

Back Propagation

Mladen Kolar and Rob McCulloch

1. Back Propagation

1. Back Propagation

Backpropagation is the basic algorithm for computing the gradient vector for a neural net model.

For a given (x, y) we need the partial derivatives of the ultimate loss with respect to all the weights and biases.

To evaluate the model we start at x and go *forward* through the layers, ending up at the output layer.

To evaluate the gradient we go *backward*, starting at the output layer and iterating back to the coefficients connecting x to the first hidden layer.

We will need a general notation for the neural net model.

Let's start by letting ℓ index the layers.

ℓ goes from 1 to L where $\ell = 1$ is the input layer (x) and L is the final output layer.

To keep things simple, we will have just one outcome with associated activation function g^L . For a single numeric outcome, g^L would typically be the identity function $I(x) = x$.

We will use the same activation function g at all the interior units (neurons).

Let p_ℓ be the number of neurons at layer ℓ .

Note that $p_1 = p$ where p is the dimension of x since that is the input layer.

Lots of Notation !!!!!:

$Z_k^{(\ell)}$: the Z value at the k^{th} unit of layer (ℓ) , $k = 1, 2, \dots, p_\ell$.

We have $Z_{\text{unit}}^{(\text{layer})}$. Similarly, we have $a_k^{(\ell)}$ with,

$$a_k^{(\ell)} = g(Z_k^{(\ell)}).$$

$w_{kj}^{(\ell)}$ = weight from $a_j^{(\ell)}$ (at layer ℓ) to $Z_k^{(\ell+1)}$ (at layer $(\ell + 1)$).

Think of w as $w_{kj}^{(\ell)} = w_{k \leftarrow j}^{(\ell)}$.

$b_k^{(\ell)}$ = intercept for $Z_k^{(\ell+1)}$ (at layer $(\ell + 1)$).

$$Z_k^{(\ell)} = b_k^{(\ell-1)} + \sum_{j=1}^{p^{(\ell-1)}} w_{kj}^{(\ell-1)} a_j^{(\ell-1)}, \quad k = 1, 2, \dots, p_\ell.$$

$$Z_k^{(\ell)} = b_k^{(\ell-1)} + \sum_{j=1}^{p^{(\ell-1)}} w_{kj}^{(\ell-1)} a_j^{(\ell-1)}, \quad k = 1, 2, \dots, p_\ell.$$

Matrix/Vector version:

$$Z^{(\ell)} = (Z_1^{(\ell)}, Z_2^{(\ell)}, \dots, Z_{p_\ell}^{(\ell)})'$$

$$a^{(\ell)} = g(Z^{(\ell)})$$

$$b^{(\ell)} = (b_1^{(\ell)}, b_2^{(\ell)}, \dots, b_{p^{(\ell+1)}}^{(\ell)})'$$

$$W^{(\ell)} = [w_{kj}^{(\ell)}], \quad p^{(\ell+1)} \times p_\ell$$

Then,

$$Z^{(\ell)} = b^{(\ell-1)} + W^{(\ell-1)} a^{(\ell-1)}$$

Begin:

$$a^{(1)} = x, \in R^p.$$

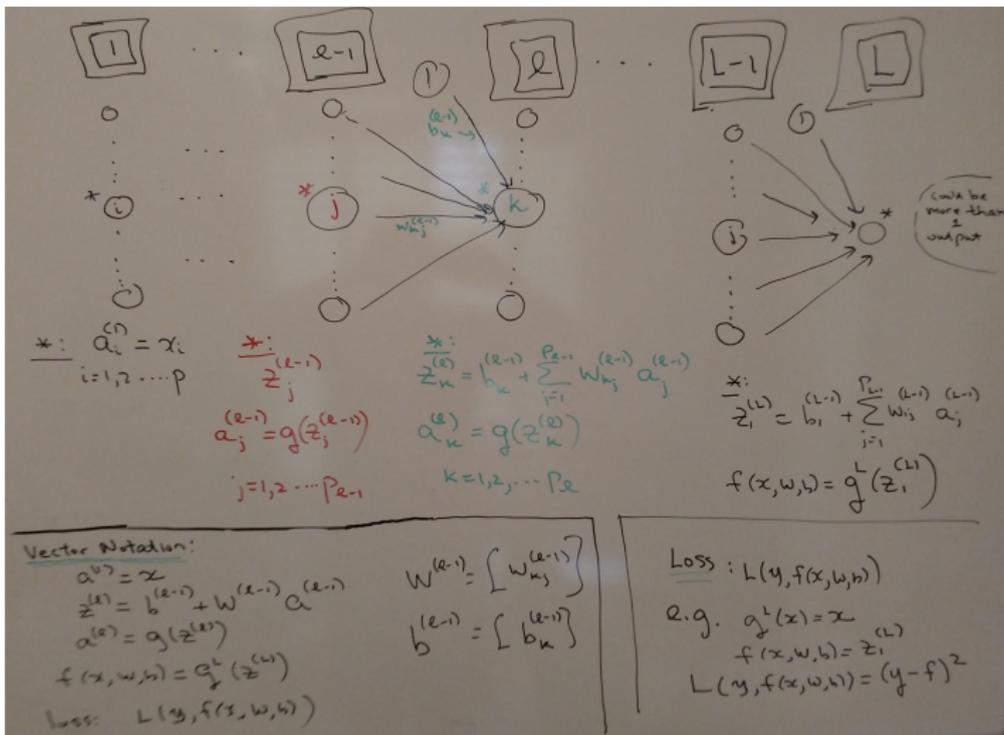
Iterate through the layers:

