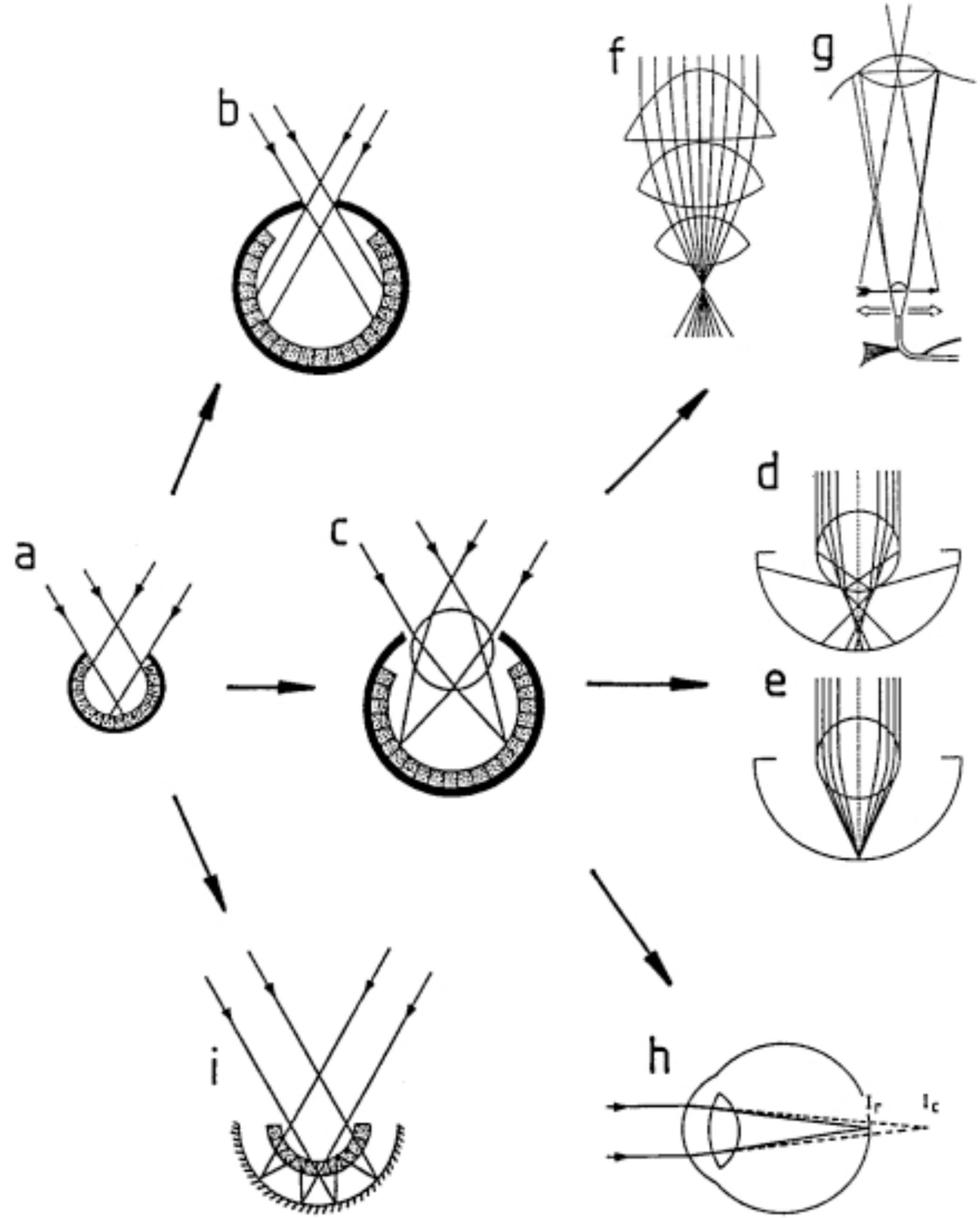
Supervised learning

Optical principles govern the design of eyes

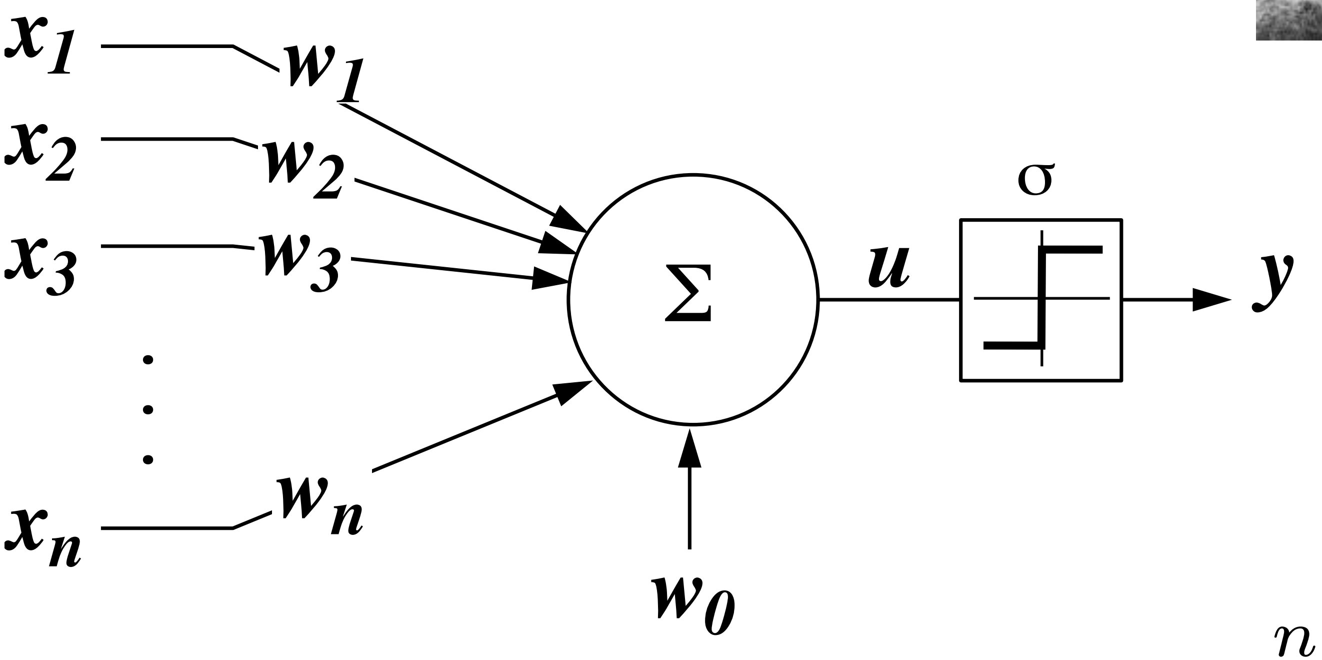
THE EVOLUTION OF EYES

Michael F. Land *Russell D. Fernald*



Perceptron model

