CAS CS 640
Artificial Intelligence

Lectures by Margrit Betke

Learning by Training Neural Nets

Adopted from P. Winston, Artificial Intelligence, 1992

CAS CS 640

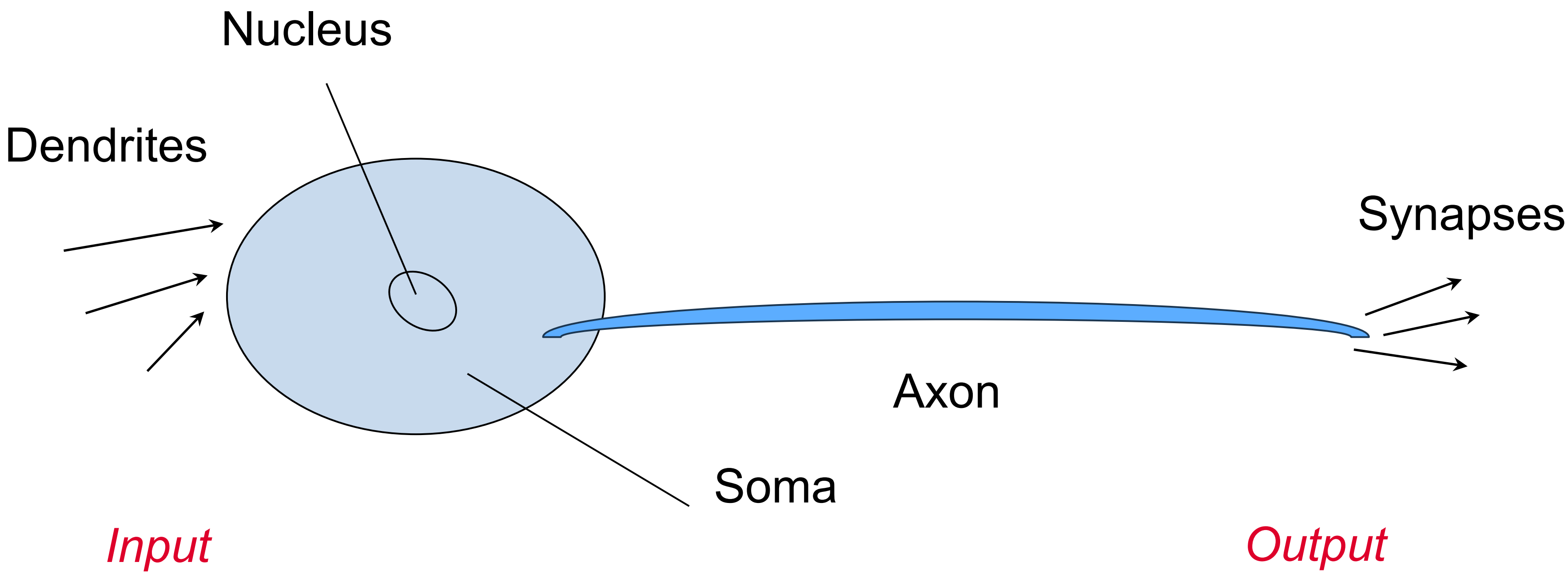
Artificial Intelligence

Lectures by Margrit Betke

Learning by Training Neural Nets, Part 1

Adopted from P. Winston, Artificial Intelligence, 1992

Real Neuron



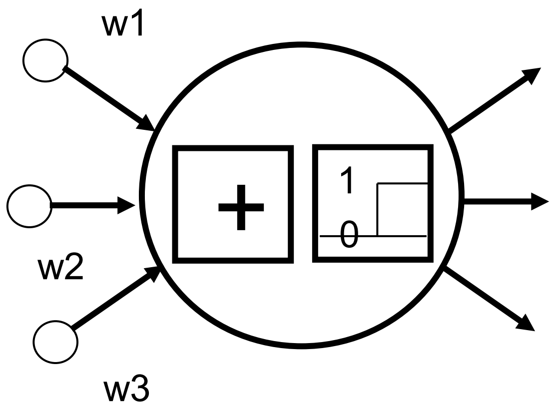
When collective input at dendrites reaches threshold, pulse travels down axon, causes excitation or inhibition of next neuron.

Real Neural Nets

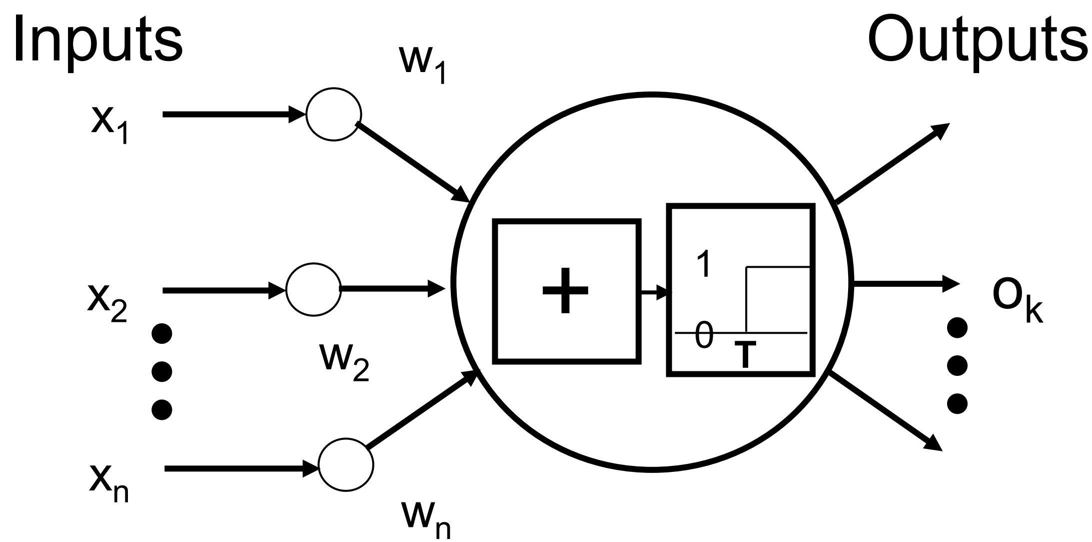
- Number of neurons in human brain: $\sim 10^{11}$
- Synapses per neuron in cerebellum (motor control): $\sim 10^{5}$
- Synapses per brain: $\sim 10^{16}$
  - Approximately equivalent to 300 times the characters in all books of US Library of Congress

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:

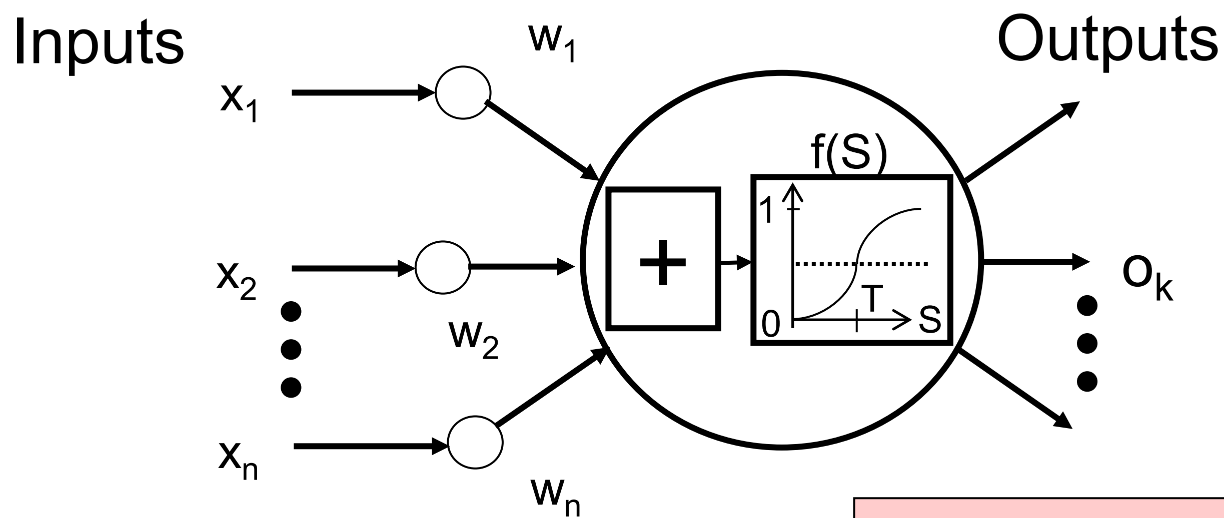
"Fire" if result >= T

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

$$o_k = 0 \text{ else}$$

Simulated Neuron: Trainable Node

differentiable f



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.

