University of California, Merced

COGS 125 / CSE 175 : Introduction to Artificial Intelligence

Programming Assignment #4 Notes

David C. Noelle, Ph.D.

Matrices

A **matrix** is like a 2-D array.

$$W = \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \\ w_{41} & w_{42} & w_{43} \end{pmatrix}$$

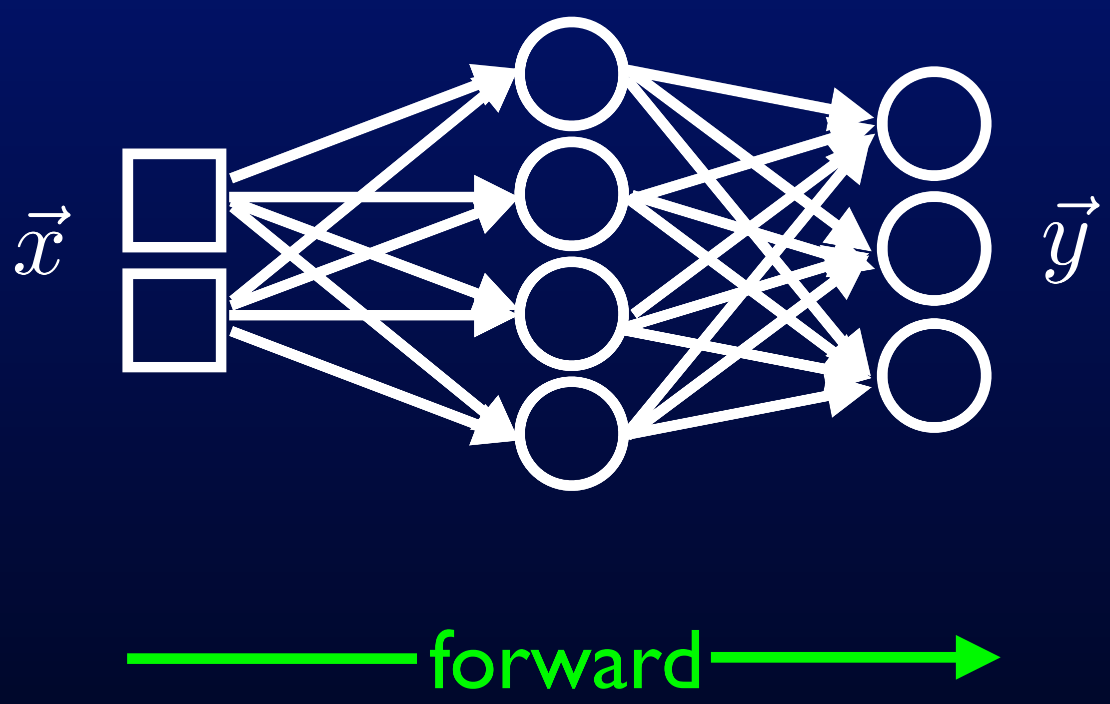
Multiplying a matrix by a vector produces another vector:

Note that the dimensionality of the vector must equal the number of columns of the matrix.

$$W\vec{x} = \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \\ w_{41} & w_{42} & w_{43} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} x_1 w_{11} + x_2 w_{12} + x_3 w_{13} \\ x_1 w_{21} + x_2 w_{22} + x_3 w_{23} \\ x_1 w_{31} + x_2 w_{32} + x_3 w_{33} \\ x_1 w_{41} + x_2 w_{42} + x_3 w_{43} \end{pmatrix}$$

The Generalized Delta Rule

Forward Activation Propagation



Moving layer by layer, inputs to outputs, calculate:

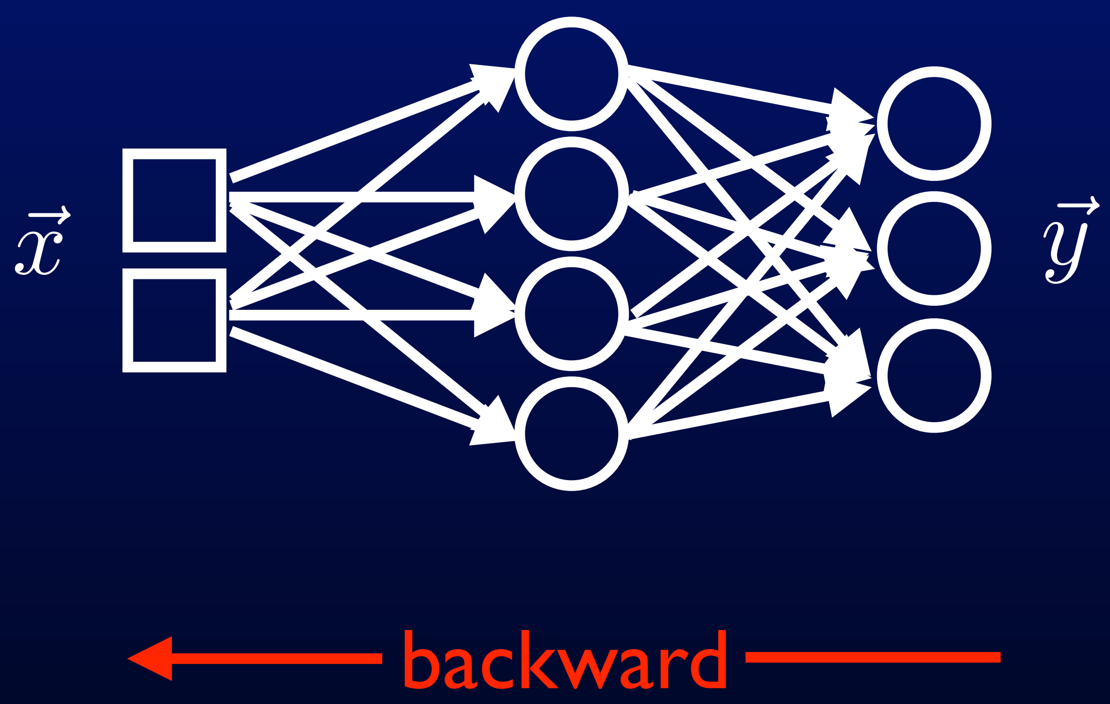
$$in_i = \sum_j w_{j,i} \; a_j$$

$$a_i = g(in_i)$$

... where g is often the logistic. (Don't forget bias weights!)

The Generalized Delta Rule

Backward Error Propagation



Moving layer by layer, outputs to inputs, calculate:

$$\delta_i \;=\; g'(in_i)\;(T_i - a_i)$$

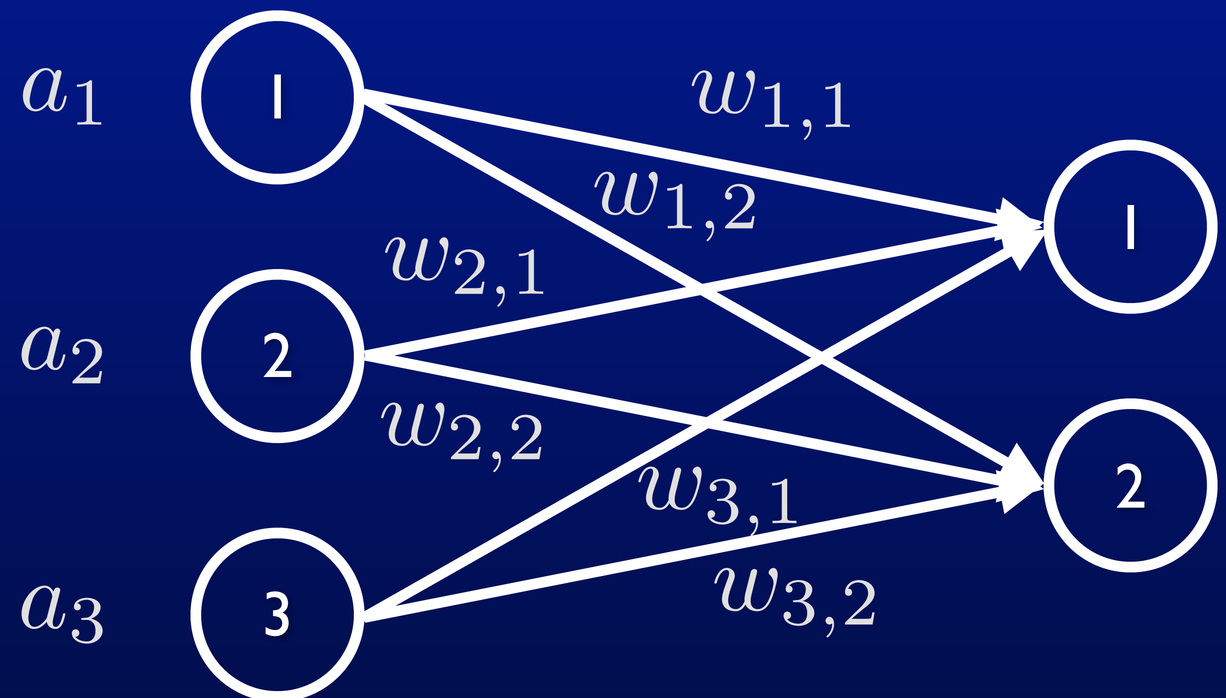
… for outputs

$$\delta_i \;=\; g'(in_i)\;\sum_k \delta_k\; w_{i,k}$$

… otherwise

$$\Delta\; w_{j,i} \;\propto\; \delta_i\; a_j$$

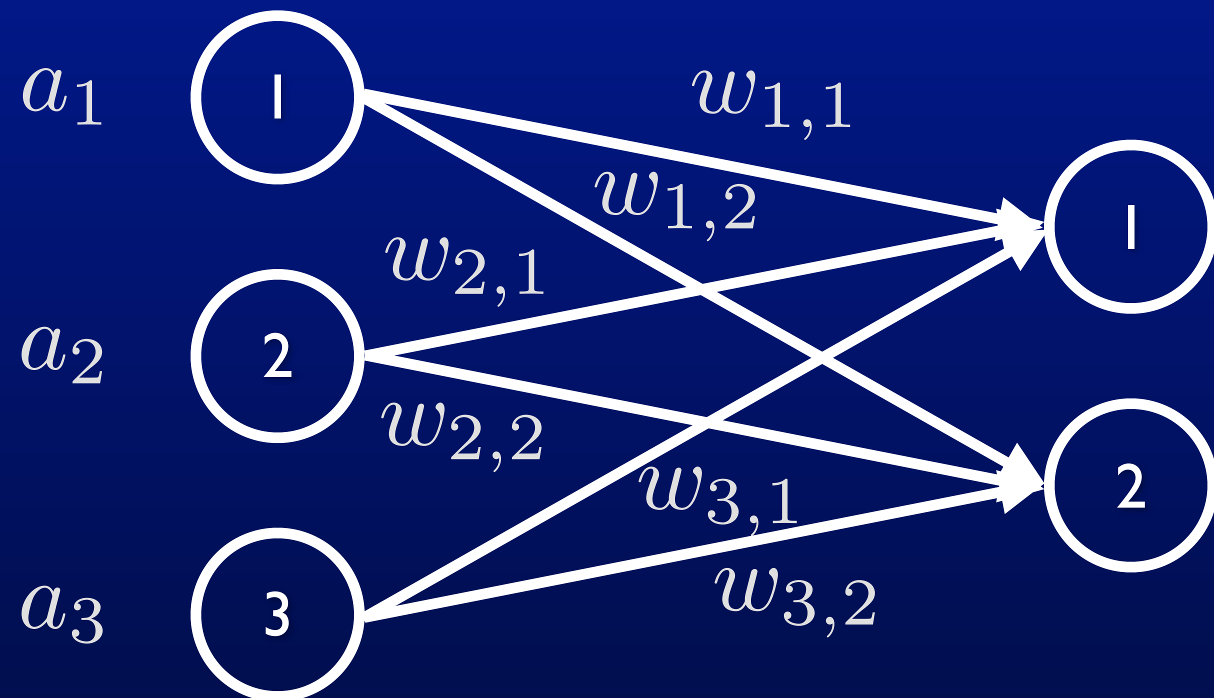
A Projection Between Layers



$$w_{i,j} \equiv \textit{the weight from unit } i \textit{ to unit } j$$

$$W = \begin{pmatrix} w_{1,1} & w_{2,1} & w_{3,1} \\ w_{1,2} & w_{2,2} & w_{3,2} \end{pmatrix}$$

