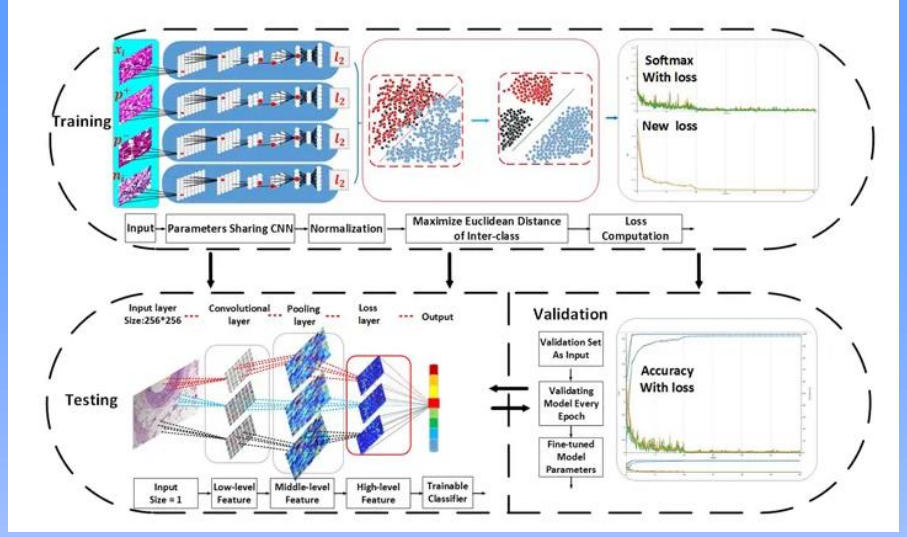
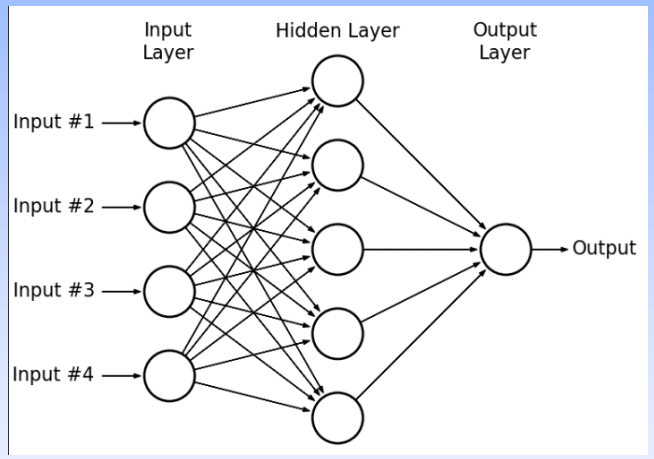
Artificial Intelligence

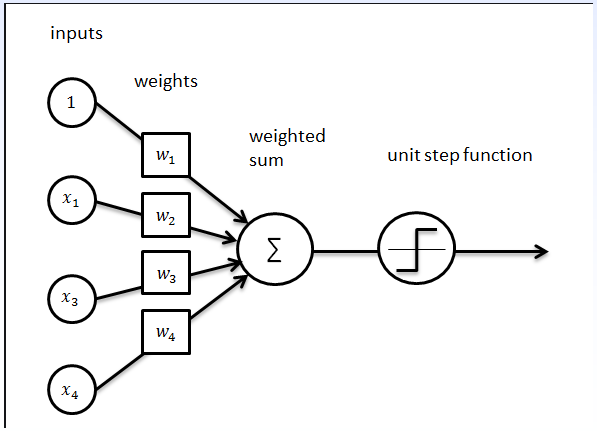
Deep learning neural networks



Multi-Layer Perceptron

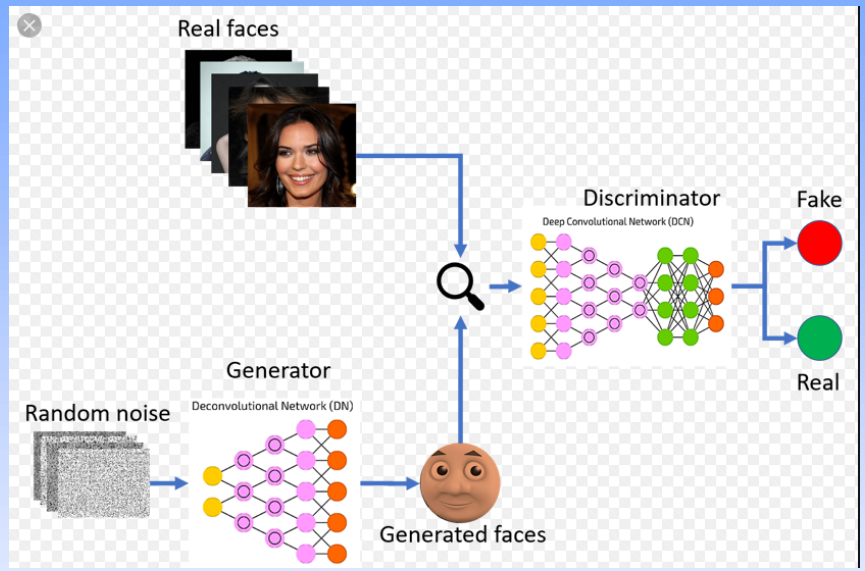


Perceptron

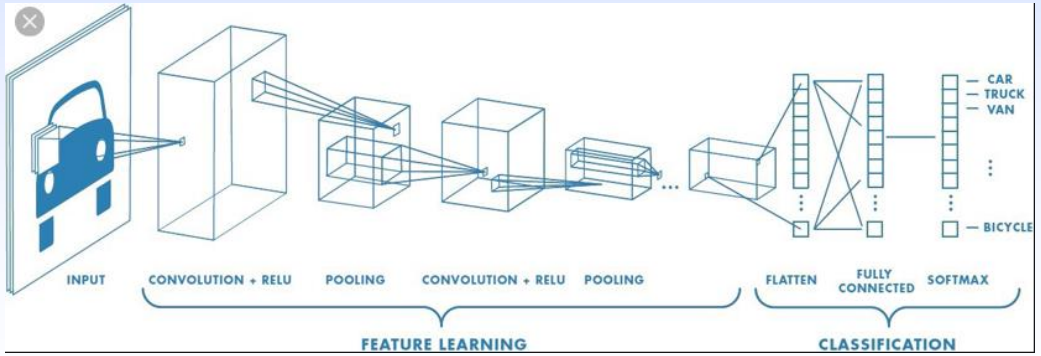


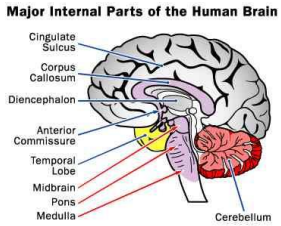
...

Generative Adversarial Networks (GAN)



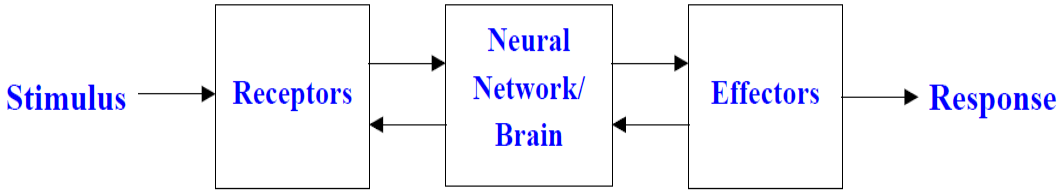
Convolutional Neural Networks





The Nervous System

The human nervous system can be broken down into three stages that may be represented in block diagram form as:



The receptors collect information from the environment – e.g. photons on the retina.

The effectors generate interactions with the environment – e.g. activate muscles.

The flow of information/activation is represented by arrows – feedforward and feedback.

Naturally, in this module we will be primarily concerned with the neural network in the middle.

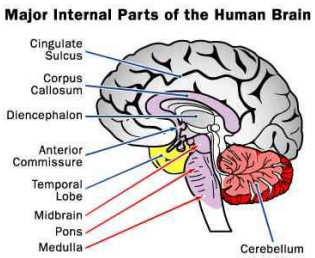
Levels of Brain Organization

The brain contains both large scale and small scale anatomical structures and different functions take place at higher and lower levels.

There is a hierarchy of interwoven levels of organization:

1. Molecules and Ions
2. Synapses
3. Neuronal microcircuits
4. Dendritic trees
5. **Neurons**
6. **Local circuits**
7. Inter-regional circuits
8. Central nervous system

The ANNs we study in this module are crude approximations to levels 5 and 6.



Brains versus Computers : Some numbers

1. There are approximately 10 billion neurons in the human cortex, compared with 10 of thousands of processors in the most powerful parallel computers.

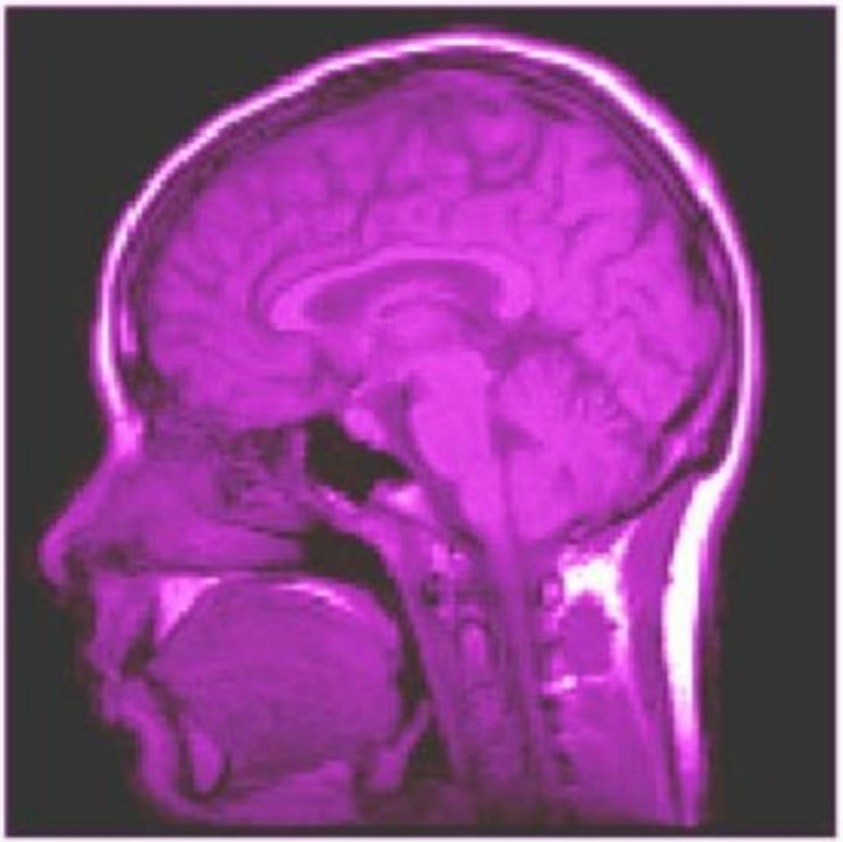
2. Each biological neuron is connected to several thousands of other neurons, similar to the connectivity in powerful parallel computers.

3. Lack of processing units can be compensated by speed. The typical operating speeds of biological neurons is measured in milliseconds (10^{-3} s), while a silicon chip can operate in nanoseconds (10^{-9} s).

4. The human brain is extremely energy efficient, using approximately 10^{-16} joules per operation per second, whereas the best computers today use around 10^{-6} joules per operation per second.

5. Brains have been evolving for tens of millions of years, computers have been evolving for tens of decades.

The Brain

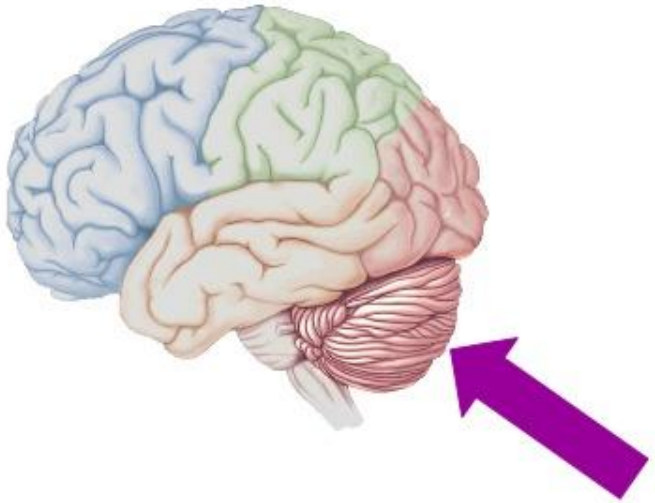


- weighs 1300 - 1400 g
- made up of about 100 billion neurons
- "the most complex living structure on the universe" Society for Neuroscience
- makes us who we are

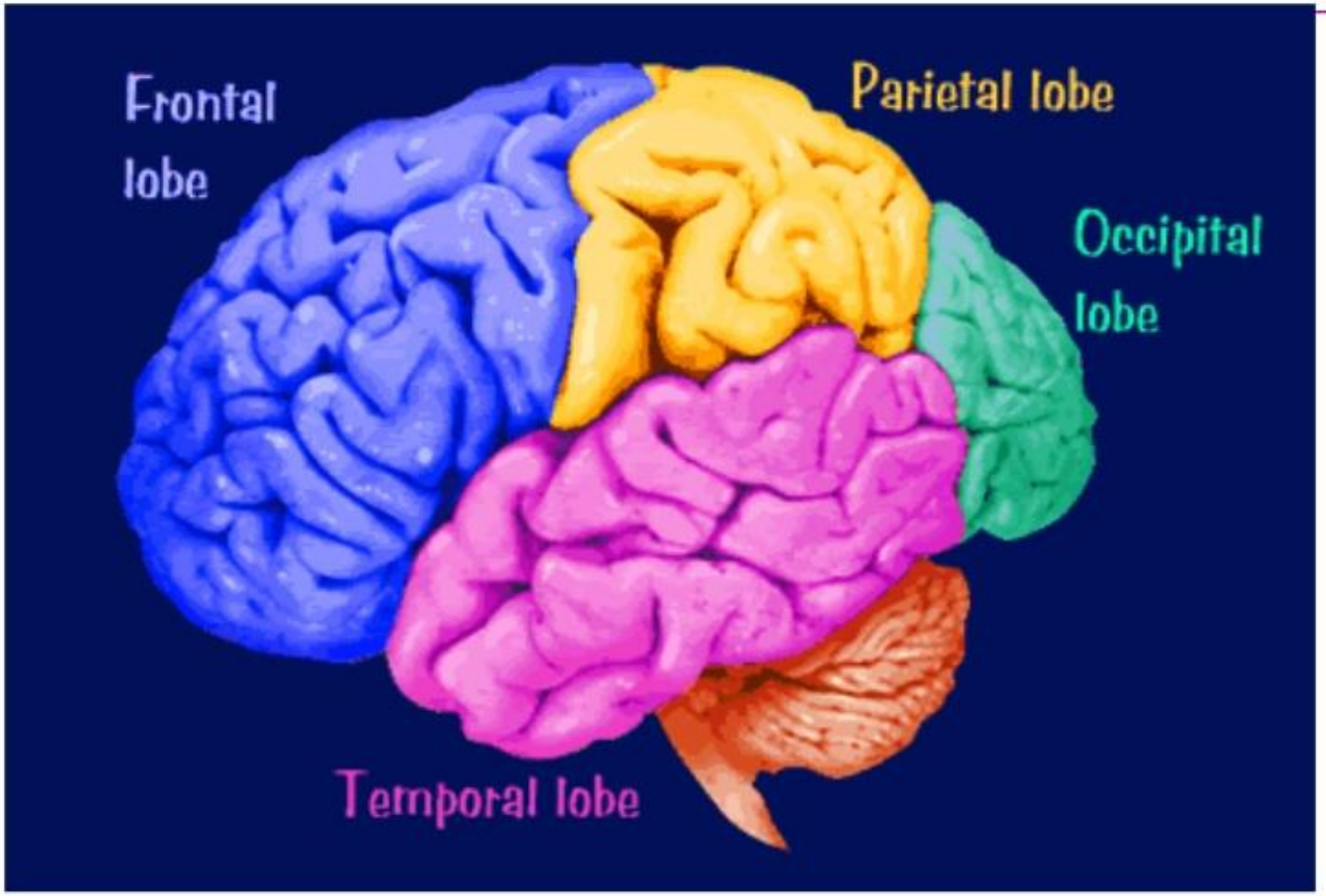
Cerebellum

- Found at the at the back of your head under the cerebrum
- Means "little brain"
- Responsible for movement, balance, posture.
- Often takes over learned activities-

  Like riding a bike!