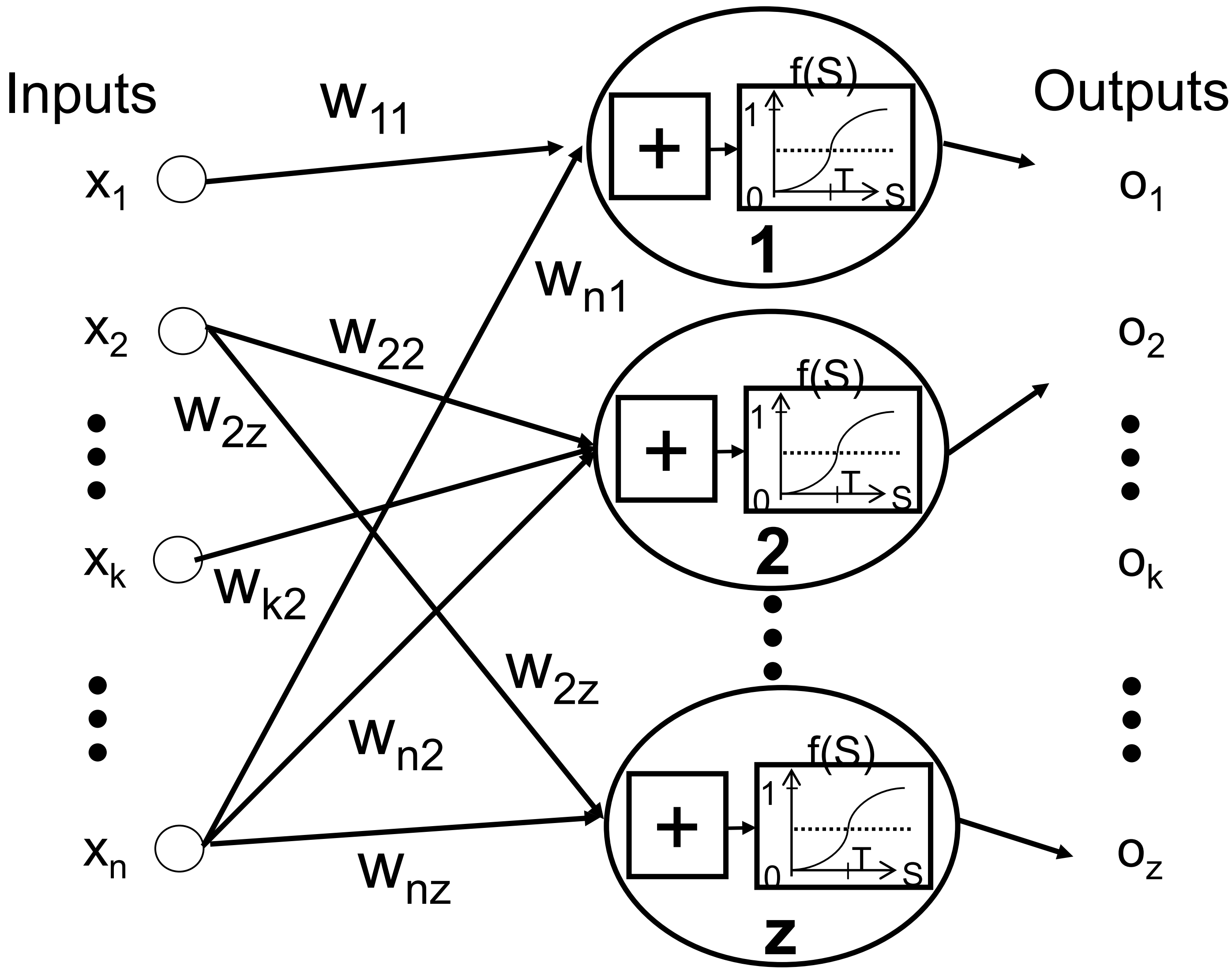
$$S = \sum_{i=1}^{n} x_i w_i$$

$$o_k = f(S)$$

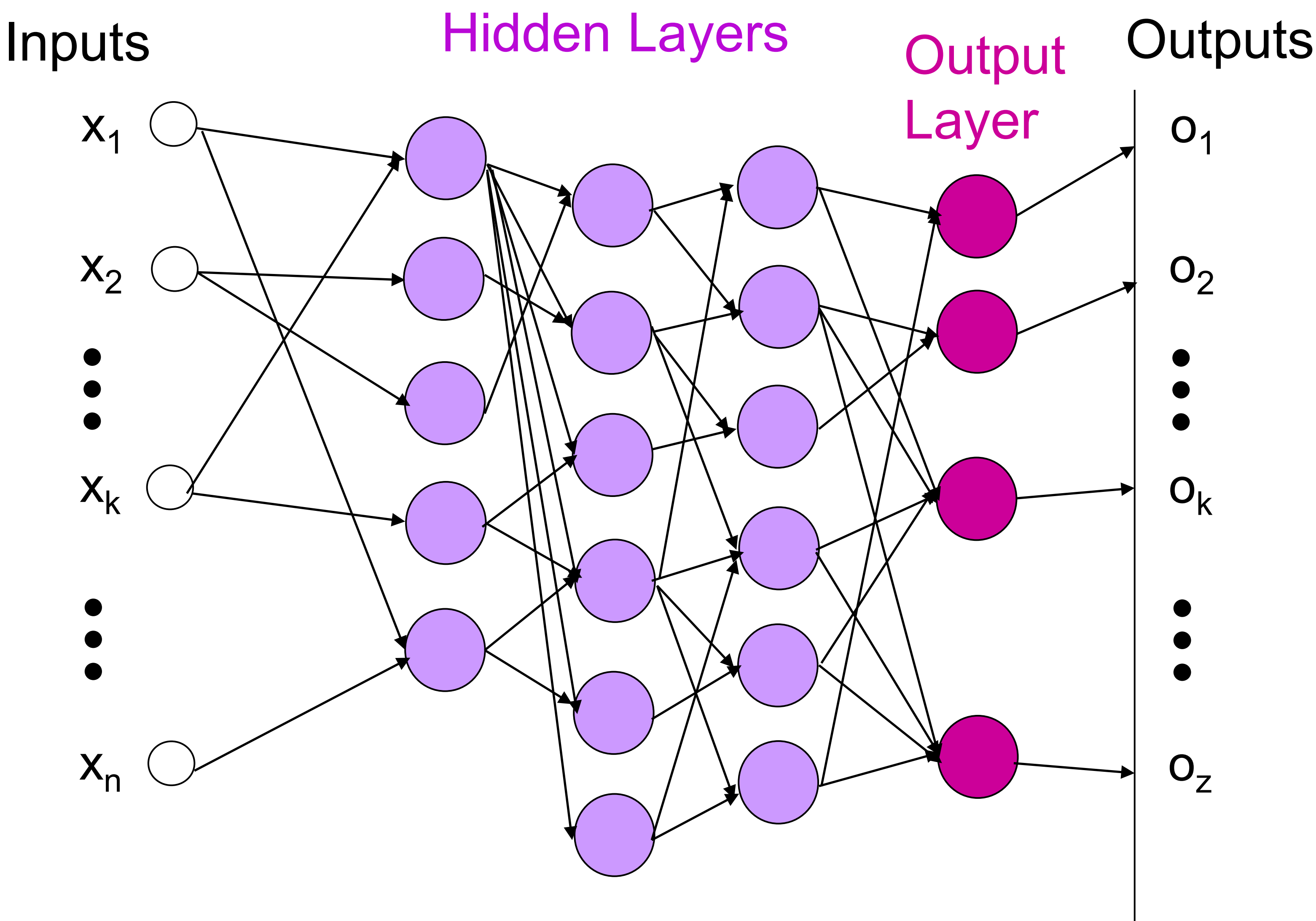
if $f(S) >= T$

$o_k >= 0$

2-Layer Neural Networks



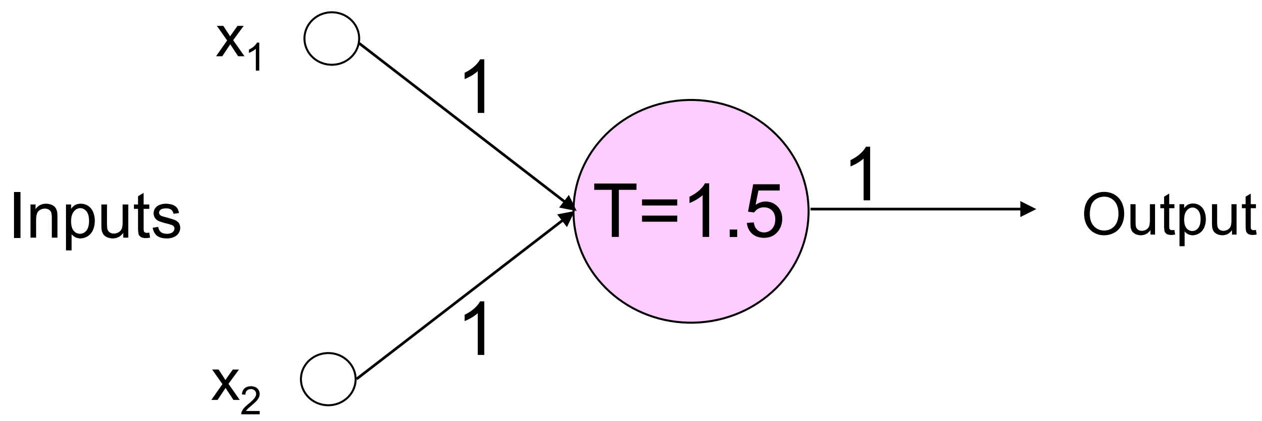
Multilayer Neural Networks



Tasks for Neural Networks

- Evaluation Problem
- Training Problem

Single Node Neural Net



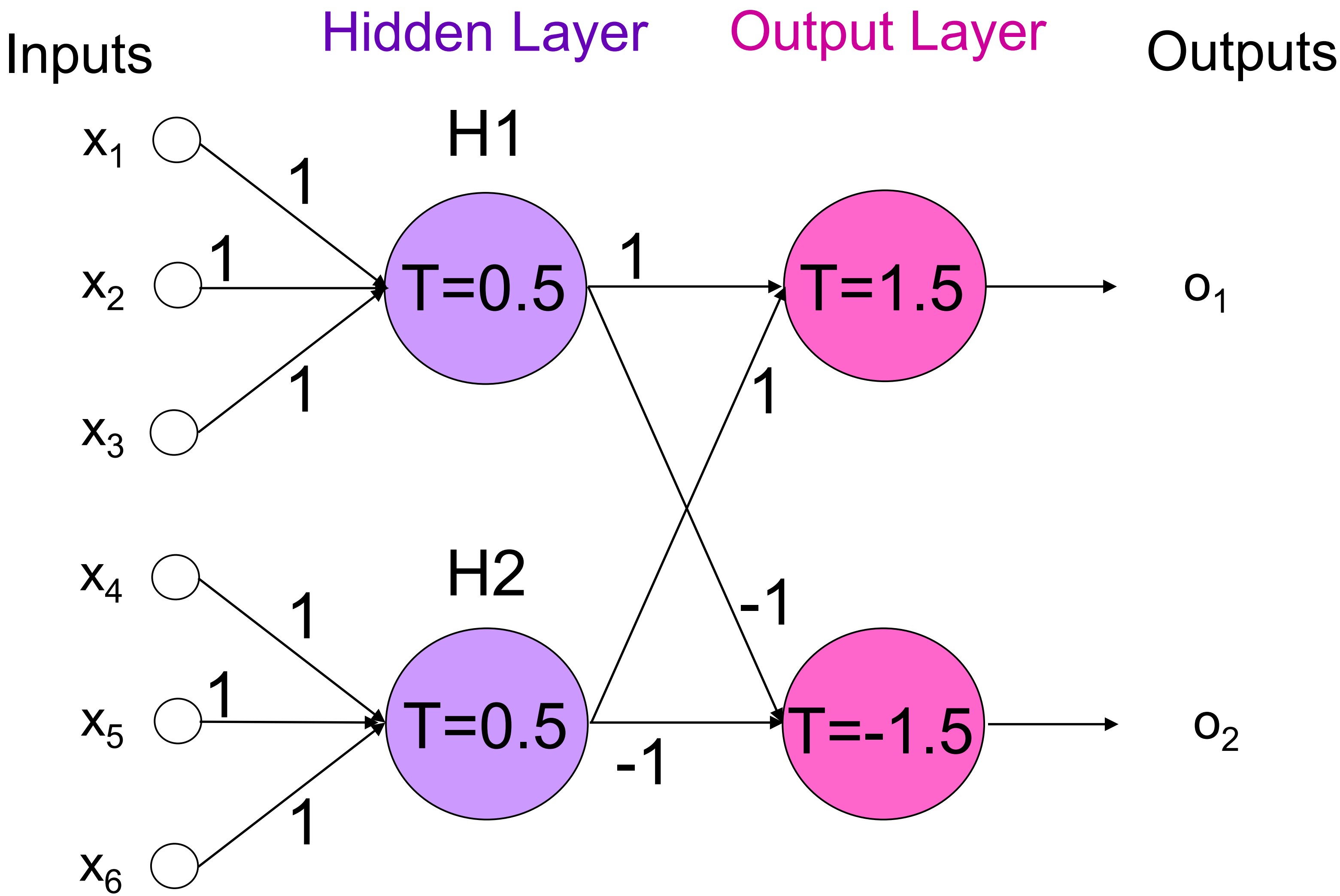
x_1	x_2	Computation	Output
0	0	0(1)+0(1)=0<1.5	0
0	1	0(1)+1(1)=1<1.5	0
1	0	1(1)+0(1)=1<1.5	0
1	1	1(1)+1(1)=2>1.5	1

How can we create a NN to recognize $Or(x_1,x_2)$?