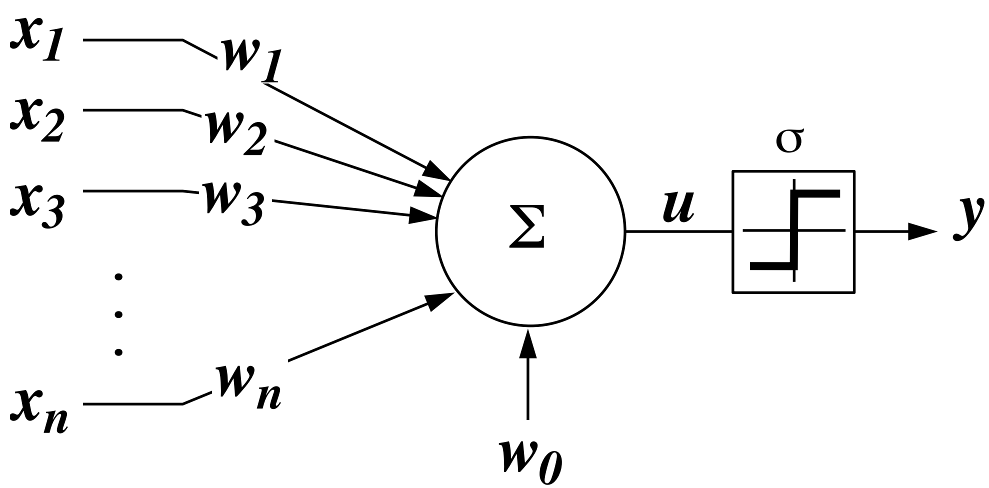
(Rosenblatt, ca. 1960)



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

$$y = \sigma(u)$$

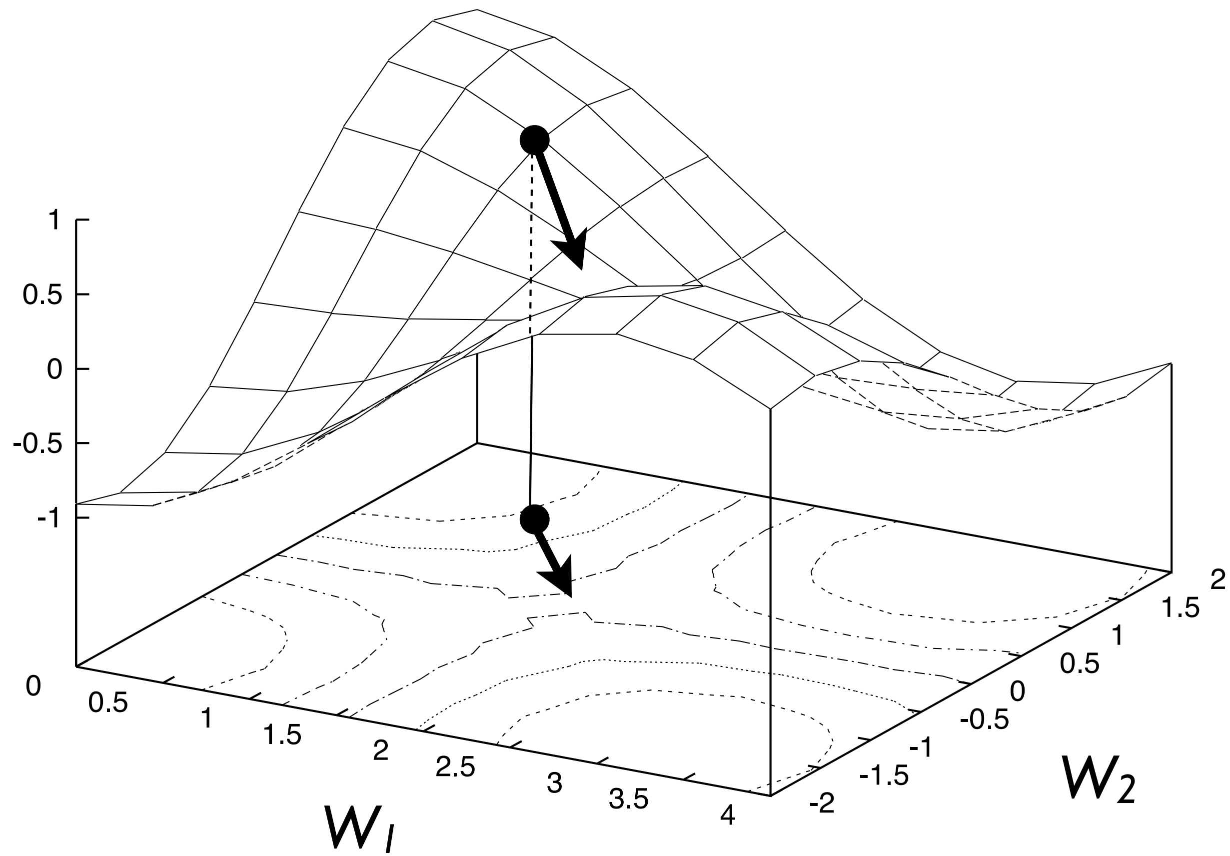
Perceptron learning rule
(Rosenblatt 1962)