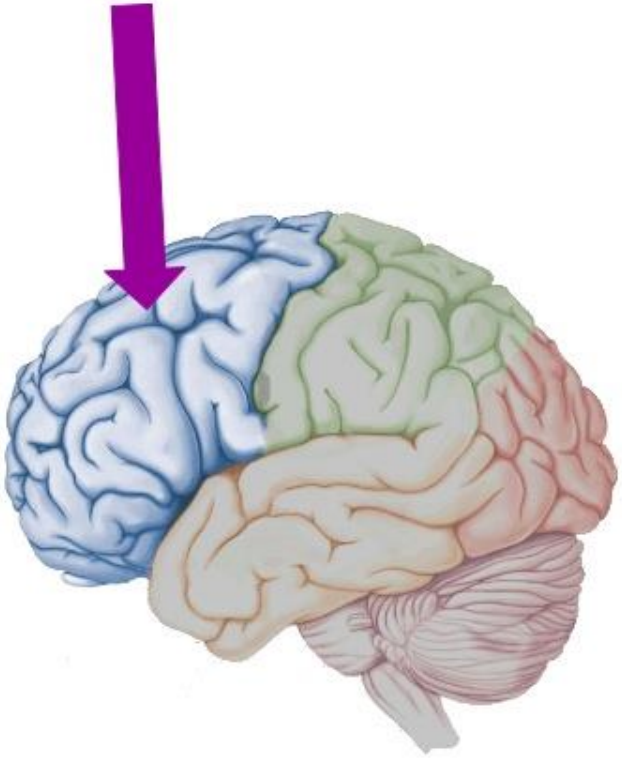


Parts of the cerebrum



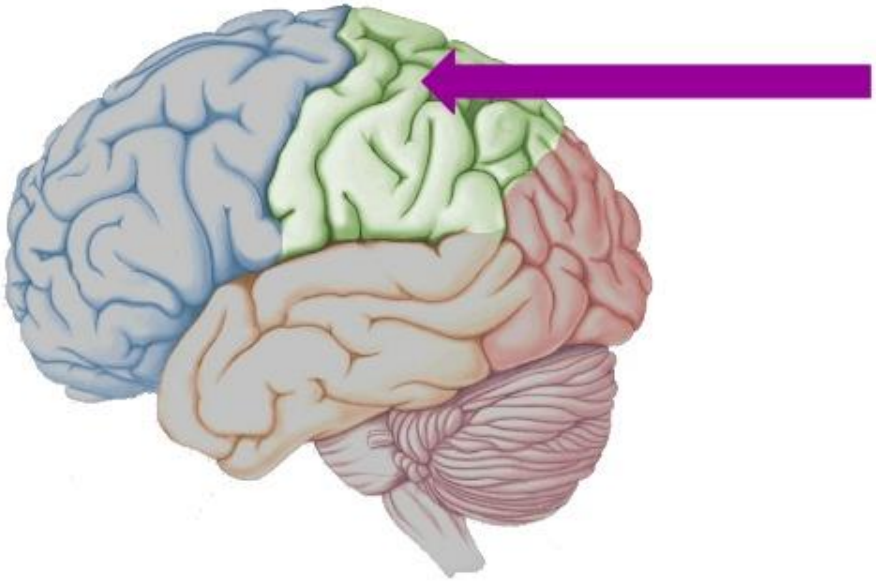
Frontal Lobe

- Found under your forehead.
- Center of reasoning
- Planning
- some parts of speech
- movement (motor cortex)
- Emotions
- problem solving.



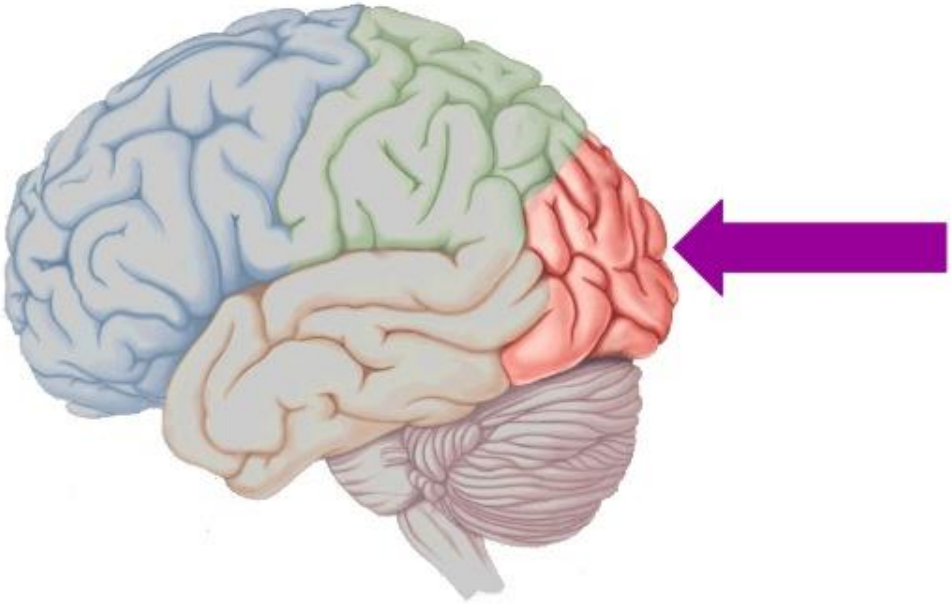
Parietal Lobe

- Found on the top of your head.
- Receives sensory input from the skin. (touch, pressure, temperature, & pain)



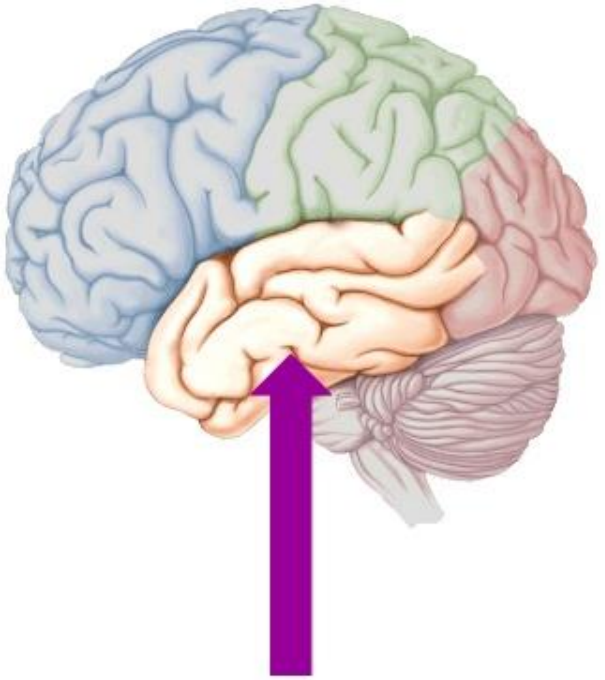
Occipital

- Found at the back of your head.
- Receives input from the eyes
- Often referred to as the visual cortex

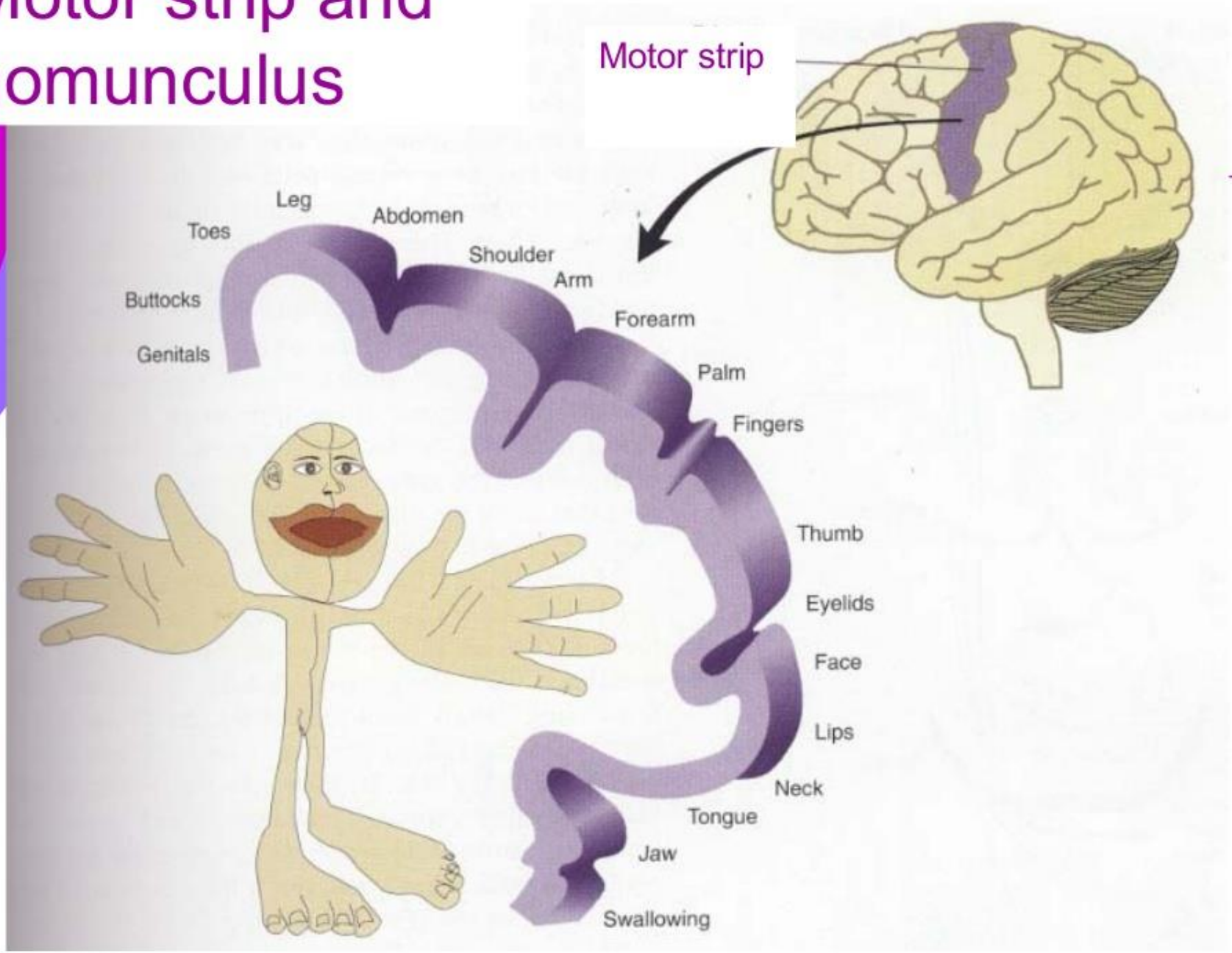


Temporal Lobe

- Found on the sides of your head above your ears.
- Functions include:
  - speech perception
  - hearing,
  - some types of memory



Motor strip and homunculus



Hemispheres

- uses logic
- detail oriented
- facts rule
- words and language
- present and past
- math and science

- Acknowledges order/pattern
- Perception
- knows object name
- reality based
- forms strategies
- Practical
- safe

Hemispheres- Right

- uses feelings
- "big picture" oriented
- imagination rules
- symbols and images
- present and future
- philosophy & religion

- believes
- appreciates
- spatial perception
- knows object function
- fantasy based
- presents possibilities
- impetuous
- risk taking

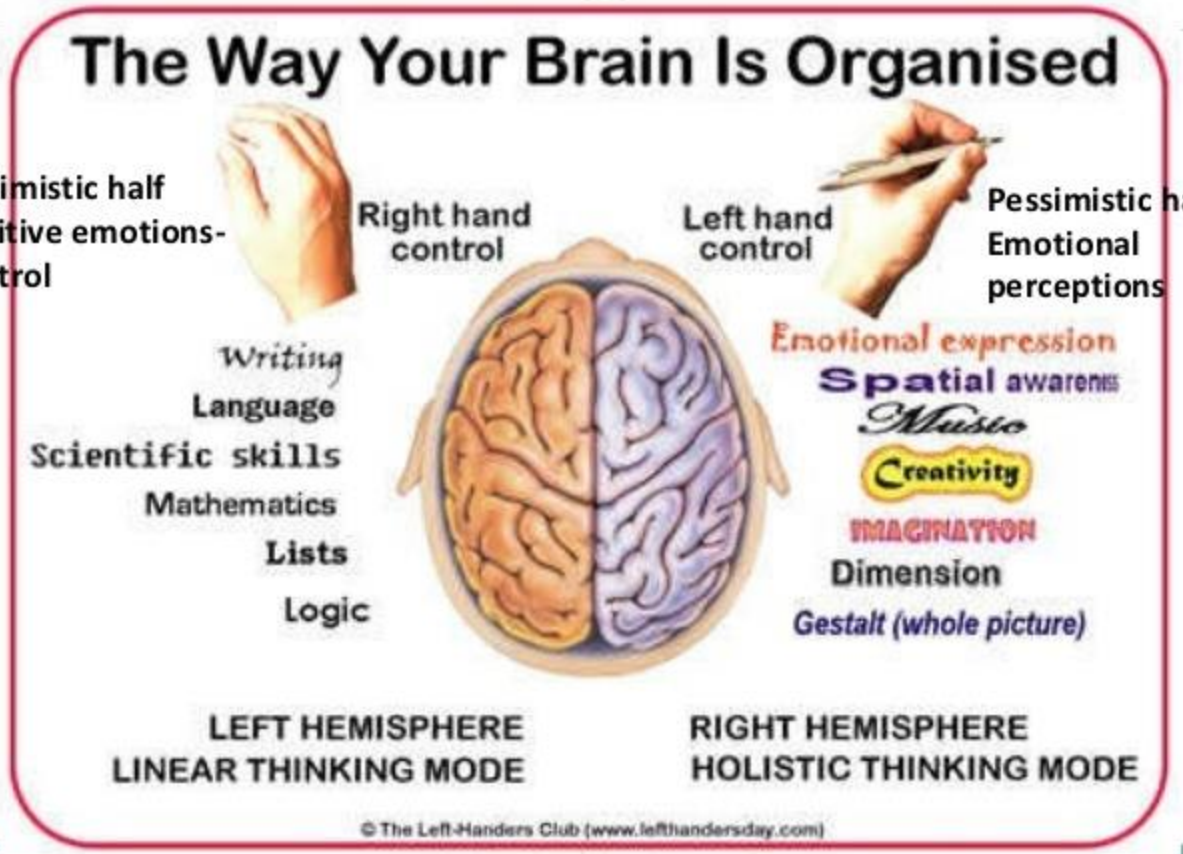
Cerebral Cortex

The cerebral cortex is the thin (3-5 mm thick) convoluted (folded) outer layer of the brain.
The value of the **folding** is that it enables a greater surface area of cerebral cortex to be contained within the skull; this gives a greater volume and enables the cerebral cortex to contain more neurons and more blood-vessels to get more oxygen and glucose (for energy) to this most fuel-hungry part of the body.

There are three main types of functional areas in the cerebral cortex:

1. the **sensory areas** which receive information provided by the various senses (especially touch, vision, and hearing);
2. The **motor cortex** which sends information to muscles so they can create bodily movements
3. The **association areas** which integrate sensory and motor information and are involved with information processing activities such as language and speech, learning, memory, thinking and problem solving

Left and Right Hemispheres



Men's and women's learning styles and capacities

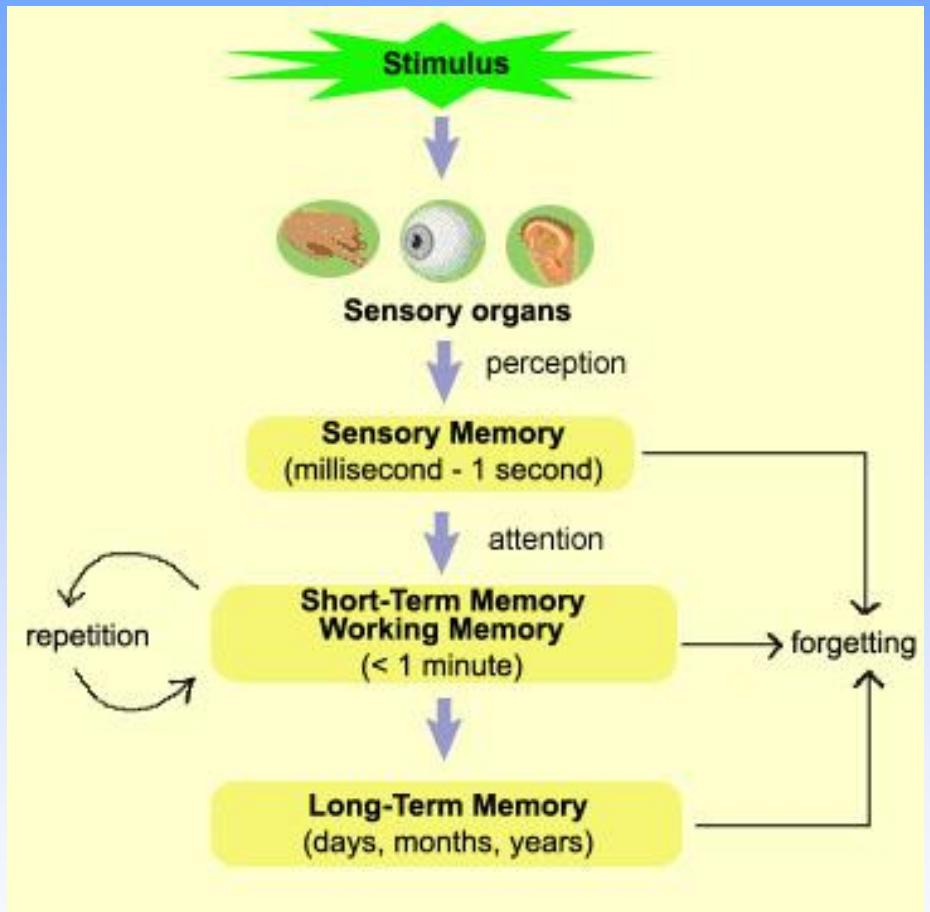
	MEN	WOMEN
Temporal lobe: this region of the cerebral cortex helps control hearing, memory and a person's sense of self and time.	In cognitively normal men, a tiny region of the temporal lobe behind the eye has about 10% fewer neurons than it does in women.	Women have more neurons in this region, which understands language as well as melodies and speech tones.
Corpus callosum: this bundle of neurons is the main bridge between the left and the right hemispheres, carrying messages between them.	A man's corpus callosum takes up less volume in his brain than a woman's does, suggesting the two hemispheres communicate .	In women, the back part of the callosum is bigger than in men. That may explain why women use both sides of their brains for language.
Anterior Commissure: this collection of nerve cells also connects the brain's two hemispheres . It is smaller and appeared earlier in evolution than corpus callosum.	In men, the commissure is smaller than it is in women, even though men's brains are, on average, larger than women's.	The larger commisure in women may be another reason their two cerebral hemispheres seem to work in partnership on tasks from language to emotional responses.

Christoph Blumrich (newsweek)

THE NAYSAYERS CLAIM (Among Other Things) "THE BRAIN IS NOT A COMPUTER!"

