Introduction to Parallel Distributed Processing models

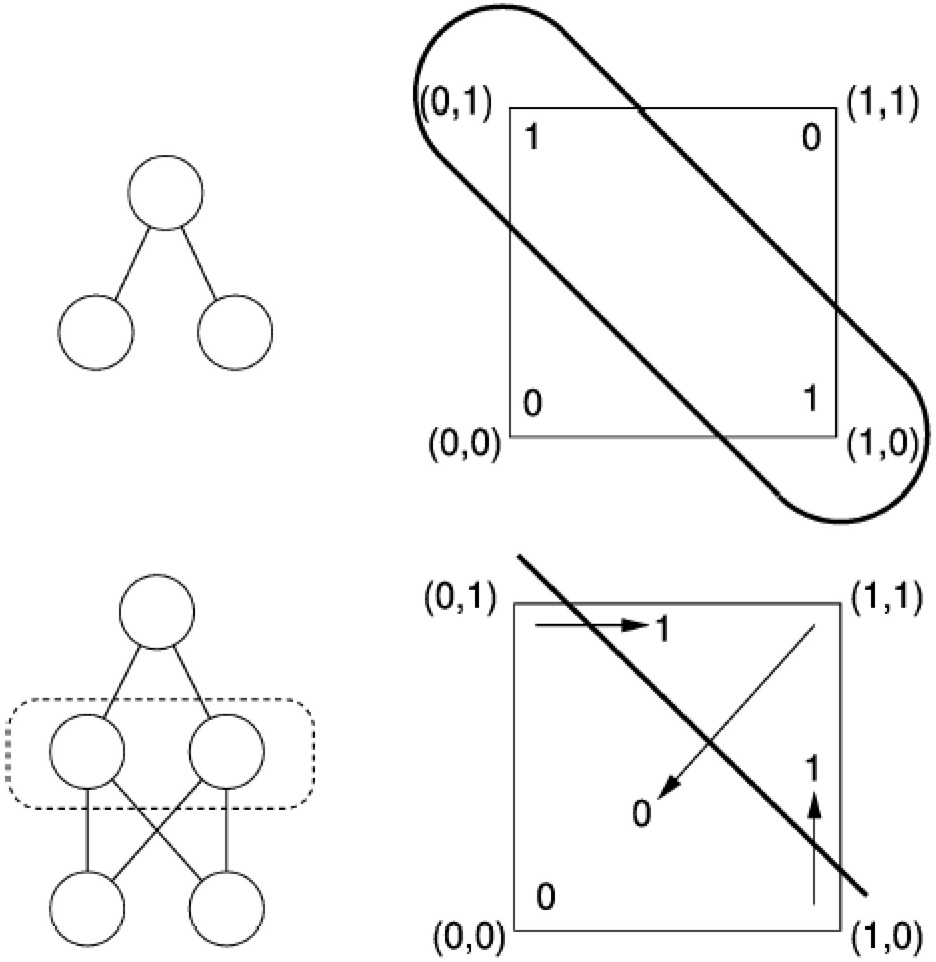
Deon T. Benton

004-Backpropagation

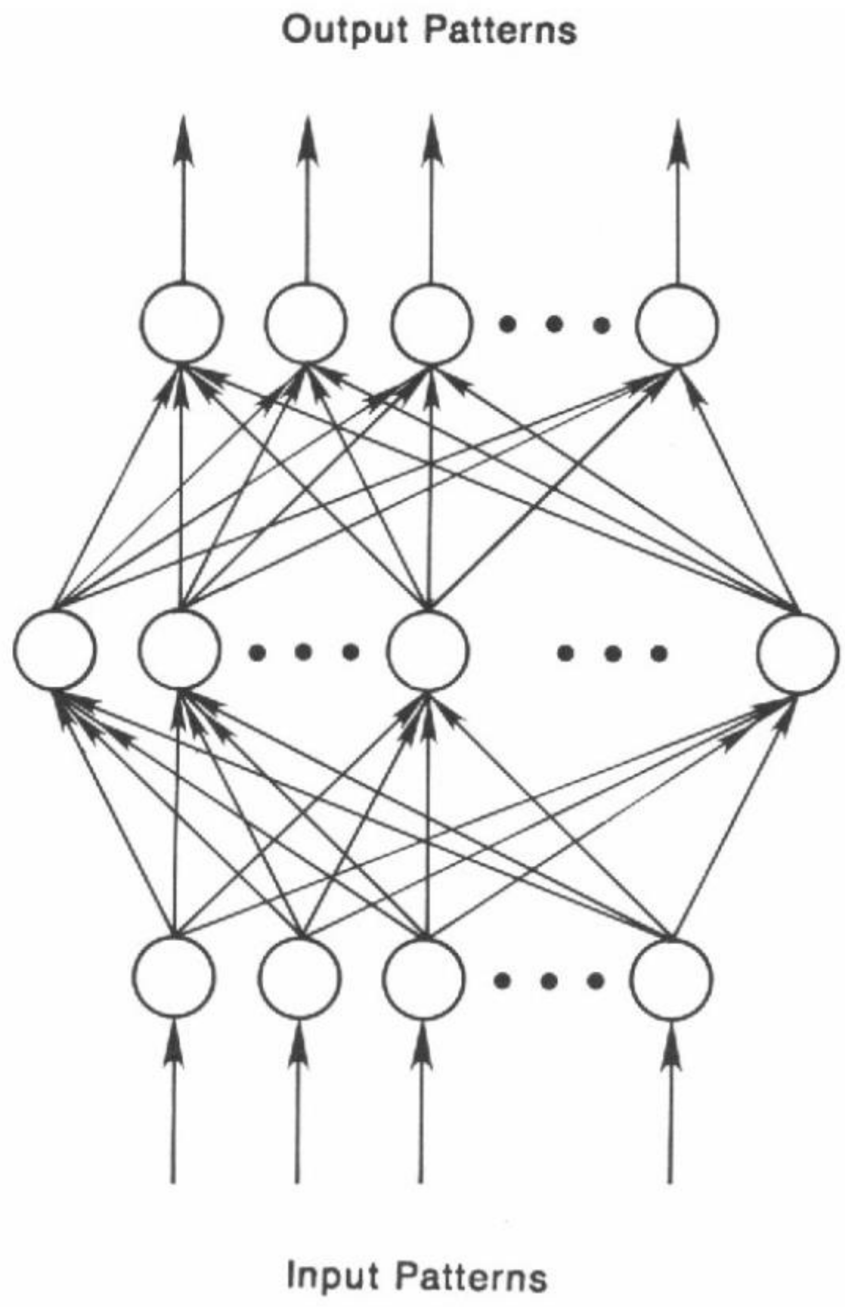
Pre-lecture quiz

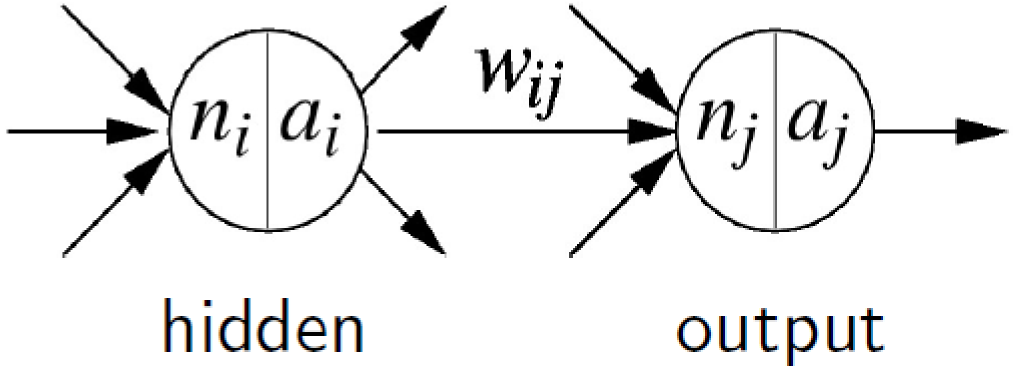
- The Delta rule is guaranteed to succeed under which conditions?
- True of False? Linear independence and orthogonality mean the same thing; that is, they are identical.
- Under what conditions are the Delta rule and the Hebb rule equivalent?