$$Z^{(\ell)} = b^{(\ell-1)} + W^{(\ell-1)}a^{(\ell-1)}, \quad a^{(\ell)} = g(Z^{(\ell)}).$$

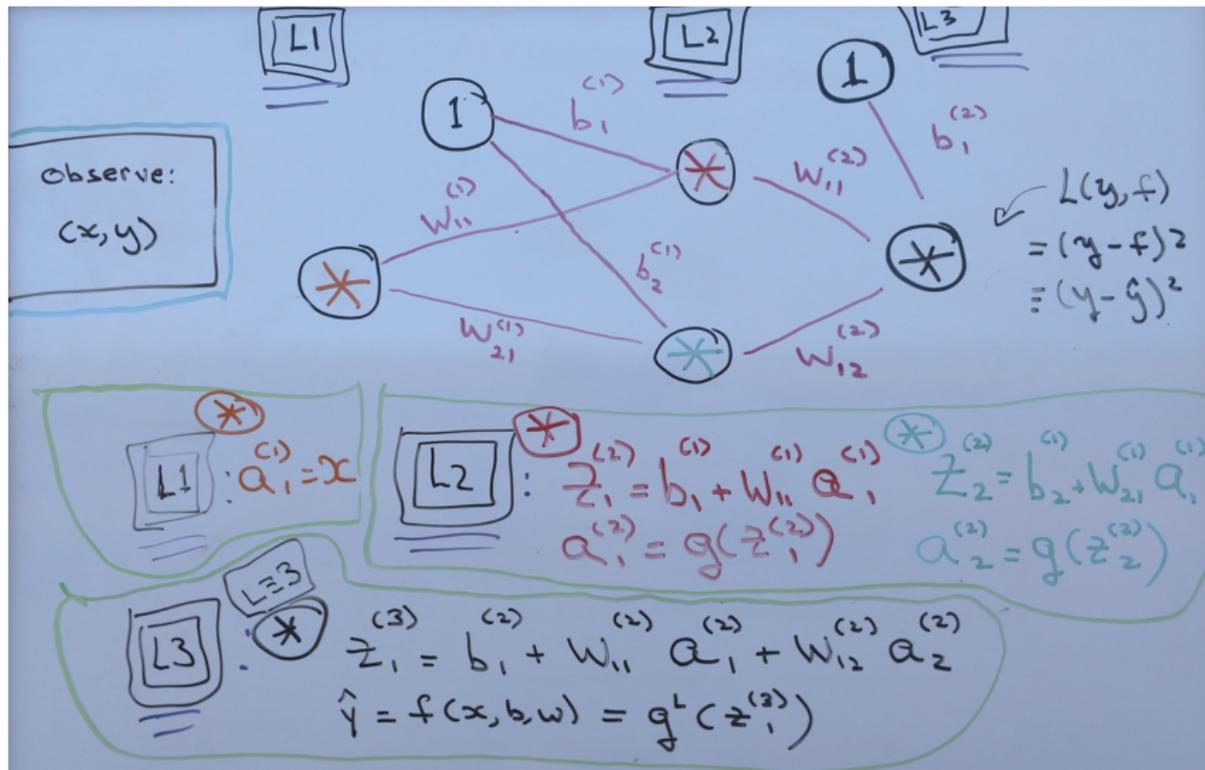
Final output layer and Loss:

$$f(x, W, b) = g^L(Z^L), \quad \text{Loss: } L(y, f(x, W, b)).$$

Here is the general model:



Simplest interesting case, just the model.



