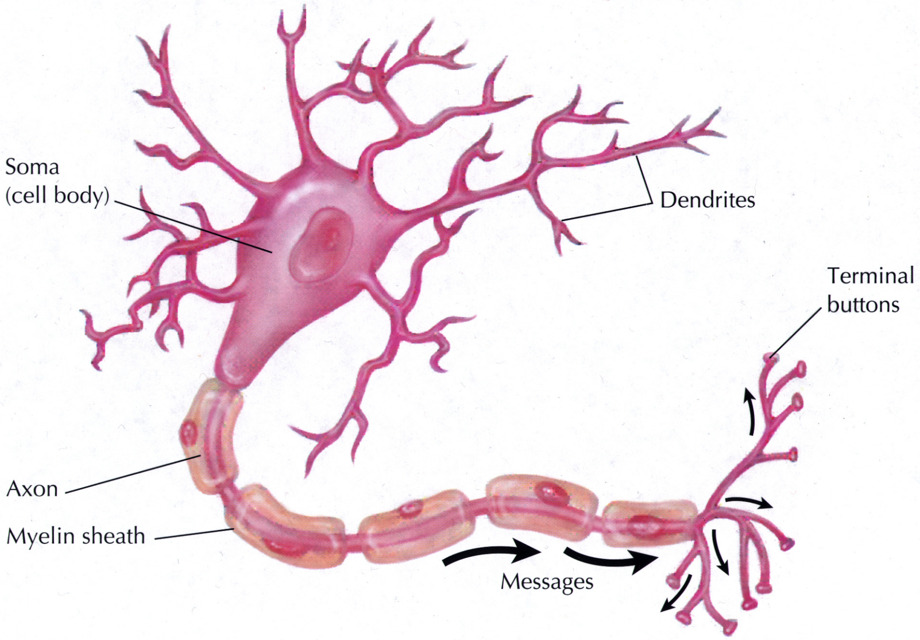
Understanding complex information-processing systems

Marr (1982)

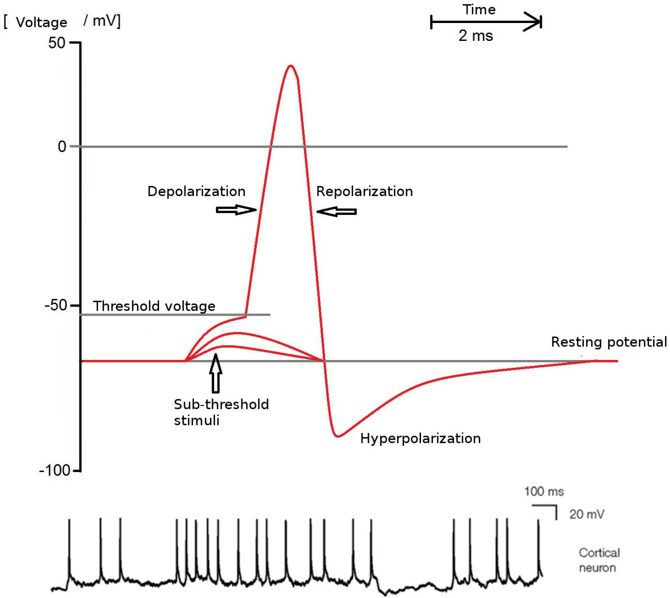
Computational theory
What is the goal of the computation, why is it appropriate, and what is the logic of the strategy by which it can be carried out?

