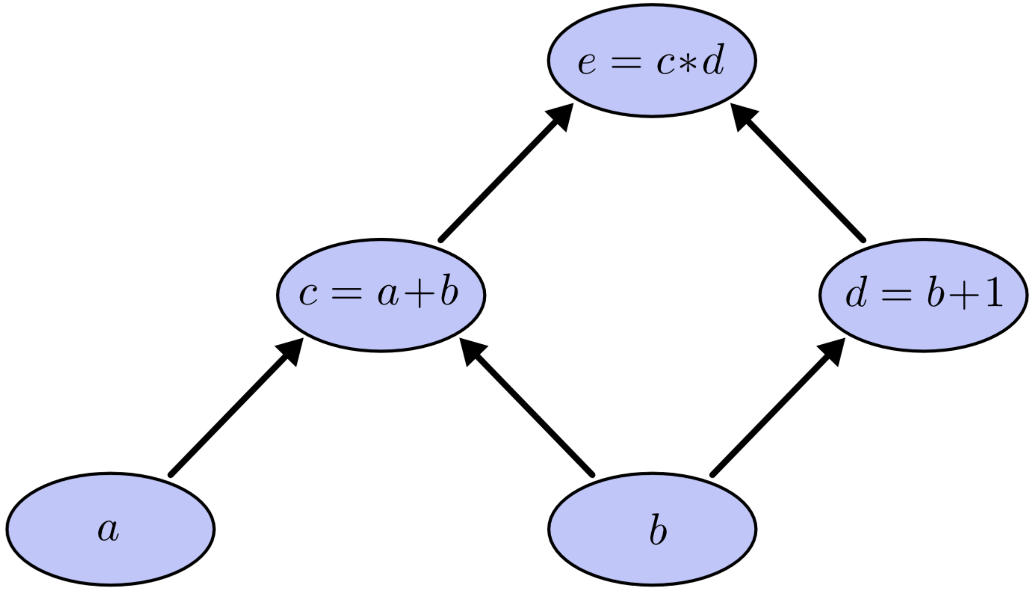
business innovation through algorithms

Computational Graphs, Neural Nets and Tensorflow

CGnal S.p.A – Corso Venezia 43 - Milano

13 dicembre 2022 | Milano

Agenda



1. Introduction to Computational Graphs
2. Introduction to Tensorflow 2.x
3. Tensorflow Lab
    1. Components of a computational graph in tensorflow

Why Computational Graphs

Why study computational graphs?

1. Training neural networks requires fast, complicated and efficient computation.
2. This computation is basically optimizing the parameters through an algorithmic process called Gradient Descent
3. Neural Networks are complex architectures that require efficient gradient computation.
4. Computational graphs are a framework to compute analytical gradients for arbitrarly complex functions

.

$$\underset{(\boldsymbol{Neural\ Net})}{\boldsymbol{f(x)}} \xrightarrow{\boldsymbol{CG}} \boldsymbol{\frac{df}{dx}}$$

Introduction to Computational Graphs

What is a computational graph?

Computational Graph is a way to represent mathematical expressions as a directed graph data structure. The nodes represent mathematical operations and the edges represent function argument/data dependency.

$$\boldsymbol{e = (a + b) * (b + 1)}$$

How to represent the expression as a computational graph ?

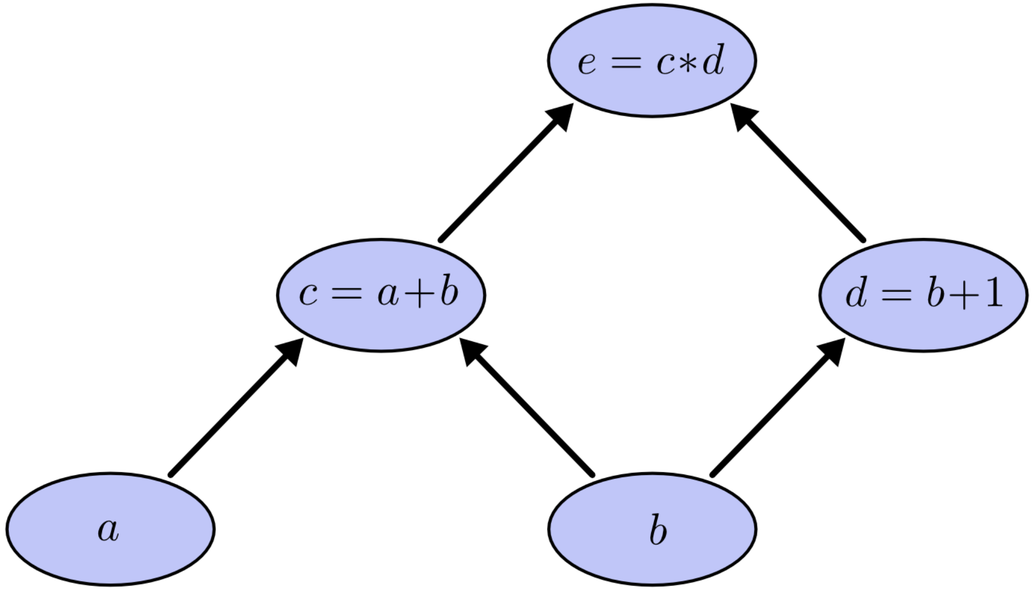
1. 3 Ops : 2 Additions and 1 Multiplication
2. Break down the expression into smaller parts
3. Write c = a+b and d = b+1

What is a computational graph?

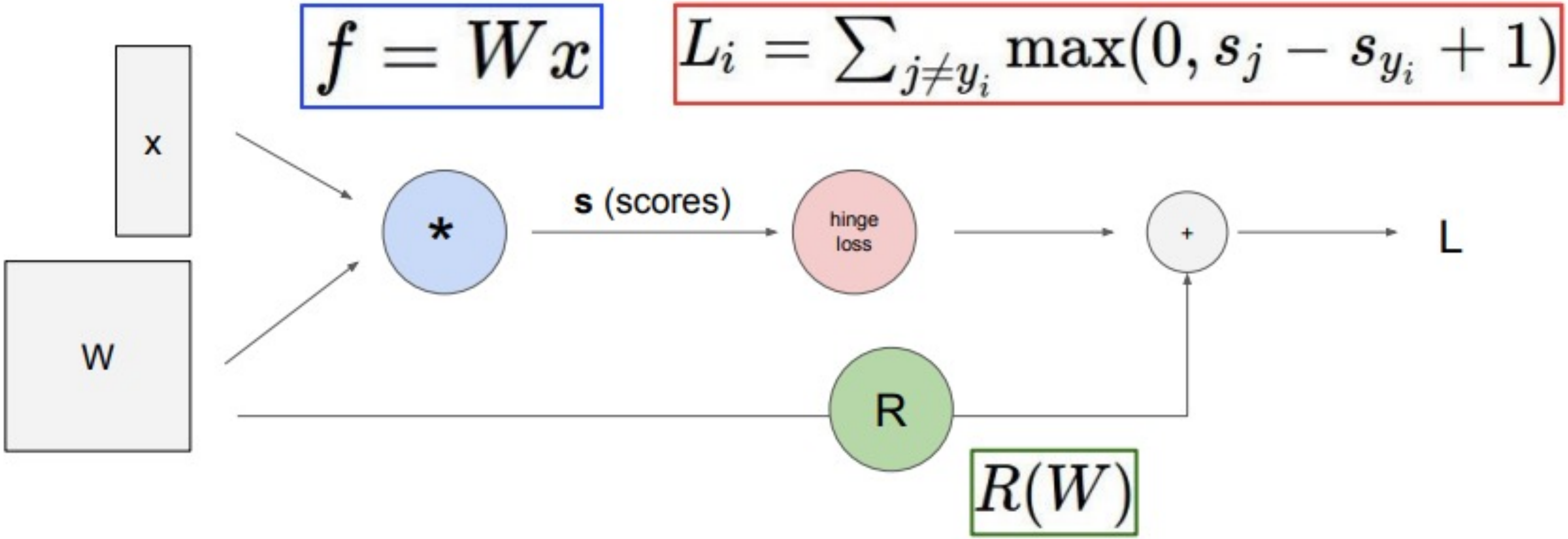
Computational Graph is a way to represent mathematical expressions as a directed graph data structure. The nodes represent mathematical operations and the edges represent function argument/data dependency.

$$\boldsymbol{e = (a + b) * (b + 1)}$$

$$\begin{aligned} \boldsymbol{c} &= \boldsymbol{a + b} \\ \boldsymbol{d} &= \boldsymbol{b + 1} \\ \boldsymbol{e} &= \boldsymbol{c * d} \end{aligned}$$



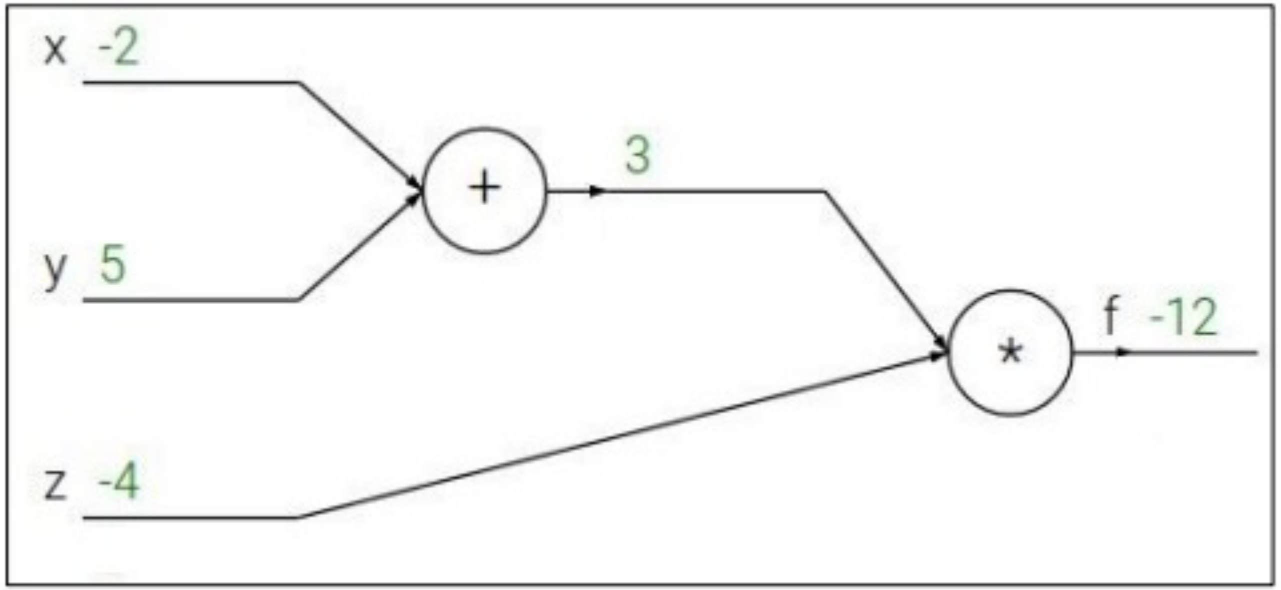
Computational Graph



Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Simple Example

$$f(x, y, z) = (x + y) * z$$
$$x = -2, y = 5, z = -4$$



Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Backpropagation: Simple Example

