Plan for today
Prof. Manevitz
manevitz@cs.Haifa.ac.il
Office Hour (after classes)

- Requirements
  - Probably two projects and exam
  - YOU MUST PASS EXAM FOR PROJECTS TO COUNT
  - You must pass exam for project to count.

Morning Class 10 -1 in Hebrew

Afternoon Class 3 – 6 PM in English

Introduction to NeuroComputation

Prof. L. Manevitz

Dept of Computer Science

TODAY

- Background
  - Brains and Computers
  - Computational Models
    - Learnability vs Programming
    - Representability vs Training Rules
- Abstractions of Neurons
- Abstractions of Networks
  - Completeness of 3-Level Mc-Cullough-Pitts Neurons
- Learnability in Perceptrons

What is /isn't in this course

In the Course

- Foundation and Examples Underlying the NN revolution.

- We will see basic techniques, where they come from, basic intuitions and some of the mathematics underlying the methodology
- There will be implementation projects (probably 2) which will count for a substantial part of the grade; probably 50%. You will do these with a partner of your choice,

Not in the Course(mostly - we may touch on some items including today)

- It is **not** a course guiding you how to use the latest "Deep NNs" programs and packages (There is another course given by Prof. Amos Azaria which does this.)
- It is **not** a course showing you how to simulate and understand the human brain and cognition. (This is another wing of NNs emphasized in a course often given in the fall by myself.)

Brains and Computers

- What is computation?
- Basic Computational Model

(Like C, Pascal, C++, etc)

(See course on Models of Computation.)

Analysis of problem and reduction to algorithm.

Implementation of algorithm by Programmer.

Computation

Brain also computes. But

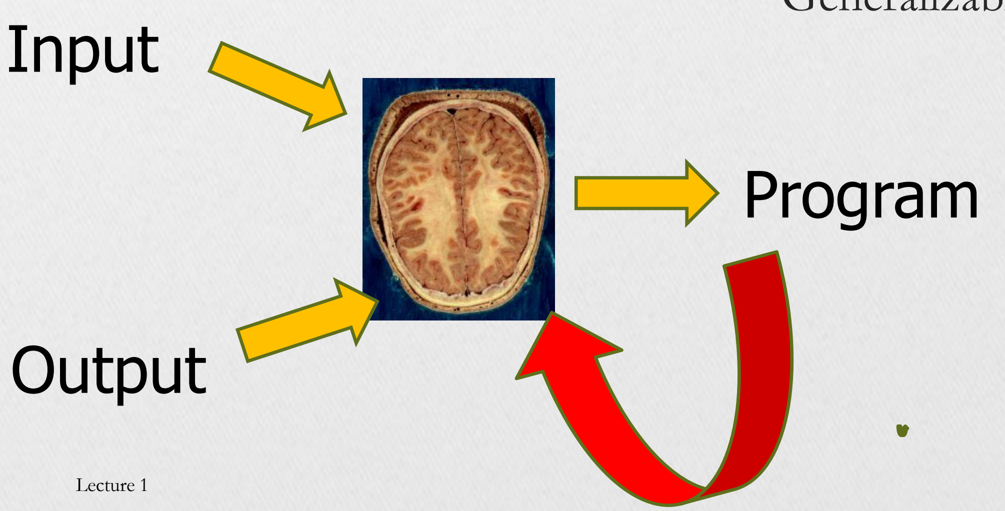
Its programming seems different.

Flexible, Fault Tolerant (neurons die every day),

Automatic Programming,

Learns from examples,

Generalizability



Computation and Psychology
Computation and Brain

- This is the main subject of our course
- What is the software/hardware as we see it via computational eyes?
- Two Directions
- Neuroscience understanding
- Understanding how it is possible at all for the brain to compute as it does
- Engineering Methodologies
- Extracting from methods of brain to create new computational methodologies

Successes

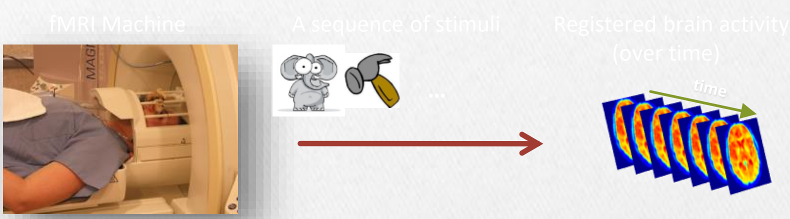
Computer Vision: Pattern Recognition and Big Data

Small Data and Brain Modeling

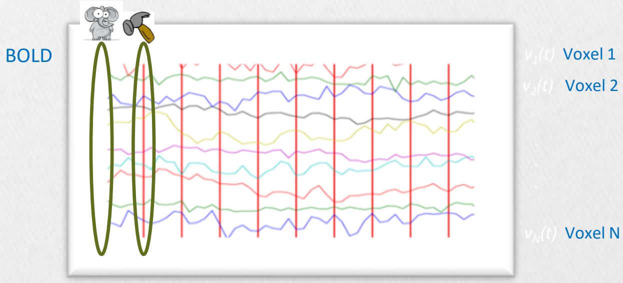
Production of Artificial Data

Clear Learning Rules

MIND READING



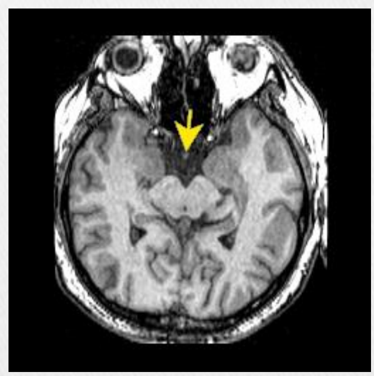
- Blood Oxygen Level-Dependent (BOLD) signal (oxygen hemodynamic response) is a measurement of the brain activity
- BOLD signal is recorded for each voxel inside the brain image
- (So MANY Features – 100,000 – 200,000)



Challenge: Computer Vision – Recognize What is Seen in Camera

Challenge:
Given an fMRI

- Can we learn to recognize from the MRI data, the cognitive task being performed?
- Automatically?



WHAT ARE THEY?

