Learning by Training Neural Nets, Part 1

Adopted from P. Winston, Artificial Intelligence, 1992
When collective input at dendrites reaches threshold, pulse travels down axon, causes excitation or inhibition of next neuron.
Real Neural Nets

- Number of neurons in human brain: $\sim 10^{11}$
- Synapses per neuron in cerebellum (motor control): $\sim 10^5$
- Synapses per brain: $\sim 10^{16}$

  Approximately equivalent to 300 times the characters in all books of US Library of Congress
Simulated Neural Nets

- NN consists of neurons or nodes
- NN have links simulating axon-synapse-dendrite connections
- Each link has weight. Like synapse, weight determines nature & strength of connection:
  - Large positive weight → strong excitation
  - Small negative weight → weak inhibition
- Like dendritic mechanisms, the activation function combines input: threshold function sums input values and passes them through threshold; output is 0 or 1.
Simulated Neuron

Activation function sums $n$ products of input $x_i$ and weight $w_i$ and compares result to threshold $T$:

“Fire” if result $\geq T$

$$o_k = 1 \text{ if } \sum_{i=1}^{n} x_i w_i \geq T$$

$$o_k = 0 \text{ else}$$
Simulated Neuron: Trainable Node

Activation function sums $n$ products of input $x_i$ and weight $w_i$, passes result $S$ into function $f$, and outputs $f(S)$.

If $f(S)$ above threshold $T$, output $>0.5$, otherwise $<0.5$. 

$$S = \sum_{i=1}^{n} x_i w_i$$

$$o_k = f(S)$$

if $f(S) >= T$

$$o_k >= 0$$
2-Layer Neural Networks

Inputs

\[ x_1 \]
\[ x_2 \]
\[ \vdots \]
\[ x_k \]
\[ \vdots \]
\[ x_n \]

\[ w_{11} \]
\[ w_{22} \]
\[ w_{k2} \]
\[ w_{n1} \]
\[ w_{n2} \]
\[ w_{nz} \]

\[ f(S) \]

\[ 1 \]

\[ 0 \]

\[ S \]

\[ + \]

\[ o_1 \]
\[ o_2 \]
\[ \vdots \]
\[ o_k \]
\[ \vdots \]
\[ o_z \]

Outputs
Multilayer Neural Networks

Inputs

$x_1$

$x_2$

$\cdots$

$x_k$

$x_n$

Hidden Layers

Output Layer

Outputs

$o_1$

$o_2$

$\cdots$

$o_k$

$\cdots$

$o_z$
Tasks for Neural Networks

- Evaluation Problem
- Training Problem
Single Node Neural Net

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>Computation</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>$0(1) + 0(1) = 0 &lt; 1.5$</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>$0(1) + 1(1) = 1 &lt; 1.5$</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>$1(1) + 0(1) = 1 &lt; 1.5$</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>$1(1) + 1(1) = 2 &gt; 1.5$</td>
<td>1</td>
</tr>
</tbody>
</table>
How can we create a NN to recognize $\text{Or}(x_1, x_2)$?

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>Computation</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0(1)+0(1)=0 $&lt; T$</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0(1)+1(1)=1 $&gt; T$</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1(1)+0(1)=1 $&gt; T$</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1(1)+1(1)=2 $&gt; T$</td>
<td>1</td>
</tr>
</tbody>
</table>
Example of a 3-layer Net

Inputs

$\begin{align*}
x_1 \\
x_2 \\
x_3 \\
x_4 \\
x_5 \\
x_6
\end{align*}$

Hidden Layer

$\begin{align*}
H1 & : T=0.5 \\
H2 & : T=0.5
\end{align*}$

Output Layer

$\begin{align*}
o_1 & : T=1.5 \\
o_2 & : T=-1.5
\end{align*}$
Acquaintance or Sibling Net

Inputs
- Robert
- Rachel
- Romeo

Hidden Layer
- H1
  - T=0.5
  - H2
    - T=0.5
    - T=-1.5

Output Layer
- T=0.5
- T=1.5
- T=-1.5

Outputs
- Acquaintances
- Siblings

Network Connections:
- Robert -> H1: T=0.5
- Rachel -> H1: T=0.5
- Romeo -> H1: T=0.5
- Joan -> H2: T=0.5
- James -> H2: T=0.5
- Juliet -> H2: T=0.5

Connections between Hidden and Output Layers:
- H1 -> Acquaintances: T=1.5
- H2 -> Siblings: T=-1.5
Acquaintances or Siblings?

