Neural Networks

The adaptive capacity give organisms the ability to learn and to remember

Learning is the process of acquiring new information while memory refers to the persistence of learning in a state that can be revealed at a later time

What is a Neural Network?

What is a Neural Network?

Neurons: **Structural Constituents** of the **Brain** as introduced by Ramon j Cajal

Much slower than electronic logic gates

The brain makes up this deficiency by having an enormous number of neurons with massive interconnections between them

The brain is regarded as highly complex, nonlinear and parallel computer

A neural network is a massively parallel distributed processor that has a natural propensity for storing eperimental knowledge and making it available for use. It resebles the brain in two respects:

1. Knowledge is acquiried by the network through a LEARNING PROCESS

2. Interconnection strengths known as synaptic weights are used to store the KNOWLEDGE

The procedure used to perform the learning process is called a learning algorithm;

It defines an interdisciplinary culture which is based on brain like learning opposing to traditional computing based on programming

Benefits

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

II. FUNDAMENTAL CONCEPTS of ARTIFICIAL NEURAL NETWORKS

Questions to be answered in this course:

- How can a network be trained efficiently?
- Why does it learn?
- What models of neurons need to be used for best performance?
- What are the best architectures for certain classes of problems?
- What are the best ways of extracting the knowledge stored in the network?

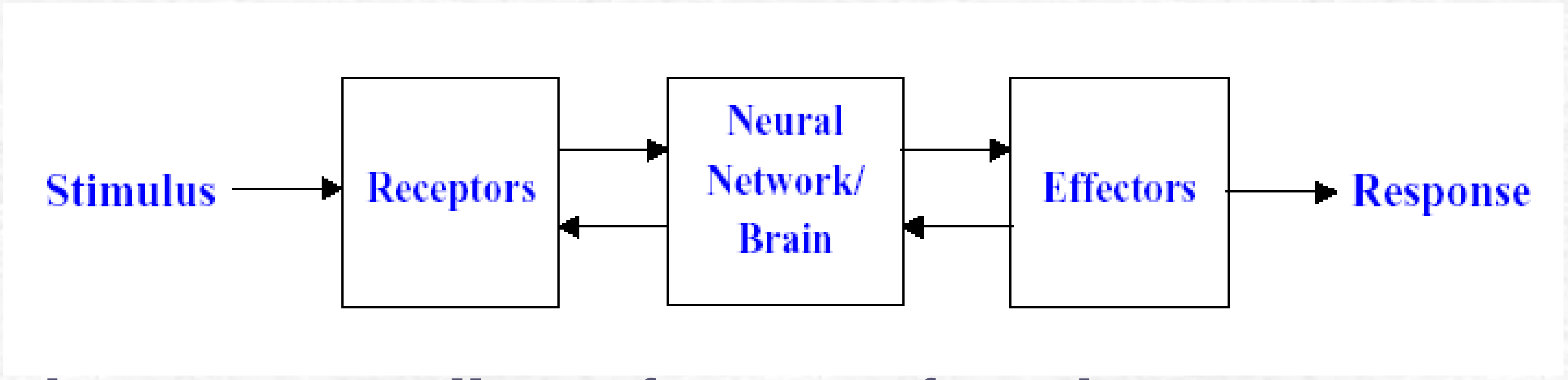
Artificial neural networks and AI’s History

- • 1943: McCulloch and Pitts develop basic models of neurons.
- • 1962: Rosenblatt's learning Perceptron.
- • 1969: Minsky and Papert published their book "Perceptrons"on the limitations of neural models.
- • 1970s: Knowledge-based systems.
- • 1980s: AI becomes an industry. Renewed enthusiasm. First commercial successes through expert systems were achieved.

Artificial neural networks and AI’s History

- • Mid to late 1980s, people feel again much more difficulties in getting expert systems work, than they expected.
- • Around 1986: Renaissance of neural networks - connectionism.
- • 1990s: More consolidated approaches to problems and techniques.More realistic expectations (i.e.reduced expectations and much better computing resources available).

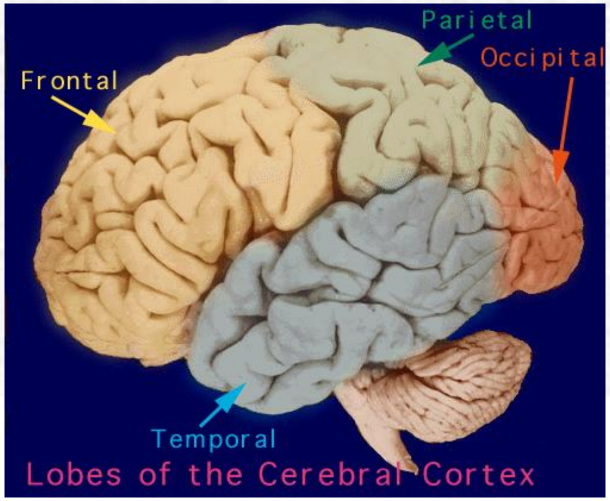
The human nervous system in block diagram

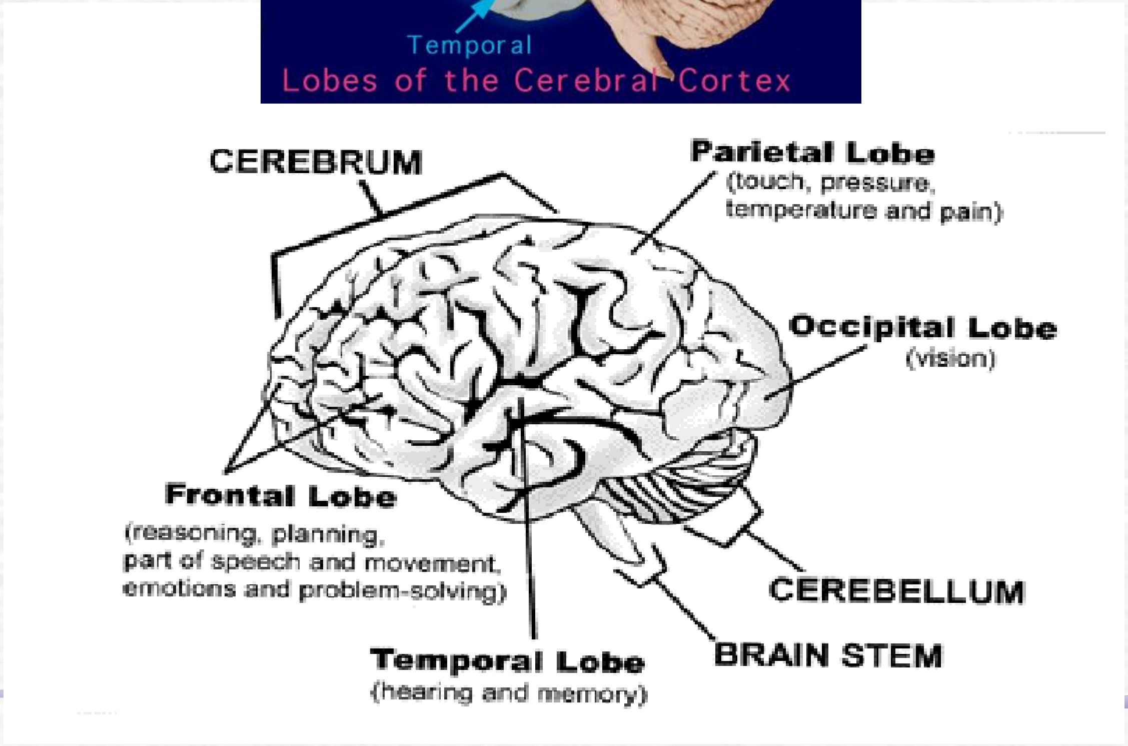


The receptors collect information from the environment (photons on the retina).
The effectors generate interactions with the environment (activate muscles).
The flow of information/activation is represented by arrows – feedforward and feedback. Naturally, in this module we will be primarily concerned with the neural network in the middle.