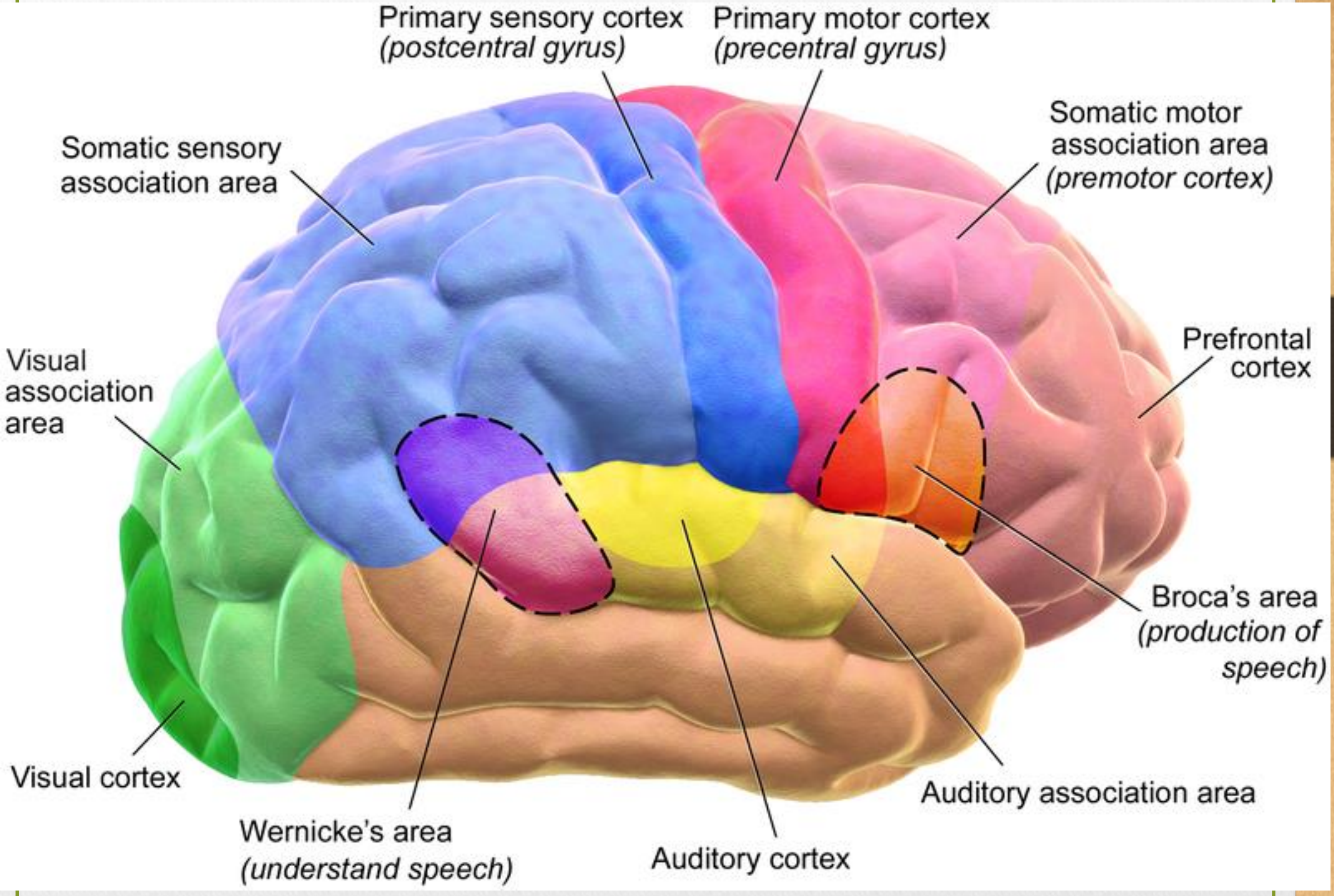
Putin

Thinking Thoughts

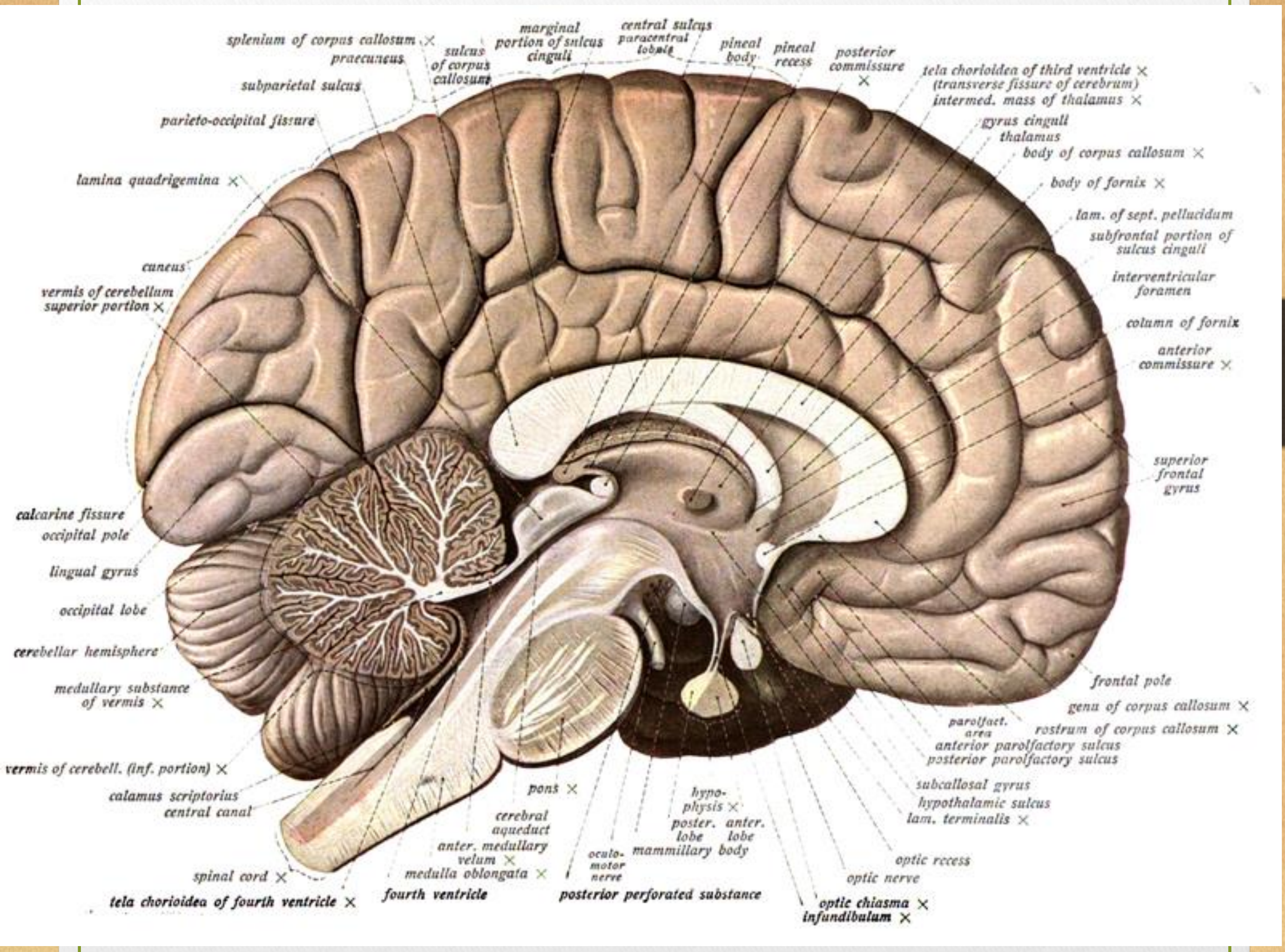
Applications

- Diagnosing Parkinson's
- Diagnosing Alzheimer's
- And so on …

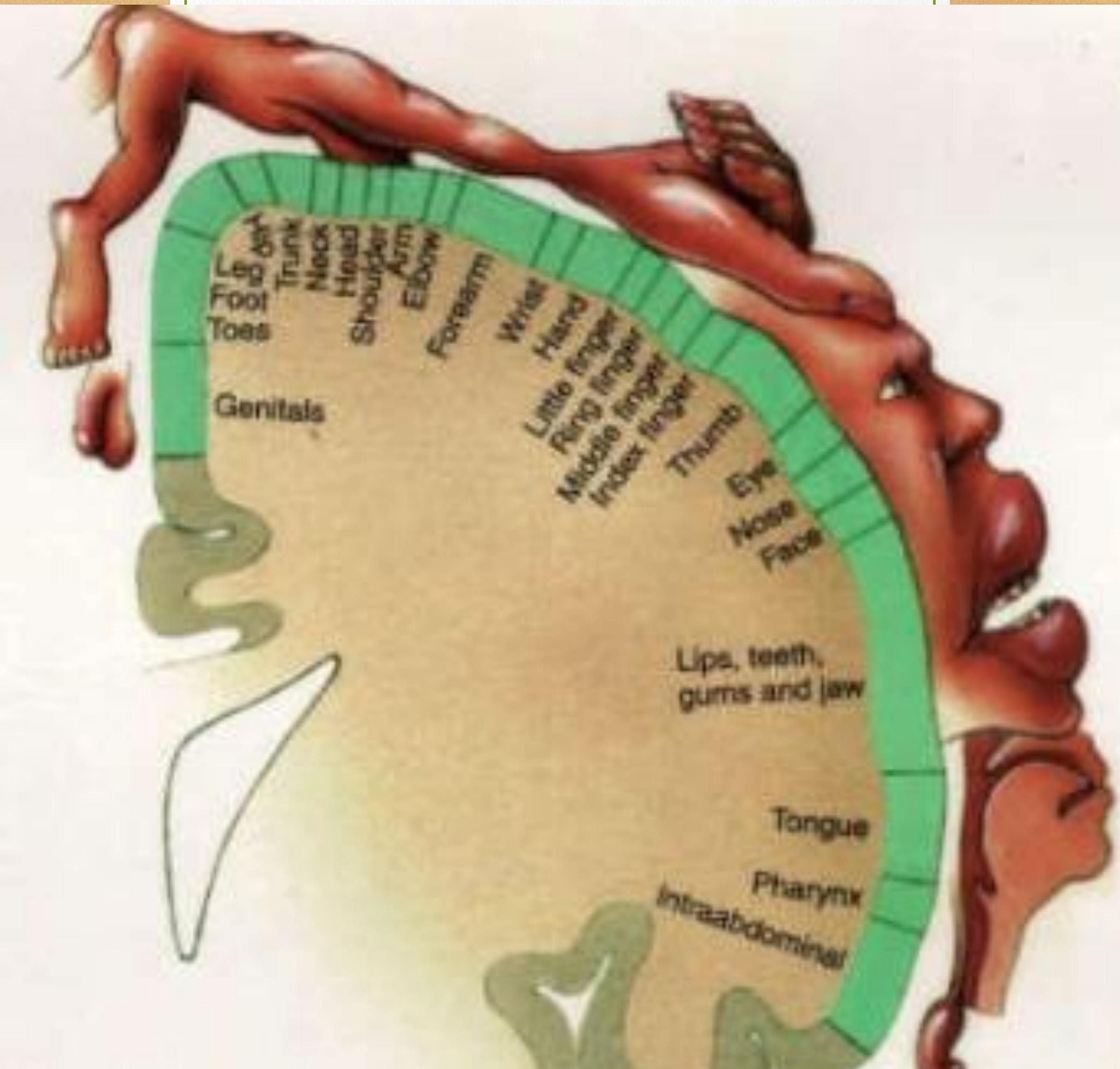
What does Brain look like?