- Inputs:
  - Robert
  - Rachel
  - Romeo
  - Joan
  - James
  - Juliet

- Outputs:
  - Acquaintances
  - Siblings

T=0.5

T=1.5

T=-1.5

\(_{(1)}+_{(1)}+_{(1)}=2 \quad >0.5 \rightarrow \text{fire}\)

\(_{(1)}+_{(1)}=_{(1)}\)

\(_{(1)}+_{(1)}=_{(1)}\)

\(_{(1)}+_{(1)}=_{(1)}\)

\(_{(-1)}+_{(-1)}=_{(-1)}\)

\(_{(-1)}+_{(-1)}=_{(-1)}\)
Acquaintances or Siblings?

Inputs

- Robert
- Rachel
- Romeo
- Joan
- James
- Juliet

Outputs

- Acquaintances
- Siblings

0(1)+1(1)+1(1)=2
>0.5 \(\rightarrow\) fire

0(1)+0(1)+0(1)=0
<0.5 \(\rightarrow\) don’t fire

1(1)+0(1)=1
<1.5 \(\rightarrow\) don’t fire

1(-1)+0(-1)=-1
>-1.5 \(\rightarrow\) fire
Acquaintances or Siblings?

Inputs
- Robert
- Rachel
- Romeo
- Joan
- James
- Juliet

Outputs
- Acquaintances
- Siblings

Rules:
1. $0(1)+0(1)+0(1)=0 <0.5 \rightarrow \text{don’t fire}$
2. $0(1)+1(1)=1 <1.5 \rightarrow \text{don’t fire}$
3. $1(1)+1(1)+0(1)=2 >0.5 \rightarrow \text{fire}$
4. $0(-1)+1(-1)=-1 >-1.5 \rightarrow \text{fire}$
Acquaintances or Siblings?

Inputs
- Robert
- Rachel
- Romeo
- Joan
- James
- Juliet

Outputs
- Acquaintances
- Siblings

T=0.5
- 0(1)+1(1)+0(1)=1
  >0.5 → fire

T=1.5
- 1(1)+1(1)=2
  >1.5 → fire

T=-1.5
- 1(-1)+1(-1)=-2
  <-1.5 → don’t fire
Tasks for Neural Networks

- Evaluation Problem
  Given a neural net $N$ and input vector $X$:
  
  Does $N$ recognize $X$, i.e., $N(X) = 1$ ?

  **Solution:** Compute output of nodes, layer by layer

  **Examples:** "And" network
  "Or" network
  "Acquaintances/Siblings" network
Tasks for Neural Networks

- **Evaluation Problem**
  Given a neural net $N$ and input vector $X$:
  
  Does $N$ recognize $X$, i.e., $N(X) = 1$?

- **Training Problem**
  Given training set $X_{\text{training}}$ of input vectors:
  
  Find neural net $N$ that recognizes inputs in training set $X_{\text{training}}$ and test set $X_{\text{test}}$. 
Training Problem – Version 1

- Training set $X_{\text{training}} = (X_{\text{positive}}, X_{\text{negative}})$ of input vectors is given
- Number of nodes is given
- Number of layers is given
- Shape of activation function is given
- Links are given

**Goal:** Learn weights and thresholds, such that

$$N(X_{\text{positive}}, i) = 1 \text{ and } N(X_{\text{negative}}, j) = 0$$

for all inputs $i$ in $X_{\text{positive}}$ and $j$ in $X_{\text{negative}}$
Training Problem – Version 2

You, the designer of the AI system, must determine:

- Number of nodes
- Number of layers
- Shape of each activation function and its threshold
- Links between nodes
- Weights on connections
- Representative set $X_{\text{training}}$

Goal: $N(X_{\text{positive}, i}) = 1$ and $N(X_{\text{negative}, j}) = 0$

where $X_{\text{positive}, i}$ and $X_{\text{negative}, j}$ in $X_{\text{test}}$
Training Problem – Version 1

- Assume number of nodes, number of layers, shape of activation function, and links are given.
- Task: Learn weights and thresholds.
- 1st step: Convert thresholds into weights and avoid having to learn two kinds of parameters:
Assume number of nodes, number of layers, shape of activation function, and links are given.

Task: Learn weights and thresholds.

