Eg. $\eta_0 = 0.1$
$T_2 = 1000$
$T_1 = 1000$
$\log(S_0)$

(back propagation; we don't collaborate. we only compute.)

# iterations : usually
$500 \times$ (grid size)

Artificial Neural Networks (ANNs)

Binary inputs — Continuous inputs

Supervised   Unsupervised        Supervised   Unsupervised

| Hopfield Net, Hamming Nets | Carpenter Grossberg Nets | Perceptrons MLP CNNs Act (hata/neurons) |
|---|---|---|

ART
(Adaptive Resonance Theory)

SOM
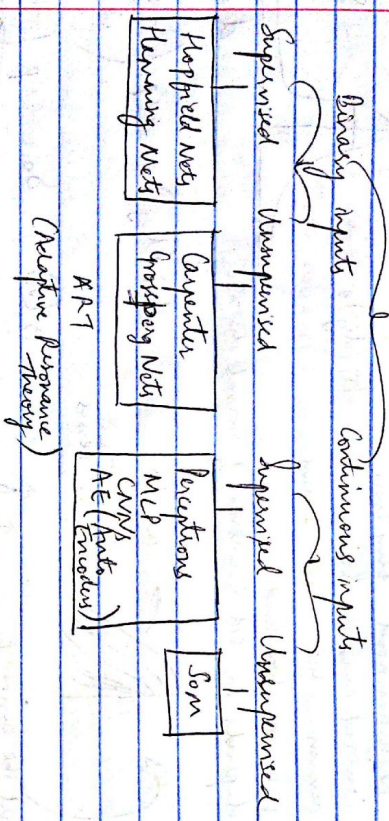
Idea of ANNs is to use Artificial Neurons
→ synapse (from other neurons)

dendrite
input

0   0
1   1   0   0

Nucleus
Cell body

Axon

Axonal Arborization
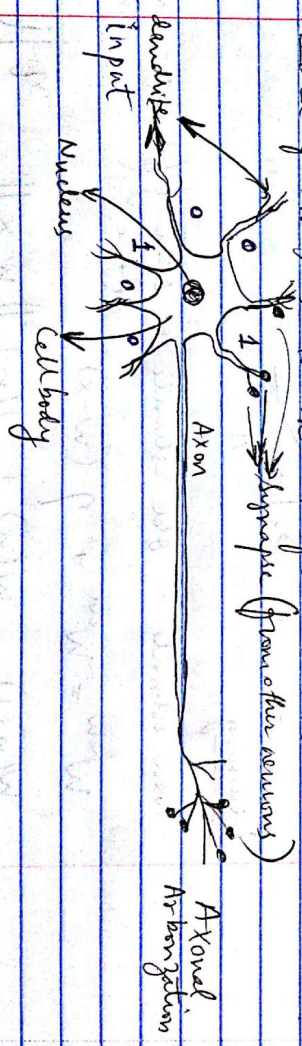
Where is the learning power?
# neurons in zebrafish ≈ 250,000
# neurons in adult humans ≈ $10^{10}$ to $10^{12}$
# connections in " " = $10^{14}$
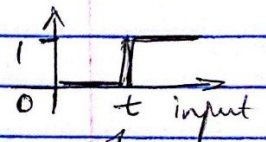
# Abstract model of a Neuron.



$X_1 \xrightarrow{w_0}$ (sum)
$X_2 \xrightarrow{w_2}$ → (activation function) → $y$ < 0 (inhibitory)
$X_n \xrightarrow{w_n}$  1 (exitatory)

inputs

weights — processing unit

$X_0 \quad w_0$

$(X_0 w_0 = bias)$
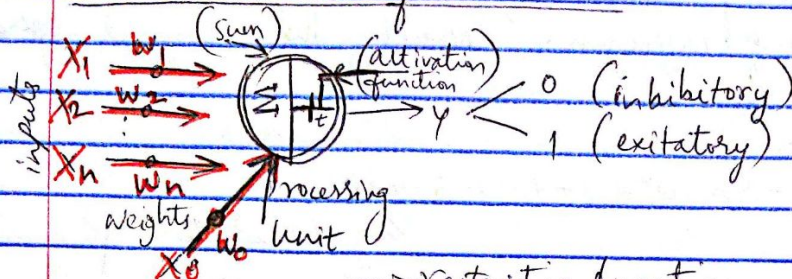
→ restricting function

$$y = g\ (\underbrace{w_1 x_1 + w_2 x_2 + \dots w_n x_n}_{\text{weighted inputs}} \underbrace{- w_0 t_0}_{\text{bias}})$$
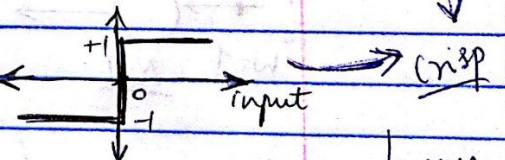
activation function

$$= g\ (\sum x_i w_i - x_0 w_0)$$

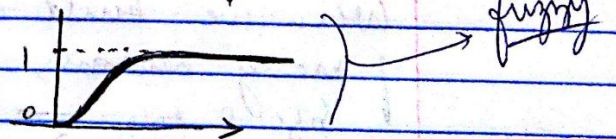For instance: $step(x) = \begin{cases} 1 & \text{if } x \geq t \leftarrow \text{threshold.} \\ 0 & \neq \text{else} \end{cases}$