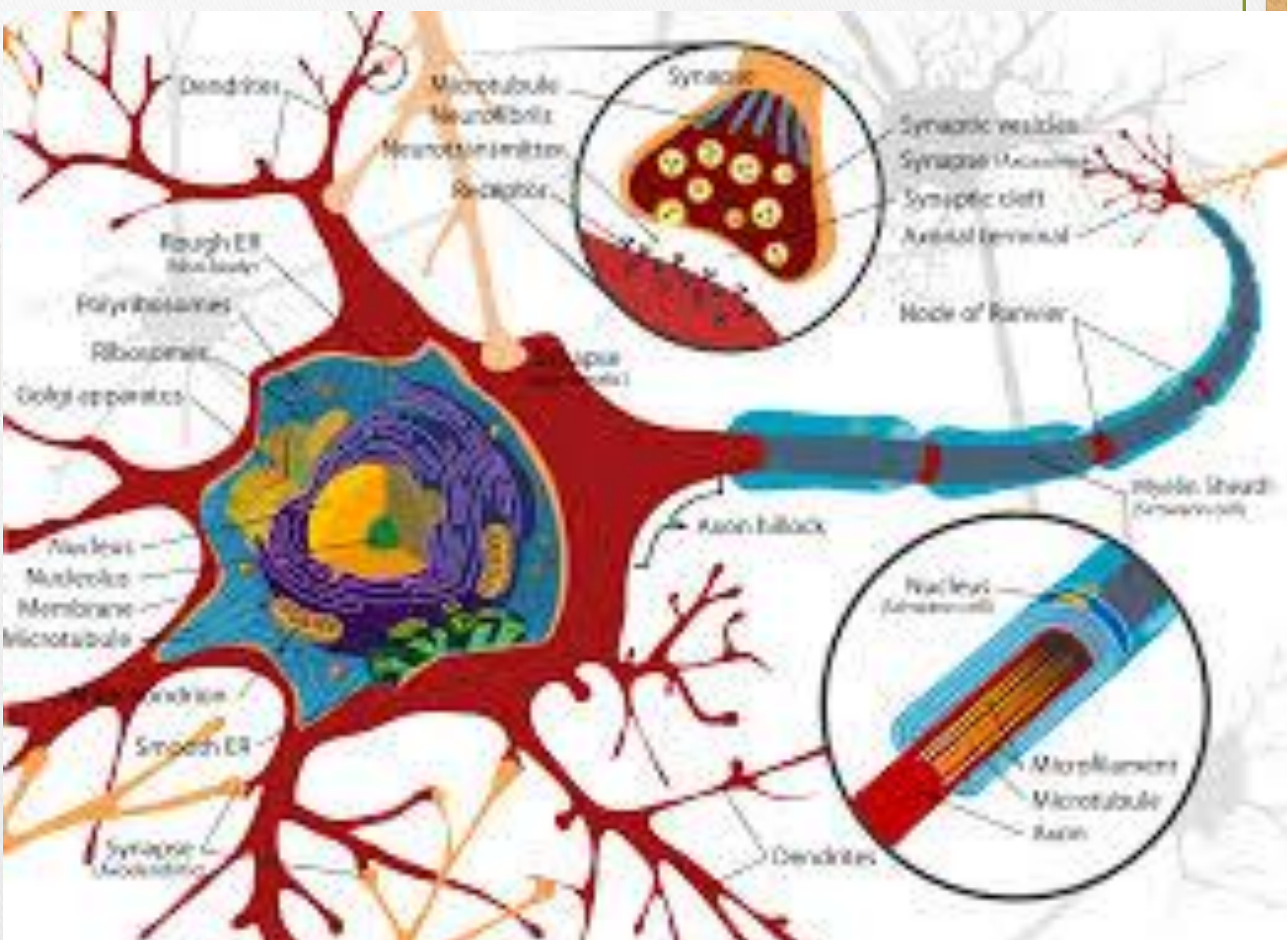
Primary sensory cortex
(postcentral gyrus)
Primary motor cortex
(precentral gyrus)
Somatic sensory association area
Somatic motor association area
(premotor cortex)
Visual association area
Prefrontal cortex
Broca's area
(production of speech)
Visual cortex
Auditory association area
Wernicke's area
(understand speech)
Auditory cortex

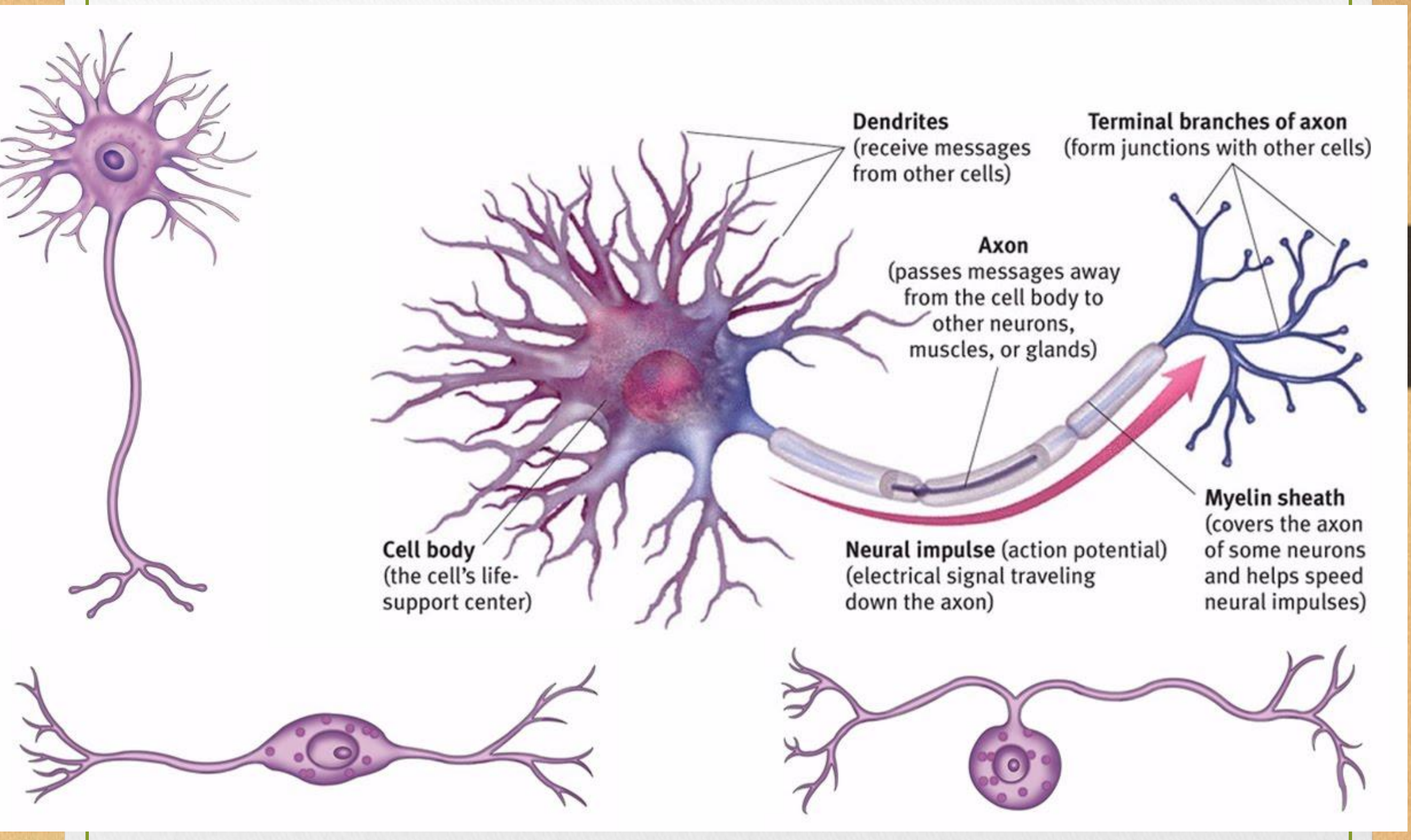

splenium of corpus callosum
praecuneus
subparietal sulcus
sulcus of corpus callosum
marginal portion of sulcus cinguli
central sulcus
paracentral lobule
pineal body
pineal recess
posterior commissure
tela chorioidea of third ventricle
(transverse fissure of cerebrum)
intermed. mass of thalamus
parieto-occipital fissure
gyrus cinguli
thalamus
body of corpus callosum
lamina quadrigemina
body of fornix
lam. of sept. pellucidum
subfrontal portion of sulcus cinguli
cuneus
vermis of cerebellum superior portion
interventricular foramen
column of fornix
anterior commissure
superior frontal gyrus
calcarine fissure
occipital pole
lingual gyrus
occipital lobe
cerebellar hemisphere
frontal pole
medullary substance of vermis
genu of corpus callosum
parolfact. area
rostrum of corpus callosum
anterior parolfactory sulcus
posterior parolfactory sulcus
vermis of cerebell. (inf. portion)
subcallosal gyrus
calamus scriptorius
central canal
pons
hypo-physis
hypothalamic sulcus
lam. terminalis
cerebral aqueduct
poster. lobe
anter. lobe
anter. medullary velum
oculo-motor nerve
mammillary body
optic recess
spinal cord
medulla oblongata
optic nerve
tela chorioidea of fourth ventricle
fourth ventricle
posterior perforated substance
optic chiasma
infundibulum

Challenge: How is Map Produced?



