CSC7130 Advanced Artificial Intelligence

An Introduction to Neural Networks and Machine Learning

Irwin King

Department of Computer Science and Engineering
The Chinese University of Hong Kong

http://www.cse.cuhk.edu.hk/~king

Course Notes

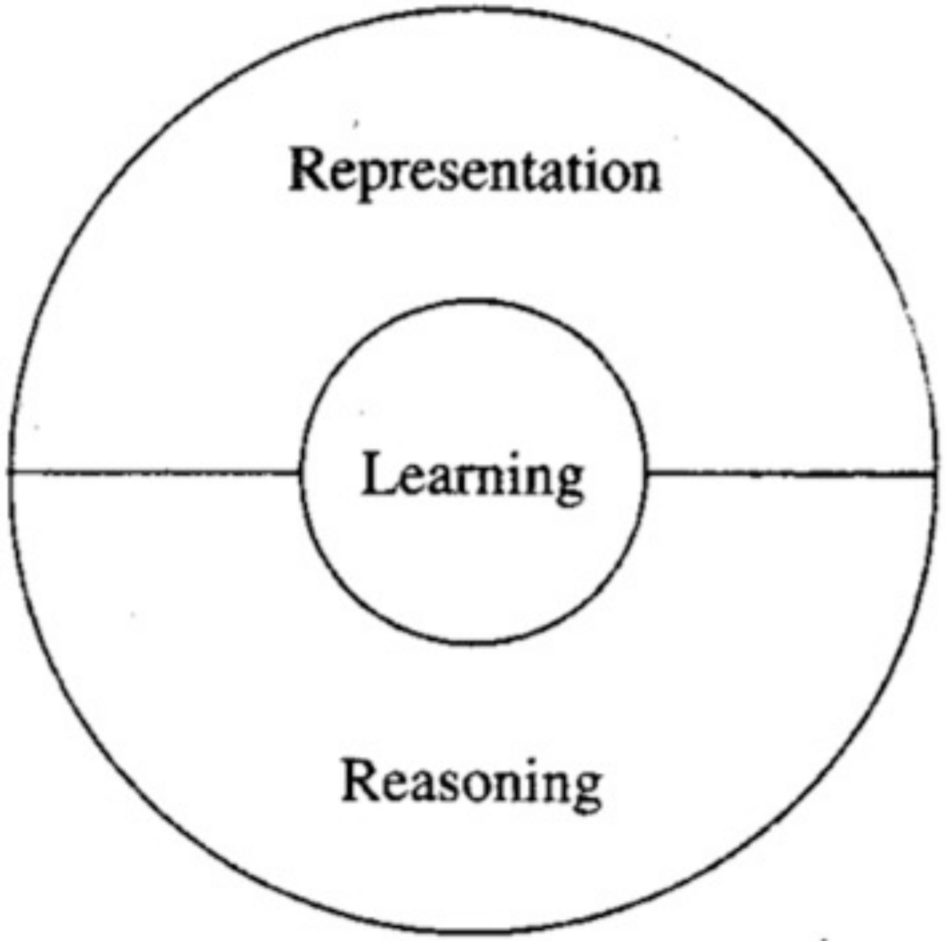
- Irwin King, 908 HSH Engineering Building, 2609 8398
- http://www.cse.cuhk.edu.hk/~king
- email: king@cse.cuhk.edu.hk
- Fax: 2603 5024
- http://wiki.cse.cuhk.edu.hk/irwin.king/teaching/csc7130/2009
- S. Haykin, Neural Networks and Learning Machines, 3rd ed., Pearson International Edition, 2009.
- X. Zhu, A. B. Goldberg, R. Brachman, and T. Dietterich, Introduction to Semi-Supervised Learning, Morgan and Claypool Publishers, 2009.

Advanced Artificial Intelligence

- Artificial Neural Networks
- Speech Processing
- Distributed Agent Technology
- Learning Theory
- Image Processing & Analysis
- Genetic Algorithm/Evolutionary Computing

Overview

- Neural Networks
  - Biological
  - Artificial
- Machine Learning Methods
- Applications



Artificial NN

- Models of a Neuron
- Network Architectures
- Learning Processes
- Learning Tasks
  - Regression
  - Classification
  - Clustering
- Perceptron
- Multilayer Perception
- Self-Organizing Maps
- Neurodynamics

Objectives for ANN

- To appreciate the advantages and limitations of neural networks for solving a wide range of practical problems
- To understand the neural network architectures which are suitable for different types of applications

Goals

- Introduction to Neural Networks
  - Biological vs. Artificial
  - Neural Network Architecture
    - Feedforward Networks
    - Recurrent Networks
  - Learning Rules
    - Supervised Learning
    - Unsupervised Learning
    - Reinforcement Learning

Benefits of Neural Networks

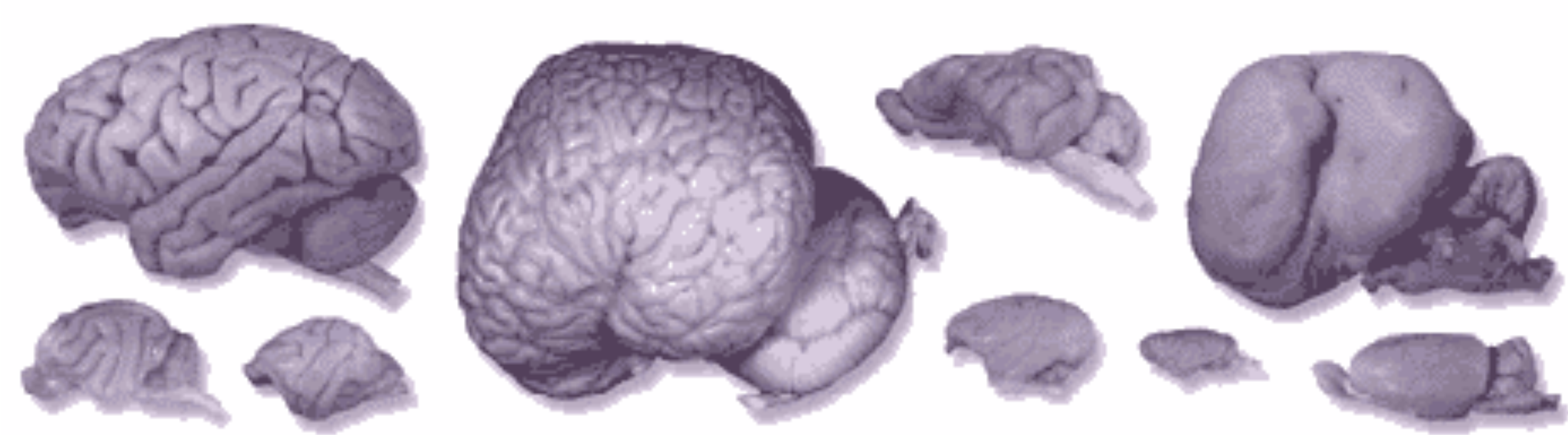
- Nonlinearity
- Input-Output Mapping
- Adaptivity
- Evidential Response
- Contextual Information
- Fault Tolerance
- VLSI Implementability
- Uniformity of Analysis and Design
- Neurobiological Analogy

Biological NN

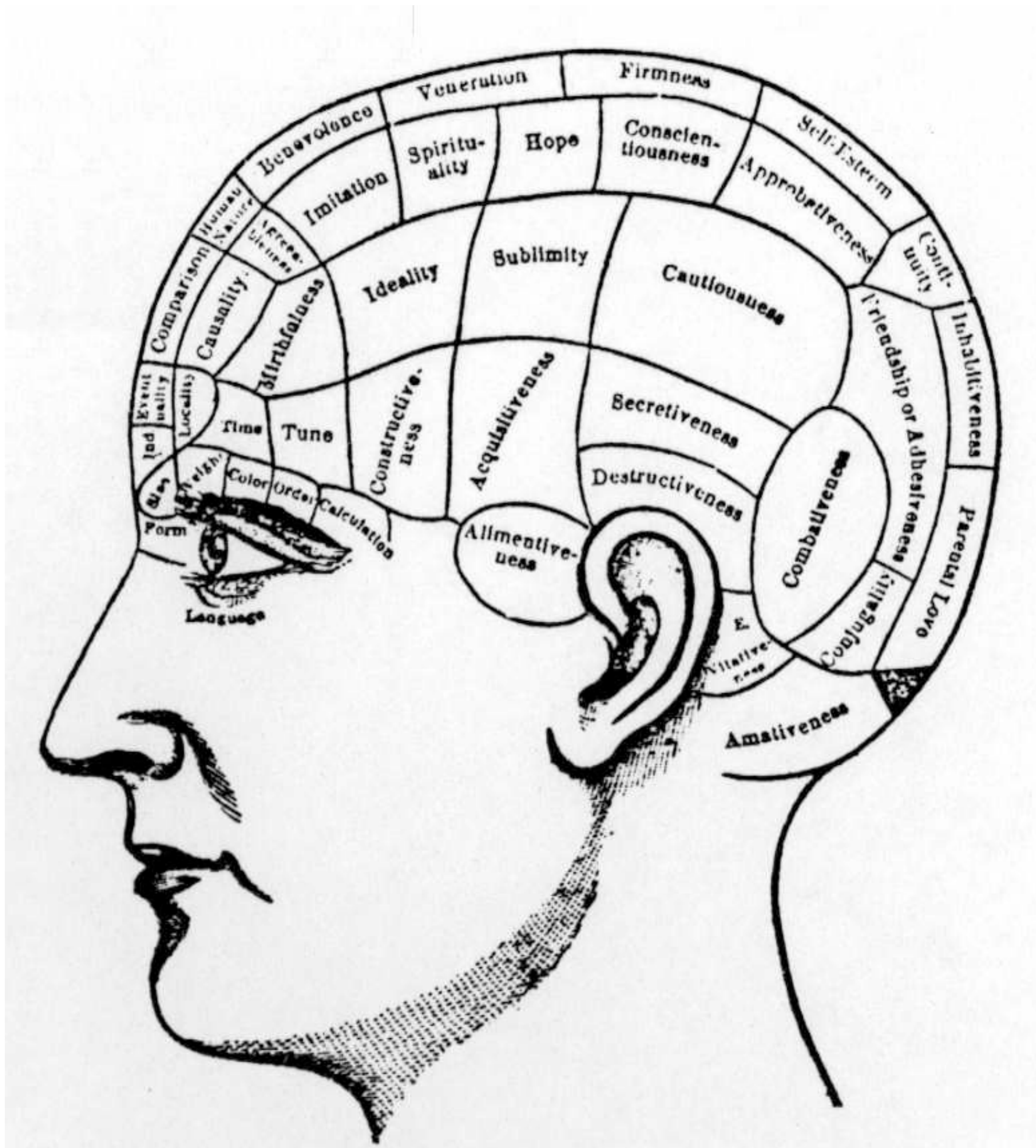
- The Human Brain
- Neurosciences:
  - Neuroanatomy
  - Neurochemistry
  - Neurophysiology
  - Psychophysics
- Actual Modeling of biological parts:
  - Hudgkin-Huxley Model,
  - Cable Equations, etc

Biological Brain

- Brains from several different species
- (Image courtesy of the Mammalian Brain Collection at the University of Wisconsin, Michigan State University and National Museum of Health and Medicine)



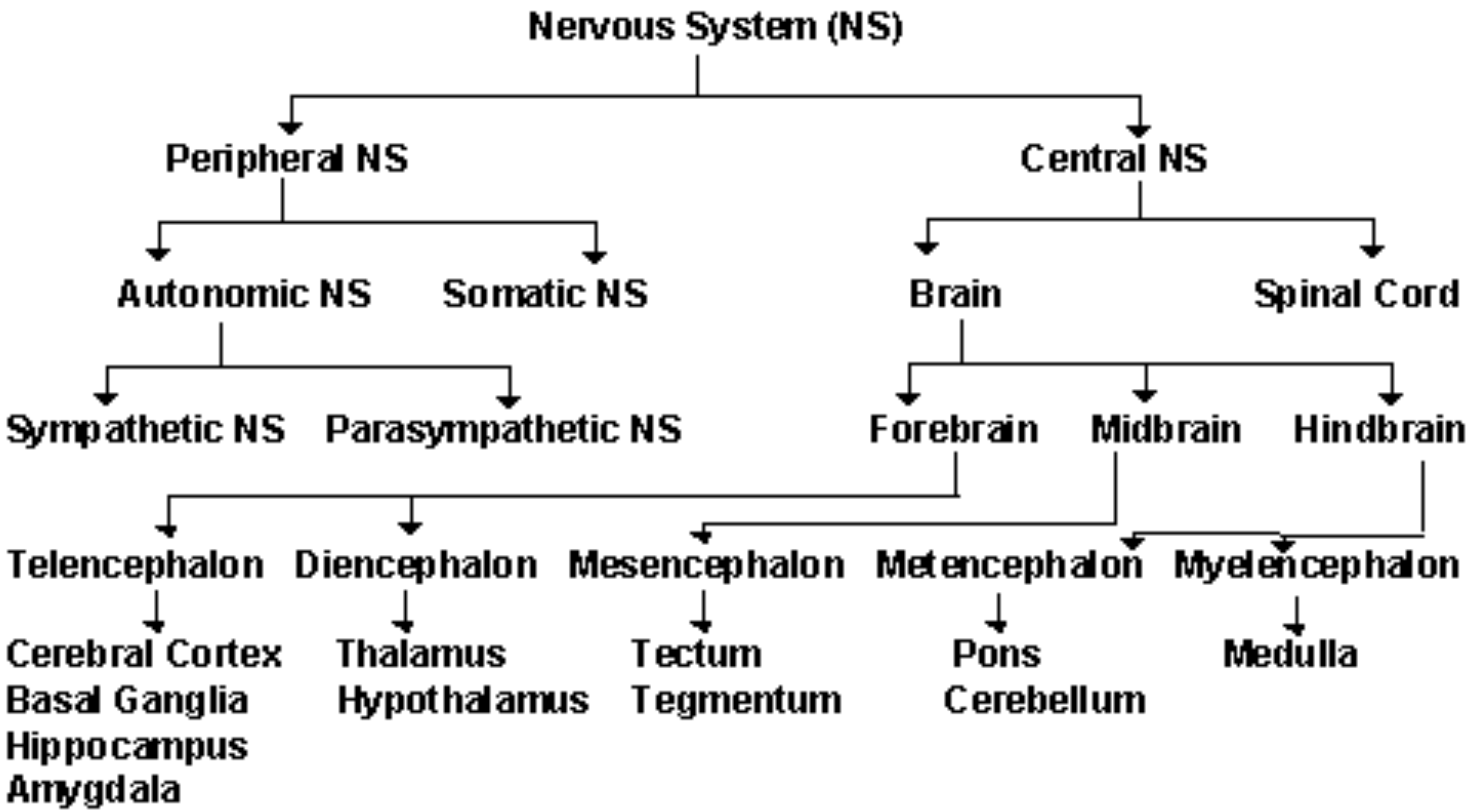
Views on Brain



Brain Facts

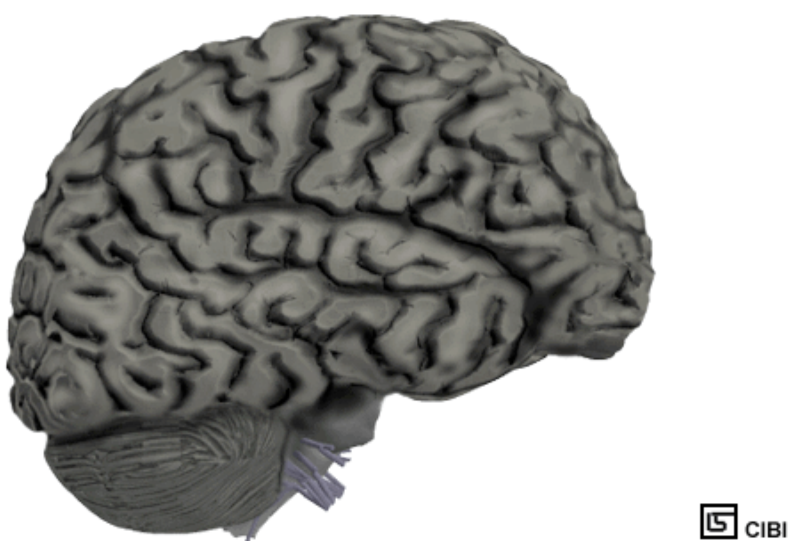
- The central nervous system is divided into two major parts: the brain and the spinal cord.
- In the average adult human, the brain weighs 1.3 to 1.4 kg. (about 3 pounds).
- The brain contains about 100 billion nerve cells (neurons) and trillions of "support cells" called glia.