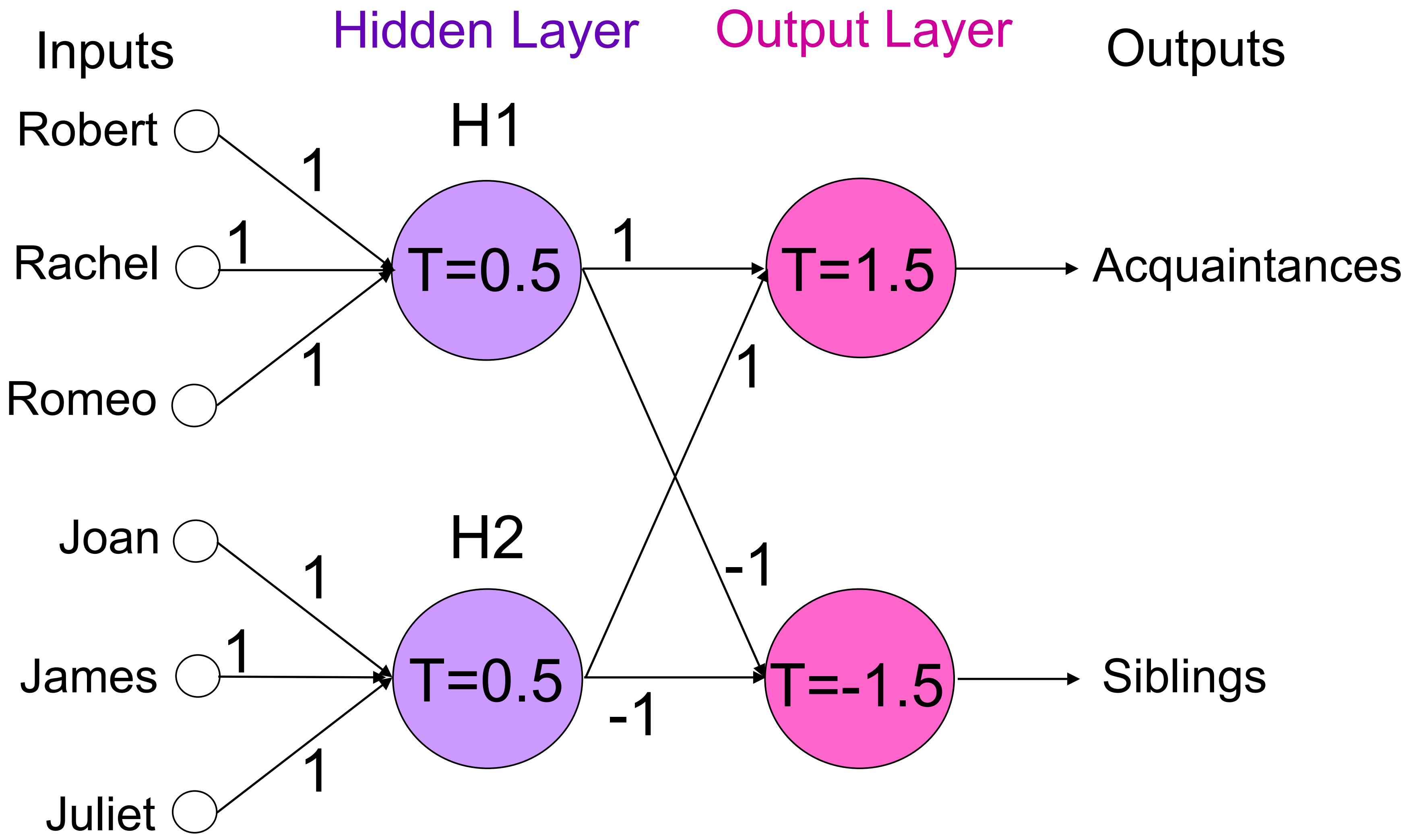
x_1	x_2	Computation	Output
0	0	$0(1)+0(1)=0 < T$	0
0	1	$0(1)+1(1)=1 > T$	1
1	0	$1(1)+0(1)=1 > T$	1
1	1	$1(1)+1(1)=2 > T$	1

T ?

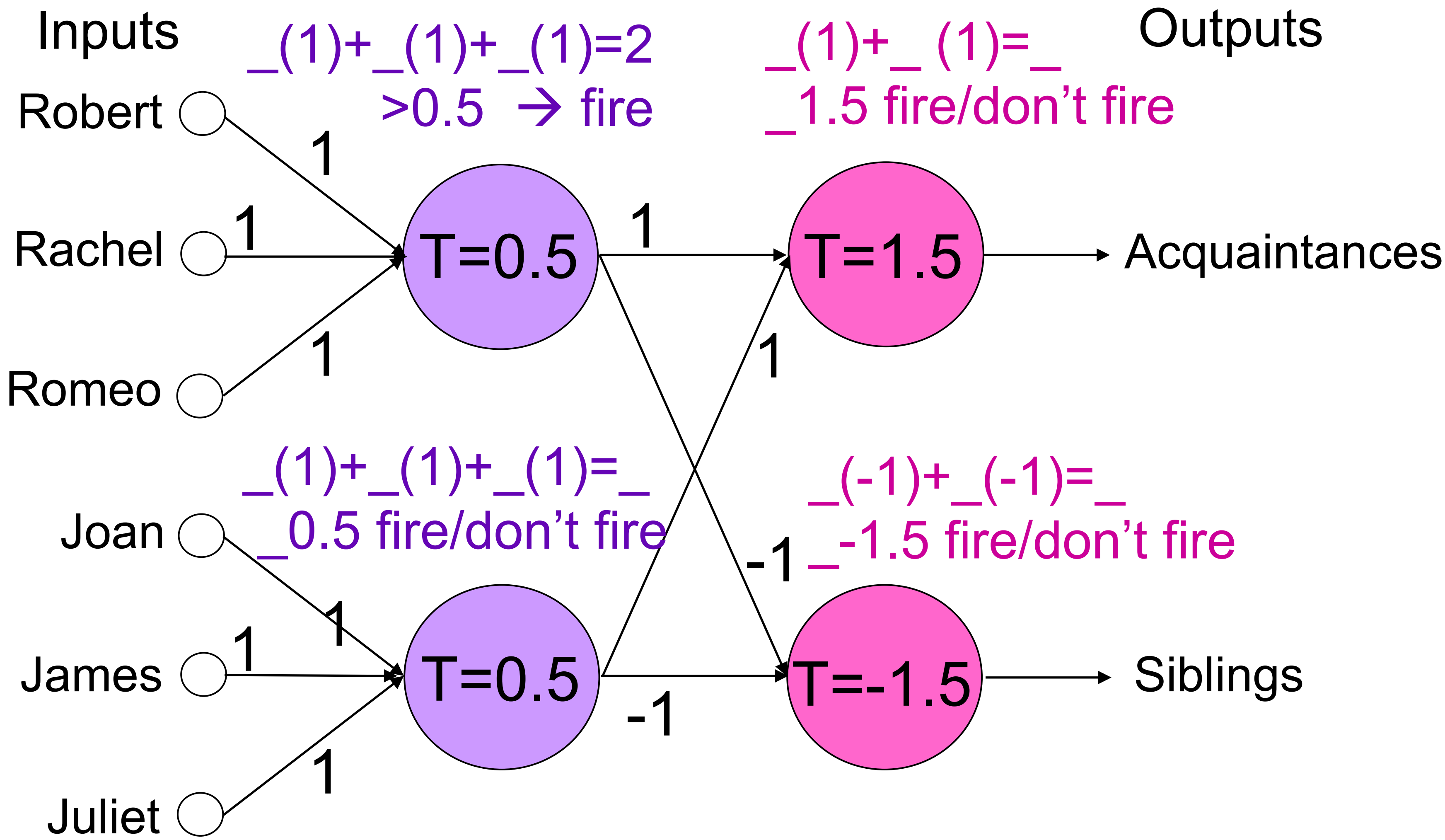
Example of a 3-layer Net



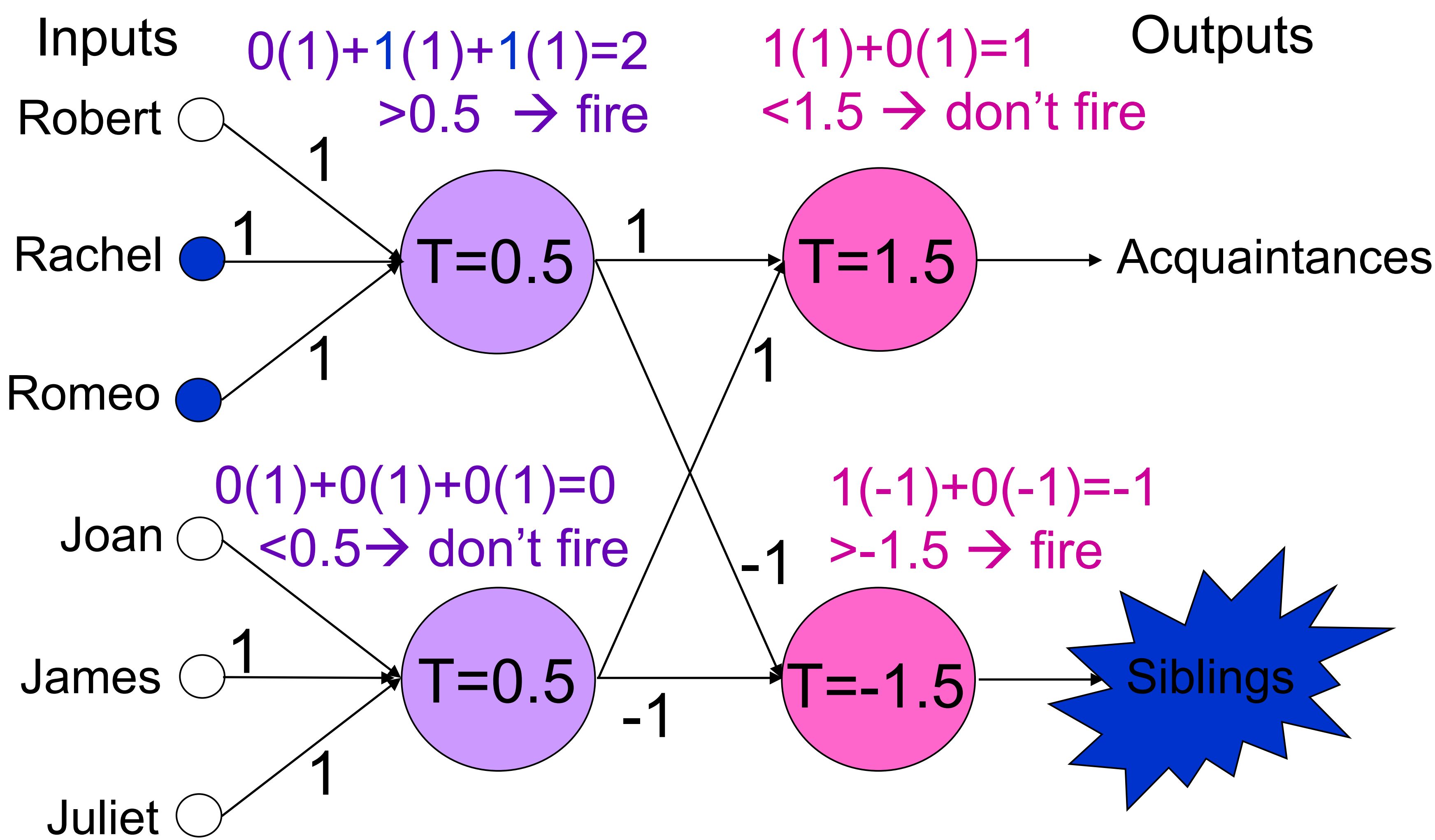
Acquaintance or Sibling Net



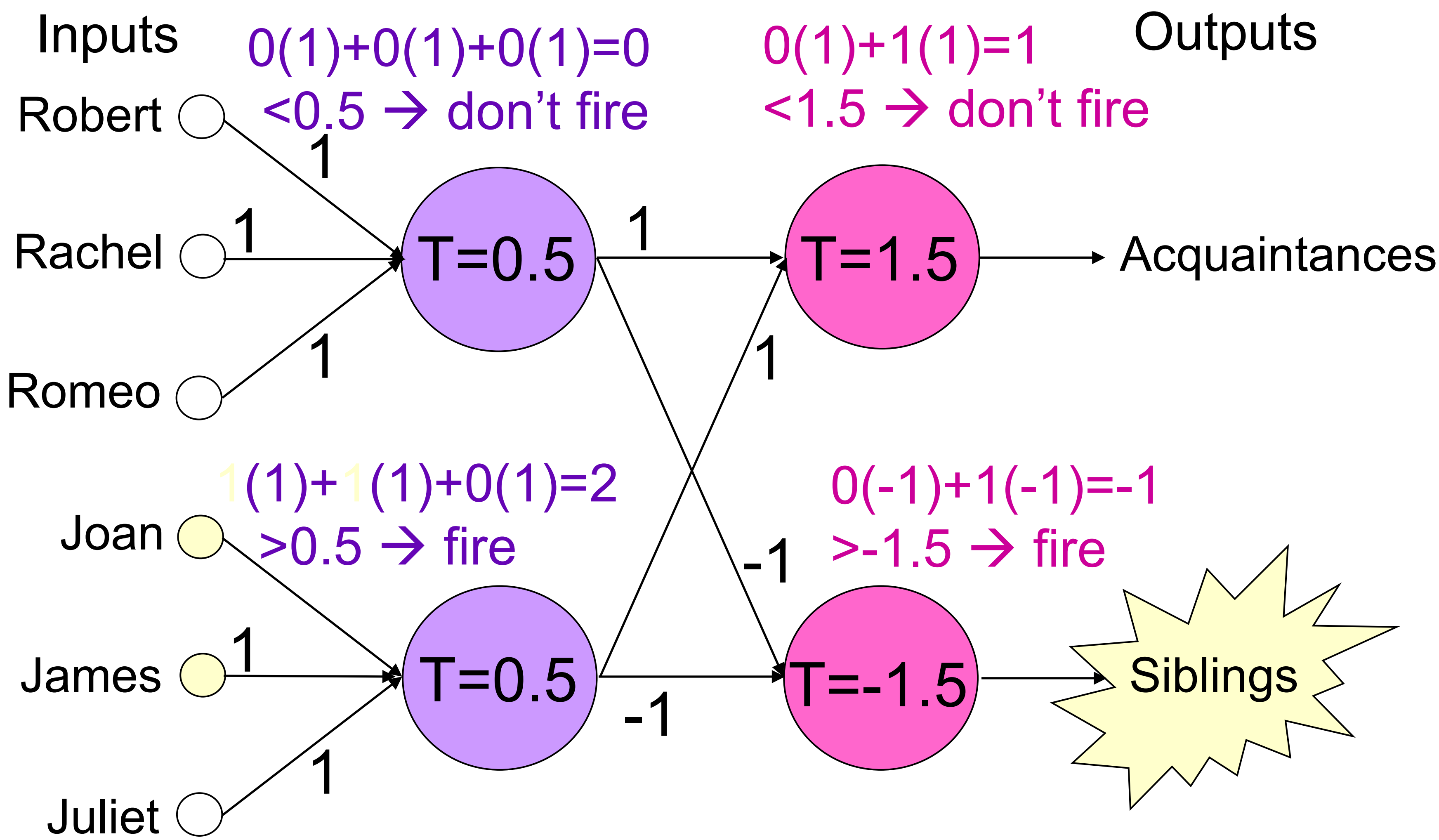
Acquaintances or Siblings?