## Sign function

$$sign(x) = \begin{cases} +1 & \text{if } x \geq 0 \\ -1 & \text{else.} \end{cases}$$
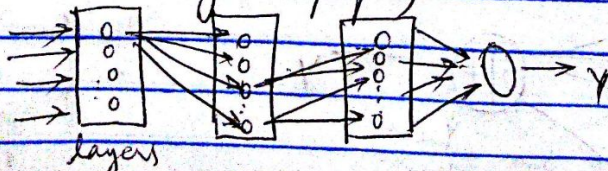
→ crisp

very common activation
function is the Sigmoidal function

→ fuzzy

Using neurons, we could
construct two types of ANNs.
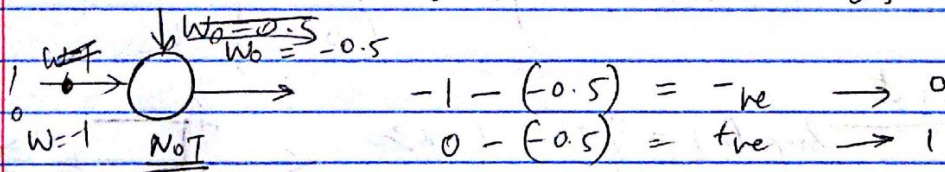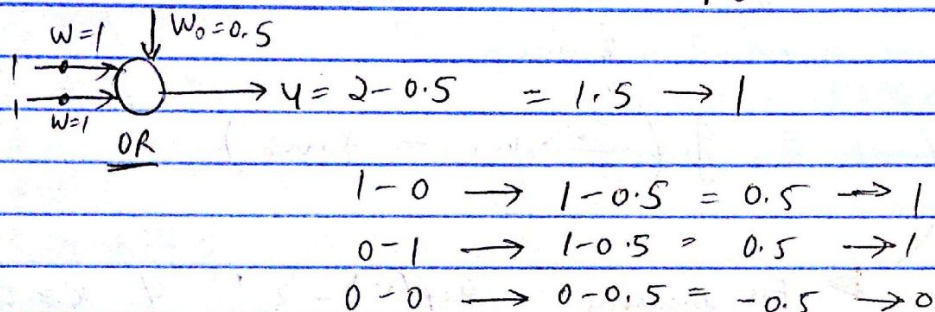
feed-forward          recurrent.
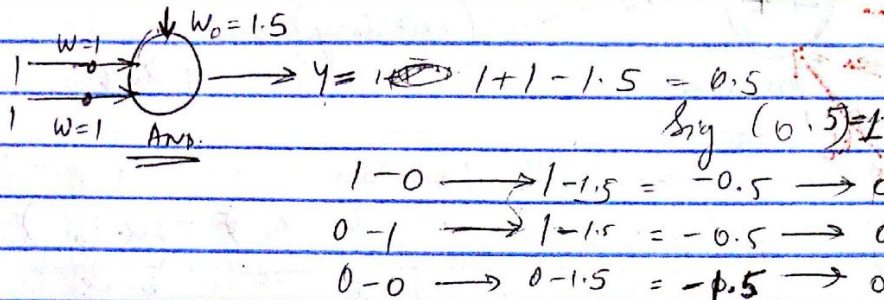
→ connections are unidirectional    → Arbitrary topologies
→ there is no cycle                 → chaotic behavior
→ organized in layers.              → e.g: Hopfield Net

DAG (directed Acyclic Graph)



layers

If ANNs are so fantastic then can they model the most basic piece of Intelligence? Namely the logical operations AND, OR and NOT?

$$W_0 = 1.5$$

$W=1$, $W=1$, AND.

$$y = 1 + 1 - 1.5 = 0.5$$
$$Sig(0.5) = 1$$

$$1 - 0 \longrightarrow 1 - 1.5 = -0.5 \longrightarrow 0$$
$$0 - 1 \longrightarrow 1 - 1.5 = -0.5 \longrightarrow 0$$
$$0 - 0 \longrightarrow 0 - 1.5 = -0.5 \longrightarrow 0$$

$$W = 1 \quad W_0 = 0.5$$
$$W = 1 \quad OR$$

$$y = 2 - 0.5 = 1.5 \longrightarrow 1$$

$$1 - 0 \longrightarrow 1 - 0.5 = 0.5 \longrightarrow 1$$
$$0 - 1 \longrightarrow 1 - 0.5 = 0.5 \longrightarrow 1$$
$$0 - 0 \longrightarrow 0 - 0.5 = -0.5 \longrightarrow 0$$

$$W_0 = -0.5$$
$$W = -1 \quad NOT$$

$$-1 - (-0.5) = -ve \longrightarrow 0$$
$$0 - (-0.5) = +ve \longrightarrow 1$$

When we build an Artificial Neural Network consisting of many neurons, then

size of ANN → too small → not capable of representing the problem
→ too large → cannot generalize to unseen data (overfitting)

* One layer → can approximate any continuous function (= perceptron)

* Two layers → can approximate any function at all.

→ perceptron →

$x_1, x_2, x_3 \ldots x_n$ → sum, activation function → Y
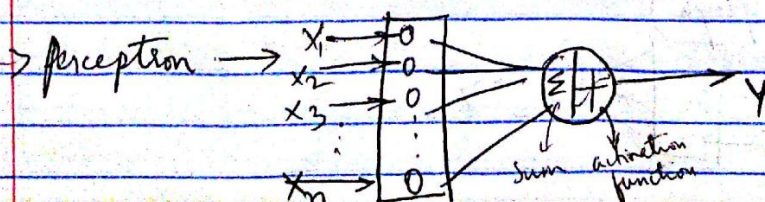
Machine Learning = function approximation

Machine learning is nothing but function approximation

theorem (Minsky & papert 1966 1969)
The perceptron rule converges to weights that correctly classify all training examples provided the given data represents a function that is linearly separable.