Note:

Backpropagation will work by computing:

$$\delta_i^{(\ell)} = \frac{\partial L}{\partial Z_i^{(\ell)}}$$

The differential effect of a change in $Z_i^{(\ell)}$ on *the ultimate loss* L .

Simplest interesting case, everything.

One x, one hidden layer with 2 neurons, one output.

$w_{ij}^{(l)}$:
Weight from neuron j of layer l to neuron i of layer $l+1$

$a_1^{(1)} = x$

$z_1^{(2)} = b_1^{(1)} + w_{11}^{(1)} a_1^{(1)}$
 $a_1^{(2)} = g(z_1^{(2)})$

$z_2^{(2)} = b_2^{(1)} + w_{21}^{(1)} a_1^{(1)}$
 $a_2^{(2)} = g(z_2^{(2)})$

Layer 1

$\frac{\partial L}{\partial w_{11}^{(1)}} = \frac{\partial L}{\partial z_1^{(2)}} \frac{\partial z_1^{(2)}}{\partial w_{11}^{(1)}} = \delta_1^{(2)} a_1^{(1)}$

$\delta_1^{(2)} = \frac{\partial L}{\partial z_1^{(2)}} = \frac{\partial L}{\partial z_1^{(2)}} \frac{\partial z_1^{(2)}}{\partial z_1^{(2)}} = \delta_1^{(2)}$

$\frac{\partial L}{\partial b_1^{(1)}} = \frac{\partial L}{\partial z_1^{(2)}} \frac{\partial z_1^{(2)}}{\partial b_1^{(1)}} = \delta_1^{(2)}$

$(p_1, p_2, p_3) = (1, 2, 1)$

$z_1^{(3)}$
 $= b_1^{(2)} + w_{11}^{(2)} a_1^{(2)} + w_{12}^{(2)} a_2^{(2)}$

$f(x, b, w) = g^L(z_1^{(3)}) \quad (L=3)$

$L(y, f) = (y - f)^2$

Chain Rule: $L \leftarrow f \leftarrow z_1^{(3)} \leftarrow w_{11}^{(2)}$

Layer 2 (Output)

$\frac{\partial L}{\partial w_{11}^{(2)}} = -2(y-f) (g')^{(2)}(z_1^{(3)}) a_1^{(2)} \equiv \delta_1^{(2)} a_1^{(2)}$

$\delta_1^{(2)} = \frac{\partial L}{\partial z_1^{(3)}} = -2(y-f) (g')^{(2)}(z_1^{(3)})$

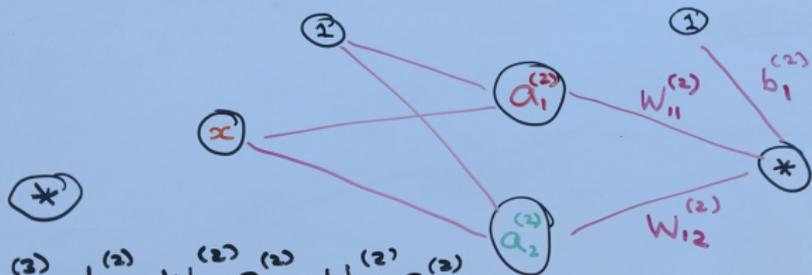
$\frac{\partial L}{\partial w_{21}^{(2)}} = \delta_2^{(2)} a_1^{(2)}$
 $\delta_2^{(2)} = \delta_1^{(2)} w_{12}^{(2)} g'(z_1^{(3)})$

$\frac{\partial L}{\partial b_1^{(2)}} = \delta_1^{(2)}$

$\delta_1^{(2)} = \frac{\partial L}{\partial z_1^{(3)}}$

9

Simplest interesting case, just $\frac{\partial L}{\partial w_{11}^{(2)}}$.



