Machine Vision

Lecture 4

Biologically inspired Vision

(Neural network and color vision)

Professor Kok-Meng Lee

Georgia Institute of Technology

George W. Woodruff School of Mechanical Engineering

Atlanta, GA 30332-0405

Webpage: www.me.gatech.edu/me6406

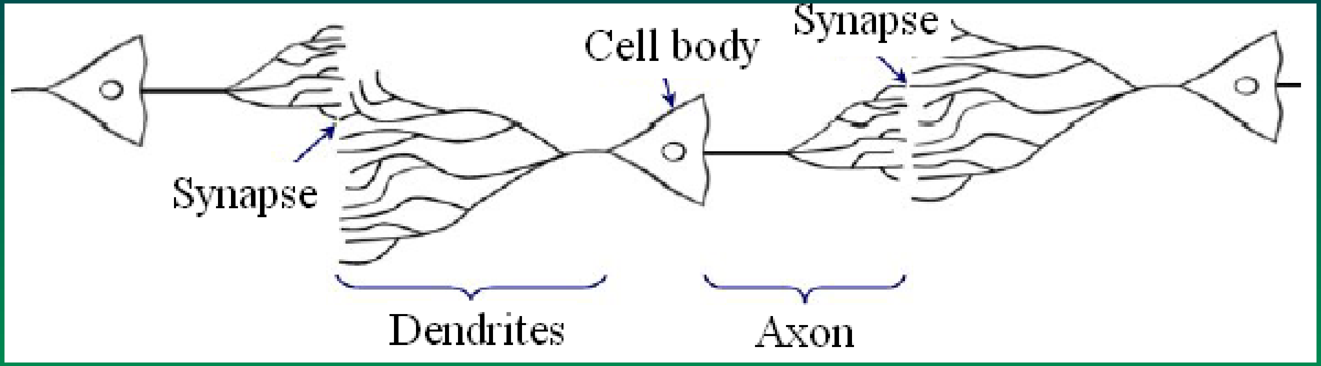
Course Outline

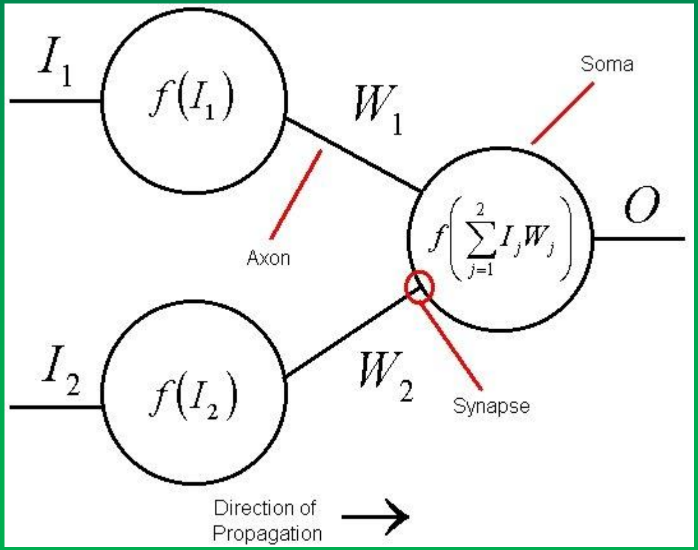
- Introduction and low-level processing
  - Physics of digital images, histogram equalization, segmentation, edge detection, linear filters.
- Model-based Vision
  - Hough transform, pattern representation, matching
- Geometric methods
  - Camera model, calibration, pose estimation
- Neural network for machine vision
  - Basics, training algorithms, and applications
- Color images and selected topics
  - Physics, perception, processing and applications