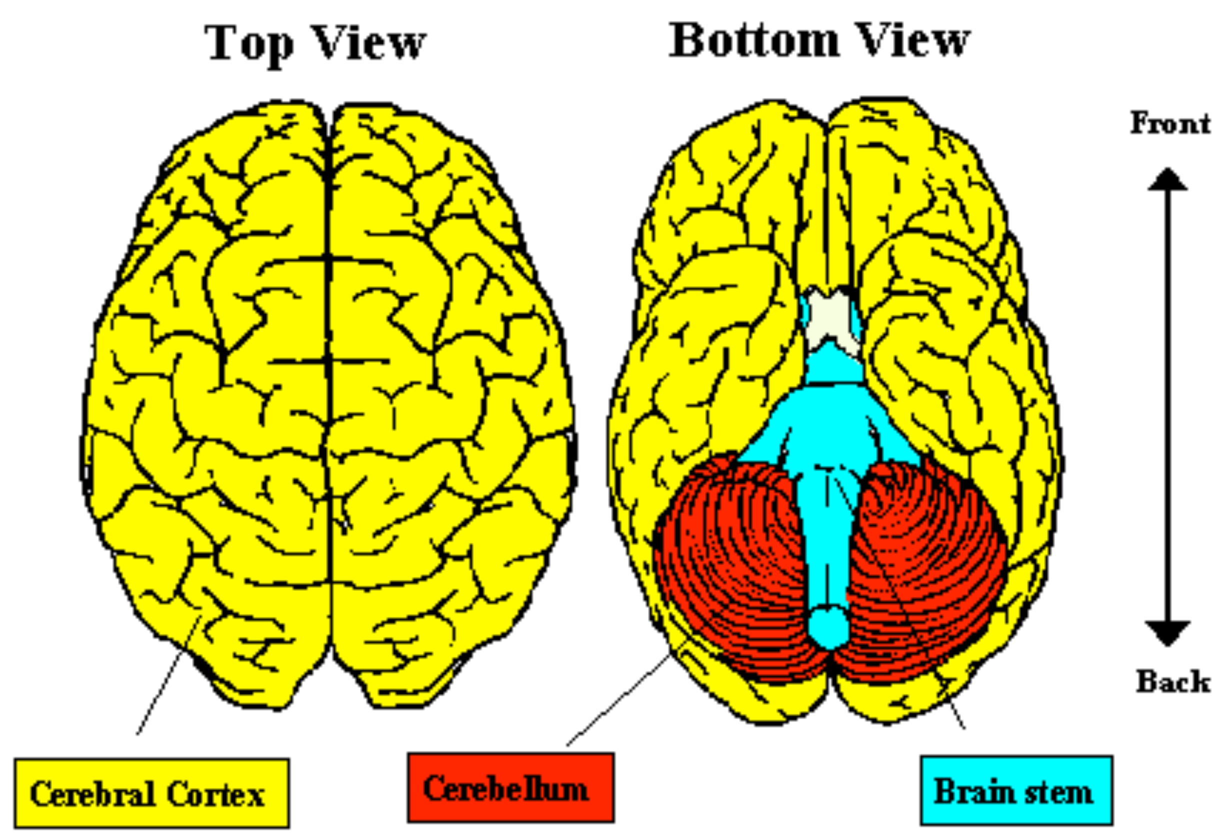
Hierarchy of Nervous System



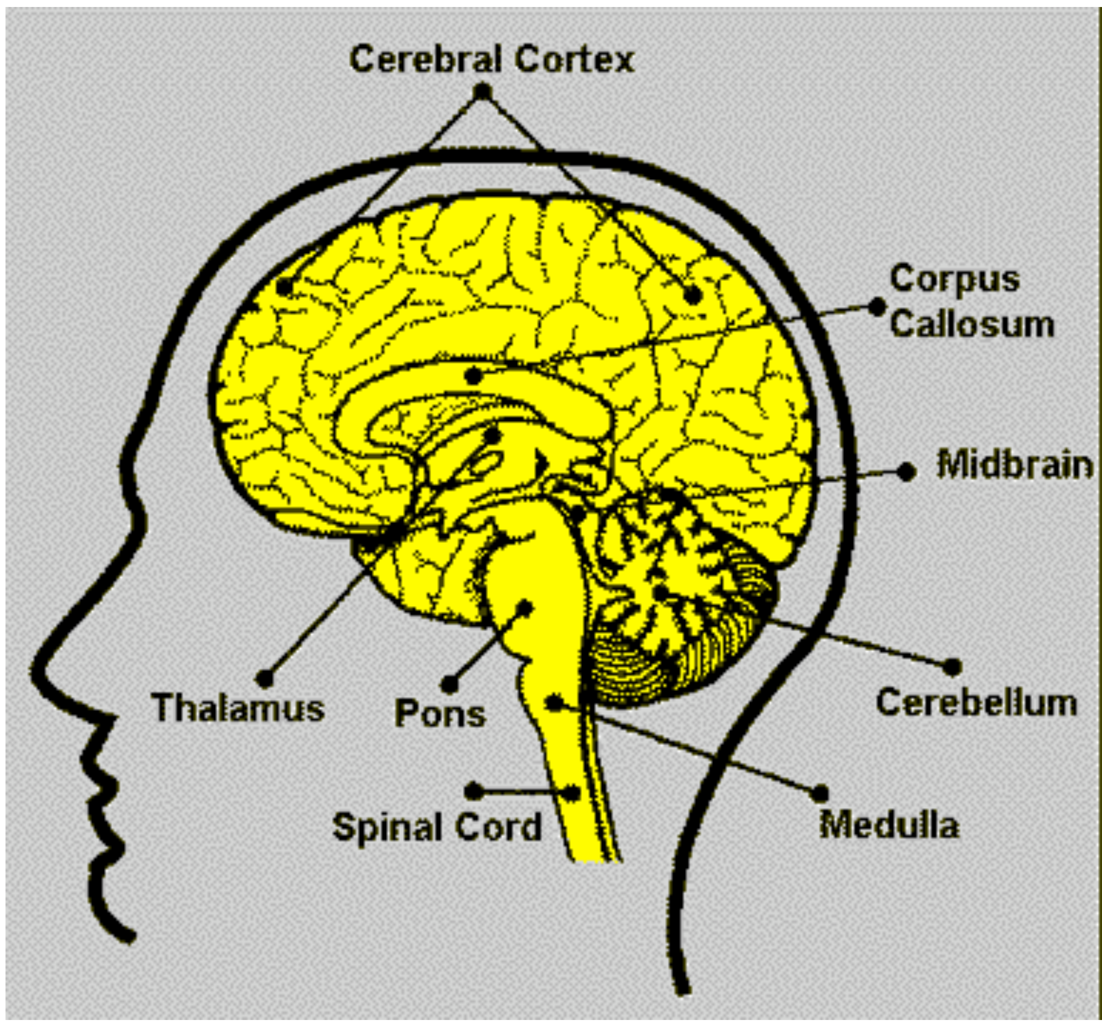
Views of the Brain

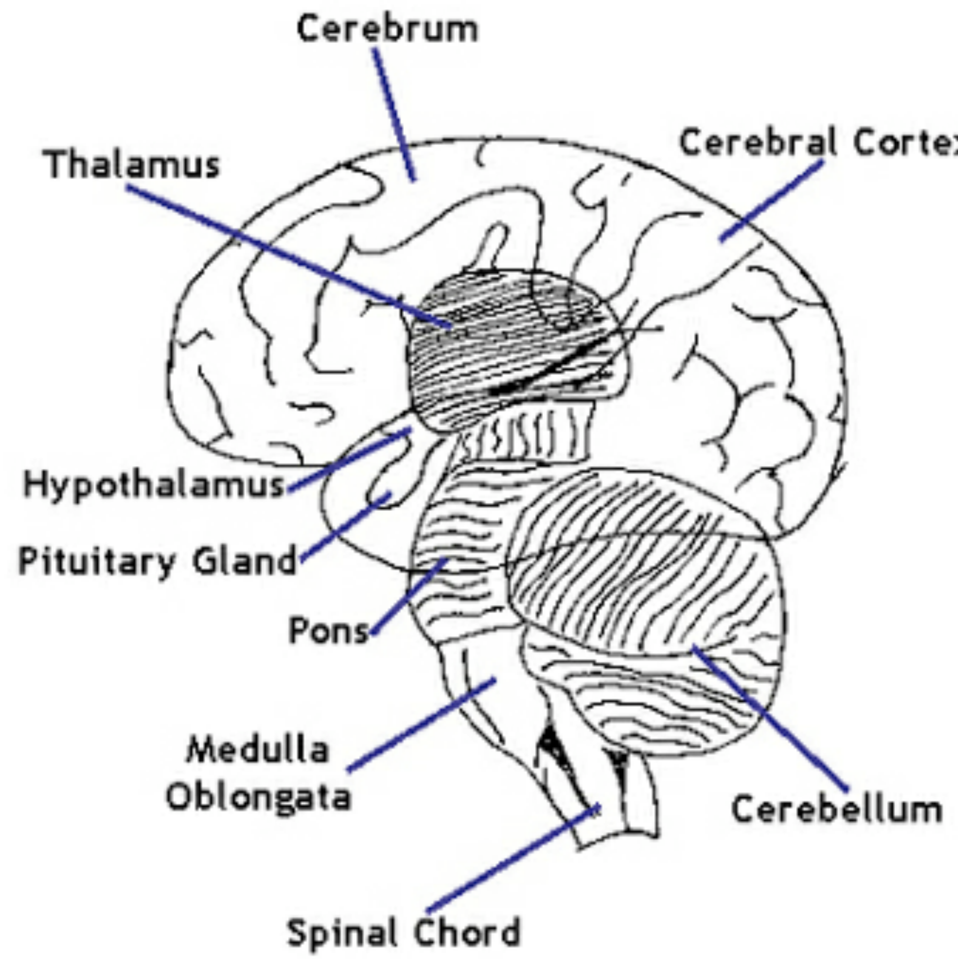
Lateral View

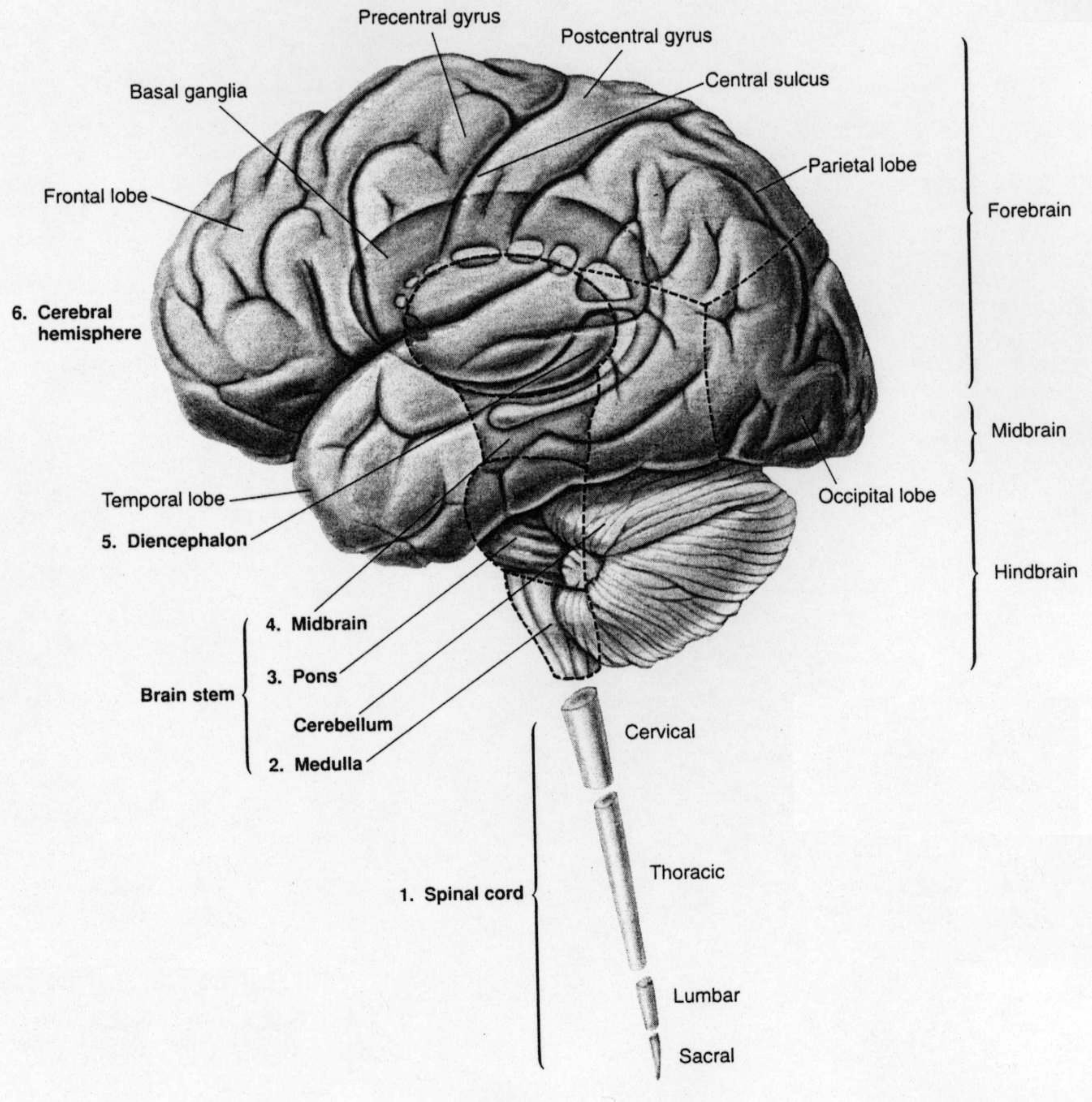


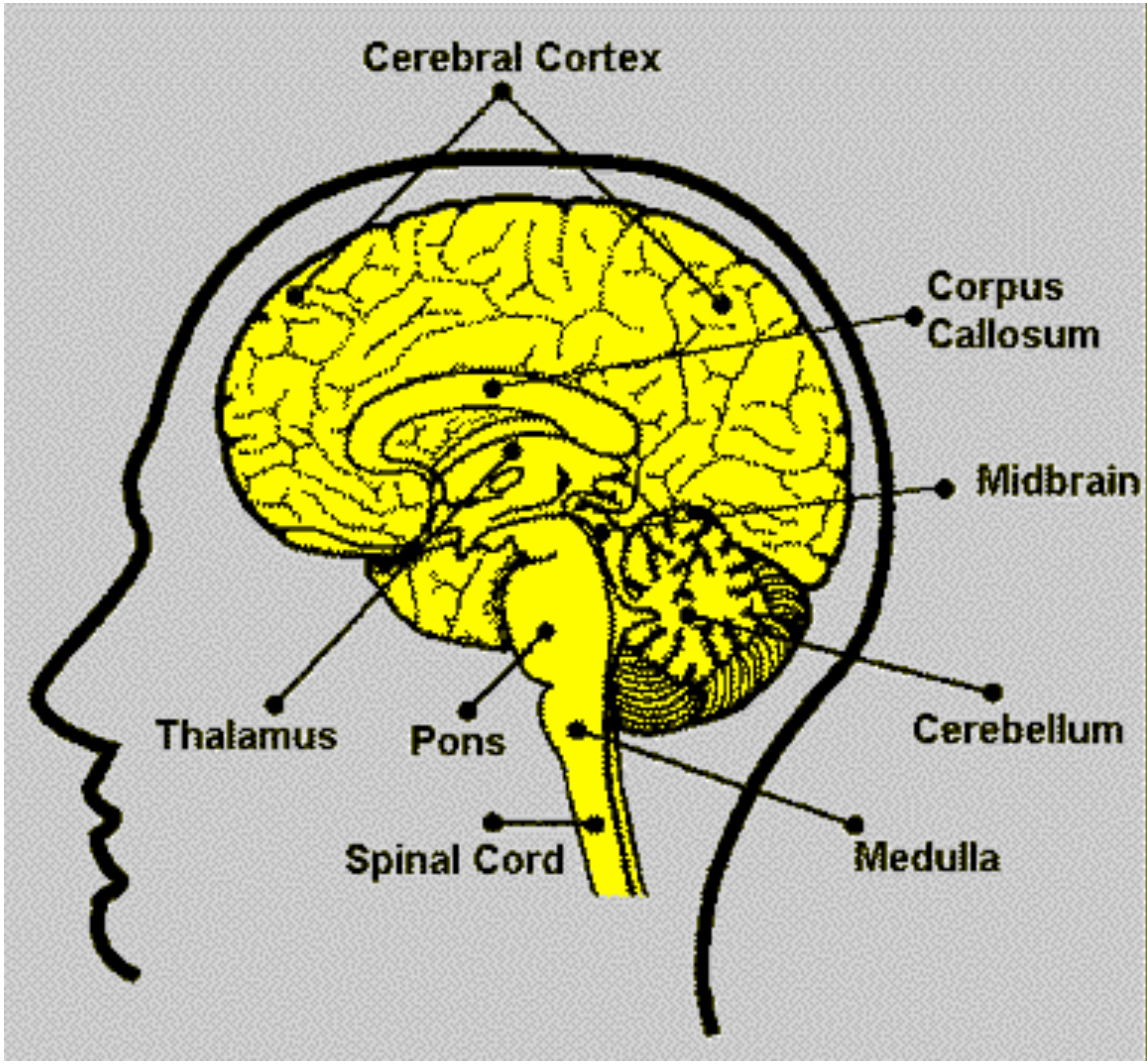


Parts in the Brain

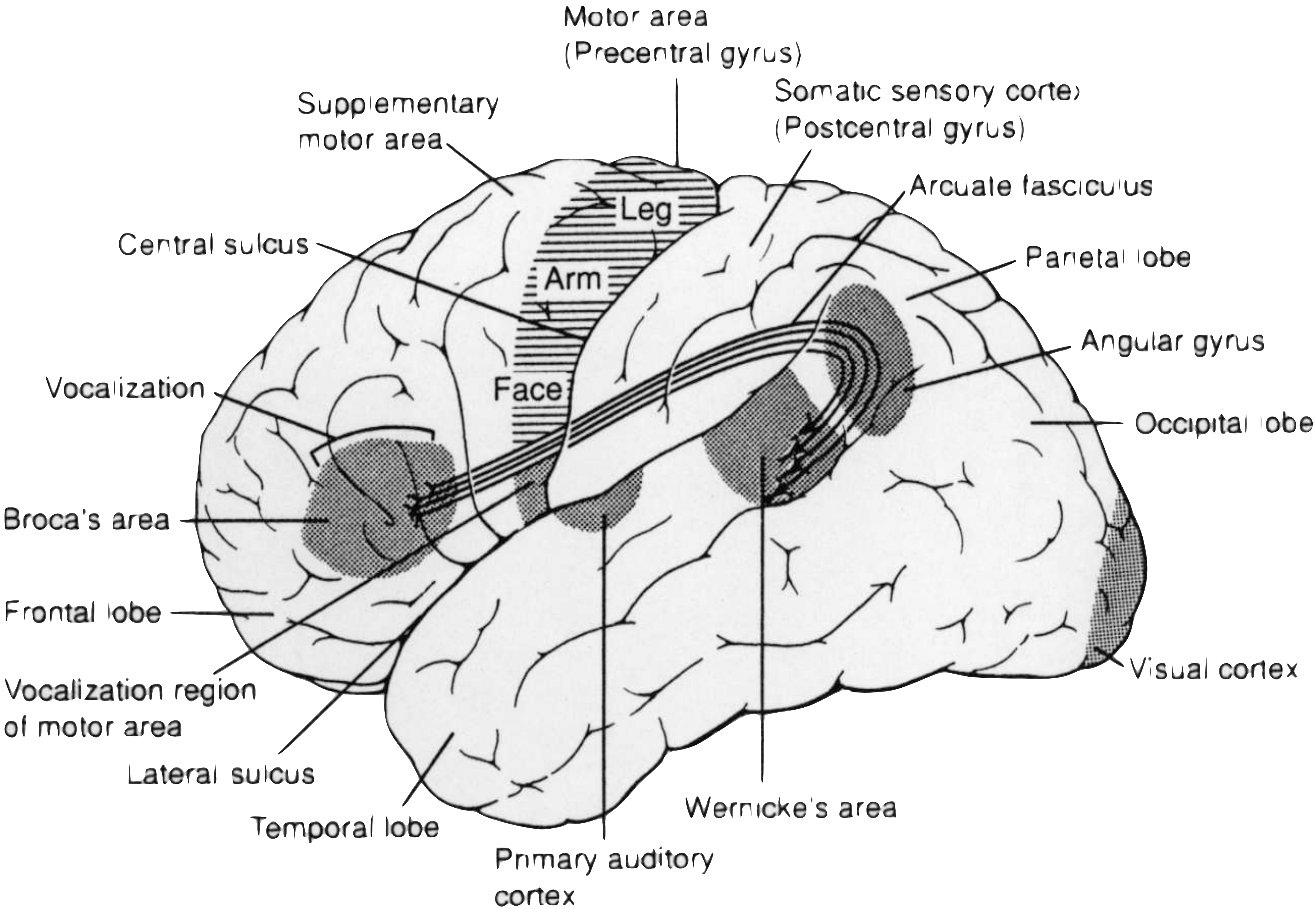




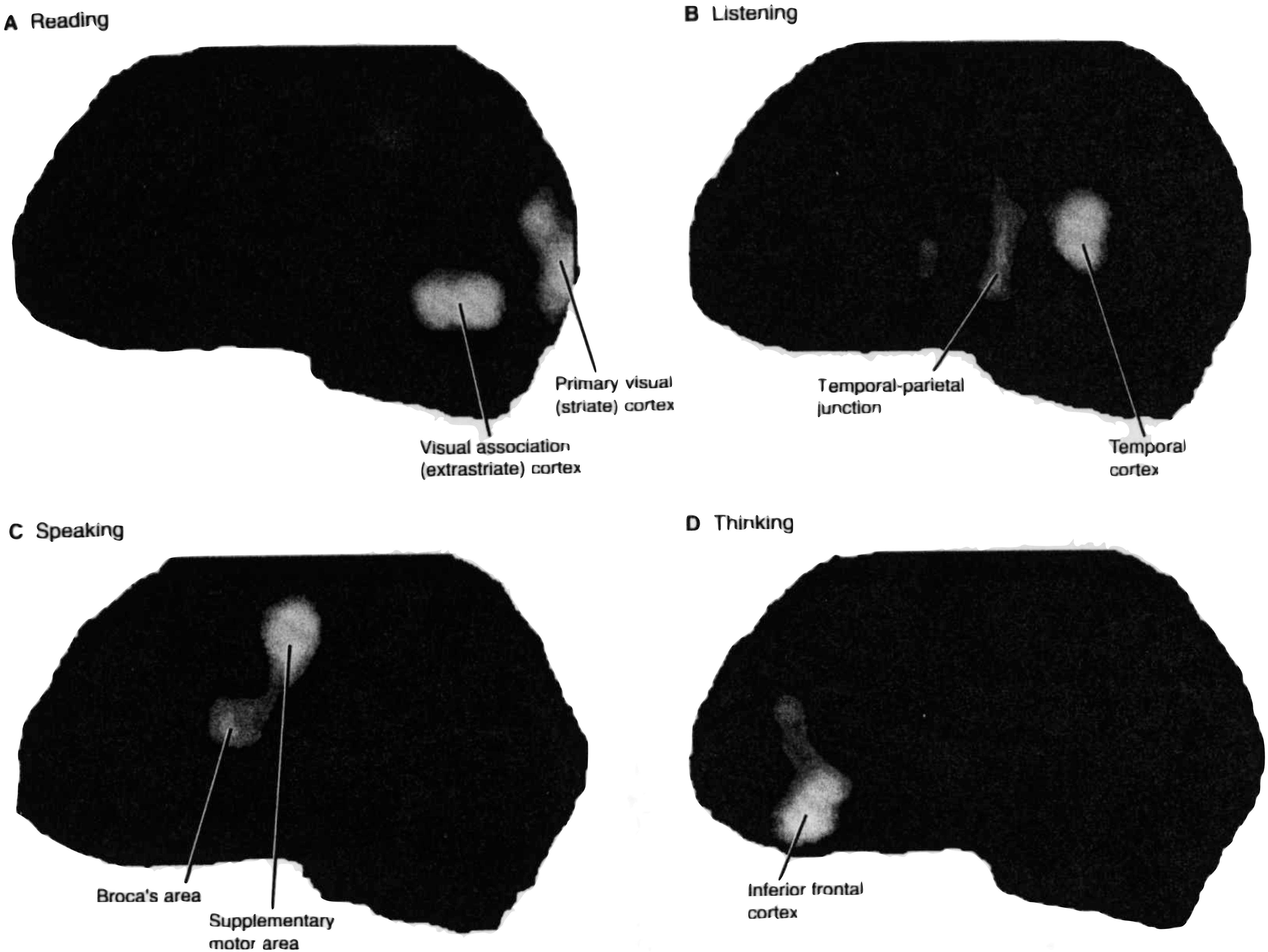

Precentral gyrus
Postcentral gyrus
Central sulcus
Basal ganglia
Parietal lobe
Frontal lobe
Forebrain
6. Cerebral hemisphere
Midbrain
Occipital lobe
Temporal lobe
5. Diencephalon
Hindbrain
4. Midbrain
3. Pons
Brain stem
Cerebellum
2. Medulla
Cervical
Thoracic
1. Spinal cord
Lumbar
Sacral



Cerebral Cortex
Corpus Callosum
Midbrain
Thalamus
Pons
Cerebellum
Spinal Cord
Medulla



Motor area
(Precentral gyrus)
Somatic sensory cortex
(Postcentral gyrus)
Supplementary
motor area
Arcuate fasciculus
Leg
Central sulcus
Parietal lobe
Arm
Angular gyrus
Vocalization
Face
Occipital lobe
Broca's area
Frontal lobe
Visual cortex
Vocalization region
of motor area
Lateral sulcus
Temporal lobe
Wernicke's area
Primary auditory
cortex


A Reading
Primary visual
(striate) cortex
Visual association
(extrastriate) cortex
B Listening
Temporal-parietal
junction
Temporal
cortex
C Speaking
Broca's area
Supplementary
motor area
D Thinking
Inferior frontal
cortex

Facts on the Brain

- The cerebral cortex in humans is a large flat sheet of neurons about 2 to 3 millimeters thick with a surface area of about 2,200 cm^2, about twice the area of a standard computer keyboard.
- The cerebral cortex contains about 10^{11} neurons, which is approximately the number of stars in the Milky Way.
- Each neuron is connected to 10^3 to 10^4 other neurons.

Facts on the Brain

- In total, the human brain contains approximately 10^{14} to 10^{15} interconnections.
- Neurons communicate through a very short train of pulses, typically milliseconds in duration.
