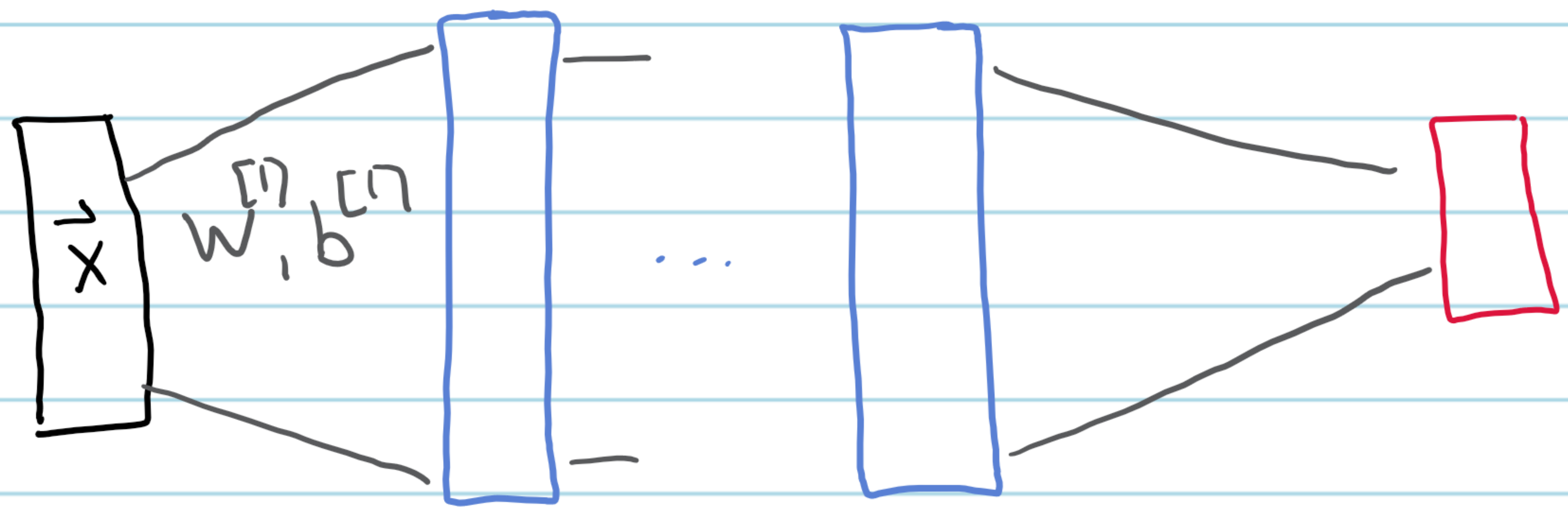
Feed Forward

Activation function

$$a^{[l]} = f_a^{[l]}\left(Z^{[l]}\right) = f_a^{[l]}\left(W^{[l]} \cdot a^{[l]} + b^{[l]}\right)$$

↳ See the code.



Feed Forward
⟶ Take $a^{[l]}$ and make $a^{[l+1]}$.

For back propagation, we get to the end and check how far the outputs are from the target values (Y).

Then we need to go back and tune the params.

$$W^{[l]} = W^{[l]} - \eta \frac{d\mathcal{L}}{dW^{[l]}} \qquad \mathcal{L} \to \text{loss}$$

$$b^{[l]} = b^{[l]} - \eta \frac{d\mathcal{L}}{db^{[l]}}$$

So, we need to calculate $\boxed{\frac{d\mathcal{L}}{dW^{[l]}}} \to dW^{[l]}$ & $\boxed{\frac{d\mathcal{L}}{db^{[l]}}} \to db^{[l]}$.

We need to calculate all of these derivatives.

Training : Gradient Descent

Let's say that we want to train an LR to fit $\{X, Y\}$.

We need a loss function. We take Cross-entropy :

$$L_{ce}(Y, \tilde{Y}) = -\sum_i Y_i \log \tilde{Y}_i + (1-Y_i)\log(1-\tilde{Y}_i).$$

For LR, we have $\tilde{Y} = \sigma(W.X + b)$ $\qquad W \to W^T$ (for simplicity)

Each step of the GD:

$$W \to W - \alpha \frac{\partial L_{ce}}{\partial W}$$

$$b \to W - \alpha \frac{\partial L_{ce}}{\partial W}$$

$$\frac{\partial L}{\partial W_i} = \frac{\partial L}{\partial \tilde{Y}} \frac{\partial \tilde{Y}}{\partial w_i} = \underset{\text{(Chain rule)}}{\cdots} = \left(\frac{-Y}{\tilde{Y}} + \frac{1-Y}{1-\tilde{Y}}\right) \underbrace{\frac{\partial \tilde{Y}}{\partial W_i}}_{\frac{\partial \tilde{Y}}{\partial z} \frac{\partial z}{\partial W}}$$

$$\frac{\partial \tilde{Y}}{\partial z} = \frac{\partial \sigma(z)}{\partial z} = \sigma(z)(1-\sigma(z)) = \tilde{Y}(1-\tilde{Y}) \qquad (n_\ell, 1)$$