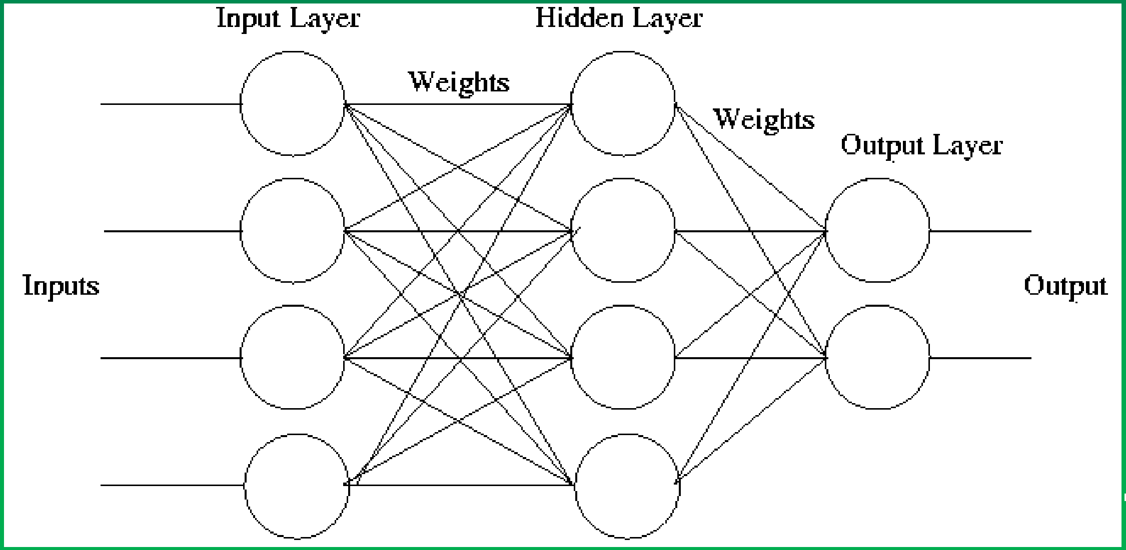
Artificial Neural Networks

as an extremely simplified model of the brain

Composed of many "neurons" that co-operate to perform the desired function.