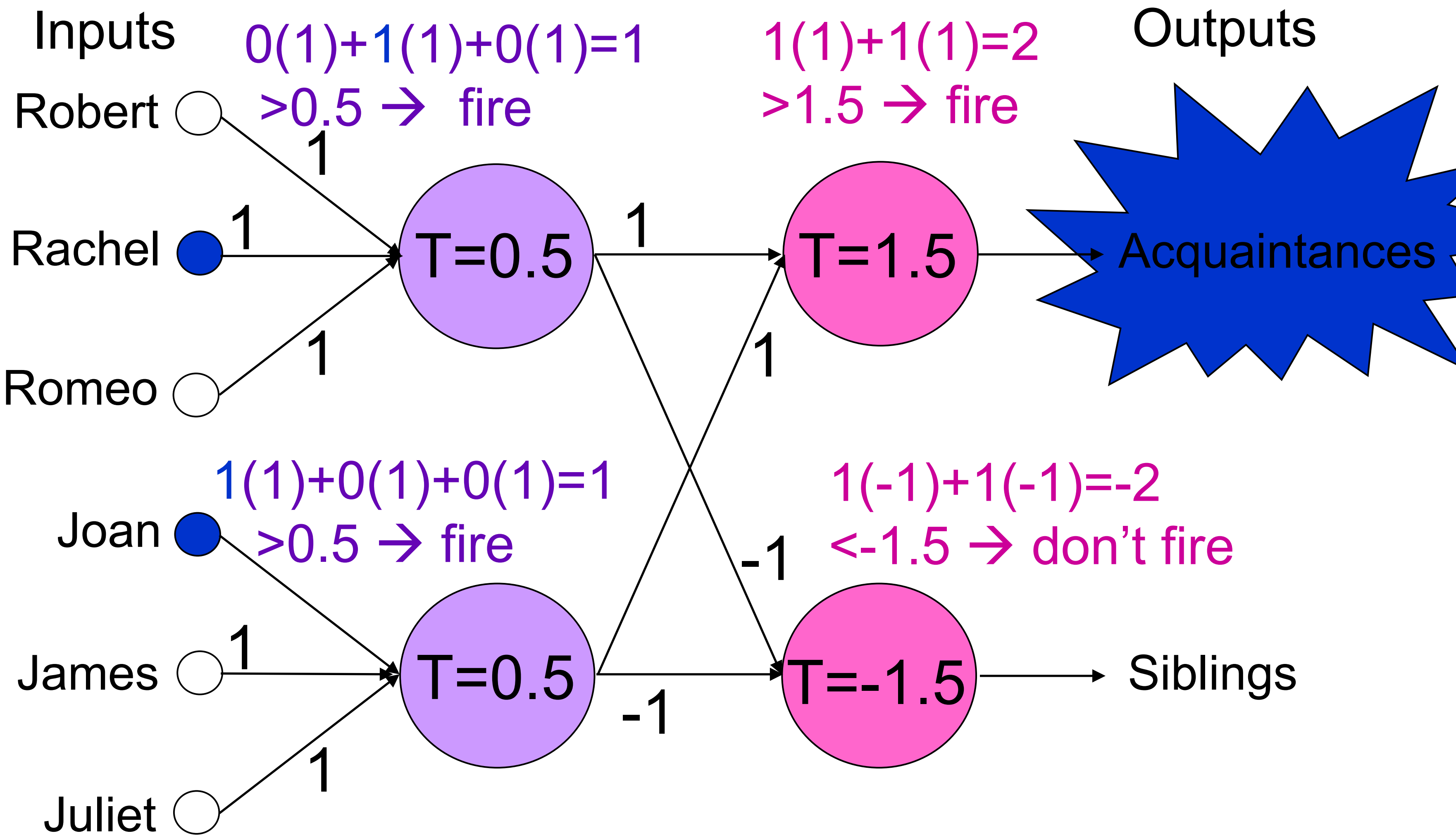
Acquaintances or Siblings?



Acquaintances or Siblings?



Acquaintances or Siblings?



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 $X_{training}$ of input vectors:

  Find neural net N that recognizes inputs in

  training set $X_{training}$ and

  test set X_{test}.

Training Problem – Version 1

- Training set $X_{training} = (X_{positive}, X_{negative})$ 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

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

for all inputs i in $X_{positive}$ and j in $X_{negative}$

Training Problem – Version 2

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

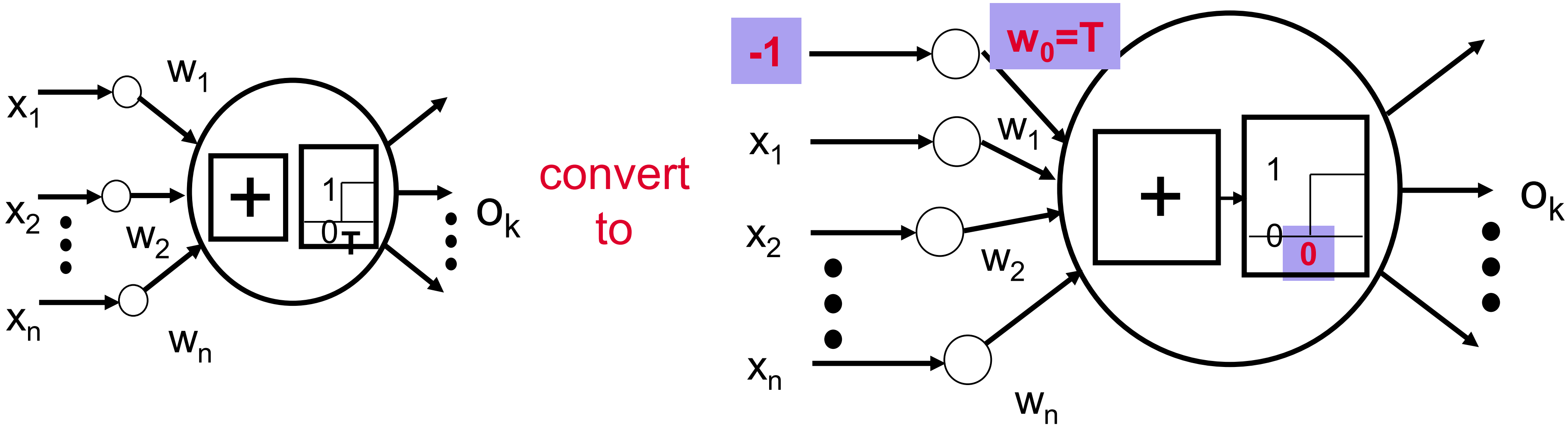
- Number of nodes
- Number of layers
- Shape of each activation function and its threshold
- Links between nodes
- Weights on connections
- Representative set $X_{training}$

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

where $X_{positive,\,i}$ and $X_{negative,\,j}$ in X_{test}

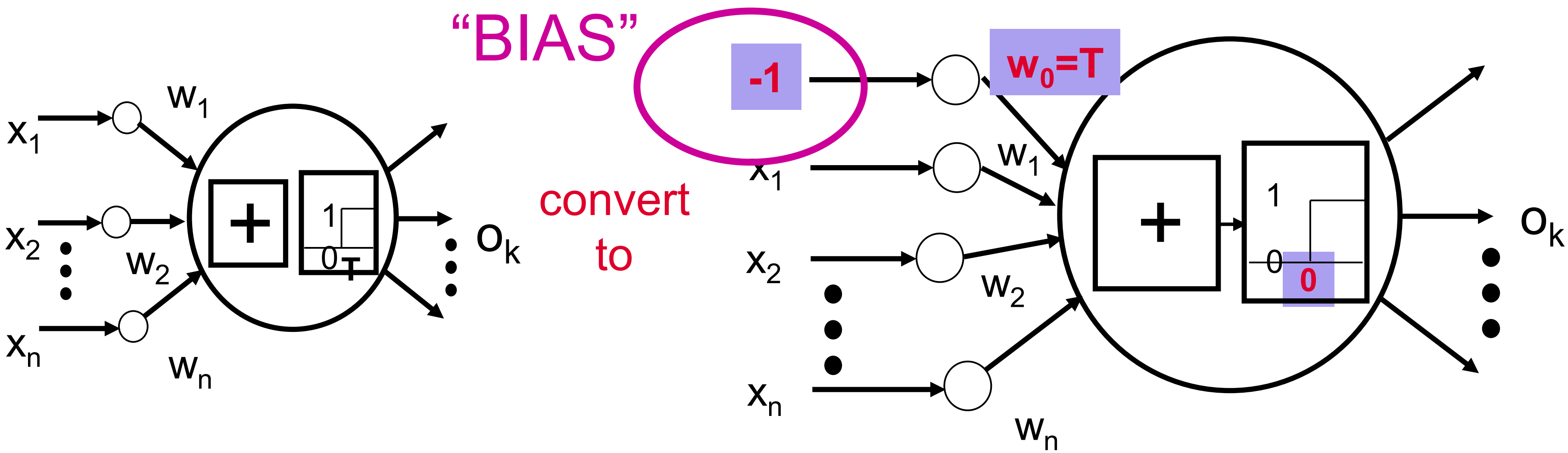
Training Problem – Version 1

- 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:



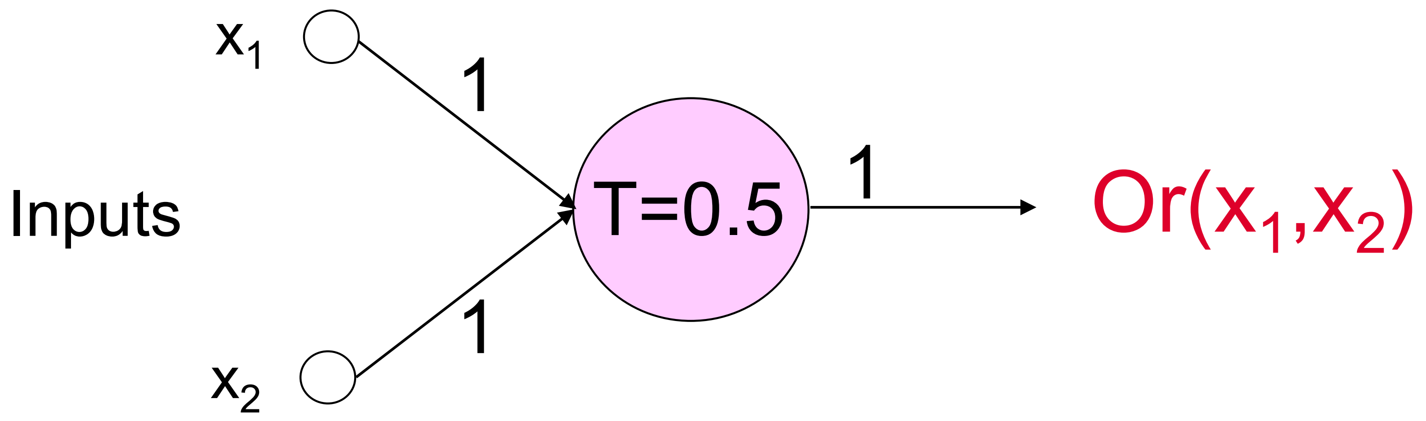
Training Problem – Version 1

- 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:



Single Node Neural Net

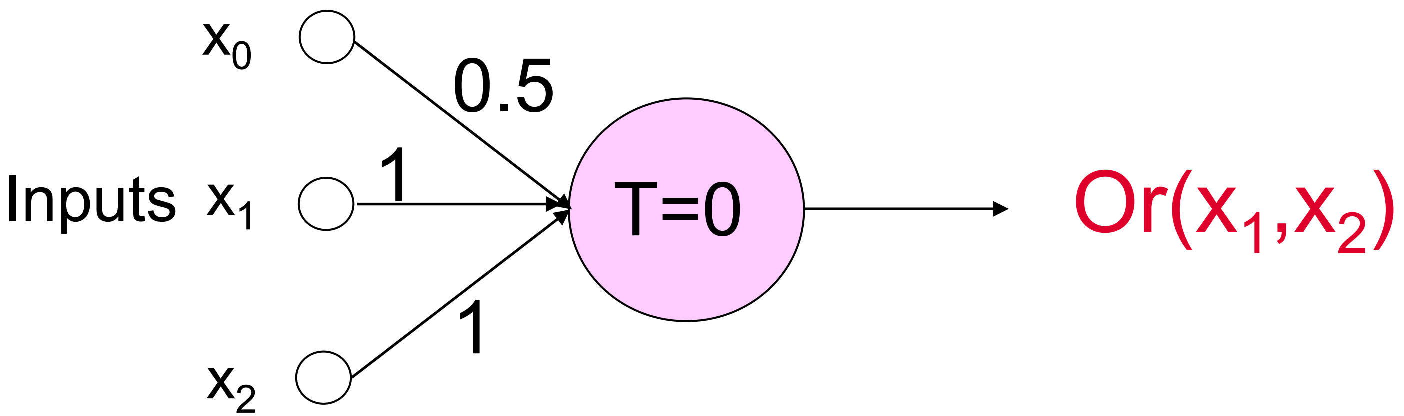
x_1	x_2	Computation	Output
0	0	$0(1)+0(1)=0 < 0.5$	0
0	1	$0(1)+1(1)=1 > 0.5$	1
1	0	$1(1)+0(1)=1 > 0.5$	1
1	1	$1(1)+1(1)=2 > 0.5$	1



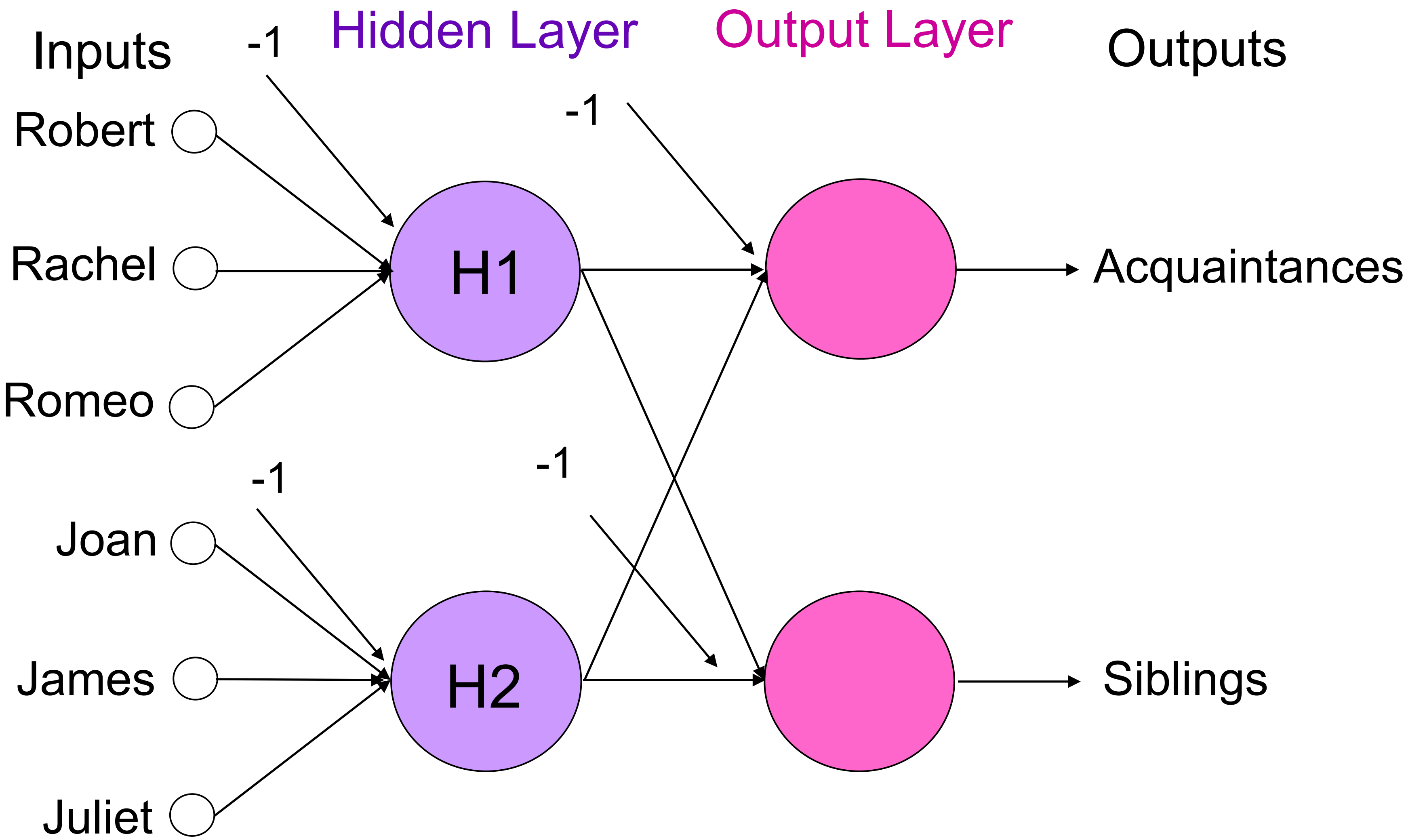
Example:
Converting Threshold to Weight

"BIAS"

x_0	x_1	x_2	Computation	Output
-1	0	0	-1(.5)+ 0(1)+0(1)=-0.5 < 0	0
-1	0	1	-1 (.5)+0(1)+1(1)=0.5 > 0	1
-1	1	0	-1 (.5)+1(1)+0(1)=0.5 > 0	1
-1	1	1	-1 (.5)+1(1)+1(1)=1.5> 0	1



Acquaintance or Sibling Net with converted thresholds



Training Problem – Version 1

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, $X_{training}$, $X_{validation}$, and $X_{testing}$:

  Find weights of neural net

Solution: Backpropagation procedure

- Version 2:

  Given dataset X:

  1. Find neural network architecture
  2. Design training protocol with $X_{training}$, $X_{validation}$, and $X_{testing}$
  3. Run Backpropagation procedure

Training Neural Networks

- Version 1: Easier Problem!

  Given a neural net N, $X_{training}$, $X_{validation}$, and $X_{testing}$:

  Find weights of neural net

Solution: Backpropagation procedure

- Version 2:

  Given dataset X:

  1. Find neural network architecture
  2. Design training protocol with $X_{training}$, $X_{validation}$, and $X_{testing}$
  3. Run Backpropagation procedure

Training Neural Networks

- Version 1:

  Given a neural net N, $X_{training}$, $X_{validation}$, and $X_{testing}$:

  Find weights of neural net

Solution: Backpropagation procedure

- Version 2:

  Given dataset X:

  Toolbox or Research!

  1. Find neural network architecture
  2. Design training protocol with $X_{training}$, $X_{validation}$, and $X_{testing}$
  3. Run Backpropagation procedure

Training Neural Networks

- Version 1:

  Given a neural net N, $X_{training}$, $X_{validation}$, and $X_{testing}$:

  Find weights of neural net

Solution: Backpropagation procedure

NEXT TOPIC!!!

- Version 2:

  Given dataset X:

  1. Find neural network architecture
  2. Design training protocol with $X_{training}$, $X_{validation}$, and $X_{testing}$
  3. Run Backpropagation procedure

CAS CS 640
Artificial Intelligence

Learning by Training Neural Nets, Part 2

Adopted from P. Winston, Artificial Intelligence, 1992

After we’ve covered the Backprop Procedure

Training Problem – Version 1

Example:

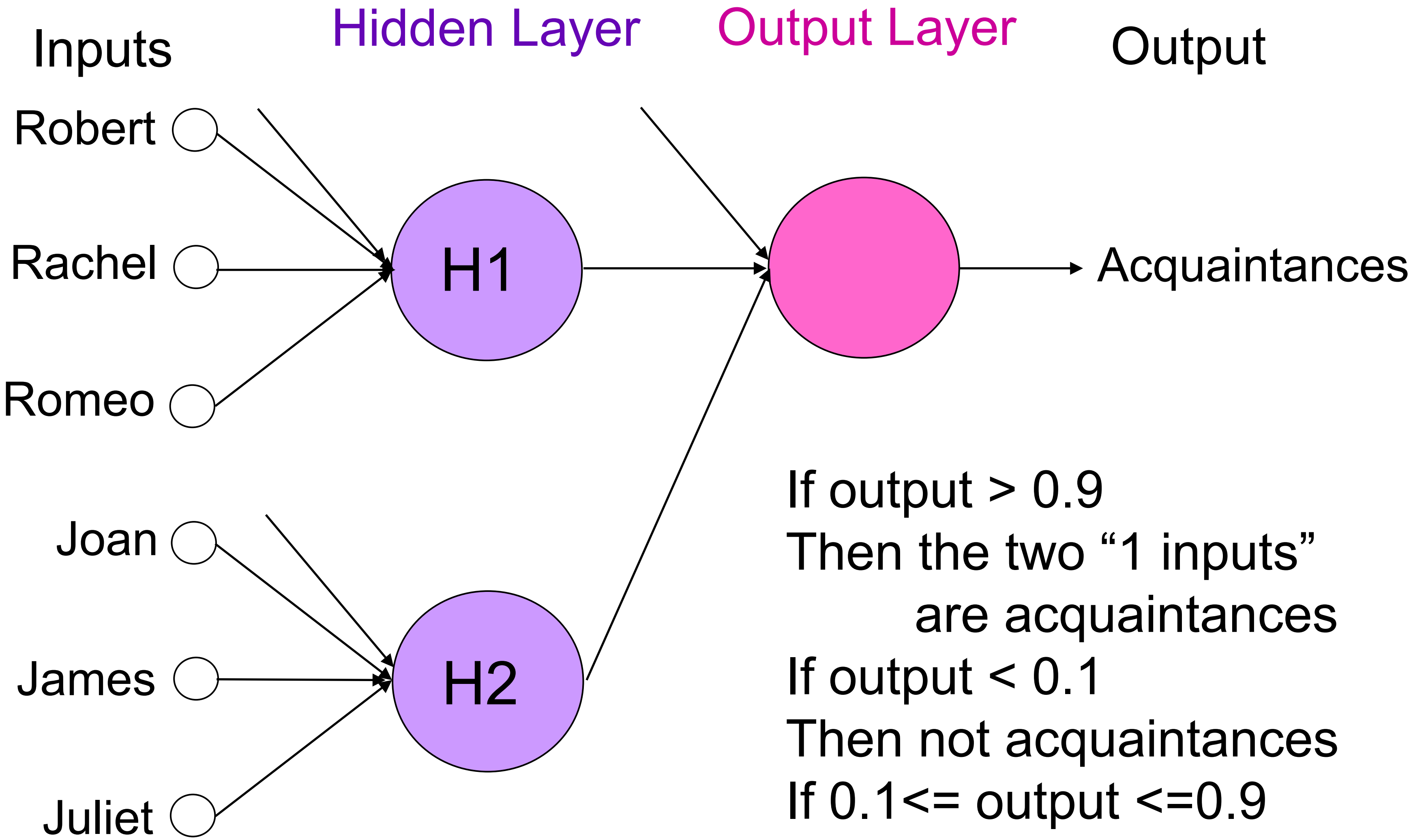
Acquaintance/Sibling Network

First:

Simplified Network:

Acquaintance Net

Acquaintance Net



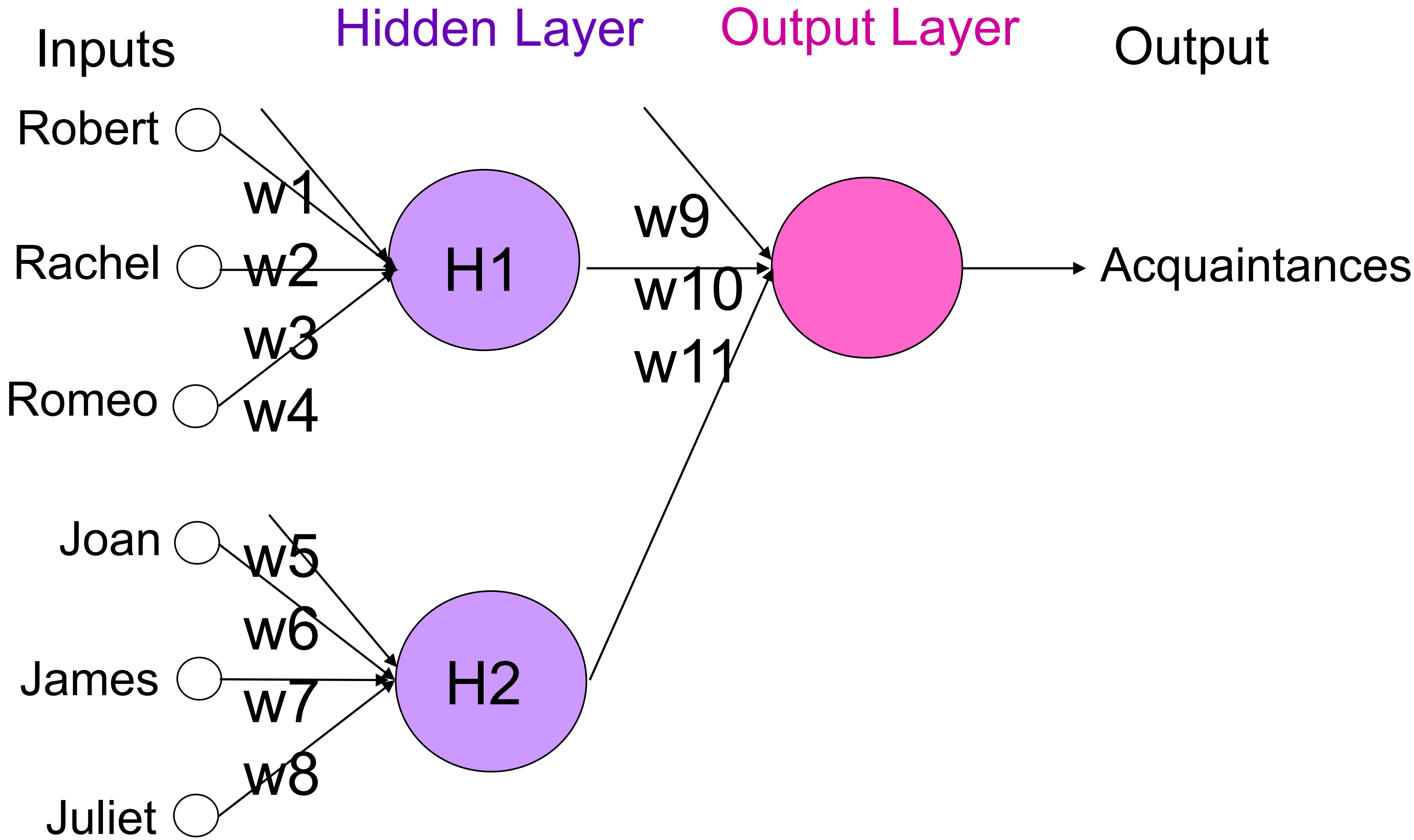
If output > 0.9
Then the two “1 inputs” are acquaintances
If output < 0.1
Then not acquaintances
If 0.1<= output <=0.9
Then ambiguous

Only two inputs may be 1

Labeled Training Data: 15

Robert	Raquel	Romeo	Joan	James	Juliet	Acquaintar
1	1	0	0	0	0	0
1	0	1	0	0	0	0
1	0	0	1	0	0	1
1	0	0	0	1	0	1
1	0	0	0	0	1	1
0	1	1	0	0	0	0
0	1	0	1	0	0	1
0	1	0	0	1	0	1
0	1	0	0	0	1	1
0	0	1	1	0	0	1
0	0	1	0	1	0	1
0	0	1	0	0	1	1
0	0	0	1	1	0	0
0	0	0	1	0	1	0
0	0	0	0	1	1	0

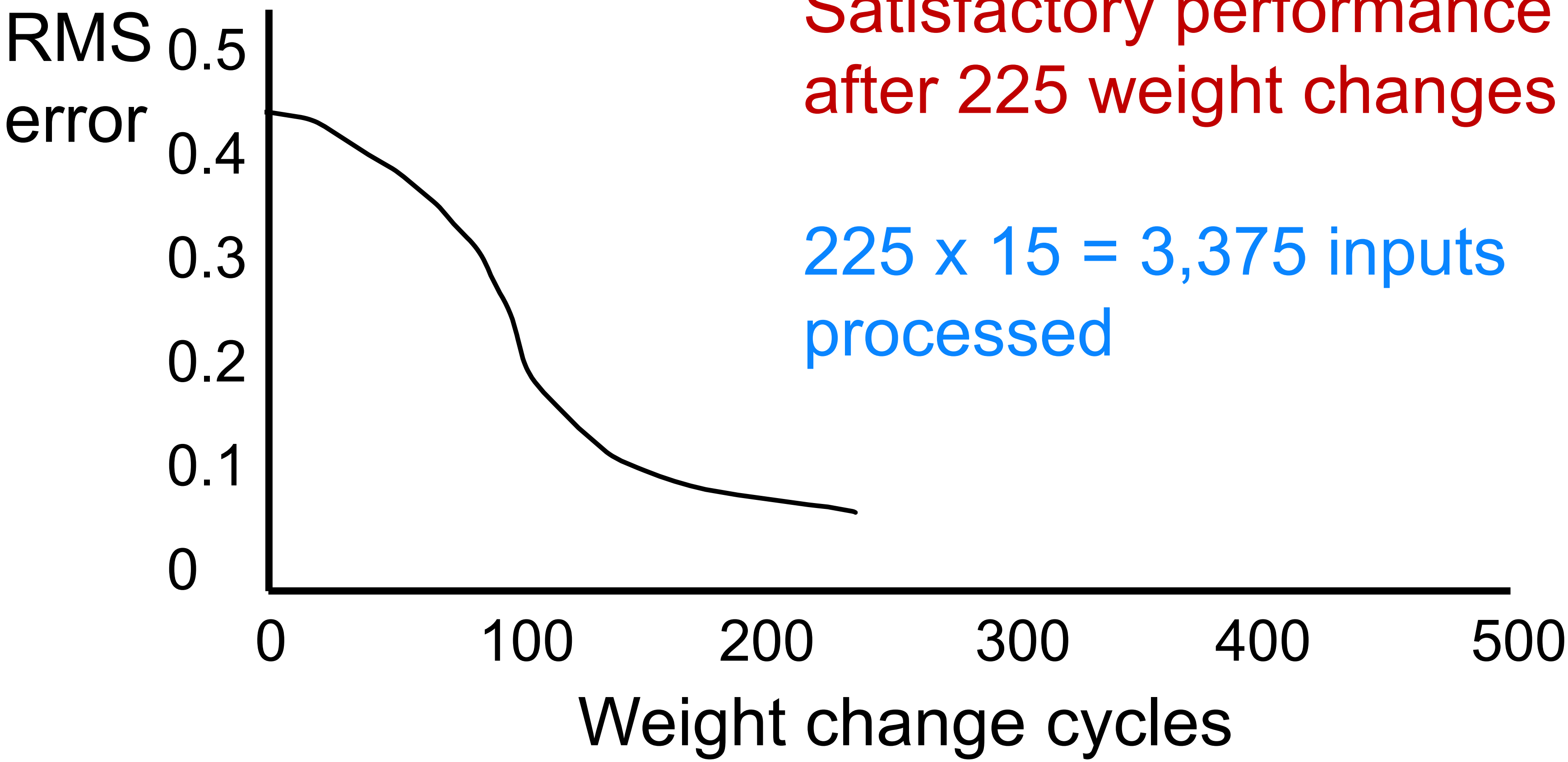
Acquaintance Net



Backprop for the Acquaintance Net Initial Values of 11 Weights:

Weight	Initial Value	Value after Backprop
w1	0.1	1.99
w2	0.2	4.65
w3	0.3	4.65
w4	0.4	4.65
w5	0.5	2.28
w6	0.6	5.28
w7	0.7	5.28
w8	0.8	5.28
w9	0.9	9.07
w10	1	6.27
w11	1.1	6.12

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

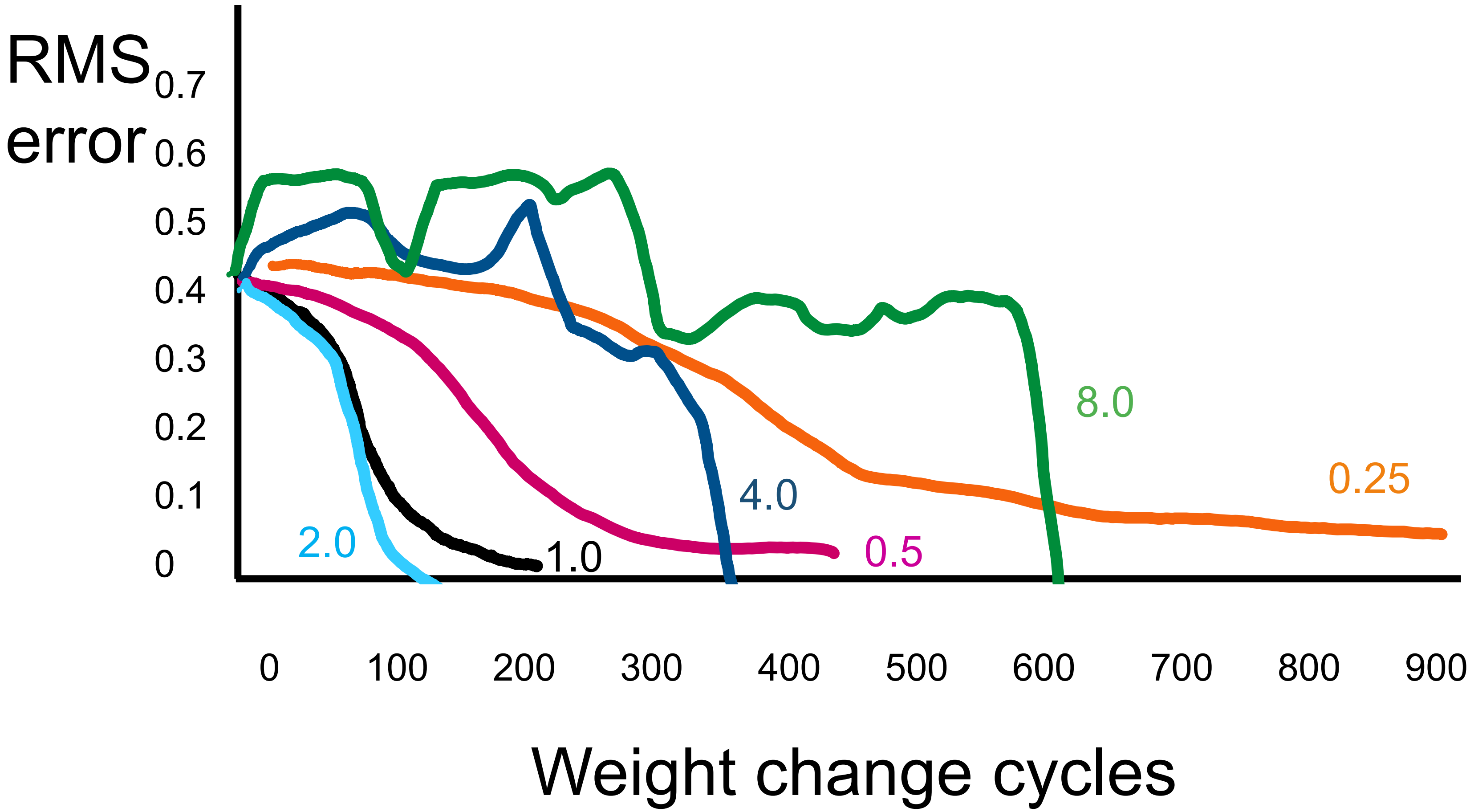
- Training may require thousands of backpropagations