- The message is modulated on the pulse-transmission frequency which can vary from a few to several hundred hertz.

Desirable Features in the Brain

- It is robust and fault tolerant.
- It is flexible.
- It can adapt and learn to a new environment.
- It can generalize.
- It is highly parallel.
- It can deal with information that is fuzzy, probabilistic, noisy, or inconsistent.
- It is small, compact, and dissipates very little power.
- It has distributed representation and computation.

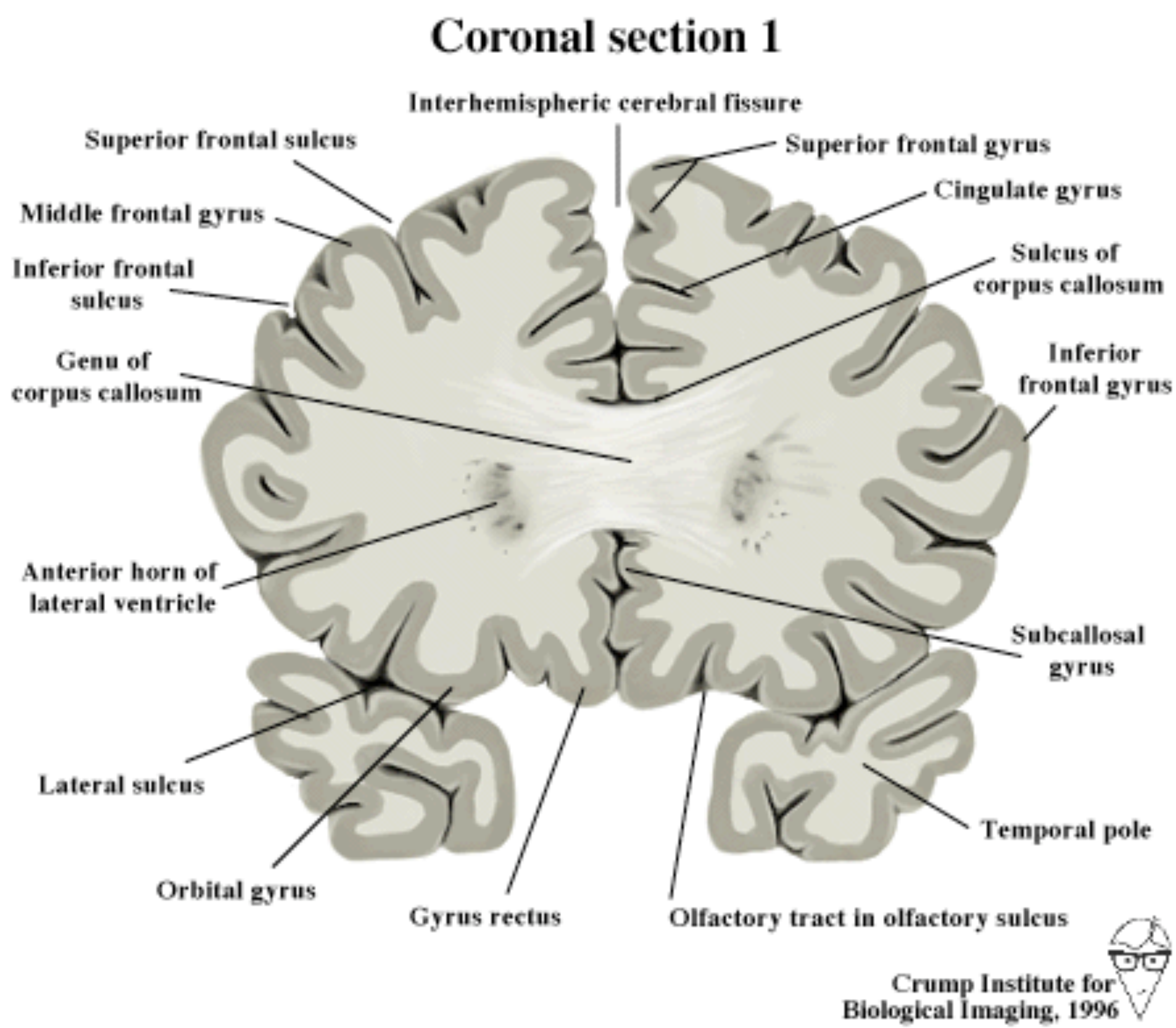
Digital vs. Biological

- Digital Computers
  - Digital (binary) &
  - More sequentially oriented
  - Operates in the nanosecond range
  - Minimize delays
  - Small amount of rigid memory
  - 10^6 number of elements
- Brain
  - Analog (continuous)
  - Highly parallel processes
  - Operates in the millisecond range
  - Uses delay for its advantages
  - Large amount of ``flexible" memory
  - 10^{11} number of elements

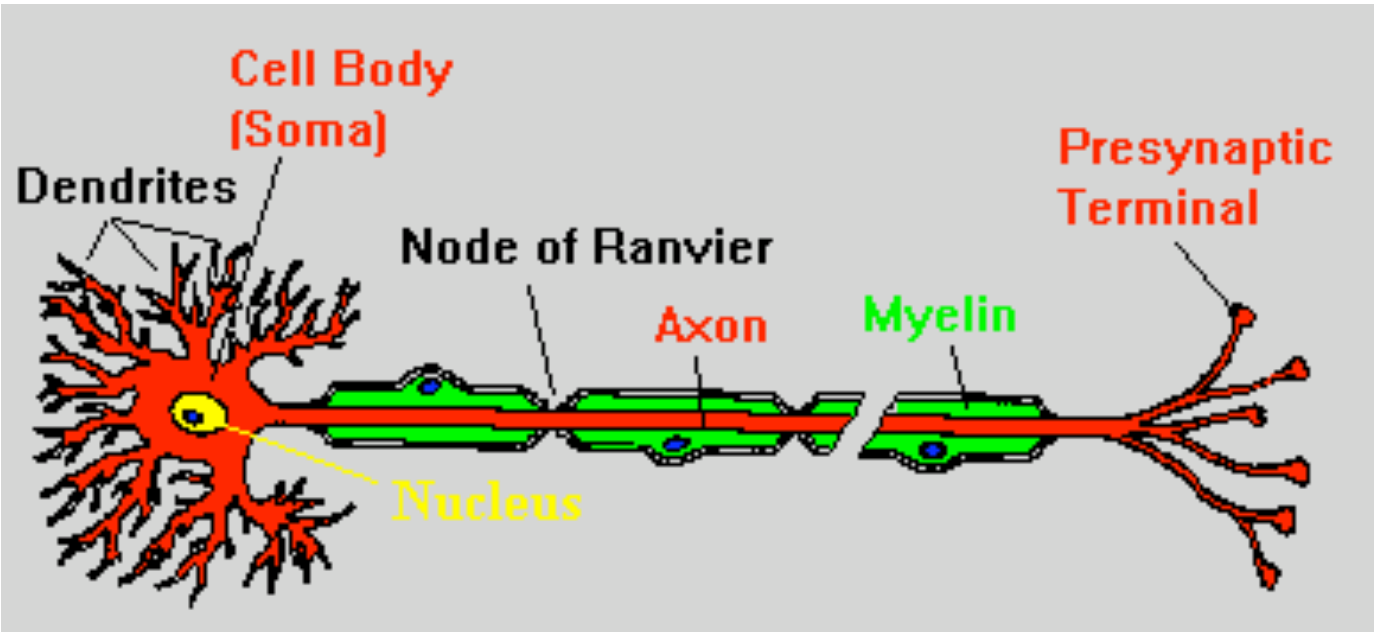
Digital vs. Biological

- Digital Computers
  - 10^2 fan-in and -out factor
  - Highly structural and precise
  - Precise Logic
  - Uses external algorithms
  - Do not handle failures well
- Brain
  - 10^5 fan-in and -out factor
  - Less structural and less precise
  - Fuzzy Logic
  - Algorithms are built within
  - Graceful Degradation

Brain Slice


Coronal section 1
Interhemispheric cerebral fissure
Superior frontal sulcus
Superior frontal gyrus
Cingulate gyrus
Middle frontal gyrus
Inferior frontal sulcus
Sulcus of corpus callosum
Genu of corpus callosum
Inferior frontal gyrus
Anterior horn of lateral ventricle
Subcallosal gyrus
Lateral sulcus
Temporal pole
Orbital gyrus
Gyrus rectus
Olfactory tract in olfactory sulcus
Crump Institute for Biological Imaging, 1996

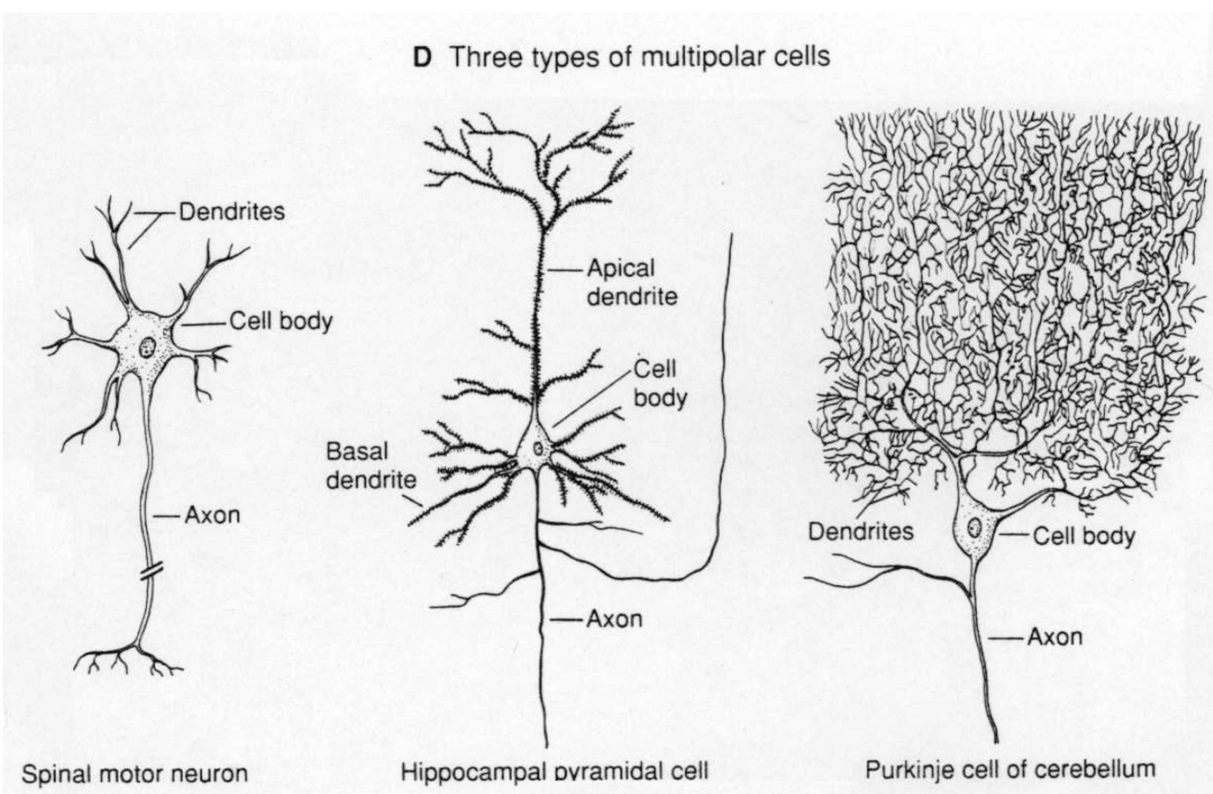
Cells



Cell Body
(Soma)
Dendrites
Node of Ranvier
Axon
Myelin
Presynaptic
Terminal
Nucleus