$$z_1^{(3)} = b_1 + w_{11}^{(2)} a_1 + w_{12}^{(2)} a_2$$

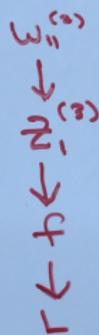
$$f(x, b, w) = g^L(z_1^{(3)})$$

$$L(y, f) = (y - f)^2$$

$$\frac{\partial L}{\partial w_{11}^{(2)}} = \underbrace{[-2(y-f)]}_{\frac{\partial L}{\partial f}} \underbrace{[(g^L)'(z_1^{(3)})]}_{\frac{\partial f}{\partial z_1^{(3)}}} \underbrace{[a_1^{(2)}]}_{\frac{\partial z_1^{(3)}}{\partial w_{11}^{(2)}}} \equiv \delta_1^{(3)} a_1^{(2)}$$

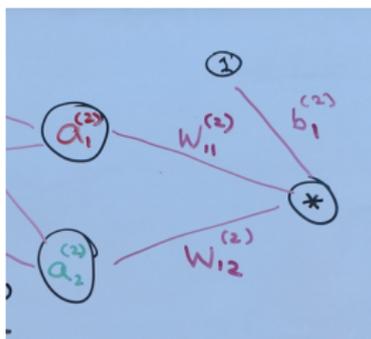
$$\frac{\partial L}{\partial w_{11}^{(2)}} = ?$$

Chain Rule !!



$$\delta_i^{(2)} = \frac{\partial L}{\partial z_i^{(2)}}$$

Same thing, just using δ :



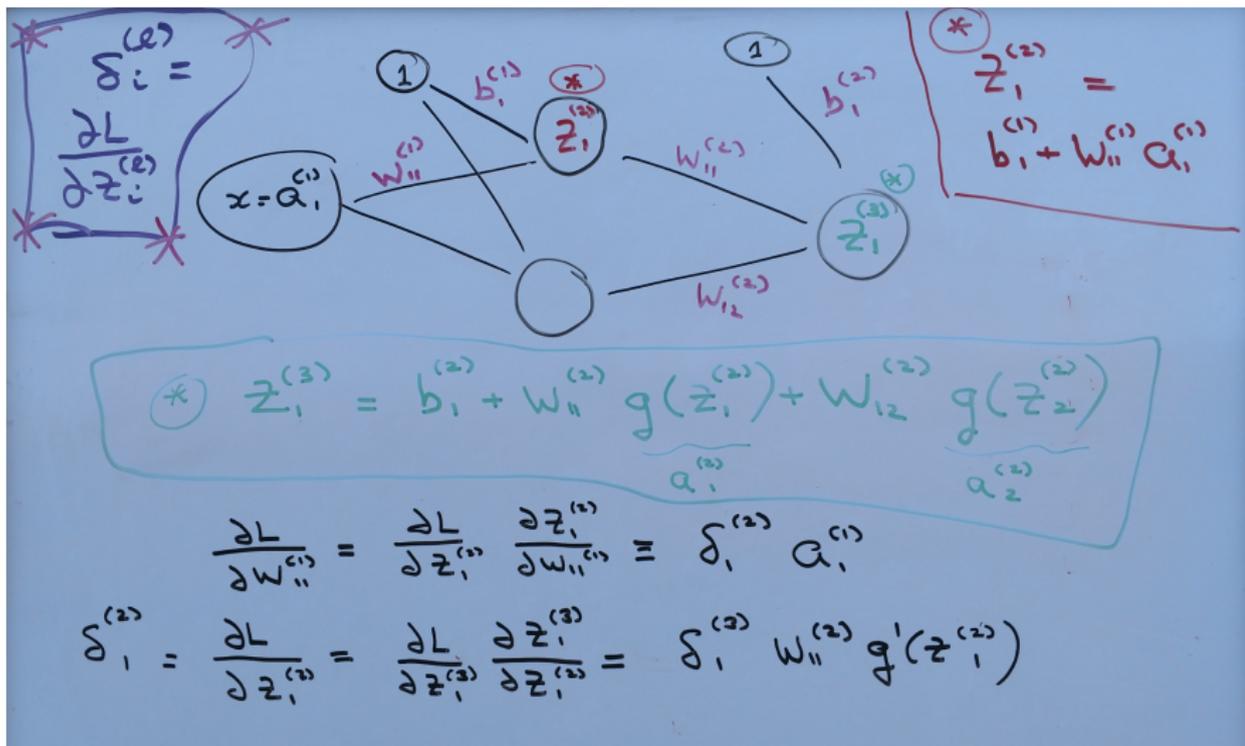
$$Z_1^{(3)} = b_1^{(2)} + w_{11}^{(2)} a_1^{(2)} + w_{12}^{(2)} a_2^{(2)}.$$

$$\frac{\partial L}{\partial w_{11}^{(2)}} = \frac{\partial L}{\partial Z_1^{(3)}} \frac{\partial Z_1^{(3)}}{\partial w_{11}^{(2)}} = \delta_1^{(3)} a_1^{(2)}.$$

