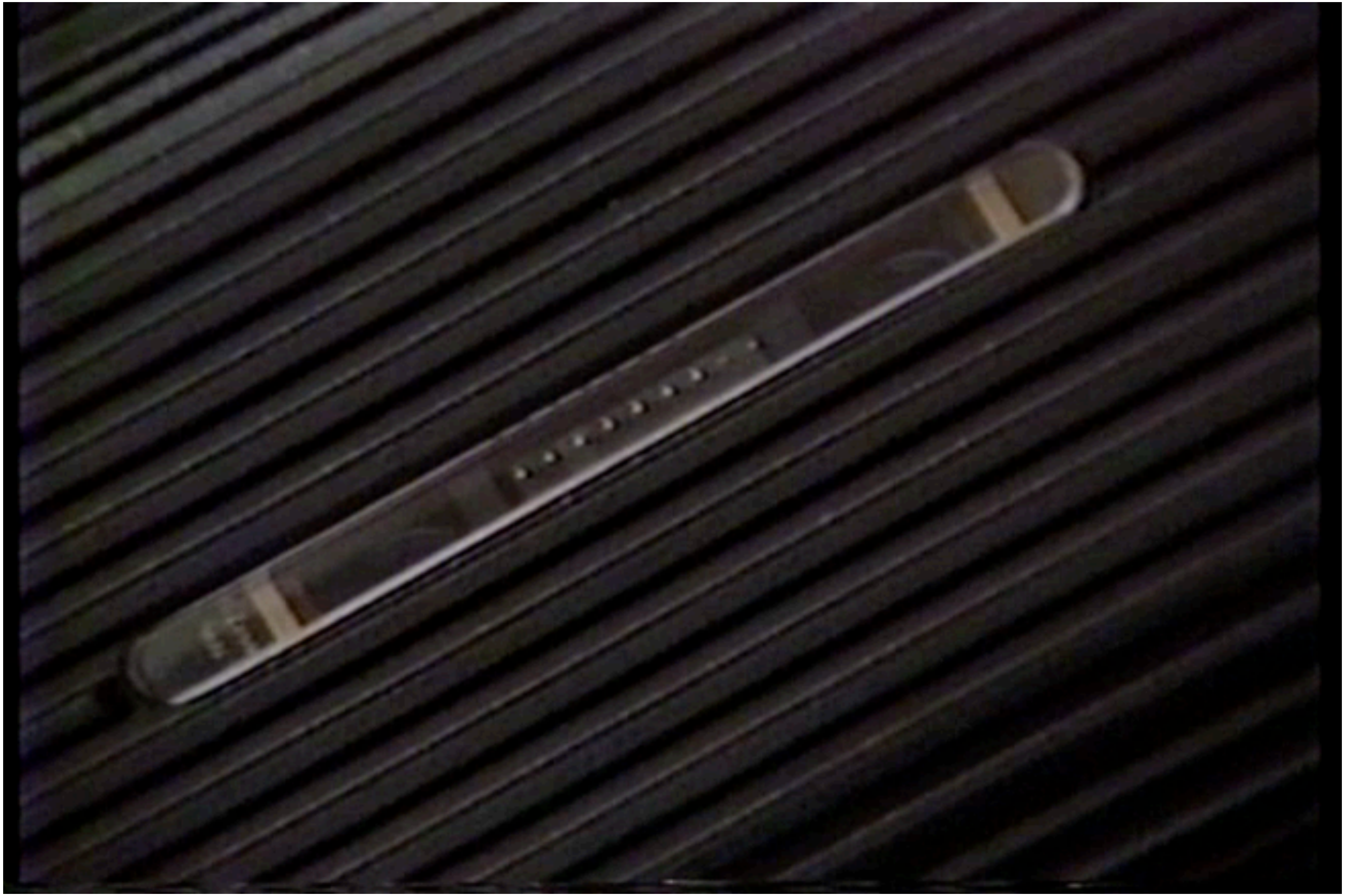
Momentum



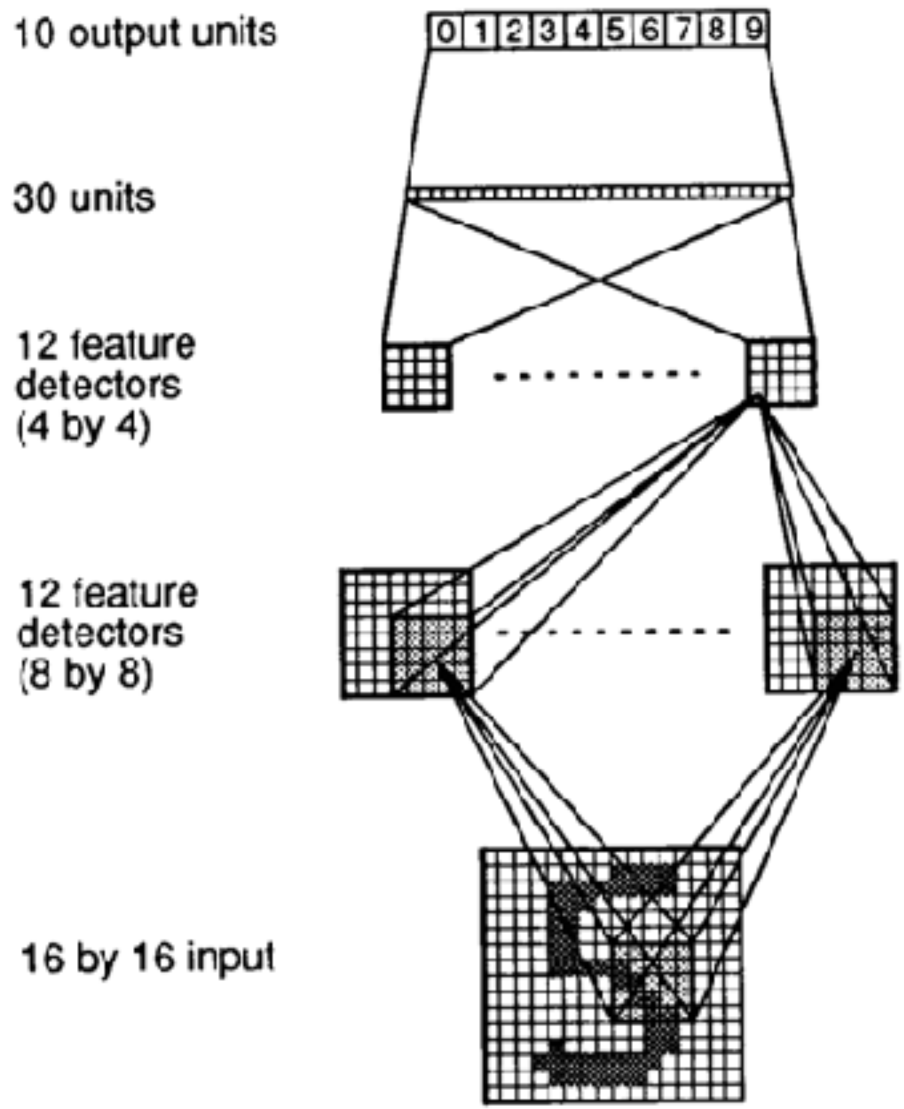
without momentum with momentum

NetTalk
(Sejnowski & Rosenberg 1987)





“LeNet”
(Yann LeCun et al., 1989)