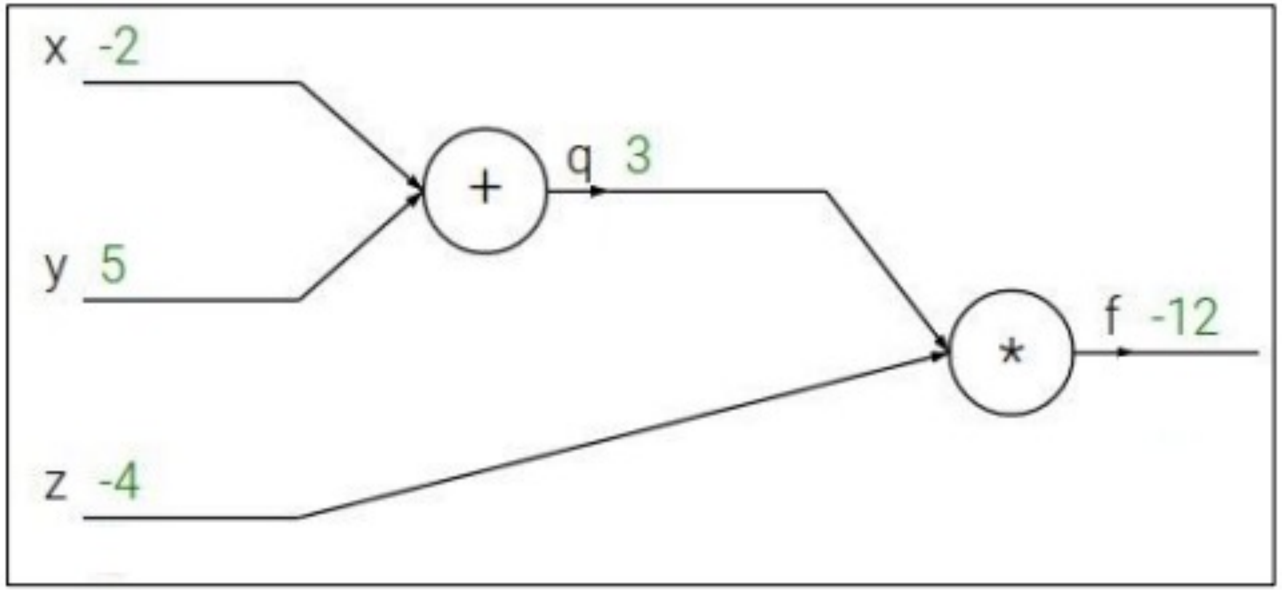
$$\boldsymbol{f(x,y,z) = (x+y) * z}$$

$$\boldsymbol{x = -2, y = 5, z = -4}$$

$$\boldsymbol{q = (x+y)} \quad \frac{\partial q}{\partial x} = 1, \ \frac{\partial q}{\partial y} = 1$$

$$\boldsymbol{f = qz} \quad \frac{\partial f}{\partial q} = z, \ \frac{\partial f}{\partial z} = q$$

Find : $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Backpropagation: Simple Example

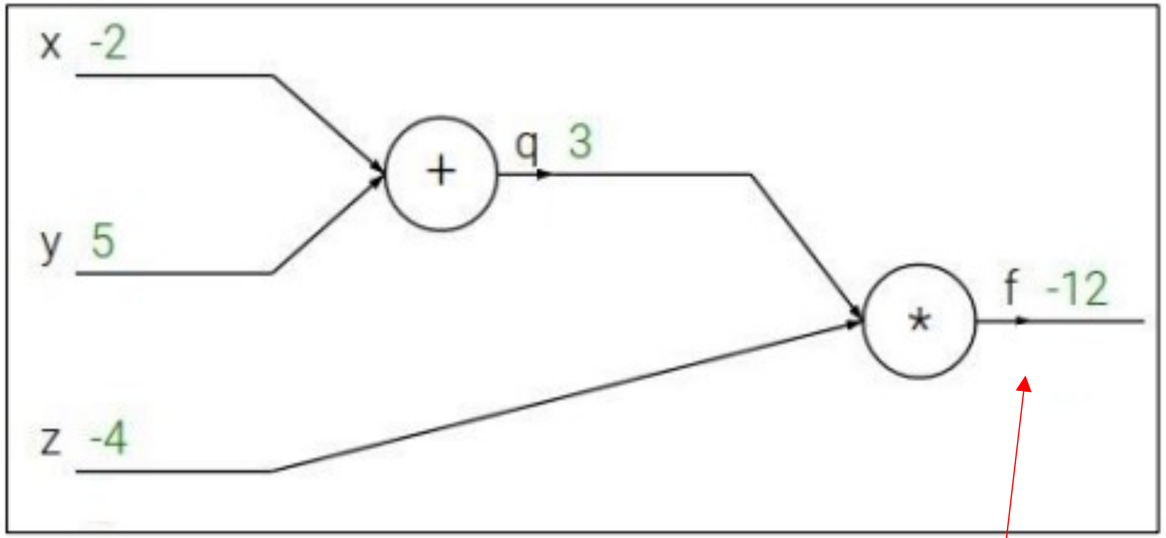
$$f(x, y, z) = (x + y) * z$$

$$x = -2, y = 5, z = -4$$

$$q = (x + y) \quad \frac{\partial q}{\partial x} = 1, \ \frac{\partial q}{\partial y} = 1$$

$$f = qz \quad \frac{\partial f}{\partial q} = z, \ \frac{\partial f}{\partial z} = q$$

Find : $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$

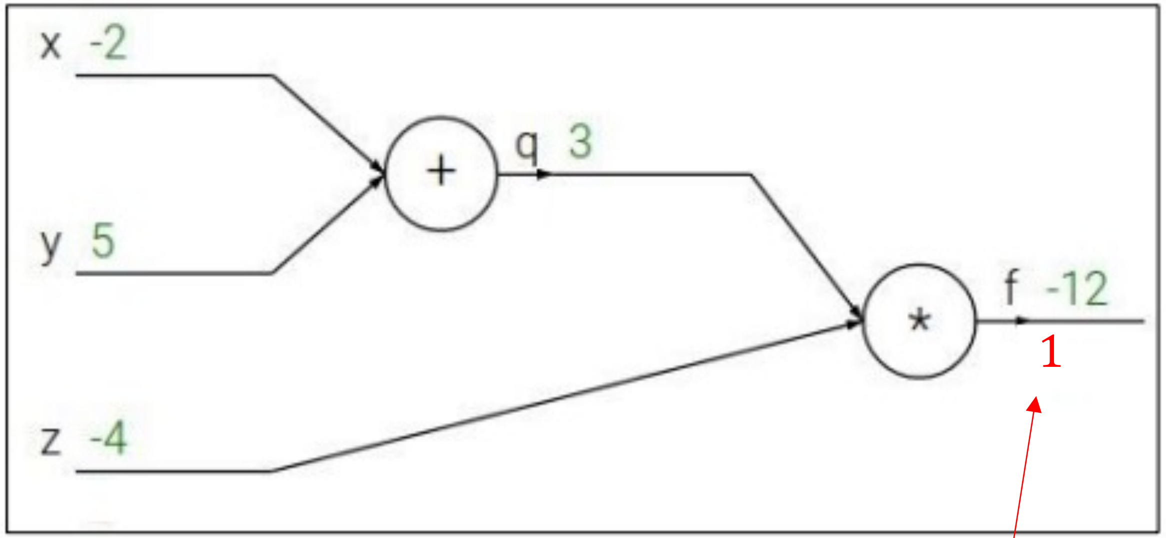


$$\frac{\partial f}{\partial f}$$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Backpropagation: Simple Example

$$\boldsymbol{f(x,y,z) = (x+y) * z}$$
$$\boldsymbol{x=-2, y=5, z=-4}$$



$$\boxed{\boldsymbol{q = (x+y)} \quad \frac{\partial q}{\partial x} = 1, \ \frac{\partial q}{\partial y} = 1}$$

$$\boxed{\boldsymbol{f = qz} \quad \frac{\partial f}{\partial q} = z, \ \frac{\partial f}{\partial z} = q}$$

$$\boxed{\frac{\partial f}{\partial f}}$$

Find : $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Backpropagation: Simple Example

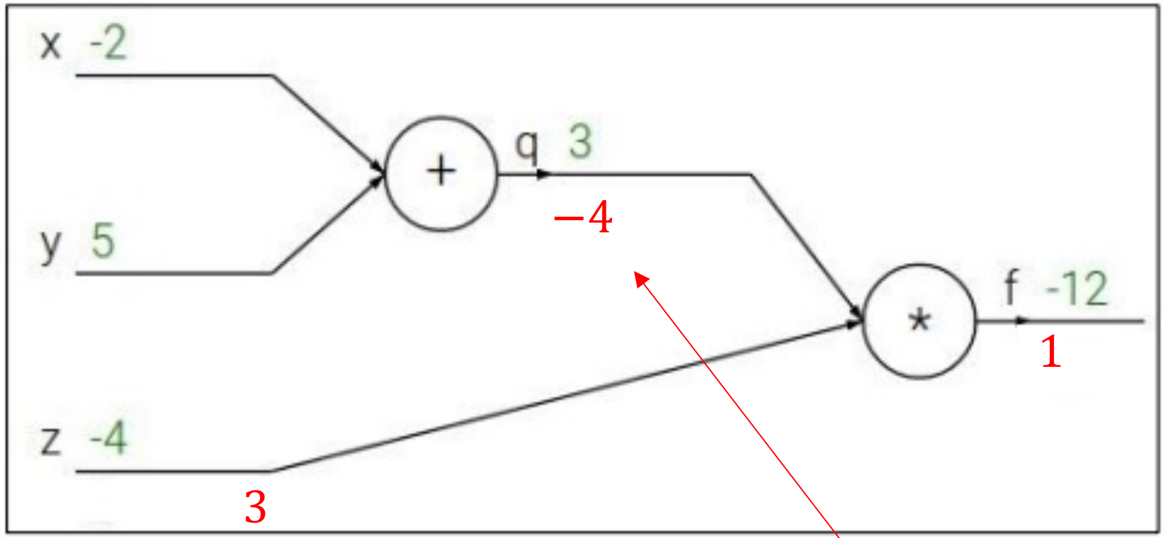
$$\boldsymbol{f(x, y, z) = (x + y) * z}$$

$$\boldsymbol{x = -2, y = 5, z = -4}$$

$$\boxed{\boldsymbol{q = (x + y)} \quad \frac{\partial q}{\partial x} = 1, \ \frac{\partial q}{\partial y} = 1}$$

$$\boxed{\boldsymbol{f = qz} \quad \frac{\partial f}{\partial q} = z, \ \frac{\partial f}{\partial z} = q}$$

Find : $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$\boxed{\frac{\partial f}{\partial z}}$ $\boxed{\frac{\partial f}{\partial q}}$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Backpropagation: Simple Example

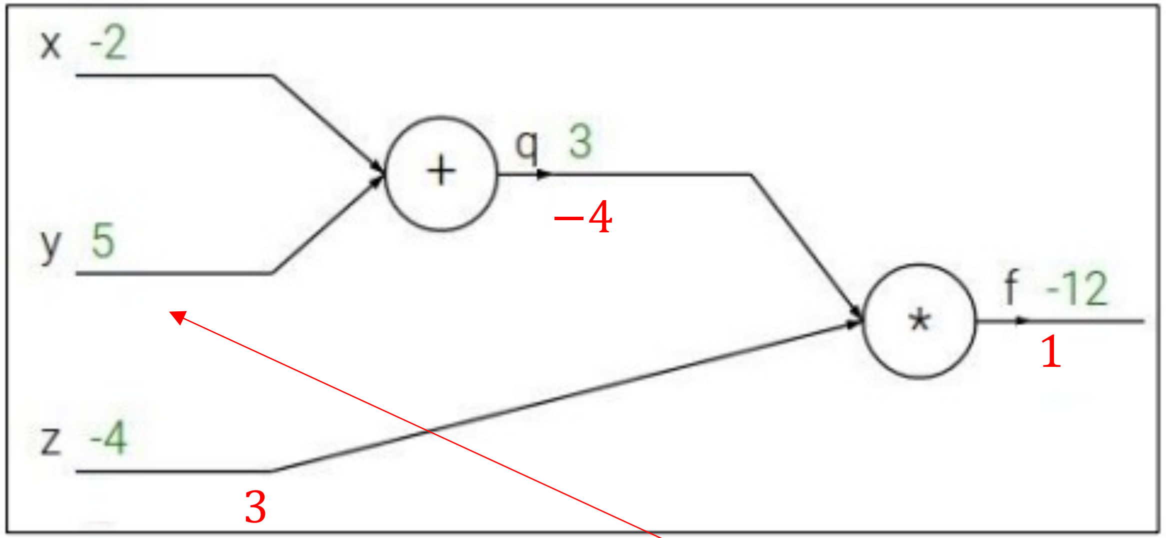
$$f(x,y,z) = (x+y) * z$$

$$x = -2, y = 5, z = -4$$

$$q = (x+y) \quad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \quad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Find : $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$\frac{\partial f}{\partial y}$?

