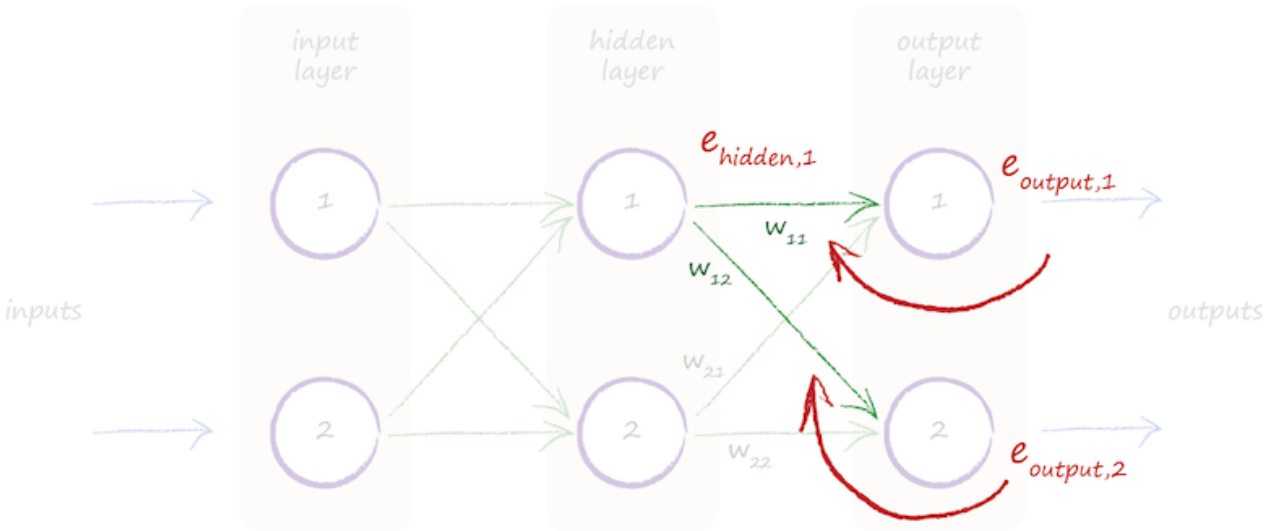
Backpropagation: Recombining the Error

In this lesson, we will try to come up with a formula to recombine bits of error at each internal node that we split in the previous lesson.

> The training data examples only tell us what the outputs from the very final nodes should be. They don't tell us what the outputs from nodes in any other layer should be. This is the core of the puzzle. We could recombine the split errors for the links using the error backpropagation we just saw earlier.