Forward Pass



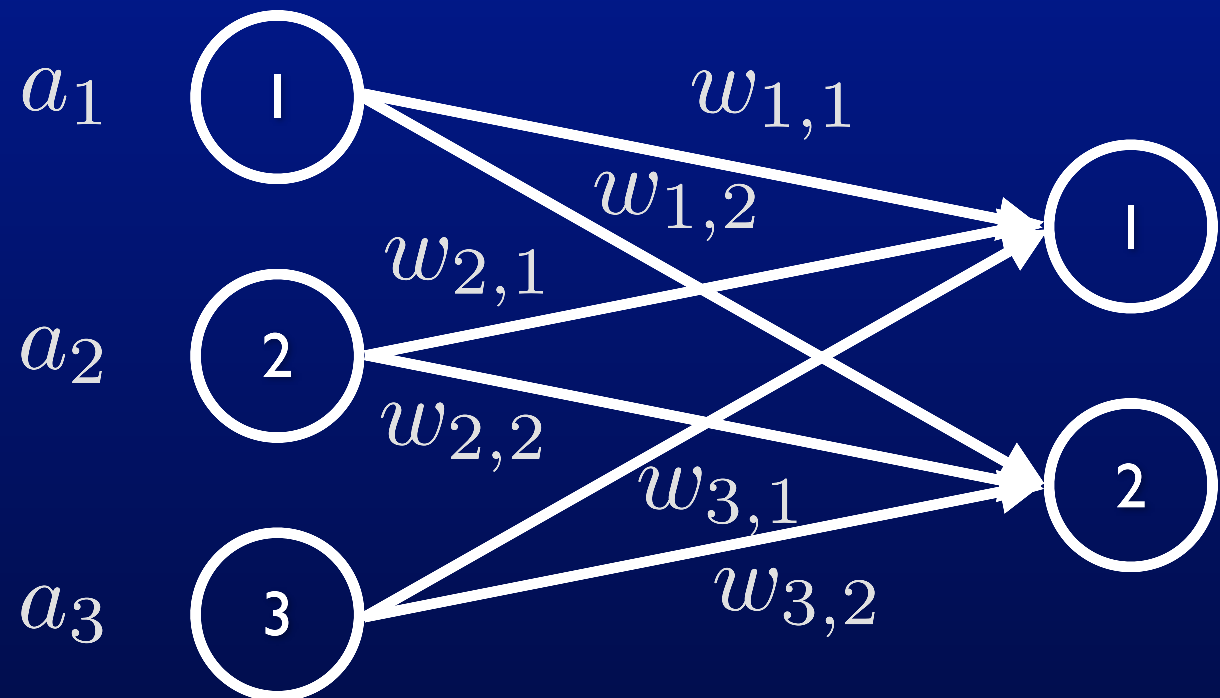
$$in_1 = \sum_{j=1}^{3} w_{j,1}\ a_j$$

$$in_2 = \sum_{j=1}^{3} w_{j,2}\ a_j$$

$$\vec{a} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \qquad W = \begin{pmatrix} w_{1,1} & w_{2,1} & w_{3,1} \\ w_{1,2} & w_{2,2} & w_{3,2} \end{pmatrix} \qquad \vec{in} = \begin{pmatrix} in_1 \\ in_2 \end{pmatrix}$$

