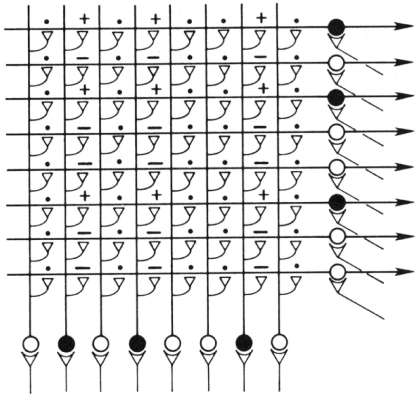
Learning associations in connectionist networks

Association by *contiguity*, generalization by *similarity* (James, 1890)

- Represent items as patterns of activity, where *similarity* is reflected by overlap or correlation between patterns
- Represent *contiguity* as simultaneous presence of patterns over two groups of units (A and B)
- Adjust weights on connections between A and B so that the pattern on A tends to cause the corresponding pattern on B
- As a result, when the same or similar pattern is presented on A, it tends to produce the corresponding pattern on B (perhaps somewhat weakened or distorted)



displayed as in Lens Link Viewer

Correlational learning: Hebb rule

What Hebb actually said:

When an axon of cell A is near enough to excite a cell B and repeatedly and consistently takes part in firing it, *some growth process or metabolic change takes place in one or both cells such that A's efficacy, as one of the cells firing B, is increased.*

The minimal version of the Hebb rule:

When there is a synapse between cell A and cell B, increment the strength of the synapse whenever A and B fire together (or in close succession).

The minimial Hebb rule as implemented in a network:

$$\triangle w_{ij} = \epsilon \; a_i \, a_j$$

notation: $\triangle$ denotes change in a variable;
ϵ is the "learning rate" (how big a change to make)

Correlation

If $w_{ij} = 0$ initially, after a set of n training trials on patterns (indexed by p) where $\triangle w_{ij} = \epsilon\ a_i a_j$,

$$w_{ij} = \epsilon \sum_{p=1}^{n} a_i^{[p]}\ a_j^{[p]}$$

notation: $a_i^{[p]}$ is unit i's activation in pattern p

Suppose a_i and a_j take on values of $+1$ or -1

- If a_i and a_j are *perfectly correlated* (always the same), $a_i^{[p]}\ a_j^{[p]} = 1$, so
$$w_{ij} = \epsilon\ n$$
- If a_i and a_j are *perfectly anticorrelated* (always different), $a_i^{[p]}\ a_j^{[p]} = -1$, so
$$w_{ij} = -\epsilon\ n$$
- If a_i and a_j are *uncorrelated* (different as often as same; *unrelated*, *independent*)
$$w_{ij} = \epsilon\ \left(\frac{n}{2}(+1) + \frac{n}{2}(-1)\right) = 0$$
- If a_i and a_j are *partially correlated* (e.g., 3/4 same and 1/4 different)
$$w_{ij} = \epsilon\ n\ \left(\frac{3}{4}(+1) + \frac{1}{4}(-1)\right) = \frac{1}{2}\epsilon\ n$$
- Thus $w_{ij} \propto \text{correlation}(a_i, a_j)$

Correlation (cont.)

A statistical correlation is defined as

$$\frac{\sum_{p=1}^{n}\left(a_i^{[p]}-\overline{a_i^{[p]}}\right)\left(a_j^{[p]}-\overline{a_j^{[p]}}\right)}{\sqrt{\left(\sum_{p=1}^{n}\left(a_i^{[p]}-\overline{a_i^{[p]}}\right)^2\sum_{p=1}^{n}\left(a_j^{[p]}-\overline{a_j^{[p]}}\right)^2\right)}}$$

where $\overline{a_i^{[p]}}$ is the mean of $a_i^{[p]}$

- Subtraction "takes out the mean"
- Denominator normalizes with respect to the *variance* of a_i and a_j

Hebb rule often includes "reference" values r_i and r_j

$$\triangle w_{ij} = \epsilon \ (a_i - r_i)\,(a_j - r_j)$$

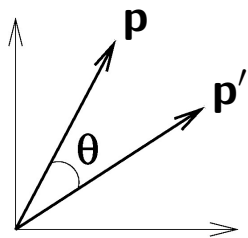
- Ordinarily do not use denominator
- Both issues go away with ± 1, zero-mean patterns

Vector similarity: Dot product

- Inner (dot) product: Measure of **similarity** between two vectors/patterns (e.g. $\mathbf{p}$ and $\mathbf{p}'$)
  Let $a_i^{[\mathbf{p}]}$ be the elements of activity pattern $\mathbf{p}$

$$\mathrm{dp}(\mathbf{p}, \mathbf{p}') = \mathbf{p} \cdot \mathbf{p}' = \sum_i a_i^{[\mathbf{p}]} a_i^{[\mathbf{p}']}$$

$$\cos \theta_{\mathbf{p}\mathbf{p}'} = \frac{\mathbf{p} \cdot \mathbf{p}'}{||\mathbf{p}|| \, ||\mathbf{p}'||}$$



- $\mathbf{p}$ and $\mathbf{p}'$ are *orthogonal* if $\mathrm{dp}(\mathbf{p}, \mathbf{p}') = 0$

- Note that $n_j = \sum_i a_i w_{ij} = \mathrm{dp}(\mathbf{a}, \mathbf{w}_j)$
  - A unit's net input is a measure of the similarity between the input pattern and the unit's "optimal" input (as defined by its weights)