XOR with intermediate (“hidden”) units



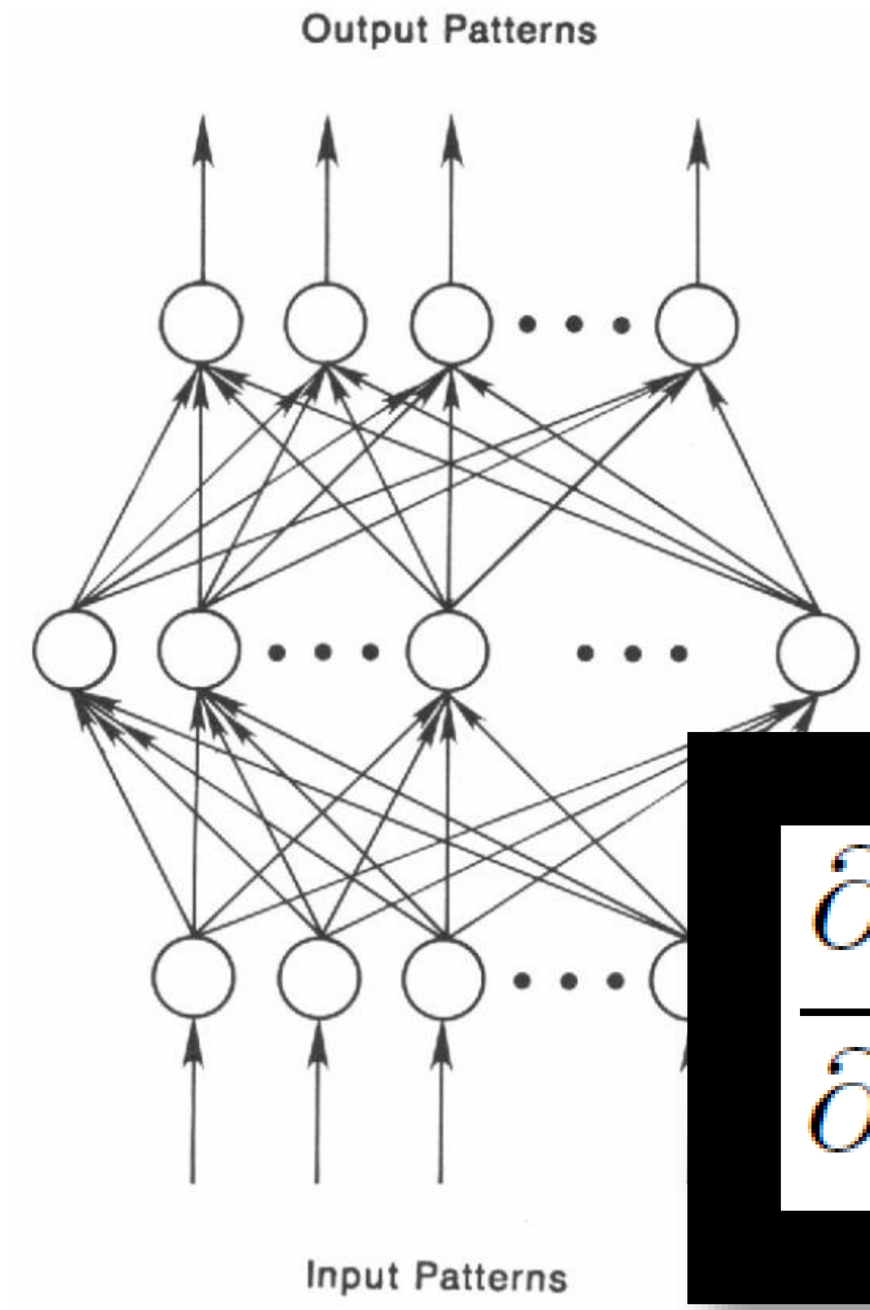
- Intermediate units can re-represent input patterns as new patterns with **altered similarities**
- Targets which are not linearly separable in the input space can be linearly separable in the intermediate representational space
- Intermediate units are called “hidden” because their activations are not determined directly by the training environment (inputs and targets)