Bipolar Neuron

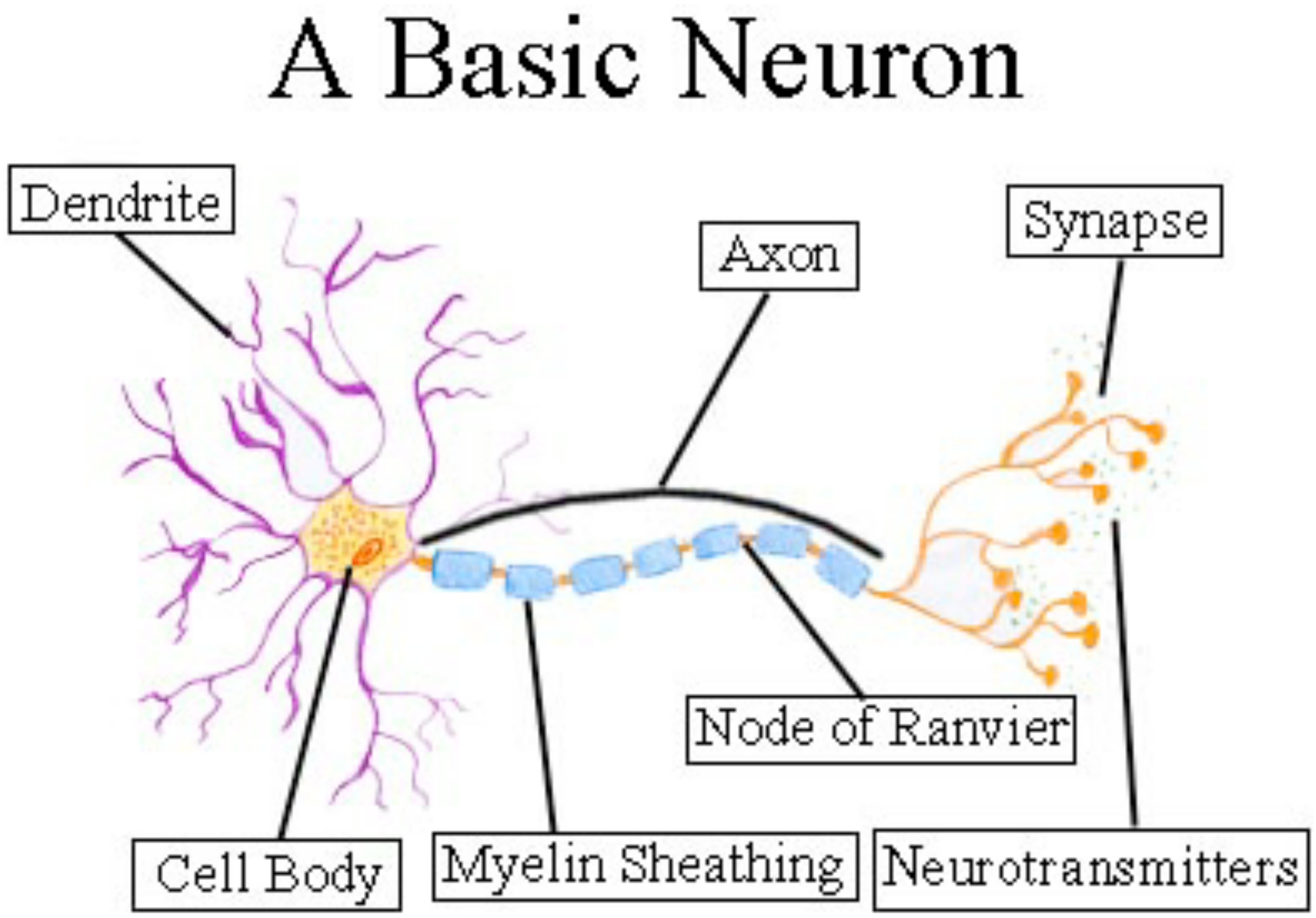
Multipolar Neuron

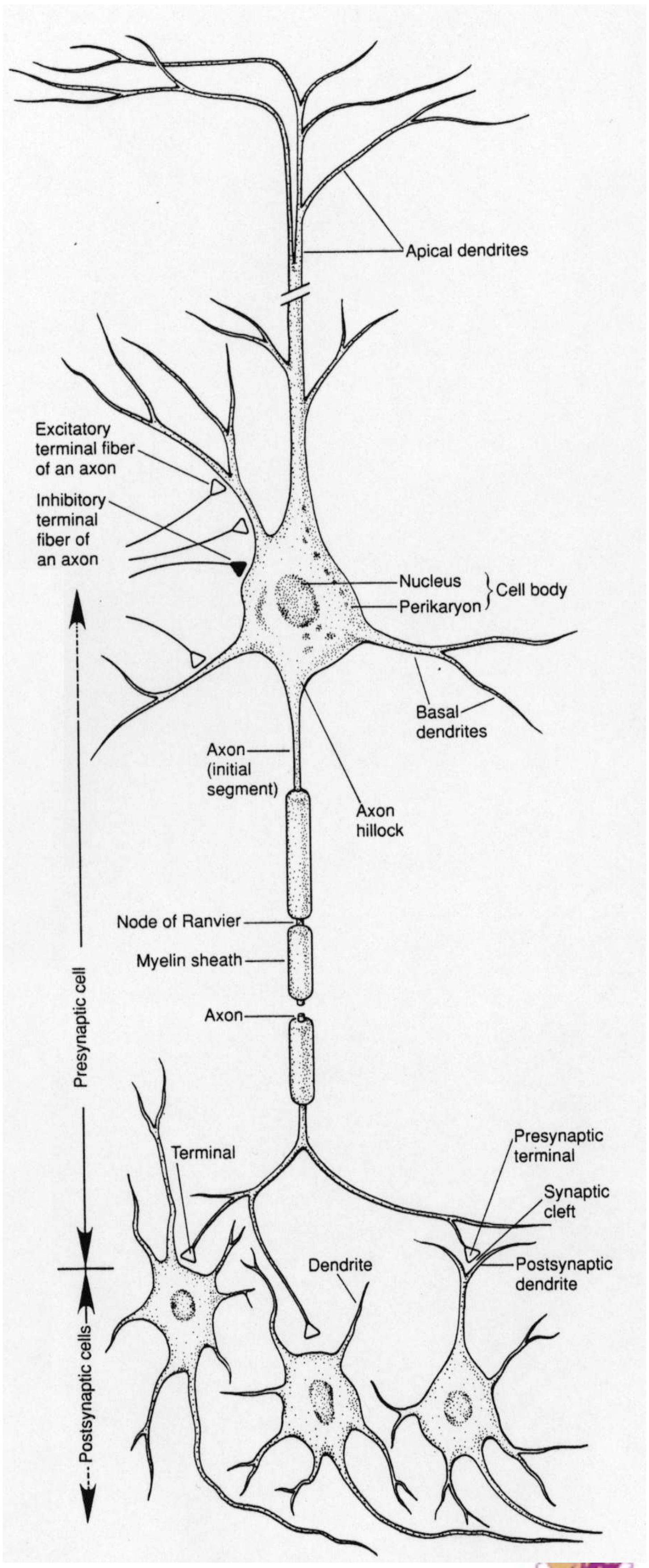


D Three types of multipolar cells
Dendrites
Cell body
Axon
Apical
dendrite
Cell
body
Basal
dendrite
Axon
Dendrites
Cell body
Axon
Spinal motor neuron
Hippocampal pyramidal cell
Purkinje cell of cerebellum

Pseudounipolar Neuron

Basic Neuron





Neuron - Terms

- Cell body (soma)
- Dendrites: out-reaching tree-like branches
- Axons: out-reaching tree-like branches
- Synaptic junctions (synapses): it is an elementary structure and functional unit between two neurons (an axon strand of one neuron and a dendrite of another)

Neuron - Terms

- Neurotransmitters : when the impulse reaches the synapse's terminal, certain chemicals called neurotransmitters are released. The neurotransmitters diffuse across the synaptic gap, to enhance or inhibit, depending on the type of the synapse, the receptor neuron's own tendency to emit electrical impulses.
- Action potential
- Refractory period

Axons vs. Dendrites

- Axons
  - Take information away from the cell body
  - Smooth Surface
  - Generally only 1 axon per cell
  - No ribosomes
  - Can have myelin
  - Branch further from the cell body
- Dendrites
  - Bring information to the cell body
  - Rough Surface (dendritic spines)
  - Usually many dendrites per cell
    Have ribosomes
  - No myelin insulation
    Branch near the cell body

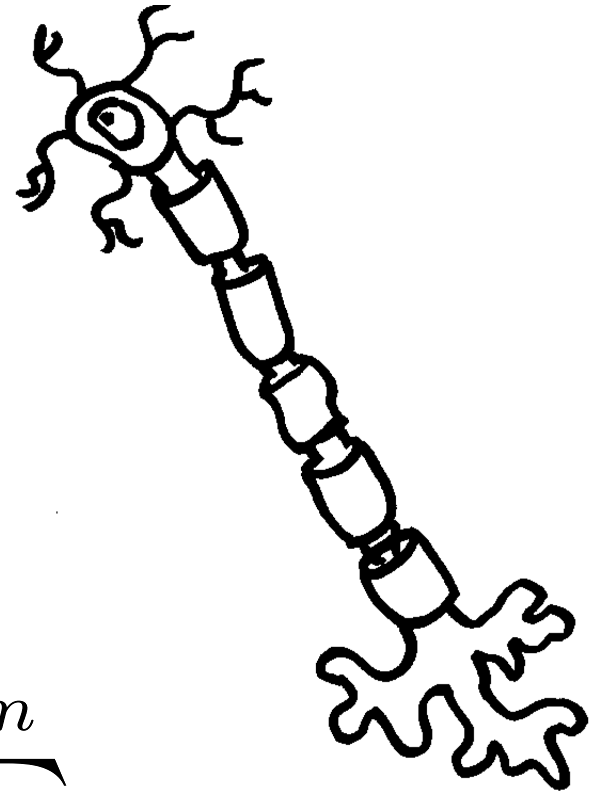
Different Levels of Modeling

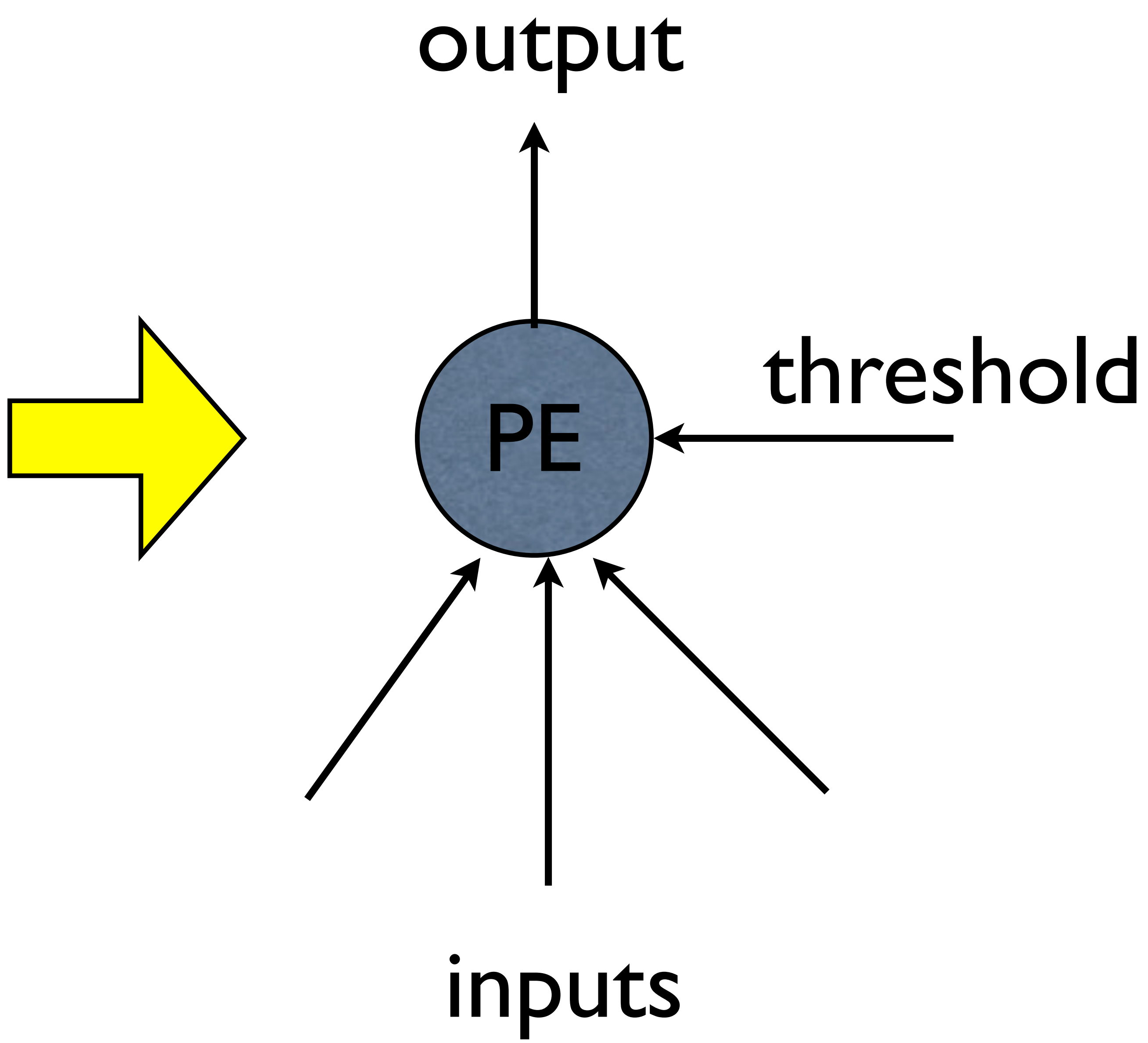
- Atomic
- Molecule
- Neuron
- Network
- Organizational
- System

- Neuronal-level (decoupled) models
- Aggregate models
- Network-level models
- Nervous-system level (organizational) models
- Mental-operation level models