1989

ALVINN, an autonomous land vehicle in a neural network

Dean A. Pomerleau
Carnegie Mellon University

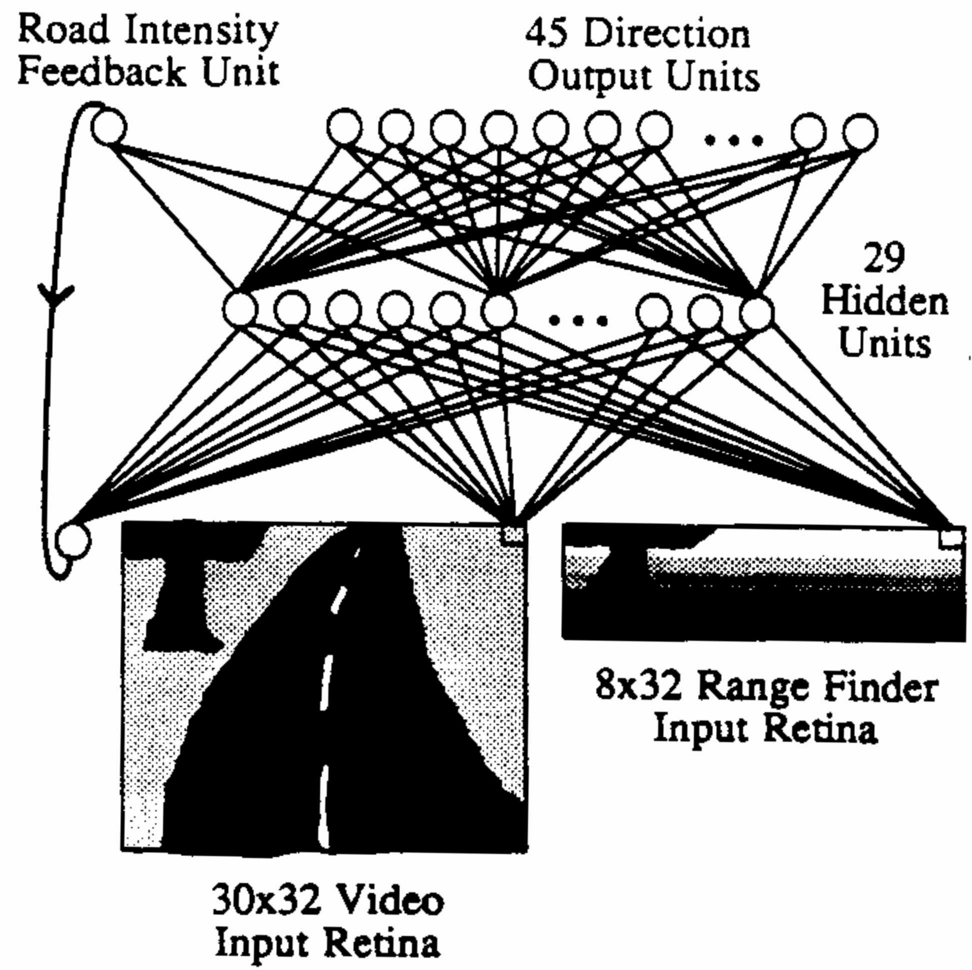
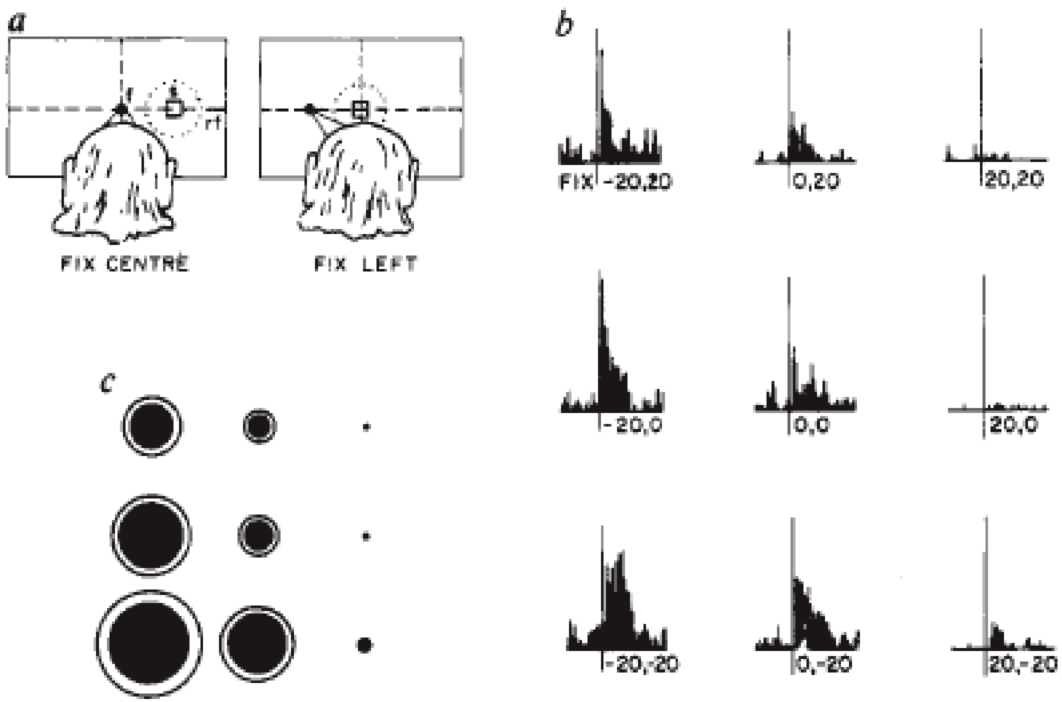
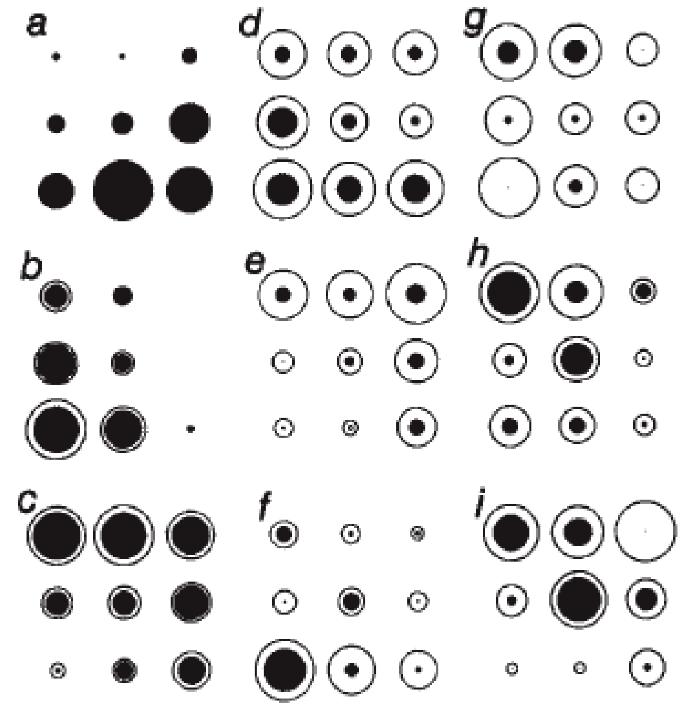