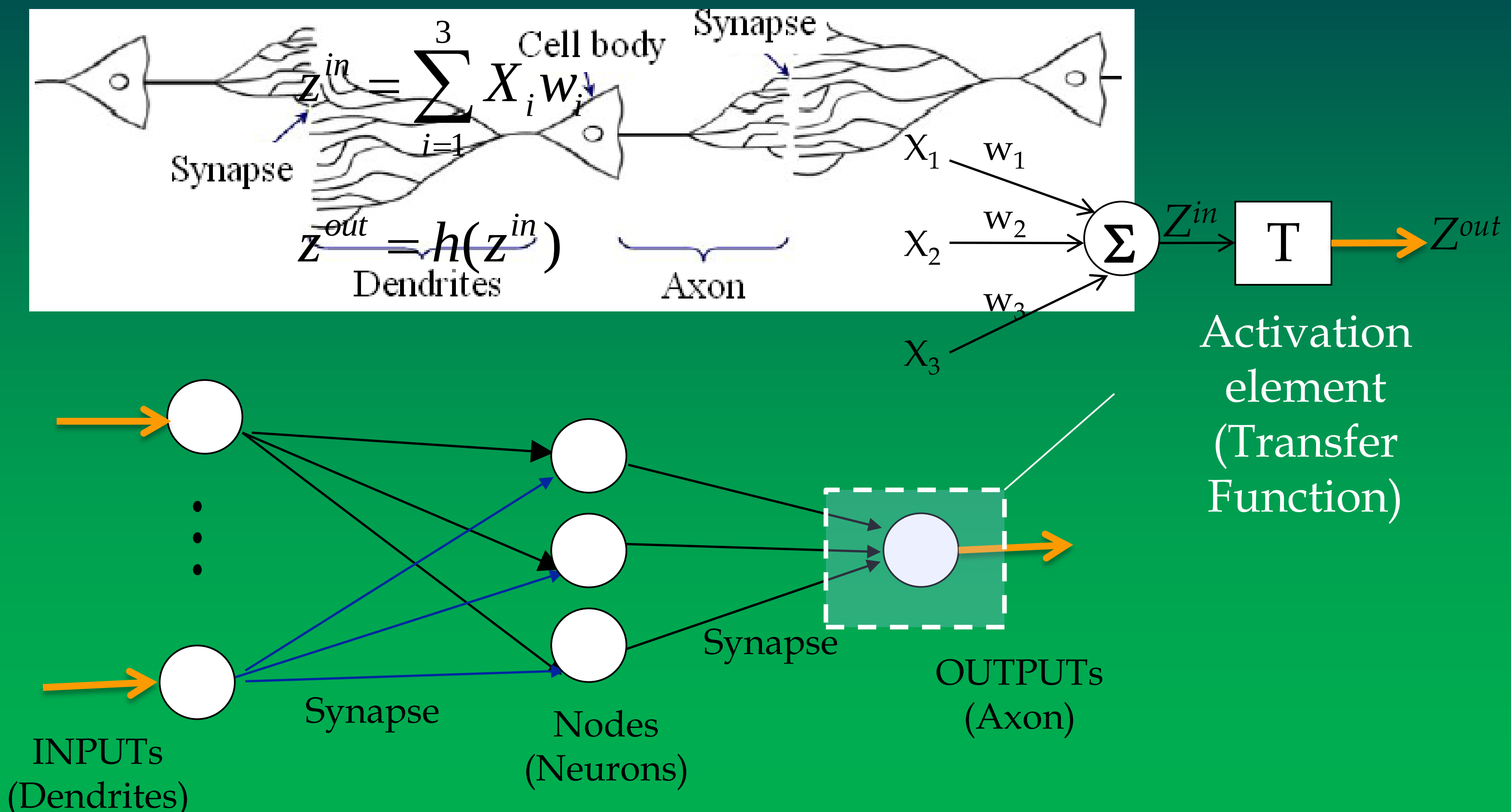




Artificial Neural Networks

as an extremely simplified model of the brain

Composed of many "neurons" that co-operate to perform the desired function.



Artificial Neural Networks *as an* extremely simplified model of the brain

What they are used for

- ► **Classification:**
  Pattern recognition, feature extraction, image matching
- ► **Noise Reduction**
  Recognize patterns and produce noiseless outputs
- ► **Prediction**
Extrapolation based on historical data

Training and Verification (Two independent sets)

- ► **Training set:**
  A group of samples used to train the neural network
- ► **Testing set:**
  A group of samples used to test the performance of the neural network and to estimate the error rate.