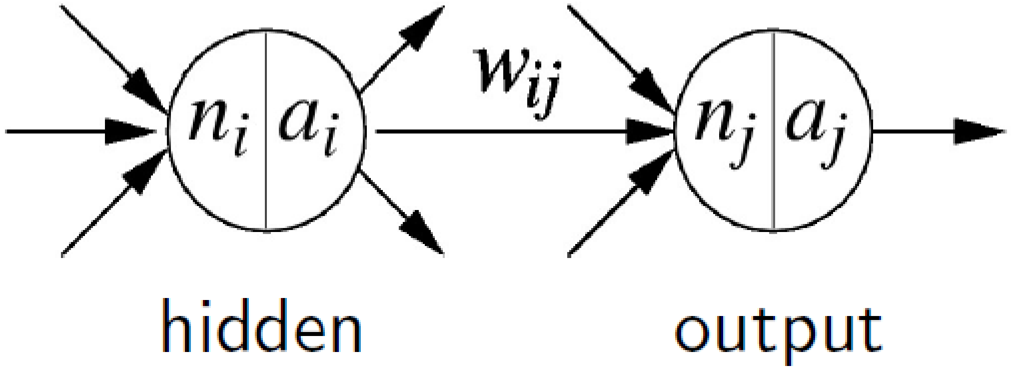




- Hidden-to-output weights can be trained with the Delta rule

Why??



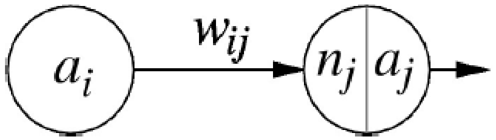


- Hidden-to-output weights can be trained with the Delta rule

$$\frac{\partial E}{\partial a_i}$$

- … can we train input-to-hidden weights?
  - Hidden units do not have targets (for determining error)
  - **Trick:** We don't need targets, we just need to know how hidden activations affect error (i.e., error derivatives)

Delta rule as gradient descent in error (sigmoid units)



$$n_j = \sum_i a_i \, w_{ij}$$

$$a_j = \frac{1}{1 + \exp(-n_j)}$$

$$\text{Error} \quad E = \frac{1}{2} \sum_j (t_j - a_j)^2$$

$$\begin{array}{ccccccc} w_{ij} & \searrow & & & t_j & \searrow & \\ a_i & \rightarrow & n_j & \rightarrow & a_j & \rightarrow & E \end{array}$$

$$\text{Gradient descent:} \qquad \Delta w_{ij} = -\epsilon \frac{\partial E}{\partial w_{ij}}$$

$$\begin{aligned} \frac{\partial E}{\partial w_{ij}} &= \frac{\partial E}{\partial a_j} \qquad \frac{\mathrm{d} a_j}{\mathrm{d} n_j} \qquad \frac{\partial n_j}{\partial w_{ij}} \\ &= -(t_j - a_j) \quad a_j (1 - a_j) \quad a_i \end{aligned}$$

$$\Delta w_{ij} = -\epsilon \frac{\partial E}{\partial w_{ij}} = \epsilon (t_j - a_j) \; a_j (1 - a_j) \; a_i$$

Quick aside: The derivative of the sigmoid function

$$\frac{d}{dn_j}(a_j) = \frac{d}{dn_j}\left(\frac{1}{1+e^{-n_j}}\right)$$

$$= \frac{d}{dn_j}(1 + \ e^{-n_j})^{-1}$$

$$= -(1 + \ e^{-n_j})^{-2}(-e^{-n_j})$$

$$= \frac{e^{-n_j}}{(1+e^{-n_j})^2}$$

$$= \frac{1}{(1+e^{-n_j})} \cdot \frac{(1+e^{-n_j})-1}{(1+e^{-n_j})}$$

$$= \frac{1}{(1+e^{-n_j})} \cdot \left(\frac{1+e^{-n_j}}{1+e^{-n_j}} + \frac{1}{1+e^{-n_j}}\right)$$

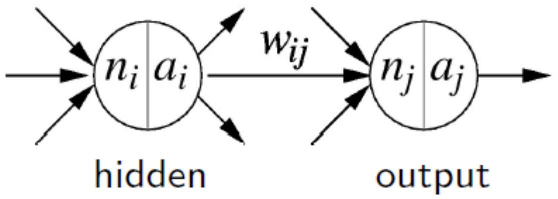
$$= \ a_j(1-a_j)$$

Generalized Delta rule (“backpropagation”)

$$n_j = \sum_i a_i \, w_{ij}$$

$$a_j = \frac{1}{1 + \exp(-n_j)}$$

$$\text{Error} \quad E = \frac{1}{2} \sum_j (t_j - a_j)^2$$



$$\begin{array}{ccccccccc} & & w_{ij} & \searrow & & & t_j & \searrow & \\ n_i & \rightarrow & a_i & \rightarrow & n_j & \rightarrow & a_j & \rightarrow & E \end{array}$$

$$\text{Gradient descent:} \qquad \triangle w_{ij} = -\epsilon \frac{\partial E}{\partial w_{ij}}$$

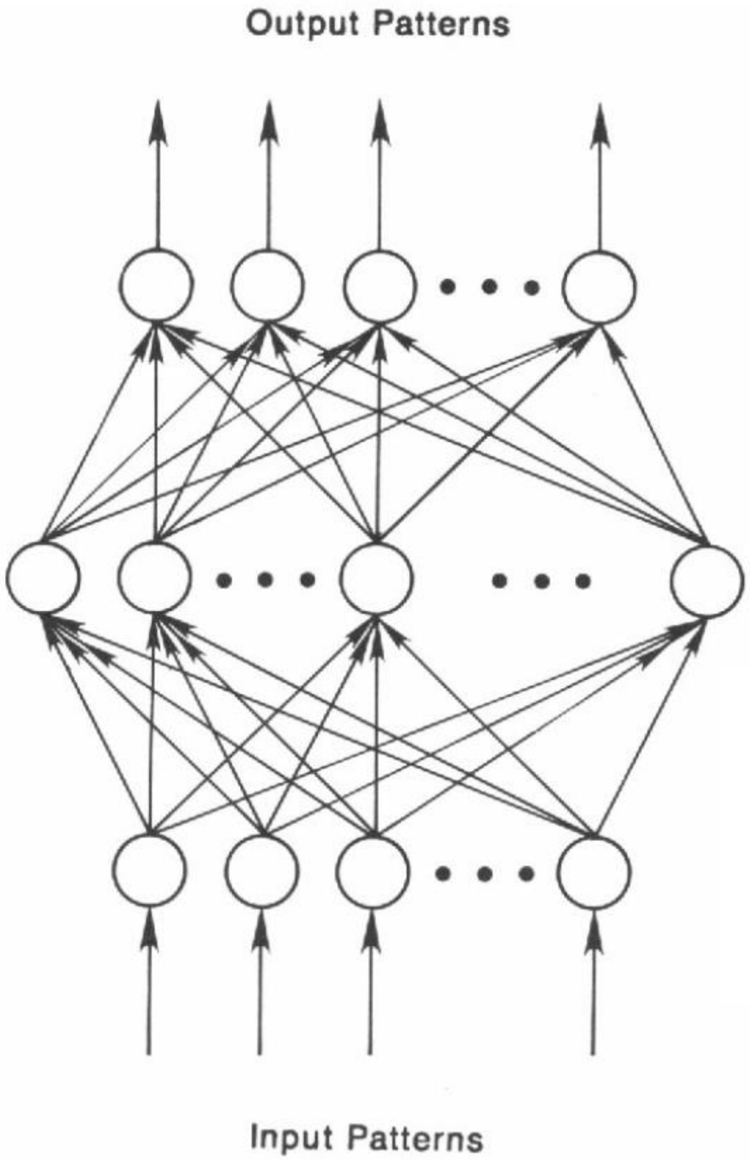
$$\frac{\partial E}{\partial n_j} = \boxed{\frac{\partial E}{\partial a_j}} \frac{\mathrm{d} a_j}{\mathrm{d} n_j} = -(t_j - a_j) \;\; a_j (1 - a_j)$$