Idealization of a Neuron

- McCulloch and Pitts Model

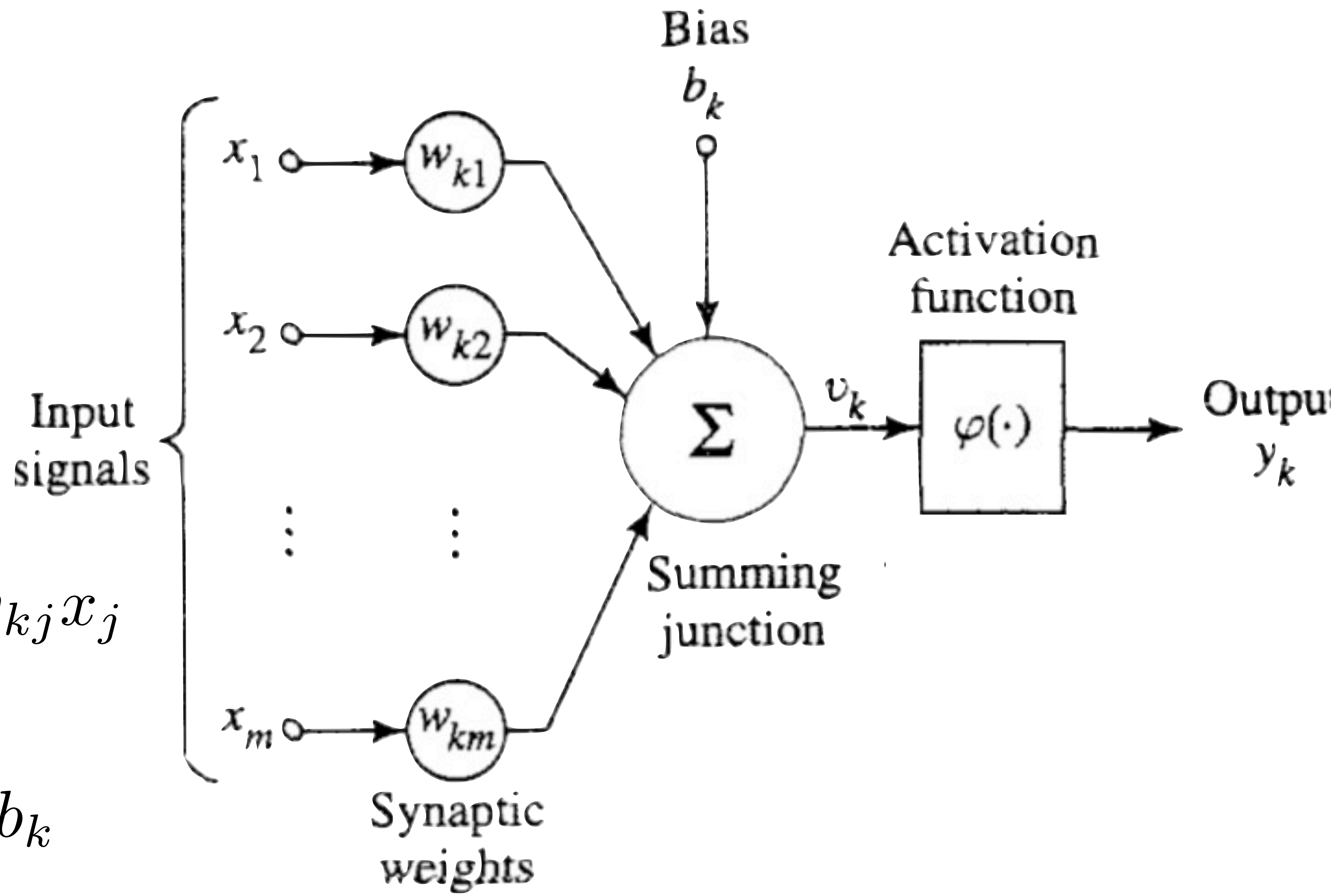




$$u_k = \sum_{j=1}^{m} w_{kj} x_j$$

$$y_k = \varphi(u_k + b_k)$$

Nonlinear Neuron Model

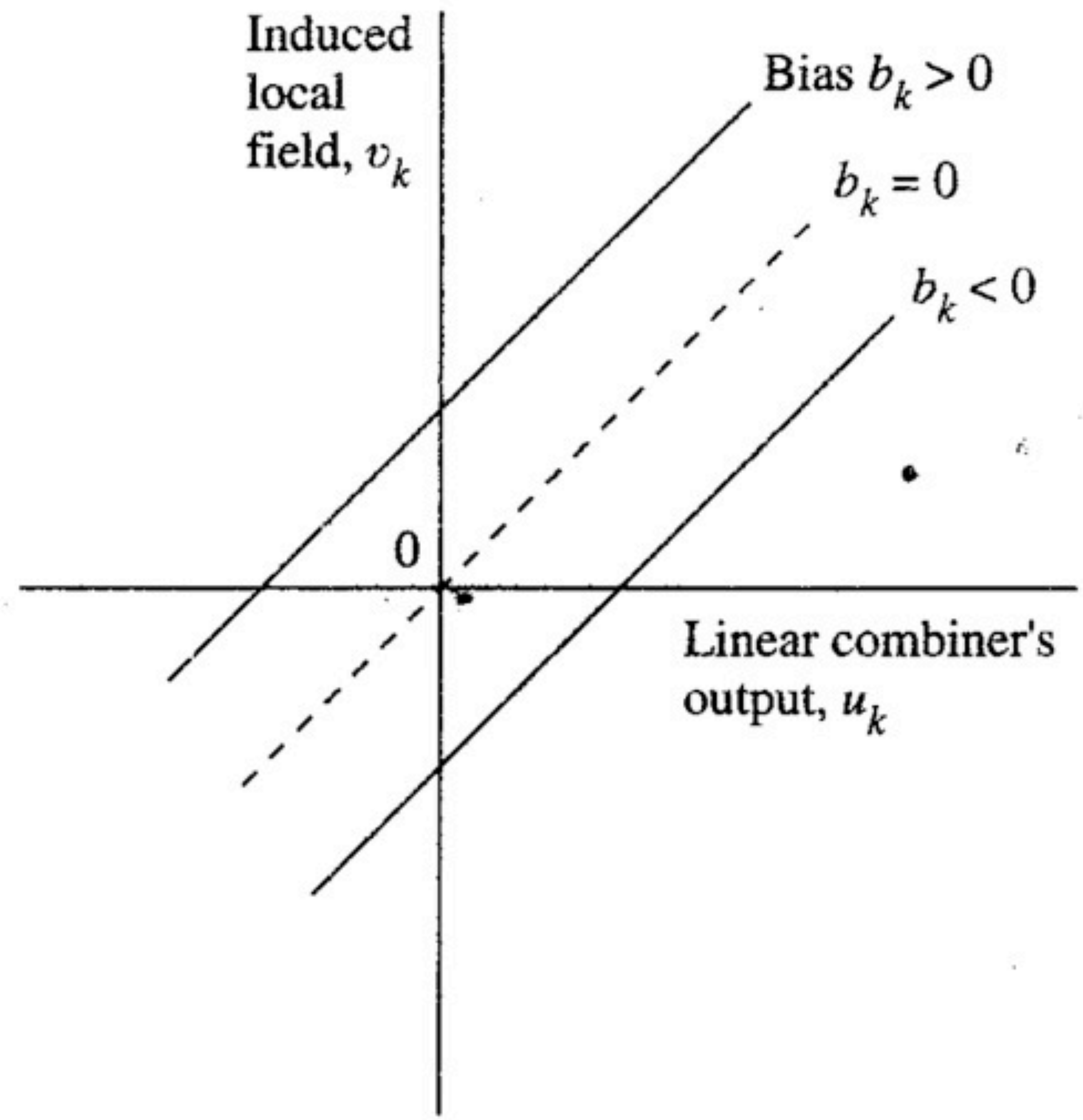


$$u_k = \sum_{j=1}^{m} w_{kj} x_j$$

$$v_k = u_k + b_k$$

$$y_k = \varphi(u_k + b_k)$$

Biased Neuron



$$u_k = \sum_{j=0}^{m} w_{kj} x_j$$

$$y_k = \varphi(v_k)$$

$$x_0 = +1$$

$$w_{k0} = b_k$$

Aspect of Neural Networks I

- A set of processing units
- A state of activation
- An output function for each unit
- A pattern of connectivity among units
- A propagation rule for propagating patterns of activities through the network of connectivities

Aspect of Neural Networks II

- An activation rule for combining the inputs impinging on a unit with the current state of that unit to produce a new level of activation for the unit.
- A learning rule whereby patterns of connectivity are modified by experience.
- An environment within which the system must operate.

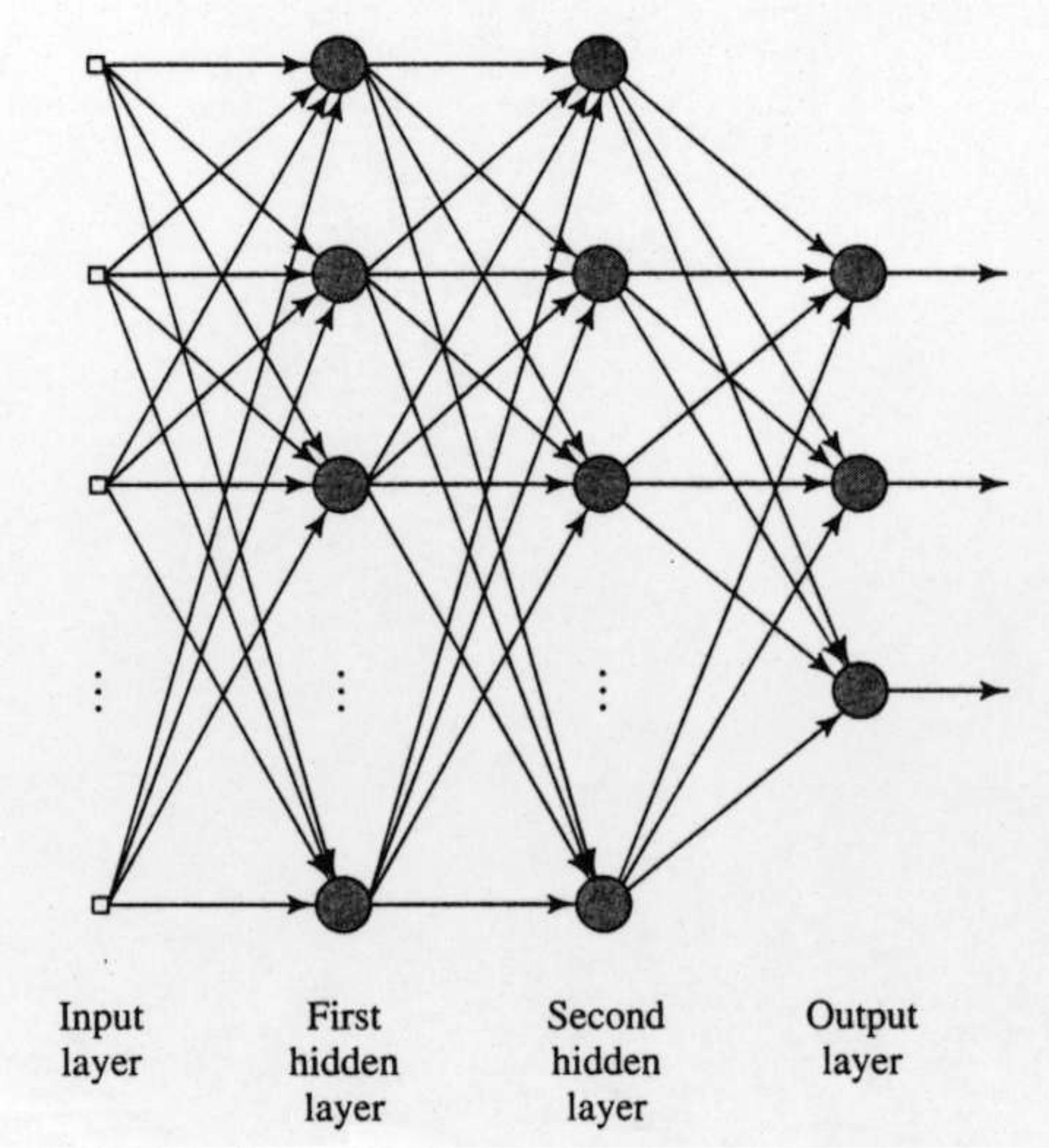
Network Architecture I

- Feedforward
  - Single-layer perceptron
  - Multilayer perceptron
  - Radial Basis Function nets
- Feedback/Recurrent
  - Competitive networks
  - Kohonen's SOM
  - Hopfield network
  - ART models
- Types of Timing Signals in ANN
  - Continuous
  - Discrete

Network Architecture II

- Single-Layer Feedforward Networks - input layer of source nodes projects into output layer of computation nodes (neurons).
- Multi-layer Feedforward Networks - presence of one or more hidden layers. Can be fully connected or partially connected.
- Recurrent Networks - has one or more feedback loops, that can originate form the hidden or output neurons.
- Lattice Structures - 1, 2, or higher-dimensional array of neurons with corresponding set of source nodes.
- Competitive Learning Networks - Hybrid, where feedforward structure contains at least one layer with intralayer recurrence (nodes connected to themselves via excitatory weights).

Example


Input layer
First hidden layer
Second hidden layer
Output layer

Learning Defined

- Learning is a process by which the free parameters of a neural network are adapted through a continuing process of stimulation by the environment in which the network is embedded.
- The type of learning is determined by the manner in which the parameter changes take place.
- Types of Learning
  - Supervised Learning - Perceptron
  - Unsupervised Learning (self-organization) - Competitive Learning
  - Reinforcement Learning
  - Hybrid Learning (combining various learning into one integral model)

Taxonomy of Learning Techniques I

- Preprogramming - Fixed weights make use of all vectors at once, e.g., Hopfield model
- Error-Correction Learning - utilizes error signals (between input & target response) to minimize statistical cost function.
- Hebbian Learning - if 2 neurons on either side of synapse activated simultaneously, strength of synapse increased.
- Competitive Learning - output neurons of NN compete among themselves for being the only active one being fired.

Taxonomy of Learning Techniques II

- Boltzman Learning - uses recurrent structure of binary neurons
- Reinforcement Learning - on-line learning of I/O mapping through trial & error to maximize a performance index.
- Supervised Learning - training vector & external teacher used to provide desired response to network, e.g., Back-Propagation
- Unsupervised Learning - network recognizes new statistically similar classes of data without explicit training.

Applications

- Pattern Classification
- Clustering and Categorization
- Function Approximation
- Prediction and Forecasting
- Optimization
- Association and Content-addressable Memory
- Control

Type of Thresholds

- Threshold Function

$$\varphi(v) = \begin{cases} 1, & v \geq 0 \\ 0, & v < 0 \end{cases}$$

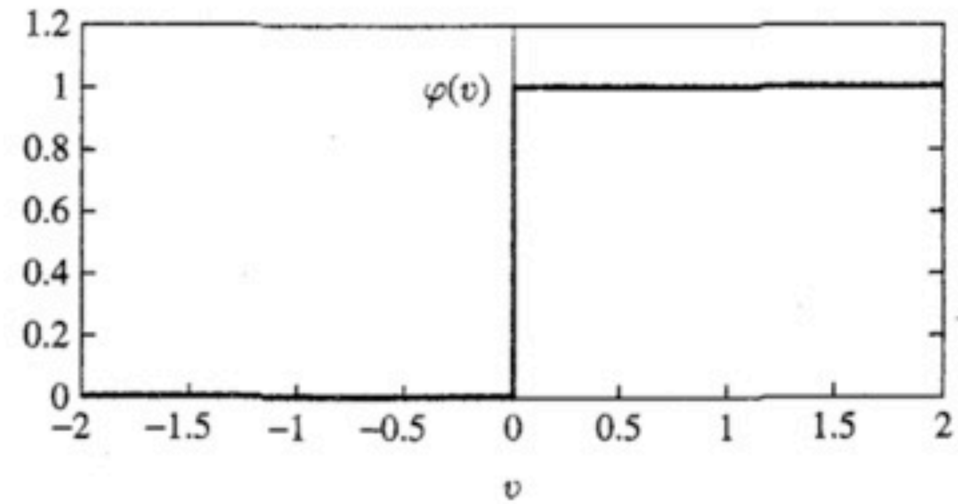
- Piecewise-Linear Function

$$\varphi(v) = \begin{cases} 1, & v \geq +\frac{1}{2} \\ v, & +\frac{1}{2} > v > -\frac{1}{2} \\ 0, & v < -\frac{1}{2} \end{cases}$$

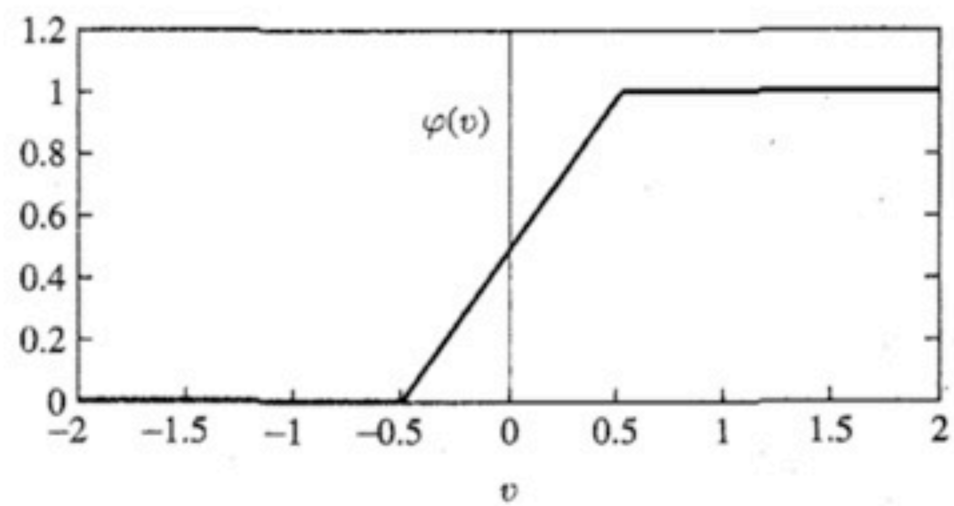
- Sigmoid Function

$$\varphi(v) = \frac{1}{1 + \exp(-av)}$$

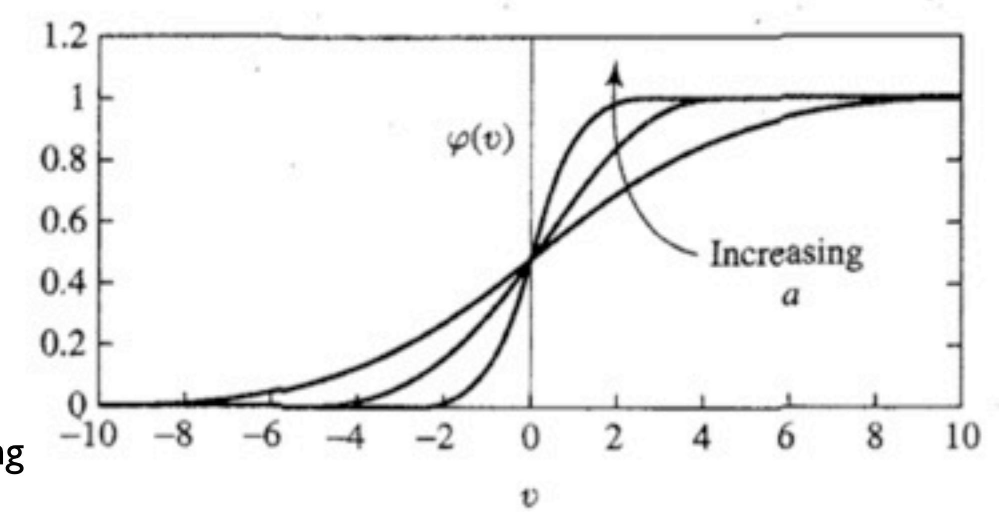
$$\varphi(v) = \tanh(v)$$



(a)



(b)



Example of Learning

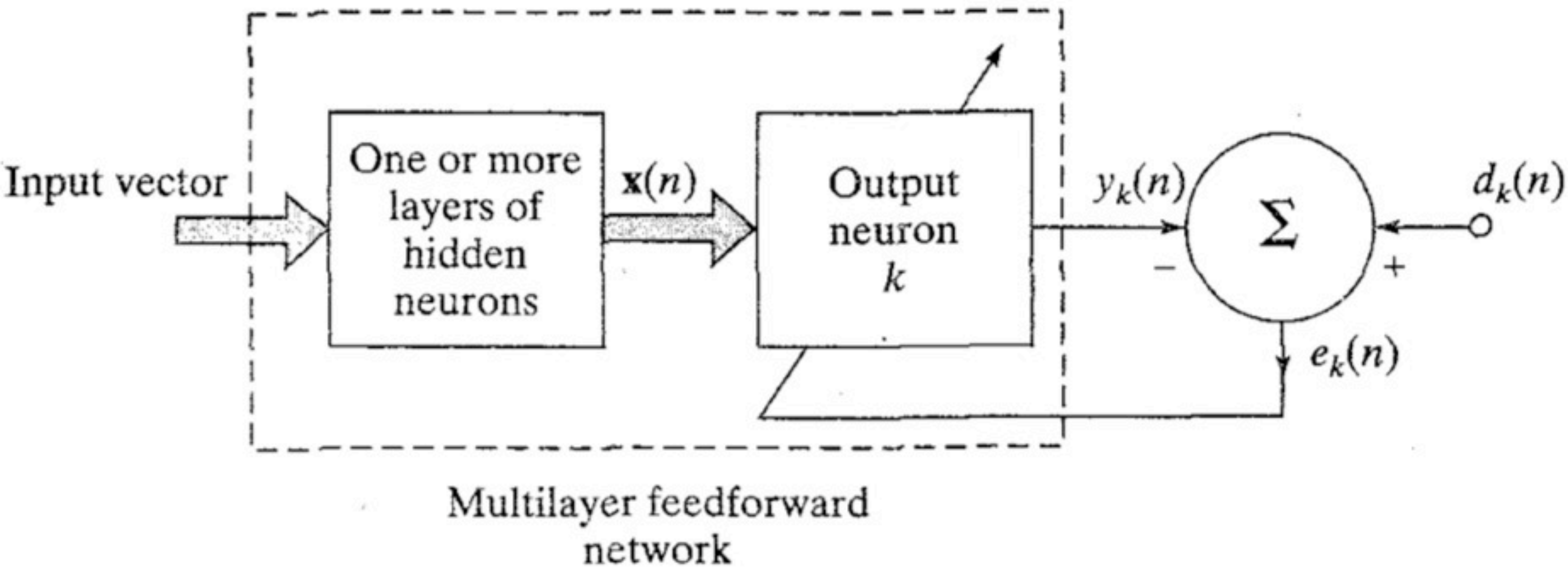
- Guessing the passing mark
  - 100 p
  - 30 f
  - 40 f
  - 60 p
  - 90 p
  - 50 f
  - 30 f