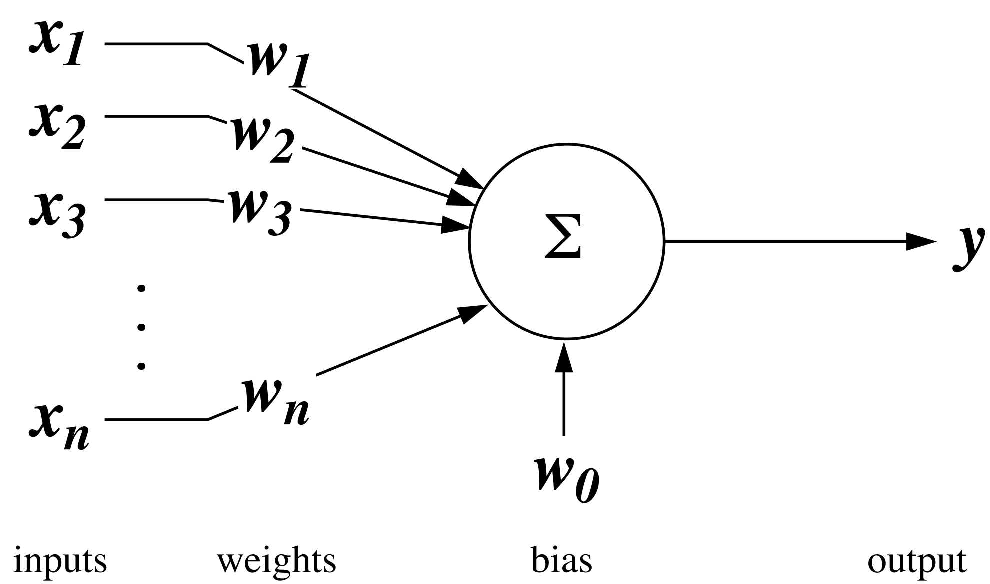
$$\Delta w_k = \begin{cases} 2\eta\, T^{(\alpha)}\, x_k^{(\alpha)} & y^{(\alpha)} \neq T^{(\alpha)} \\ 0 & \text{otherwise} \end{cases}$$

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

Gradient descent in weight space



Linear neuron learning rule (Widrow & Hoff 1960)