Intermediate notation
(“input derivatives” in Lens)

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial n_j} \frac{\partial n_j}{\partial w_{ij}} = \frac{\partial E}{\partial n_j} \, a_i$$

$$\frac{\partial E}{\partial a_i} = \sum_j \frac{\partial E}{\partial n_j} \frac{\partial n_j}{\partial a_i} = \sum_j \frac{\partial E}{\partial n_j} \, w_{ij}$$

Generalized Delta rule (“backpropagation”)



= computed from the top (hidden-to-output) layer

= computed from the bottom (input-to-hidden) layer

$$\frac{\partial E}{\partial w_{ij}L} = \frac{\partial E}{\partial a_j}\frac{\partial a_j}{\partial n_j}\frac{\partial n_i}{\partial w_{ij}L} = \frac{\partial E}{\partial n_j}\frac{\partial n_i}{\partial w_{ij}L} = \frac{\partial E}{\partial n_j}a_i$$

Sigmoid derivative$_1$

Sigmoid derivative$_2$

Sigmoid derivative$_2$

$$\frac{\partial E}{\partial w_{ij}L-1} = \frac{\partial E}{\partial a_j}\frac{da_j}{dn_j}\frac{\partial n_j}{\partial a_i}\frac{da_i}{dn_i}\frac{\partial n_i}{\partial w_{ij}L-1} = \frac{\partial E}{\partial n_j}\frac{\partial n_j}{\partial ai}\frac{da_i}{dn_i}\frac{\partial n_i}{\partial w_{ij}L-1} = \frac{\partial E}{\partial n_j}\frac{\partial n_j}{\partial w_{ij}L-1}$$

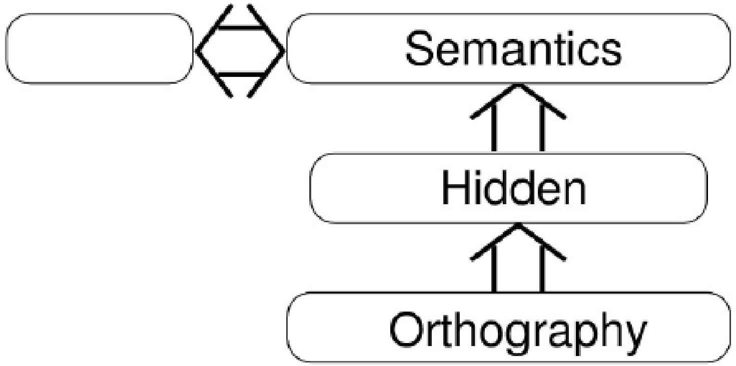
Backpropagation *in words*

Backpropagation is an algorithm that involves:

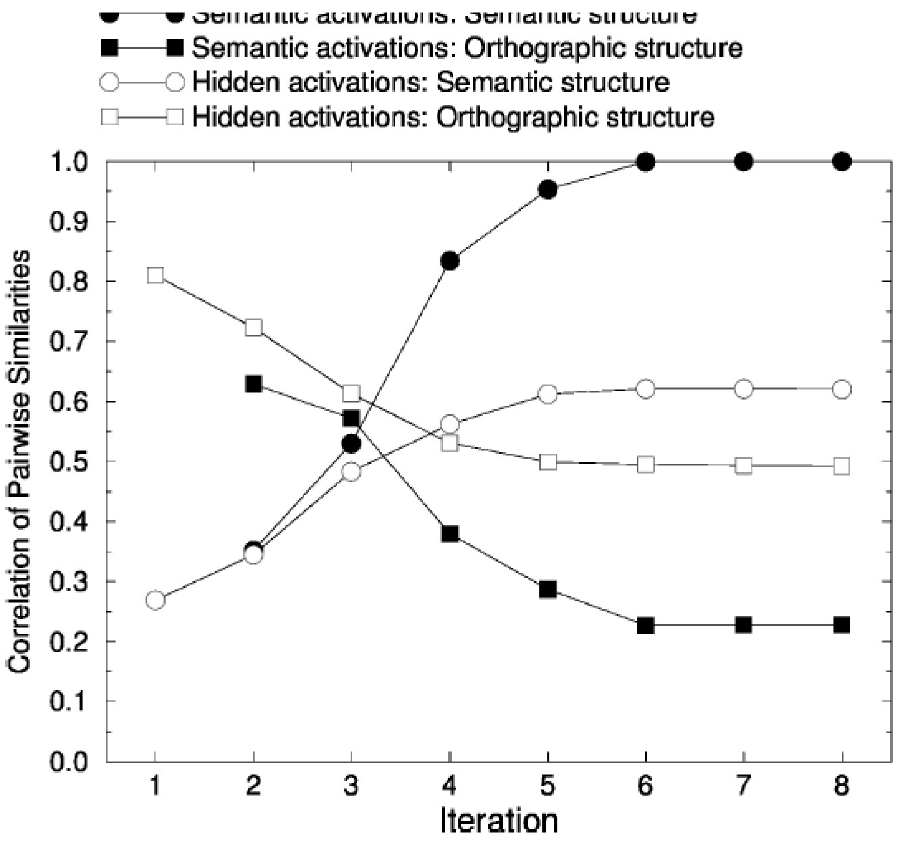
1. Computing net inputs and activations *in a forward pass.*
2. Computing total network error (over all training examples).
3. Adjusting network weights based on the total error *in the backward pass.*
4. Repeat steps 1-3 until the network converges.