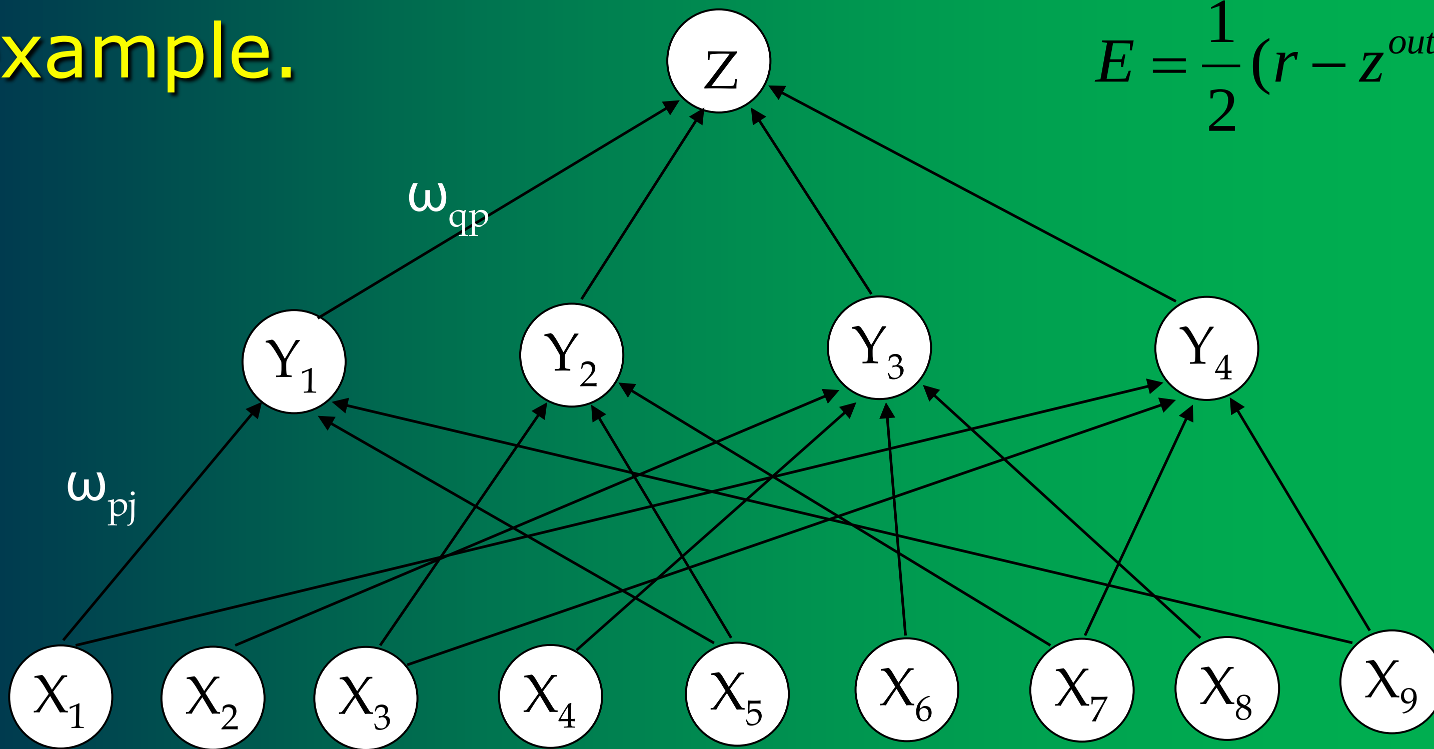
Trained by back-propagation

- an example.

Layer Q
(OUTPUT)

Layer P
(HIDDEN)

Layer J
(INPUT)



$$E = \frac{1}{2}(r - z^{out})^2$$

X_1	X_2	X_3
X_4	X_5	X_6
X_7	X_8	X_9

$$w_{qp}^{k+1} = w_{qp}^{k} - \Delta w_{qp}$$

$$\Delta w_{qp} = -\alpha \frac{\partial E}{\partial w_{qp}} = -\alpha \underbrace{\underset{-\left(r - z^{out}\right)}{\frac{\partial E}{\partial z^{out}}} \; \underset{h'(z^{in})}{\frac{\partial z^{out}}{\partial z^{in}}}}_{\delta_q} \; \underset{y_p^{out}}{\frac{\partial z^{in}}{\partial w_{qp}}} = \alpha \delta_q y_p^{out}$$

$$z^{in} = \sum_{i=1}^{3} y_i^{out} w_i$$

$$\delta_q = -\frac{\partial E}{\partial z^{in}}$$
$$= (r - z^{out}) h'(z^{in})$$

Illustrative pattern example:

Layer Q
(OUTPUT)

Layer P
(HIDDEN)

Layer J
(INPUT)

Note: The desired output of the hidden layer is not known, we approximate E_p by E.

$$\boxed{\begin{aligned} w_{pj}^{k+1} &= w_{pj}^{k} + \Delta w_{pj} \\ \Delta w_{pj} &= -\alpha \frac{\partial E}{\partial w_{pj}} = \alpha \delta_p x_j \end{aligned}}$$

$$\delta_p = -\frac{\partial E}{\partial y^{out}} \underbrace{\frac{\partial y^{out}}{\partial y^{in}}}_{h'(y^{in})} \quad \text{where } E = \frac{1}{2}(r - z^{out})^2 = \frac{1}{2}\left[r - h(\sum_{i=1}^{N_p} y_i^{out} w_{qp}) \right]^2$$

Using chain rule,

$$-\frac{\partial E}{\partial y^{out}} = -\underbrace{\frac{\partial E}{\partial z^{in}}}_{\delta_q} \underbrace{\frac{\partial z^{in}}{\partial y^{out}}}_{w_{qp}} = \delta_q w_{qp} \qquad \boxed{\delta_p = \delta_q w_{qp}}$$

Note: If Layer Q has N_q outputs,

$$E = \frac{1}{2} \sum_{q=1}^{N_q} (r_q - z_q^{out})^2 \qquad \delta_q = (r_q - z_q^{out}) h'(z_p^{in})$$

In general,

if Layer Q has N_q outputs, $E = \frac{1}{2}\sum_{q=1}^{N_q}(r_q - z_q^{out})^2$

$$w_{qp}^{k+1} = w_{pq}^{k} + \alpha\delta_q y_p \qquad \delta_q = (r_q - z_q^{out})h'(z_q^{in})$$

Hidden Layer P (N_p Nodes)

$$w_{pj}^{k+1} = w_{pj}^{k} + \alpha\delta_p x_j \qquad \delta_p = h'(y_p^{in})\sum_{q=1}^{N_q}\delta_q w_{qp}$$

