CISC 3250
Systems Neuroscience

Neuroplasticity:
Learning in Neurons

Professor Daniel Leeds
dleeds@fordham.edu
JMH 328A

Review of weights

\[ R(t) = \sum_k w_k \alpha_k(t) \]

Weights indicate

- **Connection (0 or not)**
- NT effect
  - \( w > 0 \) excitatory
  - \( w < 0 \) inhibitory
- Magnitude of impact of input

Association

We recall information through associations with other information

- **Pneumonics:**
  - Roy G. Biv
  - Please Excuse My Dear Aunt Sally ()\ Exp x / + -

- **Memories of experiences:**
  - Lake -\> Summer vacation 2014
  - Dealy -\> Final exam Fall 2013

- **Complex objects**
  - ::Bark:: -\> Dog, fur, happy/fear

Features of associators

- **Pattern completion/ generalization**

- **Fault tolerance**
  - Selected dendrites miss input, post-synaptic neuron still fires

- **Learning prototypes**
  - Neuron firing for common combinations
Pattern completion

Activation requires only a subset of desired inputs

How many inputs needed to fire?

Define input \( h = \sum_k w_k r_k^{in} \)
Neuron fires at rate \( r^{out}=1 \) when \( h > 1.5 \)
Assume \( r^{in}=1 \) when active, \( r^{in}=0 \) when inactive

Fault tolerance

Activation requires only a subset of desired inputs

How many inputs needed to fire?

Fault tolerance

Activation requires only a subset of desired inputs

How many inputs needed to fire?

Prototypes

Activation requires all desired inputs

Best labeled as prototype detector for b and c together

Best labeled as pattern completion (just need any three of the inputs to fire)

Fault tolerance

Activation requires only a subset of desired inputs

How many inputs needed to fire?

Fault tolerance

Activation requires only a subset of desired inputs

How many inputs needed to fire?

Fault tolerance

Activation requires only a subset of desired inputs

How many inputs needed to fire?

Fault tolerance

Activation requires only a subset of desired inputs

How many inputs needed to fire?

Fault tolerance

Activation requires only a subset of desired inputs

How many inputs needed to fire?
Learning to associate: Conditioning

Associating both smell and whistle with food
- **Unconditioned stimulus**: smell – already associated with food
- **Conditioned stimulus**: whistle – indicates food coming

Two forms of plasticity
- **Structural plasticity**: generation of new connections between neurons
- **Functional plasticity**: changing strength of connections between neurons

**Hebbian plasticity**: “cells that fire together, wire together”

Chemical level: NT receptors

Increase weight by improving NT detection
- **Post-synaptic**:
  - Insert more receptors into dendrite membrane
  - Improve performance of receptors
- **Pre-synaptic**:
  - Increase amount of NT released
Marr’s levels of analysis

- **Computational theory**: Learn associations among sensations

- **Representation and algorithm**: Associate each sense with set of neural outputs, adjust weights on these outputs into another neuron

- **Hardware implementation**: Insert/remove NT receptors from dendrites

---

**Math of Hebbian rate learning**

“Cells that fire together, wire together”

\[
\Delta w_{ij} = \epsilon(w) r_i r_j
\]

\[r_i \rightarrow r\text{out} \quad r_j \rightarrow r\text{in}\]

- Learning speed

---

**Using the learning rule**

Define input \( h = \sum_k w_k r_k^{\text{in}} \)

Neuron fires at rate \( r^{\text{out}} = 1 \) when \( h > 1 \)

\[\epsilon(w) = \begin{cases} 
-0.5 & w < 0 \\
0.5 & w \geq 0
\end{cases}
\]

\[
\Delta w_{ij} = \epsilon(w) r_i r_j
\]

\[r_i \rightarrow r\text{out} \quad r_j \rightarrow r\text{in}\]

---

**Weight control and decay**

- Synaptic weights are finite
- Propose learning rules that keep weights bounded

\[
\Delta w_{ij} = r_i r_j - c w_{ij}
\]

\[
\Delta w_{ij} = r_i (r_j - w_{ij}) \quad \text{Willshaw}
\]

- Or, preserve total synaptic weight across network: “normalization”

\[w_{ij} \leftarrow \frac{w_{ij}}{\sum_j w_{ij}}\]
Using weight control and decay

Define input $h = \sum_k w_k r_k^{in}$

Neuron fires at rate $r^{out}=1$ when $h > 1$

$$\Delta w_{ij} = r_i (r_j - w_{ij})$$

<table>
<thead>
<tr>
<th>$r^{in}$</th>
<th>$w$</th>
<th>$r^{in}$</th>
<th>$w$</th>
<th>$r^{in}$</th>
<th>$w$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0.5</td>
<td>1</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>1</td>
<td>0.5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

$\Delta w_{ij} = \epsilon(w) r_i r_j$

$\epsilon(w) = \begin{cases} 
-0.5 & w < 0 \\
0.5 & w \geq 0 
\end{cases}$

$w_{ij} \leftarrow w_{ij} - \frac{\Delta w_{ij}}{\sum_j w_{ij}}$