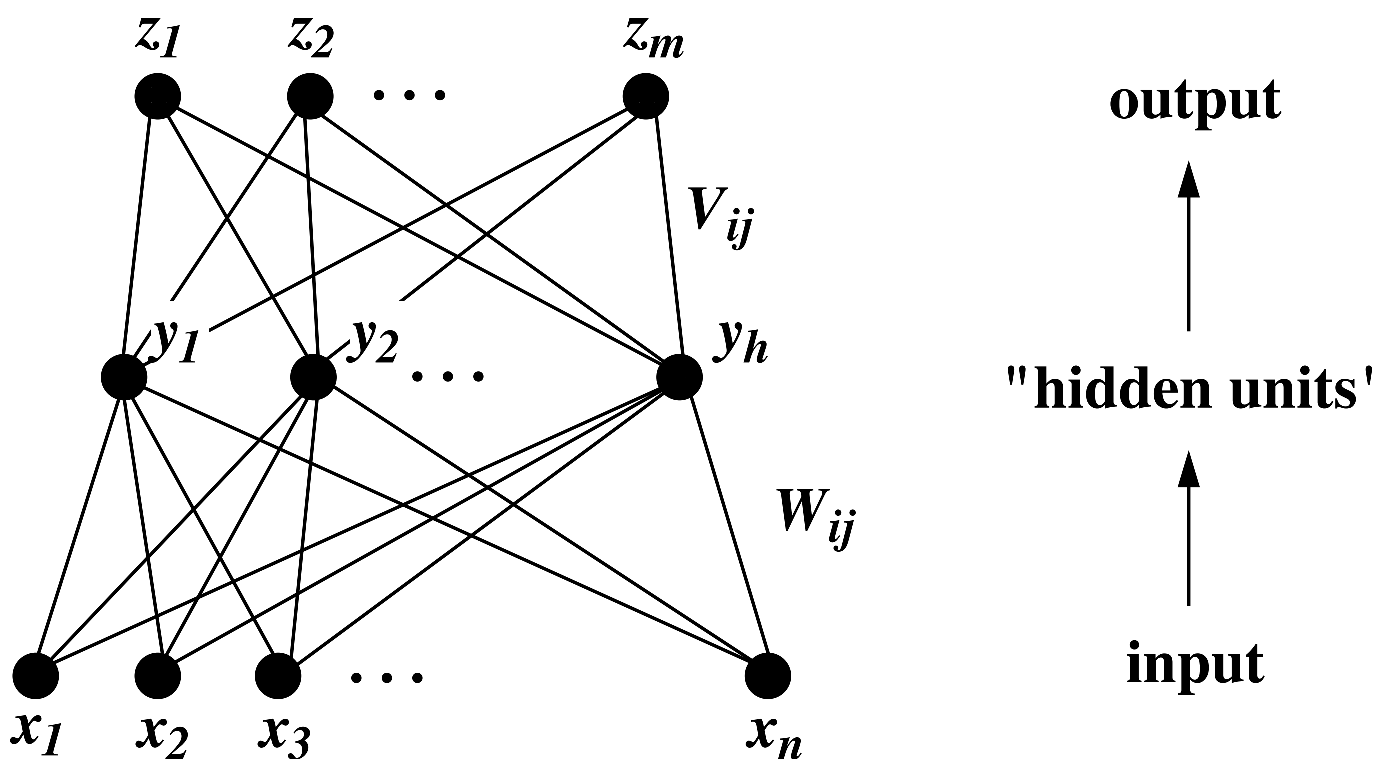


Objective function ⟶ Learning rule

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

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

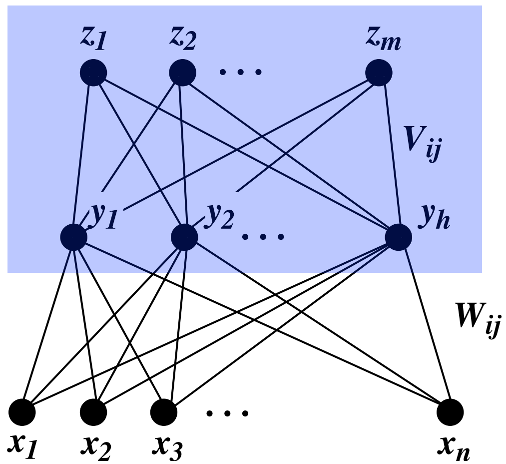
Two-layer network



$$z_i = \sigma(\sum_j V_{ij} y_j)$$

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

Learning rule for output layer



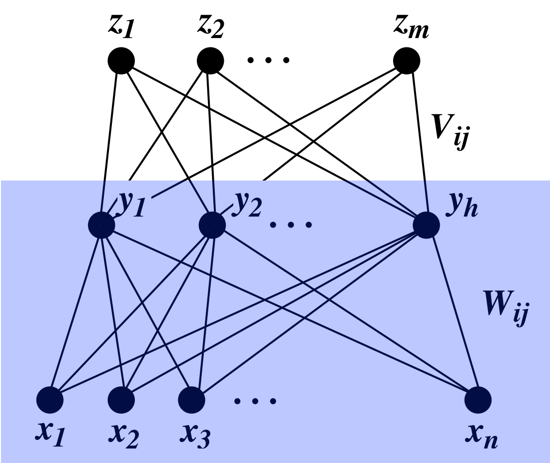
$$E^{(\alpha)} = \frac{1}{2}\sum_i \left[T_i^{(\alpha)} - z_i(\mathbf{x}^{(\alpha)})\right]^2$$