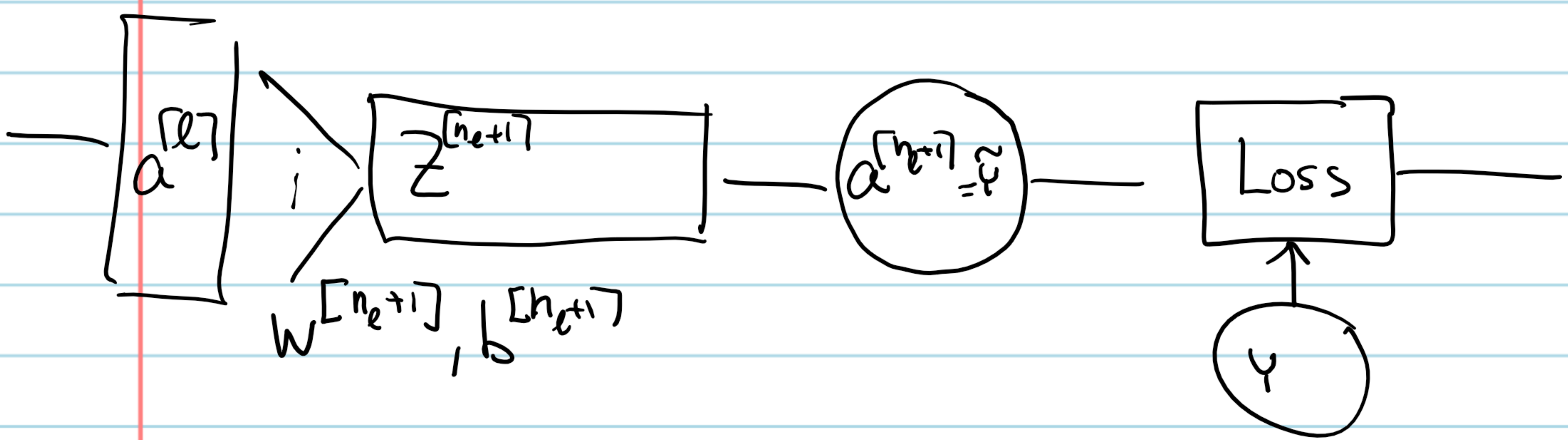
$$\triangleright \; \frac{\partial L}{\partial W_i} = \underbrace{\left(\frac{-Y}{\tilde{Y}} + \frac{1-Y}{1-\tilde{Y}}\right)\tilde{Y}(1-\tilde{Y})}_{\underbrace{\left(-Y(1-\tilde{Y}) + \tilde{Y}(1-Y)\right)}_{(\tilde{Y}-Y)}} (X_i) \qquad (1, n_s) \qquad (n_\ell, n_s)$$

$$\left(-Y(1-\tilde{Y}) + \tilde{Y}(1-Y)\right) X_i = (\tilde{Y} - Y) X_i$$

$$\frac{\partial L}{\partial b} = (\tilde{Y} - Y)$$

Backpropagation

① Start with the last layer:



$$dZ^{[n_e+1]} \rightarrow \frac{dL}{dZ^{[n_e+1]}}$$

$$dW^{[n_e+1]} = \frac{dL}{dW^{[n_e+1]}} = \underbrace{\boxed{\frac{dL}{da^{[n_e+1]}}}\boxed{\frac{da^{[n_e+1]}}{dZ^{[n_e+1]}}}}_{dZ^{[n_e+1]}}\boxed{\frac{dZ^{[n_e+1]}}{dW^{[n_e+1]}}}$$

This should be part of the preprocessing.

- Derivative of the loss function
- Derivative of the activation of the last function.

$$\frac{dZ^{[n_e+1]}}{dW^{[n_e+1]}} = a^{[n_e]}$$

In general $\displaystyle\frac{dZ^{[l]}}{dW^{[l]}} = a^{[l]}$

For the next layer, $Z^{[n_e+1]} = \left(W^{[n_e+1]} \cdot a^{[n_e]} + b^{[n_e+1]}\right)$

$$dW^{[n_e]} = \frac{dL}{dW^{[n_e]}} = \underbrace{\boxed{\frac{dL}{dZ^{[n_e+1]}}}\boxed{\frac{dZ^{[n_e+1]}}{da^{[n_e]}}}\boxed{\frac{da^{[n_e]}}{dZ^{[n_e]}}}}_{dZ^{[n_e]} \rightarrow \frac{dL}{dZ^{[n_e]}}}\boxed{\frac{dZ^{[n_e]}}{dW^{[n_e]}}}$$