$$W\ \vec{a} = \vec{in}$$

Forward Pass



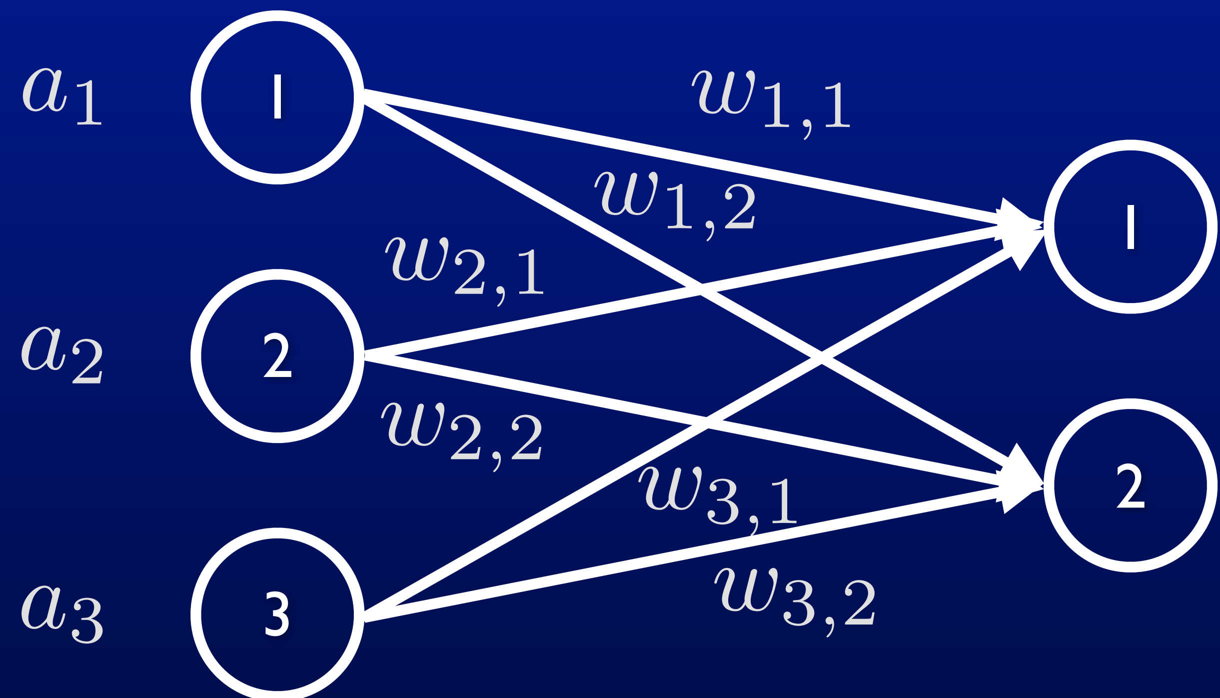
$$in_1 = \sum_{j=1}^{3} w_{j,1}\ a_j$$

$$in_2 = \sum_{j=1}^{3} w_{j,2}\ a_j$$

$$\begin{pmatrix} w_{1,1} & w_{2,1} & w_{3,1} \\ w_{1,2} & w_{2,2} & w_{3,2} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} w_{1,1}a_1 + w_{2,1}a_2 + w_{3,1}a_3 \\ w_{1,2}a_1 + w_{2,2}a_2 + w_{3,2}a_3 \end{pmatrix} = \begin{pmatrix} in_1 \\ in_2 \end{pmatrix}$$

