

Gradients:

Gradients with 1 variable functions:

Gradients

- Given a function with 1 output and  $n$  inputs  
 $f(x) = f(x_1, x_2, \dots, x_n)$
- Its gradient is a vector of partial derivatives with respect to each input

$$\frac{\partial f}{\partial x} = \left[ \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_n} \right]$$

Gradients with Multi-variables functions:

Jacobian Matrix: Generalization of the Gradient

- Given a function with  $m$  outputs and  $n$  inputs  
 $f(x) = [f_1(x_1, x_2, \dots, x_n), \dots, f_m(x_1, x_2, \dots, x_n)]$
- It's Jacobian is an  $m \times n$  matrix of partial derivatives

$$\frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \dots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

$\left( \frac{\partial f}{\partial x} \right)_{ij} = \frac{\partial f_i}{\partial x_j}$

Some example Jacobians:

Activation Functions: e.g. Sigmoid(x)

Example Jacobian: Elementwise activation Function

$$h = f(z), \text{ what is } \frac{\partial h}{\partial z}?$$
$$h_i = f(z_i)$$
$$\left( \frac{\partial h}{\partial z} \right)_{ij} = \frac{\partial h_i}{\partial z_j} = \frac{\partial}{\partial z_j} f(z_i)$$

definition of Jacobian

$$= \begin{cases} f'(z_i) & \text{if } i = j \\ 0 & \text{if otherwise} \end{cases}$$

regular 1-variable derivative

Some Other Jacobians:

Other Jacobians

$$\frac{\partial}{\partial x} (Wx + b) = W$$
$$\frac{\partial}{\partial b} (Wx + b) = I \text{ (Identity matrix)}$$
$$\frac{\partial}{\partial u} (u^T h) = h^T$$

- Compute these at home for practice!  
Check your answers with the lecture notes

$$\frac{\partial}{\partial u} (u^T h) = \frac{\partial}{\partial u} (u^T) \cdot h + \frac{\partial}{\partial u} (h) \cdot u^T$$
$$= 1s^T h + 0$$
$$= h^T$$

$$\rightarrow \frac{\partial}{\partial x} (Wx + b) = \frac{\partial}{\partial x} (Wx) + \frac{\partial}{\partial x} (b) \Rightarrow \frac{\partial}{\partial f} (a+b) = \frac{\partial}{\partial f} (a) + \frac{\partial}{\partial f} (b)$$
$$= W + 0 \Rightarrow \frac{\partial}{\partial x} (const) = 0$$
$$\rightarrow \frac{\partial}{\partial b} (Wx + b) = 0 + 1 \Rightarrow \text{similar to above}$$
$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \Rightarrow \text{Considering Jacobians.}$$

Analytical Calculation of Gradients:

$\rightarrow$  simplifying  $f(Wx+b)$  as  $f(z)$

Back to our Neural Net!

- Let's find  $\frac{\partial s}{\partial b}$
- Really, we care about the gradient of the loss  $J$ , but we will compute the gradient of the score for simplicity

$$s = u^T h$$
$$h = f(Wx + b)$$

$x$  (input)

$$\frac{\partial s}{\partial b} = \frac{\partial s}{\partial h} \cdot \frac{\partial h}{\partial z} \cdot \frac{\partial z}{\partial b} \rightarrow \oplus$$

$$= u^T \cdot \text{diag}(f'(z))$$

$$\xrightarrow{\text{Identity}} \frac{\partial (Wx+b)}{\partial b} \text{ w.r.t } b.$$

$$\xrightarrow{\text{activation fun Jacobian}}$$

$$= u^T \cdot f'(z)$$

eg.  $I \leftarrow \mathbb{I}$  share same computations. so to avoid duplications.

$$\delta = \frac{\partial s}{\partial h} \cdot \frac{\partial h}{\partial z}$$
$$\frac{\partial s}{\partial b} = \delta \cdot \frac{\partial z}{\partial b} \leftarrow$$
$$\frac{\partial s}{\partial w} = \delta \cdot \frac{\partial z}{\partial w}.$$

Backpropagation

Starting with Forward pass: Passing inputs through the computational graph from left to right

Computation Graphs and Backpropagation

- Software represents our neural net equations as a graph
- Source nodes: inputs
- Interior nodes: operations
- Edges pass along result of the operation

$s = u^T h$   
 $h = f(z)$   
 $z = Wx + b$   
 $x$  (input)

Backward pass: Passing gradients from right to left

### Backpropagation

- Then go backwards along edges
- Pass along **gradients**

$s = u^T h$   
 $h = f(z)$   
 $z = Wx + b$   
 $x$  (input)

Example of single node with single input:

Upstream gradient: gradient of final output/error wrt. Node's output  
Local gradient: gradient of node's output wrt. node's input  
Downstream gradient: gradient of final output/ error wrt. node's input

### Backpropagation: Single Node

- Each node has a **local gradient**
- The gradient of its output with respect to its input

$h = f(z)$

Example of single node with multiple inputs:

Only difference is in local gradients step. For each input of a node, we calculate a local gradient.

### Backpropagation: Single Node

- Multiple inputs → multiple local gradients

$z = Wx$

Impact of Node Types on Gradients

### Node Intuitions

$f(x, y, z) = (x + y) \max(y, z)$   
 $x = 1, y = 2, z = 0$

- + “distributes” the upstream gradient
- max “routes” the upstream gradient
- \* “switches” the upstream gradient

Manual Gradient Checking:

### Manual Gradient checking: Numeric Gradient

- For small  $h$  ( $\approx 1e-4$ ),  $f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}$
- Easy to implement correctly
- But approximate and **very** slow:
  - You have to recompute  $f$  for **every parameter** of our model
- Useful for checking your implementation
  - In the old days, we hand-wrote everything, doing this everywhere was the key test
  - Now much less needed; you can use it to check layers are correctly implemented