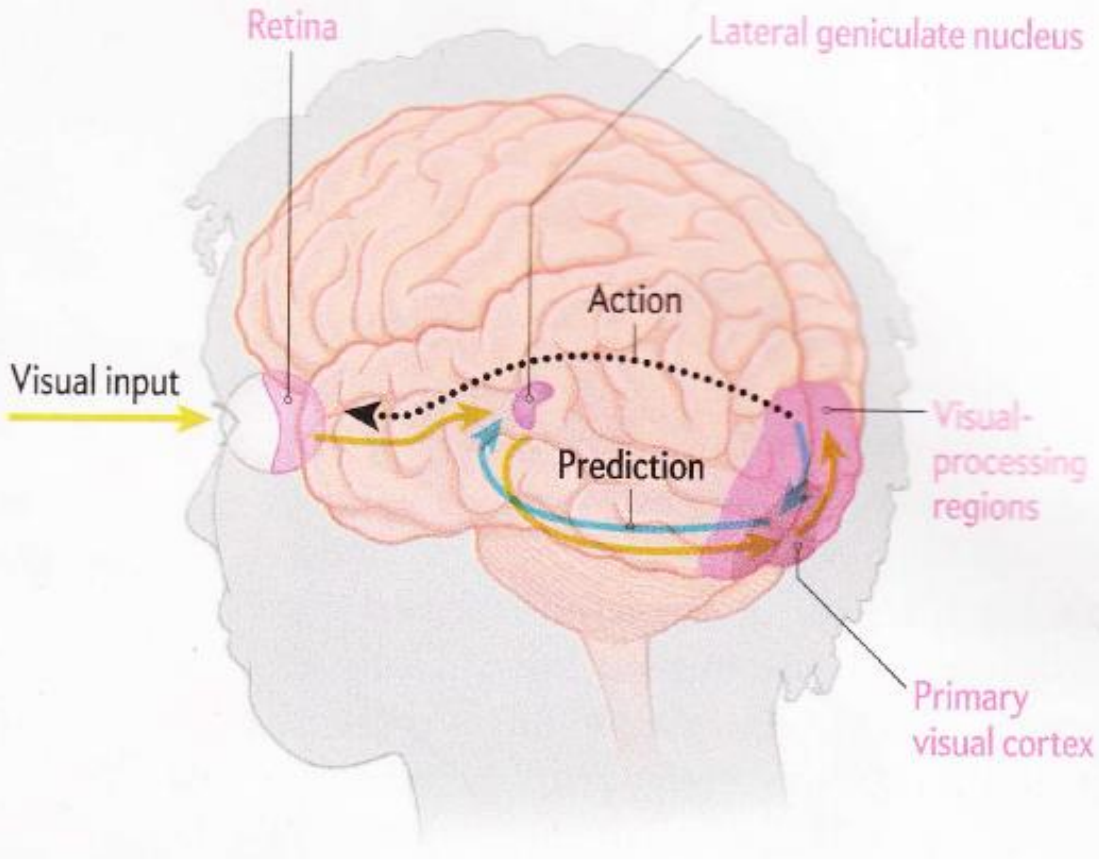
Computation:
- mainly serial
- 10^9 ops/sec
- 10^9 transistors
- digital/discrete (even binary!)
- disembodied
- silicon
- subject to crashes
- . . .

The Brain:
- highly parallel
- 10^3 ops/sec
- 10^{14} neurons; 10^{17} synapses
- analog/continuous
- embodied
- protein
- fault-tolerant
- . . .


Stimulus
Sensory organs
perception
Sensory Memory
(millisecond - 1 second)
attention
repetition
Short-Term Memory
Working Memory
(< 1 minute)
forgetting
Long-Term Memory
(days, months, years)

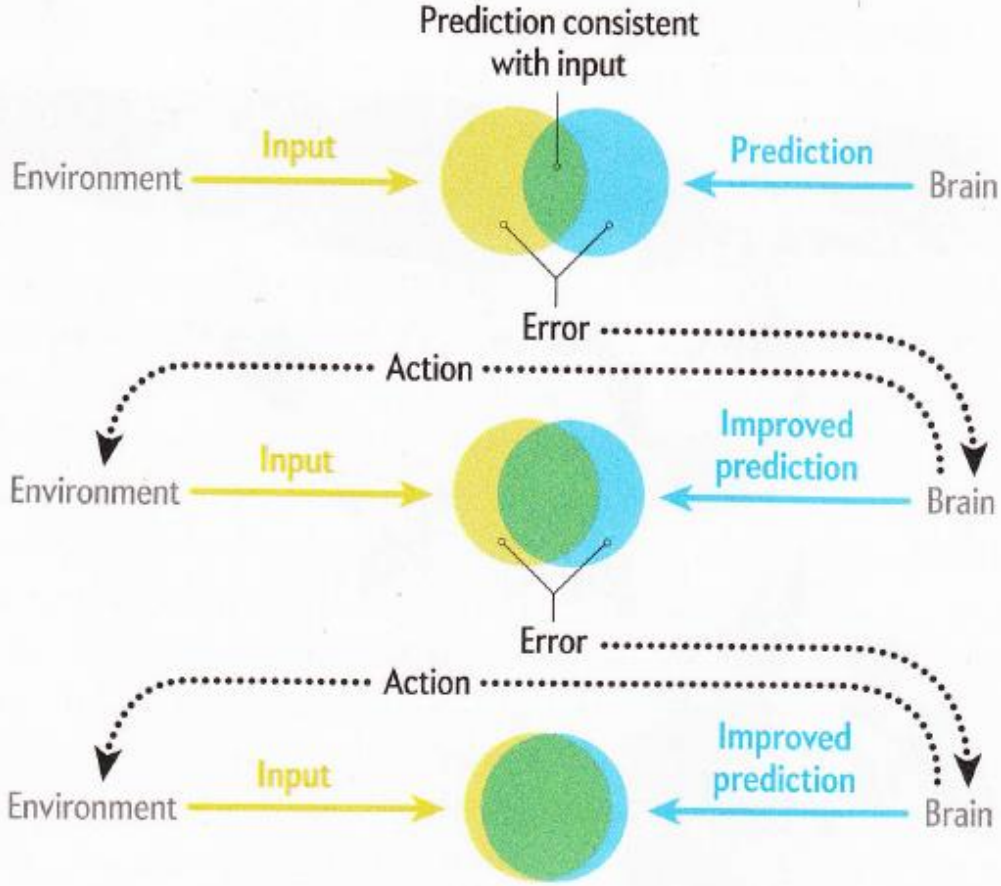
Predictive Brain

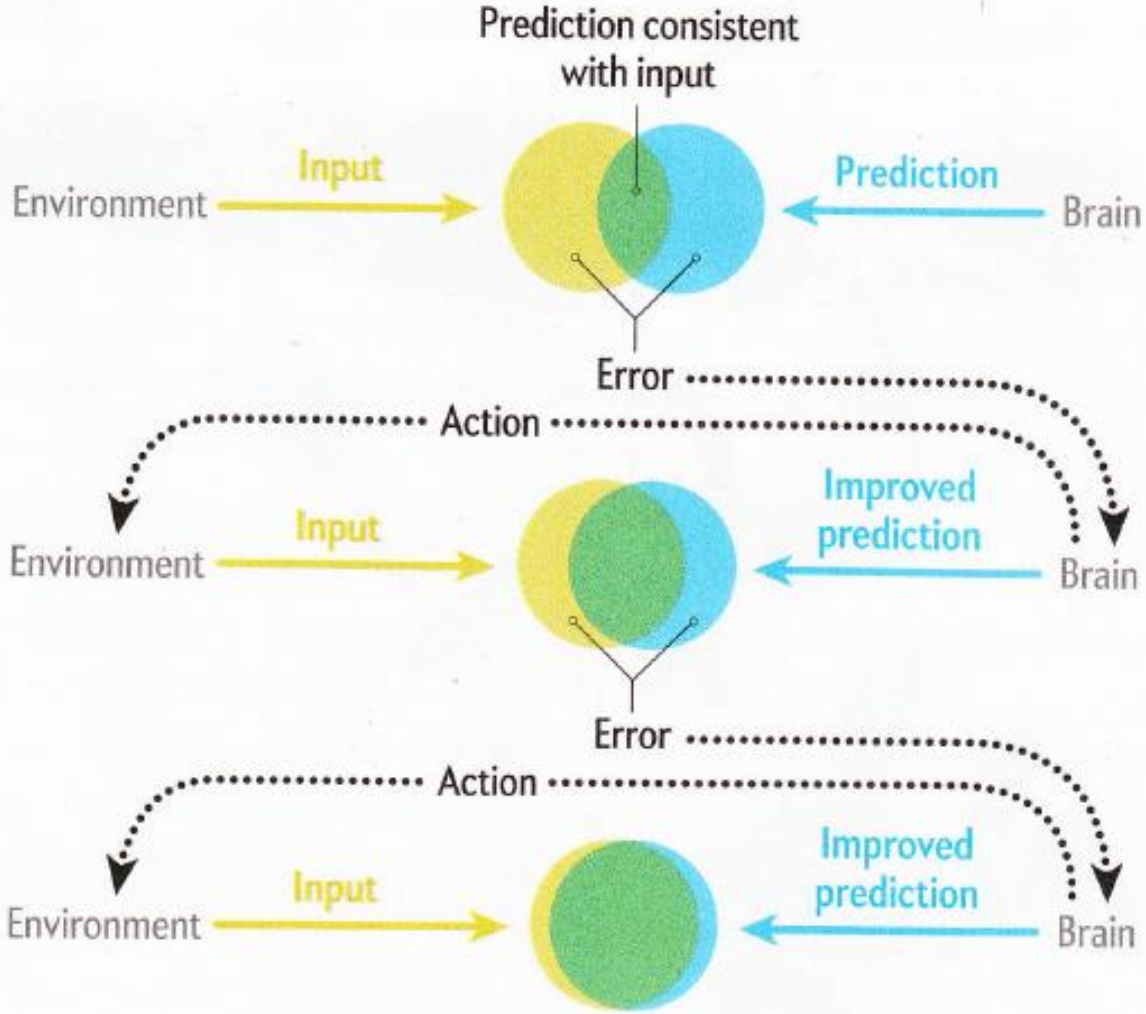
Our minds are prediction machines, using prior experience and knowledge to make sense of the deluge of information coming from our surroundings. Many neuroscientists and psychologists believe that nearly everything we do—perception, action and learning—relies on making and updating expectations.



Visual Processing

The brain's anatomy supports the idea of predictive processing. The visual cortex, for example, receives inputs from the eye, but connections also run in the other direction. Neuroscientists believe that these "downward" connections, from higher levels of the brain to the lower (such as the primary visual cortex and the lateral geniculate nucleus), carry predictions. These meet with the sensory input to generate a prediction error: the difference between what you expect and what you see. A signal coding this discrepancy returns to the higher levels of the brain. Other downward signals send commands to move the eye muscles, adjusting what we see.





Cascade of Predictions

When the brain generates a prediction error, it uses this information to update its expectations and select actions that will help resolve the discrepancy between beliefs and reality. For example, if an individual cannot determine what an object is simply by looking at it, the brain might send a command to pick up the item and examine it to gather more information.

Why is this important?

Understanding the brain is vital, not just to understand the biological mechanisms which give us our thoughts and emotions and which make us human, but for practical reasons. Understanding how the brain processes information can make a fundamental contribution to the development of new computing technology – neurorobotics and neuromorphic computing. More important still, understanding the brain is essential to understanding, diagnosing and treating brain diseases that are imposing a rapidly increasing burden on the world's aging populations.

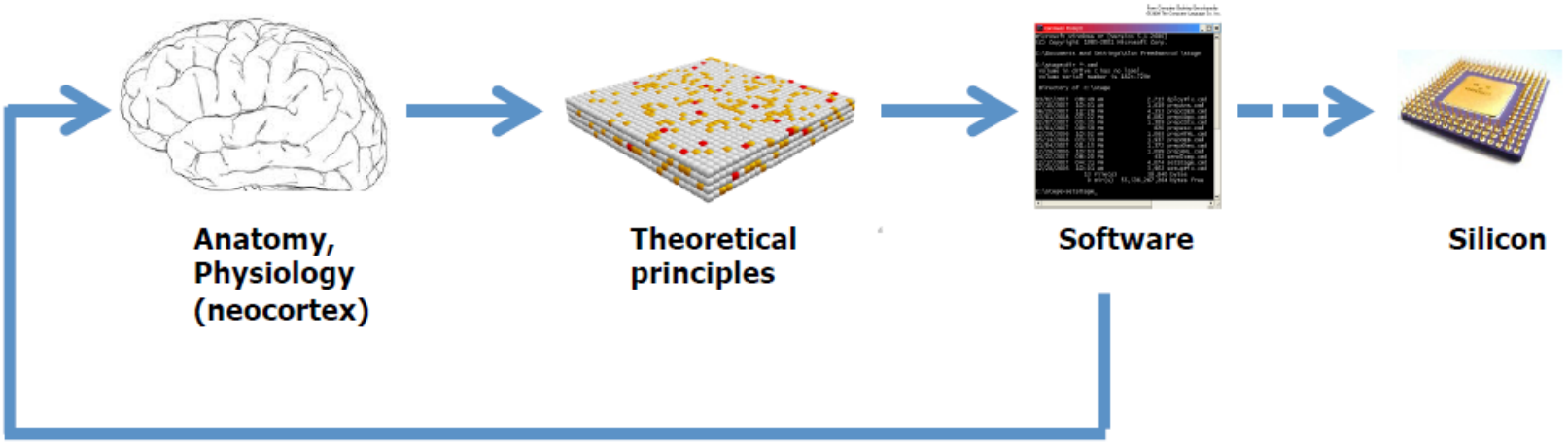
Even a brain that is much smaller than the human brain, like the brain of a rat, is so complex that may never be possible to exhaustively measure all its anatomical features or to fully characterize the physiological interactions within and between its different levels of organization. But this may not be necessary. The structure of the brain and the physiology of its components are subject to tight biological constraints, which are reflected in experimental measurements. The BBP exploits these interdependencies to build comprehensive digital reconstructions from the sparse experimental data that is available and to refine these reconstructions as the data improve. This ability makes the BBP approach inherently scalable.

Simulations suggest that our reconstructions can accurately reproduce many phenomena reported in previous laboratory experiments – without changing the parameters of the reconstruction. As digital reconstructions are refined, expanded, and validated for new kinds of experiment, they can become an ever more valuable resource for neuroscience research, allowing experiments and providing insights that would not be possible with alternative approaches.

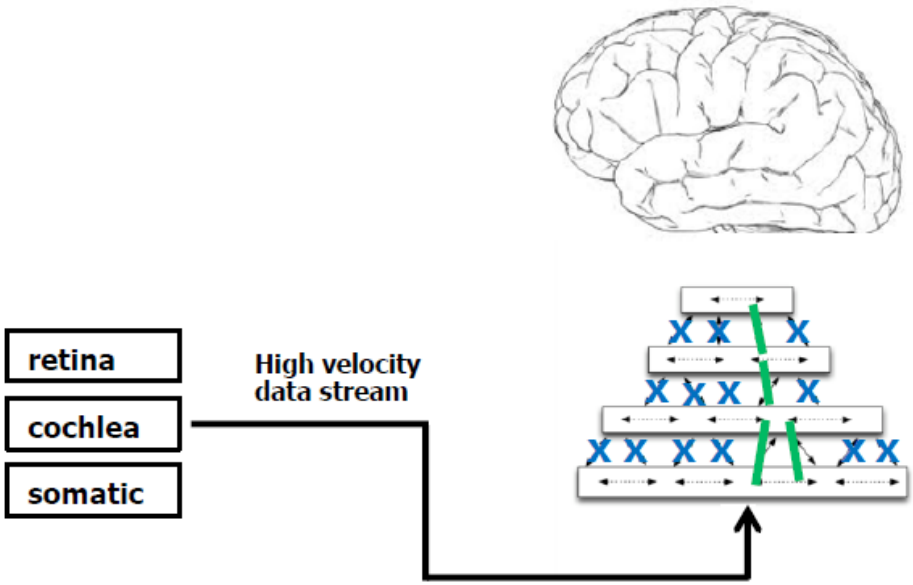
The "Just Right" Approach

1) Discover operating principles of neocortex

2) Build systems based on these principles



Principles of Neocortical Function



1) On-line learning from streaming data

2) Hierarchy of memory regions

3) Sequence memory

4) Sparse Distributed Representations

5) All regions are sensory and motor

6) Attention

THE NAYSAYERS CLAIM (Among Other Things) "THE BRAIN IS NOT A COMPUTER!"

Computation:
- mainly serial
- 10^9 ops/sec
- 10^9 transistors
- digital/discrete (even binary!)
- disembodied
- silicon
- subject to crashes
- . . .

The Brain:
- highly parallel
- 10^3 ops/sec
- 10^{14} neurons; 10^{17} synapses
- analog/continuous
- embodied
- protein
- fault-tolerant
- . . .

THINGS AI HAS TRIED

- Try to program some activities thought to require intelligence
- Try to program some fundamental processes thought to be involved in intelligence
- Try to imitate the brain
- Try to simulate the performance of ever more complex biological organisms
- Try to simulate biological evolution
- Try to “educate” simple (child-like) programs to make them more intelligent and capable

Some more Artificial Intelligence

- Neural Networks
- Genetic Algorithms
- Genetic Programming
- Behavior-Based Systems

Background

- Neural Networks can be :

    - **Biological** models

    - **Artificial** models

- Desire to produce artificial systems capable of sophisticated computations similar to the human brain.