$$W\ \vec{a} = \vec{in}$$

Forward Pass



$$in_1 = \sum_{j=1}^{3} w_{j,1}\, a_j$$

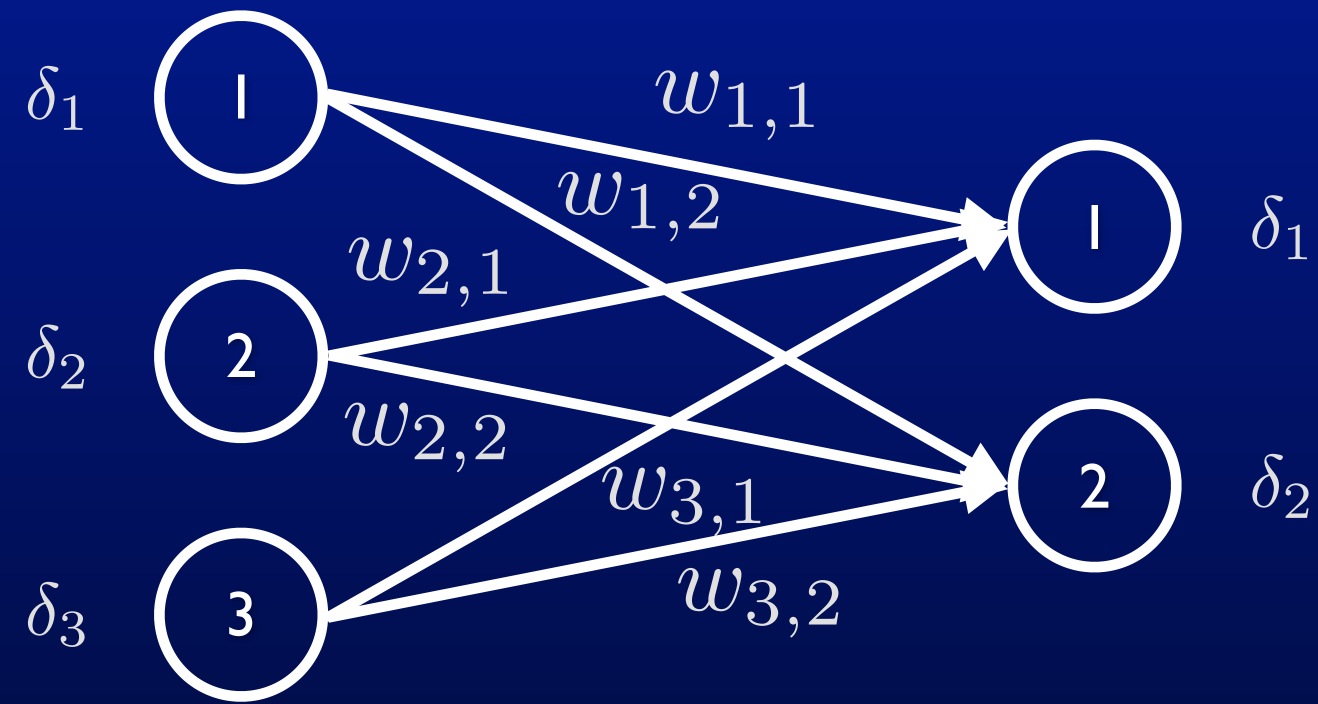
$$in_2 = \sum_{j=1}^{3} w_{j,2}\, a_j$$

```
// computeActivation -- Calculate the activation values of the units in
// this layer based on their inputs and bias weights.
public void computeActivation() {
        if (!(inputs.isEmpty())) {
                // This is not an input layer, so we can update it ...
                // Add in the bias values to the net inputs ...
                net = net.copy(bias);
                // Sum up the contributions of each projection ...
                for (Projection p : inputs)
                        net = net.sum(p.W.product(p.input.act));
                act = net.squash(min, max);
        }
}
```