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Backpropagation: Simple Example

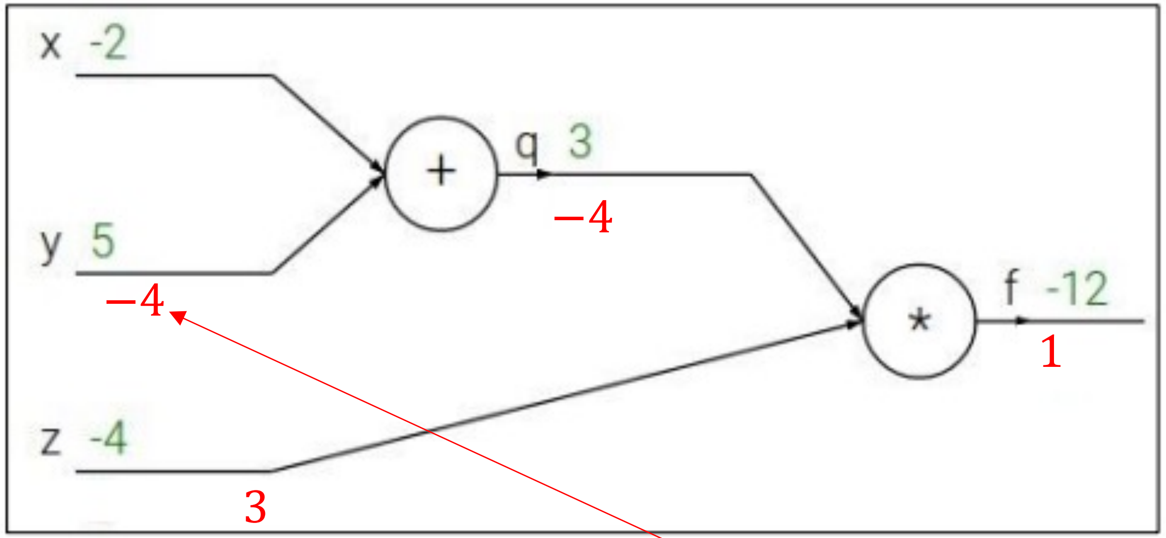
$$f(x, y, z) = (x + y) * z$$

$$x = -2, y = 5, z = -4$$

$$q = (x + y) \quad \frac{\partial q}{\partial x} = 1, \ \frac{\partial q}{\partial y} = 1$$

$$f = qz \quad \frac{\partial f}{\partial q} = z, \ \frac{\partial f}{\partial z} = q$$

Find : $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$\frac{\partial f}{\partial y}$?

Chain rule: $\frac{\partial f}{\partial y} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial y}$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Backpropagation: Simple Example

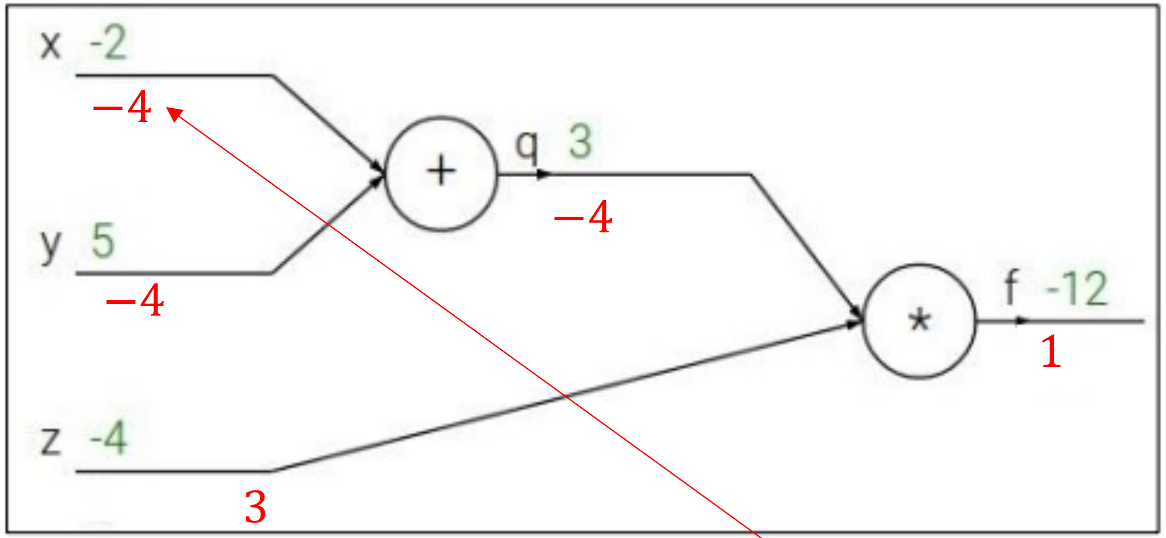
$$f(x, y, z) = (x + y) * z$$

$$x = -2, y = 5, z = -4$$

$$q = (x + y) \quad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \quad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

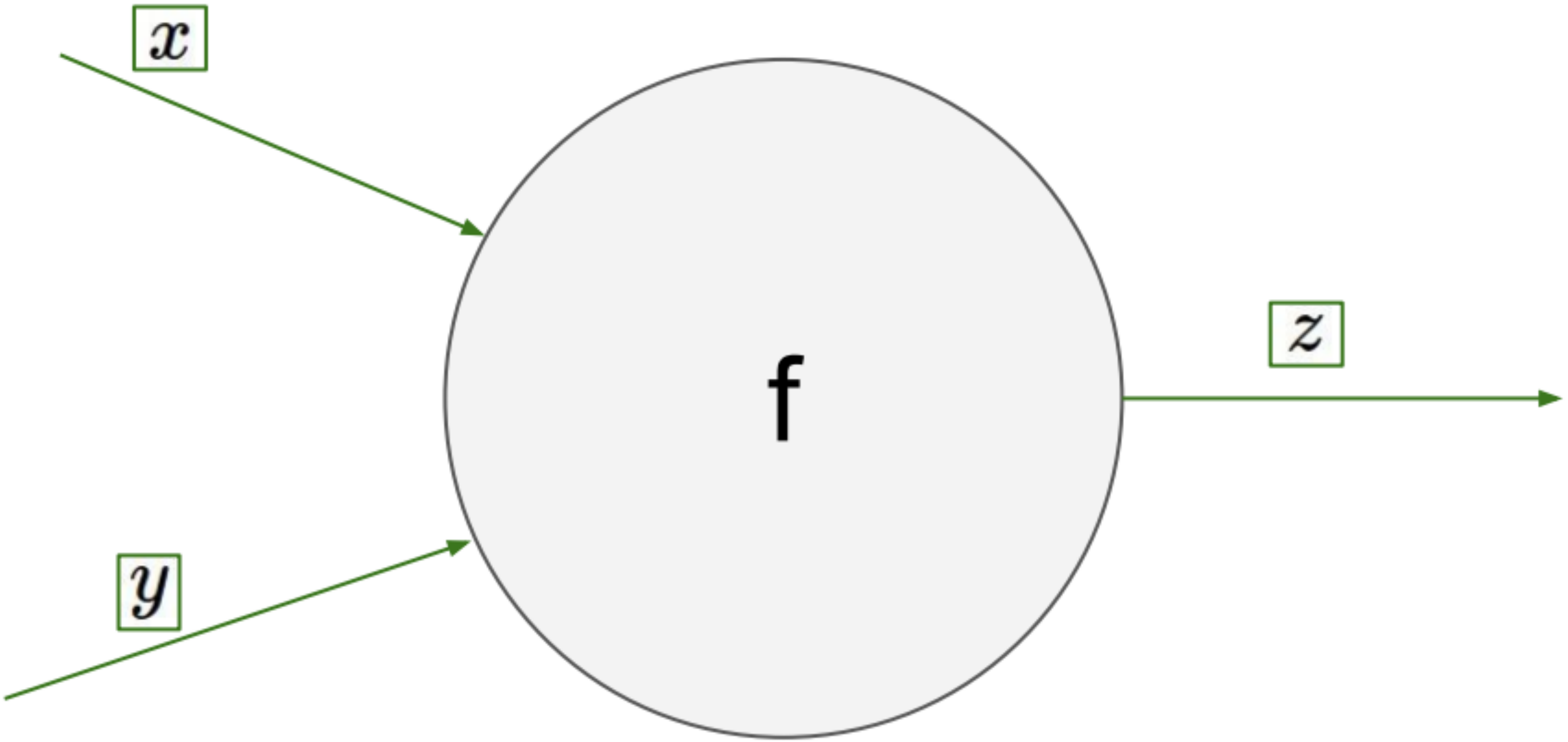
Find : $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



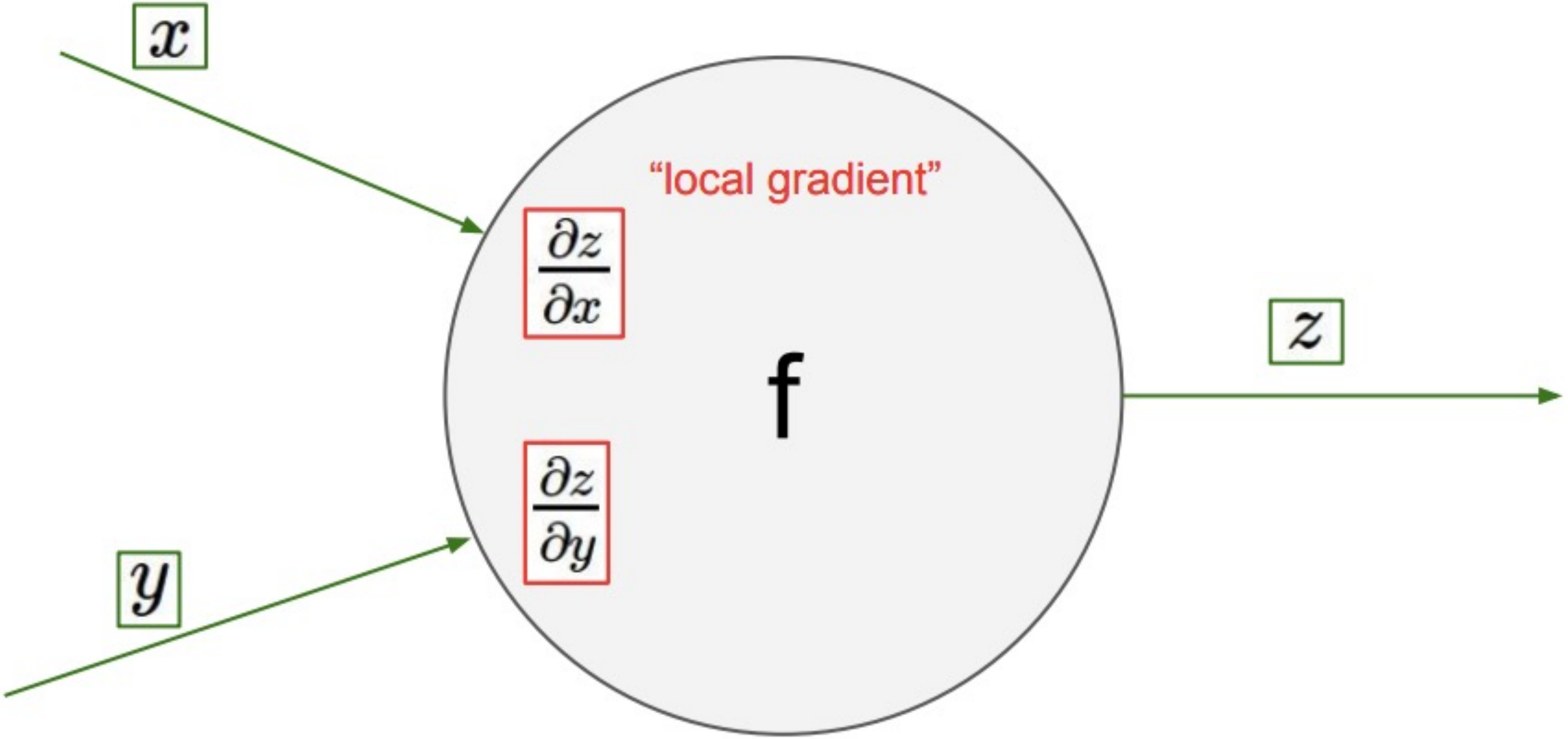
$\frac{\partial f}{\partial x}$?

