First step of backpropagation is defining the loss function. In this case of building the neural network from scratch, we will be using the Cross-entropy log loss function. For different loss function, the partial derivatives will be different.

1. Finding partial derivative of loss function (L) with respect to the output node's activation function (A) which is the final output value:

$$\begin{aligned}\frac{\partial L}{\partial A} &= \frac{\partial}{\partial A}(-ylog(A) - (1-y)log(1-A)) \\ &= -\frac{y}{A} - (-\frac{1-y}{1-A}) \\ &= -\frac{y}{A} + \frac{1-y}{1-A}\end{aligned}$$

Note: Start applying chain rule to find the rest of the partial derivatives:

2. Finding partial derivative of loss function (L) with respect to the linear output for the output nodes (Z):

$$\frac{\partial L}{\partial Z} = \frac{\partial L}{\partial A}\frac{\partial A}{\partial Z} \tag{1}$$

2.1 Find partial derivative of activation function (A) with respect to the linear output for the output nodes (Z) first, assume usage of Sigmoid Activation Function in output layer: $A = \frac{1}{1+e^{-Z}}$

$$\begin{aligned}\frac{\partial A}{\partial Z} &= \frac{\partial}{\partial Z}(\frac{1}{1+e^{-Z}}) \\ &= \frac{\partial}{\partial A}((1+e^{-Z})^{-1}) \\ &= -(1+e^{-Z})^{-2}(-1 * e^{-Z}) \\ &= \frac{e^{-Z}}{(1+e^{-Z})^2} \\ &= \frac{1+e^{-Z}-1}{(1+e^{-Z})}\frac{1}{(1+e^{-Z})} \\ &= \left[1 - \frac{1}{(1+e^{-Z})}\right]\frac{1}{(1+e^{-Z})} \\ &= (1-A)A\end{aligned}$$

$$\begin{aligned}\therefore \frac{\partial L}{\partial Z} &= \frac{\partial L}{\partial A}\frac{\partial A}{\partial Z} \\ &= \left(-\frac{y}{A} + \frac{1-y}{1-A}\right)A - y \\ &= \frac{-y(1-A) + (1-y)A}{(1-A)(A)}A - y \\ &= -y(1-A) + (1-y)A \\ &= -y - Ay + A - Ay \\ &= A - y\end{aligned}$$

3. Finding partial derivative of loss function (L) with respect to the weights (W):

$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial A}\frac{\partial A}{\partial Z}\frac{\partial Z}{\partial W} \tag{2}$$

3.1 Find partial derivative of linear output (Z) with respect to the weights (W):

$$\begin{aligned}\frac{\partial Z}{\partial W} &= \frac{\partial}{\partial W}(Wx+b)\\ &= x\\ &= A_{previous}\end{aligned}$$

Note: x is the previous activation function value which was the final value of the connecting nodes. Reiterating, the linear input to the output node is the value of the activation function from the connecting node, denoted as A_{prev}. Therefore, x = A_{prev}.

$$\begin{aligned}\therefore \frac{\partial L}{\partial W} &= \frac{\partial L}{\partial A}\frac{\partial A}{\partial Z}\frac{\partial Z}{\partial W}\\ &= (A-y)A_{previous}\end{aligned}$$

4. Finding partial derivative of loss function (L) with respect to the biases (b):

$$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial A}\frac{\partial A}{\partial Z}\frac{\partial Z}{\partial b} \tag{3}$$

3.1 Find partial derivative of linear output (Z) with respect to the weights (b):

$$\begin{aligned}\frac{\partial Z}{\partial b} &= \frac{\partial}{\partial b}(Wx+b)\\ &= 1\end{aligned}$$

$$\begin{aligned}\therefore \frac{\partial L}{\partial b} &= \frac{\partial L}{\partial A}\frac{\partial A}{\partial Z}\\ &= \frac{\partial L}{\partial Z}\\ &= A-y\end{aligned}$$

5. To backtrack through the entire network from the output node (loss function), we are still missing one partial derivatives which is $\frac{\partial L}{\partial A_{previous}}$ which is partial derivative of the loss function with respect to the previous activation value (previous node value).

$$\begin{aligned}\frac{\partial L}{\partial A_{previous}} &= \frac{\partial L}{\partial Z}\frac{\partial Z}{\partial A_{previous}}\\ &= \frac{\partial L}{\partial Z} * \frac{\partial}{\partial A_{previous}}(WA_{previous}+b)\\ &= \frac{\partial L}{\partial Z} * W\end{aligned}$$

Concluding:

That is all we need to derive in order to execute gradient descent for all of the nodes in the network. We are basically backtracking from the loss function (L) to find every parameters that affects it no matter how far back is it in the network. The magnitude and direction of each individual parameter that is affecting the loss function was gained by its respective partial derivative. Gradient descent can use this information to tune the parameters by its partial derivative with respect to the loss function. The global minimum of the loss function (given that the loss function goal is to minimize its value) can then be found.