

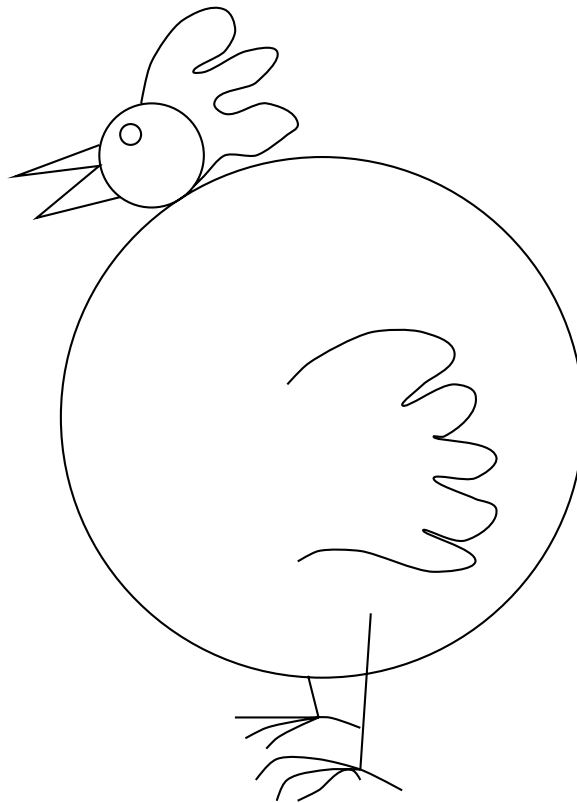
---

How to Read the Mind of A Spherical Chicken

or

Physicists Studying the Brain

---



## General Properties of Neural Systems

---

- Biological systems are inherently noisy: retrieval not exact
- Physical representation is distributed
- No “vacant” parts of the brain for future storage
- No evidence for “decision making” neurons: actions involve large numbers of neurons
- Storage is rewritable
- Not enough DNA information to determine all connections: precise connection positioning cannot be important
- Processing in parallel

## What do we know about our own thinking?

---

### Memory

- Incomplete input gets filled in from memory
- One memory can trigger another separate one: association

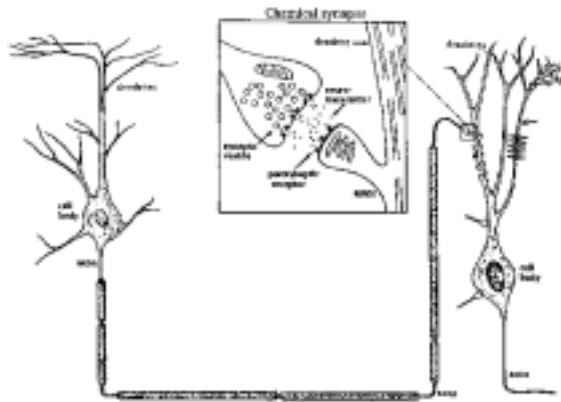
### Learning

- Repeated examples build associations
- Learning can be continuous, or move in “leaps”

