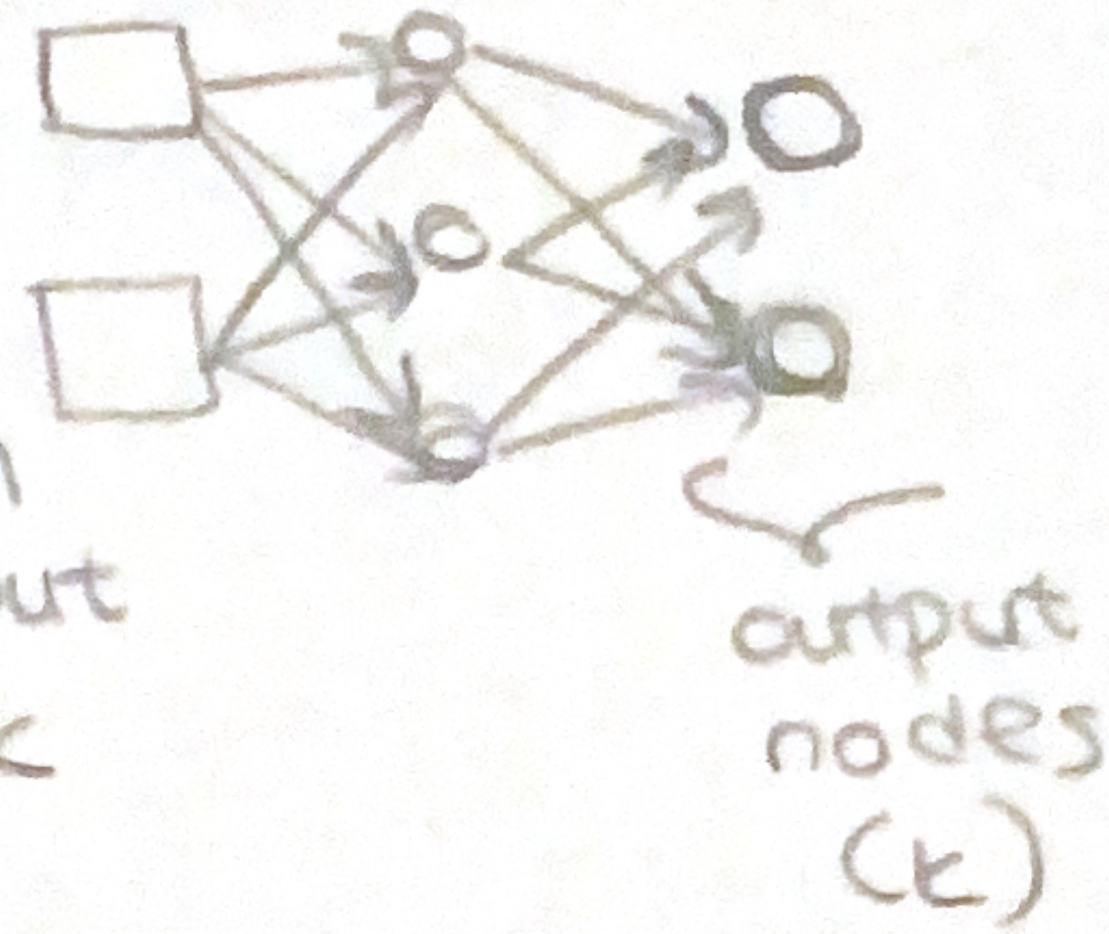
Backpropagation

$$\frac{\partial}{\partial w} Loss(w) = \frac{\partial}{\partial w}|y-h_w(x)|^2 = \frac{\partial}{\partial w}\sum(y_k-a_k)^2 =$$

$$\sum_k \frac{\partial}{\partial w}(y_k - a_k)^2 \rightarrow$$



Loss calculation of perceptron

target

activation in output node k

- In multilayer networks we take derivative of overall error gradient.

  With multiple output units (k):

  Err_k : k^{th} component of error vector $y - h_w$

  $\Delta_k = Err_k \times g'(in_k)$ } modified error

  $w_{j,k} \leftarrow w_{j,k} + \alpha \times a_j \times \Delta_k$ } weight update rule

- To update connections between input units & hidden units we need to define a quantity analogous to the error term for output nodes. (where we need error backprop)
- Hidden node j is responsible for some fraction of Δ_k
- **Δ_k values are divided according to strength of connection between hidden node & output node.** They are propagated back to provide Δ_j for hidden layer.

→ Propagation rules for Δ values : $\Delta_j = g'(in_j)\sum_k w_{j,k}\Delta_k$

- Backprop in a nutshell
  1. Compute Δ values for output units, use observed error
  2. For each layer until the earliest hidden layer:
     Propagate Δ values back to previous layer
     Update weights between two layers