Points:

if weight matrix W is initialized too large, the output of the matrix multiply too could probably have a very large range, which in turn will make all the outputs in the vector z almost binary either '1' or '0'.

1. sigmoid, $\sigma(z) = \frac{1}{1+e^{-z}}$.

2. local gradient, is a derivative that measures how much the output of the sigmoid function changes with respect to its input z.

$$\sigma'(z) = \frac{d}{dz}\left(\frac{1}{1+e^{-z}}\right)$$

$f(g(z))$; where $f(u) = \frac{1}{u}$, $g(z) = 1+e^{-z}$

$f'(u) = -\frac{1}{u^2}$ $\quad g'(z) = \frac{d}{dz}(1+e^{-z}) = -e^{-z}$

$$\frac{d\sigma(z)}{dz} = f'(g(z)) \cdot g'(z) = -\frac{1}{(1+e^{-z})^2} \cdot (-e^{-z})$$

$$\frac{d\sigma(z)}{dz} = \frac{e^{-z}}{(1+e^{-z})^2}$$

we know, $\sigma(z) = \dfrac{1}{1+e^{-z}}$

re written as, Subtract the sigmoid from 1.

$$1-\sigma(z) = 1 - \frac{1}{1+e^{-z}}$$

express 1 with common denominator, $1 = \dfrac{1+e^{-z}}{1+e^{-z}}$

$$1-\sigma(z) = \frac{1+e^{-z}}{1+e^{-z}} - \frac{1}{1+e^{-z}}$$

$$1-\sigma(z) = \frac{(1+e^{-z})-1}{(1+e^{-z})} = \frac{e^{-z}}{(1+e^{-z})(1+e^{-z})} \times (1-\sigma(z))$$

$$\Downarrow$$

$$= \boxed{\frac{d\sigma(z)}{dz} = \sigma(z) \times (1-\sigma(z))}$$

if $\sigma(z)$ is close to 1 or 0, the term $\sigma(z) \times (1-\sigma(z))$ becomes very small approaching to zero.

In;

→ Back propagation, gradients are propagated backward through the network by multiplying them together (as per the chain rule)

→ If local gradient is small, when you multiply it with other gradients in the backward pass, the overall gradient becomes effectively zero.

→ This causes the entire backward pass to have nearly zero gradients, preventing the network from learning.

ReLU (Non-linear) activation Function.

$$\text{ReLU}(z) = \max(0, z) \quad \begin{cases} z & \text{if } z > 0 \\ 0 & \text{if } z \leq 0 \end{cases}$$

```
z = np.maximum(0, np.dot(W, x))  # forward pass
```

min max

```
dW = np.outer(z > 0, x)  # representing backward pass: local gradient for W.
```

(how much each weight should be adjusted)

which elements in 'z' are greater than zero that pass through ReLU

Calculates dot product between weight matrix 'W' and input vector 'x'.

→ if any negative value it return '0'.

→ np.outer (z>0, x) : Outer product (z > 0) (a vector of 'True' or 'False') and the input vector x.

The outer product is used to determine how much each weight 'W' should be adjusted based on the input 'x' and the result 'z'.

$\boxed{\text{Chain Rule:}}$ $F(x) = f(g(x))$ for all x,

$$F'(x) = f'(g(x)) \cdot g'(x)$$

Leibiniz notation, $\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}$ z dependent on y, y " on x

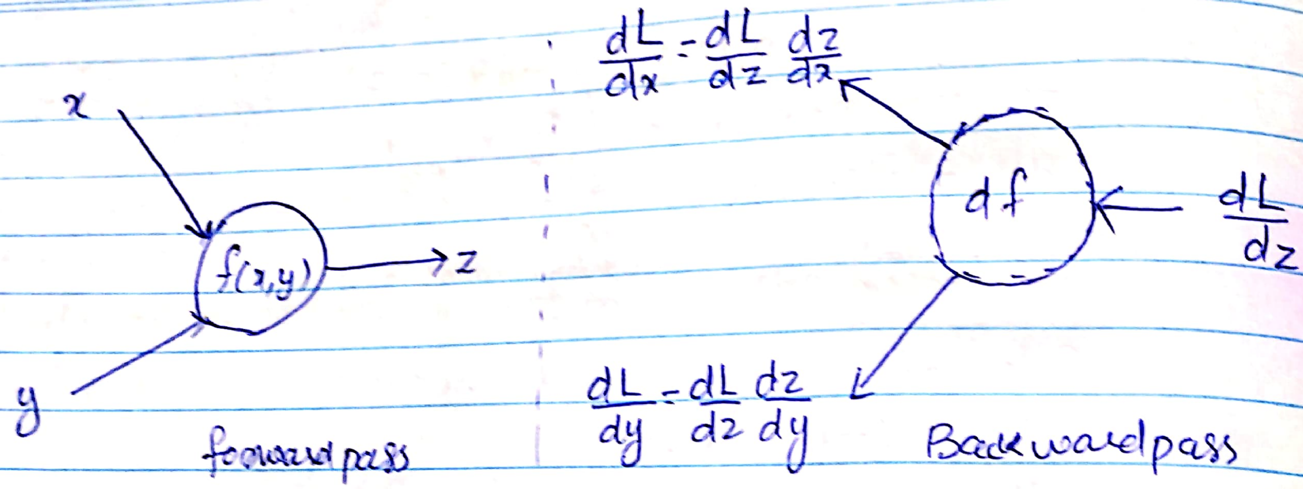
Sigmoid function :- $z = \frac{1}{(1 + np.exp(-np.dot(W, x)))}$ # forward pass

Computes the gradient of the z w.r.t input x ; `dx = np.dot(W.T, z*(1-z))` # backward pass local gradient x

(transpose of weight matrix) (derivative of sigmoid function)

output z w.r.t to gradients, W, `dW = np.outer(z*(1-z), x)` # backward pass, local gradient W

Sigmoid.



x
y
$f(x,y)$
z
forward pass
$\frac{dL}{dx} = \frac{dL}{dz}\frac{dz}{dx}$
df
$\frac{dL}{dz}$
$\frac{dL}{dy} = \frac{dL}{dz}\frac{dz}{dy}$
Backward pass