Neurons: Underlying Structure

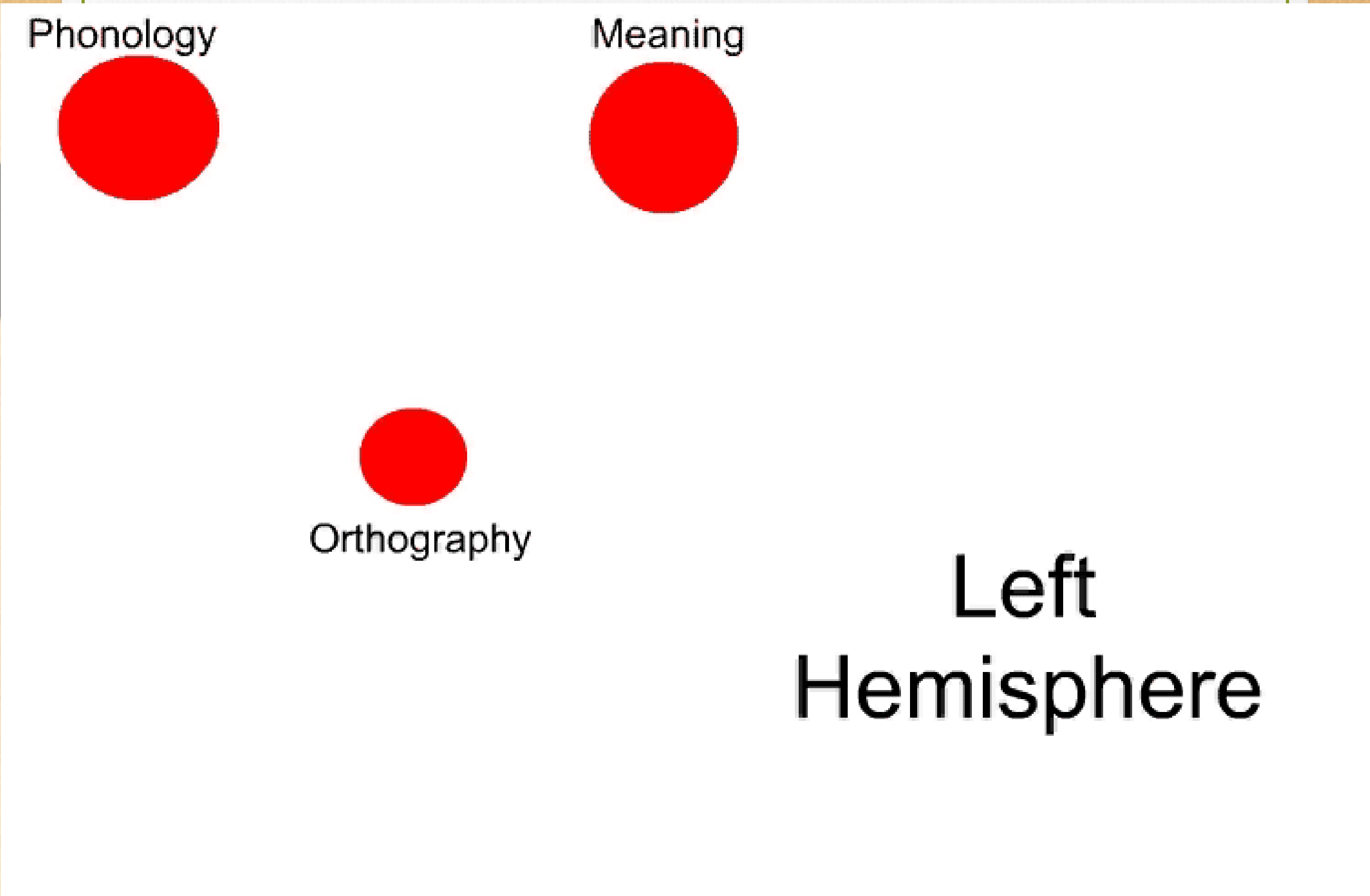


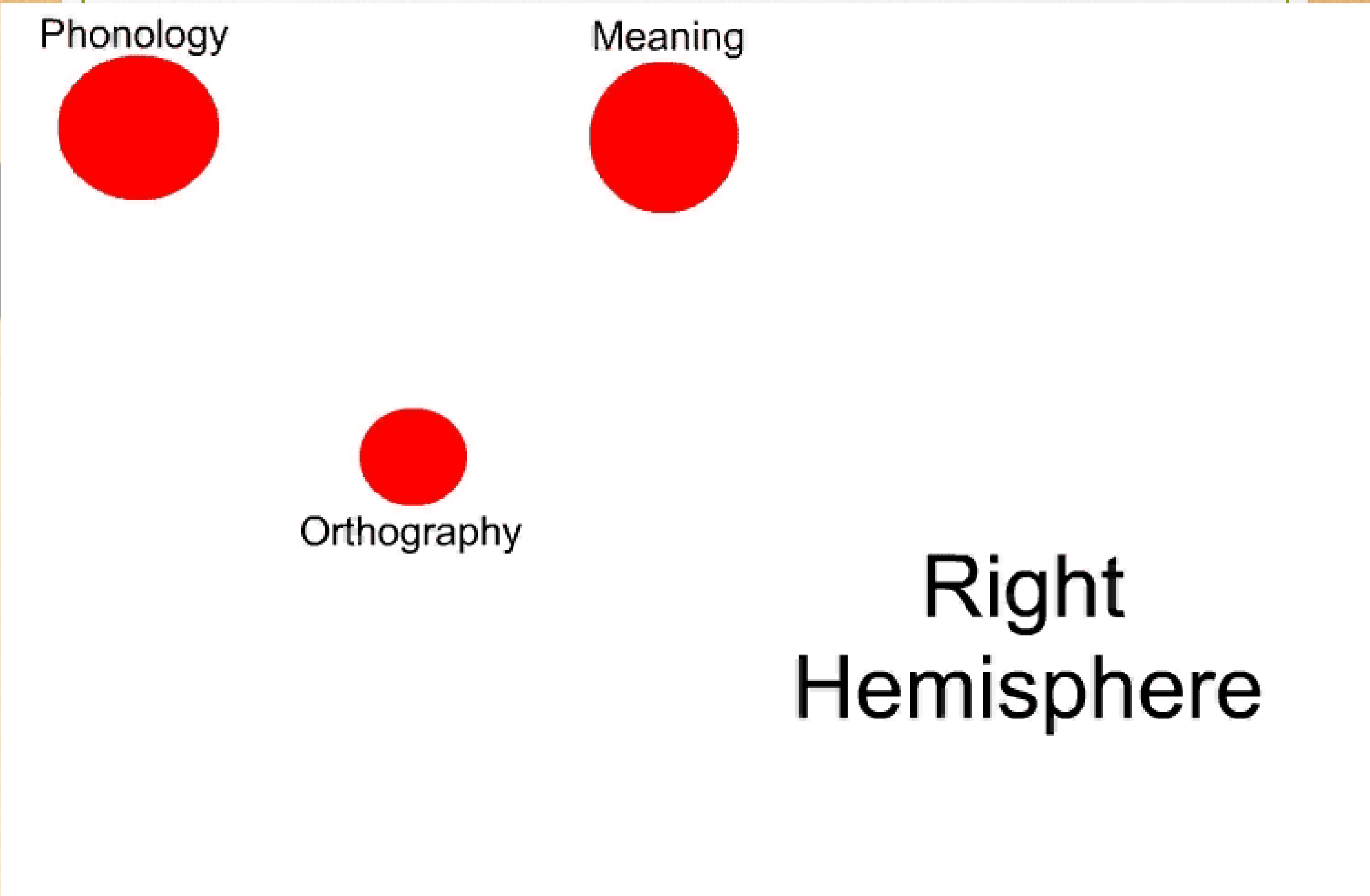

Dendrites
(receive messages from other cells)
Terminal branches of axon
(form junctions with other cells)
Axon
(passes messages away from the cell body to other neurons, muscles, or glands)
Myelin sheath
(covers the axon of some neurons and helps speed neural impulses)
Cell body
(the cell's life-support center)
Neural impulse (action potential)
(electrical signal traveling down the axon)

Split Brain

House – Split Brain Clip

Psychological – Brain Theories and Models


Phonology
Meaning
Orthography
Left Hemisphere


Phonology
Meaning
Orthography
Right
Hemisphere


מוצרי
שתיה
דברי
בישול ומאפה
כלים חד
פעמיים
09:35

Brain vs. Computer

- Brain works on slow components (10**-3 sec)
- Computers use fast components (10**-10 sec)
- Brain more efficient (few joules per operation) (factor of 10**10.)
- Uses massively parallel mechanism.
- ****Can we copy its secrets?

Brain vs Computer

- Areas that Brain is better:

Sense recognition and integration

Working with incomplete information

Pattern Recognition

Generalizing

Learning from examples

Fault tolerant (regular programming is notoriously fragile.)