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

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

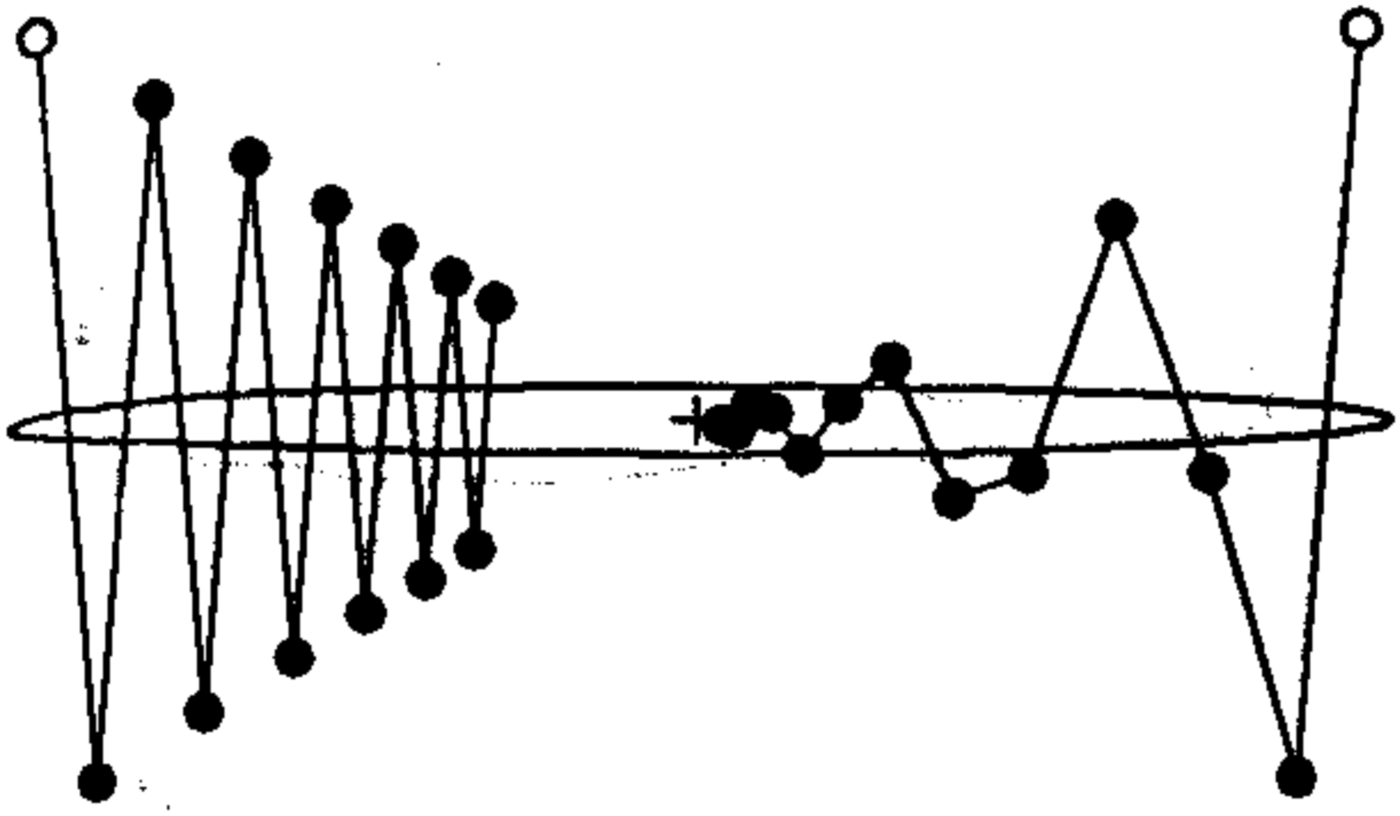
Momentum

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

Converges to

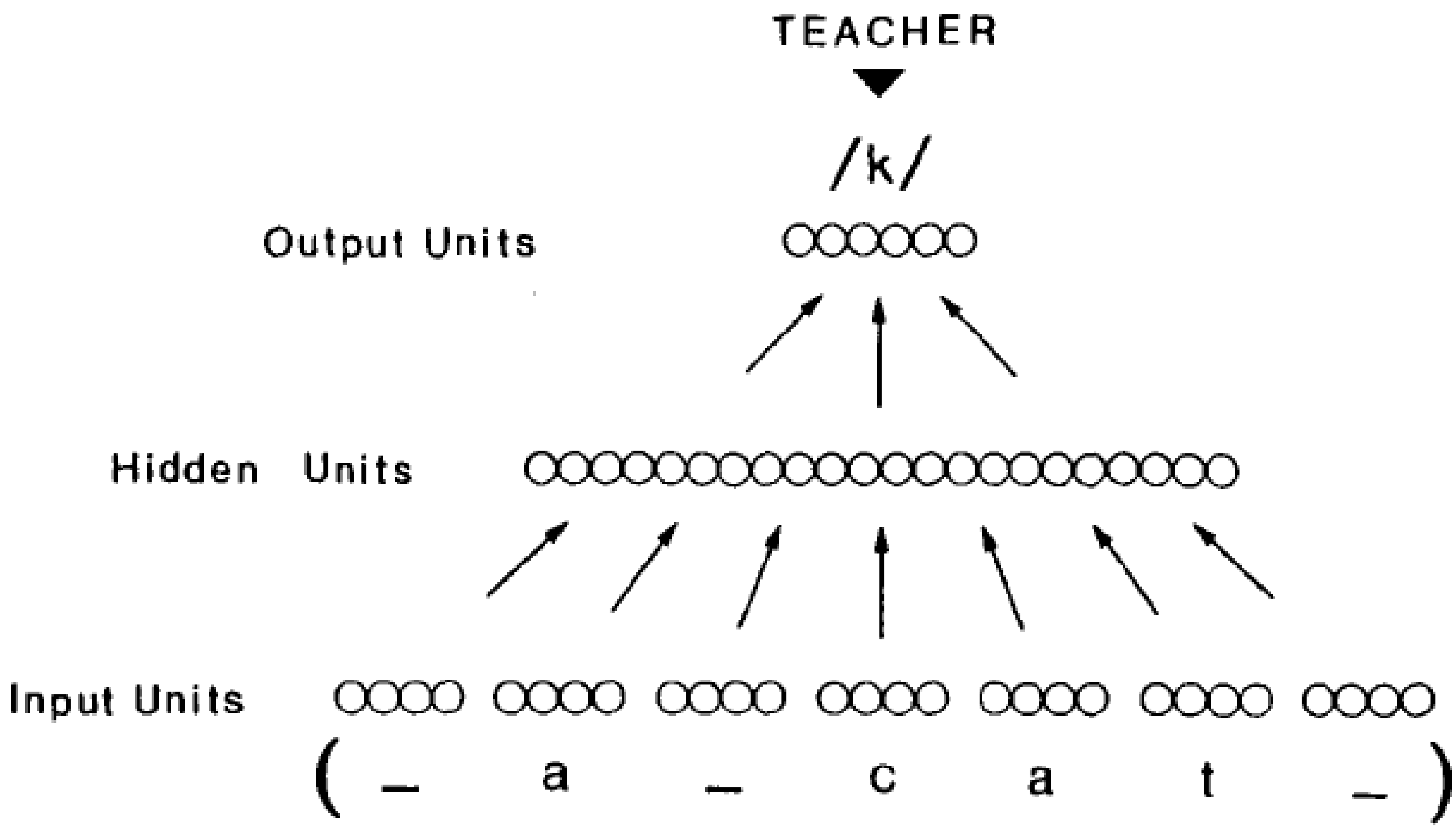
$$\Delta w_{kl} \approx -\frac{\eta}{1-\alpha} \frac{\partial E}{\partial w_{kl}}$$