Figure 1: ALVINN Architecture

Gain Fields

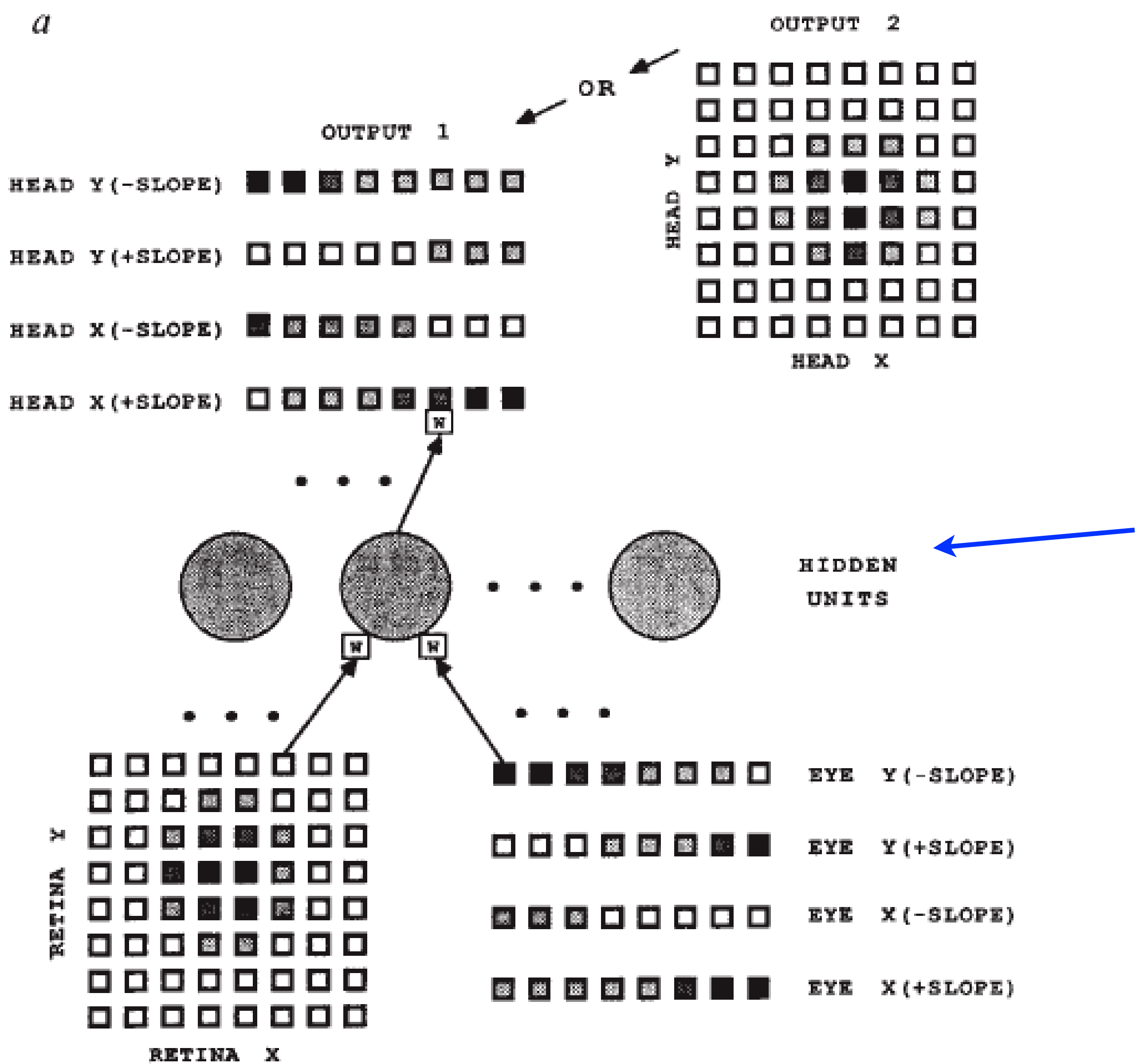
(Zipser & Anderson, 1987)