Psychology

- Personality
- Memory
- Self Reflection
- Love
- Emotions
- Logic
- Altruism
- Reading

Psychology and Psychophysics

- Reaction Time
- Clever experiments
- We'll see some related to memory as time allows

Psychology and Brain

- What is the hardware

Computation and Psychology
Computation and Brain

- This is the main subject of our course
- What is the software/hardware as we see it via computational eyes?
- Two Directions
- Neuroscience understanding
- Understanding how it is possible at all for the brain to compute as it does
- Engineering Methodologies
- Extracting from methods of brain to create new computational methodologies
- |

AI vs. NNs

- AI relates to cognitive psychology
  - Chess, Theorem Proving, Expert Systems, Intelligent agents (e.g. on Internet)
- NNs relates to neurophysiology
  - Recognition tasks, Associative Memory, Learning

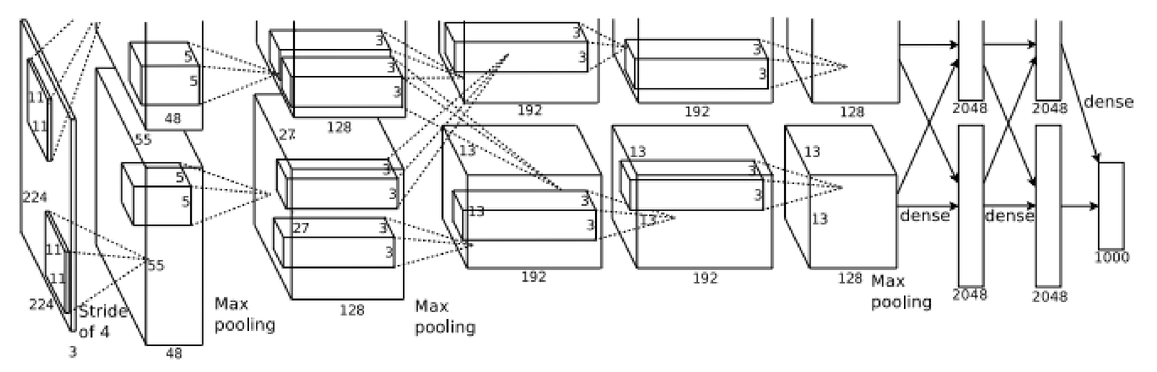
How can NNs work?

- Look at Brain:
- 10**10 neurons (10 Gigabytes) .
- 10 ** 20 possible connections with different numbers of dendrites (reals)
- Actually about 6x 10**13 connections (I.e. 60,000 hard discs for one photo of contents of one brain!)

Brain

- Complex
- Non-linear (more later!)
- Parallel Processing
- Fault Tolerant
- Adaptive
- Learns
- Generalizes
- Self Programs
- GREAT AT PATTERN RECOGNIITON

AlexNet: (60 million parameters 650,000 neurons)

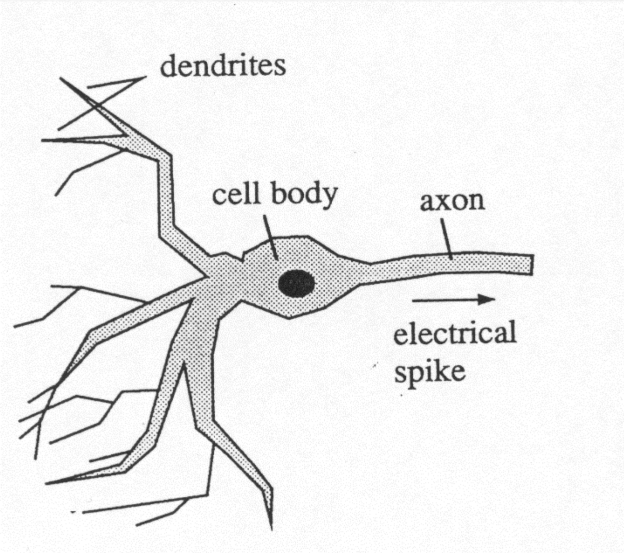


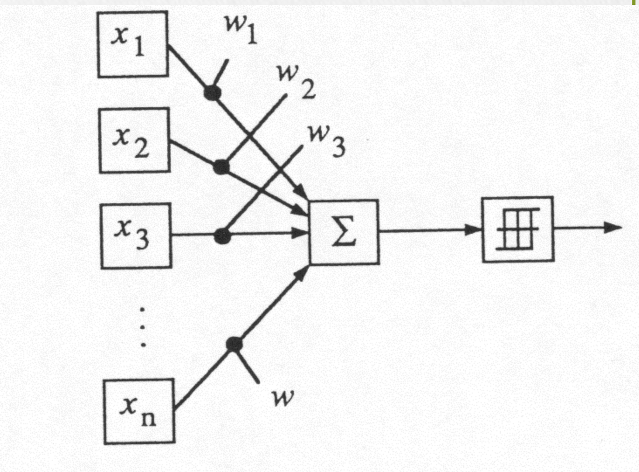
Complex, Large Networks Can Learn Representations of Complex Distributions of Data -but large data sets needed for training

Abstracting

- Note: Number of Neurons may not be crucial (apylsia or crab does many things)
- Look at simple structures first

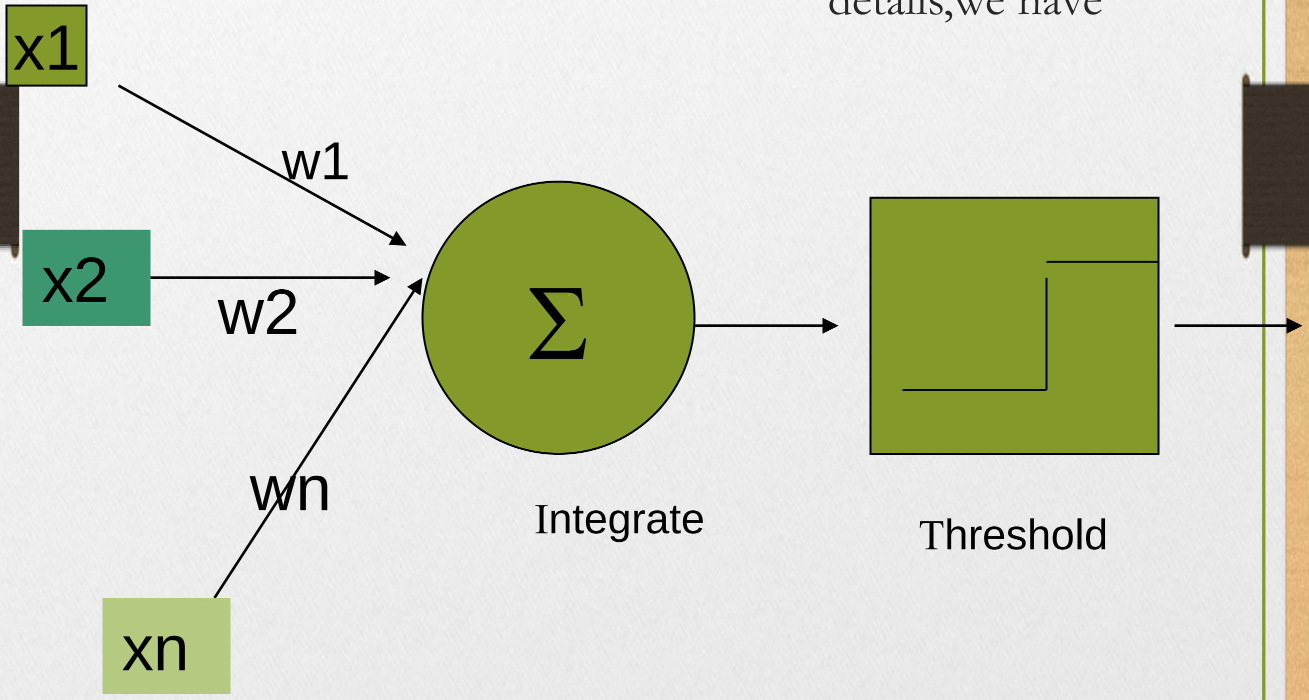
Real and Artificial Neurons





One Neuron
McCullough-Pitts

- This is very complicated. But abstracting the details,we have



(No time in M-P model)

Representability

- What Boolean functions can be represented by a Mccullough-Pitts neuron?

