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)

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 minimal Hebb rule as implemented in a network:

\[ \Delta w_{ij} = \epsilon \, a_i \, a_j \]

Correlation

A statistical correlation is defined as

\[ \sum_{p=1}^{n} (a_i^{[p]} - \bar{a}_i)(a_j^{[p]} - \bar{a}_j) \]

\[ \sqrt{\left( \sum_{p=1}^{n} (a_i^{[p]} - \bar{a}_i)^2 \right) \left( \sum_{p=1}^{n} (a_j^{[p]} - \bar{a}_j)^2 \right)} \]

where \( \bar{a}_i^{[p]} \) 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 \( r_j \)

\[ \Delta w_{ij} = \epsilon \, (a_i - r_i) \, (a_j - r_j) \]

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

Correlation (cont.)

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

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

Correlation 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 differ), \( a_i^{[p]} a_j^{[p]} = -1 \), so \( w_{ij} = -\epsilon \, n \)
- If \( a_i \) and \( a_j \) are uncorrelated (differ as often as same)

\[ w_{ij} = \epsilon \left( \frac{n}{2}(+1) + \frac{n}{2}(-1) \right) = 0 \]

- 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 \( w_{ij} \propto \text{correlation}(a_i, a_j) \)
Vector similarity: Dot product

- Inner (dot) product: Measure of similarity between two vectors/patterns (e.g. p and p').
  Let \( a[p] \) be the elements of activity pattern p.
  \[
  dp(p, p') = p \cdot p' = \sum_j a[p][j] a[p'][j]
  \]
  \[
  \cos \theta_{pp'} = \frac{p \cdot p'}{|p||p'|}
  \]
- p and p' are orthogonal if \( dp(p, p') = 0 \)
- Note that \( n_j = \sum_i a_i w_{ij} = dp(a, w_j) \)
  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 p using \( \Delta w_{ij} = \epsilon a_i a_j \)

After training, response of linear unit \( j \) to test pattern \( p' \):

\[
\begin{align*}
a_{j}[p'] &= \sum_p a_{j}[p] w_{ij} \\
&= \sum_p a_{j}[p] \left( \epsilon \sum_i a_i a_j \right) \\
&= \epsilon \sum_p a_{j}[p] \sum_i a_i a_j \\
&= \epsilon \sum_p a_{j}[p] \sum_i dp(p', p) \\
&= \epsilon \sum_p a_{j}[p] similarity(p', p)
\end{align*}
\]

Response of output unit \( j \) to pattern \( p' \) is combination of its response to known patterns p, weighted by their similarity to \( p' \).

Limitations of Hebbian learning

- Works for simple task (depends only on line 0)
- Hebb fails: Inputs 0-2 are uncorrelated with Outputs, so weights go to zero (and Input 3 alone is insufficient)
- However, there are weights that work perfectly (produced by "Delta rule"), so Hebb is sub-optimal