What do hidden representations learn

Plaut and Shallice (1993)

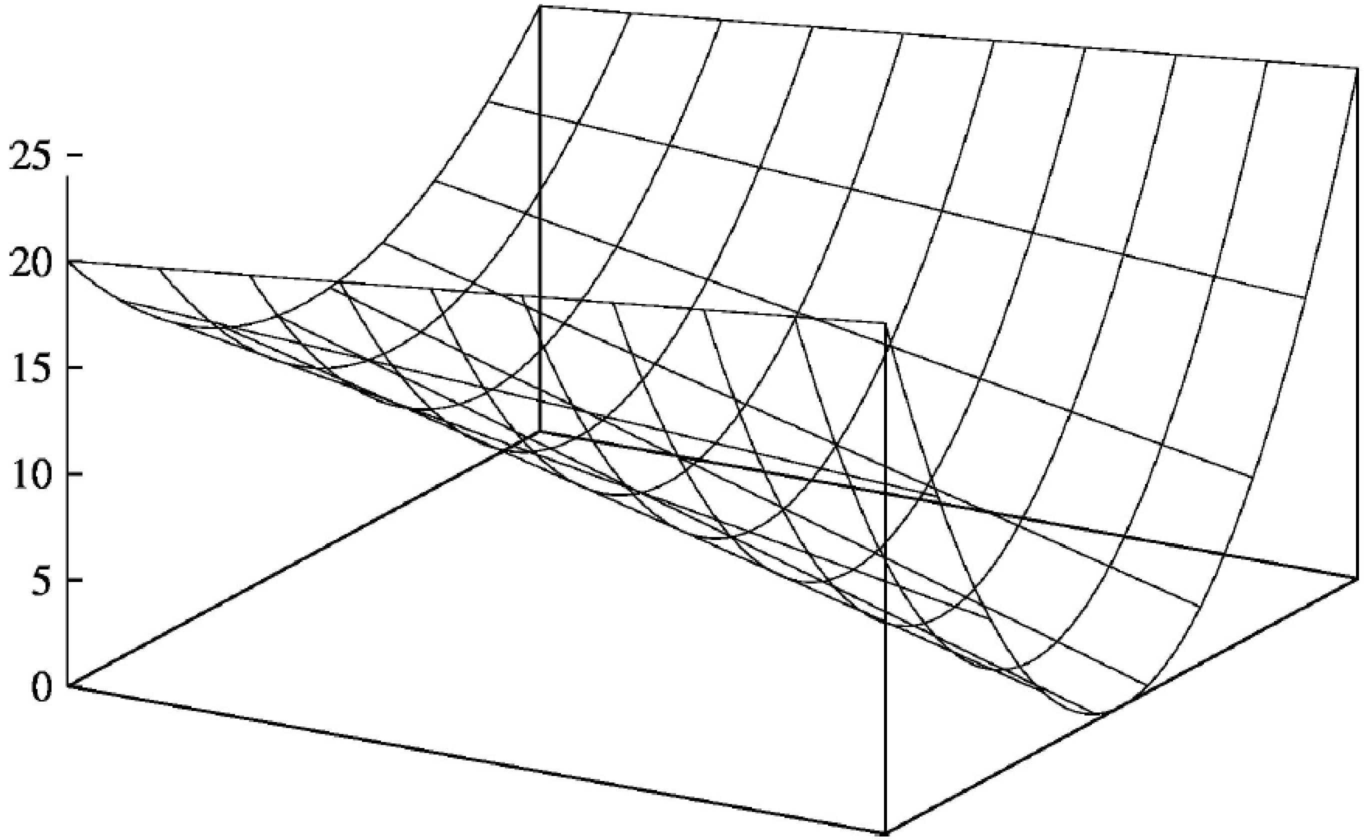


- Mapped orthography to semantics (unrelated similarities)
- Compared similarities among hidden representations to those among orthographic and semantic representations (over settling)



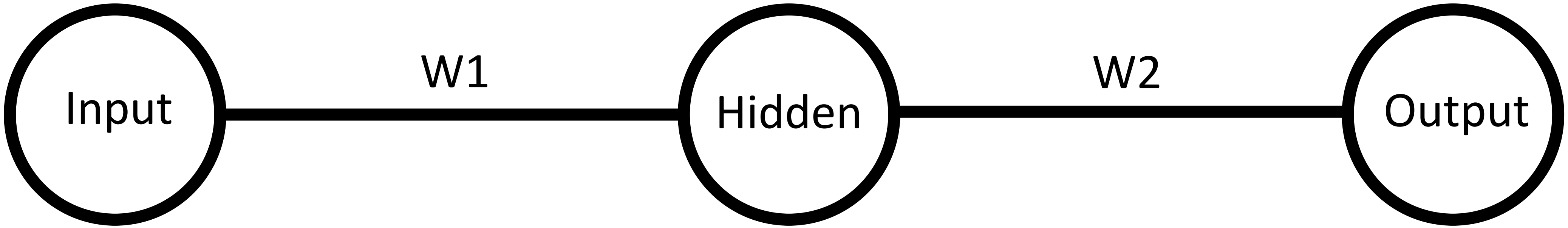
- Hidden representations **"split the difference"** between input and output similarity