- Guessing the passing mark
- 100 p
- 30 f
- 40 f
- 60 p
- 90 p
- 65 f
- 30 f

Example of Learning

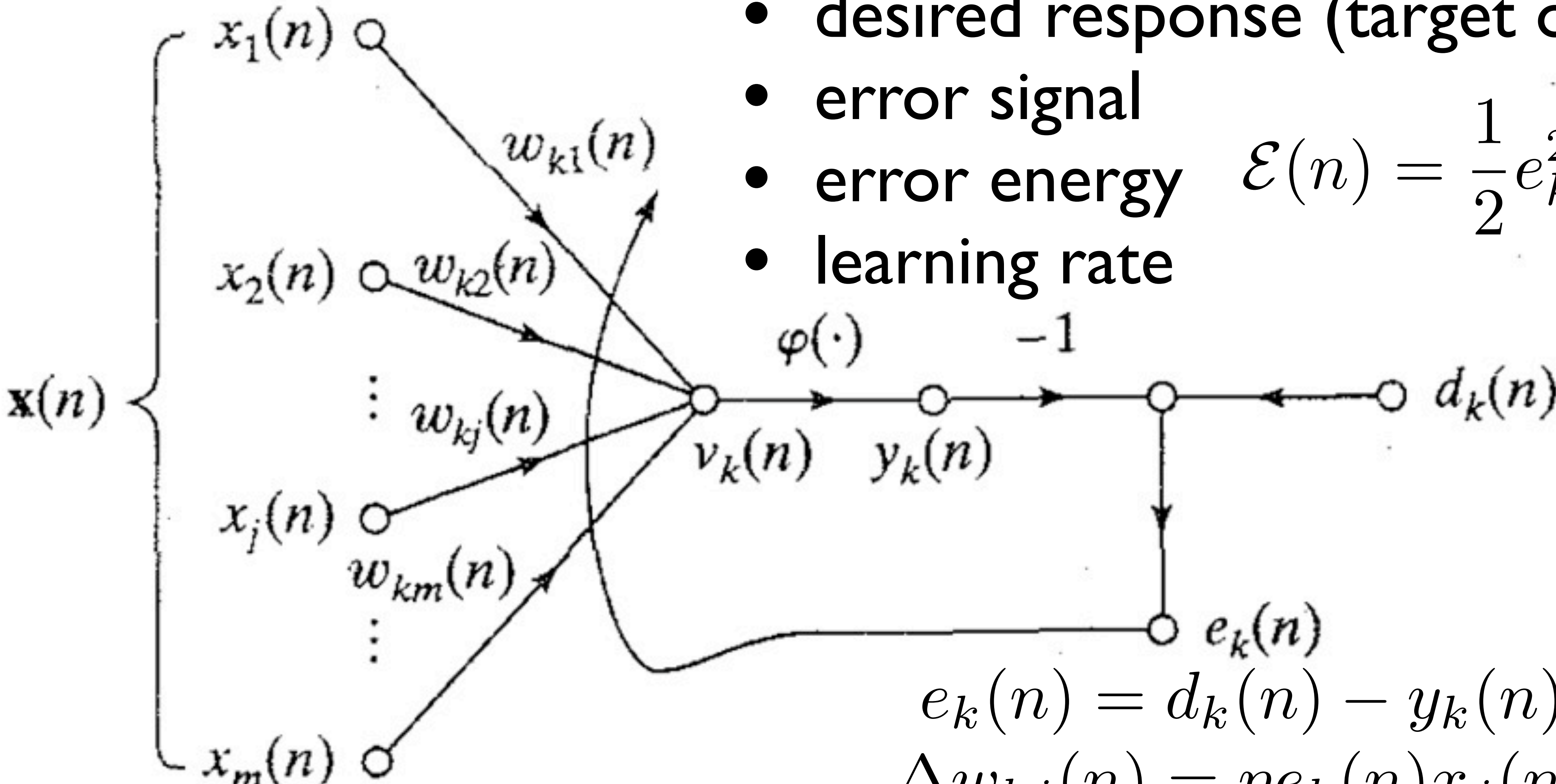
```
(60,1),(40,0),(90,1),(50,0)
x old w    y     d     e   eta    dw     w
==========================
60    45     1     1     0     1     0    45
40    45     0     0     0     1     0    45
90    45     1     1     0     1     0    45
50    45     1     0    -1     1    -1    46
60    46     1     1     0     1     0    46
40    46     0     0     0     1     0    46
90    46     1     1     0     1     0    46
50    46     1     0    -1     1    -1    47
60    47     1     1     0     1     0    47
40    47     0     0     0     1     0    47
90    47     1     1     0     1     0    47
50    47     1     0    -1     1    -1    48
90    47     1     1     0     1     0    47
50    47     1     0    -1     1    -1    48
60    48     1     1     0     1     0    48
40    48     0     0     0     1     0    48
90    48     1     1     0     1     0    48
50    48     1     0    -1     1    -1    49
60    49     1     1     0     1     0    49
40    49     0     0     0     1     0    49
90    49     1     1     0     1     0    49
50    49     1     0    -1     1    -1    50
60    50     1     1     0     1     0    50
40    50     0     0     0     1     0    50
90    50     1     1     0     1     0    50
50    50     0     0     0     1     0    50
```


Error-Correction Learning



Error-Correction Learning

- signal vector
- output signal
- desired response (target output)
- error signal
- error energy $\mathcal{E}(n) = \frac{1}{2}e_k^2(n)$
- learning rate



$$e_k(n) = d_k(n) - y_k(n)$$
$$\Delta w_{kj}(n) = \eta e_k(n) x_j(n)$$
$$w_{kj}(n+1) = w_{kj}(n) + \Delta w_{kj}(n)$$

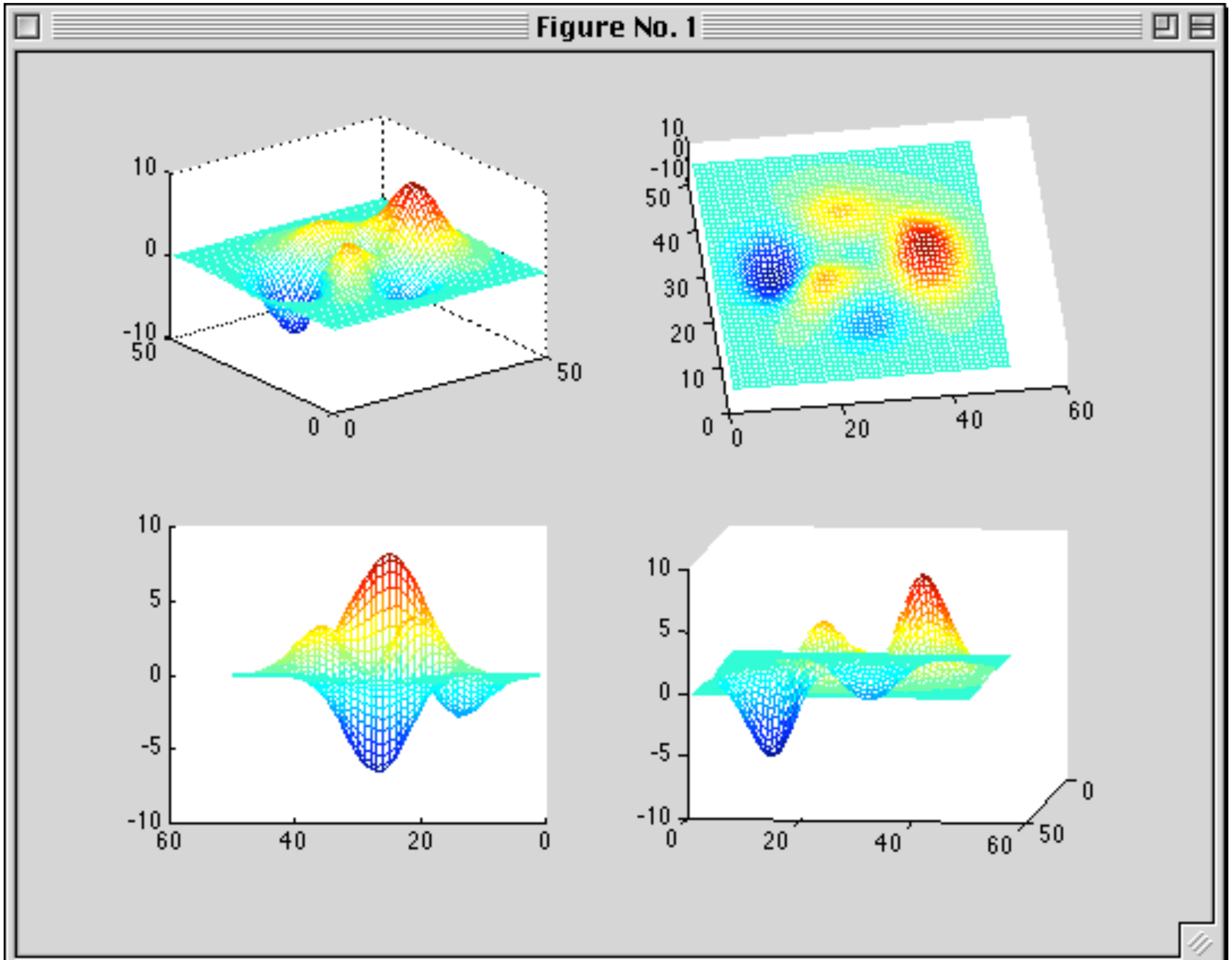
Notes

- The error signal is the difference between the target response and the actual response.
- The goal is to minimize a cost function based on the error signal so that the actual response of each output neuron approaches the target response in some statistical sense.
- The learning-rate parameter \eta is important.
  - If \eta is small, the learning proceeds smoothly, but it may take a long time for the system to converge to a stable solution.
  - If \eta is large, the rate of learning is accelerated, but the learning process may diverge and the system may become unstable.

Error Surface

- A plot of the cost function *J* versus *w* is a multidimensional surface referred to as an error-performance surface or simply error surface.
- This error surface is used to seek out minimum (maximum) states of the system.
- With linear processing units, the error surface is exactly a quadratic function of the weights in the network. The error surface is bowl-shaped with a unique minimum point.
- With nonlinear processing units, the error surface has a global minimum (perhaps multiple global minima) as well as local minima.

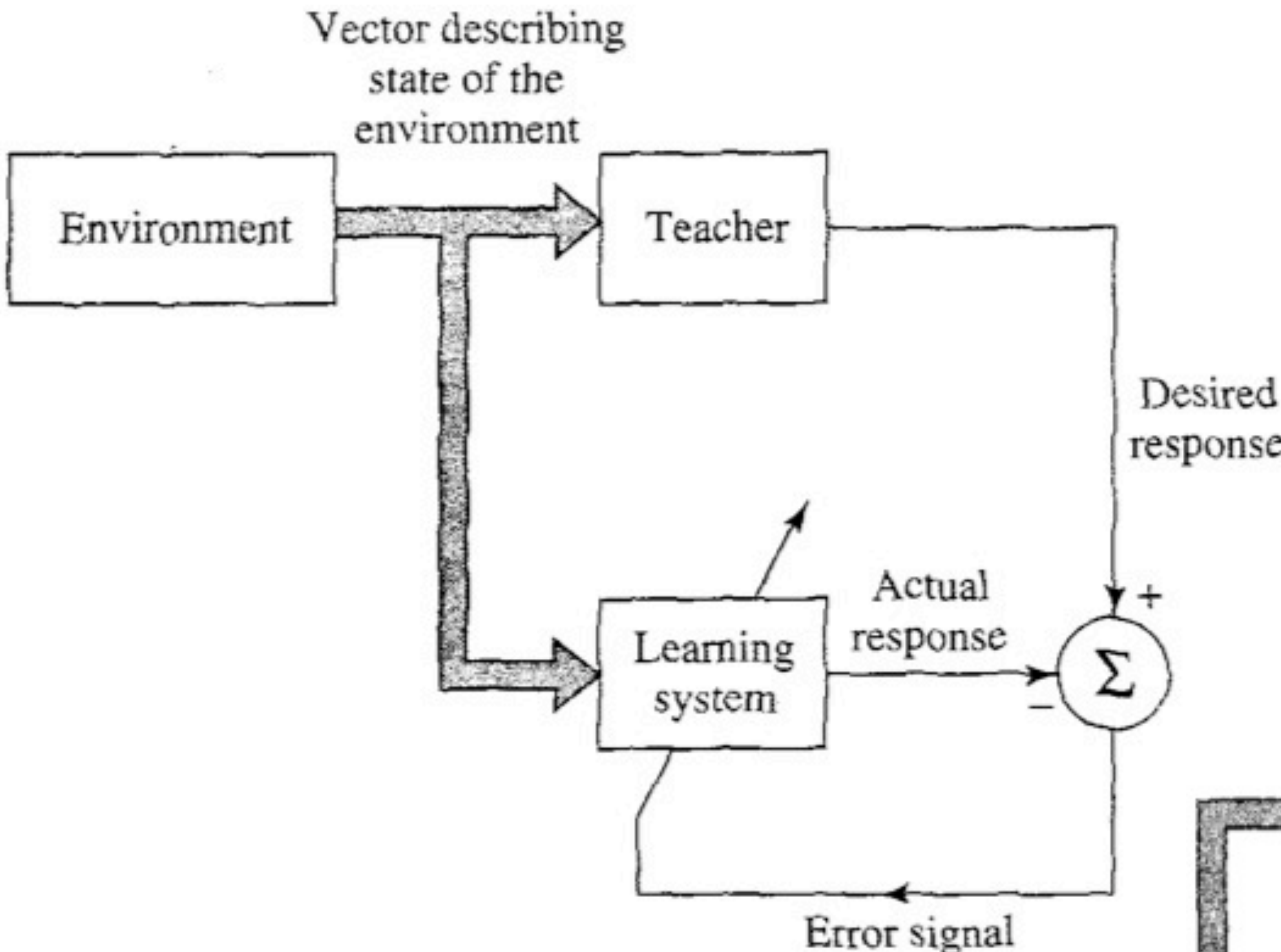
Error Surface Example



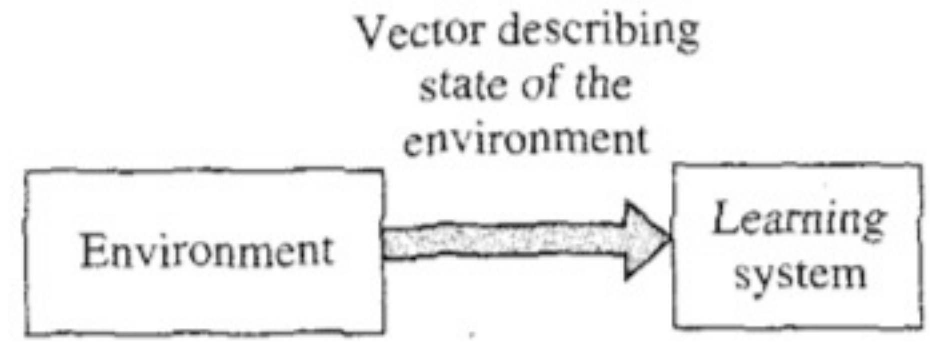
Hebbian Learning

- Hebb's postulate of learning is the oldest and most famous of all learning rules.
  - When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic changes take place in one or both cells such that A's efficiency as one of the cells firing B, is increased.
- If two neurons on either side of a synapse (connection) are activated simultaneously (i.e., synchronously), then the strength of that synapse is selectively increased.
- If two neurons on either side of a synapse are activated asynchronously, then that synapse is selectively weakened or eliminated.
- A Hebbian Synapse is a synapse that uses a time-dependent, highly local, and strongly interactive mechanism to increase synaptic efficiency as a function of the correlation between the presynaptic and postsynaptic activities.