Momentum

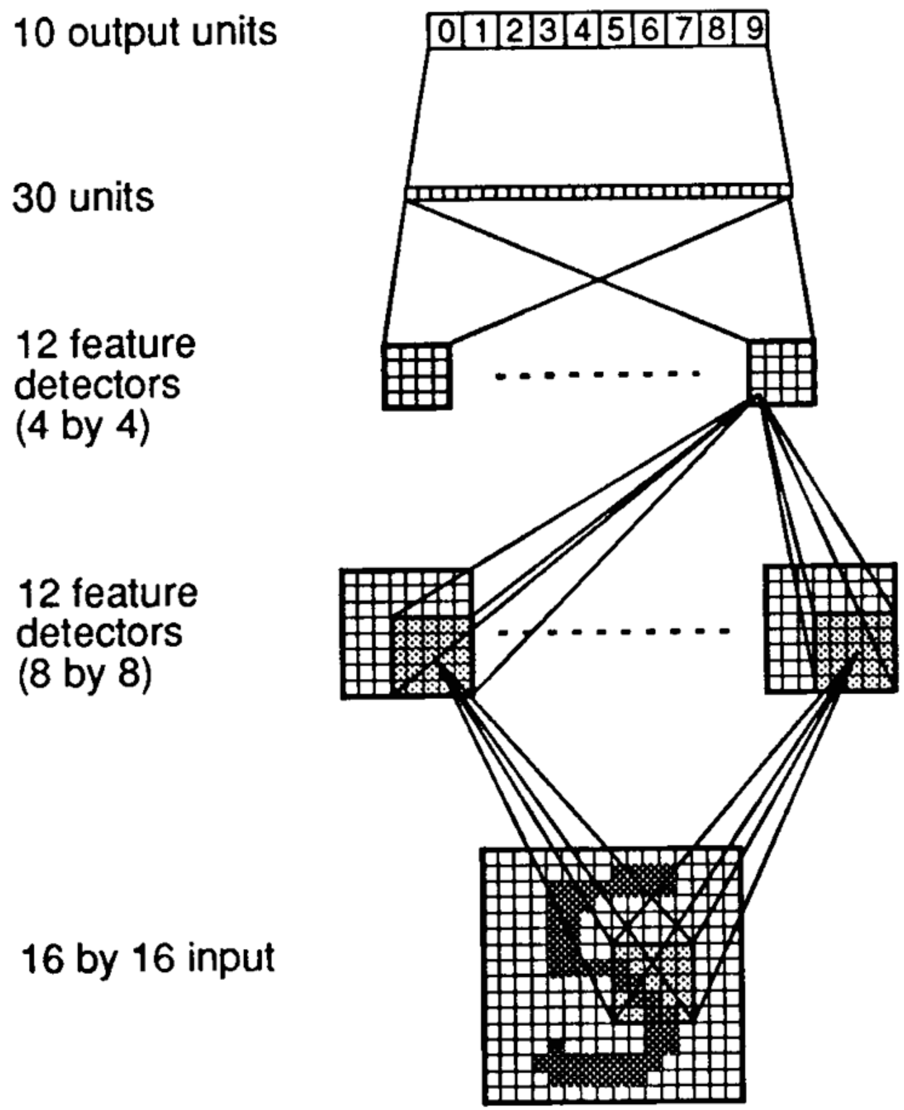


without momentum   with momentum

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

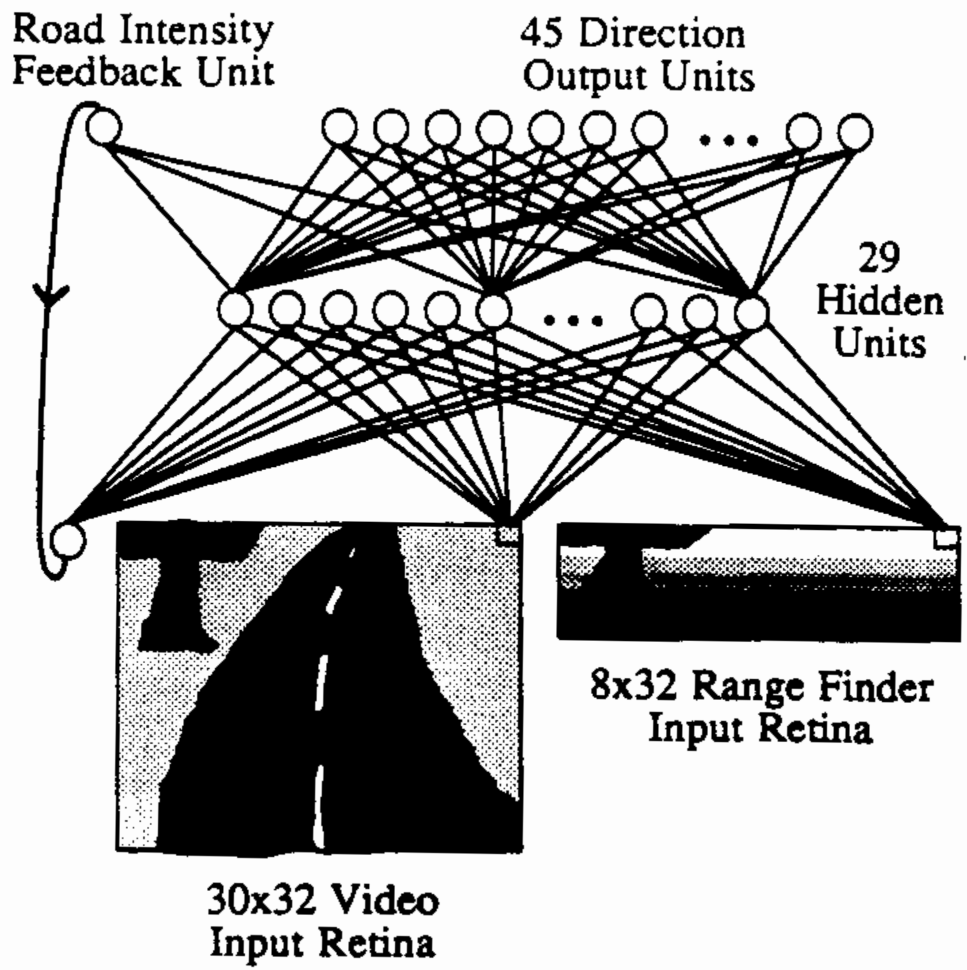
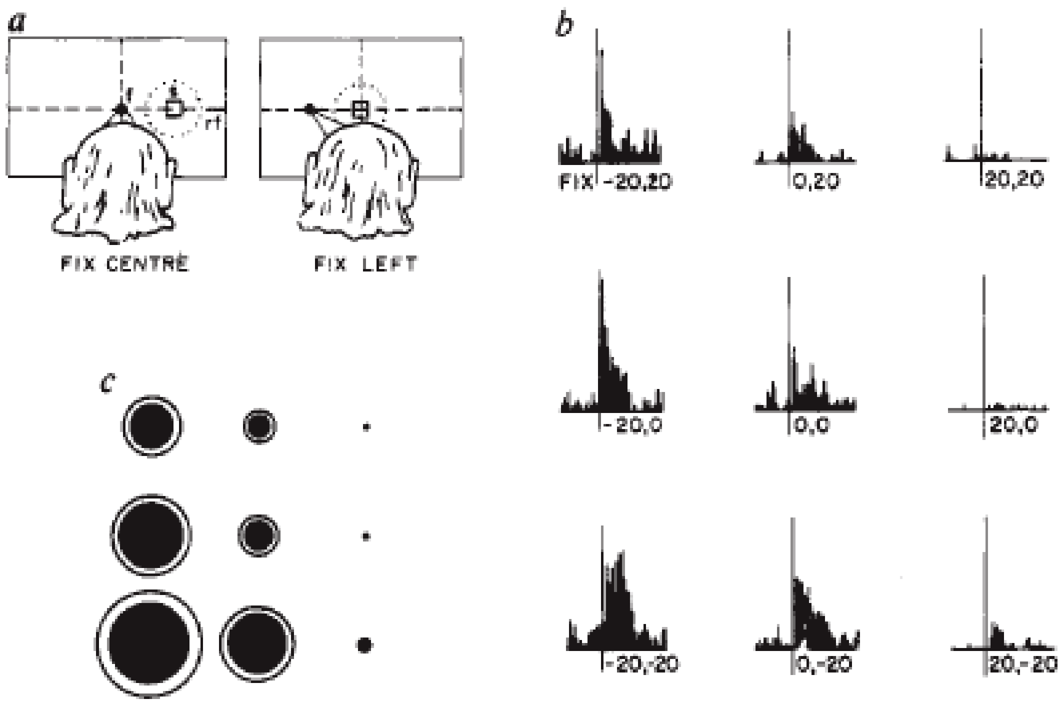
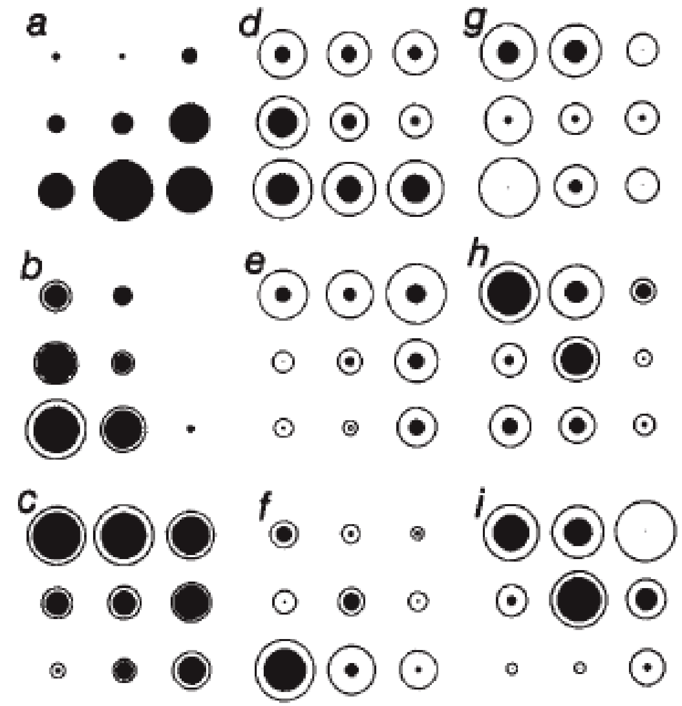