Chain rule: $\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial x}$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

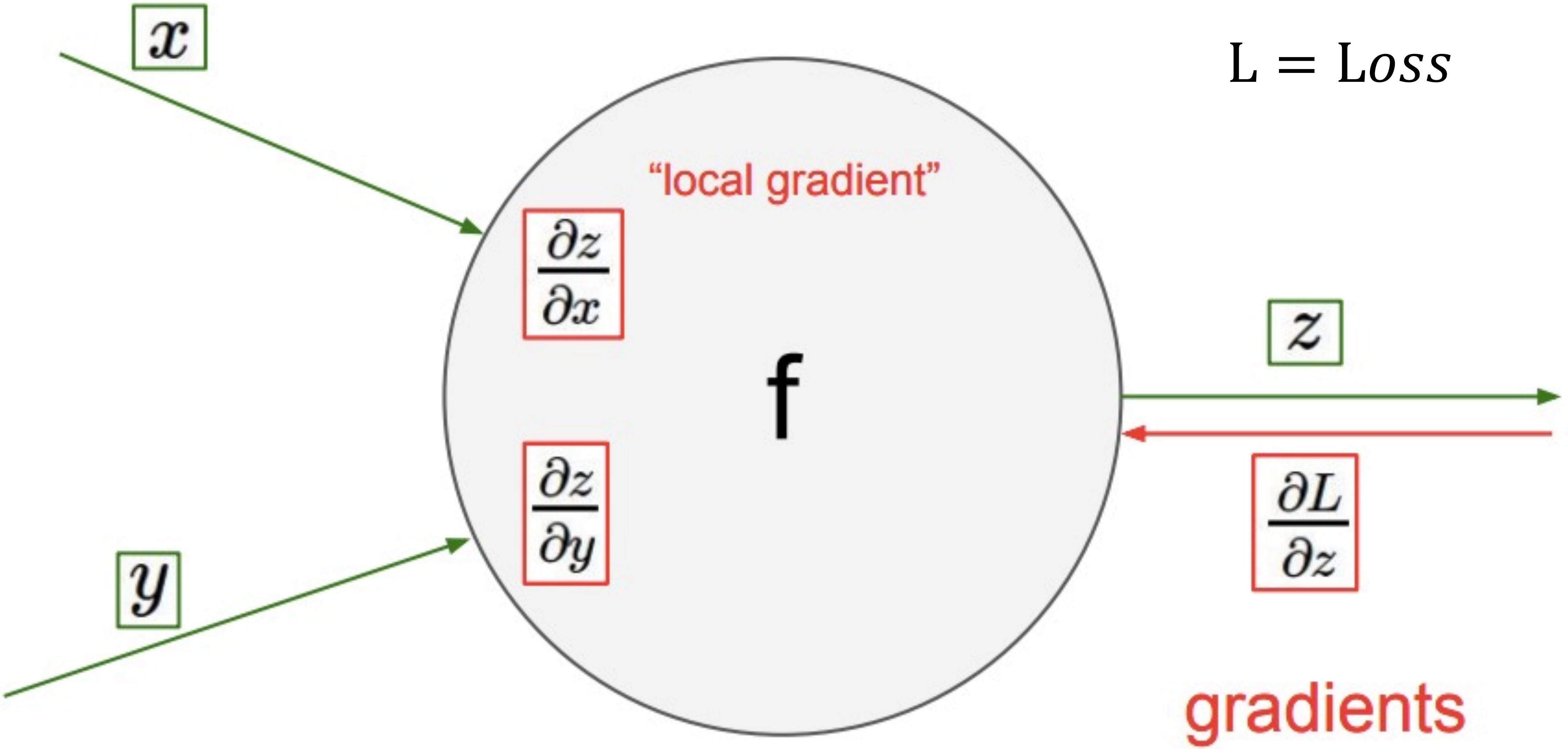


Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf



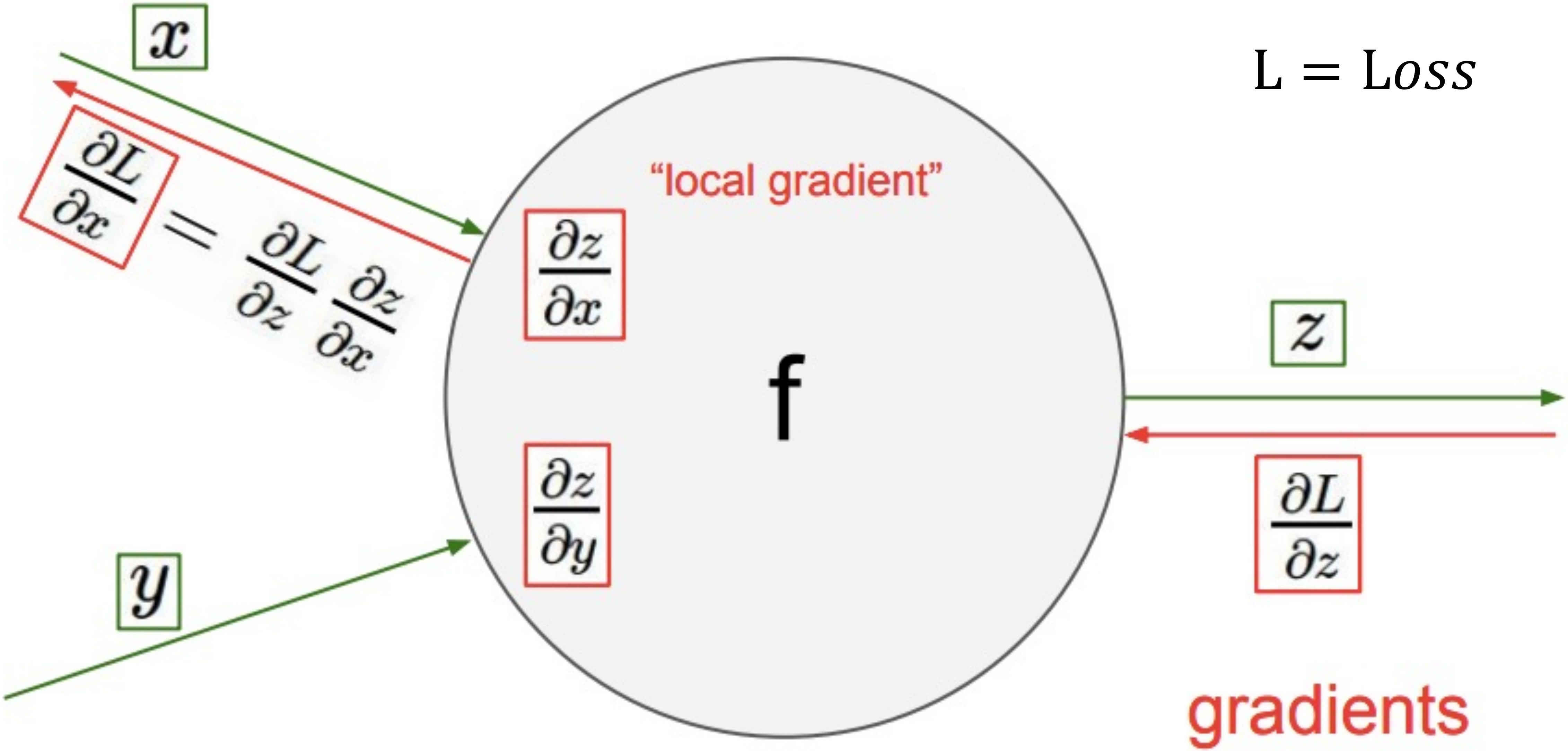
Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