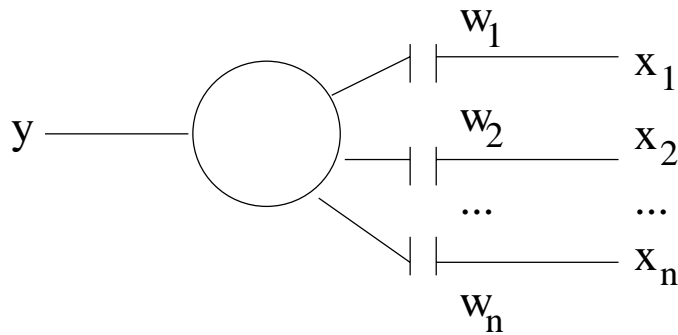
# Neuron for a Spherical Chicken

---

## Biological Neuron



## Model Neuron



- Activation

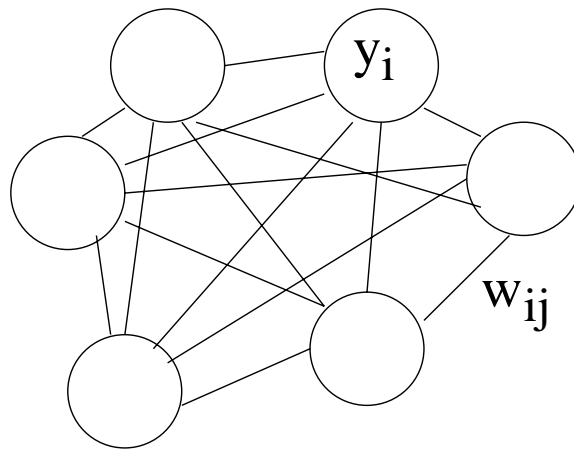
$$y = g(x \cdot w)$$

- Learning

$$\dot{w} = ?$$

## A Model for Associative Memory: Hopfield Net

---



$$\begin{aligned}y_i &= g(x \cdot w) \equiv \text{sgn} \left( \sum_j w_{ij} x_j \right) \\ &= \text{sgn} \left( \sum_j w_{ij} y_j \right)\end{aligned}$$

- To make a memory: have memorized pattern be stable

$$w_{ij} = \frac{1}{N} (x_i x_j)$$

(remember: values are all  $\pm 1$ )

## Energy Formulation of Hopfield Net

---

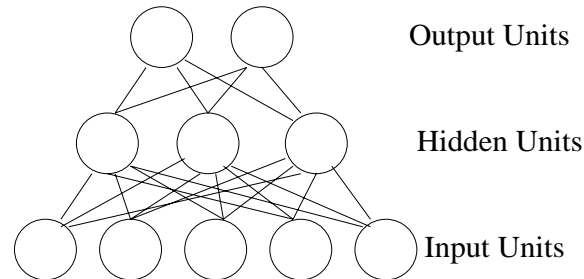
- We want to formulate an “energy” for the net, where, at the minima, the overlap between input pattern and net response is maximized

$$\begin{aligned} H &\equiv -\frac{1}{2N}(y \cdot x)^2 = -\frac{1}{2N}(\sum_i y_i x_i)(\sum_j y_j x_j) \\ &= -\frac{1}{2} \sum_{i,j} \left[ \frac{1}{N}(x_i x_j) \right] y_i y_j \\ &\equiv -\frac{1}{2} \sum_{i,j} w_{ij} y_i y_j \end{aligned}$$

- Now we can use solutions from physics, if one defines the network “energy” properly
- One can use numerical techniques to find the weights by minimizing the energy

# Back Propagation: Minimizing Response Error

---



- Organized into layers: Input, Hidden, Output
- Activation of each layer depends on the previous layer

$$\text{(Input Layer)} \quad y_i^I = x_i$$

$$\text{(Hidden Layer)} \quad y_j^H = g(y^I \cdot w^{IH}) = g\left(\sum_i w_{ji}^{IH} y_i^I\right)$$

$$\text{(Output Layer)} \quad y_k^O = g(y^H \cdot w^{HO}) = \dots$$

- “Energy” (or error) Function defined based on comparison to target values

$$E \equiv \frac{1}{2} \sum_k (t_k - y_k^O)^2$$

- Weights are adjusted using gradient of  $E$ ,  $\dot{w} \propto -\nabla E$ , and errors are propagated back along the network using the chain rule
- Result: network gives a response closer to the target response

## Unsupervised Learning

---

- Useful when the target response is not known
- Network is designed to respond to structure in the input, making use of correlations
- An example is Hebbian Learning

$$y = w \cdot x$$

$$\dot{w}_i = \eta y \cdot x_i$$

- Weights are increased along connections which are active at the same time as the whole neuron
- Result: the neuron becomes more responsive to correlations in the input

# Conclusions

---