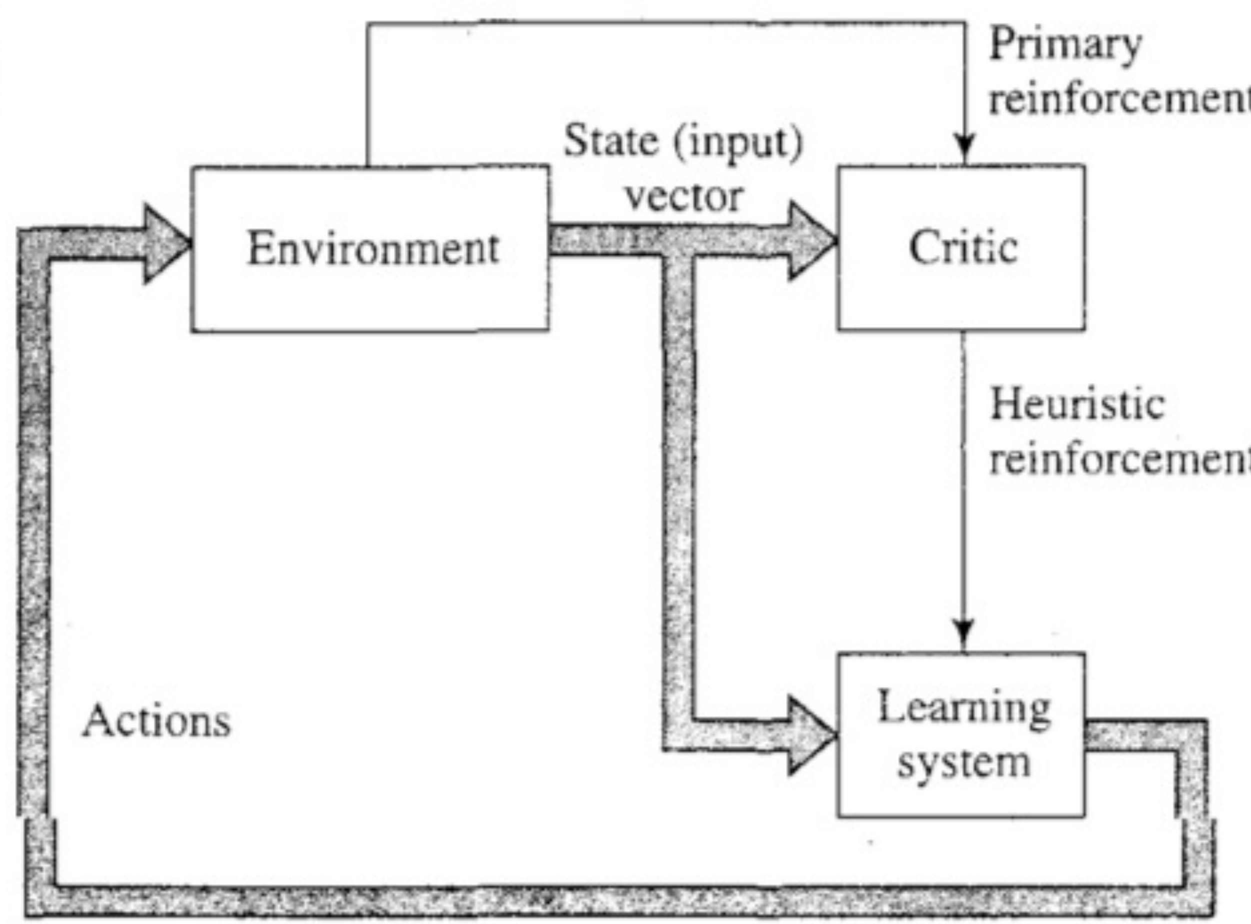
Teacher vs. No Teacher



Supervised Learning



Unsupervised Learning



Reinforcement Learning

Hebbian Learning Model

- General form

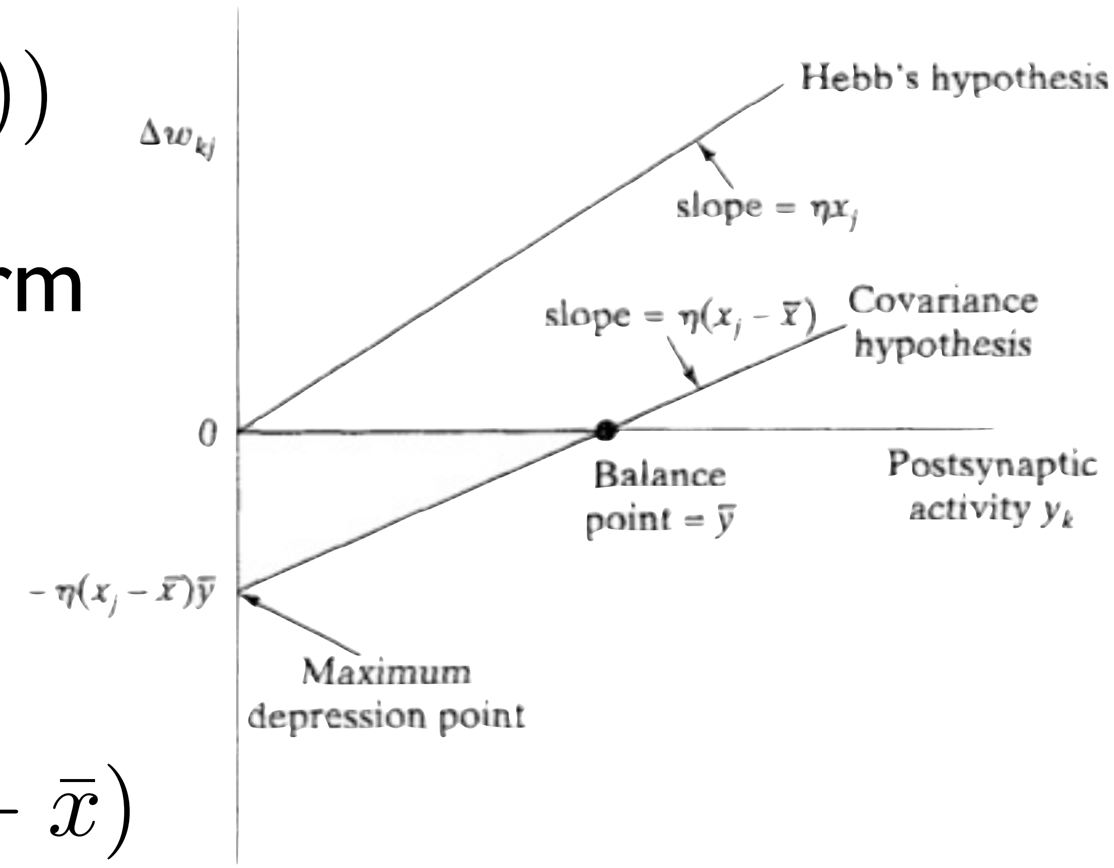
$$\Delta w_{kj}(n) = F(y_k(n), x_j(n))$$

- Simplest Hebbian learning form

$$\Delta w_{kj}(n) = \eta y_k(n) x_j(n)$$

- Covariance Hypothesis

$$\Delta w_{kj}(n) = \eta(y_k - \bar{y})(x_j - \bar{x})$$



- Strong physiological evidence for Hebbian learning in hippocampus

Notes

- It is sometimes called the activity product rule.
- Problem
  - The rule has the exponential growth problem that drives the synaptic weight into saturation.
- Solution

$$\Delta w_{kj}(n) = \eta \; y_k(n)x_j(n) - \alpha \; y_k(n)w_{kj}(n)$$

- Internal feedback acting on the neurons:
  - Positive feedback for self-amplification and therefore growth of the synaptic weight $w_j(n)$, according to its external input $x_i(n)$.
  - Negative feedback due to -y(n) for controlling the growth, thereby resulting in stabilization of the synaptic weight $w_i(n)$.
- The product -y(n) $w_j(n)$ is related to a forgetting or leakage factor.

Competitive Learning

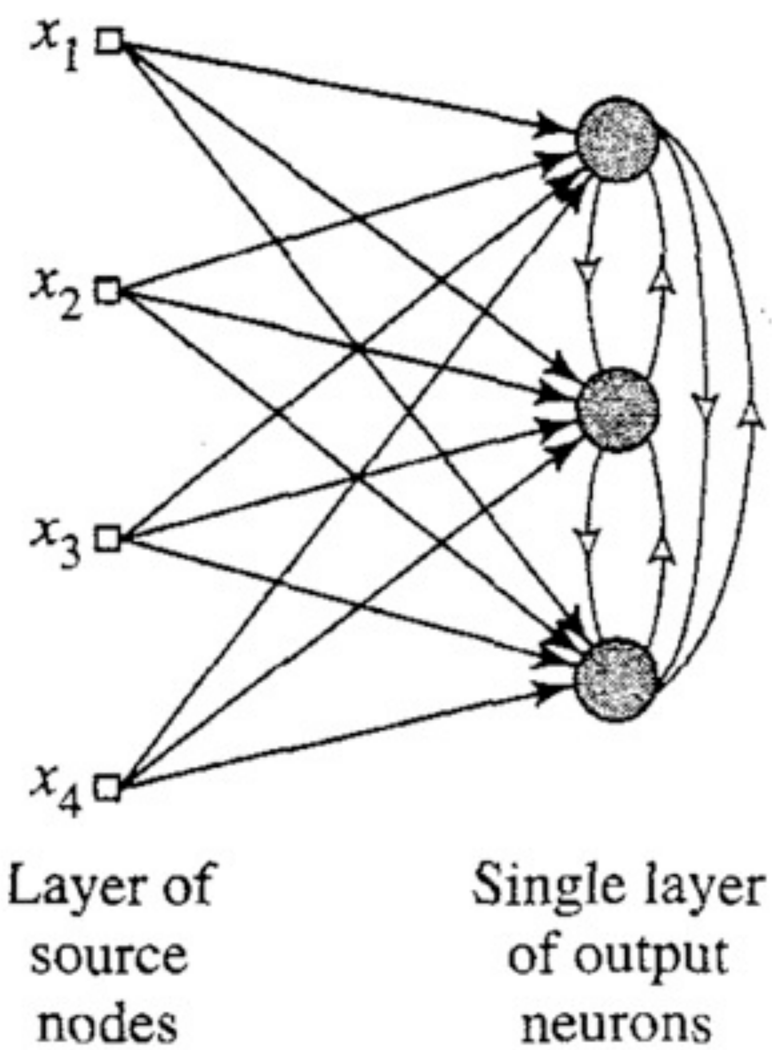
- In competitive learning the output neurons of a neural network compete among themselves for being the one to be active (fired).
  - A set of neurons that are all the same except for some randomly distributed synaptic weights, and which therefore respond differently to a given set of input patterns.
  - A limit imposed on the ``strength'' of each neuron.
  - A mechanism that permits the neurons to compete for the right to respond to a given subset of inputs, such that only one output neuron, or only one neuron per group, is active (i.e., ``on'') at a time. The neuron that wins the competition is called a winner-take-all neuron.

Competitive Learning

- This way, the individual neurons of the network learn to specialize on sets of similar patterns, and thereby become feature detectors.
- In the simplest form, the network has a single layer of output neurons, each of which is fully connected to the input nodes.
- The network may include lateral connections among the neurons.
- The lateral connections perform lateral inhibition, which each neuron tending to inhibit the neuron to which it is laterally connected.

Competitive Learning

$$y_k = \begin{cases} 1, & \text{if } v_k > v_j \text{ for all } i, j \neq k \\ 0, & \text{otherwise} \end{cases}$$

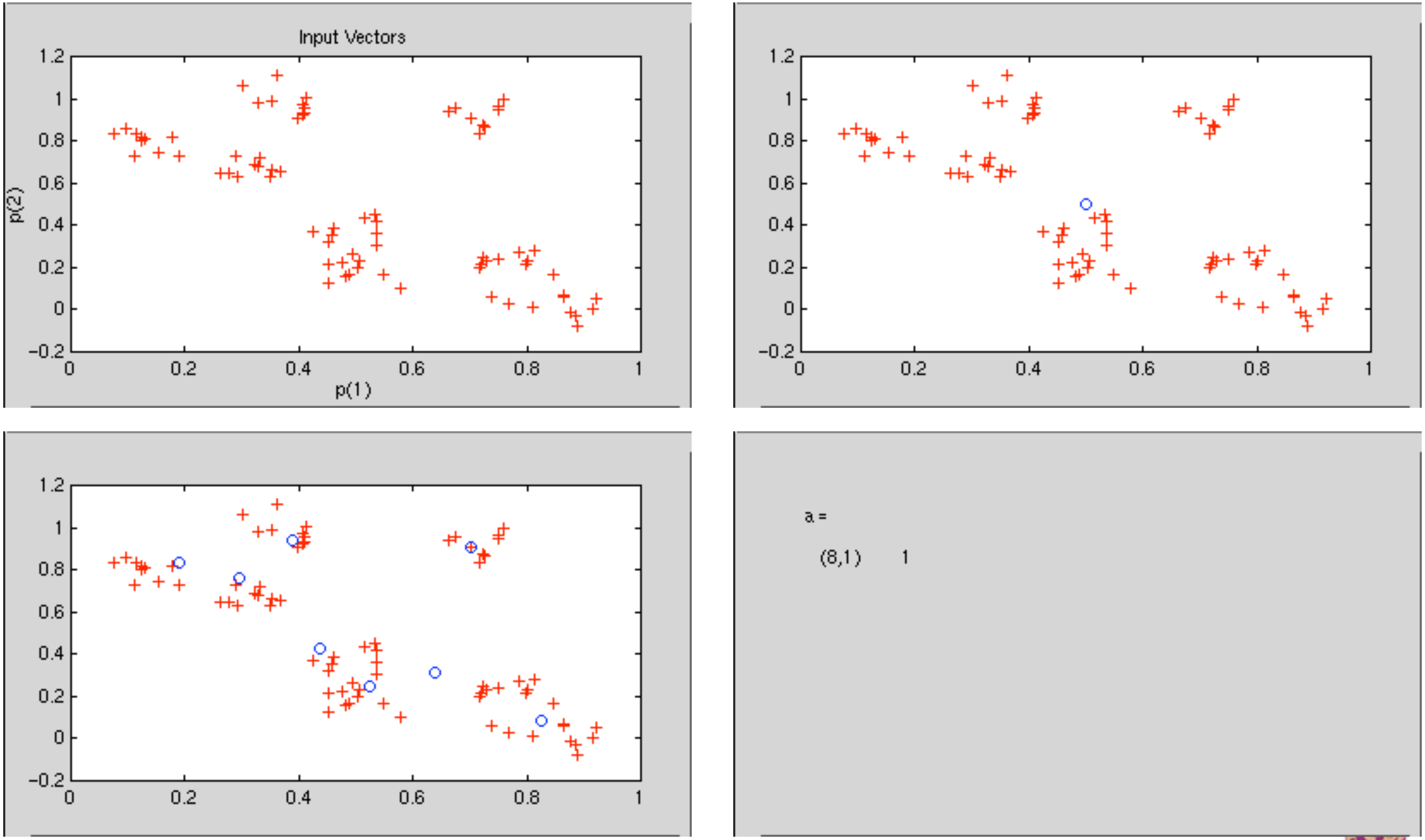


$$\sum_j w_{kj} = 1 \text{ for all } k$$

$$\sum_j w_{kj}^2 = 1 \text{ for all } k$$

$$\Delta w_{kj} = \begin{cases} \eta(x_j - w_{kj}), & \text{if neuron } k \text{ wins the competition} \\ 0, & \text{if neuron } k \text{ loses the competition} \end{cases}$$

Example



Self-Organizing Maps

- Kohonen's SOM is a simple geometric computation for the more detailed properties of the Hebb-like rule and lateral interactions.
  - Sampling
  - Similarity matching
  - Updating
- Initialization
  - Choose random and unique values for the initial weight vector $w_j(0)$ for $j = 1, 2, \ldots, N$.
- Sampling
  - Draw a sample x from the input distribution with a certain probability. x represents the sensory signal.

SOM

- Similarity Matching
  - Find the best-matching (winning) neuron i(x) at time n, using the minimum-distance Euclidean criterion:

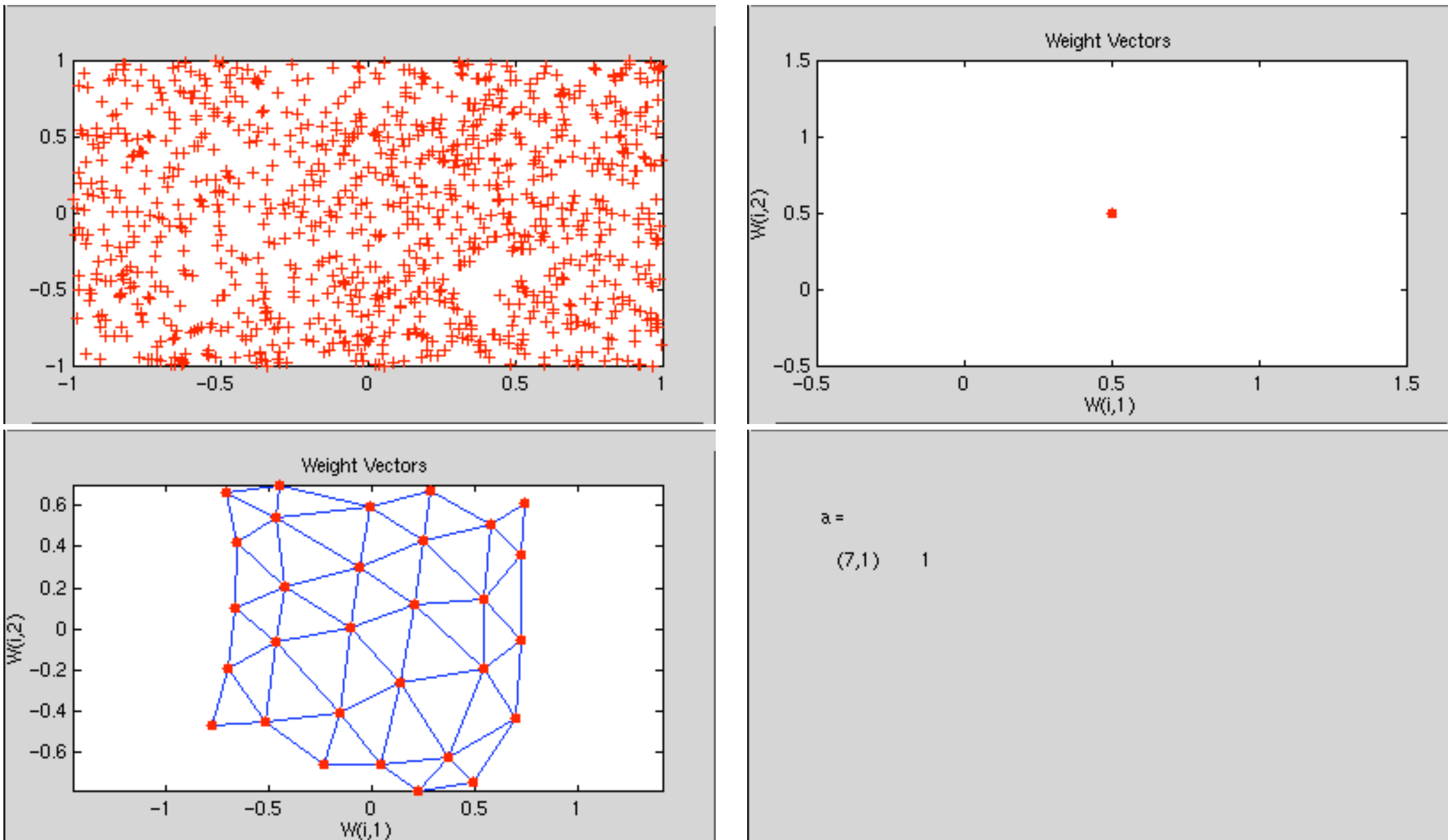
$$i(\mathbf{x}) = \text{argmin}_j ||\mathbf{x} - \mathbf{w}_j||, j = 1, 2, \ldots, l$$

- Updating
  - Adjusting the synaptic weight vectors of all neurons

$$w_j(n+1) = \begin{cases} w_j(n) + \eta(n)[x(n) - w_j(n)], & j \in \Lambda_{i(x)}(n) \\ w_j(n), & \text{otherwise} \end{cases}$$

  - where \eta(n) is the learning-rate, \Lambda$_{i(x)}$(n) is the neighborhood function centered around the winning neuron i(x); both terms are varied dynamically during learning for best results.