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

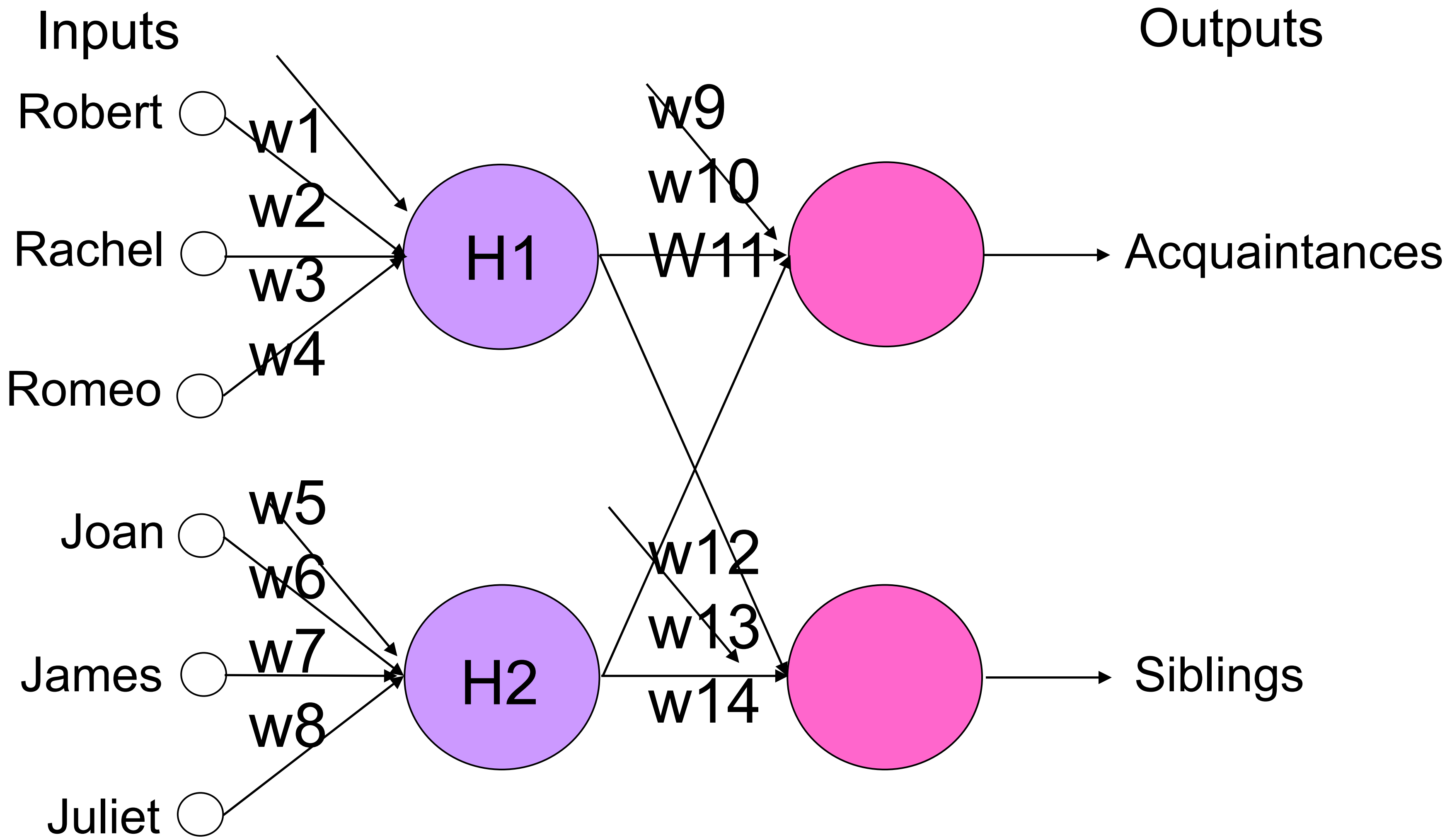
- Training may require thousands of backpropagations
- Training can get stuck or become unstable
- Training can be done in stages:

  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 extent the model by one output node

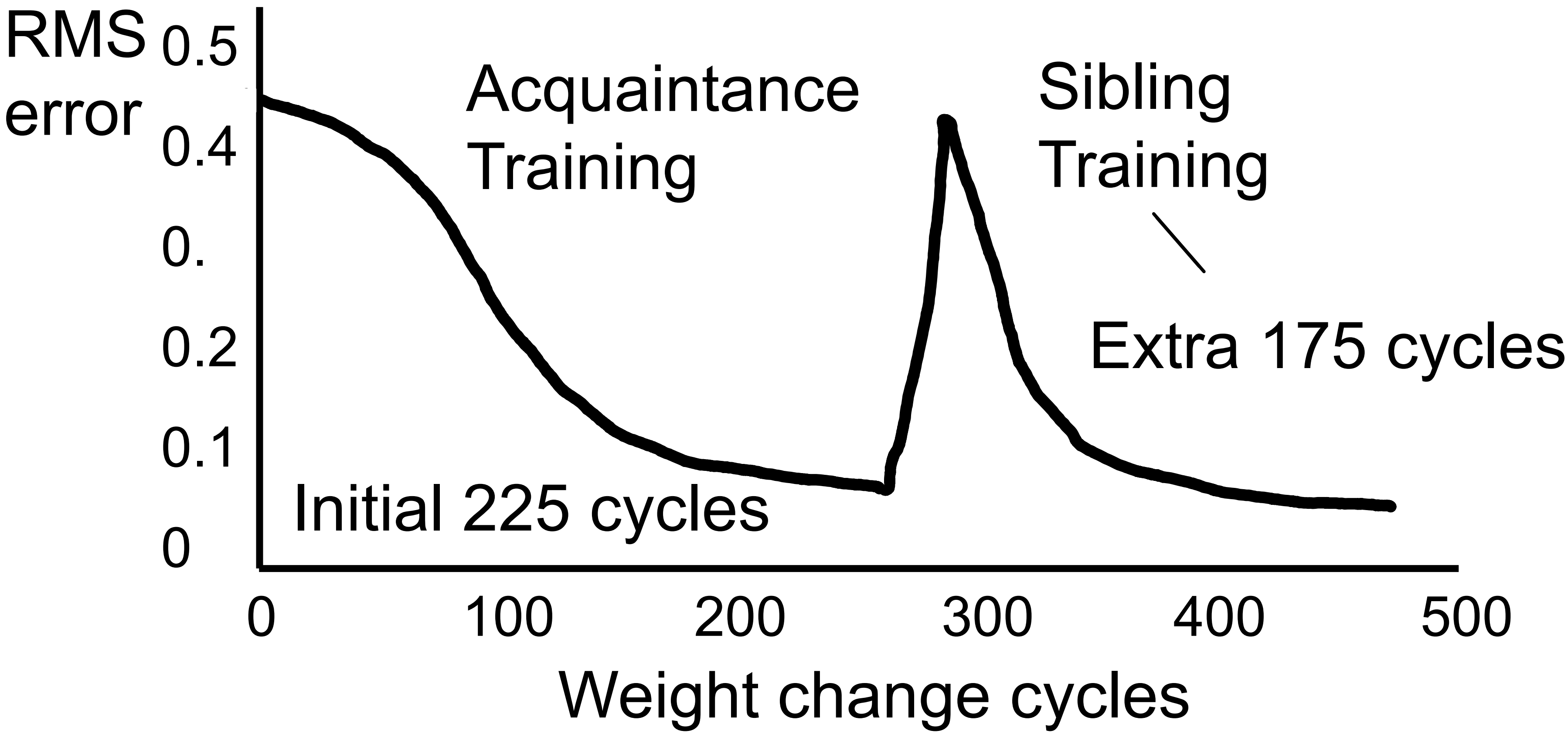
Acquaintance or Sibling Net



2-Stage Training of Ac/Sib Net

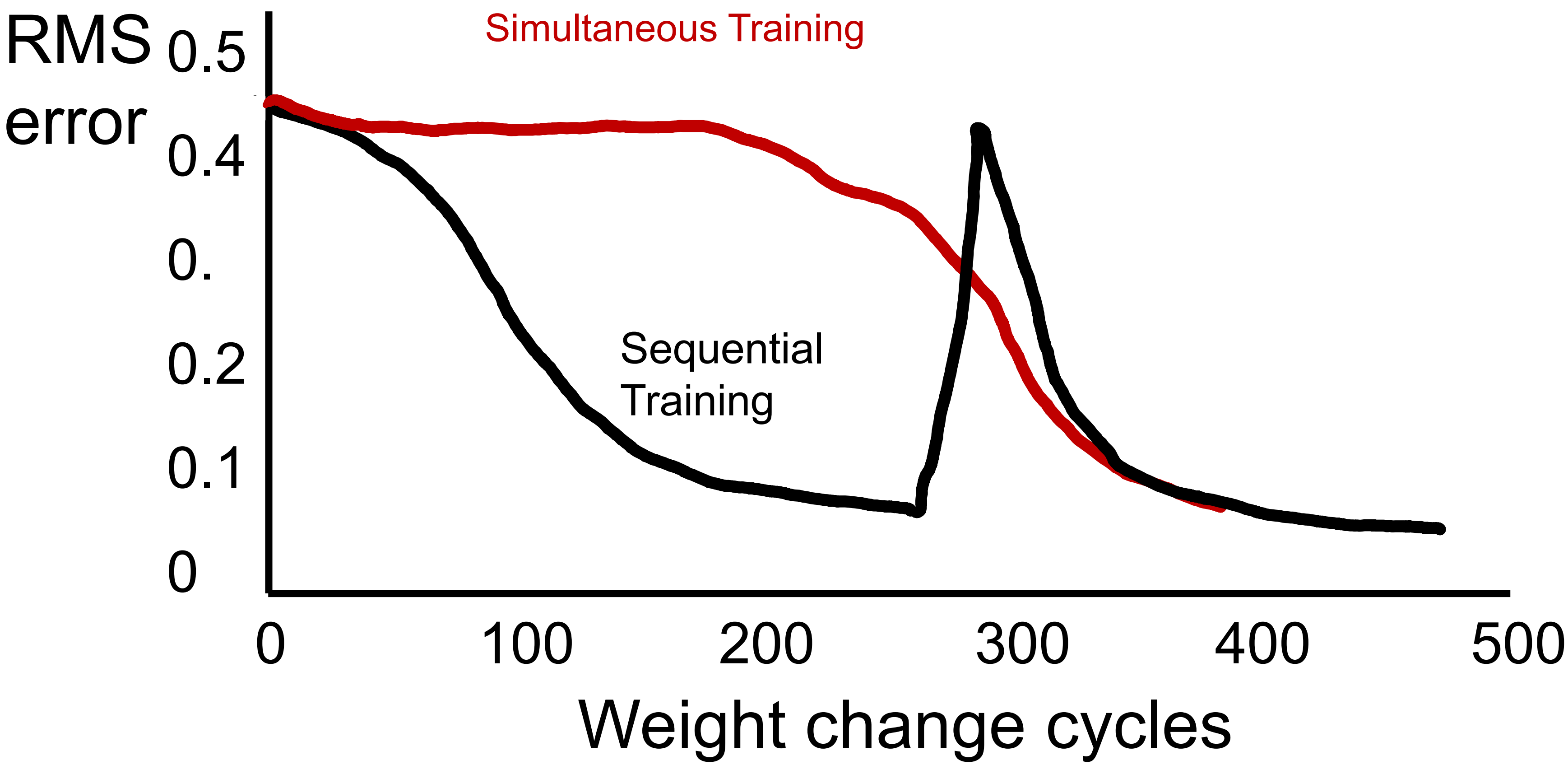
Weight	Initial Value	Value after Pretraining	Value after Sibling Training
w1	0.1	1.99	2.71
w2	0.2	4.65	6.02
w3	0.3	4.65	6.02
w4	0.4	4.65	6.02
w5	0.5	2.28	2.89
w6	0.6	5.28	6.37
w7	0.7	5.28	6.37
w8	0.8	5.28	6.37
w9	0.9	9.07	10.29
w10	1	6.27	7.04
w11	1.1	6.12	6.97
w12	1.2		-8.32
w13	1.3		-5.72
w14	1.4		-5.68