Accelerating learning: Momentum descent

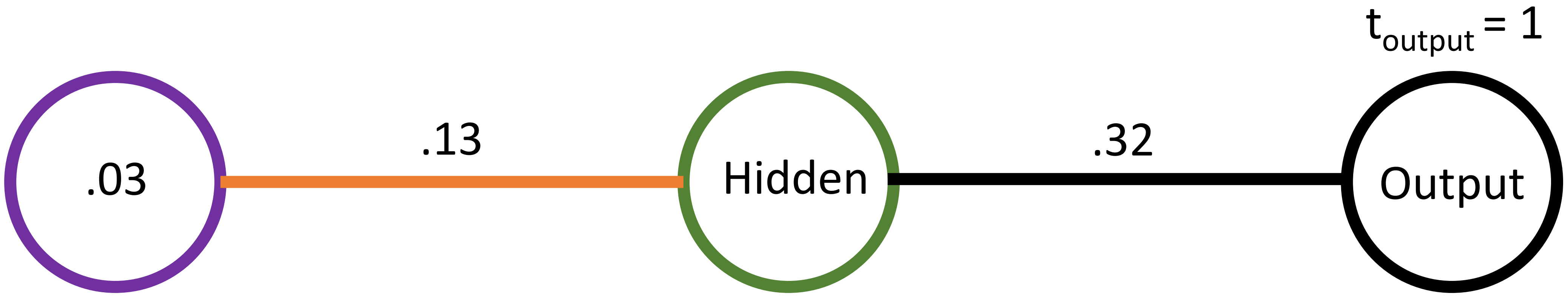


$$\triangle w_{ij}{}^{[t]} \;=\; -\epsilon\,\frac{\partial E}{\partial w_{ij}} \;+\; \alpha\left(\triangle w_{ij}^{[t-1]}\right)$$