$L = Loss$

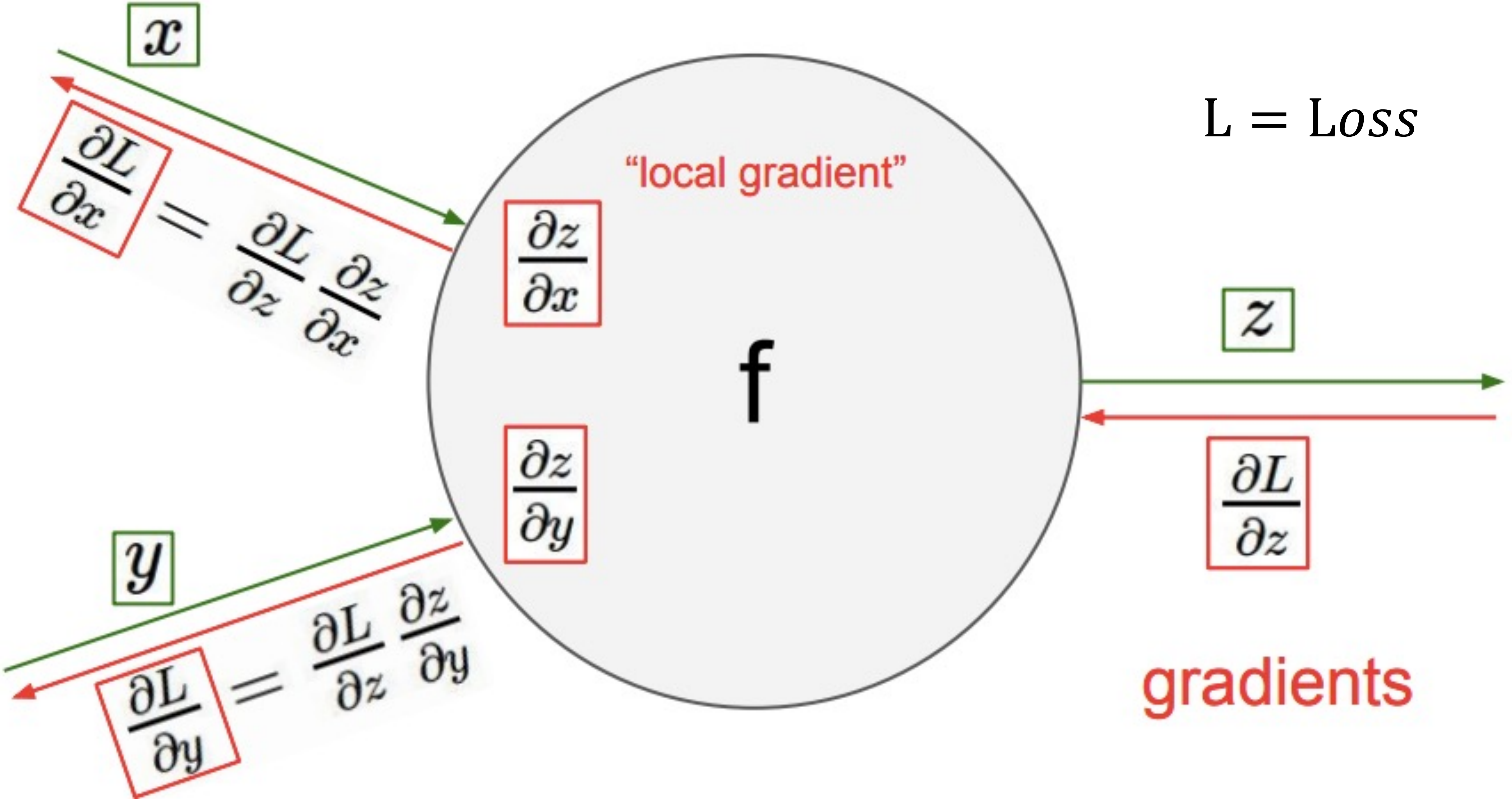


Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

$$\mathrm{L} = \mathrm{L}oss$$



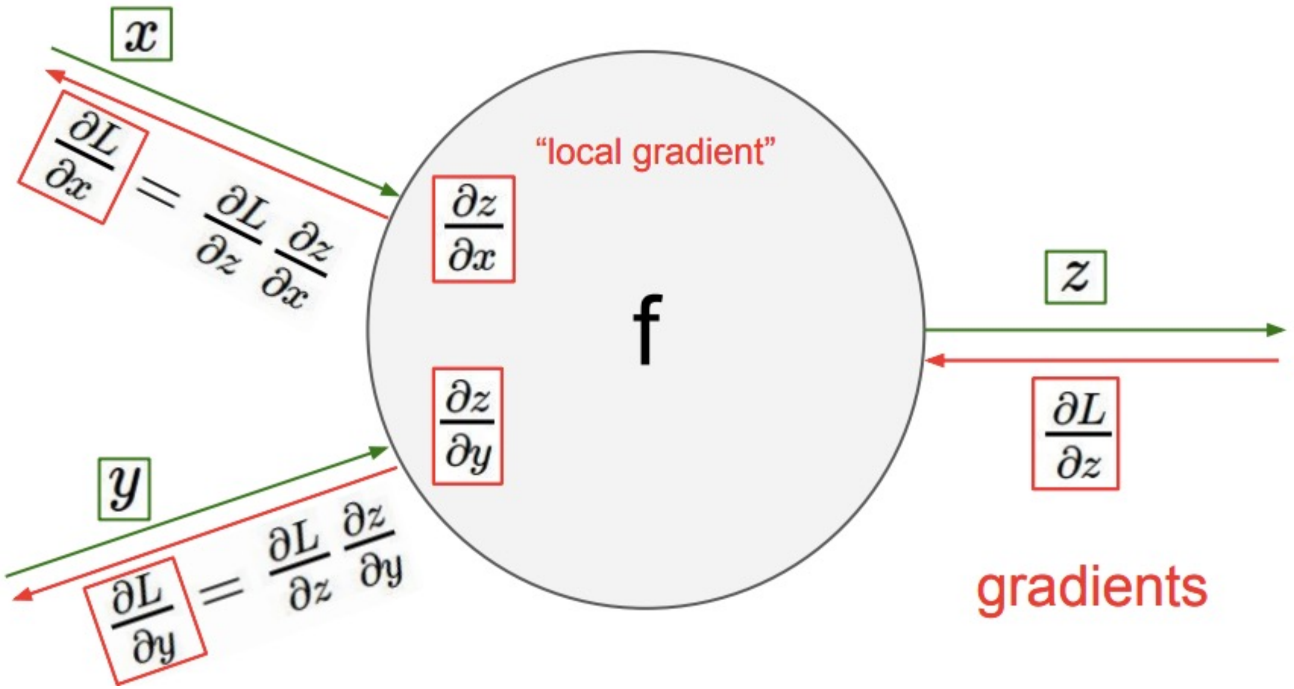
Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf



Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Takeaways

Every Node becomes ”locally” aware
i.e. every node needs to keep track of only local gradients and pass onto other nodes

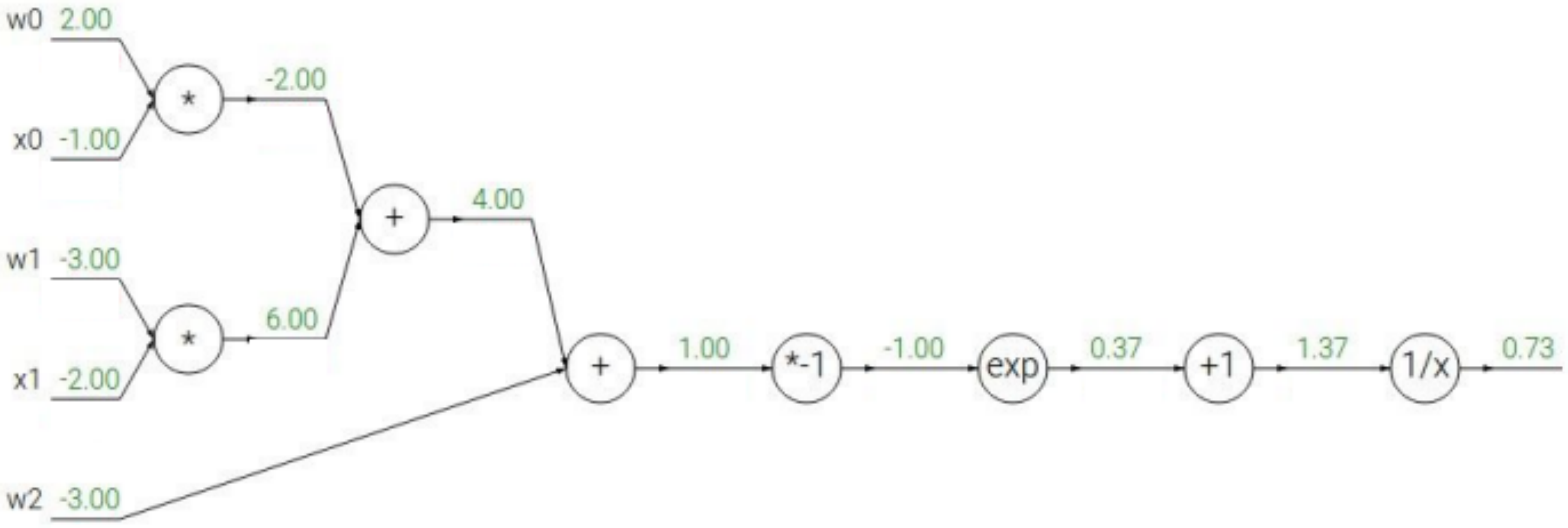


Gradient Out = Upstream Gradient * Local Gradient

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

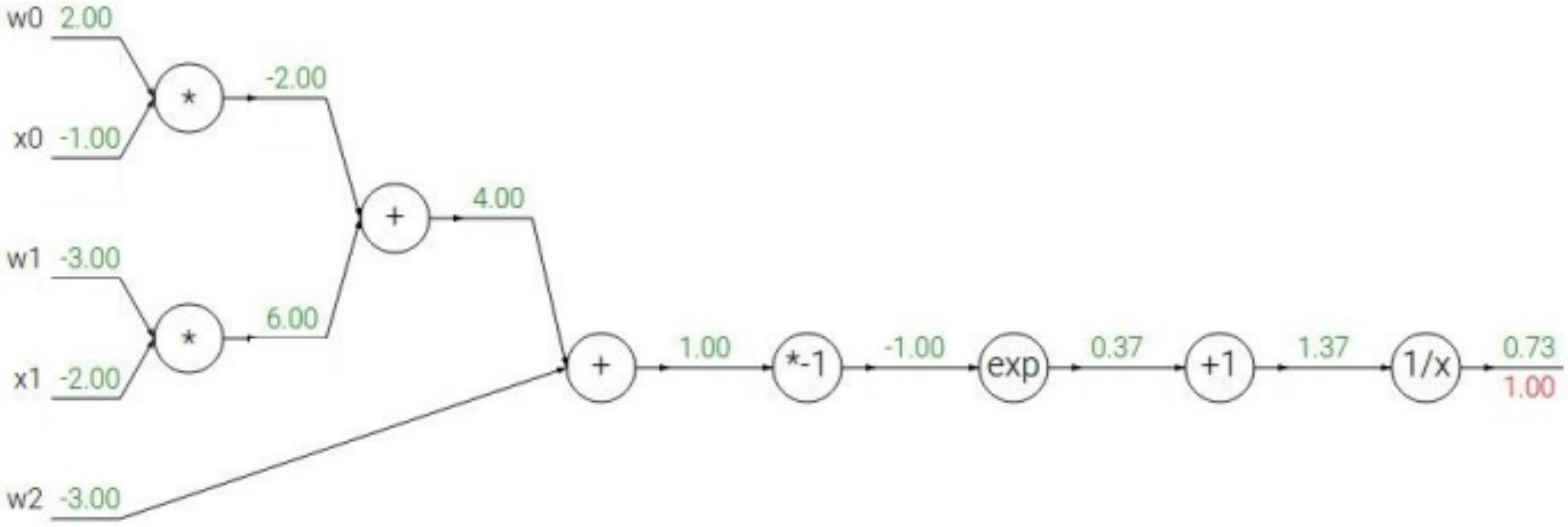
Another example:

$$f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$$



Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Another example: $f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$



$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

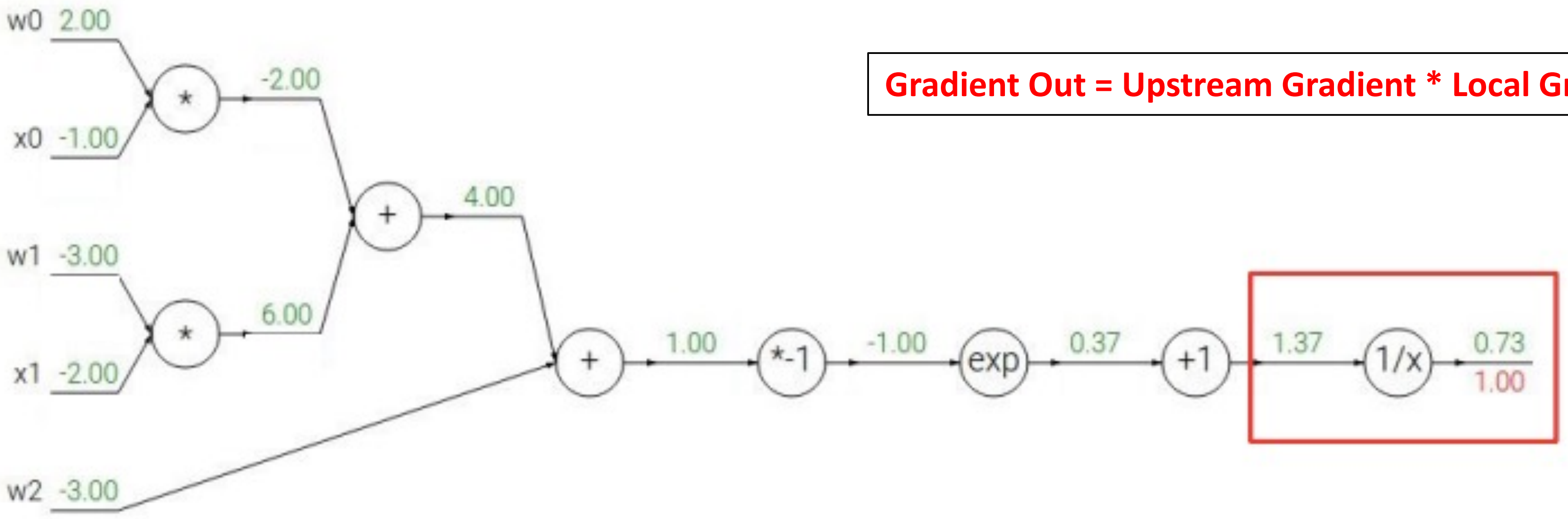
$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Another example: $f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$