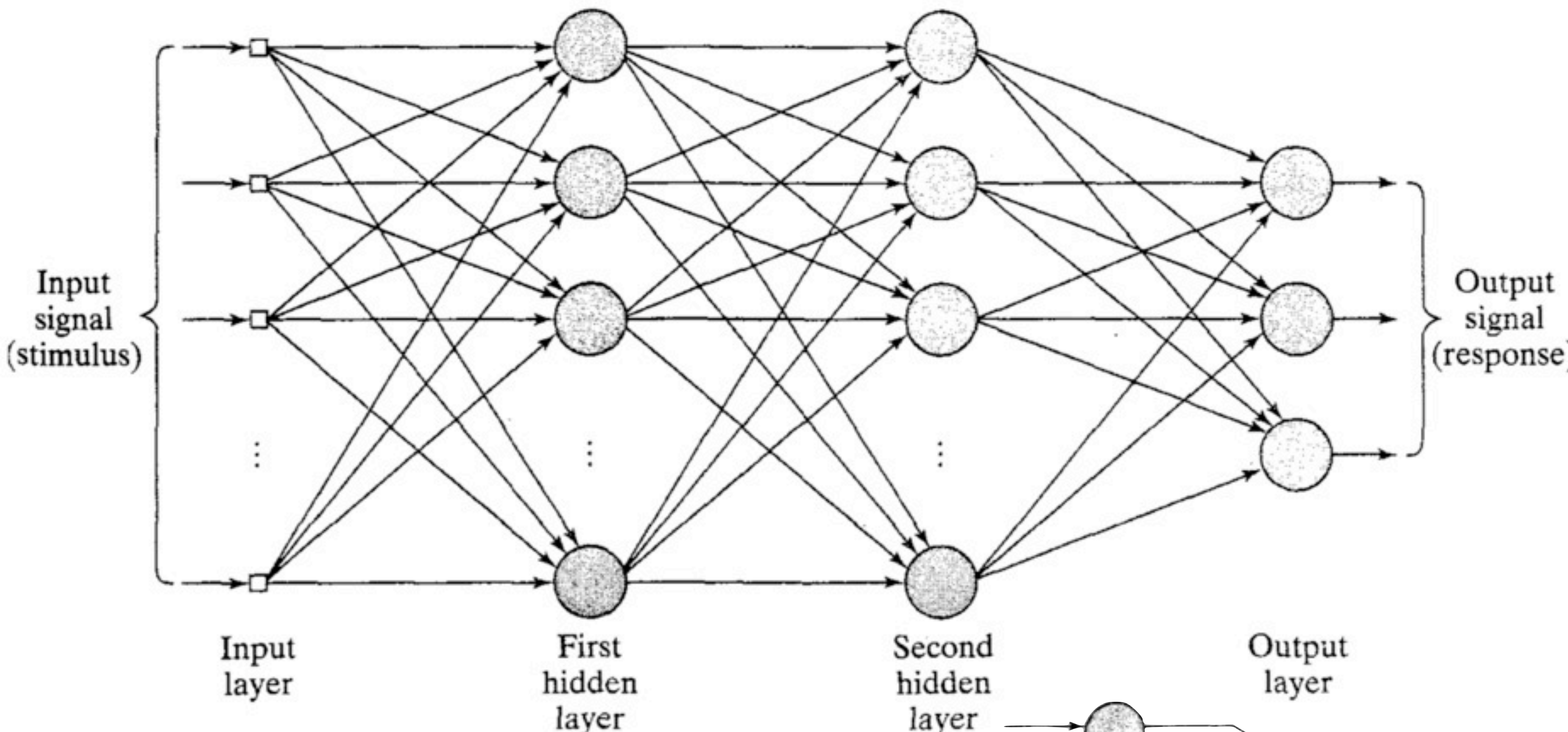
Example

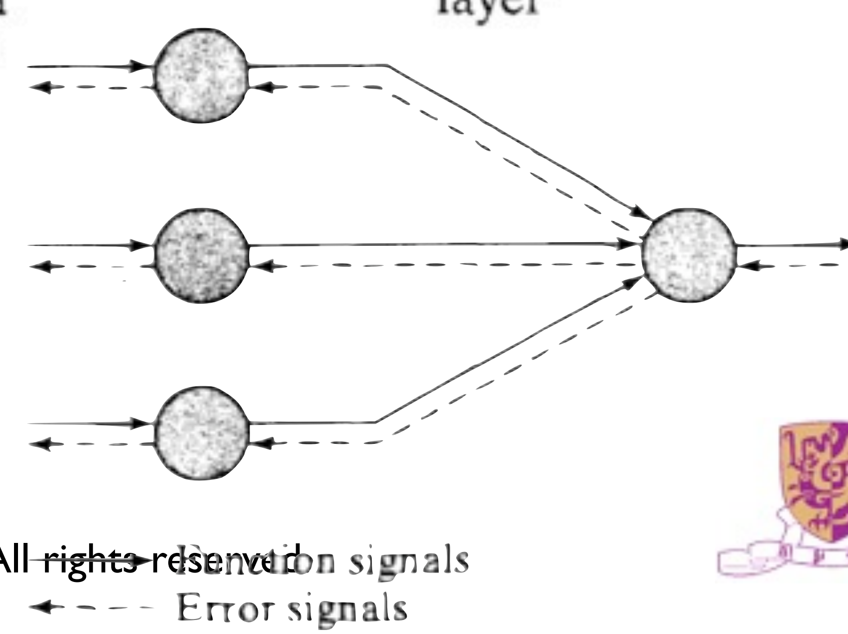

1
0.5
0
−0.5
−1
−1
−0.5
0
0.5
1
Weight Vectors
1.5
1
0.5
0
−0.5
−0.5
0
0.5
1
1.5
W(i,1)
W(i,2)
Weight Vectors
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−1
−0.5
0
0.5
1
W(i,1)
W(i,2)
a =
(7,1) 1

Back-Propagation Learning

- One of the most commonly used learning algorithm
- Supervised Learning
- Not biologically motivated
- Easy to train
- Adequate results

Multilayer Perceptron (2 hidden layers)





Notation I

- The indices i, j, and k refer to different neurons in the network; with signals propagating through the network from left to right, neuron j lies in a layer to the right of neuron i, and neuron k lies in a layer to the right of neuron j when neuron j is a hidden unit.
- The iteration n refers to the n-th training pattern (example) presented to the network.
- The symbol E(n) refers to the instantaneous sum of error squares at iteration n.
- The average of E(n) over all values of n (i.e., the entire training set) yields the average squared error E_{av}.
- The symbol $e_j(n)$ refers to the error signal at the output of neuron j for iteration n.

Notation II

- The symbol $y_j(n)$ refers to the function signal appearing at the output of neuron j at iteration n.
- The symbol $w_{ji}(n)$ denotes the synaptic weight connection the output of neuron i to the input of neuron j at iteration n.
- The correction applied to this weight at iteration n is denoted by $\Delta\, w_{ji}(n)$.

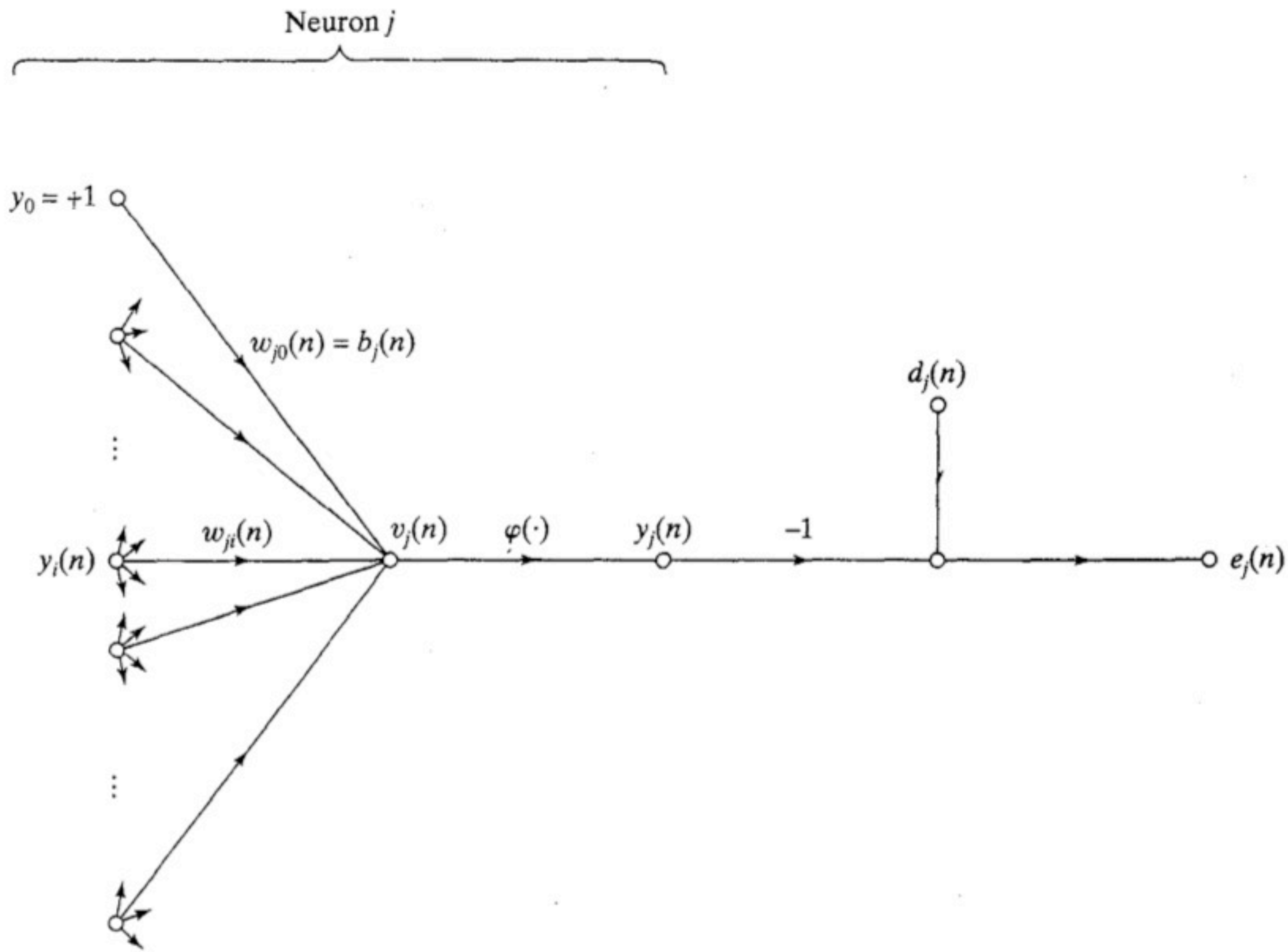
Notation III

- The net internal activity level of neuron j at iteration n is denoted by $v_j(n)$; it constitutes the signal applied to the nonlinearity associated with neuron j.
- The activation function describing the input-output functional relationship of the nonlinearity associated with neuron j is denoted by $\phi_j(\bullet)$.
- The symbol $d_j(n)$ refers to the desired response for neuron j and is used to compute $e_j(n)$.

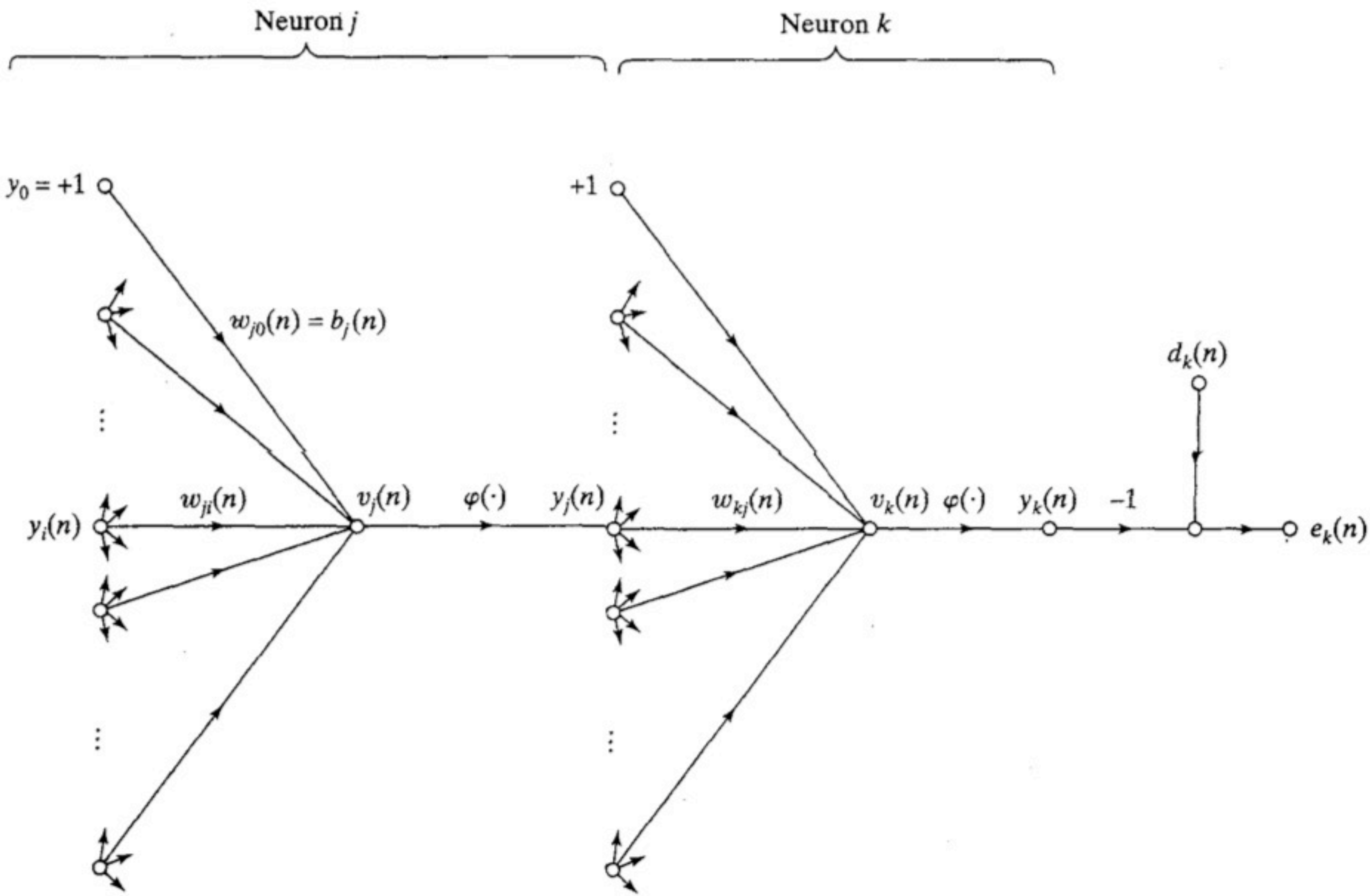
Notation IV

- The threshold applied to neuron j is denoted by \theta$_j$; its effect is represented by a synapse of weight w_{j0} = j connected to a fixed input equal to -1.
- The i-th element of the input vector (pattern) is denoted by $x_i(n)$.
- The k-th element of the overall output vector (pattern) is denoted by $o_k(n)$.
- The learning-rate parameter is denoted by \eta.

Output Neuron j



Output k to Hidden J



BP Algorithm I

1. $$e_j(n) = d_j(n) - y_j(n)$$

2. $$\mathcal{E}(n) = \frac{1}{2} \sum_{j \in C} e_j^2(n)$$

3. $$\mathcal{E}_{\mathrm{av}} = \frac{1}{N} \sum_{n=1}^{N} \mathcal{E}(n)$$

4. $$v_j(n) = \sum_{i=0}^{P} w_{ji}(n) y_i(n)$$

5. $$y_j(n) = \phi_j(v_j(n))$$

BP Algorithm II

6. $$\frac{\partial \mathcal{E}(n)}{\partial w_{ji}(n)} = \frac{\partial \mathcal{E}(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)}$$

7. Differentiate (2), we get $$\frac{\partial \mathcal{E}(n)}{\partial e_j(n)} = e_j(n)$$

8. Differentiate (1), we get $$\frac{\partial e_j(n)}{\partial y_j(n)} = -1$$

9. Differentiate (5), we get $$\frac{\partial y_j(n)}{\partial v_j(n)} = \phi'_j(v_j(n))$$

BP Algorithm III

10. Differentiate (4), we get $$\frac{\partial v_j(n)}{\partial w_{ji}(n)} = y_i(n)$$

11. Use (7) to (10) in (6) gets $$\frac{\partial \mathcal{E}(n)}{\partial w_{ji}(n)} = -e_j(n)\phi_j'(v_j(n))y_i(n)$$

12. Delta Rule $$\Delta w_{ji}(n) = -\eta \frac{\partial \mathcal{E}(n)}{\partial w_{ji}(n)}$$

13. Use (11) in (12) gets $$\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n)$$

$$\delta_j(n) = -\frac{\partial \mathcal{E}(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} = e_j(n)\phi_j'(v_j(n))$$

Forward Pass

- In the forward pass the synaptic weights remain unaltered throughout the network, and the function signals of the network are computed on a neuron-by-neuron basis.

$$y_j(n) = \phi_j(v_j(n))$$

- where $v_j(n)$ is the net internal activity level of neuron j, defined by

$$v_j(n) = \sum_{i=0}^{P} w_{ji}(n) y_i(n)$$

Backward Pass

- The backward pass starts at the output layer by passing the error signals leftward through the network, layer by layer, and recursively computing the (i.e., the local gradient) for each neuron.
- Note that for the presentation of each training example, the input pattern is fixed ("clamped") throughout the round-trip process, encompassing the forward pass followed by the backward pass.

Rate of Learning and Momentum

- Smaller the \eta
  - The smaller the changes to the synaptic weights in the network from one iteration to the next.
  - The smoother will be the trajectory in weight space.
  - It is slow to learn
- One way to modify the delta rule of (13) is to include a momentum term as

$$\Delta w_{ji}(n) = \alpha \Delta w_{ji}(n-1) + \eta \delta_j(n) y_i(n)$$

- where \alpha is usually a positive number called the momentum constant.

Reinforcement Learning

- Reinforcement learning is the on-line learning of an input-output mapping through a process of trial and error designed to maximize a scalar performance index called a reinforcement signal.
  - Non-associative reinforcement
    - Task of selecting a single optimal action rather than to associate different actions with different stimuli.
  - Associative reinforcement learning
    - Environment provides additional forms of information other than reinforcement.