1 Fully connected neural net

1.1 Forward prop

For layer $1 \leq \ell \leq L$, we have forward prop (assume single example):

$$z^{[\ell]} = W^{[\ell]} a^{[\ell-1]} + b^{[\ell]}$$
$$a^{[\ell]} = \phi^{[\ell]}(z^{[\ell]})$$

where $z^{[\ell]}$ is the pre-activation for layer ℓ, and $a^{[\ell]}$ is the activation for layer ℓ. Note we use $a^{[0]} = x$ (ie inputs) for consistency.

Dimensions:

- $W^{[\ell]} \in \mathbb{R}^{n^{[\ell]} \times n^{[\ell-1]}}$
- $z^{[\ell]}, a^{[\ell]}, b^{[\ell]} \in \mathbb{R}^{n^{[\ell]}}$

Note: $z = Wx + b$ is not "the only correct choice" that has to be made. It's chosen because it allows for simple linear interactions between features; the ith neuron in layer ℓ mixes all neurons in layer $\ell - 1$ using the ith row of W. Then, $+b$ lets each neuron shift in a constant direction which isn't achievable by the existing linear map Wx. Then, nonlinear activation functions allow features to interact with each other nonlinearly.

- For example, $\text{ReLU}(x_1 + x_2 - 1)$ makes changing x_1 only matter when $x_1 + x_2 > 1$. The influence of x_1 depends on x_2.

1.2 Backprop

Now, for backward prop, for layer $1 \leq \ell < L$, we have $\dfrac{\partial J}{\partial a^{[\ell]}}$ calculated for us by layer $\ell + 1$ (base case $\ell = L$ grabs it directly from the model's eval loss (for example, $J = \dfrac{1}{2}(a^{[L]} - y)^2$ gives $\dfrac{\partial J}{\partial a^{[L]}} = a^{[L]} - y$)). Assume single example again:

$$\frac{\partial J}{\partial z^{[\ell]}} = \frac{\partial J}{\partial a^{[\ell]}} \odot \frac{\partial a^{[\ell]}}{\partial z^{[\ell]}} = \boxed{\frac{\partial J}{\partial a^{[\ell]}} \odot \phi'^{[\ell]}\left(z^{[\ell]}\right)}$$

For $\dfrac{\partial J}{\partial W^{[\ell]}}$, chain rule gets a bit difficult; $\dfrac{\partial J}{\partial z^{[\ell]}} \in \mathbb{R}^{n^{[\ell]}}$, but $\dfrac{\partial z^{[\ell]}}{\partial W^{[\ell]}} \in \mathbb{R}^{n^{[\ell]} \times n^{[\ell]} \times n^{[\ell-1]}}$. We can do this by recovering the matrix form from individual components. Given that $z_p = W_p a^{[\ell-1]} + b_p$,

$$\frac{\partial J}{\partial W_{pq}^{[\ell]}} = \frac{\partial J}{\partial z_p^{[\ell]}} \frac{\partial z_p^{[\ell]}}{\partial W_{pq}^{[\ell]}} = \frac{\partial J}{\partial z_p^{[\ell]}} \cdot a_q^{[\ell-1]}$$

So for row p, we use the same $\dfrac{\partial J}{\partial z_p^{[\ell]}}$ value and multiply against a row of $a^{[\ell-1]}$. By inspection, recover that:

$$\frac{\partial J}{\partial W^{[\ell]}} = \boxed{\frac{\partial J}{\partial z^{[\ell]}} \left(a^{[\ell-1]}\right)^T}$$

For $\dfrac{\partial J}{\partial b^{[\ell]}}$, it gets easy:

$$\frac{\partial J}{\partial b^{[\ell]}} = \frac{\partial J}{\partial z^{[\ell]}} \frac{\partial z^{[\ell]}}{\partial b^{[\ell]}} = \frac{\partial J}{\partial z^{[\ell]}} \cdot 1 = \boxed{\frac{\partial J}{\partial z^{[\ell]}}}$$

Finally, propagate $\dfrac{\partial J}{\partial a^{[\ell-1]}}$. For a fixed q,

$$\frac{\partial J}{\partial a_q^{[\ell-1]}} = \sum_p \frac{\partial J}{\partial z_p^{[\ell]}} \frac{\partial z_p^{[\ell]}}{\partial a_q^{[\ell-1]}} = \sum_p \frac{\partial J}{\partial z_p^{[\ell]}} W_{pq}^{[\ell]}$$

So for the qth row (or entry, technically) of $\dfrac{\partial J}{\partial a^{[\ell-1]}}$, take column q in $W^{[\ell]}$, and fill the $\dfrac{\partial J}{\partial a_q^{[\ell-1]}}$ value with $\dfrac{\partial J}{\partial z^{[\ell]}} \cdot W_{:,q}^{[\ell]}$ (aka dot product against column q of W). This is the same as $(W_{:,q}^{[\ell]})^T \dfrac{\partial J}{\partial z^{[\ell]}}$; we essentially make $W_{:,q}^{[\ell]}$ a row vector and treat it as a $1 \times n^{[\ell]}$ matrix to achieve the same dot product via "matmul". Parallelize this across the rows of $\dfrac{\partial J}{\partial a^{[\ell-1]}}$ and get

$$\frac{\partial J}{\partial a^{[\ell-1]}} = \boxed{(W^{[\ell]})^T \frac{\partial J}{\partial z^{[\ell]}}}$$

Of course, the point is mainly that we get $\frac{\partial J}{\partial W^{[\ell]}}$ and $\frac{\partial J}{\partial b^{[\ell]}}$; everything else is to serve this. Update parameters for every layer:

$$W^{[\ell]} := W^{[\ell]} - \alpha \frac{\partial J}{\partial W^{[\ell]}}$$
$$b^{[\ell]} := b^{[\ell]} - \alpha \frac{\partial J}{\partial b^{[\ell]}}$$

1.3 Batching

Adding functionality to feed in multiple examples to the network instead of a single one is nearly identical for everything. We stack examples as columns; previously, we had individual column vectors for a single example, so we simply stack them together and make a matrix this time. Since the network mostly processes each example independently, forward prop and backprop essentially parallelize across examples independently pretty easily. However, some elements like W and b interact with all examples and thus need slightly different backprop:

$$\frac{\partial J}{\partial W^{[\ell]}} = \frac{\partial J}{\partial Z^{[\ell]}} \left(A^{[\ell-1]}\right)^T$$
$$\frac{\partial J}{\partial b^{[\ell]}} = \sum_i \frac{\partial J}{\partial Z^{[\ell]}_{:,i}}$$

Also, make sure J has some kind of normalization along number of examples so numbers don't change scale over different batch sizes.