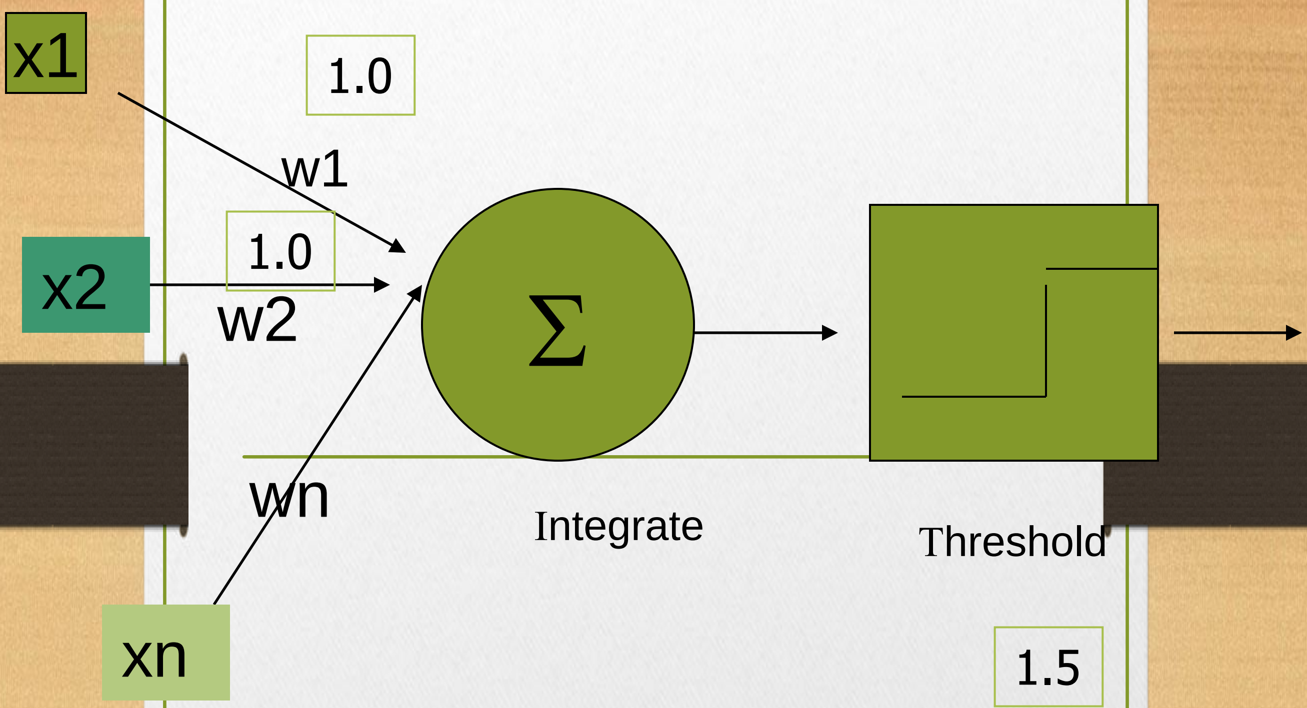
Representability

- What functions can be represented by a network of Mccullough-Pitts neurons?
- Theorem: Every logic function of an arbitrary number of variables can be represented by a three level network of neurons.

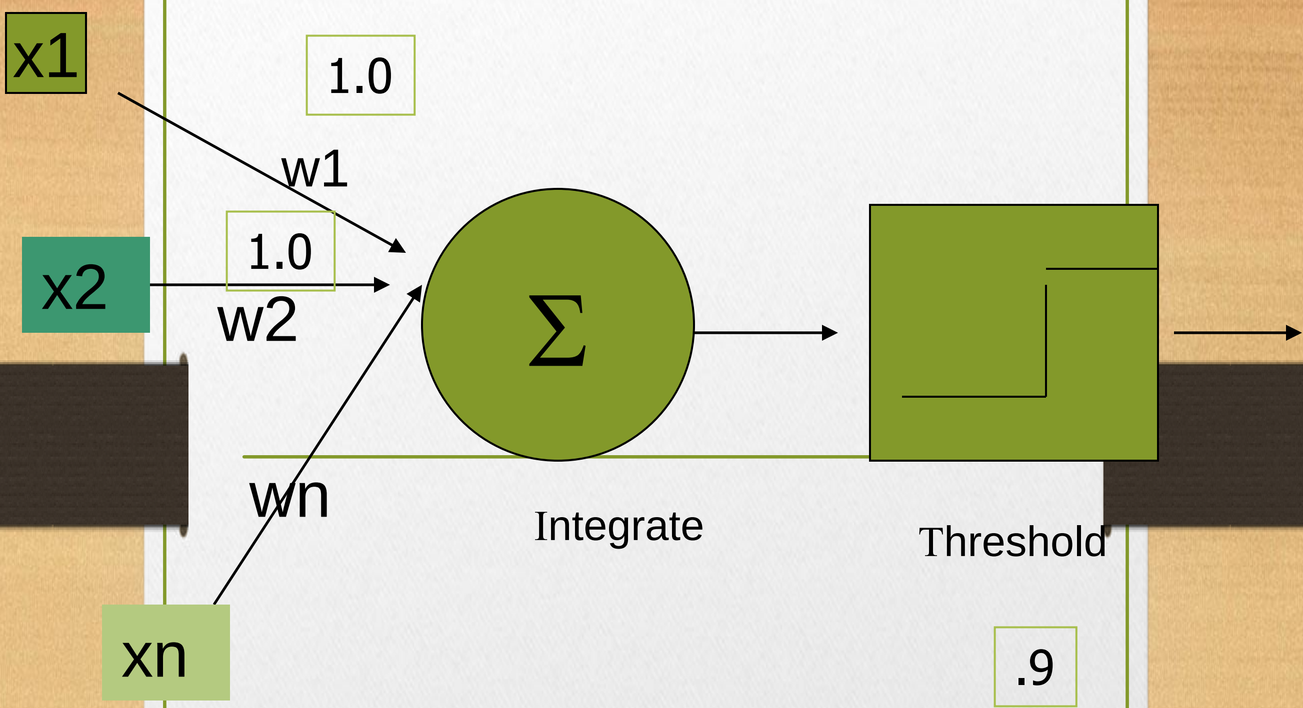
Proof

- Show simple functions: and, or, not, implies
- <u>Recall representability of logic functions by DNF form.</u>

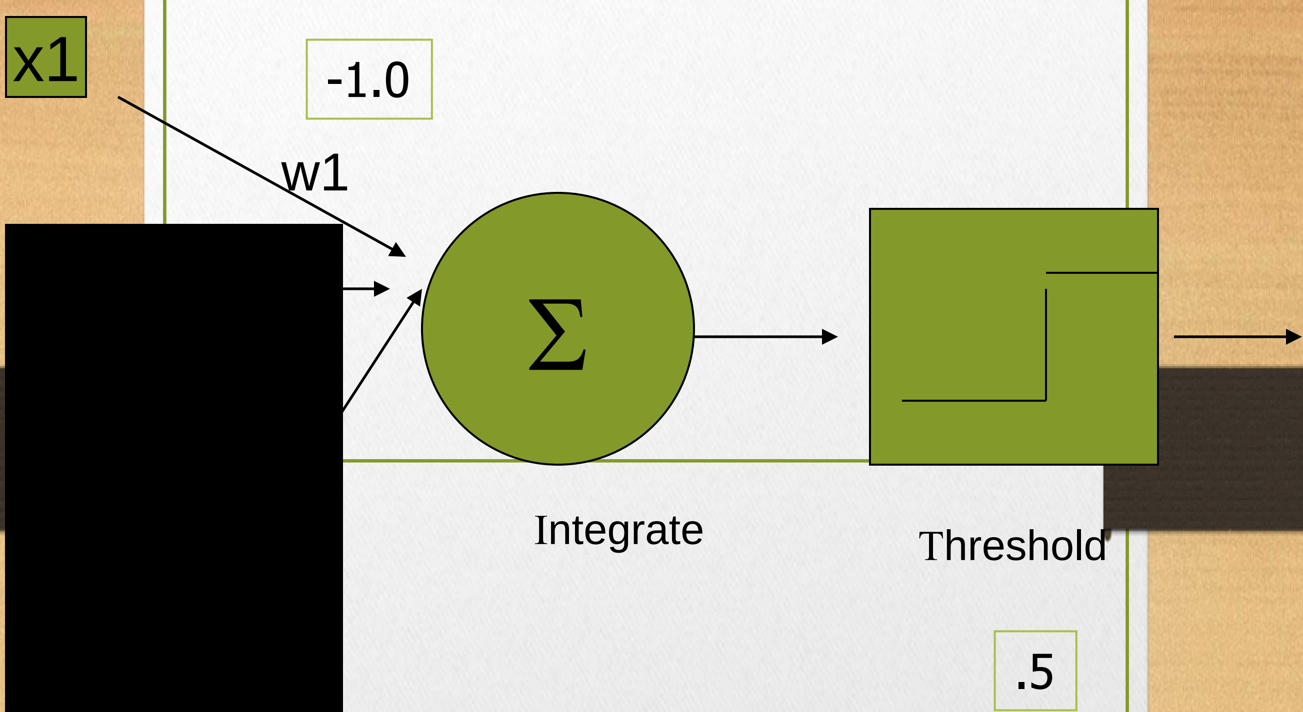
AND, OR, NOT



AND, OR, NOT



AND NOT



DNF and All Functions

- Theorem
  - Any logic (boolean) function of any number of variables can be represented in a network of McCullough-Pitts neurons.
  - In fact the depth of the network is three.
  - Proof: Use DNF and And, Or, Not representation

Other Questions?

- What if we allow REAL numbers as inputs/outputs?

What real functions can be represented?

What if we modify threshold to some other function; so output is not {0,1}. What functions can be represented?

We will return to this question

Representability and Generalizability

Learnability and Generalizability

- The previous theorem tells us that neural networks are potentially powerful, but doesn't tell us how to use them.
- We desire simple networks with uniform training rules.