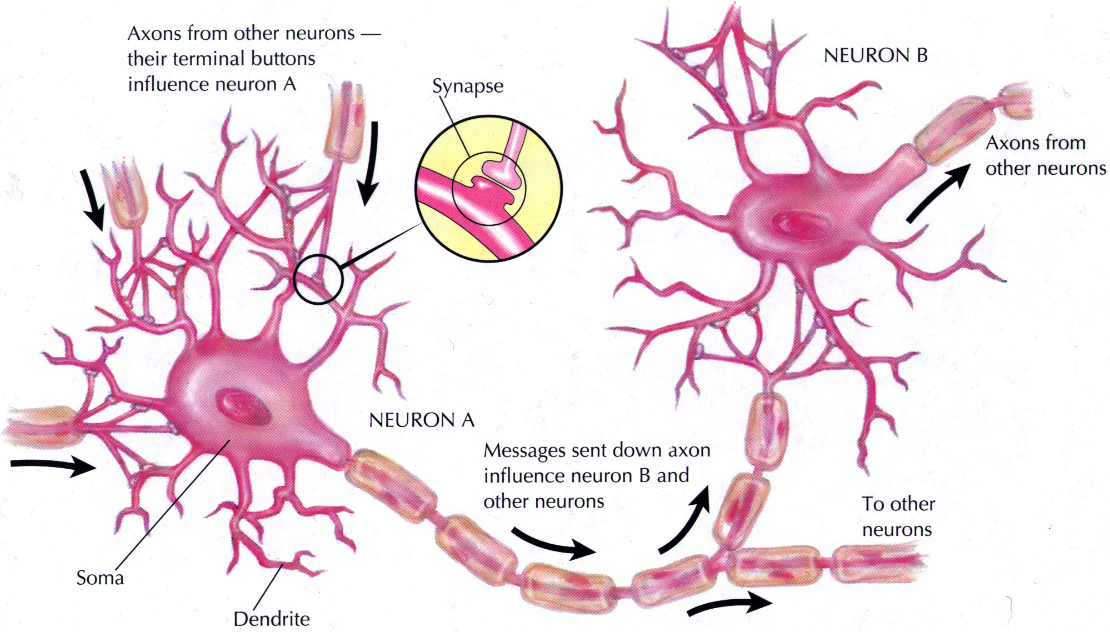
Representation and algorithm
How can this computational theory be implemented? What is the representation for the input and output, and what is the algorithm for the transformation?

Hardware implementation
How can the representation and algorithm be realized physically?

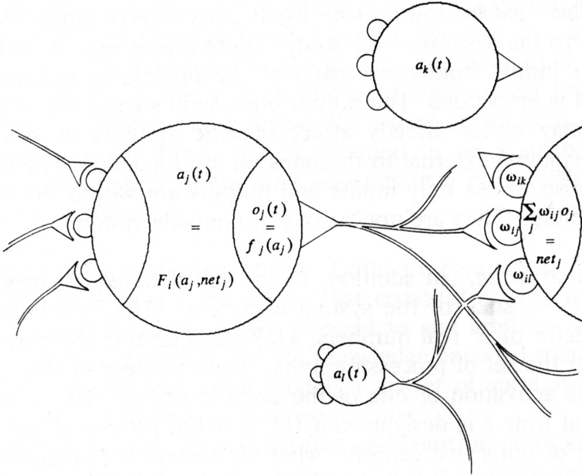

Soma
(cell body)
Dendrites
Terminal
buttons
Axon
Myelin sheath
Messages

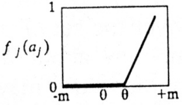
Action potential



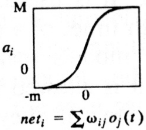


Axons from other neurons — their terminal buttons influence neuron A
Synapse
NEURON B
Axons from other neurons
NEURON A
Messages sent down axon influence neuron B and other neurons
To other neurons
Soma
Dendrite