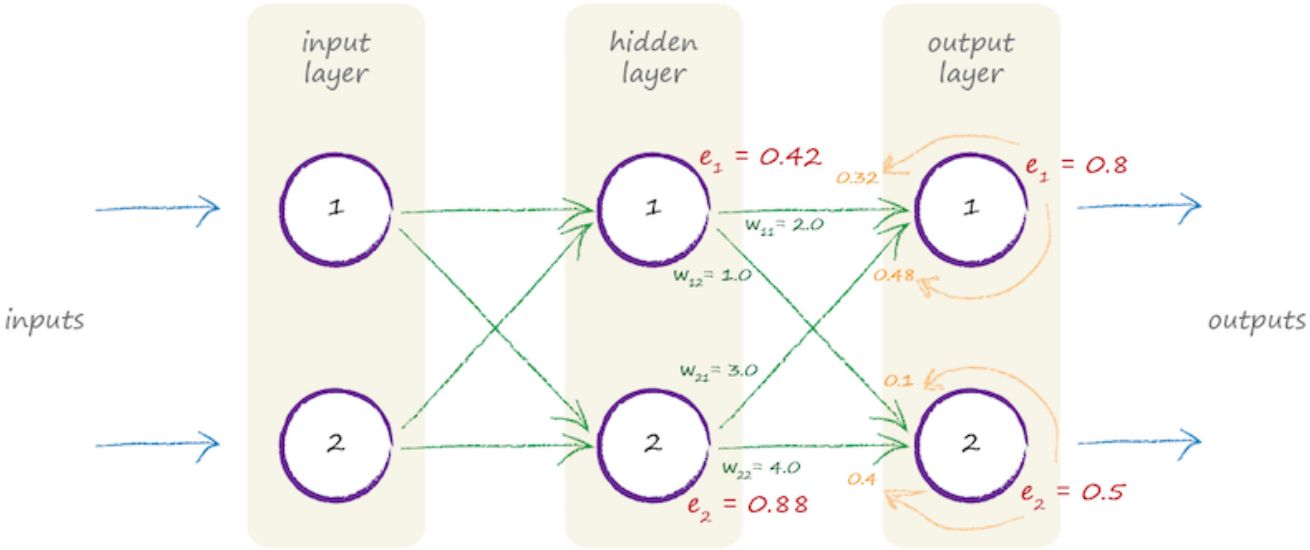


So the error in the first hidden node is the sum of the split errors in all the links connecting forward from the same node. In the diagram above, we have a fraction of the output error $e_{output,1}$ on the link with weight w_{11} and also a fraction of the output error $e_{output,2}$ from the second output node on the link with weight w_{12}. So let's write this down.

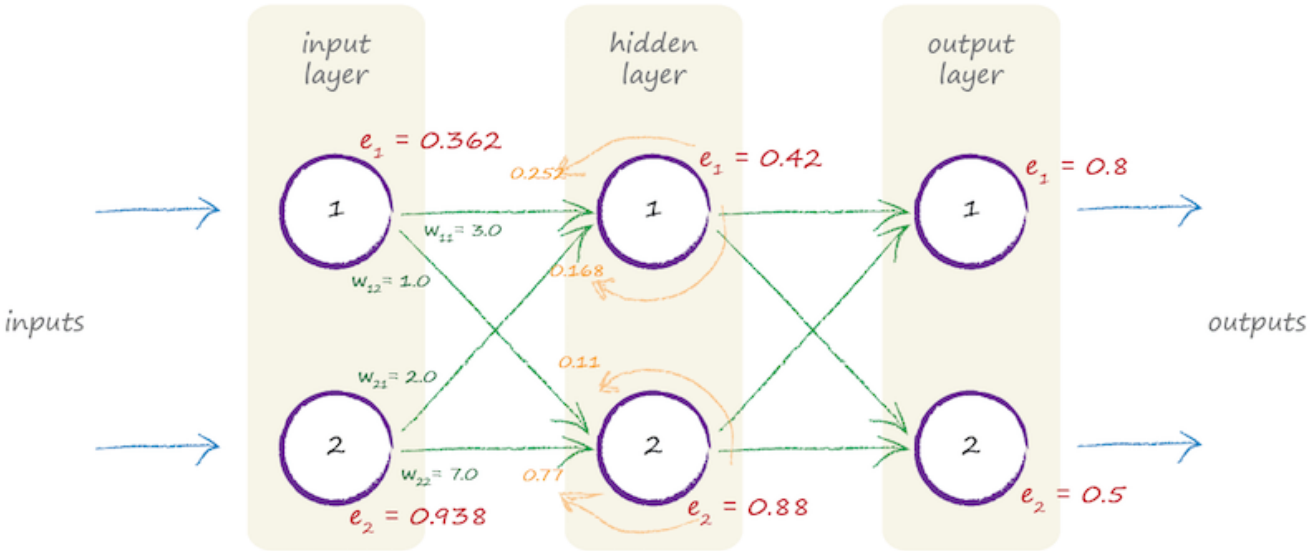
$$e_{hidden,1} = \text{sum of split errors on links } w_{11} \text{ and } w_{12}$$

$$= e_{output,1} * \frac{w_{11}}{w_{11} + w_{21}} + e_{output,2} * \frac{w_{12}}{w_{12} + w_{22}}$$

It helps to see all this theory in action, so the following illustrates the propagation of errors back into a simple three-layer network with actual numbers.



Let's follow one error back. You can see the error 0.5 at the second output layer node being split proportionately into 0.1 and 0.4 across the two connected links which have weights 1.0 and 4.0. You can also see that the recombined error at the second hidden layer node is the sum of the connected split errors, which are 0.48 and 0.4, to give 0.88. The next diagram shows the same idea applied to the preceding layer, working further back.



Till this point, we have only covered back-propagation by using "Split and Recombine" approach, in the next lesson we will take a look at a different

Recombine" approach, in the next lesson we will take a look at a different approach.