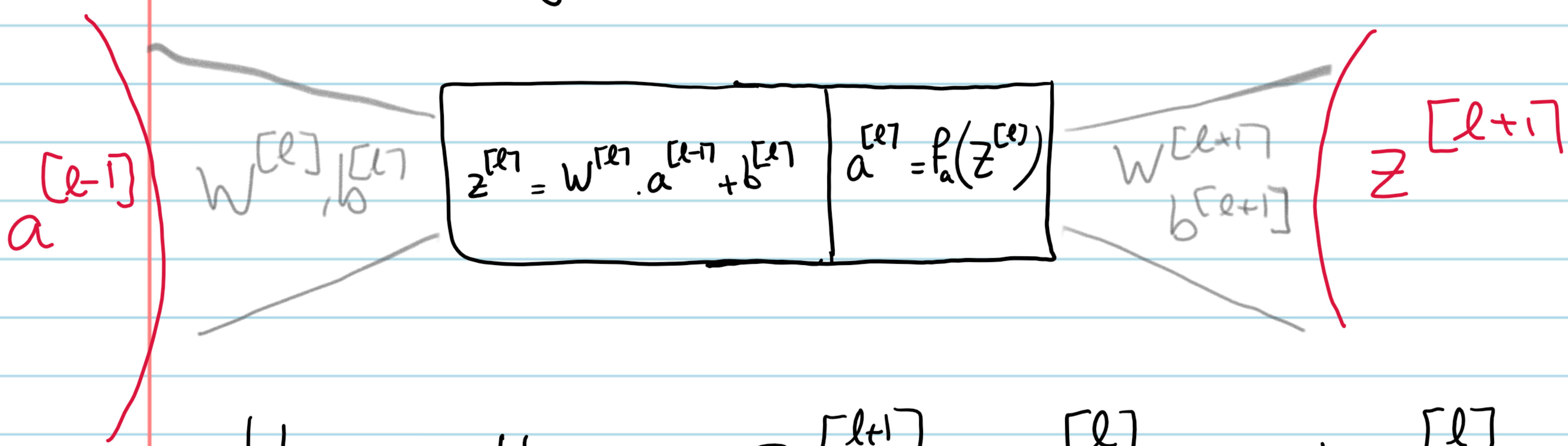
From last layer. $W^{[n_e+1]}$ derivative of activation function $a^{[n_e-1]}$

Similarly:

$$db^{[n_e+1]} = dZ^{[n_e+1]}$$

$$db^{[n_e]} = dZ^{[n_e]}$$

It is easy to see that in general:



$$\frac{dL}{dW^{[l]}} = \underbrace{\frac{dL}{dZ^{[l+1]}} \frac{dZ^{[l+1]}}{da^{[l]}} \frac{da^{[l]}}{dZ^{[l]}}}_{\frac{dL}{dZ^{[l]}}} \frac{dZ^{[l]}}{dW^{[l]}}$$

So, it is useful to keep $dZ^{[l]}$ as all the derivatives can be expressed in terms of them.

$$dW^{[l]} = \frac{dL}{dW^{[l]}} = \frac{dL}{dZ^{[l]}} \frac{dZ^{[l]}}{dW^{[l]}} = \frac{dL}{dZ^{[l]}} a^{[l-1]} = dZ^{[l]} a^{[l-1]}$$

(n_l, n_s)

(n_{l-1}, n_s)

$$db^{[l]} = \frac{dL}{db^{[l]}} = \frac{dL}{dZ^{[l]}} \underbrace{\frac{dZ^{[l]}}{db^{l}}}_{1} = dZ^{[l]}$$

$$dZ^{l} = \underbrace{\frac{dL}{dZ^{[l+1]}}}_{\text{From previous layer.}} \underbrace{\frac{dZ^{[l+1]}}{da^{[l]}}}_{W^{[l+1]}} \underbrace{\frac{da^{[l]}}{dZ^{[l]}}}_{\text{Activation function}}$$

$$= dZ^{[l+1]} W^{[l+1]} (f_a^{[l]})'$$

Tensor dimensions

We can do the backpropagation for each variable $dW^{[l]}_{ij}$ seperately.
But this is not efficient. To vectorize the calculation, we need to keep the matrices and turn the multipications to matrix products.
For this, we need to match the dim of the tensors.

$$dZ^{[l]} = \underbrace{dZ^{[l+1]}}_{(n_l, n_s)} \underbrace{W^{[l+1]}}_{(n_{l+1}, n_l)} \left(f^{[l]}_a\right)'$$

$$dZ^{[l]} = \left(W^{[l+1]} \cdot dZ^{[l+1]}\right) \left(f^{[l]}_a\right)' \left(Z^{[l]}\right)$$

$$(n_{l+1}, n_l) \cdot (n_l, n_s) * (n_l, n_s)$$

$$(n_{l+1}, n_s)$$

normal multipication

$$dW^{[l]} = \underset{(n_l, n_s)}{dZ^{[l]}} \quad \underset{(n_{l-1}, n_s)}{a^{[l-1]}} \rightarrow dZ^{[l]} = dZ^{[l]} \cdot \left(a^{[l-1]}\right)^T$$

$$(n_l, n_s) \cdot (n_s, n_{l-1})$$

$$(n_l, n_{l-1})$$

$$\underset{(n_l, 1)}{db^{[l]}} = \underset{(n_l, n_s)}{dZ^{[l]}} \rightarrow \sum_j \left(dZ^{[l]}\right)_{ij} \rightarrow (n_l, 1)$$