Structure of a Human Brain

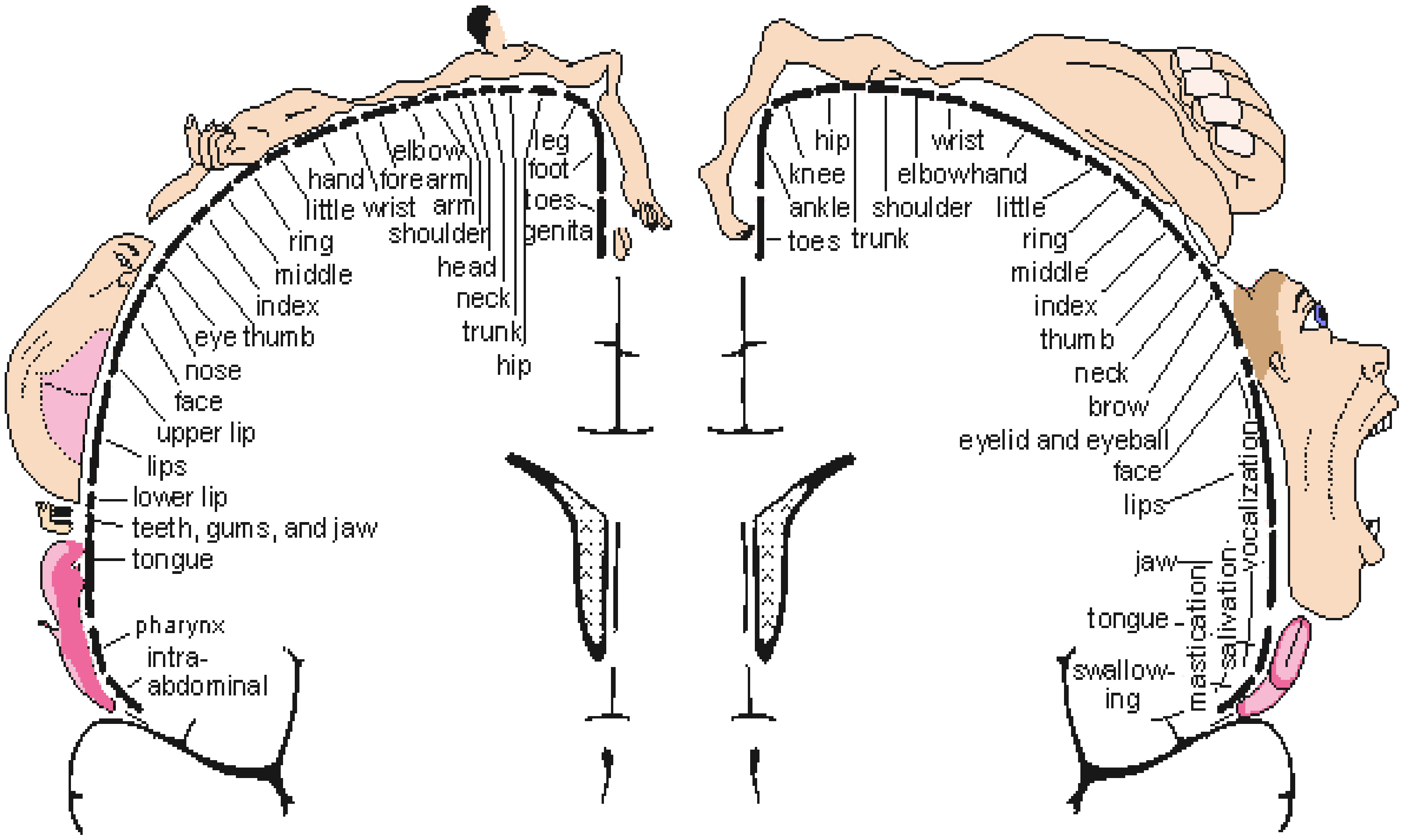
The central nervous system is divided into two major parts:
• the brain
• and the spinal cord.
– The spinal cord is about 43 cm long in adult women and 45 cm long in adult men and weighs about 35-40 gm.




CEREBRUM
Parietal Lobe
(touch, pressure, temperature and pain)
Occipital Lobe
(vision)
Frontal Lobe
(reasoning, planning, part of speech and movement, emotions and problem-solving)
CEREBELLUM
BRAIN STEM
Temporal Lobe
(hearing and memory)

Cerebral Cortex and Cerebellum

- The word "cortex" comes from the Latin word for "bark" (of a tree). This is because the cortex is a sheet of tissue that makes up the outer layer of the brain. The thickness of the cerebral cortex varies from 2 to 6mm. The right and left sides of the cerebral cortex are connected by a thick band of nerve fibers called the "corpus callosum."
- **Functions:** Thought, Voluntary movement, Language, Reasoning, Perception
- The word "cerebellum" comes from the Latin word for "little brain”
- **Functions:** Movement, Balance, Posture
-



SENSORY CORTEX

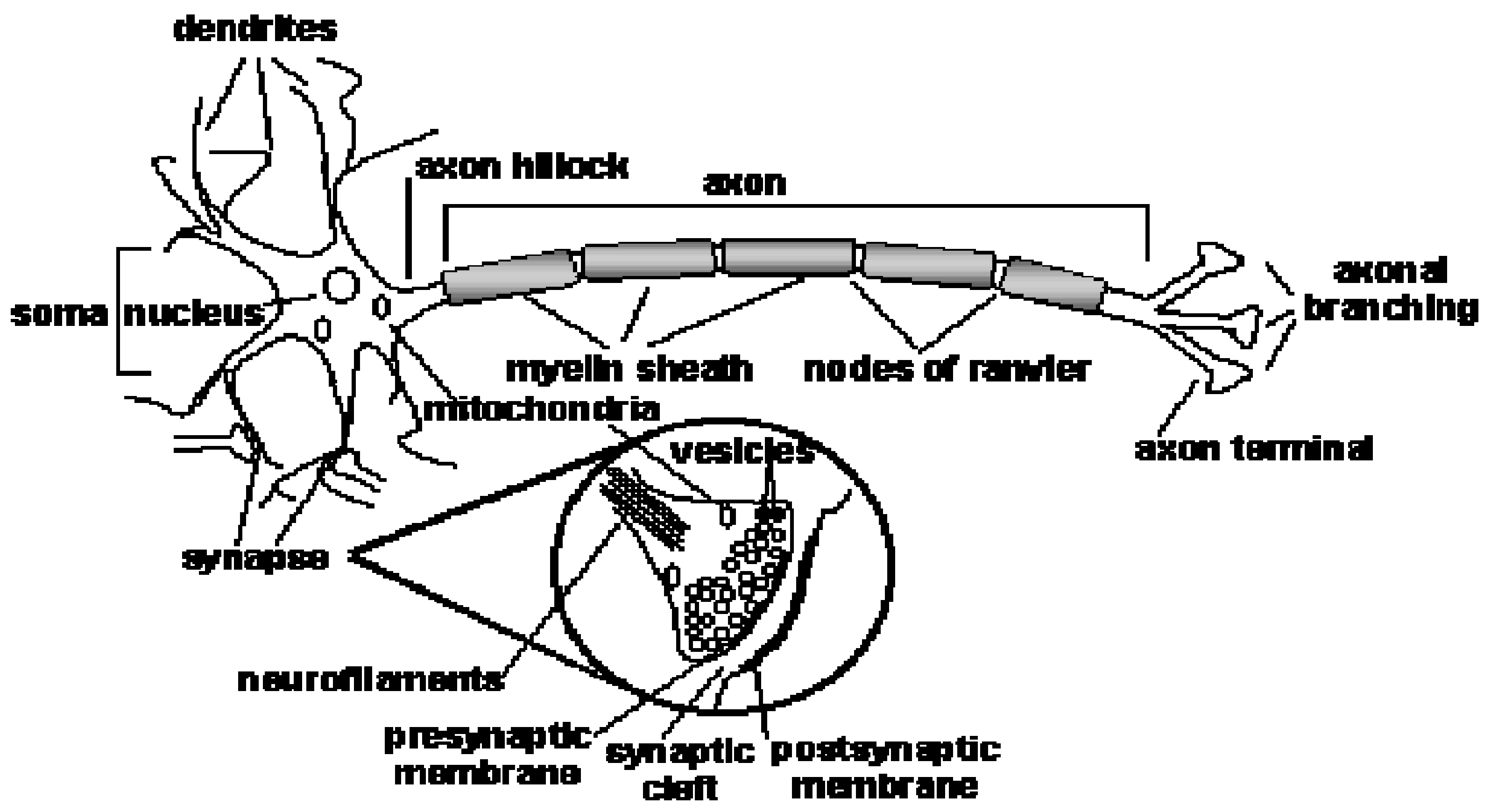
MOTOR CORTEX

Brains versus Computers : Some numbers

- Approximately 10 billion neurons in the human cortex / 10 of thousands of processors in the most powerful parallel computers.
- Each biological neuron is connected to several thousands of other neurons, similar to the connectivity in powerful parallel computers.
- 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).
- The human brain is extremely energy efficient: Approximately 10^{-16} joules per operation per second / the best computers today use around 10^{-6} joules per operation per second.