Similarly,

$$\frac{\partial L}{\partial w_{12}^{(2)}} = \delta_1^{(3)} a_2^{(2)}, \quad \frac{\partial L}{\partial b_1^{(2)}} = \delta_1^{(3)}.$$

Simplest interesting case, just $\frac{\partial L}{\partial w_{11}^{(1)}}$.



$$Z_2^{(2)} = b_2^{(1)} + w_{21}^{(1)} a_1^{(1)}.$$

Similarly,

$$\delta_2^{(2)} = \frac{\partial L}{\partial Z_2^{(2)}} = \frac{\partial L}{\partial Z_1^{(3)}} \frac{\partial Z_1^{(3)}}{\partial Z_2^{(2)}} = \delta_1^{(3)} w_{12}^{(2)} g'(Z_2^{(2)}).$$

$$\frac{\partial L}{\partial w_{21}^{(1)}} = \frac{\partial L}{\partial Z_2^{(2)}} \frac{\partial Z_2^{(2)}}{\partial w_{21}^{(1)}} = \delta_2^{(2)} a_1^{(1)}.$$

And,

$$\frac{\partial L}{\partial b_1^{(1)}} = \delta_1^{(2)}, \quad \frac{\partial L}{\partial b_2^{(1)}} = \delta_2^{(2)}.$$

How it Works

Key Quantities:

$\delta_i^{(l)}$: effect on loss of a change
in $z_i^{(l)}$

Iterate

① initialize by computing $\delta_i^{(L)}$

② iterate $(l+1) \rightarrow (l)$ getting
 $\delta_j^{(l)}$ from $\delta_i^{(l+1)}$ — "backprop"

③ Get partials for layer l
parameters $b^{(l)}, W^{(l)}$ from $\delta_i^{(l+1)}$

Here are the partial derivatives associated with the parameters at layer $L - 1$.

This will also initialize the back-propagation algorithm for computing the partials for parameters associated with the other layers.

The diagram shows a neuron in layer L (indicated by a box labeled L) receiving inputs from a previous layer $L-1$ (indicated by a box labeled $L-1$). The inputs are $a_j^{(L-1)}$ and a bias $b_1^{(L)}$. The neuron's output is $z_1^{(L)}$. The error at this neuron is $\delta_1^{(L)}$.

$$z_1^{(L)} = b_1^{(L)} + \sum_{j=1}^{p_{L-1}} w_{1j}^{(L)} a_j^{(L-1)}$$

$$f = g^L(z_1^{(L)})$$

$$L = (y - f)^2$$

$$\frac{\partial L}{\partial w_{1j}^{(L)}} = -2(y - f)(g^L)'(z_1^{(L)}) a_j^{(L-1)} \equiv \delta_1^{(L)} a_j^{(L-1)}$$

$$\frac{\partial L}{\partial b_1^{(L)}} = \delta_1^{(L)} = \frac{\partial L}{\partial z_1^{(L)}}$$

$$\frac{\partial L}{\partial w^{(L-1)}} = \delta_1^{(L)} \odot a^{(L-1)} \quad ; \quad \frac{\partial L}{\partial b_1^{(L-1)}} = \delta_1^{(L)}$$

Latex for previous hand written slide.

$$Z_1^{(L)} = b_1^{(L-1)} + \sum_{j=1}^{p_{L-1}} w_{1j}^{(L-1)} a_j^{(L-1)}.$$

$$f(x, b, w) = g^L(Z_1^{(L)}), \quad L(y, f) = (y - f)^2$$

$$\delta_1^{(L)} = \frac{\partial L}{\partial Z_1^{(L)}} = \frac{\partial L}{\partial f} \frac{\partial f}{\partial Z_1^{(L)}} = [-2(y - f)] \left[(g^L)'(Z_1^{(L)}) \right].$$

$$\frac{\partial L}{\partial w_{1j}^{(L-1)}} = \frac{\partial L}{\partial Z_1^{(L)}} \frac{\partial Z_1^{(L)}}{\partial w_{1j}^{(L-1)}} = \delta_1^{(L)} a_j^{(L-1)}.$$

$$\frac{\partial L}{\partial b_1^{(L-1)}} = \delta_1^{(L)}$$

Multivariate version of chain rule.