$W\,\vec{a} = \vec{in}$ = `p.W.product(p.input.act)`

Backward Pass

need to calculate these deltas



already calculated these deltas

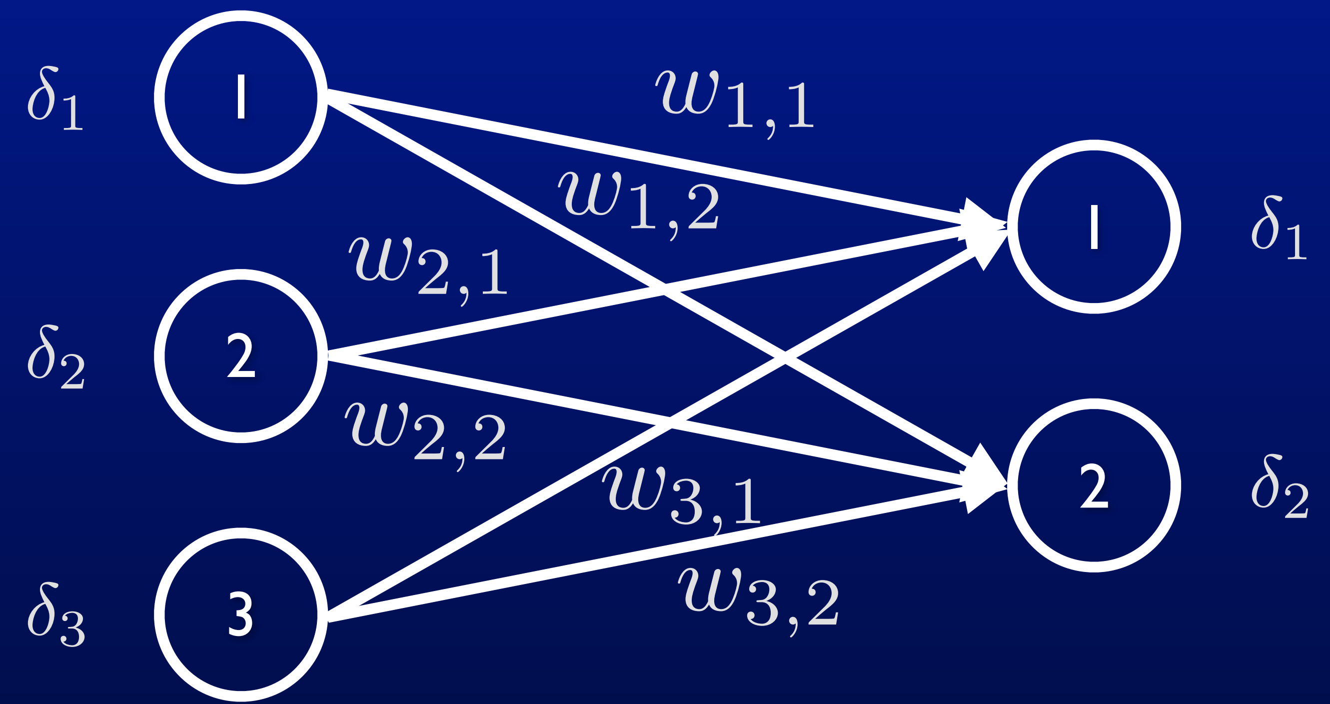
$$\delta_1 = g'(in_1) \sum_{k=1}^{2} \delta_k \, w_{1,k}$$

$$\delta_2 = g'(in_2) \sum_{k=1}^{2} \delta_k \, w_{2,k}$$

$$\delta_3 = g'(in_3) \sum_{k=1}^{2} \delta_k \, w_{3,k}$$

Backward Pass

need to calculate these deltas



already calculated these deltas

$$\delta_1 \;=\; g'(in_1) \sum_{k=1}^{2} \delta_k \; w_{1,k} \;=\; g'(in_1) \; \sigma_1$$

$$\delta_2 \;=\; g'(in_2) \sum_{k=1}^{2} \delta_k \; w_{2,k} \;=\; g'(in_2) \; \sigma_2$$