2.1. Biological Neuron

Schematic Diagram of a Biological Neuron



Basic Components of Biological Neurons

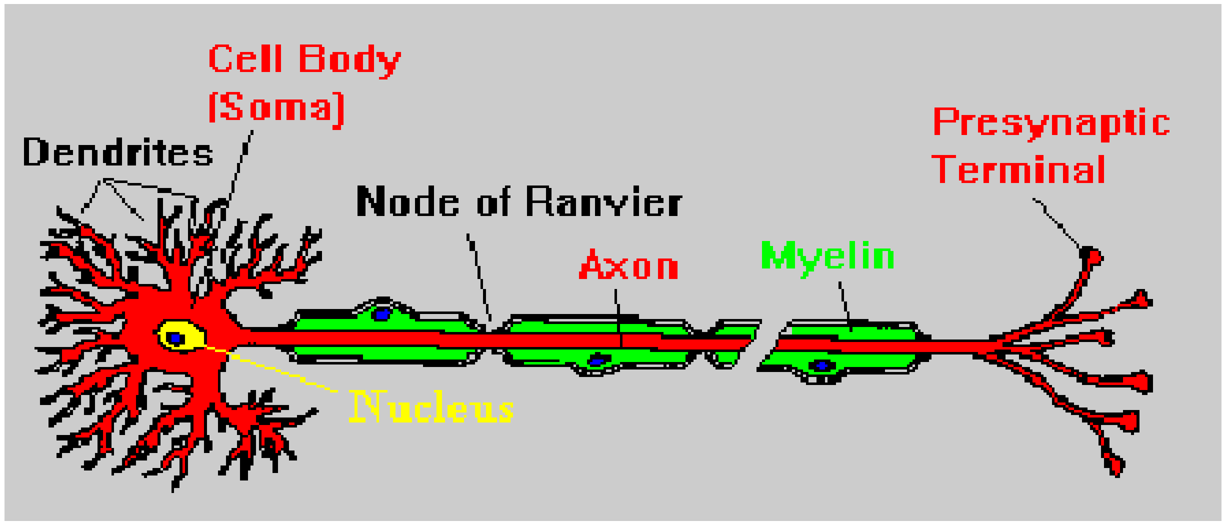
- The majority of ***neurons*** encode their activations or outputs as a series of brief electrical pulses (i.e. spikes or action potentials).
- The neuron’s ***cell body (soma)*** processes the incoming activations and convert them into output activations.
- The neuron’s ***nucleus*** contains the genetic material in the form of DNA. This is the same as in most types of cells, not just neurons.
- ***Dendrites*** are fibres which emanate from the cell body and provide the receptive zones that receive activation from other neurons.
- ***Axons*** are fibres acting as transmission lines that send activation to other neurons.

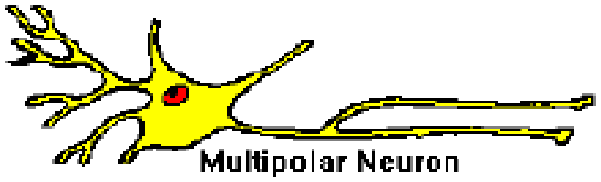
- The junctions that allow signal transmission between the axons and dendrites are called ***synapses***.
- The process of transmission is by diffusion of chemicals called ***neurotransmitters*** across the synaptic cleft.

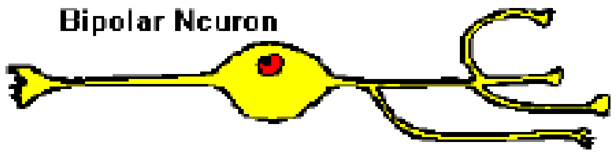
Neurons versus body cells

- 1.Neurons have specialized extensions called dendrites and axons. Dendrites bring information to the cell body and axons take information away from the cell body.
- 2.Neurons communicate with each other through an electrochemical process.
- 3.Neurons contain some specialized structures (for example, synapses) and chemicals (for example, neurotransmitters).

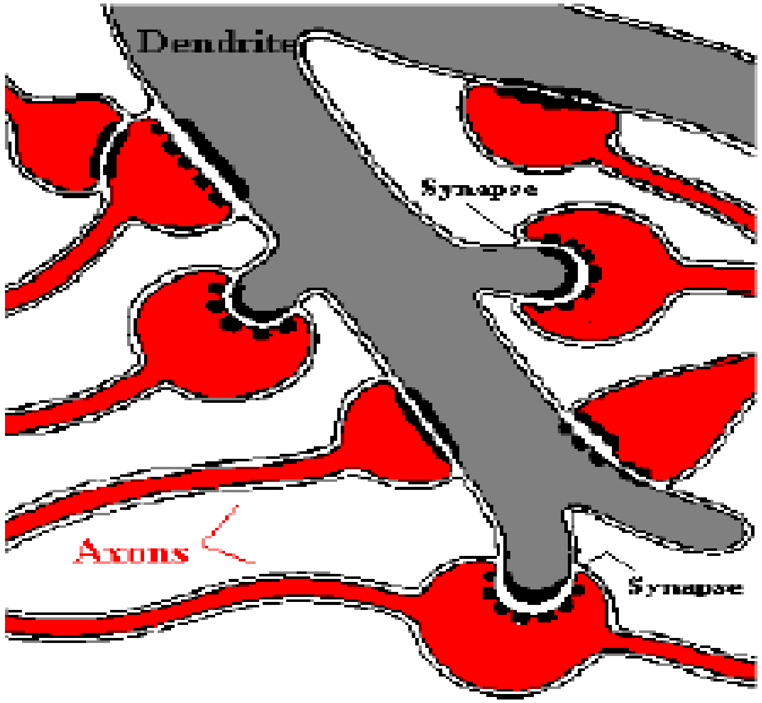
Different Structures of a Neuron







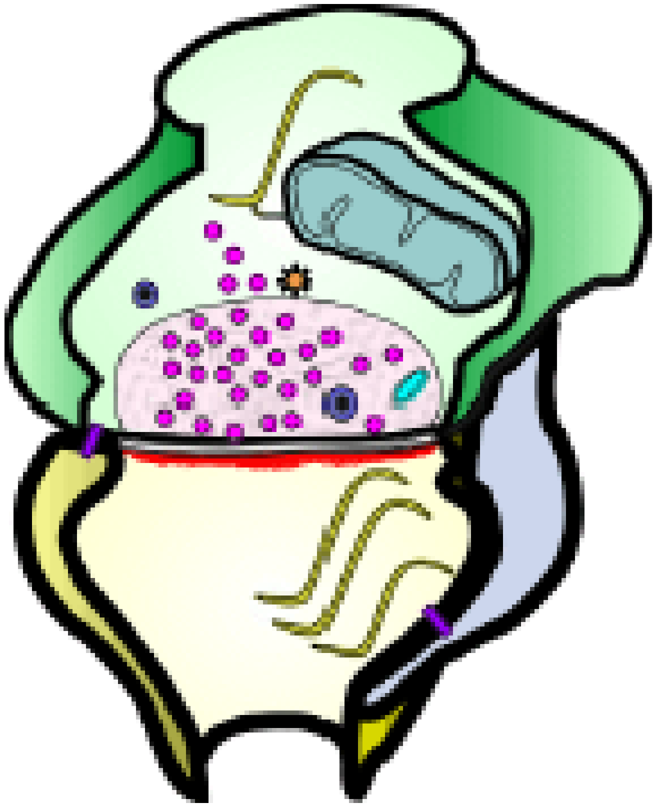
Dendrites and Synapses



The Synapse

- In most synapses the direct cause of change in potential is not electrical but chemical: The electrical pulse reaches the endbulb and causes the release of transmitter molecules from little packets (vesicles) through the synaptic membrane.
- Transmitter than diffuses through the synaptic cleft to the other side.When the transmitter reaches the post-synaptic membrane, it causes the change in polarisation of the membrane.The change in potential can be **excitatory** (moving the potential towards the threshold) or **inhibitory** moving the potential away from the threshold.