RMS Error during Two-State Training



400 cycles total

Simultaneous Training of 14 Weights of Full Acquaintance/Sibling Net



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

Robert	Raquel	Romeo	Joan	James	Juliet	Acquaintance	Sibling
1	1	0	0	0	0	0	1
1	0	1	0	0	0	0	1
1	0	0	1	0	0	1	0
1	0	0	0	1	0	1	0
1	0	0	0	0	1	1	0
0	1	1	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	0	0	1	0	1	0
0	1	0	0	0	1	1	0
0	0	1	1	0	0	1	0
0	0	1	0	1	0	1	0
0	0	1	0	0	1	1	0
0	0	0	1	1	0	0	1
0	0	0	1	0	1	0	1
0	0	0	0	1	1	0	1

We used all 15 samples for training! Nothing left for testing…

Robert	Raquel	Romeo	Joan	James	Juliet	Acquaintance	Sibling
1	1	0	0	0	0	0	1
1	0	1	0	0	0	0	1
1	0	0	1	0	0	1	0
1	0	0	0	1	0	1	0
1	0	0	0	0	1	1	0
0	1	1	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	0	0	1	0	1	0
0	1	0	0	0	1	1	0
0	0	1	1	0	0	1	0
0	0	1	0	1	0	1	0
0	0	1	0	0	1	1	0
0	0	0	1	1	0	0	1
0	0	0	1	0	1	0	1
0	0	0	0	1	1	0	1

Use 3 Samples for Testing, Train on the Remaining 12 Samples

Robert	Raquel	Romeo	Joan	James	Juliet	Acquaintance	Sibling
1	1	0	0	0	0	0	1
1	0	1	0	0	0	0	1
1	0	0	1	0	0	1	0
1	0	0	0	1	0	1	0
1	0	0	0	0	1	1	0
0	1	1	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	0	0	1	0	1	0
0	1	0	0	0	1	1	0
0	0	1	1	0	0	1	0
0	0	1	0	1	0	1	0
0	0	1	0	0	1	1	0
0	0	0	1	1	0	0	1
0	0	0	1	0	1	0	1
0	0	0	0	1	1	0	1

Testing Result:

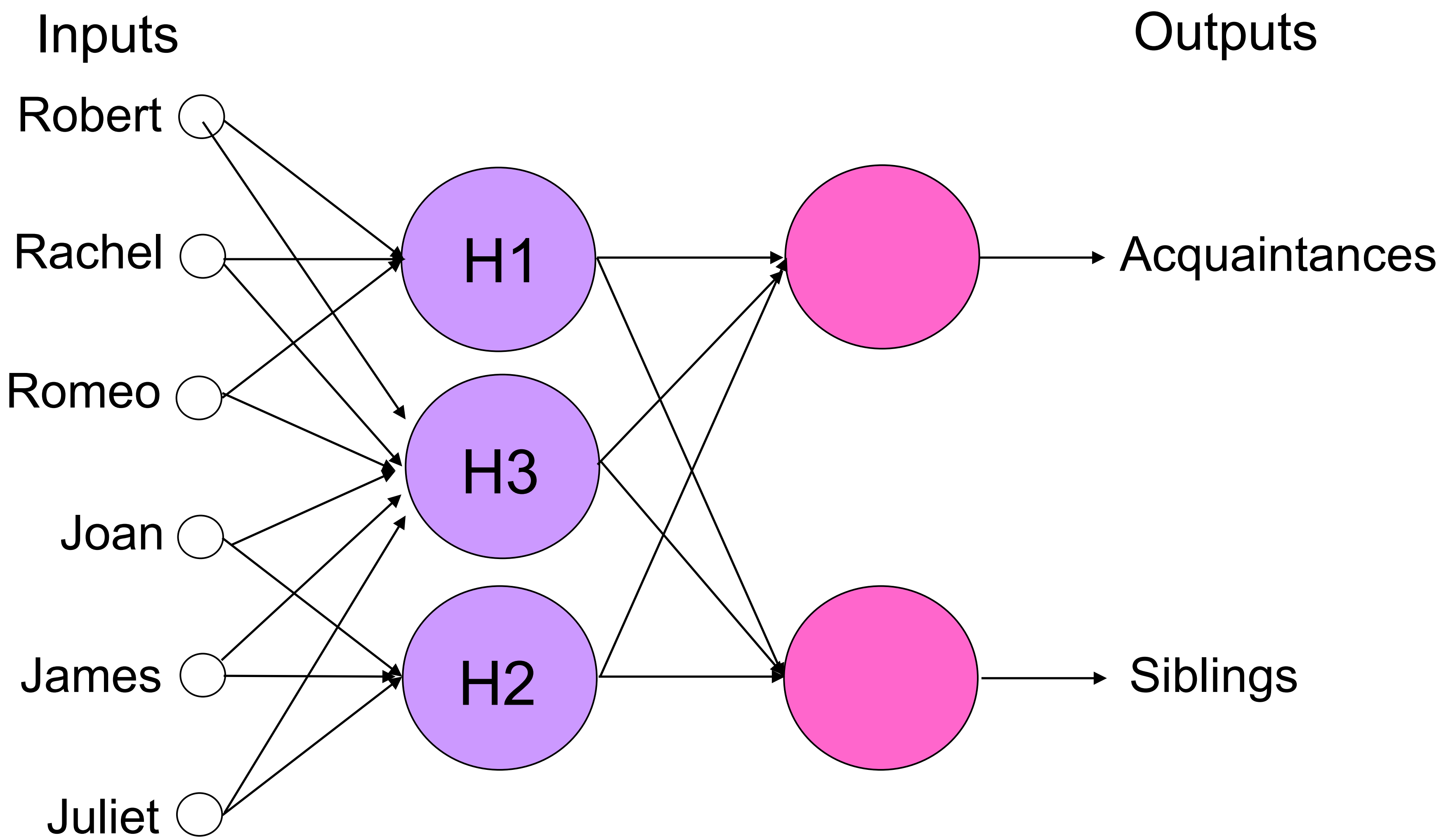
	Acquaintance		Sibling	
	Desired	Computed	Desired	Computed
Robert/Juliet	1	0.92	0	0.06
Romeo/Joan	1	0.92	0	0.06
James/Juliet	0	0.09	1	0.91

Interpretation:
Trained Acq/Sib net deals successfully with previously unseen data = it can predict!

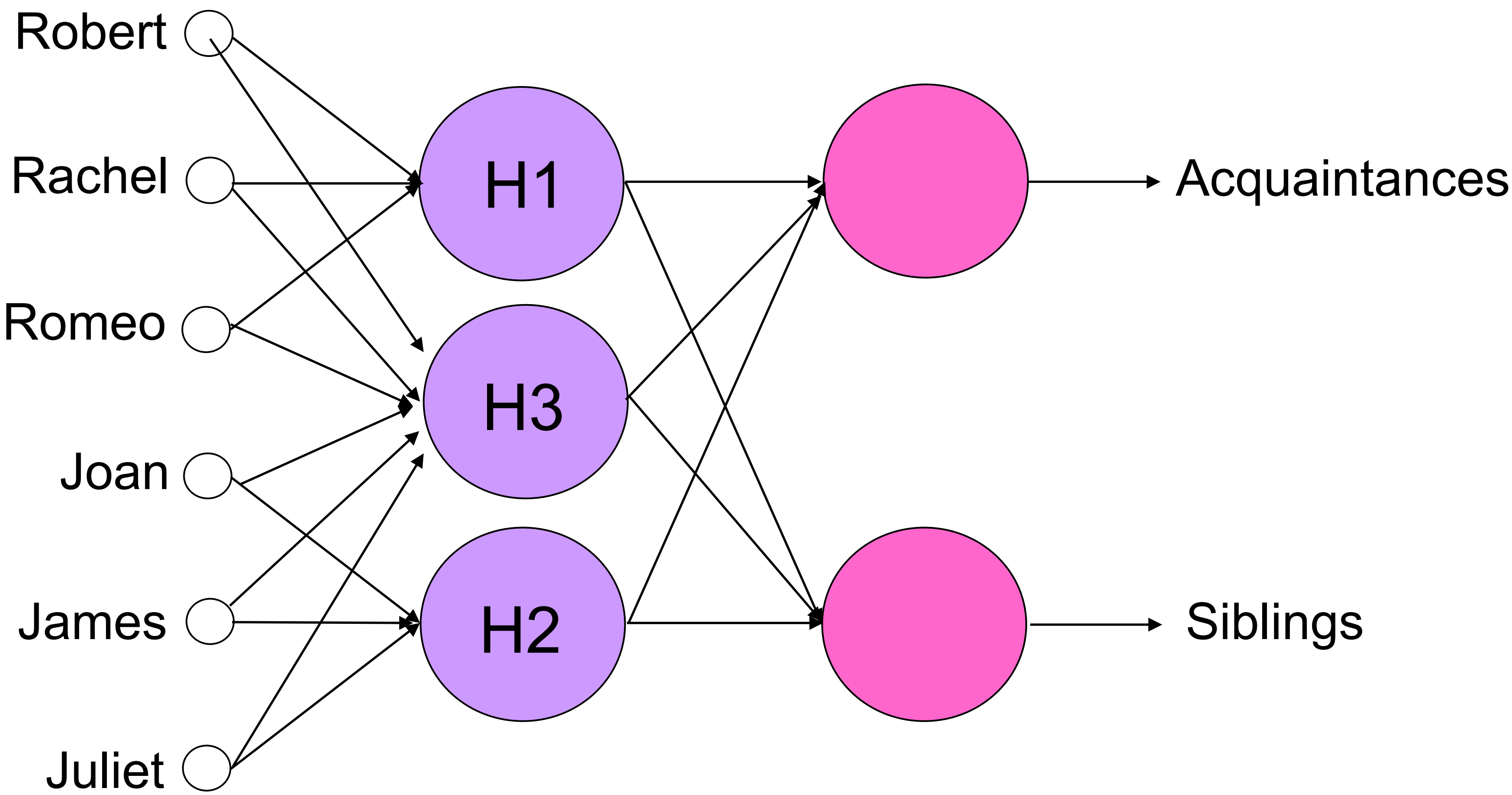
Characteristics of Back-Propagation

- 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?



Training only takes 300 cycles – the extra weights make it too easy to deal with the training set



Testing Result:

	Acquaintance		Sibling	
	Desired	Computed	Desired	Computed
Robert/Juliet	1	0.99	0	0
Romeo/Joan	1	0.06	0	0.94
James/Juliet	0	0.97	1	0.01

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."