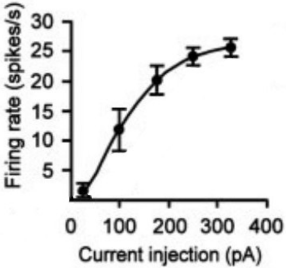




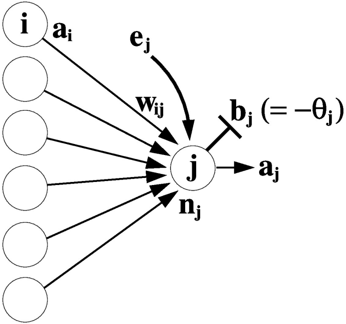
Threshold Output Function



Sigmoid Activation Function



Notation/terminology



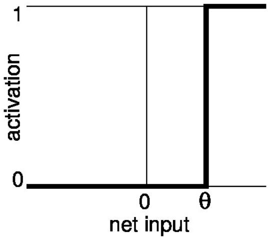
i, j	indices of units (i sending, j receiving)
a_j	**activation** (or "output") of unit j
n_j	summed **net input** to unit j
w_{ij}	**weight** on connection from unit i to unit j
e_j	**external input** to unit j
θ_j	**threshold** for unit j
b_j	**bias** (tonic input) to unit j $(= -\theta_j)$

Types of units

Binary threshold unit

$$n_j = \sum_i a_i w_{ij} + e_j$$

$$a_j = \begin{cases} 1 & \text{if } n_j > \theta_j \\ 0 & \text{otherwise} \end{cases}$$



If "bias" $b_j = -\theta_j$, this is the same as

$$n_j = \sum_i a_i w_{ij} + e_j + b_j \qquad a_j = \begin{cases} 1 & \text{if } n_j > 0 \\ 0 & \text{otherwise} \end{cases}$$

Will generally omit b_j and e_j in equations

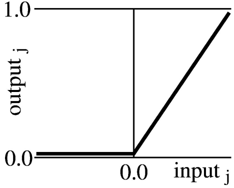
- Bias b_j can be treated as weight w_{ij} from special unit with fixed activation $a_i = 1$.
- External input e_j can be treated as incoming activation a_i across connection with fixed weight $w_{ij} = 1$.

Linear units

$$a_j = n_j = \sum_i a_i w_{ij}$$

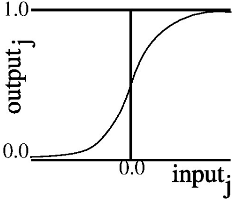
Rectified linear units (ReLUs)

$$a_j = \max(0.0, n_j)$$



Sigmoidal ("logistic", "semi-linear") units

$$a_j = \sigma(n_j) = \frac{1}{1 + \exp(-n_j)}$$



Binary stochastic units

$$p(a_j = 1) = \frac{1}{1 + \exp(-n_j)}$$

Continuous time-averaged (cascaded) units [two alternatives]

$$\textit{input}: \qquad n_j^{[t]} = \tau \textstyle\sum_i a_i^{[t-1]} w_{ij} \;+\; (1-\tau)\; n_j^{[t-1]} \qquad\qquad a_j^{[t]} = \sigma\left(n_j^{[t]}\right)$$

$$\textit{output}: \qquad n_j^{[t]} = \textstyle\sum_i a_i^{[t-1]} w_{ij} \qquad\qquad a_j^{[t]} = \tau\;\sigma\left(n_j^{[t]}\right) \;+\; (1-\tau)\; a_j^{[t-1]}$$

Interactive activation

(Jets & Sharks model; Schema model; McClelland & Rumelhart letter/word model)

$$\begin{aligned} n_j^{[t]} &= \sum_i a_i^{[t-1]} w_{ij} + e_j^{[t]} \\ a_j^{[t]} &= (1-\mathrm{decay})\, a_j^{[t-1]} \;+ \begin{cases} n_j^{[t]} \left(\mathrm{max} - a_j^{[t-1]}\right) & \text{if} \quad n_j^{[t]} > 0 \\ n_j^{[t]} \left(a_j^{[t-1]} - \mathrm{min}\right) & \text{otherwise} \end{cases} \end{aligned}$$

$$\mathrm{decay} = 0.1 \qquad \mathrm{max} = 1.0 \qquad \mathrm{min} = -0.2$$