How training patterns influence unit activations

If $w_{ij} = 0$ initially, after training on a set of patterns $\mathbf{p}$ using $\triangle w_{ij} = \epsilon\ a_i\, a_j$,

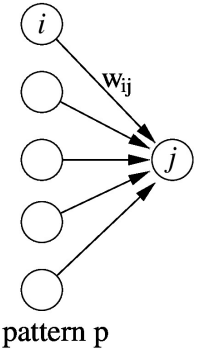
$$w_{ij} = \epsilon \sum_{\mathbf{p}} a_i^{[\mathbf{p}]}\, a_j^{[\mathbf{p}]}$$

note: indexing over patterns $\mathbf{p}$

After training, response of **linear unit** to test pattern $\mathbf{p}'$:

$$\begin{aligned} a_j^{[\mathbf{p}']} \;=\; n_j^{[\mathbf{p}']} &= \sum_i a_i^{[\mathbf{p}']}\, w_{ij} \\ &= \sum_i a_i^{[\mathbf{p}']} \left(\epsilon \sum_{\mathbf{p}} a_i^{[\mathbf{p}]}\, a_j^{[\mathbf{p}]} \right) \\ &= \epsilon \sum_{\mathbf{p}} a_j^{[\mathbf{p}]} \sum_i a_i^{[\mathbf{p}']}\, a_i^{[\mathbf{p}]} \\ &= \epsilon \sum_{\mathbf{p}} a_j^{[\mathbf{p}]}\ \mathrm{dp}(\mathbf{p}', \mathbf{p}) \\ &= \epsilon \sum_{\mathbf{p}} a_j^{[\mathbf{p}]}\ \mathrm{similarity}(\mathbf{p}', \mathbf{p}) \end{aligned}$$

net input $n_j = \sum_i a_i w_{ij}$



Response of output unit j to pattern $\mathbf{p}'$ is *combination* of its responses to trained patterns $\mathbf{p}$, weighted by their *similarity* to $\mathbf{p}'$

If test pattern $\mathbf{p}'$ is orthogonal to all training patterns $\mathbf{p}$, $\mathrm{dp}(\mathbf{p}', \mathbf{p}) = 0$ for all $\mathbf{p}$, so

$$a_j^{[\mathbf{p}']} \;=\; \epsilon \sum_{\mathbf{p}} a_j^{[\mathbf{p}]}\, \mathrm{dp}(\mathbf{p}', \mathbf{p}) \;=\; \epsilon \sum_{\mathbf{p}} a_j^{[\mathbf{p}]}\, 0 \;=\; 0$$

If all training patterns are orthogonal to each other (and assuming $\epsilon = 1$), then

- If $\mathbf{p}'$ is one of the training patterns (say $\mathbf{p}^*$), recall is perfect:

$$a_j^{[\mathbf{p}']} \;=\; a_j^{[\mathbf{p}^*]}\, \mathrm{dp}(\mathbf{p}^*, \mathbf{p}^*) \;+ \sum_{p \neq p^*} a_j^{[\mathbf{p}]}\, \mathrm{dp}(\mathbf{p}', \mathbf{p}) \;=\; a_j^{[\mathbf{p}^*]} \;+ \sum_{p \neq p^*} a_j^{[\mathbf{p}]} 0 \;=\; a_j^{[\mathbf{p}^*]}$$

- If $\mathbf{p}'$ is *similar* to only one training pattern ($\mathbf{p}^*$) and orthogonal to the rest, the output is $a_j^{[\mathbf{p}^*]}$ scaled by the degree of similarity:

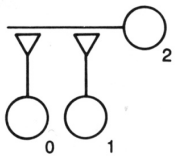
$$a_j^{[\mathbf{p}']} \;=\; a_j^{[\mathbf{p}^*]}\, \mathrm{dp}(\mathbf{p}', \mathbf{p}^*) \;+ \sum_{p \neq p^*} a_j^{[\mathbf{p}]}\, \mathrm{dp}(\mathbf{p}', \mathbf{p}) \;=\; a_j^{[\mathbf{p}^*]}\, \mathrm{dp}(\mathbf{p}', \mathbf{p}^*)$$

In general, the output to any input pattern is a weighted combination of the outputs of all trained patterns, scaled by their similarity to the input.

- If the combination agrees with $a_j^{[\mathbf{p}']}$, this is **facilitation** (or generalization if $\mathbf{p}'$ is novel)
- If the combination disagrees with $a_j^{[\mathbf{p}']}$, this is **interference** (or poor generalization)

Limitations of Hebbian learning

A

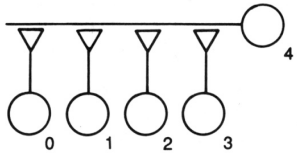


B

Input		Output
0	1	2
+	+	+
+	−	+
−	+	−
−	−	−

- Works for simple task (depends only on line 0)

C

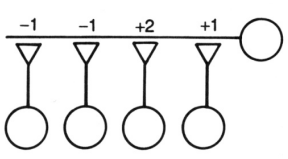


D

Input				Output
0	1	2	3	4
+	−	+	−	+
+	+	+	+	+
+	+	+	−	−
+	−	−	+	−

- Hebb fails: Inputs 0,1,3 are uncorrelated with Outputs, so weights go to zero (and Input 2 alone is insufficient)

E



- However, there *are* weights that work perfectly (produced by "Delta rule"), so Hebb is sub-optimal