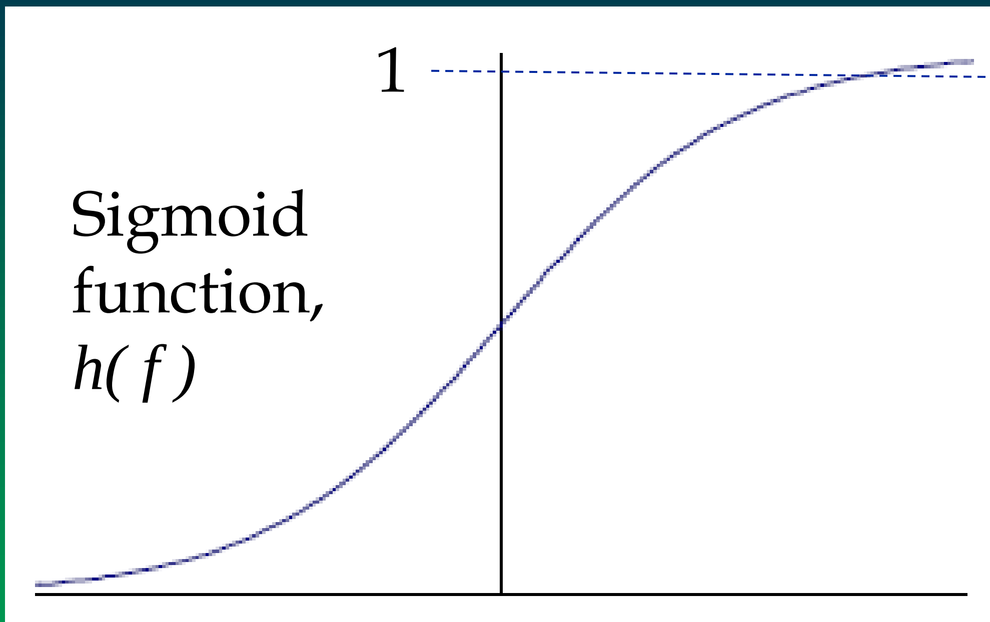
Hidden Layer J (N_q Inputs)

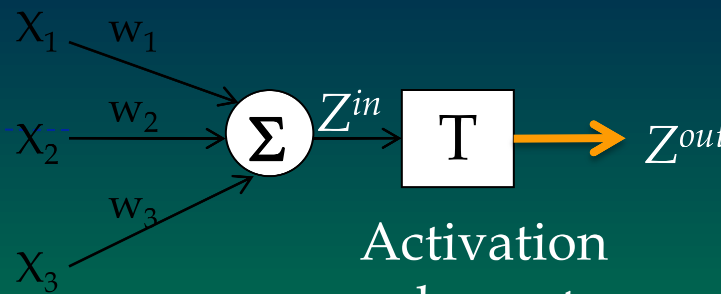
$$w_{jk}^{k+1} = w_{jk}^{k} + \alpha\delta_J u_k \qquad \delta_j = h'(y_p^{in})\sum_{p=1}^{N_p}\delta_p w_{pj}$$

Hidden Layer K (N_k Inputs)

$$\vdots$$

Activation element – an example





Activation element (Transfer Function)

$$z^{out} = h(f) = \frac{1}{1+e^{-f}}$$

$$\frac{dh(f)}{df} = \left(1+e^{-f}\right)^{-2}(-e^{-f}) = \frac{1}{1+e^{-f}}\underbrace{\frac{e^{-s}}{1+e^{-f}}}_{1-\frac{1}{1+e^{-f}}} = z(1-z^{out})$$

Algorithms

- Initialize weights (small randomized values)
- The above weight changes are performed for a single training pattern, called a <u>learning step</u>.
- After a learning step is finished, the next training pattern is submitted and the learning step is repeated.
- The learning steps proceed until all the patterns in the training set have been exhausted. This terminates the complete learning cycle, known as <u>one epoch</u>.
- The cumulative cycle error is computed for the complete learning cycle using

  $$E = \frac{1}{2}\sum_{q=1}^{N_q}(r_q - z_q^{out})^2$$

  and then compared with the maximum error allowed.
- If the total error is not satisfied, a new learning cycle will be initiated. The input patterns are recycled to train the network continuously for more epochs until satisfactory outputs are obtained.