Figure 1: ALVINN Architecture

Gain Fields

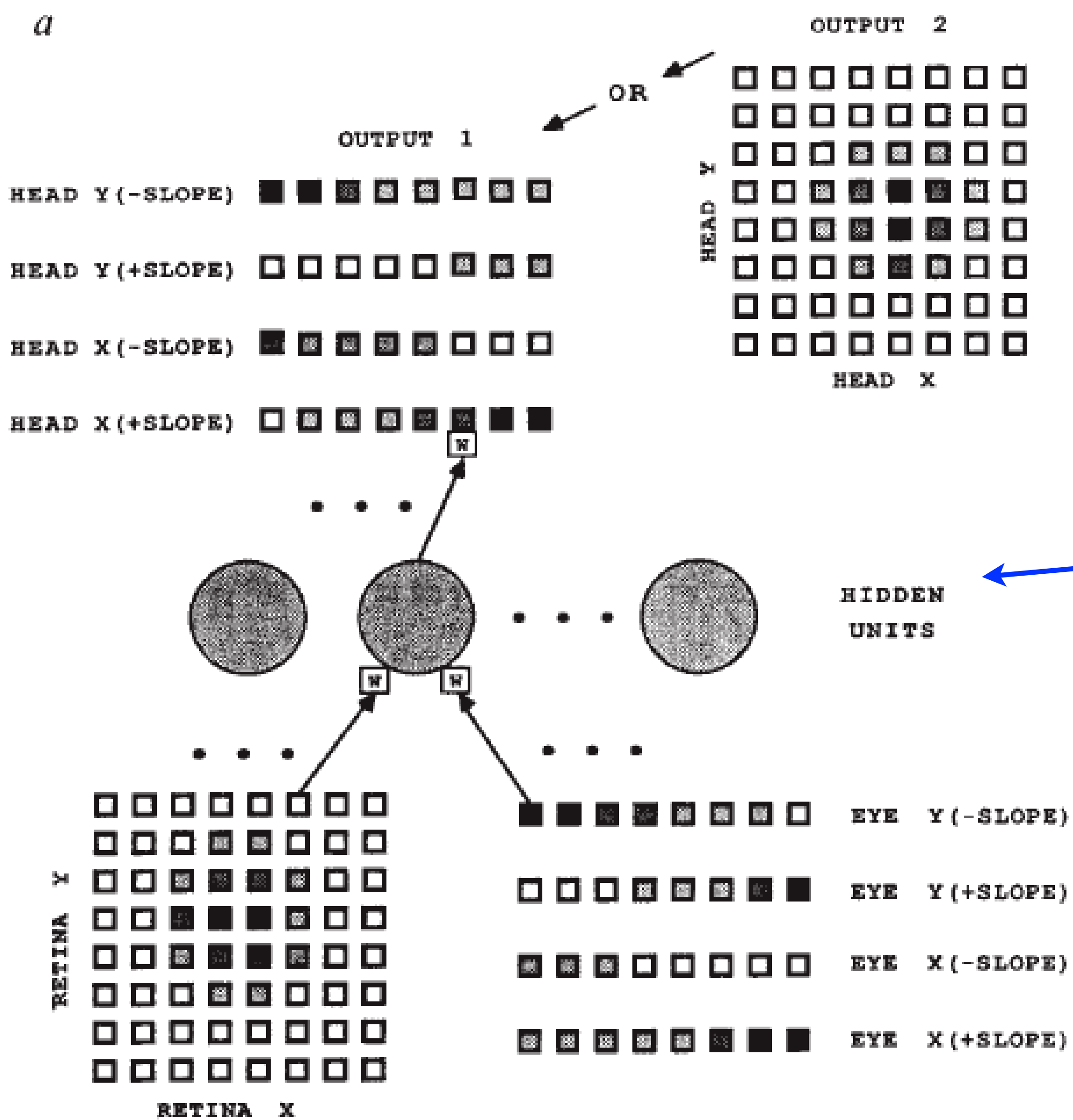
(Zipser & Anderson, 1987)