Biological analogy and some main ideas

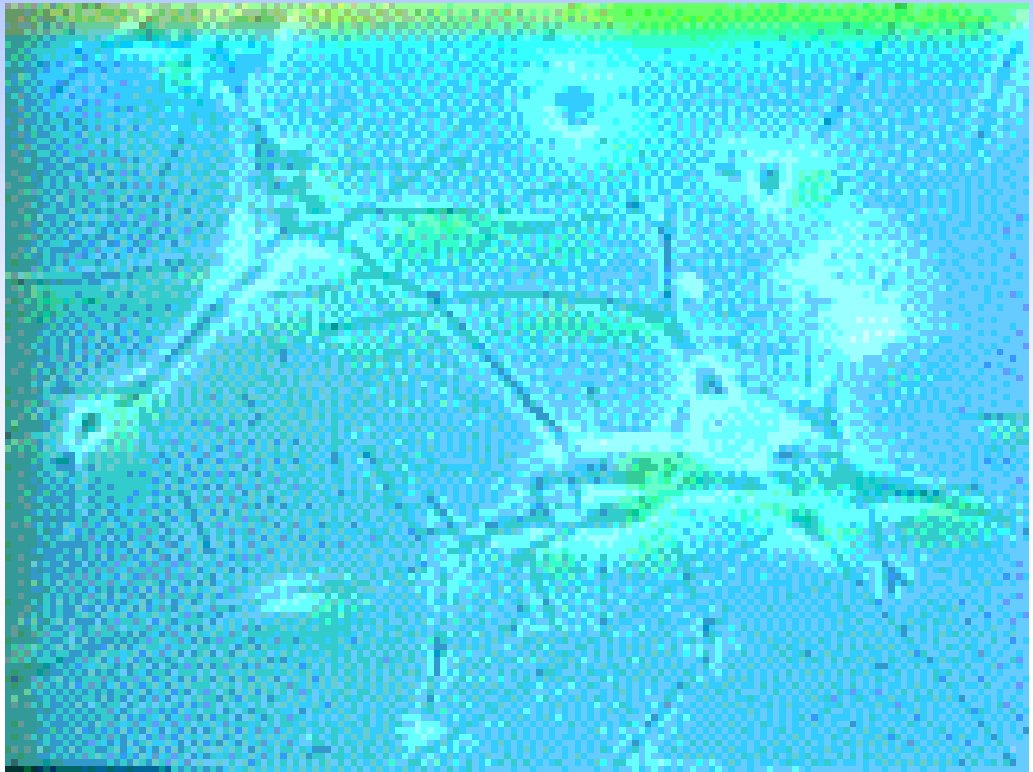
- The brain is composed of a mass of interconnected neurons
  - each neuron is connected to many other neurons
- Neurons transmit signals to each other
- Whether a signal is transmitted is an all-or-nothing event (the electrical potential in the cell body of the neuron is thresholded)
- Whether a signal is sent, depends on the strength of the bond (synapse) between two neurons

How Does the Brain Work ? (1)

NEURON

- The cell that performs information processing in the brain.

- Fundamental functional unit of all nervous system tissue.



How Does the Brain Work ? (2)

Each consists of :

SOMA, DENDRITES, AXON, and SYNAPSE.

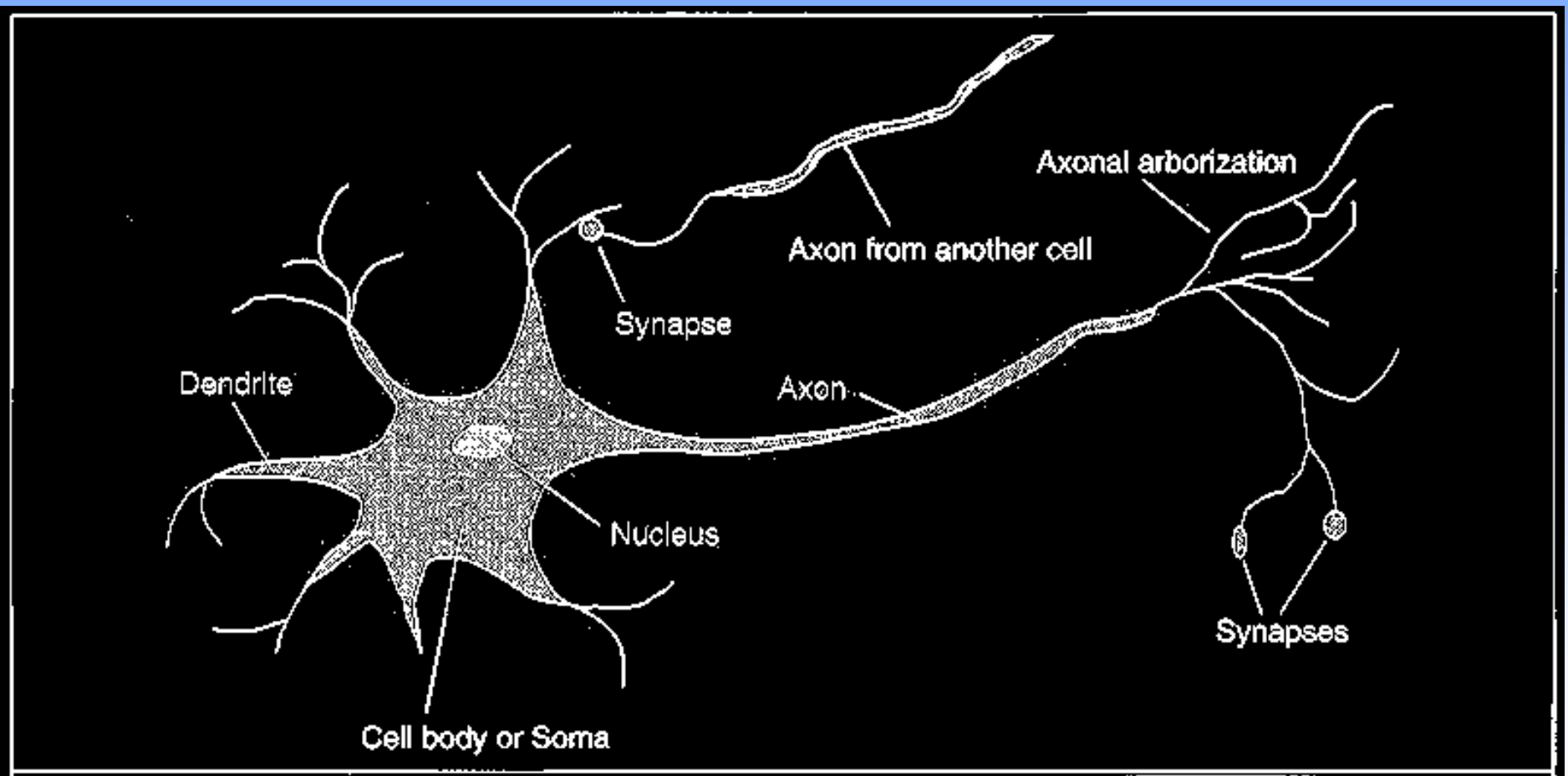


Figure 19.1 The parts of a nerve cell or neuron. In reality, the length of the axon should be about 100 times the diameter of the cell body.

Brain vs. Digital Computers (1)

- Computers require hundreds of cycles to simulate a firing of a neuron.

- The brain can fire all the neurons in a single step.
  ⇒ Parallelism

- Serial computers require billions of cycles to perform some tasks but the brain takes **less than a second.**
  e.g. Face Recognition

Comparison of Brain and computer

	Human	*Computer*
Processing Elements	100 Billion neurons	10 Million gates
Interconnects	1000 per neuron	A few
Cycles per sec	1000	500 Million
2X improvement	200,000 Years	2 Years

Brain vs. Digital Computers (2)

Future : combine parallelism of the brain with the switching speed of the computer.

	Computer	Human Brain
Computational units	1 CPU, 10^5 gates	10^{11} neurons
Storage units	10^9 bits RAM, 10^{10} bits disk	10^{11} neurons, 10^{14} synapses
Cycle time	10^{-8} sec	10^{-3} sec
Bandwidth	10^9 bits/sec	10^{14} bits/sec
Neuron updates/sec	10^5	10^{14}

Figure 19.2 A crude comparison of the raw computational resources available to computers (*circa* 1994) and brains.

History

- **1943:** McCulloch & Pitts show that **neurons** can be combined to construct a **Turing machine** (using ANDs, Ors, & NOTs)
- **1958:** Rosenblatt shows that **perceptrons** will converge if what they are trying to learn can be represented
- **1969:** Minsky & Papert showed the **limitations** of perceptrons, killing research for a decade
- **1985: backpropagation** algorithm revitalizes the field

Definition of Neural Network

A Neural Network is a system composed of many simple processing elements operating in parallel which can acquire, store, and utilize experiential knowledge.

What is Artificial Neural Network?

Neurons vs. Units (1)

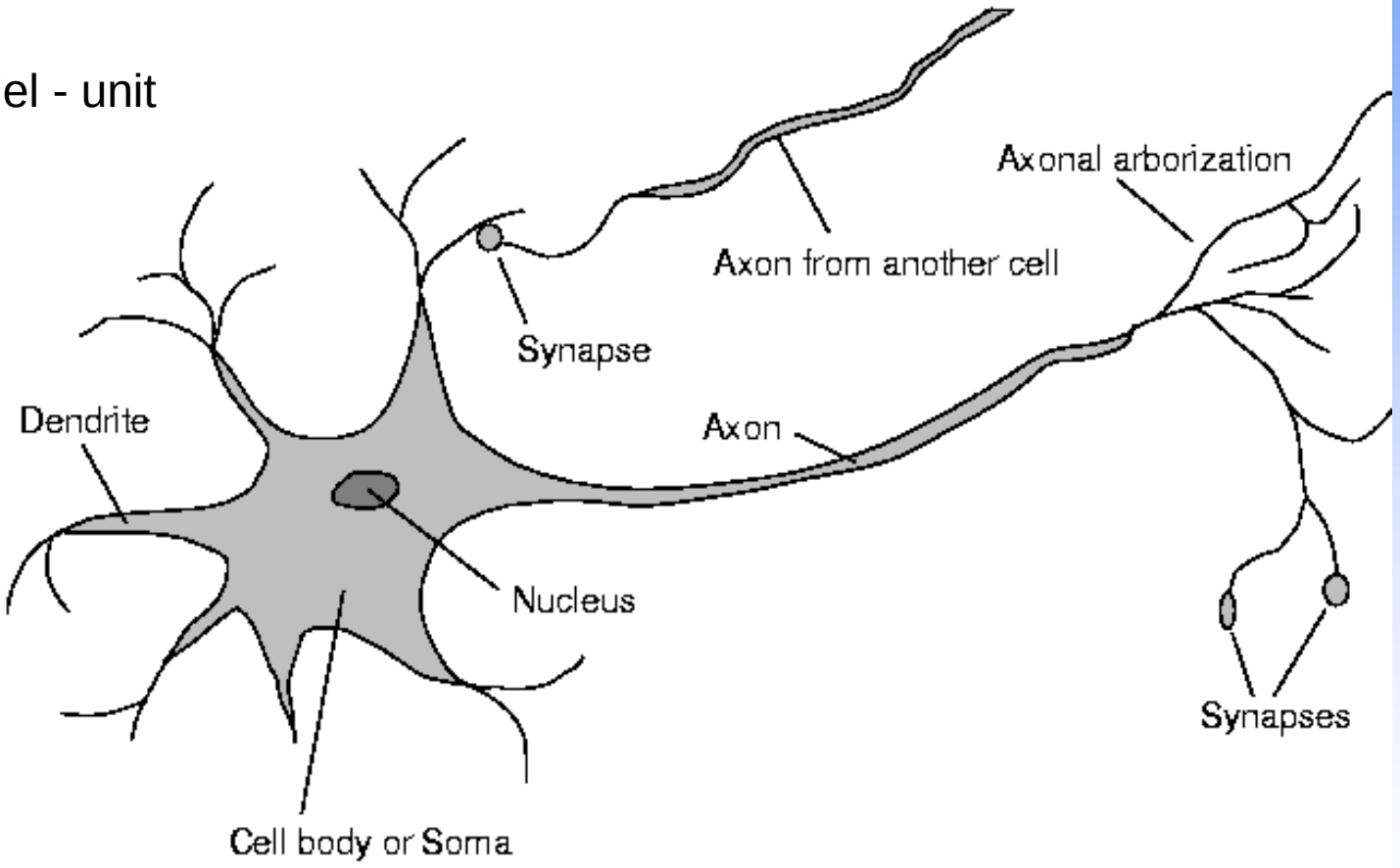
- Each element of NN is a node called unit.

- Units are connected by links.

- Each link has a numeric weight.

Neurons vs units (2)

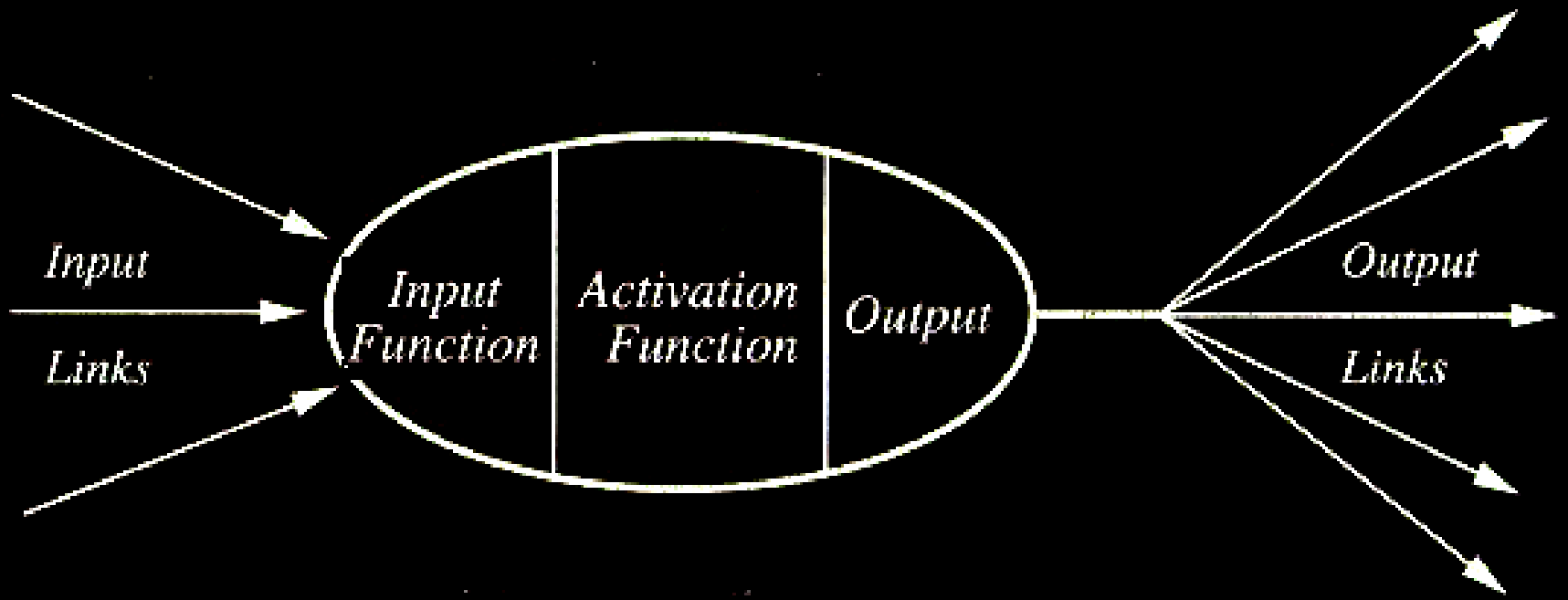
Real neuron is far away from our simplified model - unit



Chemistry, biochemistry, quantumness.

Computing Elements

A typical unit:



Planning in building a Neural Network

Decisions must be taken on the following:

- The number of units to use.

- The type of units required.

- Connection between the units.

How NN learns a task.

Issues to be discussed

- Initializing the weights.

- Use of a learning algorithm.

- Set of training examples.

- Encode the examples as inputs.

- Convert output into meaningful results.