Gradient Out = Upstream Gradient * Local Gradient

$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$\boxed{f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2}$$

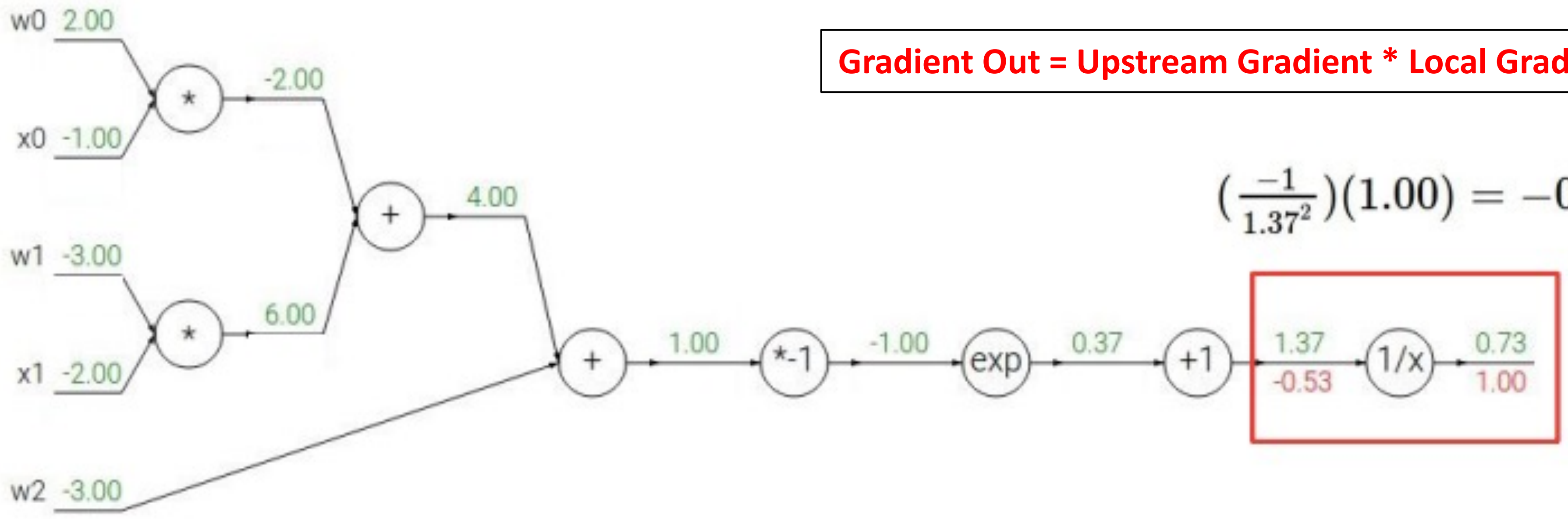
$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Another example: $f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$

Gradient Out = Upstream Gradient * Local Gradient

$$\left(\frac{-1}{1.37^2}\right)(1.00) = -0.53$$



$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

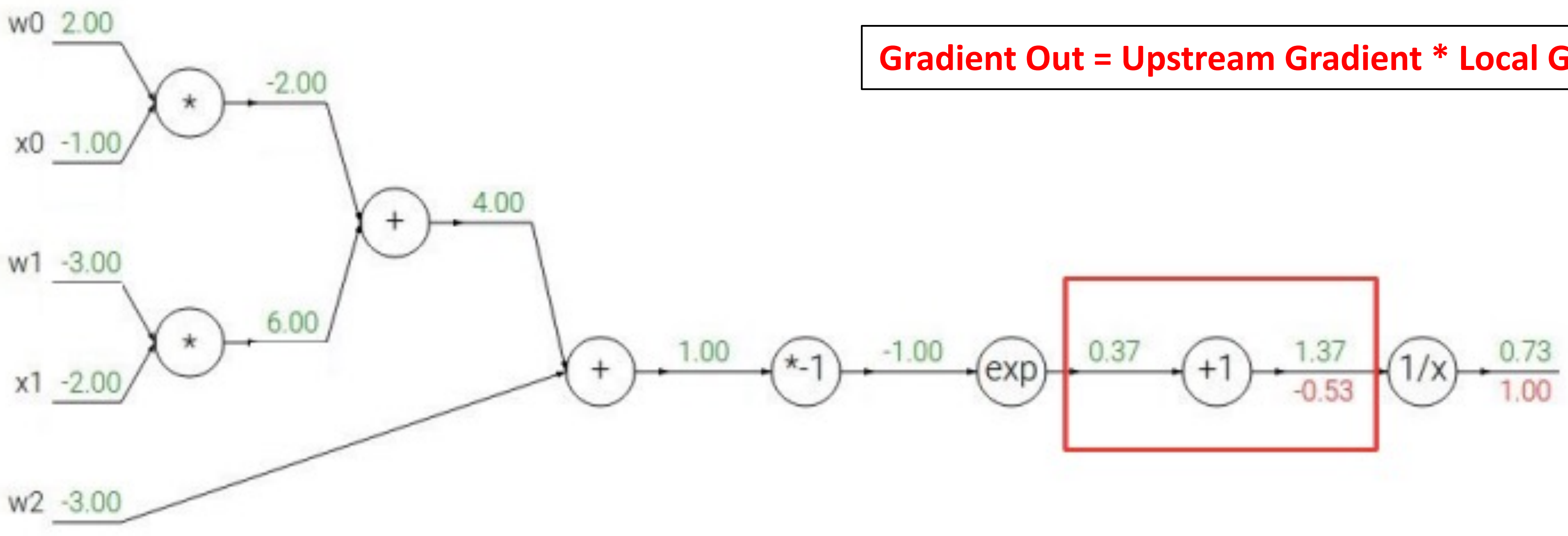
$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Another example: $f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$



Gradient Out = Upstream Gradient * Local Gradient

$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

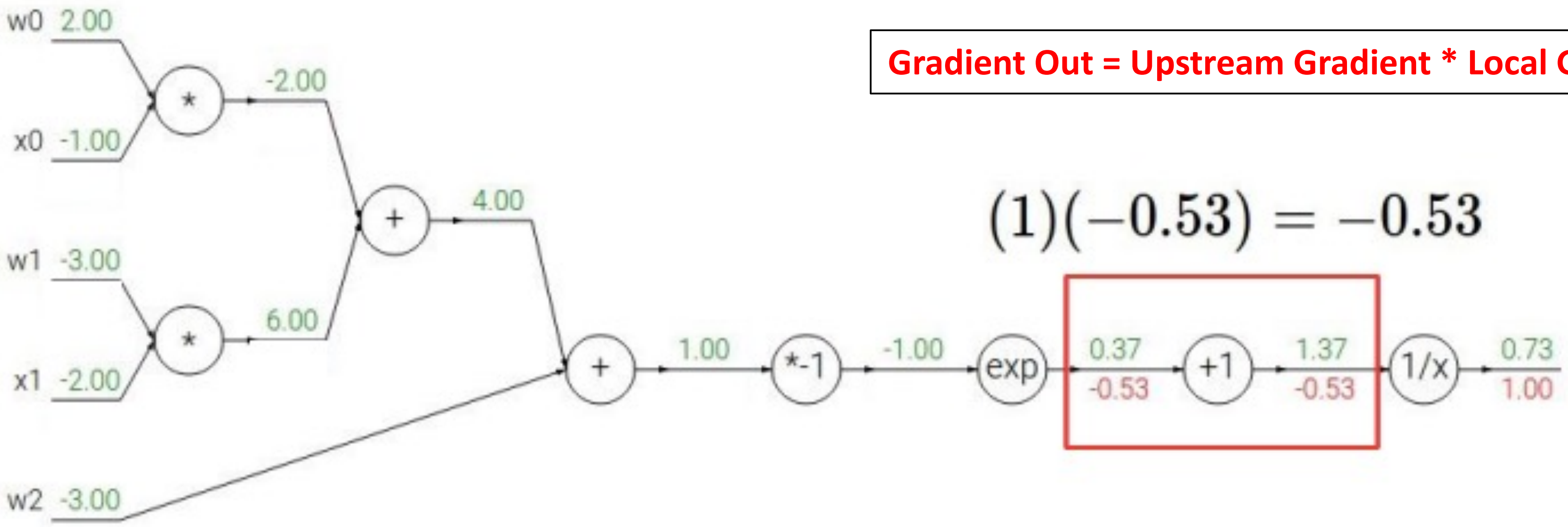
$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Another example: $f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$

Gradient Out = Upstream Gradient * Local Gradient



$$(1)(-0.53) = -0.53$$

$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$

$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$

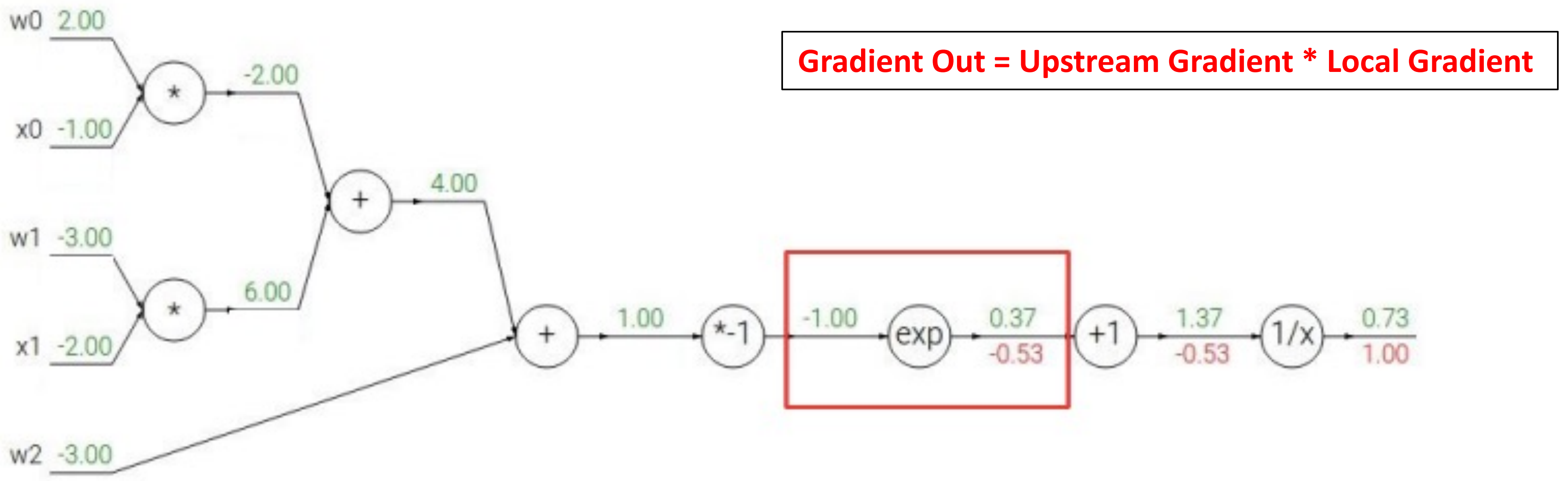
$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$