A schematic Synapse



Presynaptic Axon

Postsynaptic Spine

Active Zone

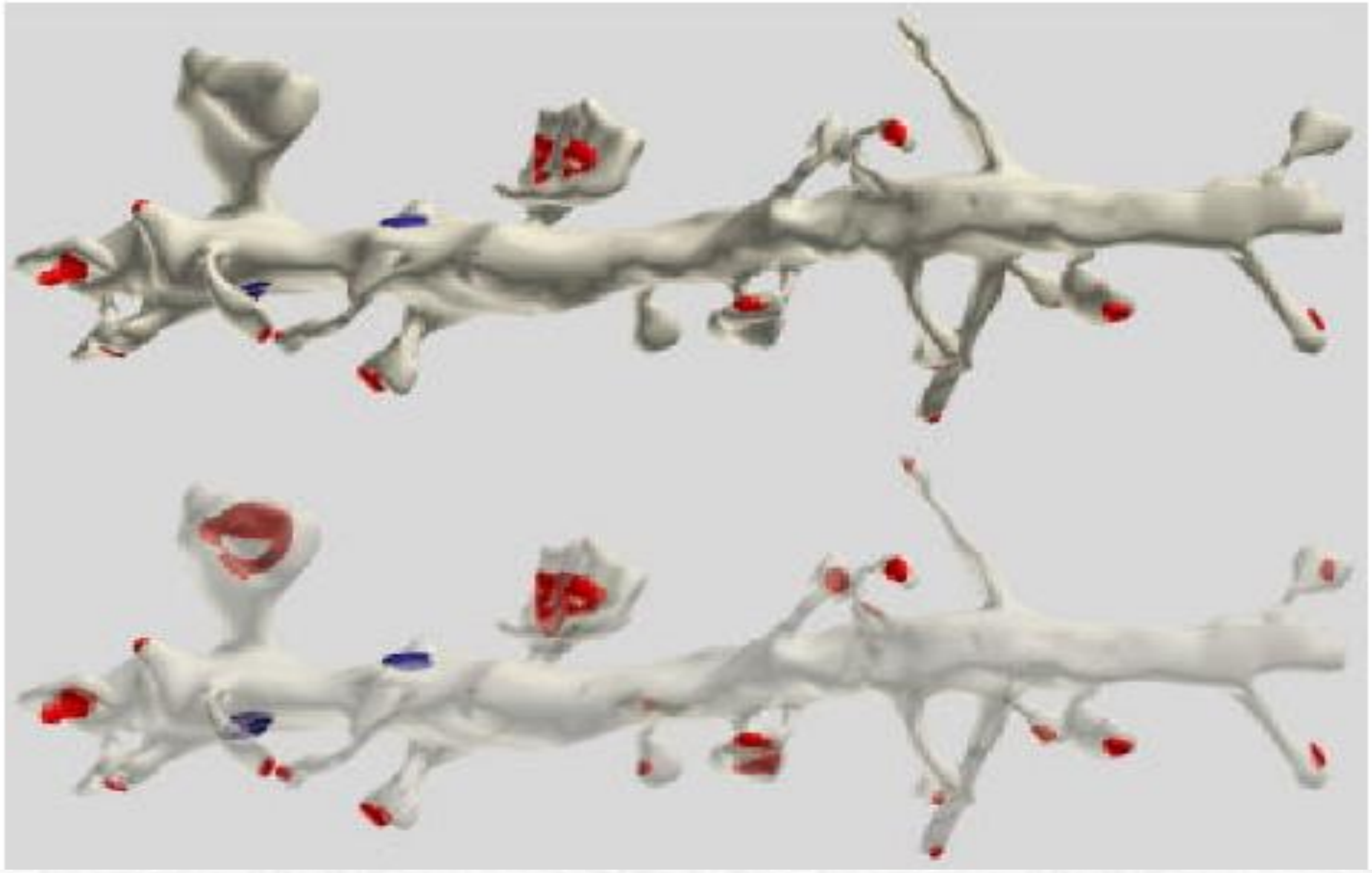
Synaptic Vesicle

Synaptic Cleft

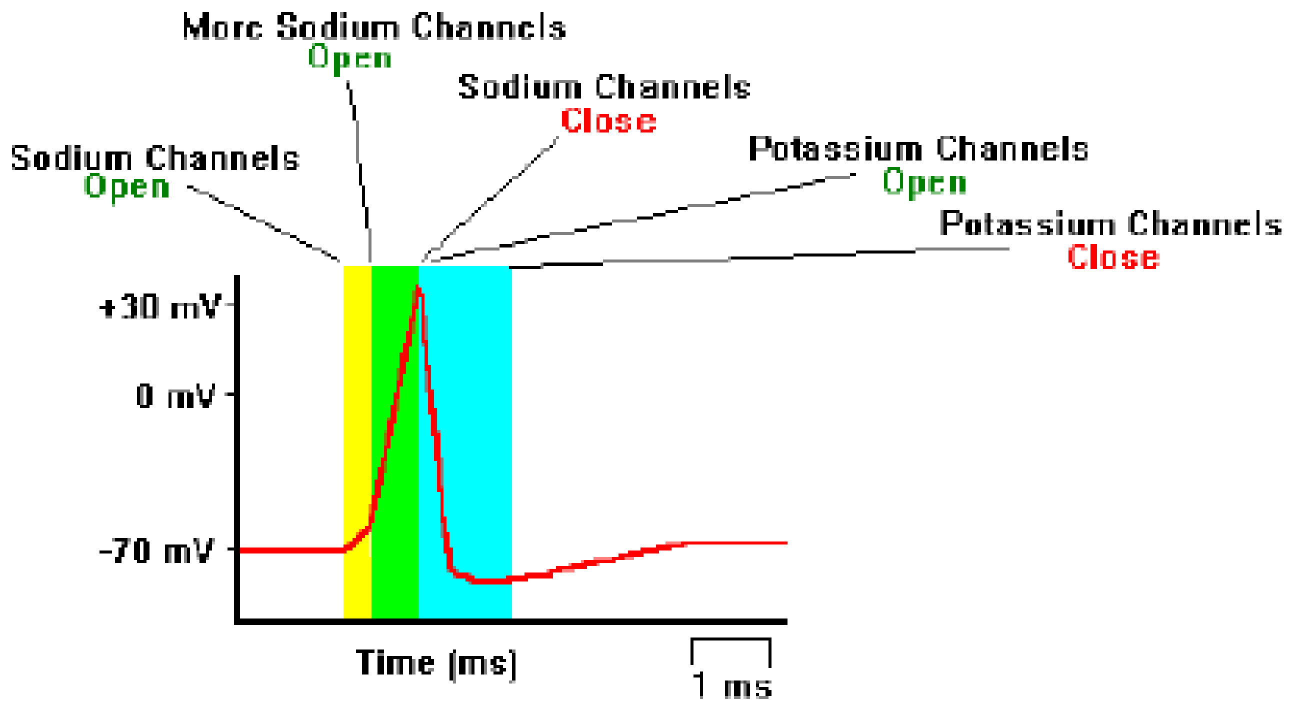
Postsynaptic Density

3-D Image of a Dendrite

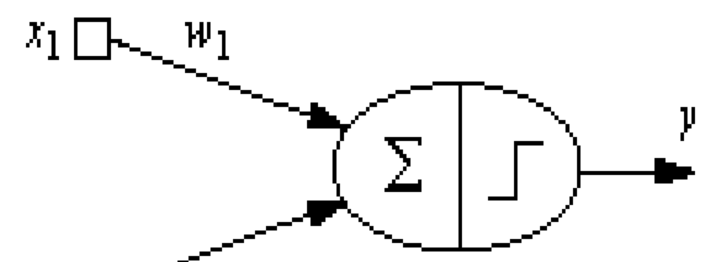
(rat, hippocampus)