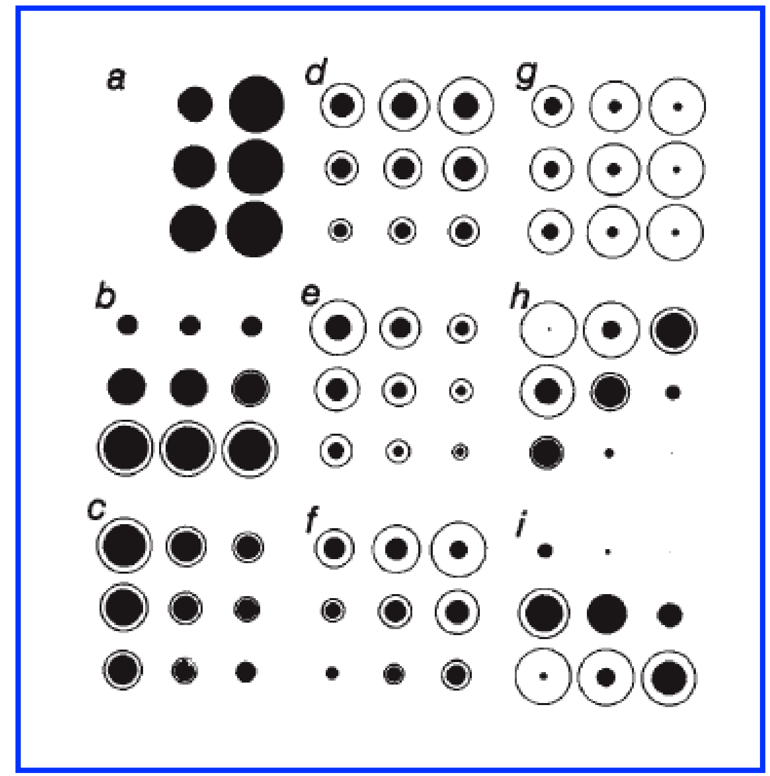




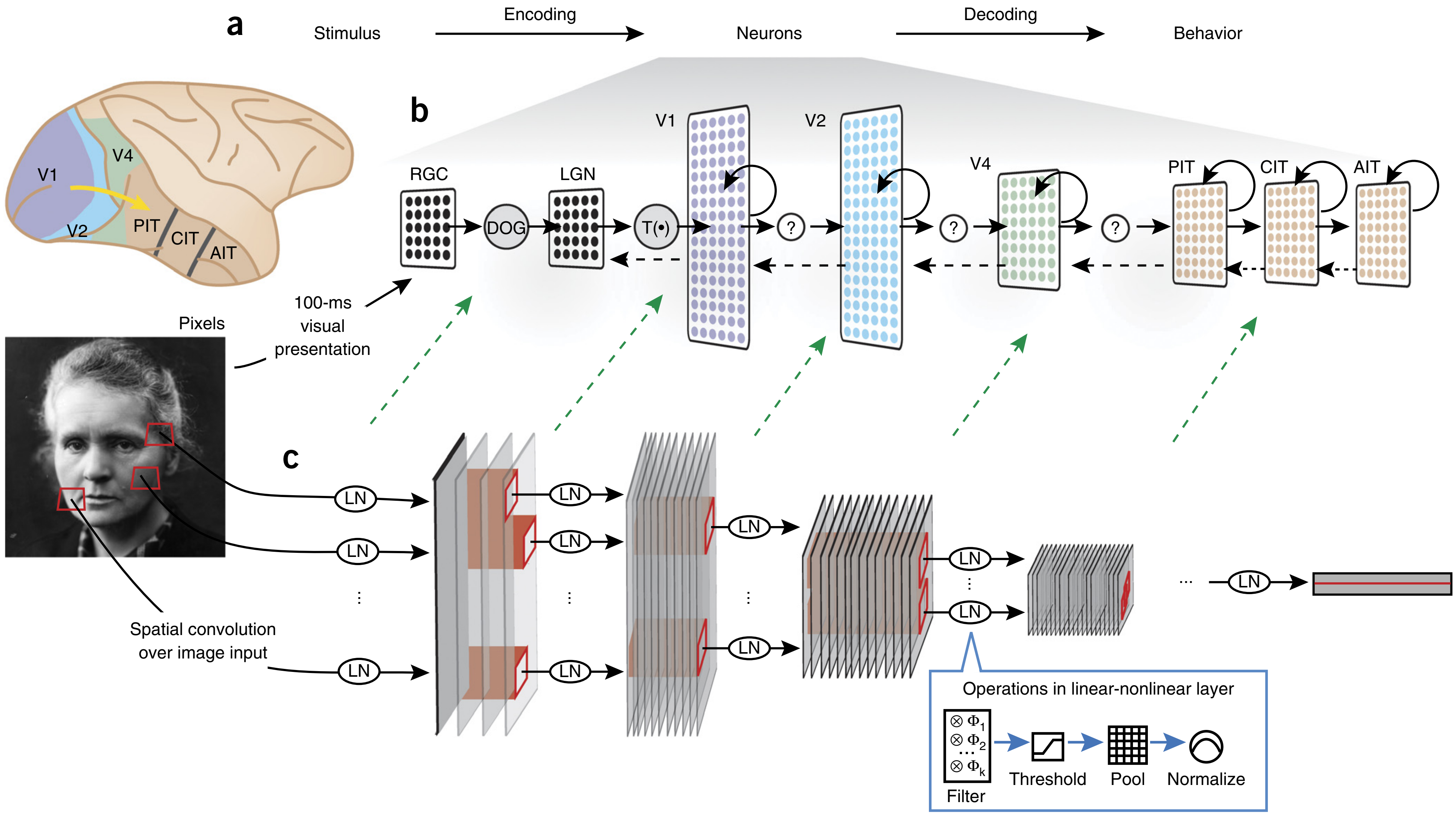
Gain Fields

(Zipser & Andersen, 1987)





Deep networks appear to predict responses of V4 and IT neurons (Yamins & DiCarlo 2016)



Machine Learning, 20, 273–297 (1995)

Support-Vector Networks

CORINNA CORTES corinna@neural.att.com
VLADIMIR VAPNIK vlad@neural.att.com
AT&T Bell Labs., Holmdel, NJ 07733, USA