Neural Network Example

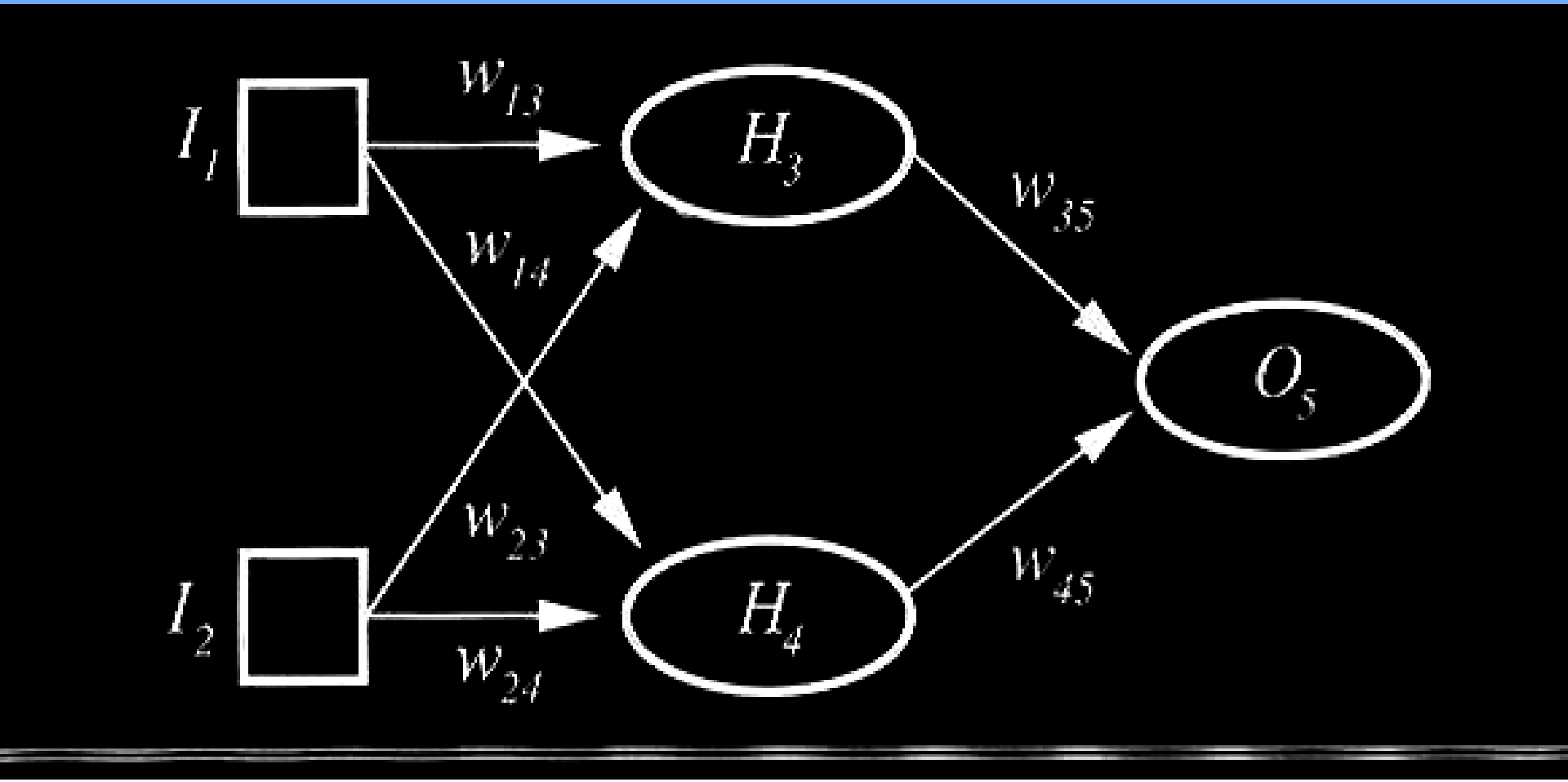


Figure 19.7. A very simple, two-layer, feed-forward network with two inputs, two hidden nodes, and one output node.

Simple Computations in this network

- There are **2 types of components:** Linear and Non-linear.

- **Linear:** Input function
  - calculate weighted sum of all inputs.

- **Non-linear:** Activation function
  - transform sum into activation level.

Calculations

Input function:

$$in_i = \sum_j W_{j,i} a_j = \mathbf{W}_i \cdot \mathbf{a}_j$$

Activation function **g**:

$$a_i \leftarrow g(in_i) = g\left(\sum_j W_{j,i} a_j\right)$$

A Computing Unit.

Now in more detail but for a particular model only

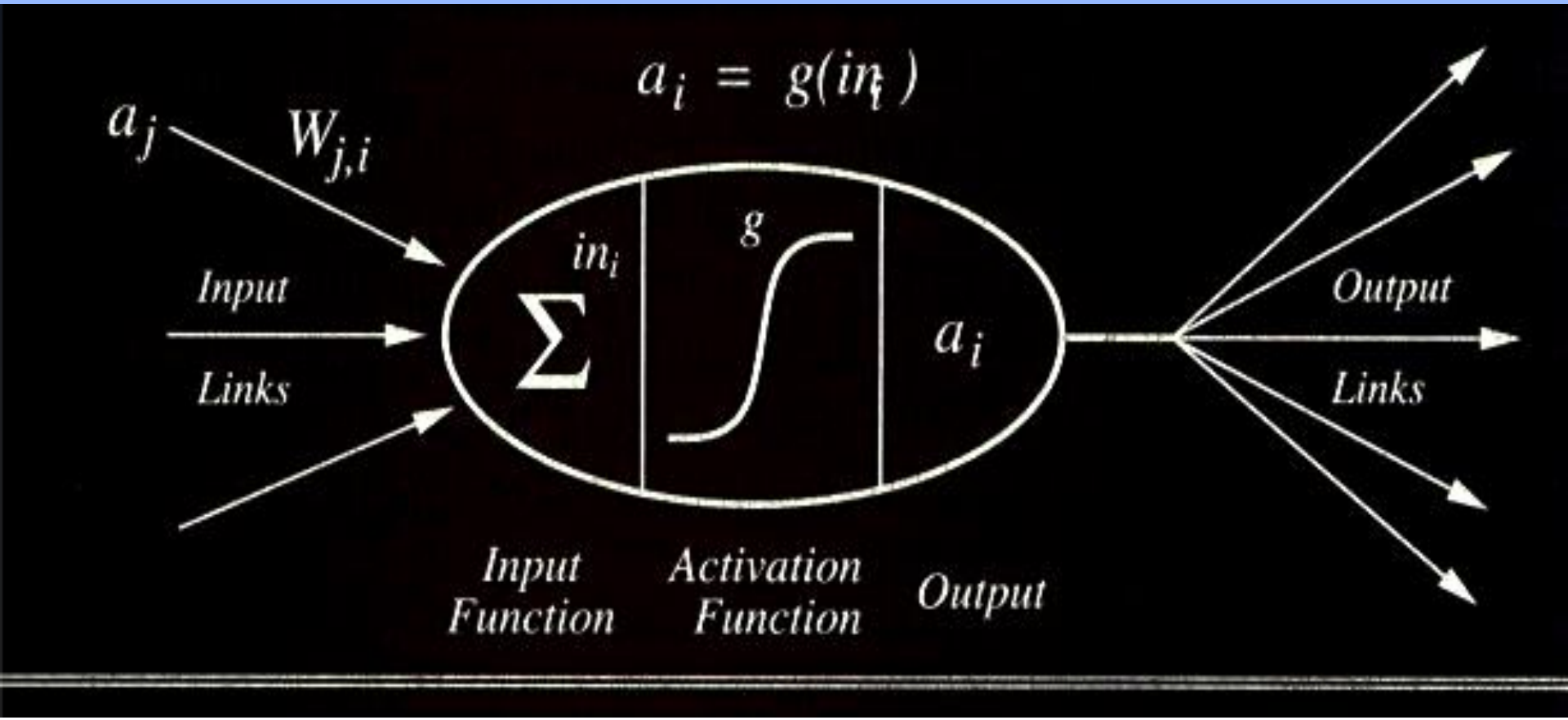


Figure 19.4. A unit

Activation Functions

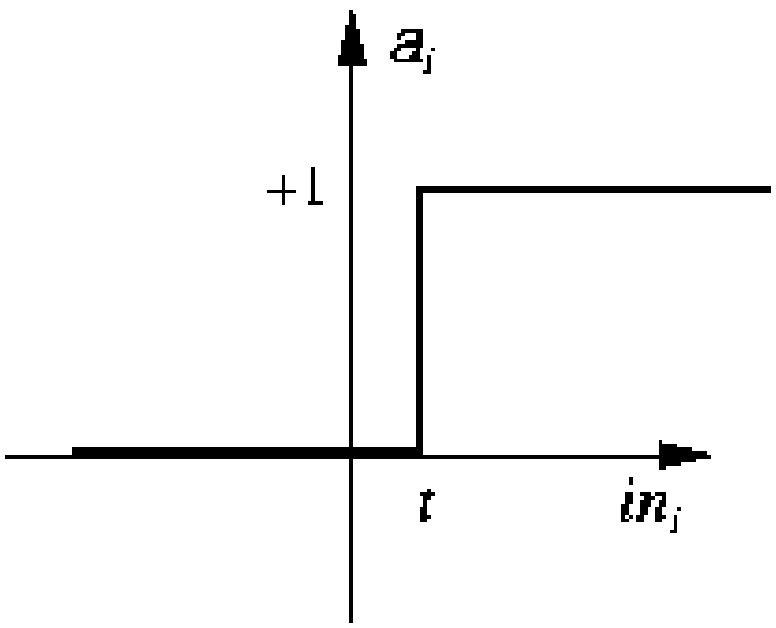
- Use different functions to obtain different models.

- 3 most common choices :

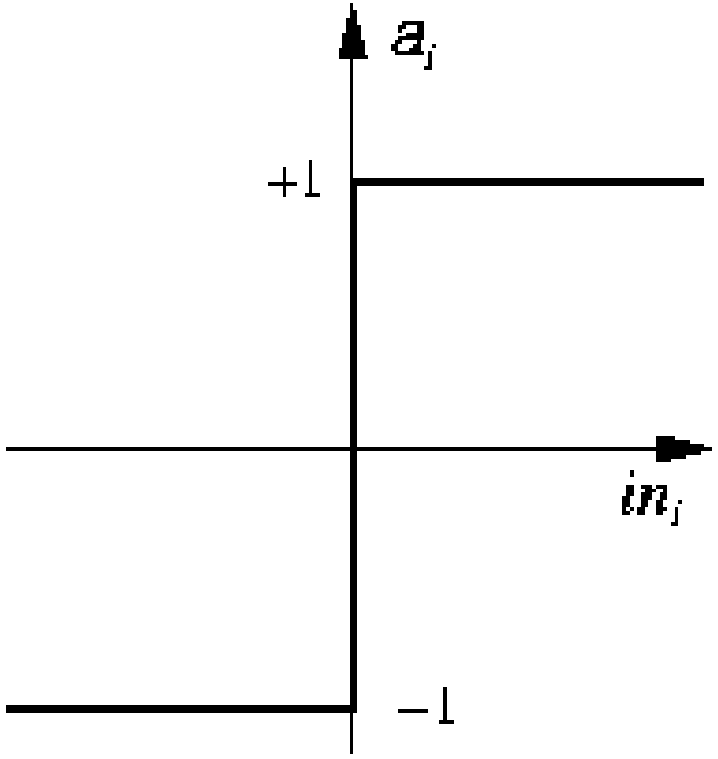
1) Step function
2) Sign function
3) Sigmoid function

- An output of **1 represents firing** of a neuron down the axon.

Step Function Perceptrons



(a) Step function



(b) Sign function

3 Activation Functions

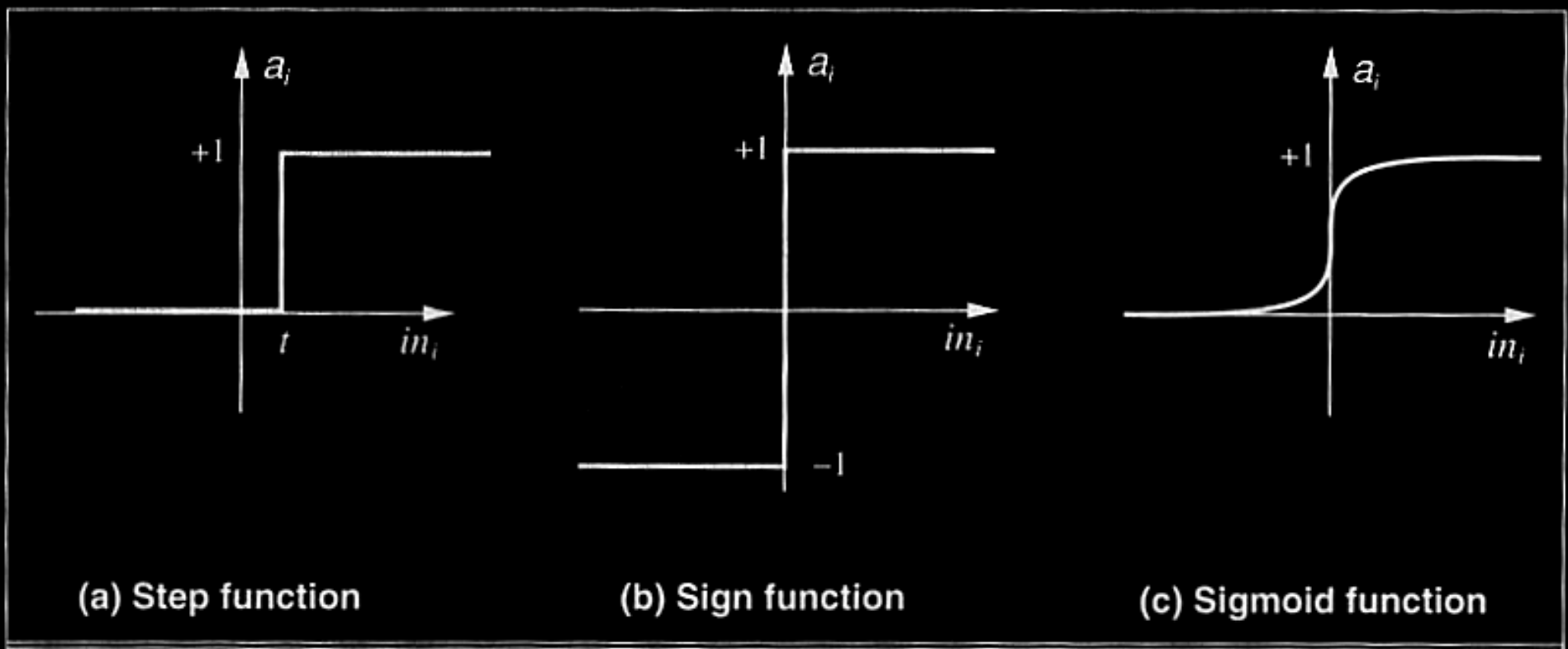


Figure 19.5 Three different activation functions for units.

$$\text{step}_t(x) = \begin{cases} 1, & \text{if } x \geq t \\ 0, & \text{if } x < t \end{cases} \qquad \text{sign}(x) = \begin{cases} +1, & \text{if } x \geq 0 \\ -1, & \text{if } x < 0 \end{cases} \qquad \text{sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

Are current computer a wrong model of thinking?

- Humans can't be doing the sequential analysis we are studying
  - Neurons are a million times slower than gates
  - Humans don't need to be rebooted or debugged when one bit dies.

100-step program constraint

- Neurons operate on the order of 10^{-3} seconds
- Humans can process information in a fraction of a second (face recognition)
- Hence, at most a couple of hundred serial operations are possible
- That is, even in parallel, no “chain of reasoning” can involve more than 100 -1000 steps

Standard structure of an artificial neural network

- Input units
  - represents the input as a fixed-length vector of numbers (user defined)
- Hidden units
  - calculate thresholded weighted sums of the inputs
  - represent intermediate calculations that the network learns
- Output units
  - represent the output as a fixed length vector of numbers

Representations

- **Logic rules**
  - If color = red ^ shape = square then +
- **Decision trees**
  - tree
- **Nearest neighbor**
  - training examples
- **Probabilities**
  - table of probabilities
- Neural networks
  - inputs in [0, 1]

Can be used for all of them

Many variants exist

Notations

Notation

Notation	Meaning
a_i $\mathbf{a}_i$	Activation value of unit i (also the output of the unit) Vector of activation values for the inputs to unit i
g g'	Activation function Derivative of the activation function
Err_i Err^e	Error (difference between output and target) for unit i Error for example e
I_i $\mathbf{I}$ $\mathbf{I}^e$	Activation of a unit i in the input layer Vector of activations of all input units Vector of inputs for example e

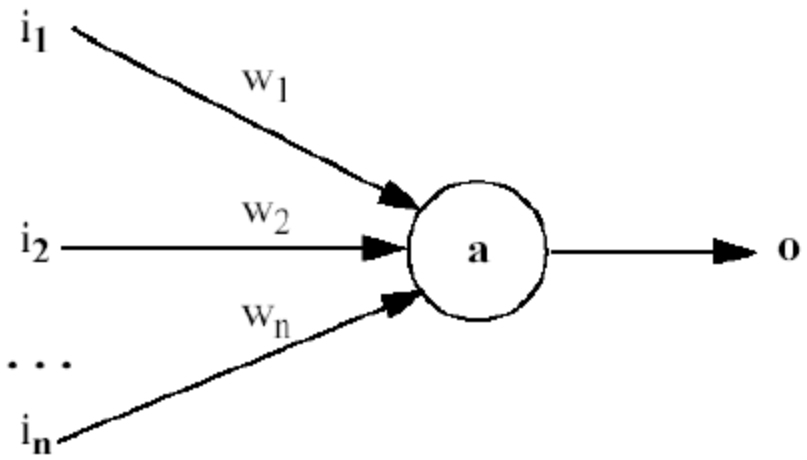
Notation (cont.)

in_i	Weighted sum of inputs to unit i
N	Total number of units in the network
O	Activation of the single output unit of a perceptron
O_i	Activation of a unit i in the output layer
$\mathbf{O}$	Vector of activations of all units in the output layer
t	Threshold for a step function
T	Target (desired) output for a perception
$\mathbf{T}$	Target vector when there are several output units
$\mathbf{T}^e$	Target vector for example e
$W_{j,i}$	Weight on the link from unit j to unit i
W_i	Weight from unit i to the output in a perception
$\mathbf{W}_i$	Vector of weights leading into unit i
$\mathbf{W}$	Vector of all weights in the network

Operation of individual units

- $Output_i = f(W_{i,j} * Input_j + W_{i,k} * Input_k + W_{i,l} * Input_l)$
  - where f(x) is a threshold (activation) function
  - $f(x) = 1 / (1 + e^{-Output})$
    - "sigmoid"
  - f(x) = step function

Artificial Neural Networks

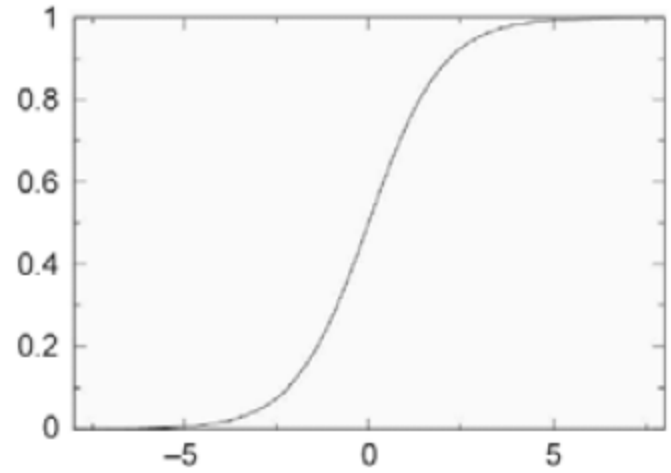


Activation

$$a(I, W) = \sum_{k=1}^{n} i_k \cdot w_k$$

Output

$$o(I, W) = \frac{1}{1 + e^{-\rho \cdot a(I, W)}}$$



Sigmoid
activation function

Units in Action

- Individual units representing Boolean functions

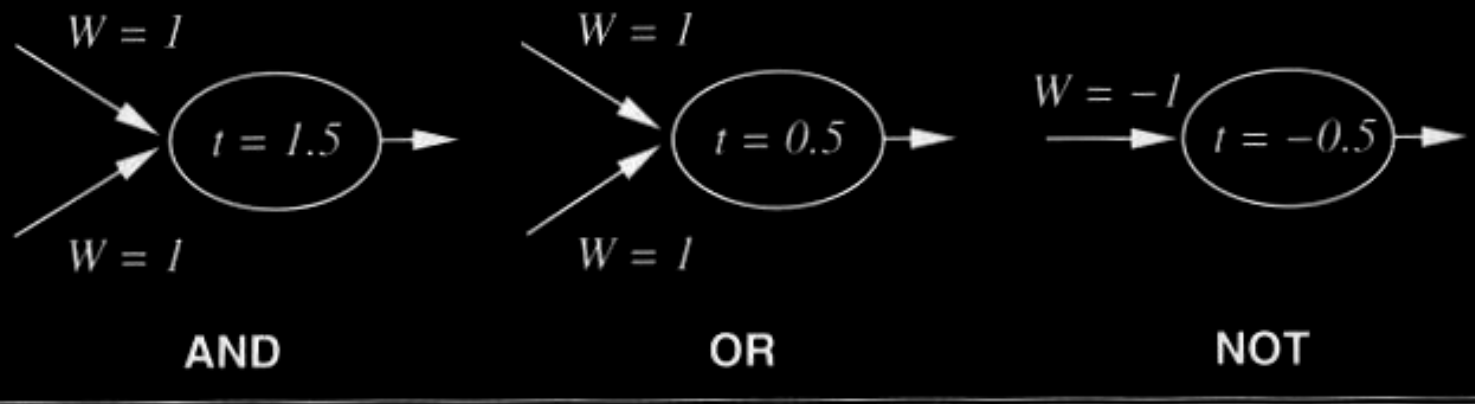


Figure 19.6 Units with a step function for the activation function can act as logic gates, given appropriate thresholds and weights.