Metabolism when neuron becomes active



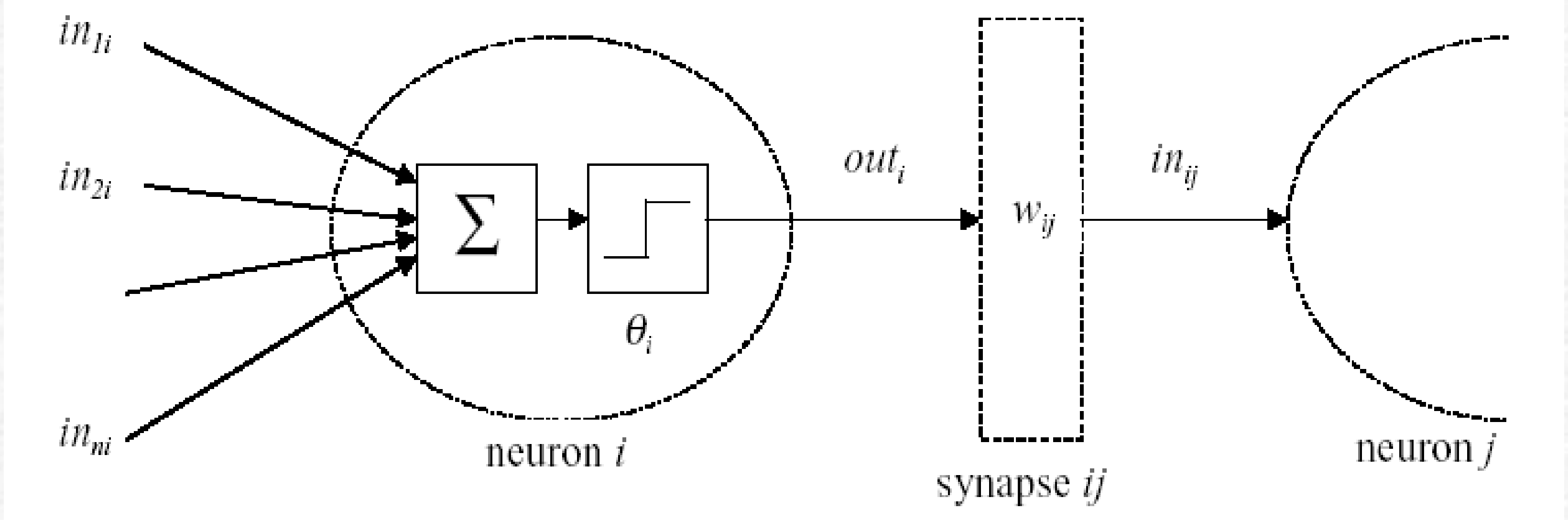
2.2 Artificial Neuron (McCulloch-Pitts)



$$y = \begin{cases} -1 & \text{if} \quad \sum_{j=1,2} w_j x_j + b < 0 \\ 1 & \text{if} \quad \sum_{j=1,2} w_j x_j + b \geq 0 \end{cases}$$

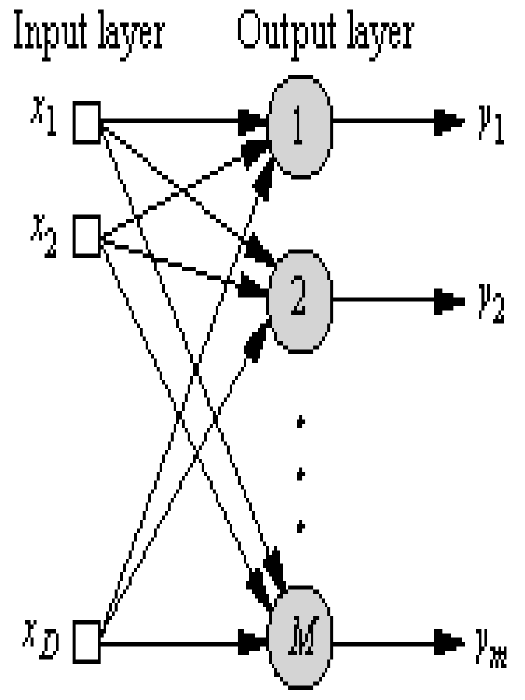
- INPUTS:Receptors
- WEIGHTS: Different effects for different receptors
- SUMMING JUNCTION: Summation of effects of receptors
- TRANSFER FUNCTION and BIAS: Threshold potential for triggering
- OUTPUTS: Number of packets released

Networks of McCulloch-Pitts Neurons



$$in_{ki} = out_k w_{ki} \qquad out_i = \mathrm{sgn}\left(\sum_{k=1}^{n} in_{ki} - \theta_i\right) \qquad in_{ij} = out_i w_{ij}$$

2.3. Rosenblatt's Perceptron



The perceptron with D inputs and M outputs (D-M)

- The perceptron has multiple inputs fully connected to an output layer with multiple McCulloch-Pitts neurons . Each input xj is multiplied by an adjustable constant wij (the weight) before being fed to the ith processing element

$$y_i = f(net_i) = f\left(\sum_j w_{ij} xj + b_i\right)$$

Implementing Logic Gates with M-P Neurons

- We can use McCulloch-Pitts neurons to implement the basic logic gates. All we need to do is find the appropriate connection weights and neuron thresholds to produce the right outputs for each set of inputs.
- The resulting networks, however, will usually have a much more complex architecture than a simple Perceptron.
- We generally want to avoid decomposing complex problems into simple logic gates, by finding the weights and thresholds that work directly in a Perceptron architecture.

Implementation of Logical NOT, AND, and OR

In each case we have inputs *in* and outputs *out*, and need to determine the weights and thresholds. It is easy to find solutions by inspection:

NOT

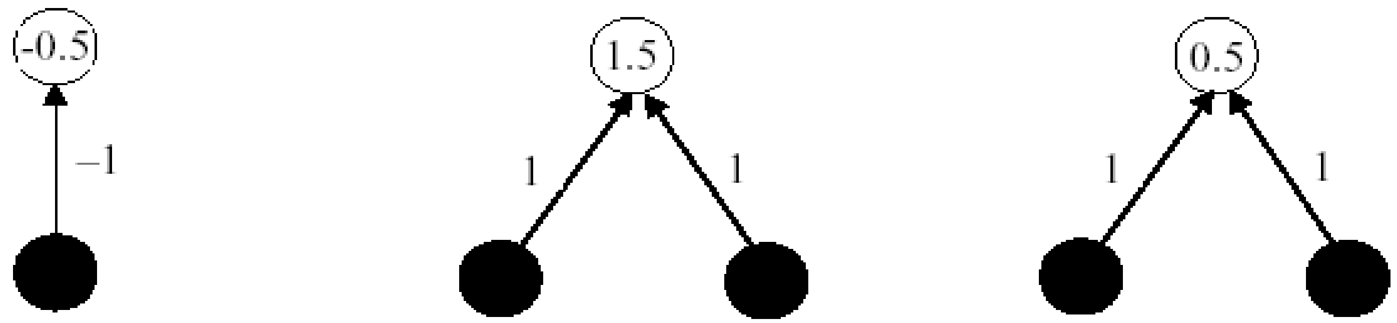
in	out
0	1
1	0

AND

in_1	in_2	out
0	0	0
0	1	0
1	0	0
1	1	1

OR

in_1	in_2	out
0	0	0
0	1	1
1	0	1
1	1	1



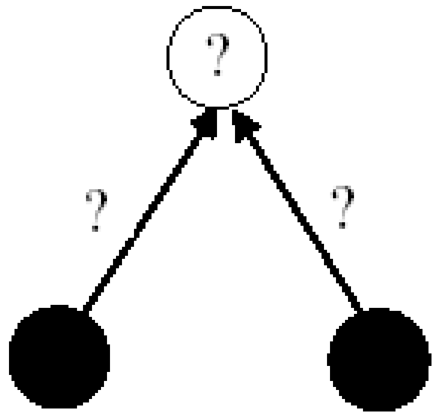
The Need to Find Weights Analytically

Constructing simple networks by hand is one thing. But what about harder problems?