One Neuron (Perceptron)

- What can be represented by one neuron?
- Is there an automatic way to learn a function by examples?

Perceptron Training Rule

- Loop:

  Take an example. Apply to perceptron.

  If correct answer, return to loop.

  If incorrect, go to FIX.

FIX: Adjust network weights by input example .

Go to Loop:.

Example of Perceptron Learning

- X1 = 1 (+) x2 = -.5 (-)
- X3 = 3 (+) x4 = -2 (-)
- Expanded Vector
  - Y1 = (1,1) (+) y2= (-.5,1)(-)
  - Y3 = (3,1) (+) y4 = (-2,1) (-)

Random initial weight

(-2.5, 1.75)

Graph of Learning

Trace of Perceptron

- W1 y1 = (-2.5,1.75) (1,1)<0 wrong
- W2 = w1 + y1 = (-1.5, 2.75)
- W2 y2 = (-1.5, 2.75)(-.5, 1)>0 wrong
- W3 = w2 – y2 = (-1, 1.75)
- W3 y3 = (-1,1.75)(3,1) <0 wrong
- W4 = w4 + y3 = (2, 2.75)

Perceptron Convergence Theorem

- If the concept is representable in a perceptron then the perceptron learning rule will converge in a finite amount of time.
- (MAKE PRECISE and Prove)

What is a Neural Network?

- What is an abstract Neuron?
- What is a Neural Network?
- How are they computed?
- What are the advantages?
- Where can they be used?

- Agenda
- What to expect

Perceptron Algorithm

- Start: Choose arbitrary value for weights, W
- Test: Choose arbitrary example X
- If X pos and WX >0 or X neg and WX <= 0 go to Test
- Fix:
  - If X pos W := W +X;
  - If X negative W:= W –X;
  - Go to Test;

Perceptron Conv. Thm.

- Let F be a set of unit length vectors. If there is a vector V* and a value e>0 such that V*X > e for all X in F then the perceptron program goes to FIX only a finite number of times.

Proof of Conv Theorem

- Note:

1. By hypothesis, there is a d >0

such that V*X > d   for *all* x e F

1. Can eliminate threshold

(add additional dimension to input) W(x,y,z) > threshold if and only if

W* (x,y,z,1) > 0

2. Can assume all examples are positive ones

(Replace negative examples

by their negated vectors)

W(x,y,z) <0 if and only if

W(-x,-y,-z) > 0.

Proof (cont).

- Consider quotient V*W/|W|.

(note: this is multidimensional

cosine between V* and W.)

Recall V* is unit vector .

Quotient <= 1.

Proof(cont)

- Now each time FIX is visited W changes via ADD.

$$V^* \ W(n+1) = V^*(W(n) + X)$$

$$= V^* \ W(n) + V^*X$$

$$>= V^* \ W(n) + \delta$$

Hence

$$V^* \ W(n) >= n\,\delta \qquad (*)$$

Proof (cont)

- Now consider denominator:
- $|W(n+1)| = W(n+1)W(n+1) =$ $(W(n) + X)(W(n) + X) =$ $|W(n)|^{**}2 + 2W(n)X + 1$

(recall $|X| = 1$ and $W(n)X < 0$ since X is positive example and we are in FIX)

$< \quad |W(n)|^{**}2 + 1$

So after n times

$|W(n+1)|^{**}2 < n \qquad (^{**})$

Proof (cont)

- Putting (*) and (**) together:

Quotient = V*W/|W|

> $n\delta$/ sqrt(n)

Since Quotient <=1 this means

n < $(1/\delta)$**2.

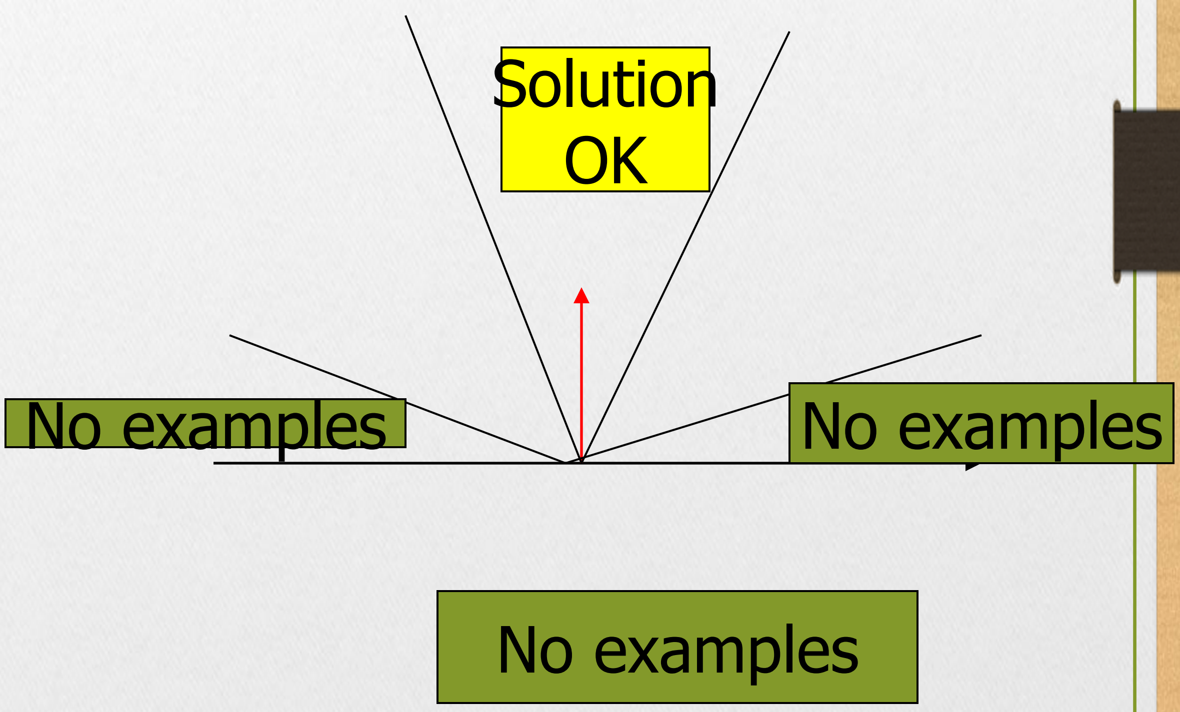
This means we enter FIX a bounded number of times.

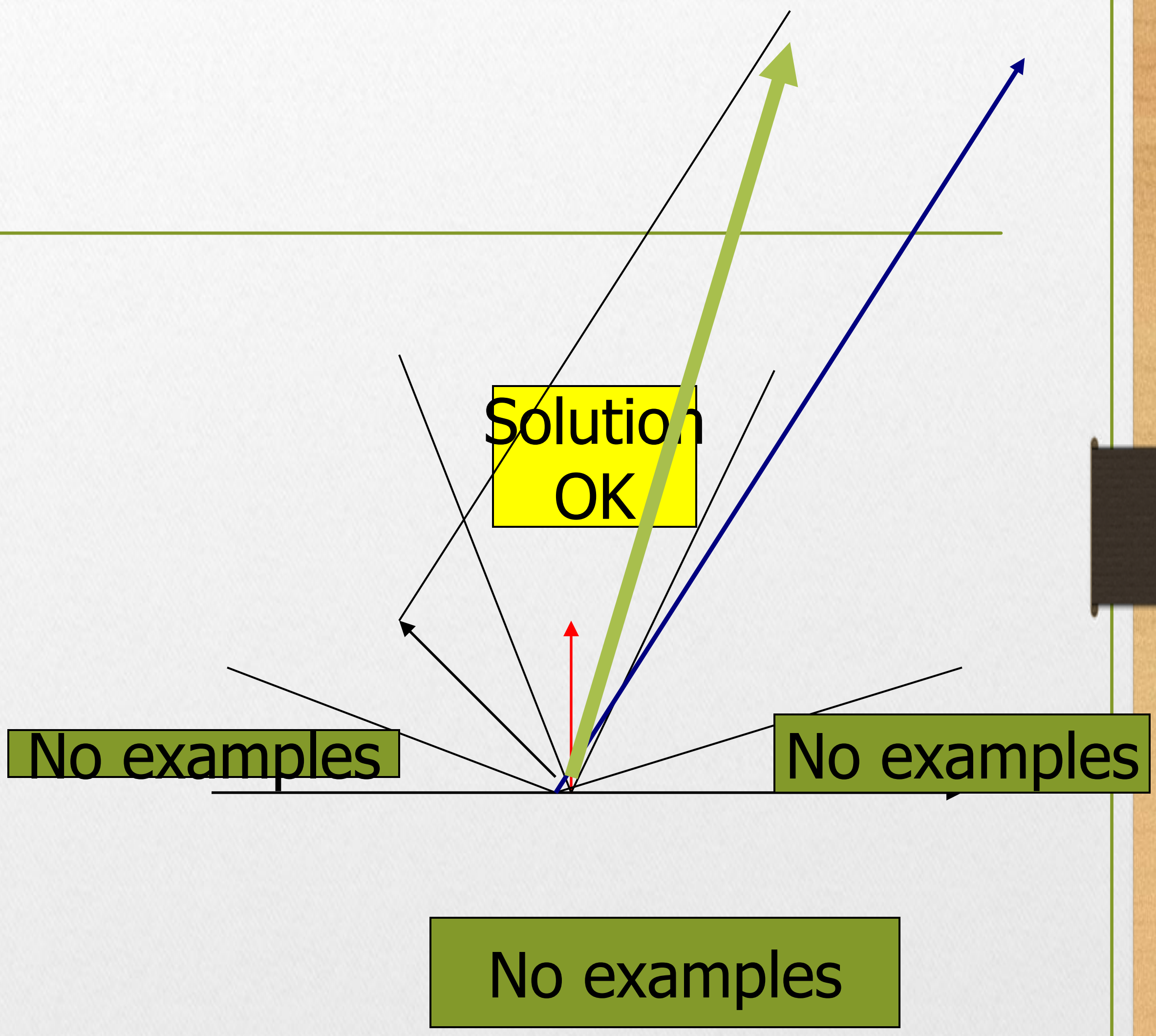
Q.E.D.