$$f(x) = \begin{pmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_p(x) \end{pmatrix} \quad f: \mathbb{R} \rightarrow \mathbb{R}^p$$
$$g: \mathbb{R}^p \rightarrow \mathbb{R}$$

$$h(x) = g(f_1(x), f_2(x), \dots, f_p(x))$$

$$x \in \mathbb{R} \rightarrow \begin{pmatrix} f_1(x) = y_1 \\ \vdots \\ f_p(x) = y_p \end{pmatrix} \in \mathbb{R}^p \rightarrow z \in \mathbb{R} = g(y)$$

$$h = g \circ f$$

$$h' = \nabla g \cdot f' = \sum \frac{\partial g}{\partial y_i} \frac{dy_i}{dx}$$

Here is the iteration for computing the key $\delta_j^{(\ell)}$ quantities.

The diagram shows two layers, ℓ and $\ell+1$, each enclosed in a rounded rectangle. Below them, a vertical column of nodes in layer ℓ is shown, with a central node labeled i . Arrows point from node i to a vertical column of nodes in layer $\ell+1$, with a central node labeled k .

The key equation is:

$$\text{key} = \delta_j^{(\ell)} = \frac{\partial L^{(\ell)}}{\partial z_j^{(\ell)}} \quad j=1,2,\dots,p_\ell$$

The next equation shows the computation of the next layer's net input:

$$z_k^{(\ell+1)} = b_k^{(\ell+1)} + \sum_{i=1}^{p_\ell} w_{ki}^{(\ell)} a_i^{(\ell)} = b_k^{(\ell+1)} + \sum_{i=1}^{p_\ell} w_{ki}^{(\ell)} g(z_i^{(\ell)})$$

The error backpropagation equation is:

$$\delta_i^{(\ell)} = \frac{\partial L^{(\ell)}}{\partial z_i^{(\ell)}} = \sum_{k=1}^{p_{\ell+1}} \frac{\partial L}{\partial z_k^{(\ell+1)}} \frac{\partial z_k^{(\ell+1)}}{\partial z_i^{(\ell)}} = \sum_k \delta_k^{(\ell+1)} w_{ki}^{(\ell)} g'(z_i^{(\ell)}) = g'(z_i^{(\ell)}) \sum_k \delta_k^{(\ell+1)} w_{ki}^{(\ell)}$$

The final boxed equation is:

$$\delta^{(\ell)} = g'(z^{(\ell)}) \odot \{W^{(\ell)}\}^T \delta^{(\ell+1)}$$

$$\begin{aligned}
Z_k^{(\ell+1)} &= b_k^\ell + \sum_{i=1}^{p_\ell} w_{ki}^{(\ell)} a_i^{(\ell)} \\
&= b_k^\ell + \sum_{i=1}^{p_\ell} w_{ki}^{(\ell)} g(Z_i^{(\ell)})
\end{aligned}$$

$$\begin{aligned}
\delta_i^{(\ell)} = \frac{\partial L}{\partial Z_i^{(\ell)}} &= \sum_{k=1}^{p_{\ell+1}} \frac{\partial L}{\partial Z_k^{(\ell+1)}} \frac{\partial Z_k^{(\ell+1)}}{\partial Z_i^{(\ell)}} \\
&= \sum_{k=1}^{p_{\ell+1}} [\delta_k^{(\ell+1)}] [w_{ki}^{(\ell)} g'(Z_i^{(\ell)})] \\
&= g'(Z_i^{(\ell)}) \sum_{k=1}^{p_{\ell+1}} [\delta_k^{(\ell+1)}] [w_{ki}^{(\ell)}]
\end{aligned}$$

$$\delta^{(\ell)} = g'(Z^{(\ell)}) \odot \left[[W^{(\ell)}]' \delta^{(\ell+1)} \right]$$

where

$$a \odot b = (a_i b_i)$$

is *elementwise* multiplication, and

$g'(Z^{(\ell)})$ means apply $g' : R \rightarrow R$ to each element of $Z^{(\ell)}$.