Calculate appropriate parameters rather than looking for solutions by trial and error

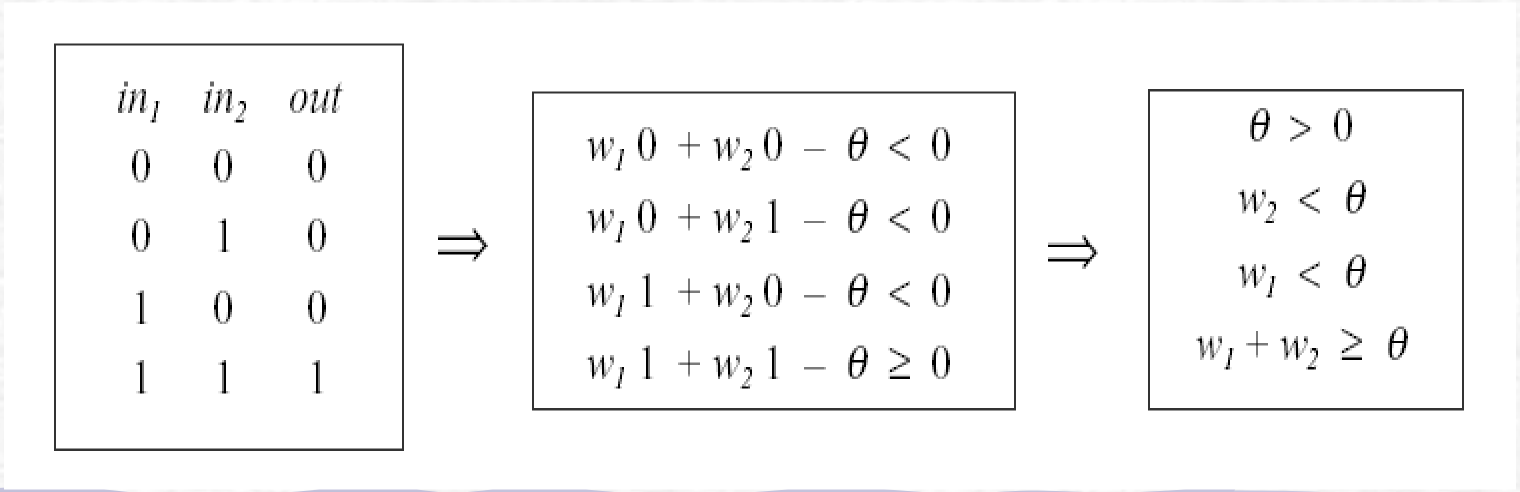
XOR

in_1	in_2	out
0	0	0
0	1	1
1	0	1
1	1	0



Finding Weights Analytically for the AND Network

- We have two weights *w1* and *w2* and the threshold θ, and for each training pattern we need to satisfy
  $out = \mathrm{sgn}(w1\,in1 + w2\,in2 - \theta)$
  So the training data lead to four inequalities:
  There is no unique solution



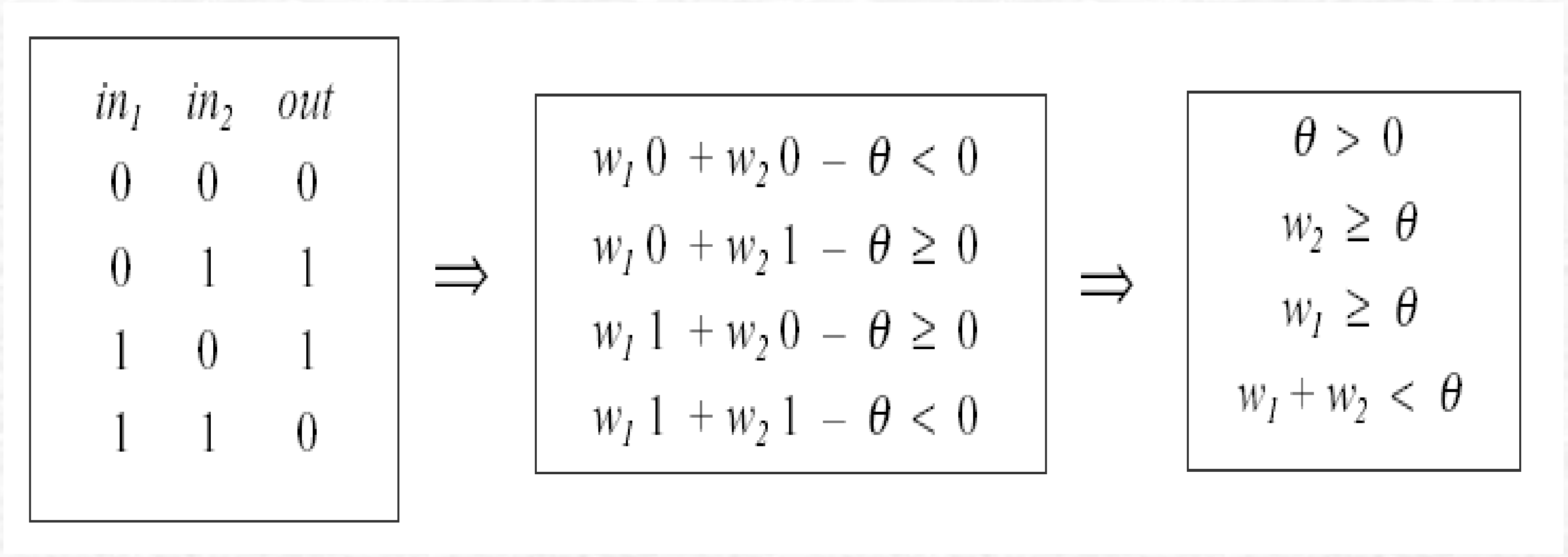
in_1	in_2	out
0	0	0
0	1	0
1	0	0
1	1	1

$$\Rightarrow \quad \begin{array}{l} w_1 0 + w_2 0 - \theta < 0 \\ w_1 0 + w_2 1 - \theta < 0 \\ w_1 1 + w_2 0 - \theta < 0 \\ w_1 1 + w_2 1 - \theta \geq 0 \end{array} \quad \Rightarrow \quad \begin{array}{c} \theta > 0 \\ w_2 < \theta \\ w_1 < \theta \\ w_1 + w_2 \geq \theta \end{array}$$

Limitations of Simple Perceptrons

We can follow the same procedure for the XOR network:

Clearly the second and third inequalities are incompatible with the fourth, so there is in fact no solution. We need more complex networks,



in_1	in_2	out
0	0	0
0	1	1
1	0	1
1	1	0

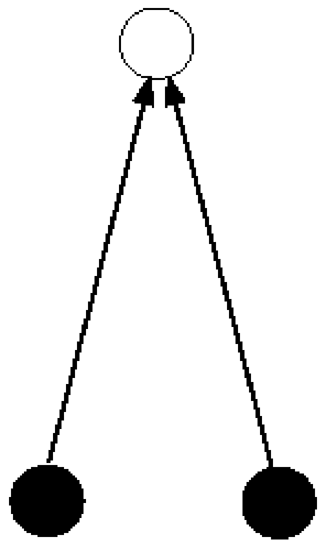
$$\Rightarrow \quad \begin{array}{l} w_1 0 + w_2 0 - \theta < 0 \\ w_1 0 + w_2 1 - \theta \geq 0 \\ w_1 1 + w_2 0 - \theta \geq 0 \\ w_1 1 + w_2 1 - \theta < 0 \end{array} \quad \Rightarrow \quad \begin{array}{c} \theta > 0 \\ w_2 \geq \theta \\ w_1 \geq \theta \\ w_1 + w_2 < \theta \end{array}$$

ANN Architectures/Structures/Topologies

Mathematically, ANNs can be represented as ***weighted directed graphs***. Three common ANN architectures are:
Single-Layer Feed-forward NNs One input and one output layer of processing units. No feed-back connections. (For example, a simple Perceptron.)
Multi-Layer Feed-forward NNs One input, one output, and one or more hidden layers of processing units. The hidden layers sit inbetween the input and output layers, and are thus *hidden* from the outside world.(For example, a Multi-Layer Perceptron.)
Recurrent NNs Has at least one feed-back connection. May or may not have hidden units. (For example, a Simple Recurrent Network.)
Further interesting variations include: short-cut connections, partial connectivity, timedelayed connections, Elman networks, Jordan networks, moving windows,