1st step: Convert thresholds into weights and avoid having to learn two kinds of parameters:

“BIAS”
## Single Node Neural Net

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>Computation</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0(1)+0(1)=0 &lt; 0.5</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0(1)+1(1)=1 &gt; 0.5</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1(1)+0(1)=1 &gt; 0.5</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1(1)+1(1)=2 &gt; 0.5</td>
<td>1</td>
</tr>
</tbody>
</table>

$$\text{Or}(x_1, x_2)$$

![Diagram of a single node neural net with inputs $x_1$ and $x_2$, computation output, and threshold $T=0.5$.]
Example: Converting Threshold to Weight

“BIAS”

<table>
<thead>
<tr>
<th>$x_0$</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>Computation</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>-1 (.5) + 0(1) + 0(1) = -0.5 &lt; 0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>0</td>
<td>1</td>
<td>-1 (.5) + 0(1) + 1(1) = 0.5 &gt; 0</td>
<td>1</td>
</tr>
<tr>
<td>-1</td>
<td>1</td>
<td>0</td>
<td>-1 (.5) + 1(1) + 0(1) = 0.5 &gt; 0</td>
<td>1</td>
</tr>
</tbody>
</table>
| -1    | 1     | 1     | -1 (.5) + 1(1) + 1(1) = 1.5 > 0                  | 1      

Input nodes $x_0$, $x_1$, and $x_2$ with weights $0.5$, $1$, and $1$ respectively. The computation output for each combination is shown in the table. The threshold $T=0$ determines the output for the OR function $\text{Or}(x_1, x_2)$.
Acquaintance or Sibling Net with converted thresholds

Inputs
Robert
Rachel
Romeo
Joan
James
Juliet

Hidden Layer
H1
H2

Output Layer
-1
-1
-1
-1

Acquaintances
Siblings

Outputs
Example:

Acquaintance/Sibling Network

After thresholds were converted to weights, the only parameters to learn are the weights.

Solution procedure: Backpropagation
Training Neural Networks

- **Version 1:**
  
  Given a neural net $N$, $X_{\text{training}}$, $X_{\text{validation}}$, and $X_{\text{testing}}$:
  
  Find weights of neural net

  **Solution:** Backpropagation procedure

- **Version 2:**
  
  Given dataset $X$:
  
  1. Find neural network architecture
  2. Design training protocol with $X_{\text{training}}$, $X_{\text{validation}}$, and $X_{\text{testing}}$
  3. Run Backpropagation procedure
Training Neural Networks

Version 1:

- Given a neural net $N$, $X_{\text{training}}$, $X_{\text{validation}}$, and $X_{\text{testing}}$:
  - Find weights of neural net

  Solution: Backpropagation procedure

Version 2:

- Given dataset $X$:
  1. Find neural network architecture
  2. Design training protocol with $X_{\text{training}}$, $X_{\text{validation}}$, and $X_{\text{testing}}$
  3. Run Backpropagation procedure

Easier Problem!
Training Neural Networks

- **Version 1:**
  Given a neural net $N$, $X_{\text{training}}$, $X_{\text{validation}}$, and $X_{\text{testing}}$:
  Find weights of neural net
  
  **Solution:** Backpropagation procedure

- **Version 2:**
  Given dataset $X$:
  1. Find neural network architecture
  2. Design training protocol with $X_{\text{training}}$, $X_{\text{validation}}$, and $X_{\text{testing}}$
  3. Run Backpropagation procedure

Toolbox or Research!
Training Neural Networks

- **Version 1:**
  
  Given a neural net $N$, $X_{\text{training}}$, $X_{\text{validation}}$, and $X_{\text{testing}}$:
  
  Find weights of neural net

  Solution: Backpropagation procedure

- **Version 2:**
  
  Given dataset $X$:
  
  1. Find neural network architecture
  2. Design training protocol with $X_{\text{training}}$, $X_{\text{validation}}$, and $X_{\text{testing}}$
  3. Run Backpropagation procedure

NEXT TOPIC!!!
After we’ve covered the Backprop Procedure
Training Problem – Version 1

Example:
Acquaintance/Sibling Network

First:
Simplified Network:
Acquaintance Net
Acquaintance Net

**Inputs**
- Robert
- Rachel
- Romeo

**Hidden Layer**
- H1
- H2

**Output Layer**
- Output

If output > 0.9
Then the two “1 inputs” are acquaintances
If output < 0.1
Then not acquaintances
If 0.1 <= output <= 0.9
Then ambiguous

Only two inputs may be 1
Labeled Training Data: 15

<table>
<thead>
<tr>
<th>Robert</th>
<th>Raquel</th>
<th>Romeo</th>
<th>Joan</th>
<th>James</th>
<th>Juliet</th>
<th>Acquaintant</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</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>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Acquaintance Net

Inputs
Robert
Rachel
Romeo

Hidden Layer
H1
w1
w2
w3
w4

Output Layer
H2
w5
w6
w7
w8

Output
Acquaintances
w9
w10
w11
Backprop for the Acquaintance Net

Initial Values of 11 Weights:

<table>
<thead>
<tr>
<th>Weight</th>
<th>Initial Value</th>
<th>Value after Backprop</th>
</tr>
</thead>
<tbody>
<tr>
<td>w1</td>
<td>0.1</td>
<td>1.99</td>
</tr>
<tr>
<td>w2</td>
<td>0.2</td>
<td>4.65</td>
</tr>
<tr>
<td>w3</td>
<td>0.3</td>
<td>4.65</td>
</tr>
<tr>
<td>w4</td>
<td>0.4</td>
<td>4.65</td>
</tr>
<tr>
<td>w5</td>
<td>0.5</td>
<td>2.28</td>
</tr>
<tr>
<td>w6</td>
<td>0.6</td>
<td>5.28</td>
</tr>
<tr>
<td>w7</td>
<td>0.7</td>
<td>5.28</td>
</tr>
<tr>
<td>w8</td>
<td>0.8</td>
<td>5.28</td>
</tr>
<tr>
<td>w9</td>
<td>0.9</td>
<td>9.07</td>
</tr>
<tr>
<td>w10</td>
<td>1</td>
<td>6.27</td>
</tr>
<tr>
<td>w11</td>
<td>1.1</td>
<td>6.12</td>
</tr>
</tbody>
</table>
RMS error during Training of Acquaintance Net

Satisfactory performance after 225 weight changes

225 x 15 = 3,375 inputs processed

Learning rate = 1
Characteristics of Back-Propagation

- Training may require thousands of backpropagations
Characteristics of Back-Propagation

- Training may require thousands of backpropagations
- Training can get stuck or become unstable:
  - $r = 1.0$  225 weight changes
  - $r = 2.0$  150 weight changes
  - $r = 0.25$  900 weight changes
  - $r = 0.5$  425 weight changes
  - $r = 4.0$  serious instability
  - $r = 8.0$  serious instability
Backprop can get stuck or become unstable
Characteristics of Back-Propagation

- Training may require thousands of backpropagations
- Training can get stuck or become unstable
  - No general learning rate rule
  - Rate selection is problem dependent

  If learning rate too low: slow training
  If learning rate too high: instability
Characteristics of Back-Propagation

- Training may require thousands of backpropagations
- Training can get stuck or become unstable
- Training can be done in stages:
  Later stages refine training of network in earlier stages
Characteristics of Back-Propagation

- Training may require thousands of backpropagations
- Training can get stuck or become unstable
- Training can be done in stages:
  Later stages refine training of network in earlier stages

Example:

To train Acquaintance or Sibling Net, use the trained Acquaintance Net as the pre-trained model and extend the model by one output node
Acquaintance or Sibling Net

Inputs
- Robert
- Rachel
- Romeo
- Joan
- James
- Juliet

Outputs
- Acquaintances
- Siblings

Diagram:
- H1
- H2
- w1, w2, w3, w4, w5, w6, w7, w8, w9, w10, w11, w12, w13, w14
## 2-Stage Training of Ac/Sib Net

<table>
<thead>
<tr>
<th>Weight</th>
<th>Initial Value</th>
<th>Value after Pretraining</th>
<th>Value after Sibling Training</th>
</tr>
</thead>
<tbody>
<tr>
<td>w1</td>
<td>0.1</td>
<td>1.99</td>
<td>2.71</td>
</tr>
<tr>
<td>w2</td>
<td>0.2</td>
<td>4.65</td>
<td>6.02</td>
</tr>
<tr>
<td>w3</td>
<td>0.3</td>
<td>4.65</td>
<td>6.02</td>
</tr>
<tr>
<td>w4</td>
<td>0.4</td>
<td>4.65</td>
<td>6.02</td>
</tr>
<tr>
<td>w5</td>
<td>0.5</td>
<td>2.28</td>
<td>2.89</td>
</tr>
<tr>
<td>w6</td>
<td>0.6</td>
<td>5.28</td>
<td>6.37</td>
</tr>
<tr>
<td>w7</td>
<td>0.7</td>
<td>5.28</td>
<td>6.37</td>
</tr>
<tr>
<td>w8</td>
<td>0.8</td>
<td>5.28</td>
<td>6.37</td>
</tr>
<tr>
<td>w9</td>
<td>0.9</td>
<td>9.07</td>
<td>10.29</td>
</tr>
<tr>
<td>w10</td>
<td>1</td>
<td>6.27</td>
<td>7.04</td>
</tr>
<tr>
<td>w11</td>
<td>1.1</td>
<td>6.12</td>
<td>6.97</td>
</tr>
<tr>
<td>w12</td>
<td>1.2</td>
<td>-8.32</td>
<td></td>
</tr>
<tr>
<td>w13</td>
<td>1.3</td>
<td>-5.72</td>
<td></td>
</tr>
<tr>
<td>w14</td>
<td>1.4</td>
<td>-5.68</td>
<td></td>
</tr>
</tbody>
</table>
RMS Error during Two-State Training