Network Structures

Feed-forward neural nets:

Links can only go in one direction.

Recurrent neural nets:

Links can go anywhere and form arbitrary topologies.

Feed-forward Networks

- Arranged in ***layers***.

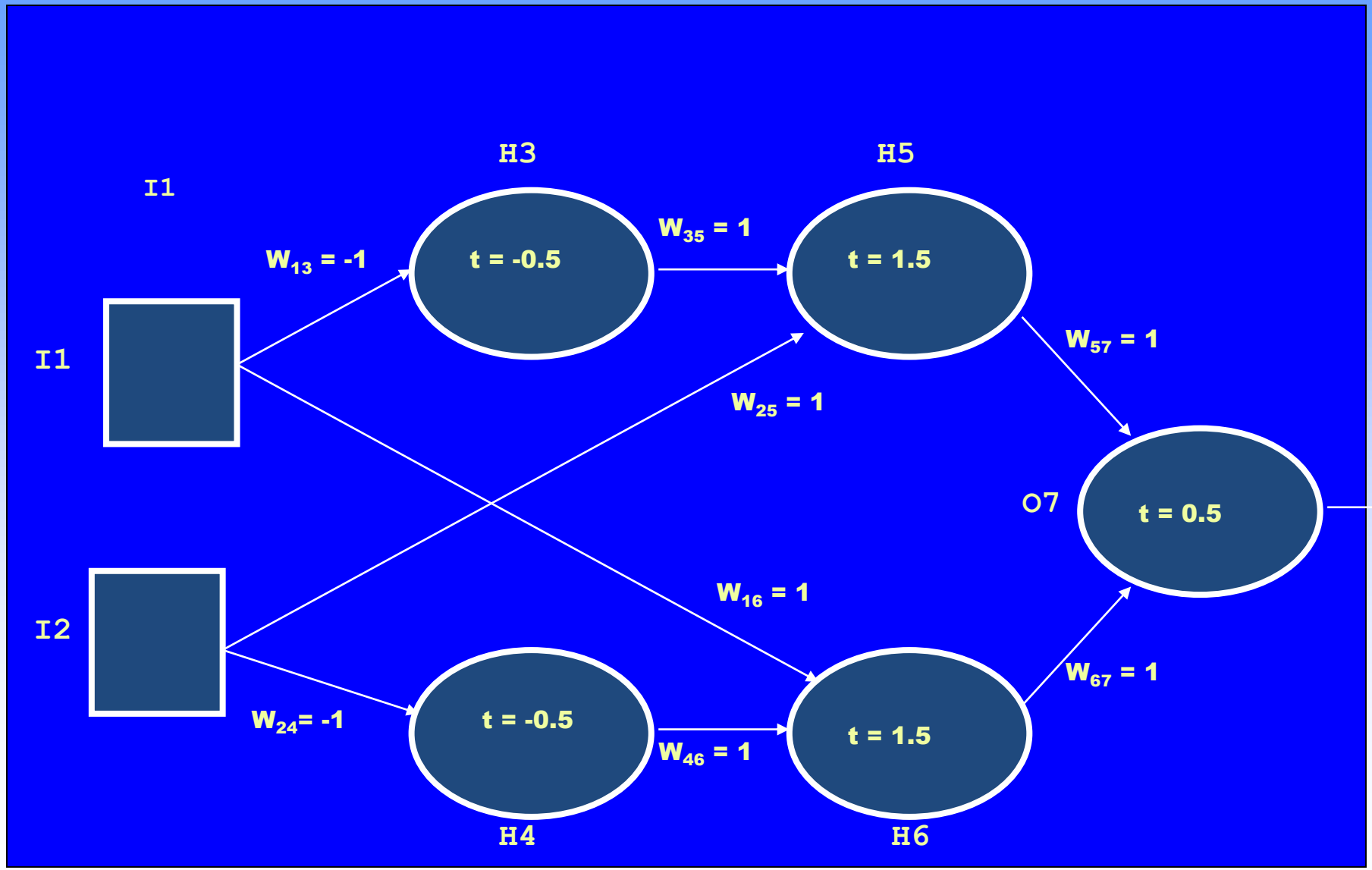
- Each unit is linked only in the unit in next layer.

- No units are linked between the same layer, back to the previous layer or skipping a layer.

- Computations can proceed uniformly from input to output units.

- No internal state exists.

Feed-Forward Example



Inputs skip the layer in this case

Multi-layer Networks and Perceptrons

- Have one or more layers of hidden units.

- With two possibly very large hidden layers, it is possible to implement any function.

- Networks without hidden layer are called perceptrons.

- Perceptrons are very limited in what they can represent, but this makes their learning problem much simpler.

Recurrent Network (1)

- The brain is not and cannot be a feed-forward network.

- Allows activation to be fed back to the previous unit.

- Internal state is stored in its activation level.

- Can become unstable

-Can oscillate.

Recurrent Network (2)

- May take long time to compute a stable output.

- Learning process is much more difficult.

- Can implement more complex designs.

- Can model certain systems with internal states.

Perceptrons

Perceptrons

- First studied in the late 1950s.

- Also known as Layered Feed-Forward Networks.

- The only efficient learning element at that time was for single-layered networks.

- Today, used as a synonym for a single-layer, feed-forward network.

Perceptron

Cornell Aeronautical Laboratory

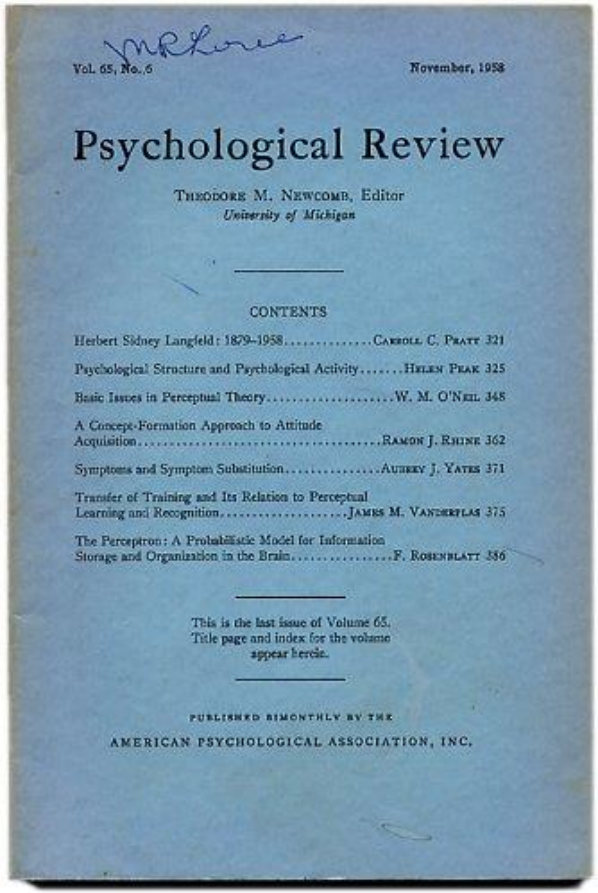
Rosenblatt &
Mark I Perceptron:
the first machine that could "learn" to recognize and identify optical patterns.

Perceptron

- Invented by Frank Rosenblatt in 1957 in an attempt to understand human memory, learning, and cognitive processes.
- The first neural network model by computation, with a remarkable learning algorithm:
  - If function can be represented by perceptron, the learning algorithm is guaranteed to quickly converge to the hidden function!
- Became the foundation of pattern recognition research

One of the earliest and most influential neural networks:
An important milestone in AI.

Perceptron



Vol. 65, No. 6 November, 1958

Psychological Review

THEODORE M. NEWCOMB, Editor
University of Michigan

CONTENTS

This is the last issue of Volume 65. Title page and index for the volume appear herein.

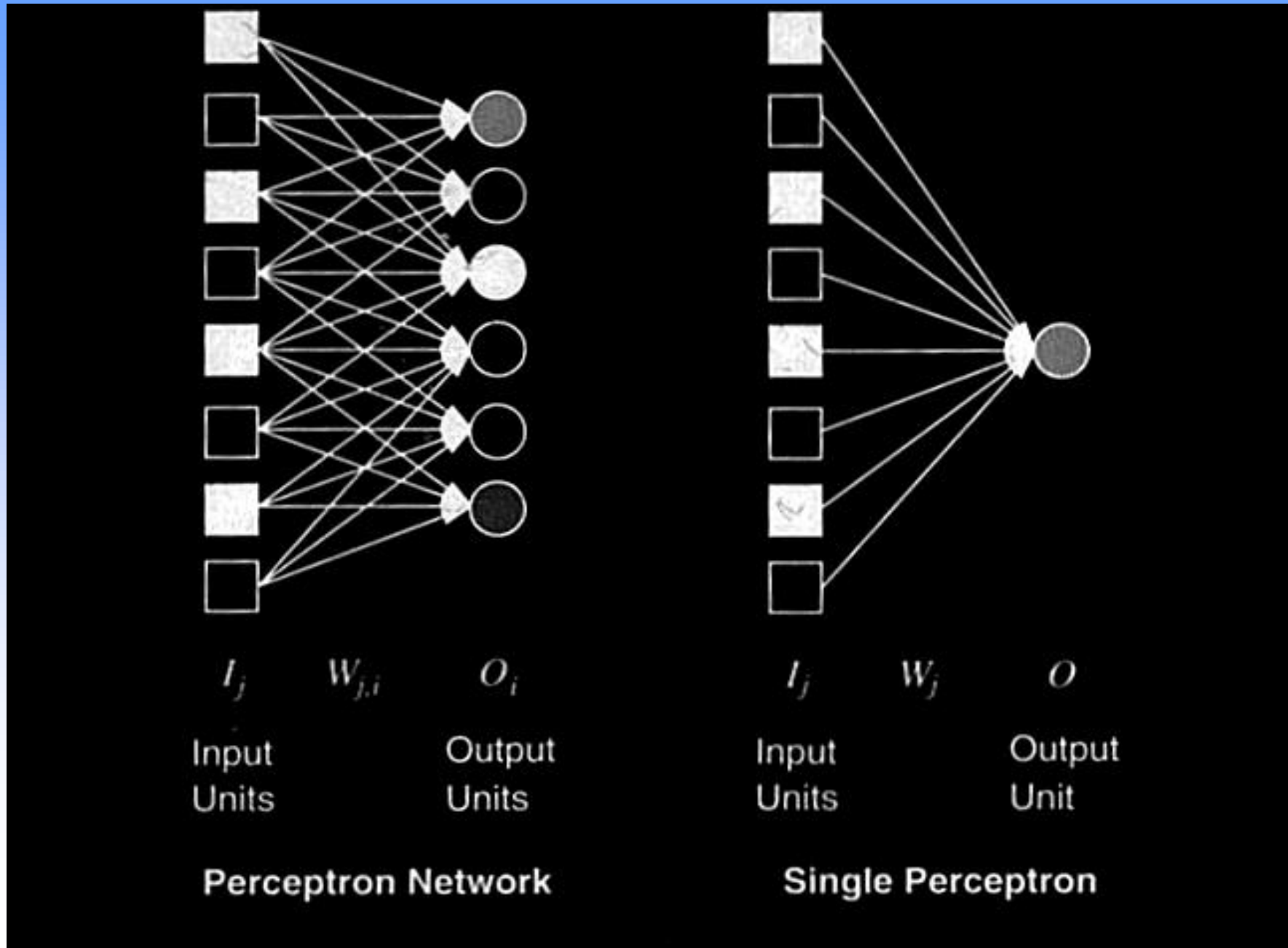
PUBLISHED BIMONTHLY BY THE
AMERICAN PSYCHOLOGICAL ASSOCIATION, INC.

ROSENBLATT, Frank.
(Cornell Aeronautical Laboratory at Cornell University)

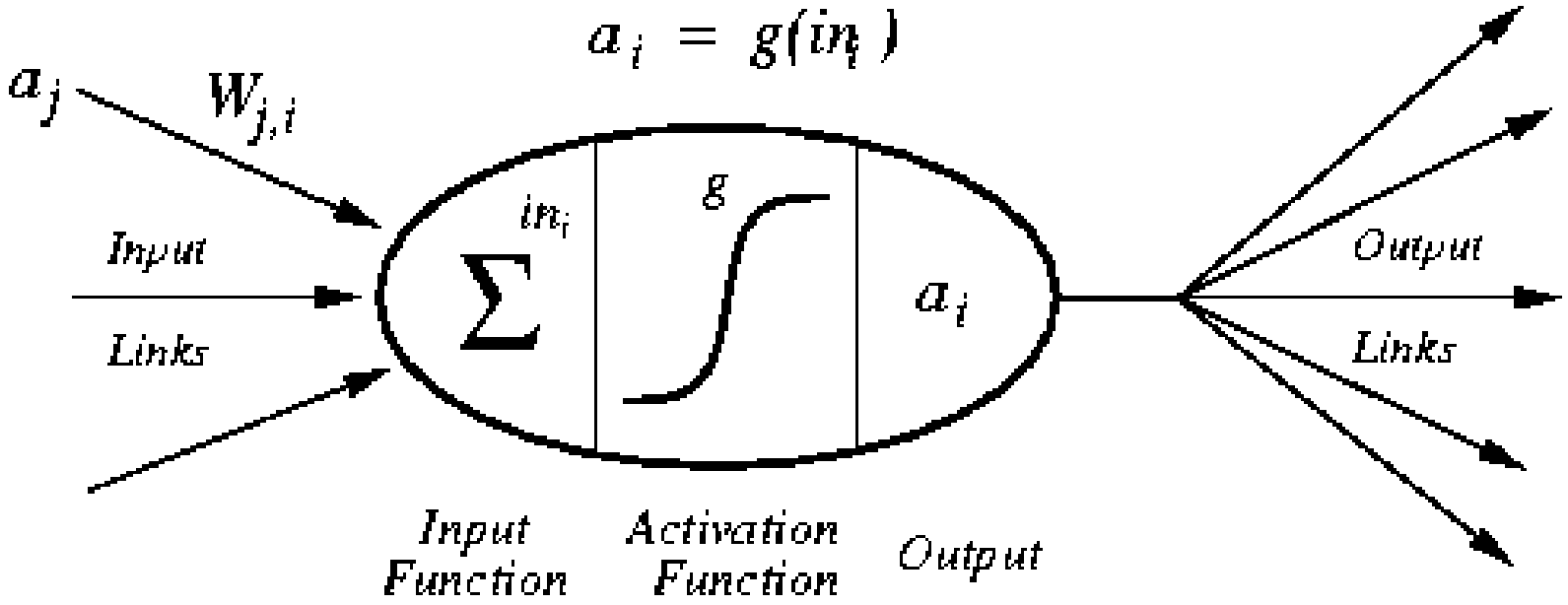
The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain.

In, Psychological Review, Vol. 65, No. 6, pp. 386-408, November, 1958.