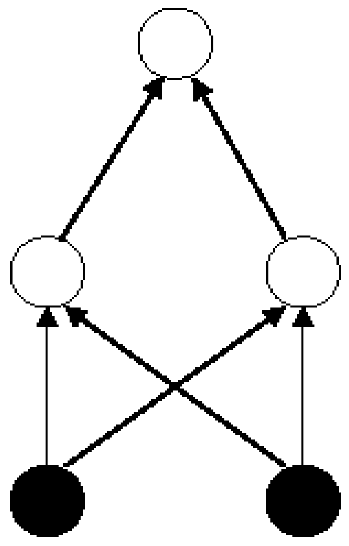
Examples of Network Architectures

Single Layer Feed-forward



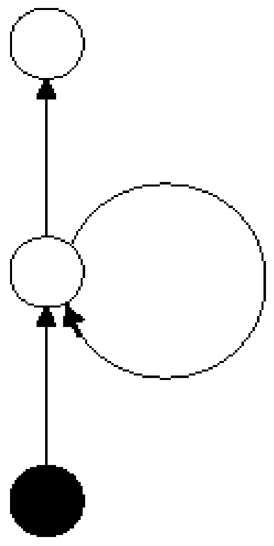
Single-Layer Perceptron

Multi-Layer Feed-forward



Multi-Layer Perceptron

Recurrent Network

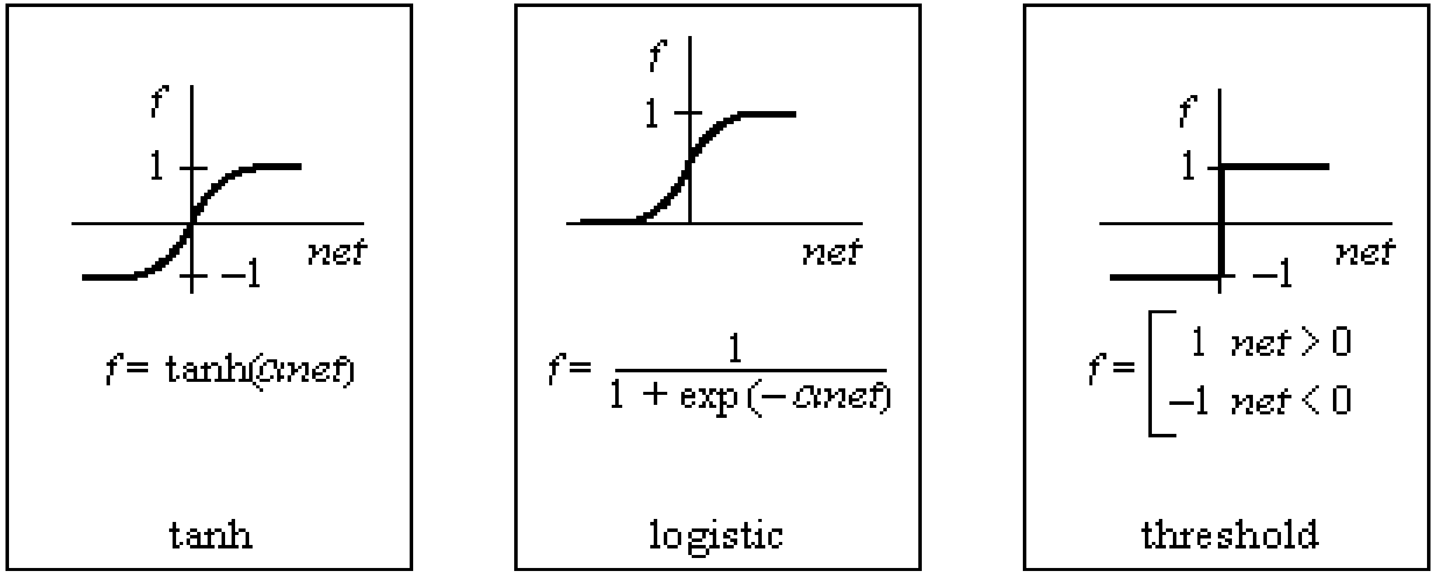


Simple Recurrent Network

Activation Functions (Squashing Functions)

hyperbolic $\quad f(net) = \tanh(\alpha net)$

logistic $\quad f(net) = \dfrac{1}{1 + \exp(-\alpha net)}$



Common nonlinearities in neurocomputing

2.4. Multilayer Perceptrons

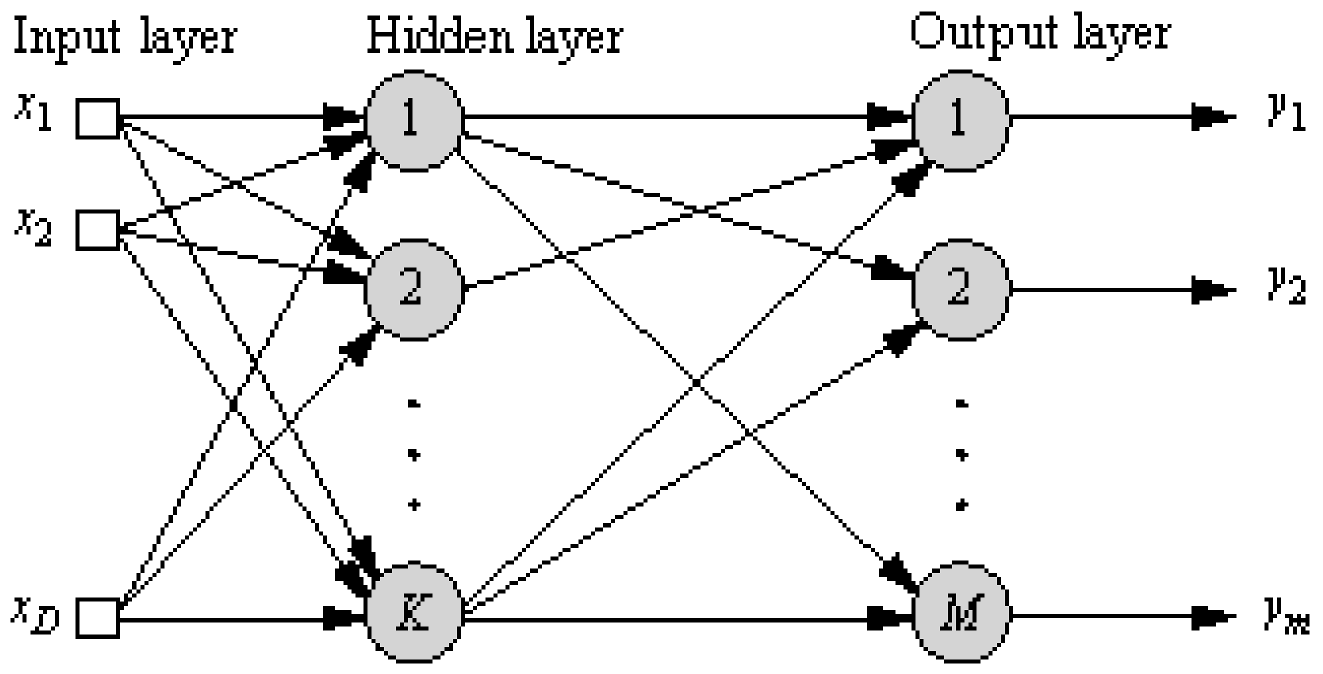


Figure 3.14 **A multilayer perceptron with one hidden layer (D-K-M)**

The Threshold as a Weight

It would simplify the mathematics if we could treat the neuron threshold as if it were just another connection weight. The crucial thing we need to compute is:

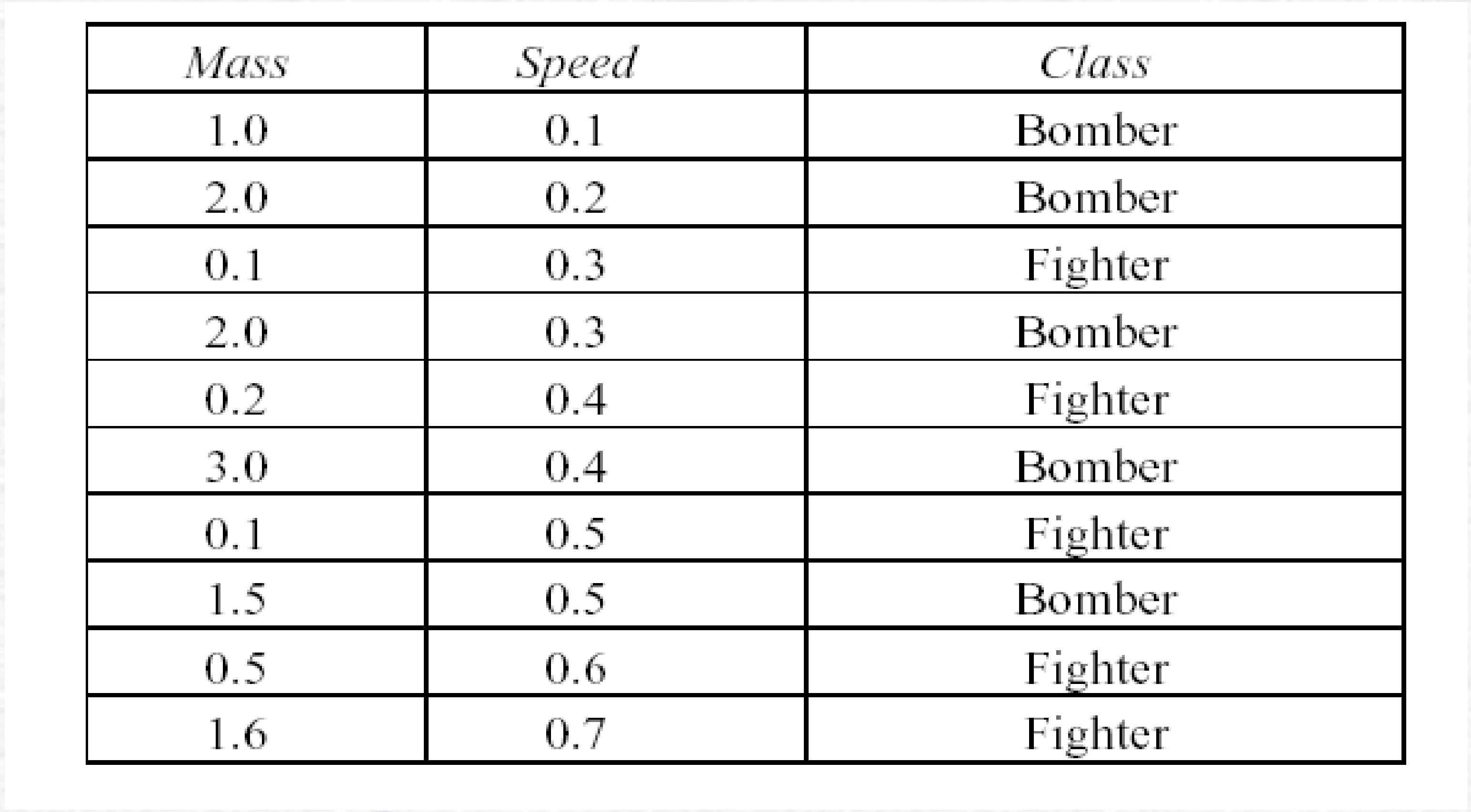
$$\sum_{i=1}^{n} out_i w_{ij} - \theta_j = out_1 w_{1j} + out_2 w_{2j} + \ldots + out_n w_{nj} - \theta_j$$

It is easy to see that if we define $w_{0j} = -\theta_j$ and $out_0 = 1$ then this becomes:

$$\sum_{i=1}^{n} out_i w_{ij} - \theta_j = out_1 w_{1j} + out_2 w_{2j} + \ldots + out_n w_{nj} + out_0 w_{0j} = \sum_{i=0}^{n} out_i w_{ij}$$

Example : A Classification Task

A typical neural network application is classification. Consider the simple example of classifying aeroplanes given their masses and speeds:



Mass	*Speed*	*Class*
1.0	0.1	Bomber
2.0	0.2	Bomber
0.1	0.3	Fighter
2.0	0.3	Bomber
0.2	0.4	Fighter
3.0	0.4	Bomber
0.1	0.5	Fighter
1.5	0.5	Bomber
0.5	0.6	Fighter
1.6	0.7	Fighter

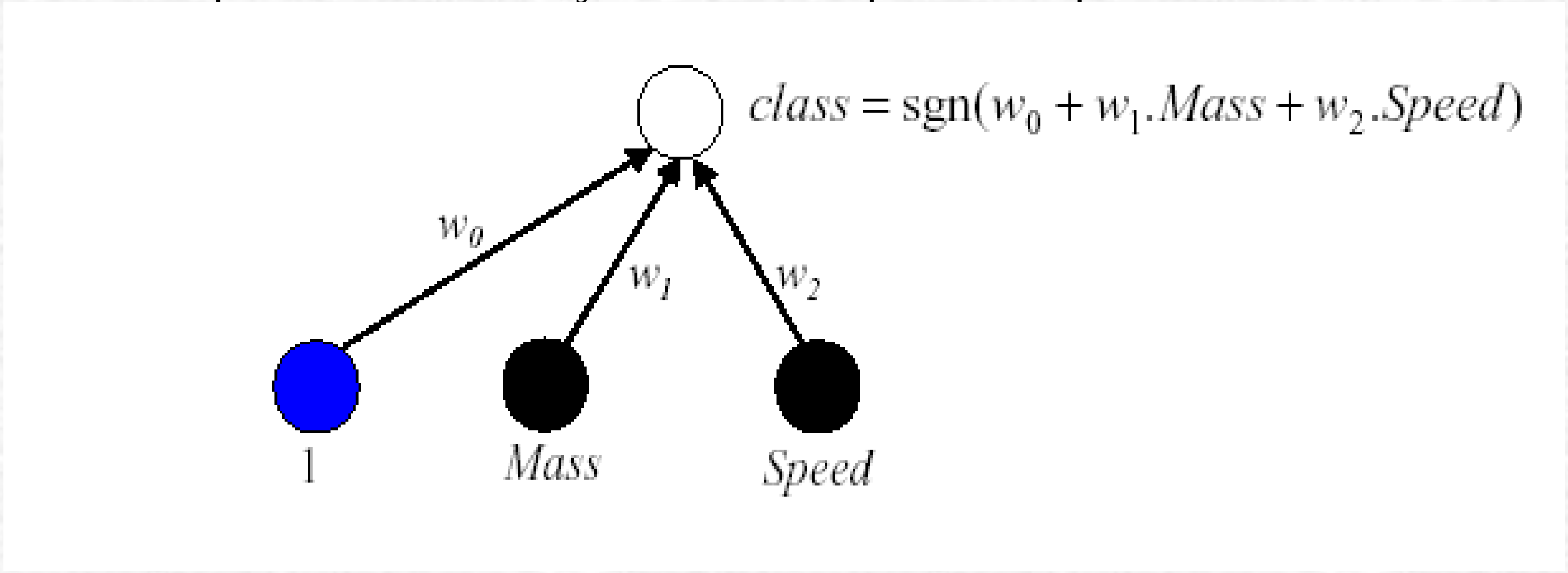
General Procedure for Building a Neural Network

- 1. Understand and specify your problem in terms of inputs and required outputs, e.g.for classification the outputs are classes usually represented as binary vectors.
- 2. Take the simplest form of network you think might be able to solve your problem, e.g. a simple Perceptron.
- 3. Try to find appropriate connection weights (including neuron thresholds) so that the network produces the right outputs for each input in its training data.
- 4. Make sure that the network works on its ***training data***, and test its generalization by checking its performance on new ***testing data***.
- 5. If the network doesn't perform well enough, go back to stage 3, stage 2, stage 1 and try harder.

Building a Neural Network for Our Example

For our aeroplane classifier example, our inputs can be direct encodings of the masses and speeds.

- Generally we would have one output unit for each class, with activation 1 for ‘yes’ and 0 for ‘no’.
- With just two classes here, we can have just one output unit, with activation 1 for ‘fighter’ and 0 for ‘bomber’ (or vice versa).
- The simplest networkto try first is a simple Percerptron.



2.5. Knowledge Representation

- Knowledge refers to stored information or models used by a person or machine to interpret, predict, and appropriately respond to the outside world
- Prior information, observations and measurements (noisy)

Rules in Knowledge Representation

- RULE.1. Similar inputs from similar classes should usually produce similar representation inside the network. They should be classified as belonging to the same cathegory.

  The measure of similarity between inputs:

  * Euclidian Distance

  * Dot Product (Inner Product)

- RULE.2. Items to be cathegorized as separate classes should be given widely different representation. The main differences should be emphasized.

- RULE.3. If a particular feature is important, there should be a large number of neurons involved in the representation of that item in the network.
- RULE.4. Prior information and invariances should be built into design of a neural network, thereby simplifying the network design by not having to learn them.