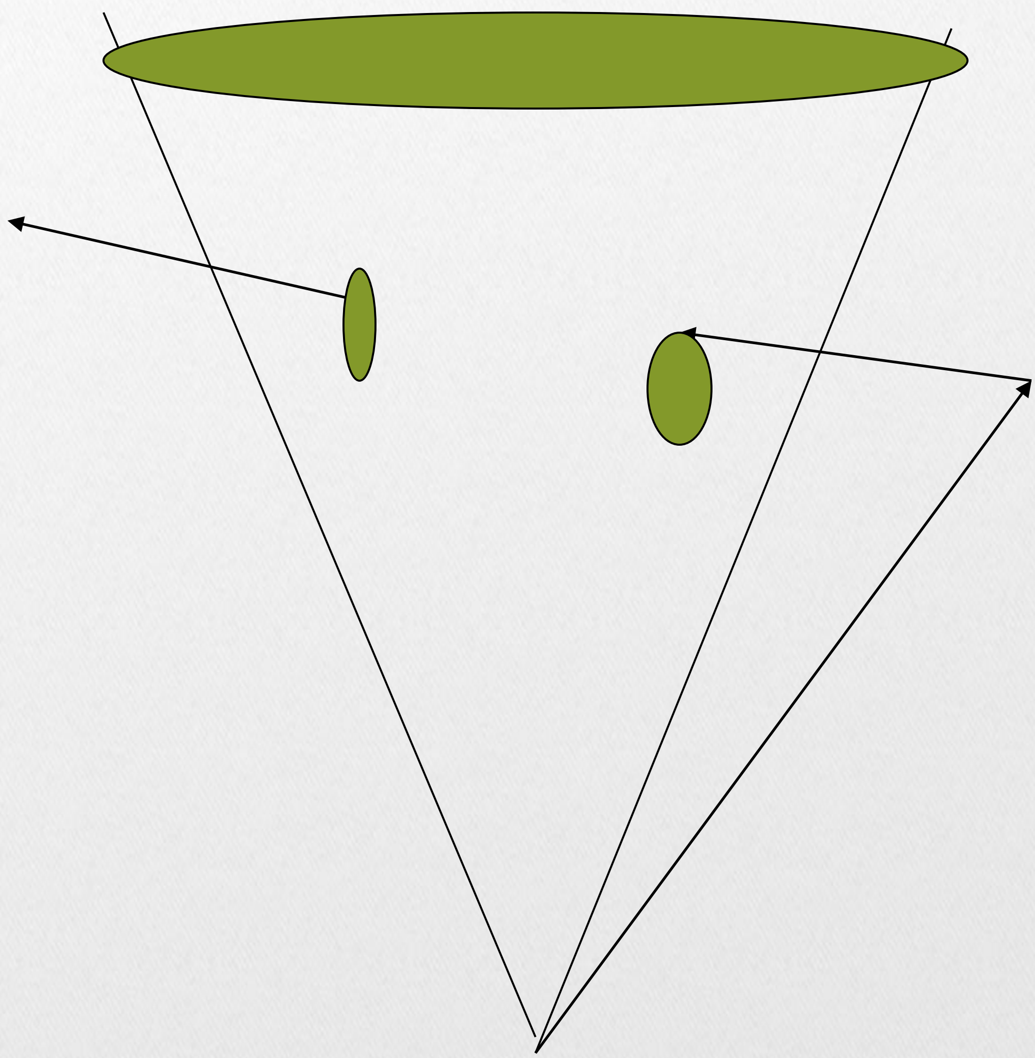
Geometric Proof

- See hand slides.

Perceptron Diagram 1




Solution OK
No examples
No examples
No examples



Additional Facts

- Note: If X's presented in systematic way, then solution W always found.
- Note: Not necessarily same as V*
- Note: If F not finite, may not obtain solution in finite time
- Can modify algorithm in minor ways and stays valid (e.g. not unit but bounded examples); changes in W(n).

Perceptron Convergence Theorem

- If the concept is representable in a perceptron then the perceptron learning rule will converge in a finite amount of time.

Important Points

- Theorem only guarantees result IF representable!
- Usually we are not interested in just representing something but in its generalizability – how will it work on examples we havent seen!

Percentage of Boolean Functions Representable by a Perceptron

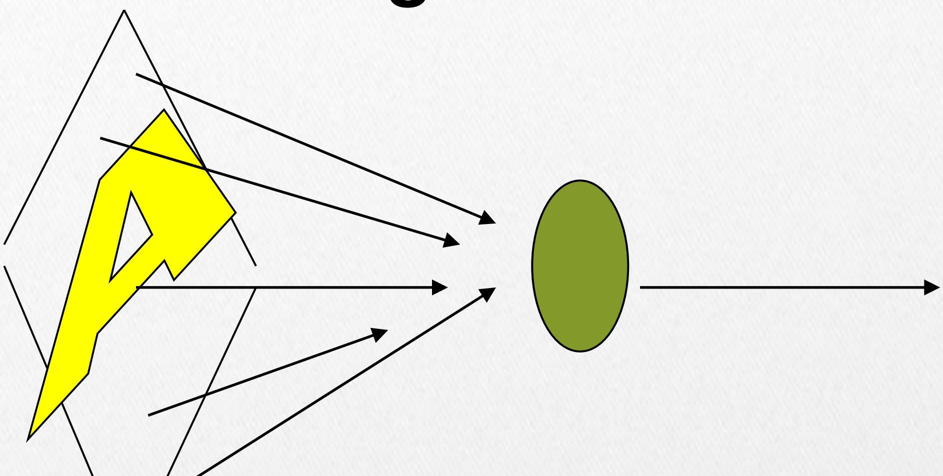
Input	Functions	Perceptron4
1	4	4
2	16	14
3	256	104
4	65,536	1,882
5	10**9	94,572
6	10**19	15,028,134
7	10**38	8,378,070,864
8	10**77	17,561,539,552,946

Generalizability

- Typically train a network on a sample set of examples
- Use it on general class
- Training can be slow; but execution is fast.

Perceptron

- **weights**



$\sum w_i x_i >$ threshold $\Leftrightarrow A$ in receptive field

$\sum w_i x_i > \theta \Leftrightarrow$ The letter A is in the recept

- **Pattern Identification**
- **(Note: Neuron is trained)**

Applying Algorithm to “And”

- W0 = (0,0,1) or random
- X1 = (0,0,1) result 0
- X2 = (0,1,1) result 0
- X3 = (1,0, 1) result 0
- X4 = (1,1,1) result 1

“And” continued

- Wo X1 > 0 wrong;
- W1 = W0 – X1 = (0,0,0)
- W1 X2 = 0 OK (Bdry)
- W1 X3 = 0 OK
- W1 X4 = 0 wrong;
- W2 = W1 +X4 = (1,1,1)
- W3 X1 = 1 wrong
- W4 = W3 –X1 = (1,1,0)
- W4X2 = 1 wrong
- W5 = W4 – X2 = (1,0, -1)
- W5 X3 = 0 OK
- W5 X4 = 0 wrong
- W6 = W5 + X4 = (2, 1, 0)
- W6 X1 = 0 OK
- W6 X2 = 1 wrong
- W7 = W7 – X2 = (2,0, -1)

“And” page 3

- W8 X3 = 1 wrong
- W9 = W8 – X3 = (1,0, 0)
- W9X4 = 1 OK
- W9 X1 = 0 OK
- W9 X2 = 0 OK
- W9 X3 = 1 wrong
- W10 = W9 – X3 = (0,0,-1)
- W10X4 = -1 wrong
- W11 = W10 + X4 = (1,1,0)
- W11X1 =0 OK
- W11X2 = 1 wrong
- W12 = W12 – X2 = (1,0, -1)

What wont work?

- Try XOR.

What wont work?

- Example: Connectedness with bounded diameter perceptron.

- Compare with Convex with

(use sensors of order three).

Limitations of Perceptron

- Representability
  - Only concepts that are linearly separable.
  - Compare: Convex versus connected
  - Examples: XOR vs OR