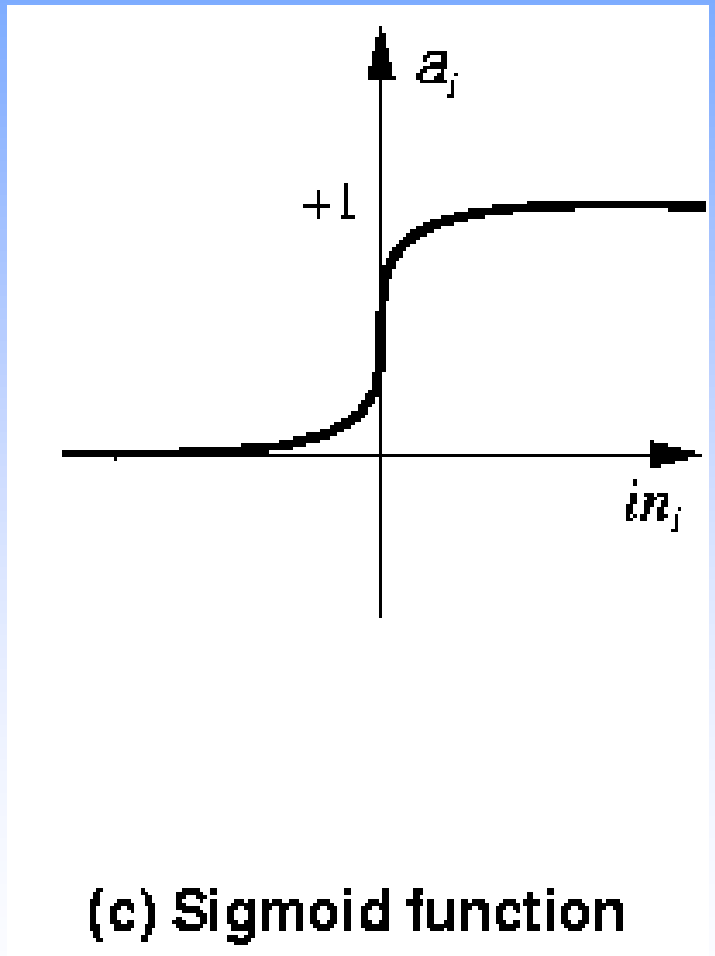
Fig. 19.8. Perceptrons



Perceptrons



Sigmoid Perceptron



(c) Sigmoid function

Perceptron learning rule

- Teacher specifies the **desired output** for a given input
- Network calculates what it thinks the output should be
- Network changes its weights in proportion to the error between the desired & calculated results
- $\Delta w_{i,j} = \alpha * [teacher_i - output_i] * input_j$
  - where:
  - α is the learning rate;
  - $teacher_i - output_i$ is the error term;
  - and $input_j$ is the input activation
- $w_{i,j} = w_{i,j} + \Delta w_{i,j}$

Delta rule

Adjusting perceptron weights

- $\Delta w_{i,j} = \alpha * [teacher_i - output_i] * input_j$
- $miss_i$ is $(teacher_i - output_i)$

	miss<0	miss=0	miss>0
input < 0	alpha	0	-alpha
input = 0	0	0	0
input > 0	-alpha	0	alpha

- Adjust each $w_{i,j}$ based on $input_j$ and $miss_i$
- The above table shows adaptation.
- Incremental learning.

Node biases

- A node's output is a weighted function of its inputs

- What is a bias?
- How can we learn the bias value?

- Answer: treat them like just another weight

Training biases (Θ)

- A node's output:
  - 1 if $w_1x_1 + w_2x_2 + \ldots + w_nx_n >= \Theta$
  - 0 otherwise

bias

- Rewrite
  - $w_1x_1 + w_2x_2 + \ldots + w_nx_n - \Theta >= 0$
  - $w_1x_1 + w_2x_2 + \ldots + w_nx_n + \Theta(-1) >= 0$

- Hence, the bias is just another weight whose activation is always -1

- Just add one more input unit to the network topology

Perceptron convergence theorem

- If a set of <input, output> pairs are learnable (representable), the delta rule will find the necessary weights
  - in a finite number of steps
  - independent of initial weights
- However, a single layer perceptron can only learn linearly separable concepts
  - it works **iff** gradient descent works

What can Perceptrons Represent ?

- Some complex Boolean function can be represented.

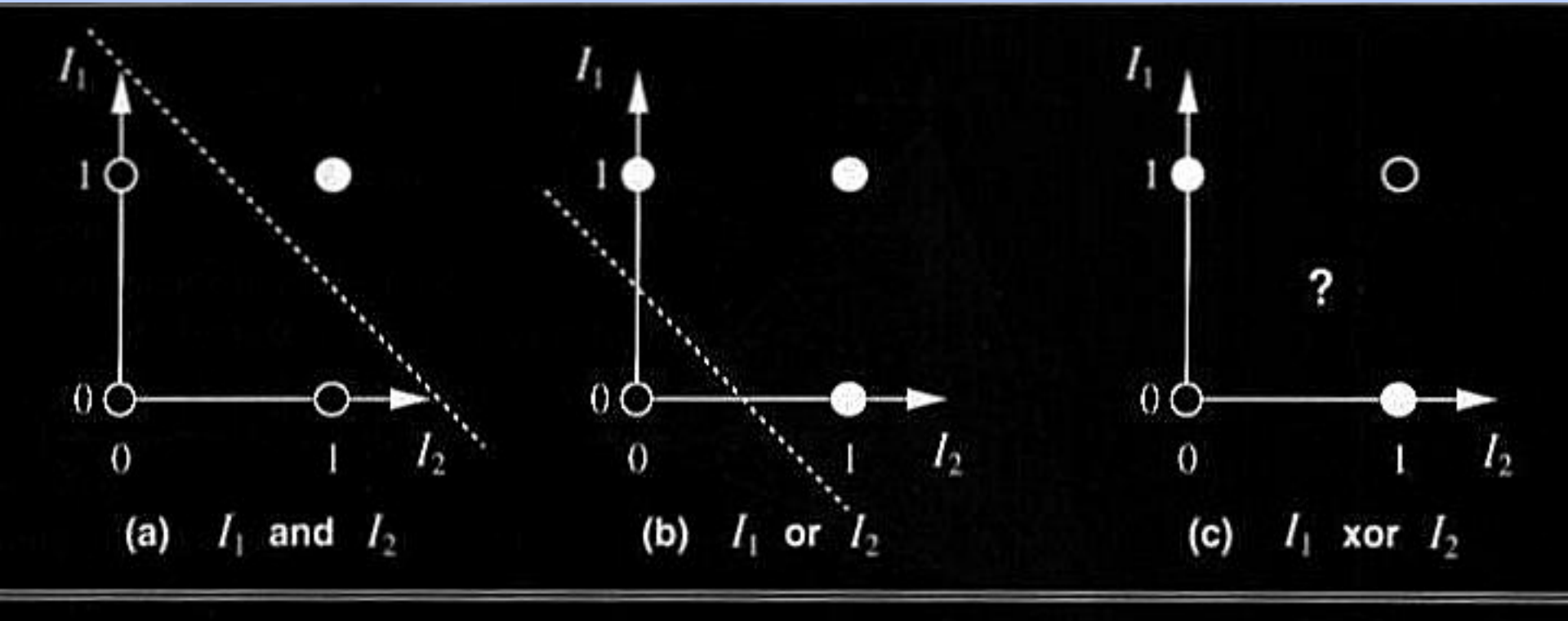
For example:

Majority function - will be covered in this lecture.

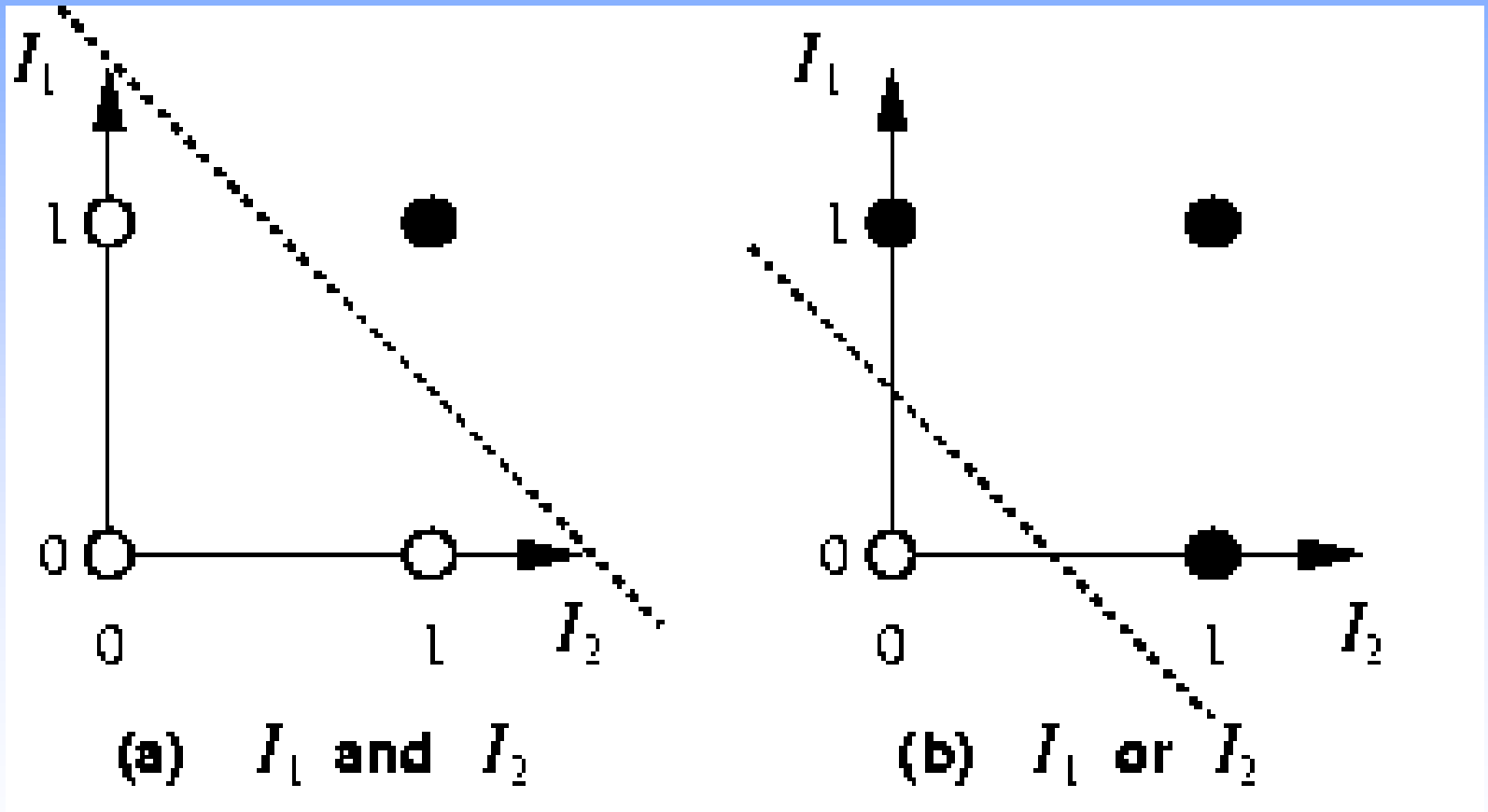
- Perceptrons are limited in the Boolean functions they can represent.

The Separability Problem and EXOR trouble

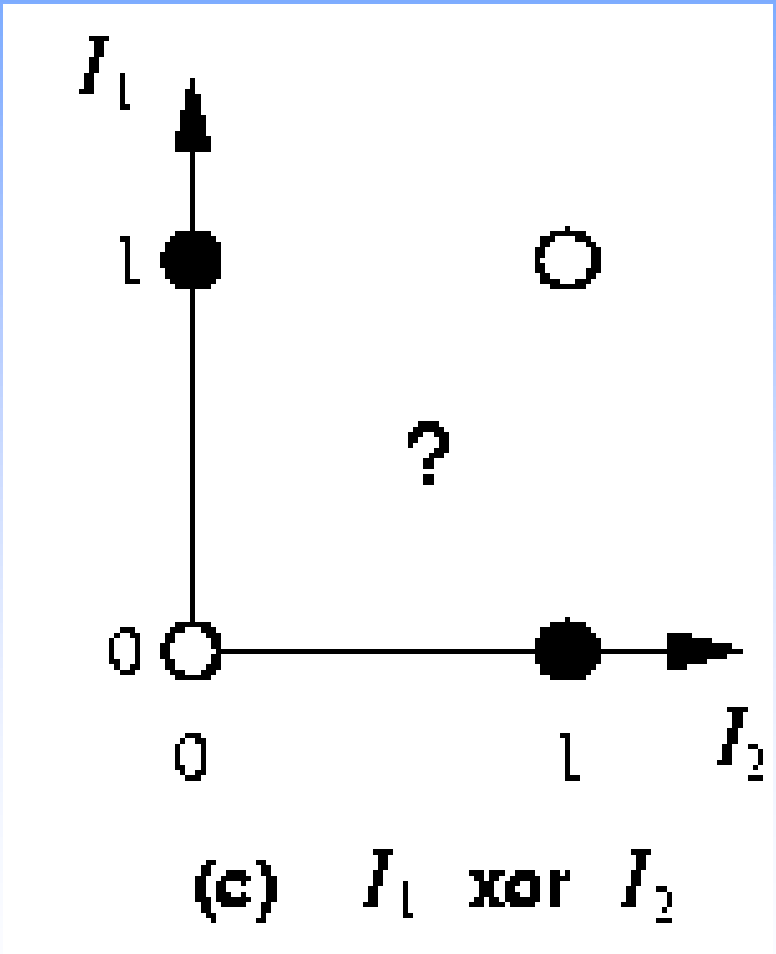
Figure 19.9. Linear Separability in Perceptrons



AND and OR linear Separators

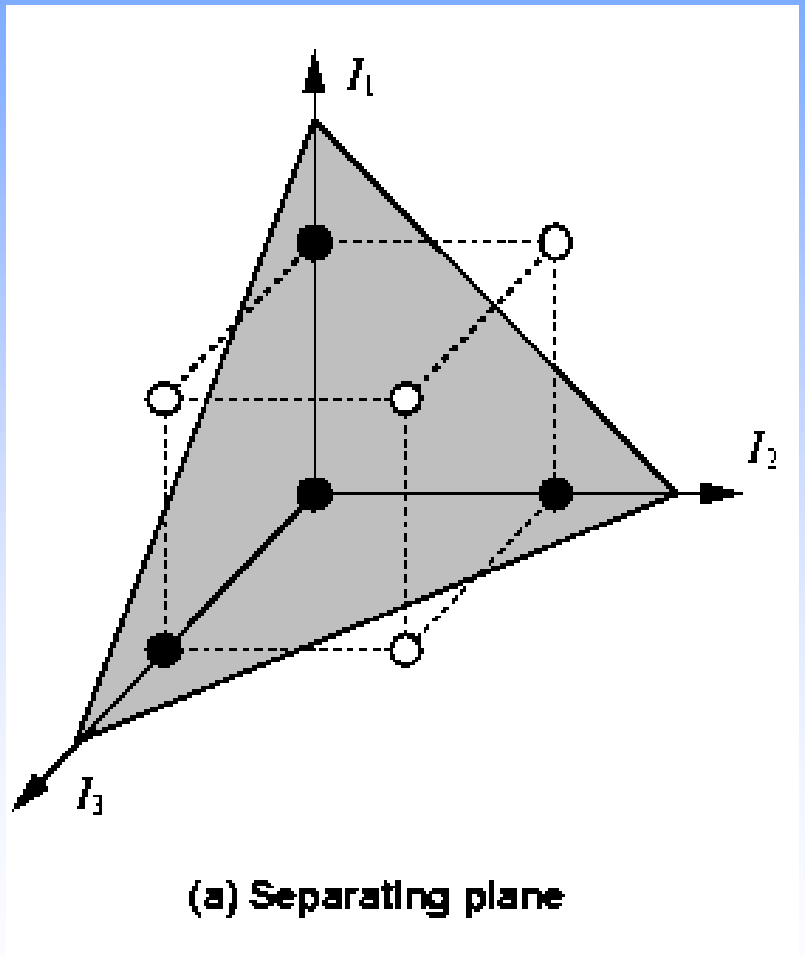


How do we compute XOR?