Backpropagation: A really simple example
The forward pass



Backpropagation: A really simple example
The forward pass

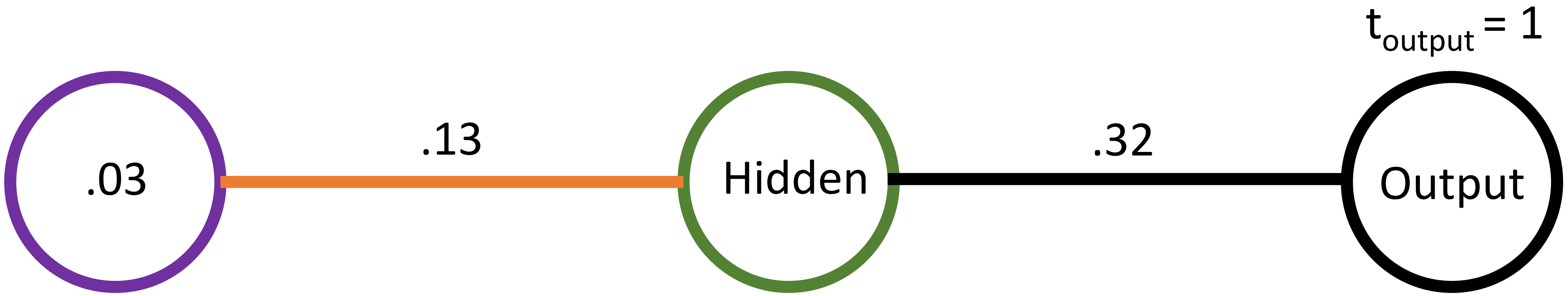


$$net_{hidden} = a_{input} * w_{input*hidden}$$

$$a_{hidden} = \frac{1}{1 + e^{-net_{hidden}}}$$

Backpropagation: A really simple example

The forward pass

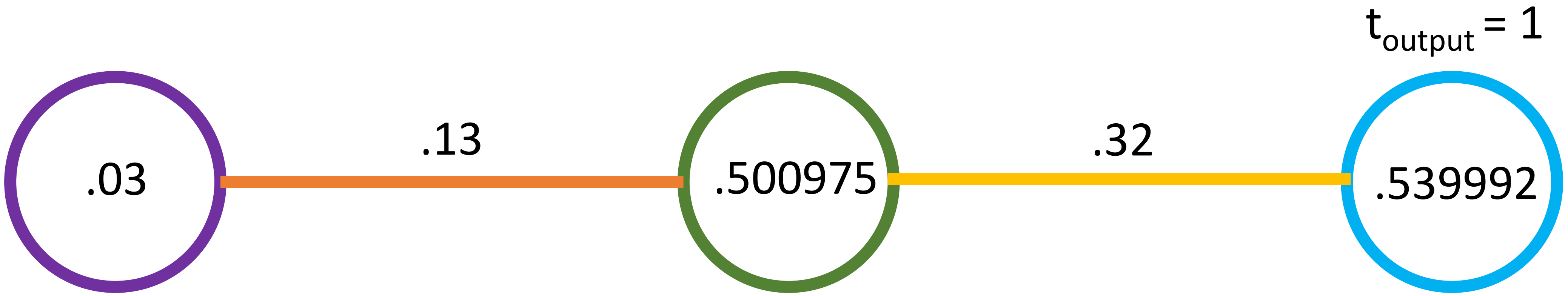


$net_{hidden} = .03 * .13 = .0039$

$$a_{hidden} = \frac{1}{1 + e^{-.0039}} \sim .500975$$

Backpropagation: A really simple example

The forward pass

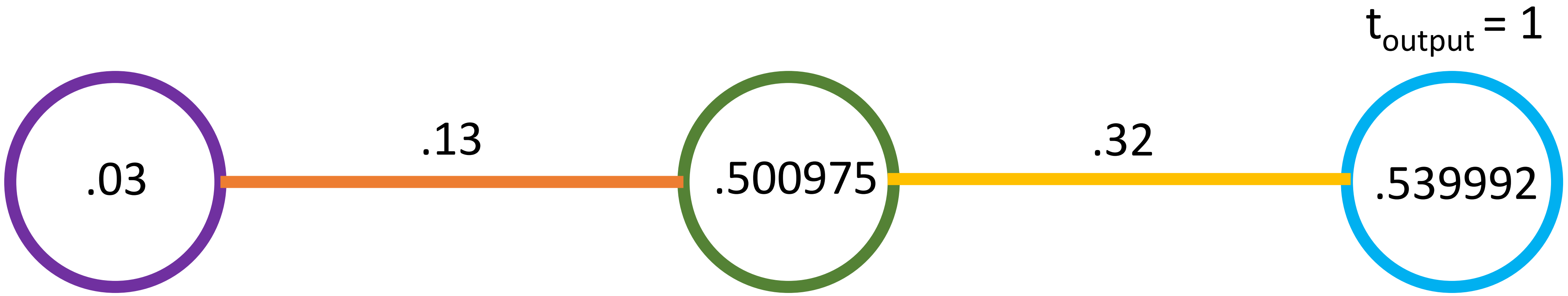


$net_{output} = .500975 * .32 = .160312$

$$a_{output} = \frac{1}{1 + e^{-.160312}} \sim .539992$$

Backpropagation: A really simple example

The forward pass

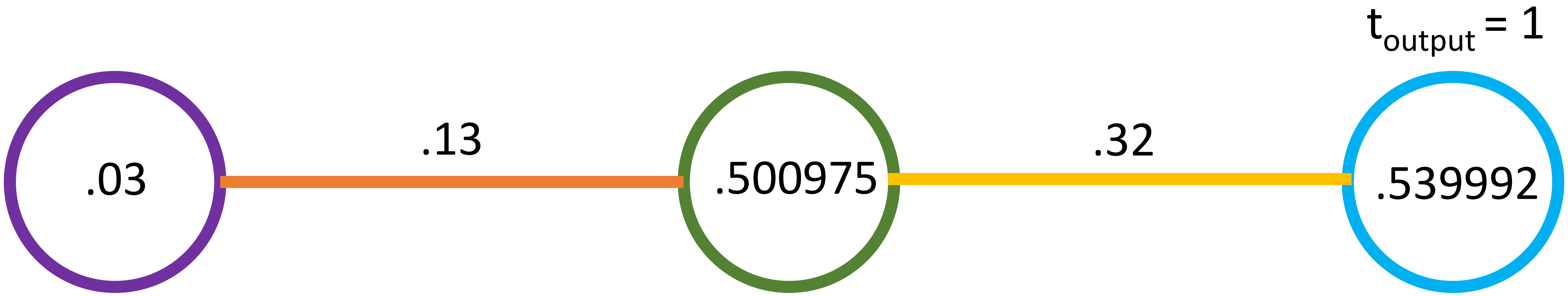


Network error $= \frac{1}{2}(t_j - a_j)^2$

Network error $= \frac{1}{2}(1 - .539992)^2 = .105804$

Backpropagation: A really simple example

The backwards pass



$$\begin{matrix} w_{ij} & \searrow & & & t_j & \searrow & \\ a_i & \to & n_j & \to & a_j & \to & E \end{matrix}$$

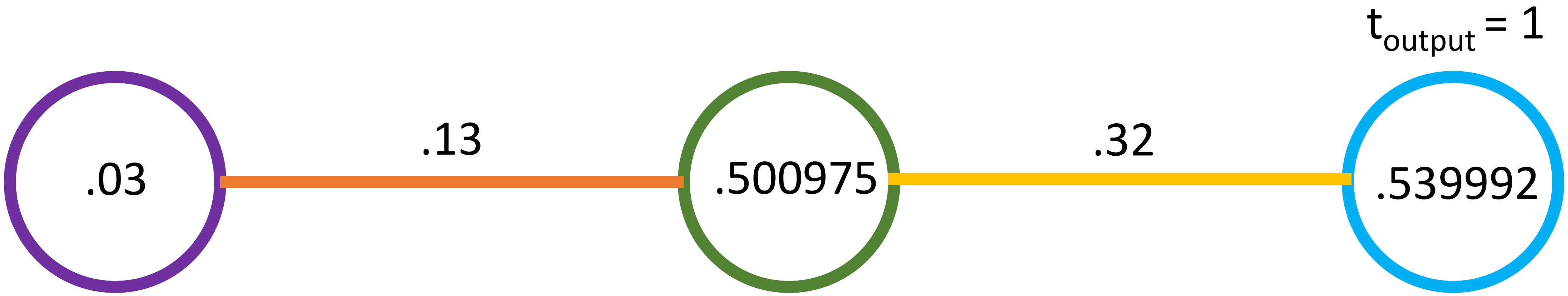
$$\frac{\partial E}{\partial w_2} = \frac{\partial E}{\partial a_{output}} \frac{\partial a_{output}}{\partial net_{output}} \frac{\partial net_{output}}{\partial w_2}$$

$$= -\in (t_j - a_j) \quad a_j(1 - a_j) \quad a_i$$

This is just the Delta rule for sigmoid unit!

Backpropagation: A really simple example

The backwards pass



$$\frac{\partial E}{\partial w_2} = \frac{\partial E}{\partial a_{output}} \; \frac{\partial a_{output}}{\partial net_{output}} \; \frac{\partial net_{output}}{\partial w_2}$$

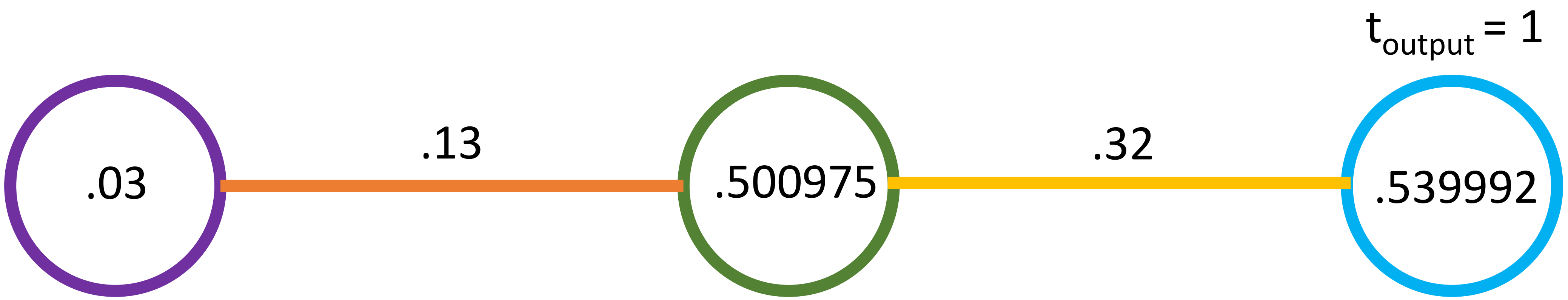
$$\begin{array}{ccccccc} w_{ij} & \searrow & & & t_j & \searrow & \\ a_i & \rightarrow & n_j & \rightarrow & a_j & \rightarrow & E \end{array}$$

$$= (1 - .539992)\ .539992(1 - .539992)\ \ .500975$$

$$= .460008\ .248401\ .500975$$

$$= .057245$$

Backpropagation: A really simple example
The backwards pass



$$\frac{\partial E}{\partial w_2} = \frac{\partial E}{\partial a_{output}} \quad \frac{\partial a_{output}}{\partial net_{output}} \quad \frac{\partial net_{output}}{\partial w_2}$$