Note:

$$Z^{(\ell)} \in R^{p_\ell}. \quad g'(Z^{(\ell)}) \in R^{p_\ell}. \quad \delta^{(\ell)} \in R^{p_\ell}.$$

$$\delta^{(\ell+1)} \in R^{(p_{\ell+1})}.$$

$$W^{(\ell)} \text{ is } p_{(\ell+1)} \times p_\ell.$$

Here are the partial derivatives in terms of the $\delta_j^{(l)}$.

The diagram shows two layers of nodes. The top layer has nodes labeled l and $l+1$, each enclosed in a box. Below them are nodes i and k . A horizontal arrow labeled $w_{ki}^{(l)}$ connects node i to node k . A diagonal arrow labeled $b_k^{(l)}$ points to node k . Vertical ellipses indicate other nodes in the layers.

$$z_n^{(l+1)} = b_n^{(l)} + \sum_i w_{ki}^{(l)} a_i^{(l)}$$

$$\frac{\partial L}{\partial w_{ki}^{(l)}} = \frac{\partial L}{z_k^{(l+1)}} \frac{\partial z_k^{(l+1)}}{\partial w_{ki}^{(l)}}$$

$$= \delta_k^{(l+1)} a_i^{(l)}$$

$$\frac{\partial L}{\partial b_k^{(l)}} = \frac{\partial L}{z_k^{(l+1)}} \frac{\partial z_k^{(l+1)}}{\partial b_k^{(l)}} = \delta_k^{(l+1)}$$

$$\frac{\partial L}{\partial w^{(l)}} = \left[\delta^{(l+1)} \right] \left[a^{(l)} \right]^T$$

$$\frac{\partial L}{\partial b^{(l)}} = \delta^{(l+1)}$$

$$Z_k^{(\ell+1)} = b_k^\ell + \sum_{i=1}^{p_\ell} w_{ki}^{(\ell)} a_i^{(\ell)}.$$

$$\begin{aligned} \frac{\partial L}{\partial w_{ki}^{(\ell)}} &= \frac{\partial L}{\partial Z_k^{(\ell+1)}} \frac{\partial Z_k^{(\ell+1)}}{\partial w_{ki}^{(\ell)}} \\ &= \delta_k^{(\ell+1)} a_i^{(\ell)} \end{aligned}$$

$$\frac{\partial L}{\partial b_k^{(\ell)}} = \delta_k^{(\ell+1)}$$

$$\frac{\partial L}{\partial W^{(\ell)}} = \left[\frac{\partial L}{\partial w_{ki}^{(\ell)}} \right] = [\delta^{(\ell+1)}] [\mathbf{a}^{(\ell)}]'$$

$$\frac{\partial L}{\partial b^{(\ell)}} = \left[\frac{\partial L}{\partial b_k^{(\ell)}} \right] = \delta^{(\ell+1)}$$

Neural Nets in a Nutshell

Model and Loss

$$a^{(1)} = x; \quad z^{(2)} = b^{(2)} + W^{(2,1)} a^{(1)}; \quad a^{(2)} = g^{(2)}(z^{(2)})$$

$$f(x, b, w) = a^{(2)}; \quad \min_{b, w} \frac{1}{n} \sum_{i=1}^n L(y_i; f(x_i, b, w))$$

Gradient Computation (Backprop)

$$- \delta^{(1)} = \frac{\partial L}{\partial f} (g^{(1)})'(z^{(1)})$$

$$- \delta^{(2)} = (g^{(2)})'(z^{(2)}) \odot [W^{(2,1)}]^T \delta^{(1)}$$

$$- \frac{\partial L}{\partial w^{(2)}} = [\delta^{(2)}] [a^{(1)}]^T \quad \frac{\partial L}{\partial b^{(2)}} = \delta^{(2)}$$

*

- Er schedule
- Nesterov Momentum
- L¹, L² regularization
- Dropout
- ⋮

SGD: Stochastic Gradient Descent

Epochs: $k=1, 2, \dots, K$ (pass through data)

Minibatches: $\{x_i^b, y_i^b\} \quad \begin{matrix} i=1, 2, \dots, m \\ b=1, 2, \dots, B \end{matrix}$

ϵ_k : learning rate

$\Theta = (b, w)$

for $k=1, 2, \dots, K$

for $b=1, 2, \dots, B$

$$\Theta \rightarrow \Theta - \epsilon_k \frac{1}{m} \sum_{i=1}^m \nabla L(x_i^b, y_i^b; \Theta) \quad *$$