Separation in n-1 dimensions

majority



(a) Separating plane

Example of 3Dimensional space

Perceptrons & XOR

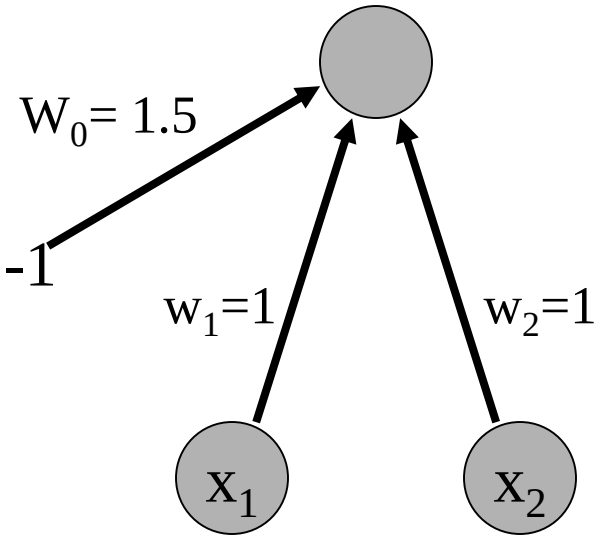
- XOR function

Input1	Input2	Output
0	0	0
0	1	1
1	0	1
1	1	0

  - no way to draw a line to separate the positive from negative examples

Boolean AND

input x1	input x2	ouput
0	0	0
0	1	0
1	0	0
1	1	1

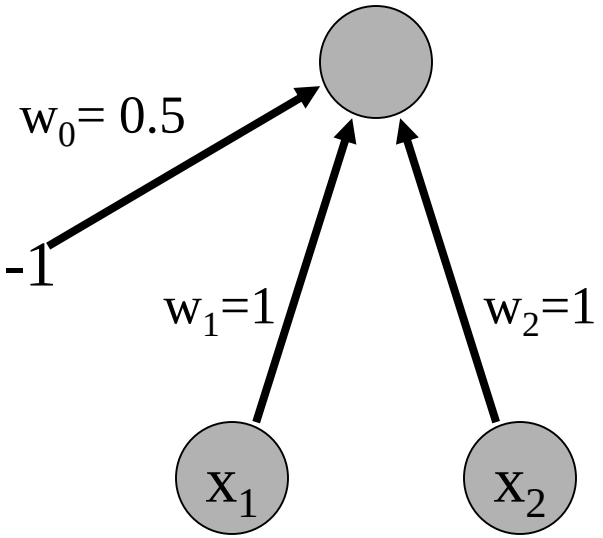


Activation of threshold units when:

$$\sum_{j=1}^{n} W_{j,i} a_j > W_{0,i}$$

Boolean OR

input x1	input x2	ouput
0	0	0
0	1	1
1	0	1
1	1	1

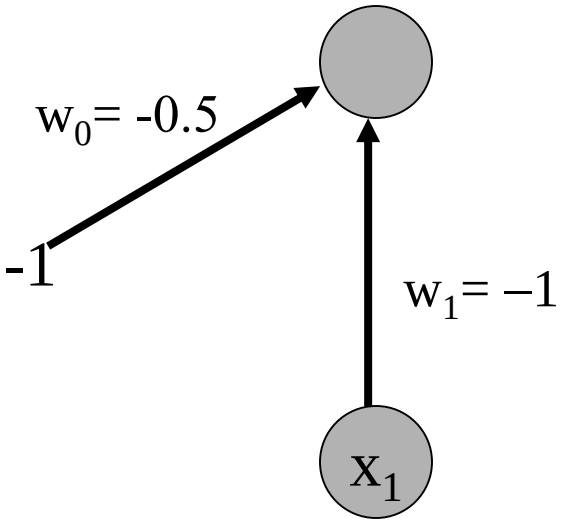


Activation of threshold units when:

$$\sum_{j=1}^{n} W_{j,i} a_j > W_{0,i}$$

Inverter

input x1	output
0	1
1	0

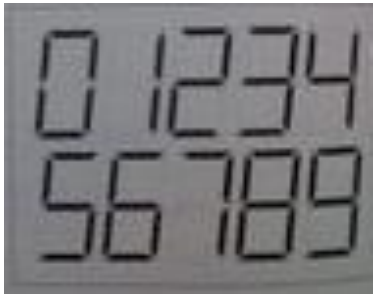


Activation of threshold units when:

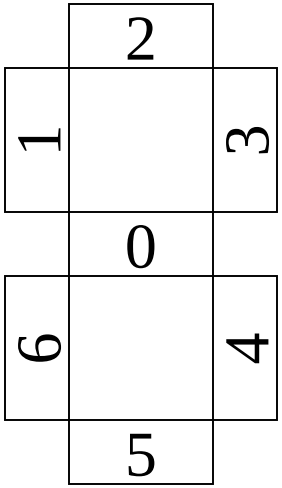
$$\sum_{j=1}^{n} W_{j,i} a_j > W_{0,i}$$

So, units with a threshold activation function can act as logic gates given the appropriate input and bias weights.

Perceptron to Learn to Identify Digits
(From Pat. Winston, MIT)

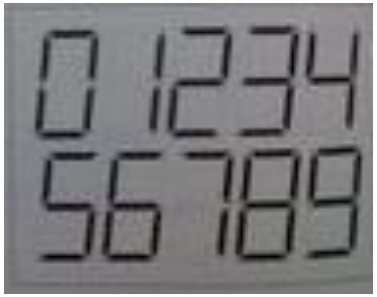


Seven line segments are enough to produce all 10 digits

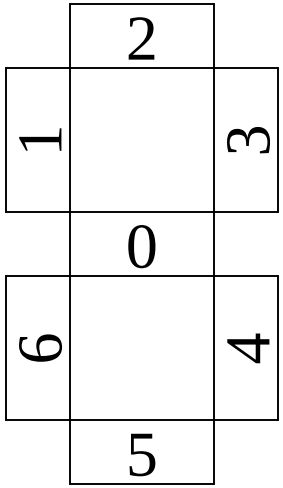


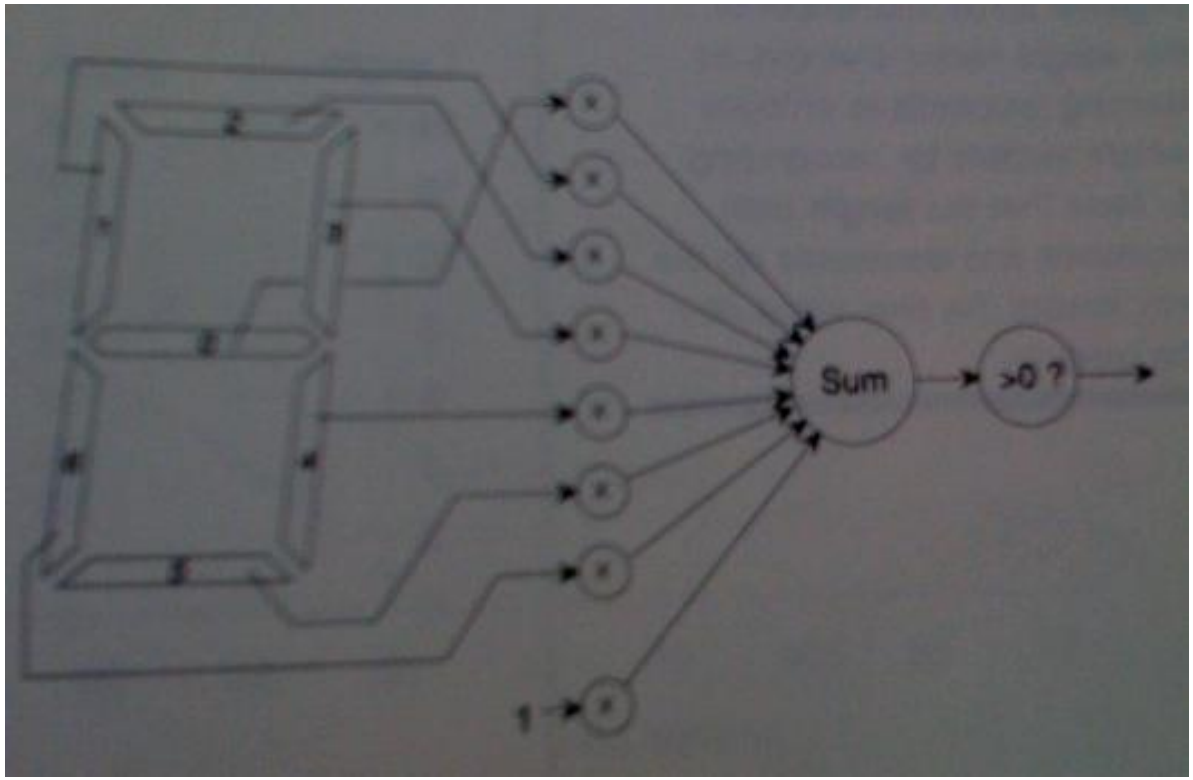
Digit	x_0	x_1	x_2	x_3	x_4	x_5	x_6
0	0	1	1	1	1	1	1
9	1	1	1	1	1	1	0
8	1	1	1	1	1	1	1
7	0	0	1	1	1	0	0
6	1	1	1	0	1	1	1
5	1	1	1	0	1	1	0
4	1	1	0	1	1	0	0
3	1	0	1	1	1	1	0
2	1	0	1	1	0	1	1
1	0	0	0	1	1	0	0

Perceptron to Learn to Identify Digits
(From Pat. Winston, MIT)



Seven line segments are enough to produce all 10 digits

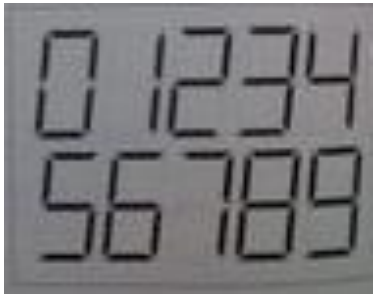




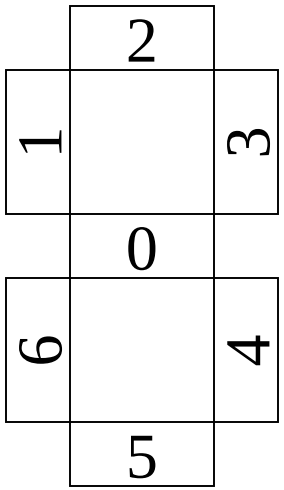
A vision system reports which of the seven segments in the display are on, therefore producing the inputs for the perceptron.

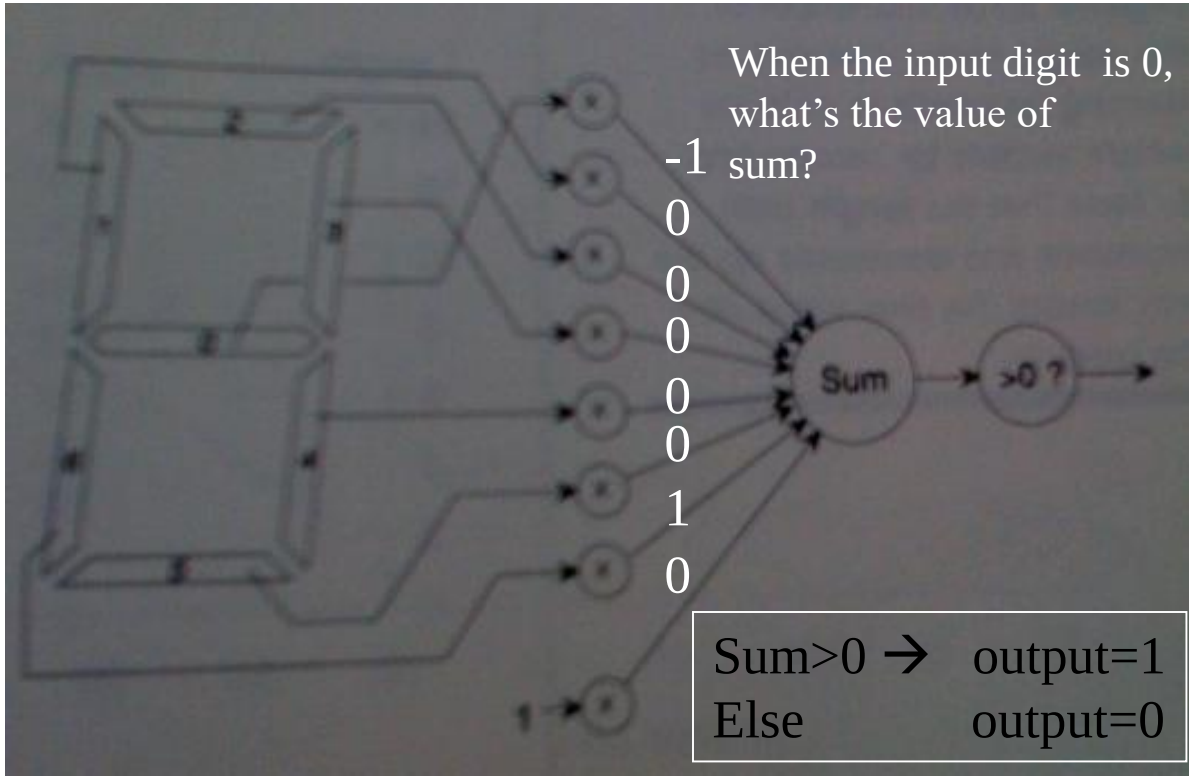
Perceptron to Learn to Identify Digit 0

Digit	x_0	x_1	x_2	x_3	x_4	x_5	x_6	X_7 (fixed input)
0	0	1	1	1	1	1	1	1



Seven line segments are enough to produce all 10 digits





A vision system reports which of the seven segments in the display are on, therefore producing the inputs for the perceptron.

Derivation of a learning rule for Perceptrons Minimizing Squared Errors

Threshold perceptrons have some advantages , in particular

→ Simple learning algorithm that fits a threshold perceptron to any linearly separable training set.

Key idea: Learn by adjusting weights to reduce error on training set.

→ update weights repeatedly (epochs) for each example.

We'll use:

→Sum of squared errors (e.g., used in linear regression), classical error measure

→Learning is an optimization search problem in weight space.

Derivation of a learning rule for Perceptrons Minimizing Squared Errors

Let $S = \{(\mathbf{x}_i, y_i): i = 1, 2, ..., m\}$ be a training set. (Note, **x** is a vector of inputs, and y is the vector of the true outputs.)

Let $h_{\mathbf{w}}$ be the perceptron classifier represented by the weight vector **w**.

Definition:

$$E(\mathbf{x}) = Squared\ Error(\mathbf{x}) = \frac{1}{2}(y - h_{\mathbf{w}}(\mathbf{x}))^2$$

Derivation of a learning rule for Perceptrons Minimizing Squared Errors

The squared error for a single training example with input **x** and true output y is:

$$E = \frac{1}{2} Err^2 \equiv \frac{1}{2}(y - h_{\mathbf{W}}(\mathbf{x}))^2 ,$$

Where $h_{\mathbf{w}}$ **(x)** is the output of the perceptron on the example and y is the true output value.

We can use the gradient descent to reduce the squared error by calculating the partial derivatives of E with respect to each weight.

$$\frac{\partial E}{\partial W_j} = Err \times \frac{\partial Err}{\partial W_j} = Err \times \frac{\partial}{\partial W_j}\left(y - g(\Sigma_{j=0}^{n} W_j x_j)\right)$$
$$= -Err \times g'(in) \times x_j$$

Note: g'(in) derivative of the activation function. For sigmoid g'=g(1-g). For threshold perceptrons, Where g'(n) is undefined, the original perceptron rule simply omitted it.

$$\frac{\partial E}{\partial W_j} = -Err \times g'(in) \times x_j$$

Gradient descent algorithm → we want to reduce , *E*, *f*or each weight w_i , change weight in direction of steepest descent:

$$W_j \leftarrow W_j + \alpha \times Err \times g'(in) \times x_j$$

α learning rate

$$W_j \leftarrow W_j + \alpha \times I_j \times Err$$

Intuitively:

Err = y – h_W(x) positive

output is too small → weights are increased for positive inputs and decreased for negative inputs.

Err = y – h_W(x) negative

→ opposite

Perceptron Learning: Intuition

Rule is intuitively correct!

Greedy Search:

Gradient descent through weight space!

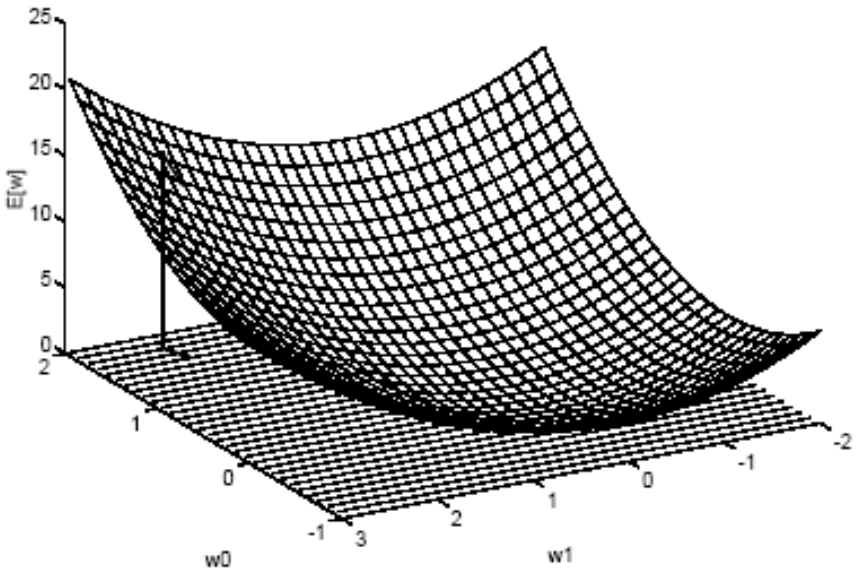
Surprising proof of convergence:

Weight space has no local minima!

With enough examples, it will find the target function!

(provide α not too large)

Gradient descent in weight space



From T. M. Mitchell, *Machine Learning*

$$W_j \leftarrow W_j + \alpha \times I_j \times Err$$

Perceptron learning rule:

1. Start with random weights, $\mathbf{w} = (w_1, w_2, \ldots, w_n)$.
2. Select a training example $(\mathbf{x}, y) \in S$.
3. Run the perceptron with input **x** and weights **w** to obtain g
4. Let α be the training rate (a user-set parameter).

$$\forall w_i,\ w_i \leftarrow w_i + \Delta w_i,$$

where

$$\Delta w_i = \alpha(y - g(in))g'(in)x_i$$

5. Go to 2.

Epoch → cycle through the examples

Epochs are repeated until some stopping criterion is reached—typically, that the weight changes have become very small.

The stochastic gradient method selects examples randomly from the training set rather than cycling through them.

Perceptron Learning: Gradient Descent Learning Algorithm

function PERCEPTRON-LEARNING(*examples*, *network*) **returns** a perceptron hypothesis
inputs: *examples*, a set of examples, each with input $\mathbf{x} = x_1, \ldots, x_n$ and output y
network, a perceptron with weights $W_j,\ j = 0 \ldots n$, and activation function g

repeat
for each e **in** *examples* **do**
$in \leftarrow \sum_{j=0}^{n} W_j\, x_j[e]$
$Err \leftarrow y[e] - g(in)$
$W_j \leftarrow W_j + \alpha \times Err \times g'(in) \times x_j[e]$
until some stopping criterion is satisfied
return NEURAL-NET-HYPOTHESIS(*network*)

Figure 20.21 The gradient descent learning algorithm for perceptrons, assuming a differentiable activation function g. For threshold perceptrons, the factor $g'(in)$ is omitted from the weight update. NEURAL-NET-HYPOTHESIS returns a hypothesis that computes the network output for any given example.