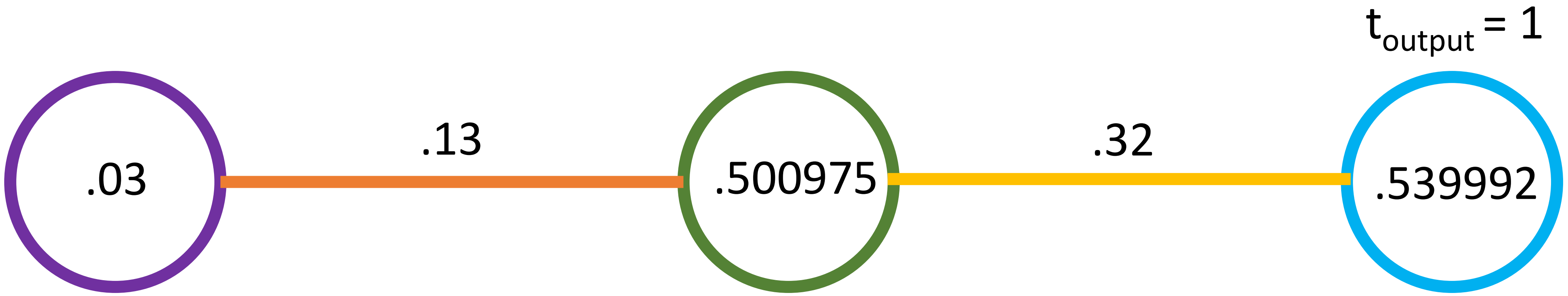
$$\begin{array}{ccccccc} w_{ij} & \searrow & & & t_j & \searrow & \\ a_i & \rightarrow & n_j & \rightarrow & a_j & \rightarrow & E \end{array}$$

$$\Delta w_2 = .32 - 0.5 * .057245$$

$$New\_w_2 = .3486225$$

Backpropagation: A really simple example

The backwards pass



$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial a_{output}} \frac{\partial a_{output}}{\partial net_{output}} \frac{\partial net_{output}}{\partial a_{hidden}} \frac{\partial a_{hidden}}{\partial net_{hidden}} \frac{\partial net_{hidden}}{\partial w_1}$$

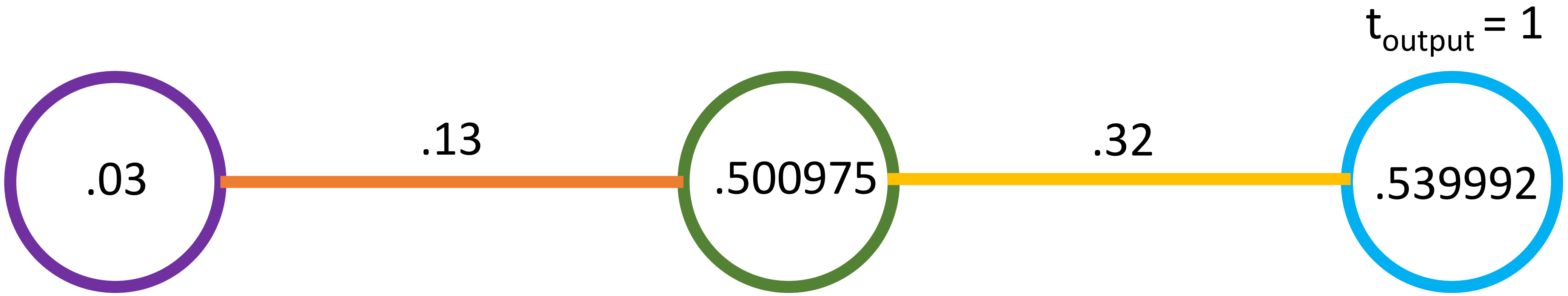
$$-\epsilon\,(t_j - a_j) \quad a_j(1 - a_j) \quad w_{ij} \quad a_k(1 - a_k) \quad a_{ki}$$

$$= (1 - .539992)\ .539992(1 - .539992)\ .32\ .500975(1 - .500975)\ .03$$

$$= .460008 * .248401 * .32 * .249999 * .03 = .000274$$

Backpropagation: A really simple example

The backwards pass



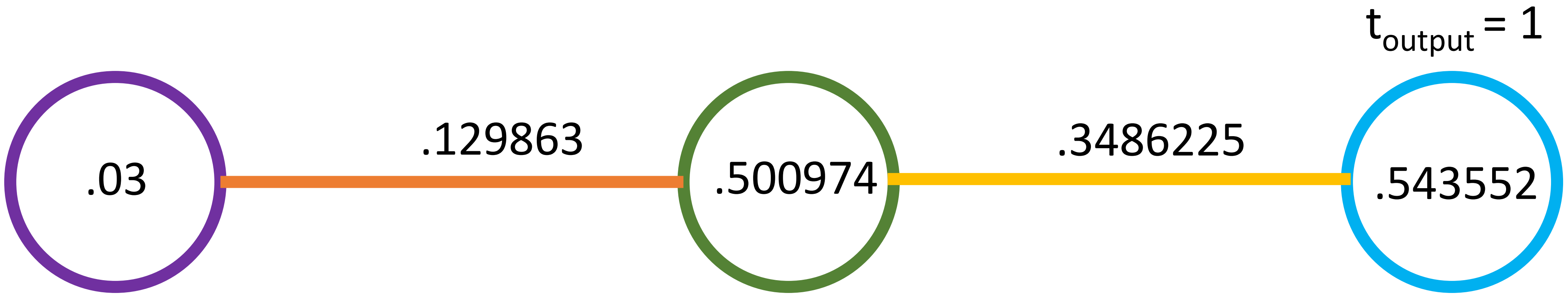
$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial a_{output}} \frac{\partial a_{output}}{\partial net_{output}} \frac{\partial net_{output}}{\partial a_{hidden}} \frac{\partial a_{hidden}}{\partial net_{hidden}} \frac{\partial net_{hidden}}{\partial w_1}$$

$$\Delta w_1 = .13 - 0.5 * .000274$$

$$New\_w_1 = .129863$$

Backpropagation: A really simple example

Update the weights!



We're doing better, albeit slightly! Keep going!