- Initial 225 cycles
- Extra 175 cycles
- 400 cycles total

RMS error vs. Weight change cycles

- Acquaintance Training
- Sibling Training

400 cycles total
Simultaneous Training of 14 Weights of Full Acquaintance/Sibling Net

![Graph showing Simultaneous and Sequential Training](image)
Characteristics of Back-Propagation

- Training may require thousands of backpropagations
- Training can get stuck or become unstable
- Training can be done in stages
- Trained neural nets can make predictions
## Labeled Dataset: 15 Samples

<table>
<thead>
<tr>
<th>Robert</th>
<th>Raquel</th>
<th>Romeo</th>
<th>Joan</th>
<th>James</th>
<th>Juliet</th>
<th>Acquaintance</th>
<th>Sibling</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
We used all 15 samples for training! Nothing left for testing…

<table>
<thead>
<tr>
<th></th>
<th>Robert</th>
<th>Raquel</th>
<th>Romeo</th>
<th>Joan</th>
<th>James</th>
<th>Juliet</th>
<th>Acquaintance</th>
<th>Sibling</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Use 3 Samples for Testing, Train on the Remaining 12 Samples

<table>
<thead>
<tr>
<th>Robert</th>
<th>Raquel</th>
<th>Romeo</th>
<th>Joan</th>
<th>James</th>
<th>Juliet</th>
<th>Acquaintance</th>
<th>Sibling</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

53
## Testing Result:

<table>
<thead>
<tr>
<th></th>
<th>Acquaintance</th>
<th></th>
<th>Sibling</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Desired</td>
<td>Computed</td>
<td>Desired</td>
<td>Computed</td>
</tr>
<tr>
<td>Robert/Juliet</td>
<td>1</td>
<td>0.92</td>
<td>0</td>
<td>0.06</td>
</tr>
<tr>
<td>Romeo/Joan</td>
<td>1</td>
<td>0.92</td>
<td>0</td>
<td>0.06</td>
</tr>
<tr>
<td>James/Juliet</td>
<td>0</td>
<td>0.09</td>
<td>1</td>
<td>0.91</td>
</tr>
</tbody>
</table>

**Interpretation:**
Trained Acq/Sib net deals successfully with previously unseen data = it can predict!
Characteristics of Back-Propagation

- Training may require thousands of backpropagations
- Training can get stuck or become unstable
- Training can be done in stages
- Trained neural nets can make predictions
- Excess weights lead to overfitting
Would a net with more trainable weights do better?

Inputs
- Robert
- Rachel
- Romeo
- Joan
- James
- Juliet

Outputs
- Acquaintances
- Siblings
Training only takes 300 cycles – the extra weights make it too easy to deal with the training set.
Testing Result:

<table>
<thead>
<tr>
<th></th>
<th>Acquaintance</th>
<th></th>
<th>Sibling</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Desired</td>
<td>Computed</td>
<td>Desired</td>
<td>Computed</td>
</tr>
<tr>
<td>Robert/Juliet</td>
<td>1</td>
<td>0.99</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Romeo/Joan</td>
<td>1</td>
<td>0.06</td>
<td>0</td>
<td>0.94</td>
</tr>
<tr>
<td>James/Juliet</td>
<td>0</td>
<td>0.97</td>
<td>1</td>
<td>0.01</td>
</tr>
</tbody>
</table>

Interpretation: **Overfitting Occurred**
Trained Acq/Sib net does not deal successfully with previously unseen data.
It cannot predict two of the three test cases correctly.
Heuristic to Avoid Overfitting

Number of trainable weights influencing a particular output should be less than the number of training samples

- In Acquaintance/Sibling Network with two hidden nodes: 11 trainable weights & 12 input-output samples: a dangerously small margin!
- In Acquaintance/Sibling Network with three hidden nodes: 19 trainable weights & 12 input-output samples: 7 (>50%) more weights than i/o samples – overfitting is inevitable!
Characteristics of Back-Propagation

shown for Sibling/Acquaintance Neural Net
generalizes to all neural networks

- Training may require thousands of backpropagations
- Training can get stuck or become unstable
- Training can be done in stages
- Trained neural nets can make predictions
- Excess weights lead to overfitting
Patrick Winston
(1943–2019)

“Neural-net experts are artists; they are not mere handbook users.”