$\boxed{f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1}$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Another example:

$$f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$$



$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

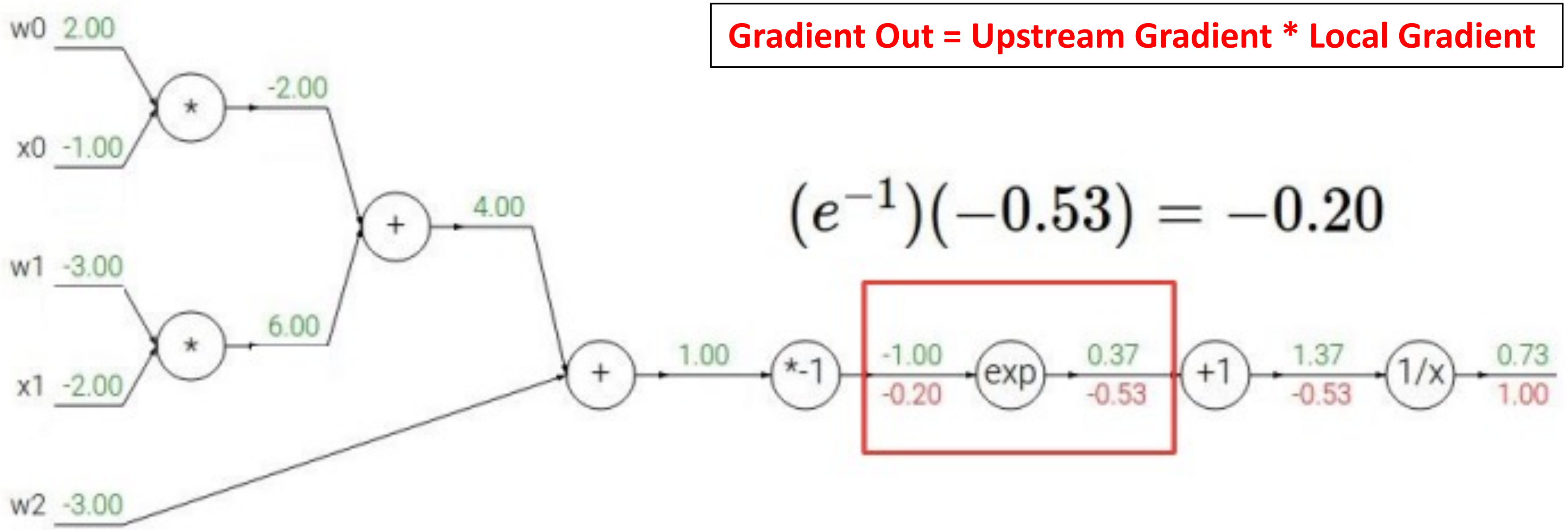
$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Another example:

$$f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$$

Gradient Out = Upstream Gradient * Local Gradient



$$(e^{-1})(-0.53) = -0.20$$

$$\boxed{f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x}$$

$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

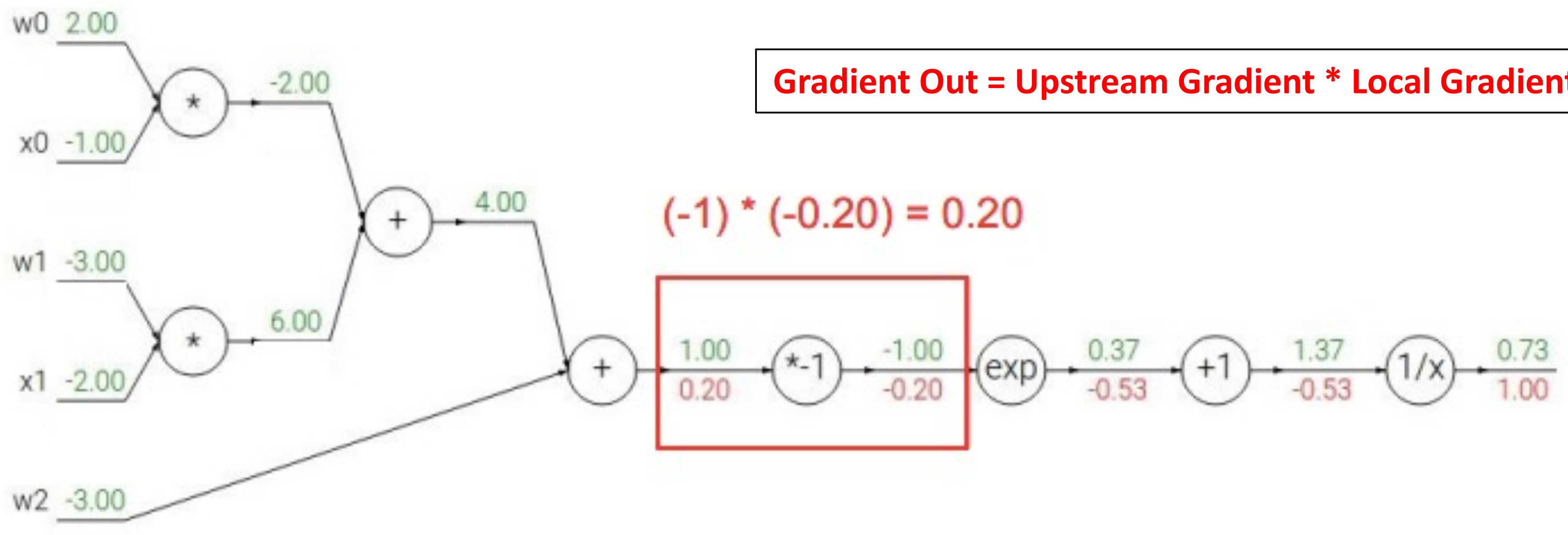
$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Another example: $f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$

Gradient Out = Upstream Gradient * Local Gradient



$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

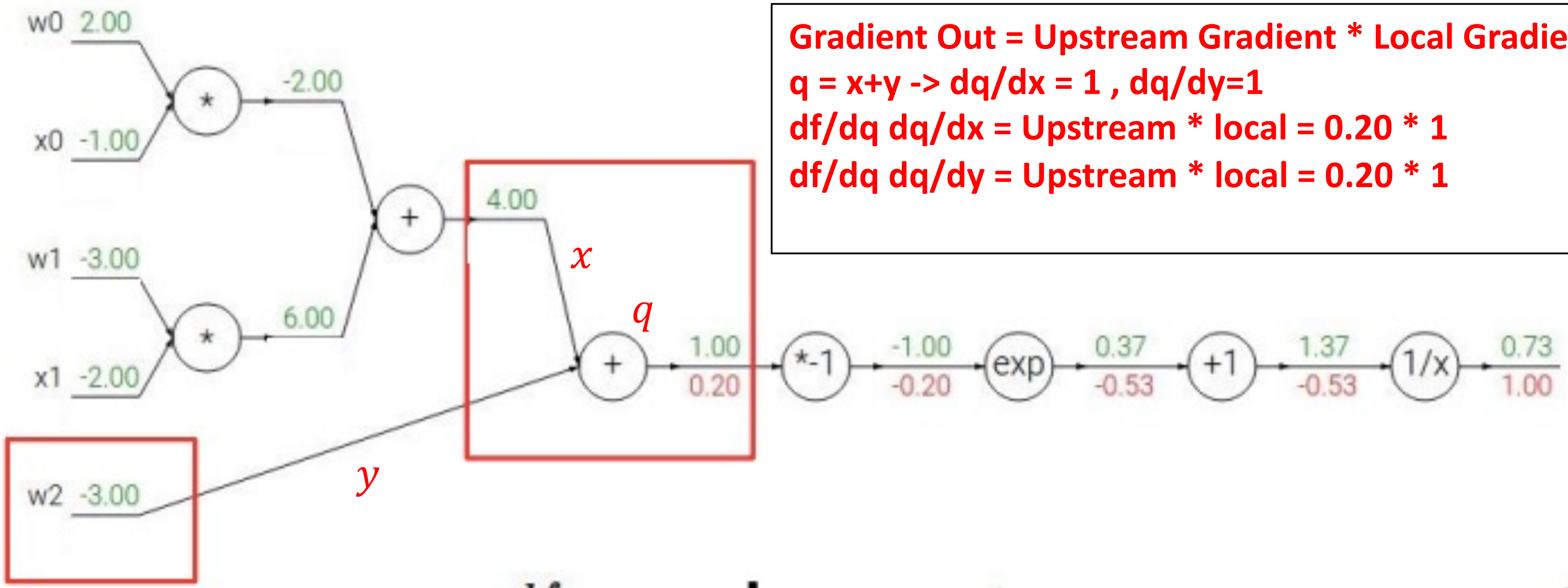
$$\boxed{f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a}$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Another example: $f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$



Gradient Out = Upstream Gradient * Local Gradient
q = x+y -> dq/dx = 1 , dq/dy=1
df/dq dq/dx = Upstream * local = 0.20 * 1
df/dq dq/dy = Upstream * local = 0.20 * 1

$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$

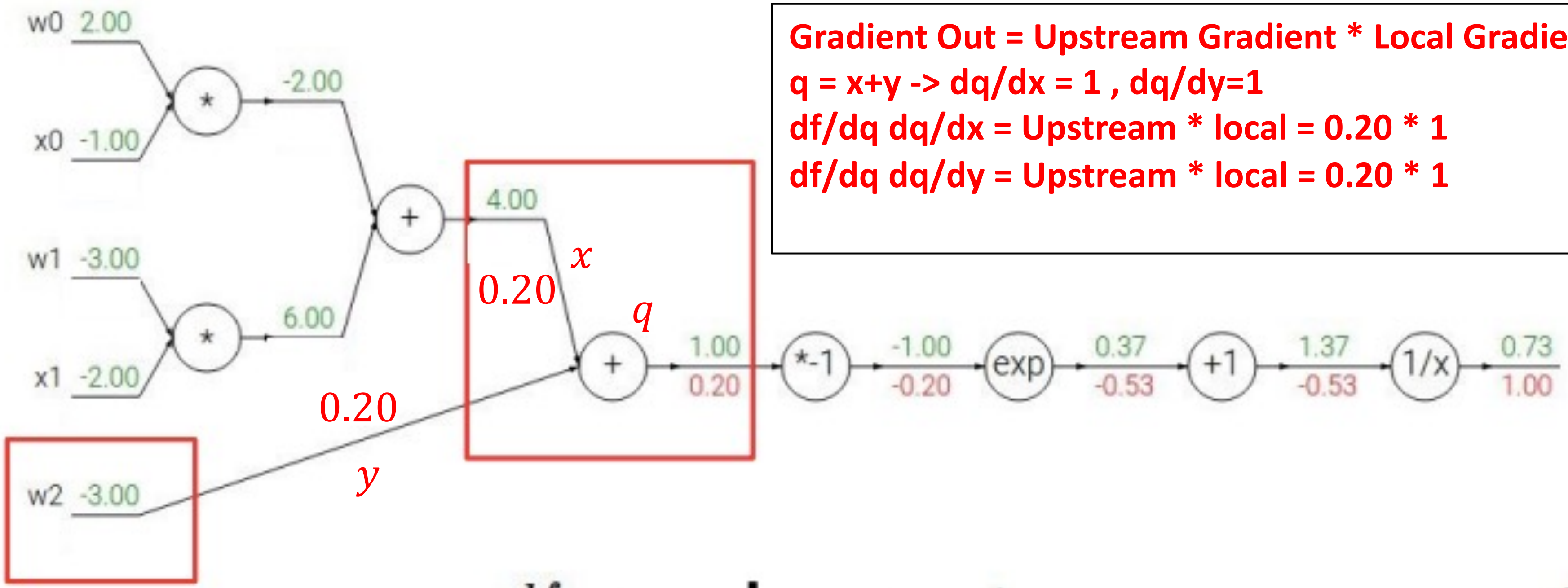
$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$