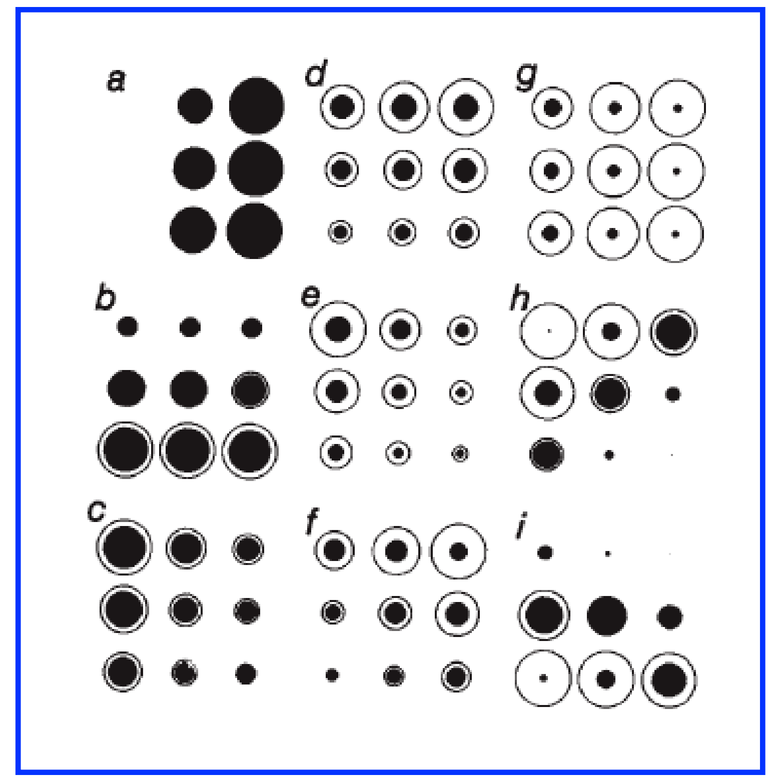




Gain Fields

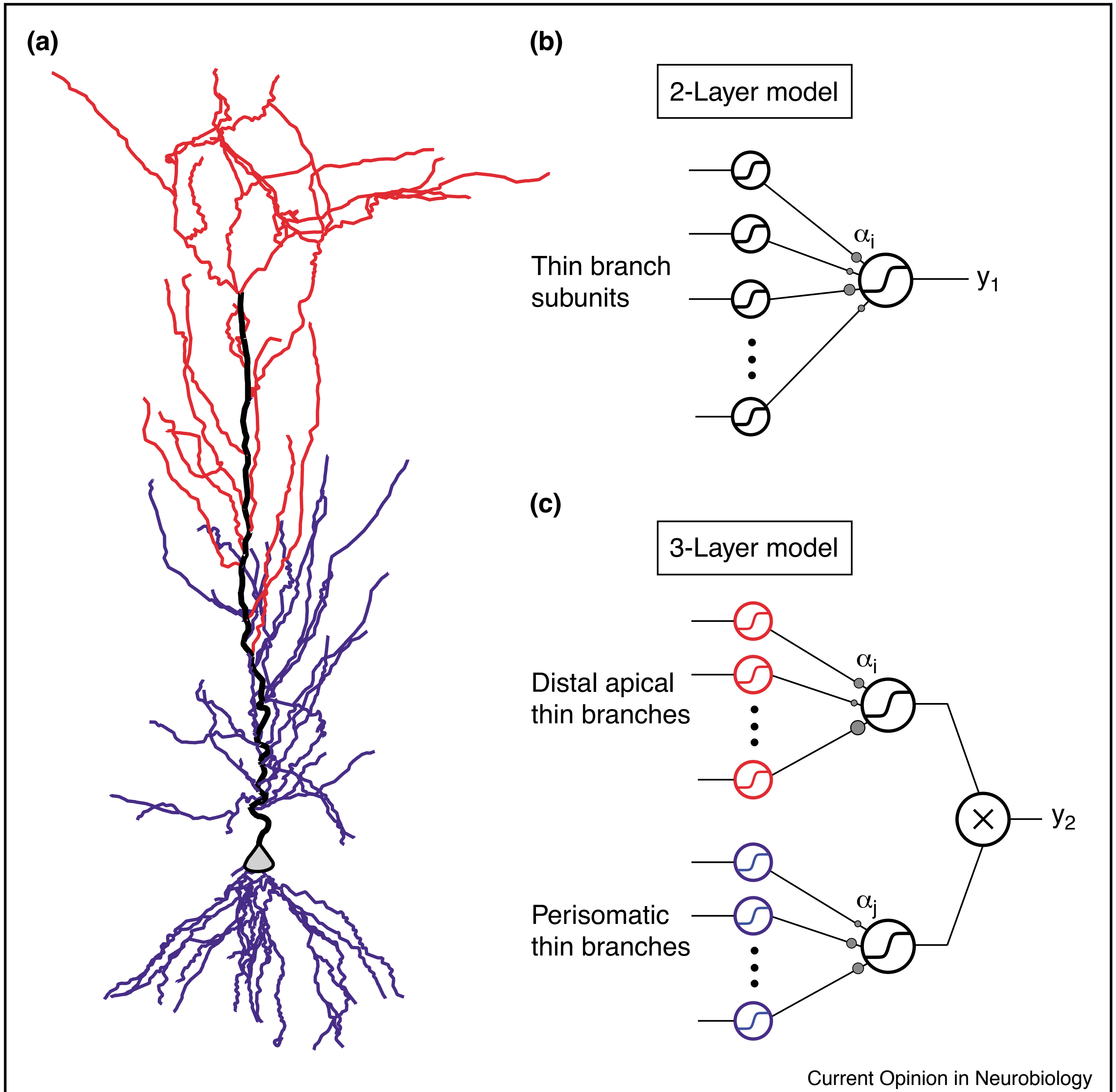
(Zipser & Anderson, 1987)





Dendritic nonlinearities

(Hausser & Mel, 2003)



Consider:

$$u = w_1 \, x_1 + w_2 \, x_2 + w_{12} \, x_1 \, x_2$$

$$y = \sigma(u)$$