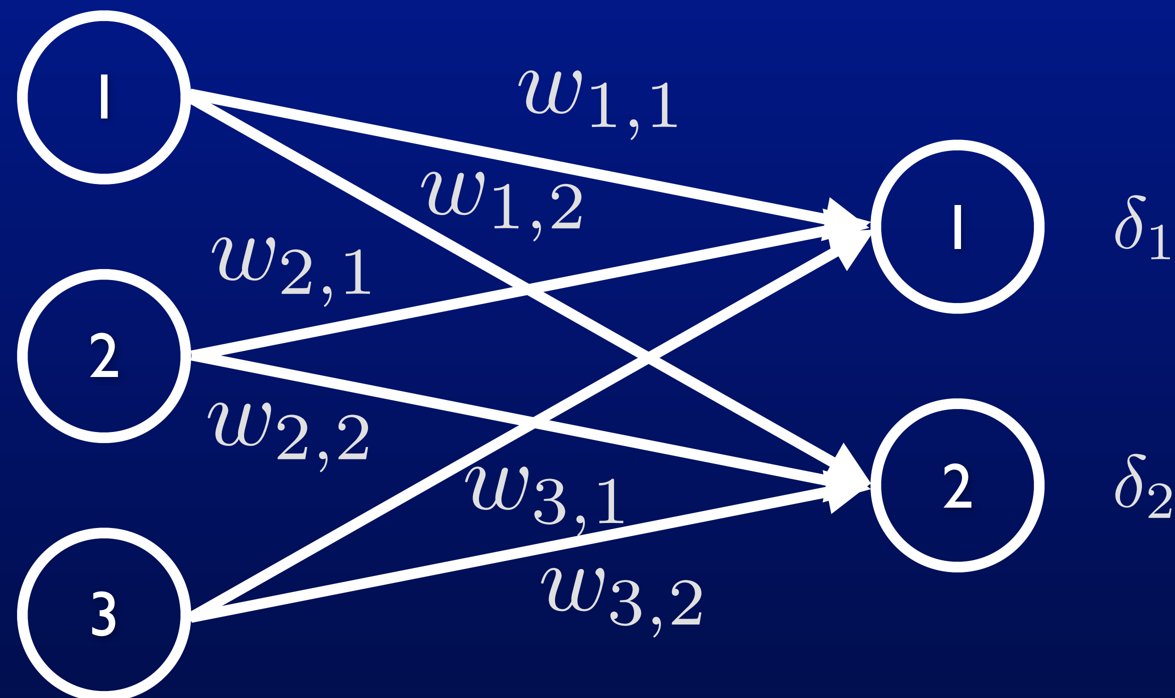
$$\delta_3 \;=\; g'(in_3) \sum_{k=1}^{2} \delta_k \; w_{3,k} \;=\; g'(in_3) \; \sigma_3$$

Backward Pass

$$\delta_1 = g'(in_1)\ \sigma_1$$

$$\delta_2 = g'(in_2)\ \sigma_2$$

$$\delta_3 = g'(in_3)\ \sigma_3$$



already calculated these deltas

$$\vec{\sigma} = \begin{pmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \end{pmatrix} \qquad W = \begin{pmatrix} w_{1,1} & w_{2,1} & w_{3,1} \\ w_{1,2} & w_{2,2} & w_{3,2} \end{pmatrix} \qquad \vec{\delta} = \begin{pmatrix} \delta_1 \\ \delta_2 \end{pmatrix}$$

How do we calculate $\vec{\sigma}$ from W and $\vec{\delta}$?

Matrices

A **matrix** is like a 2-D array.

$$W = \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \\ w_{41} & w_{42} & w_{43} \end{pmatrix}$$

Multiplying a matrix by a vector produces another vector:

Note that the dimensionality of the vector must equal the number of columns of the matrix.

$$W\vec{x} = \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \\ w_{41} & w_{42} & w_{43} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} x_1 w_{11} + x_2 w_{12} + x_3 w_{13} \\ x_1 w_{21} + x_2 w_{22} + x_3 w_{23} \\ x_1 w_{31} + x_2 w_{32} + x_3 w_{33} \\ x_1 w_{41} + x_2 w_{42} + x_3 w_{43} \end{pmatrix}$$