$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$

$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Another example: $f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$



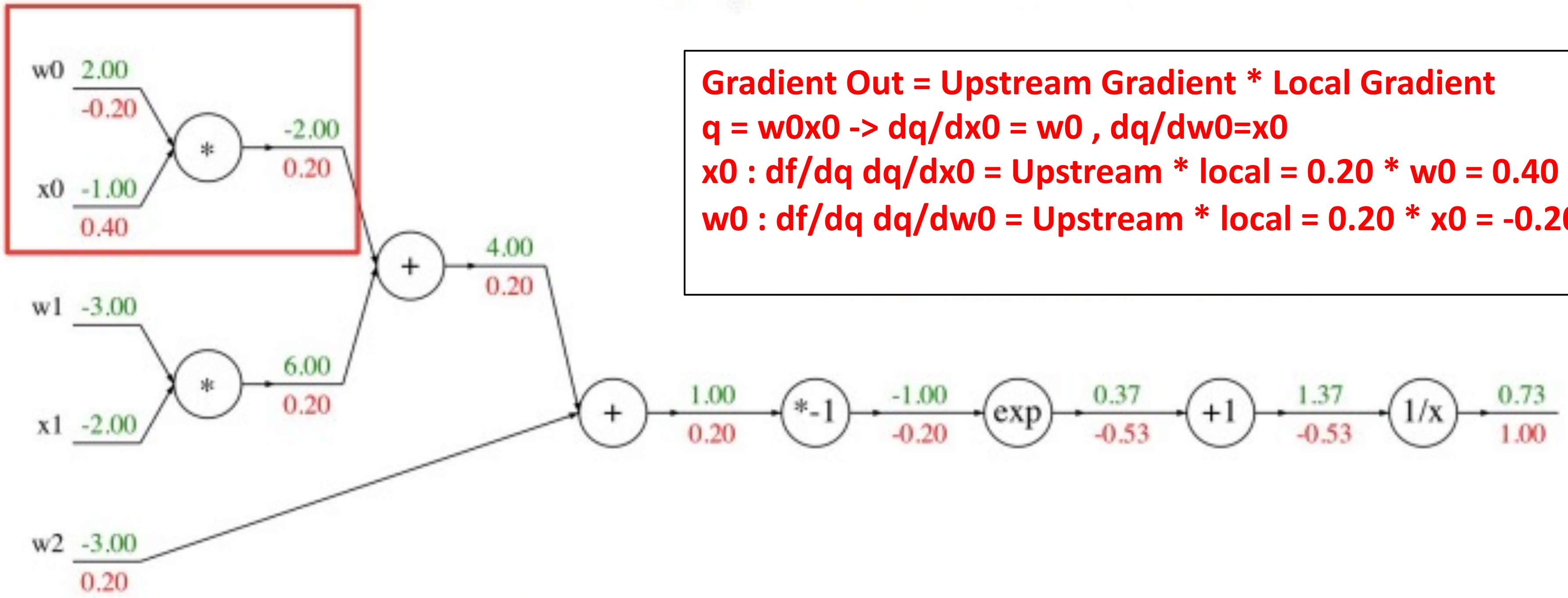
Gradient Out = Upstream Gradient * Local Gradient
q = x+y -> dq/dx = 1 , dq/dy=1
df/dq dq/dx = Upstream * local = 0.20 * 1
df/dq dq/dy = Upstream * local = 0.20 * 1

$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$ | $f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$

$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$ | $f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Another example: $f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$



Gradient Out = Upstream Gradient * Local Gradient
q = w0x0 -> dq/dx0 = w0 , dq/dw0=x0
x0 : df/dq dq/dx0 = Upstream * local = 0.20 * w0 = 0.40
w0 : df/dq dq/dw0 = Upstream * local = 0.20 * x0 = -0.20

$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$

$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$

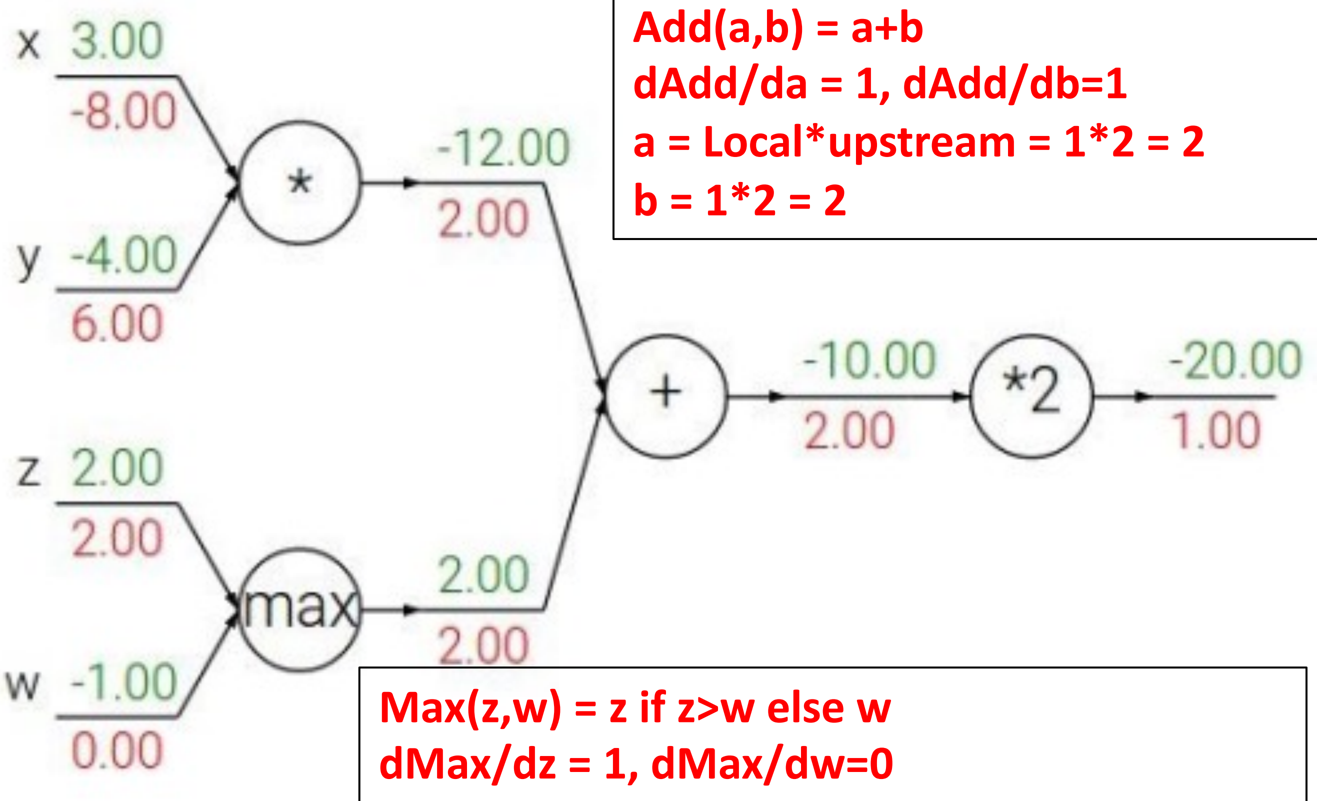
$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$

$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Properties of operators in a backward pass

1. **ADD GATE** : Gradient Distributor
2. **Max Gate** : Gradient Router
3. **Mul gate** : Gradient Switcher



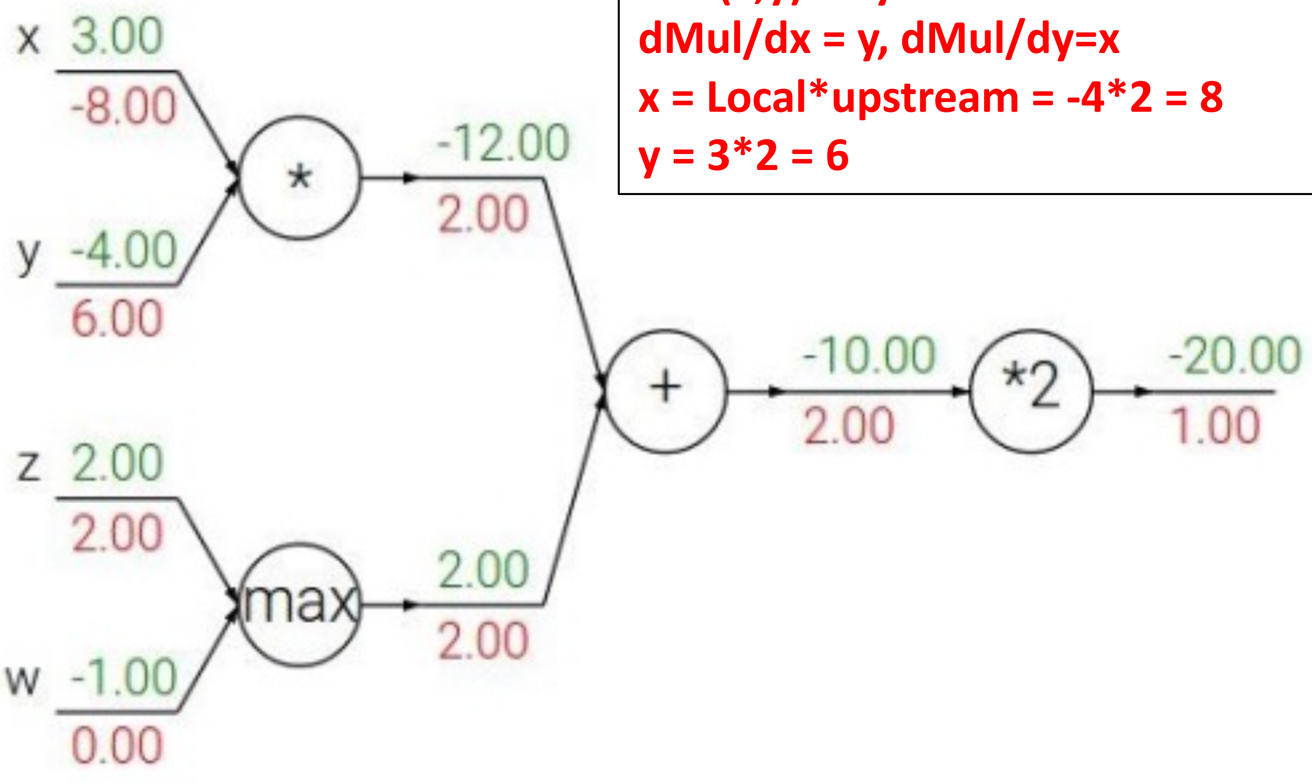
Add(a,b) = a+b
dAdd/da = 1, dAdd/db=1
a = Local*upstream = 1*2 = 2
b = 1*2 = 2

Max(z,w) = z if z>w else w
dMax/dz = 1, dMax/dw=0
z: Local*upstream = 1*2 = 2
w: 0*2=0

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Properties of operators in a backward pass

1. **ADD GATE** : Gradient Distributor
2. **Max Gate** : Gradient Router
3. **Mul gate** : Gradient Switcher



Mul(x,y) = xy
dMul/dx = y, dMul/dy=x
x = Local*upstream = -4*2 = 8
y = 3*2 = 6

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Summary So far

1. Neural Nets are very big computational graphs: Impractical to write down formulas for every node
2. **Backprop** : Recursive application of chain rule to compute grads in a computational graph
3. **Forward pass** : Stores list of ops and saves intermediate results
4. **Backward pass** : Apply chain rule to compute gradient i.e. local grad * upstream grad

Credits: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture4.pdf

Introduction to Neural Networks

Neural Networks

Linear Score function

$$f = Wx$$

2 Layer Neural Net: Linearity + Max

$$f = W_2 max(0, W_1 x)$$

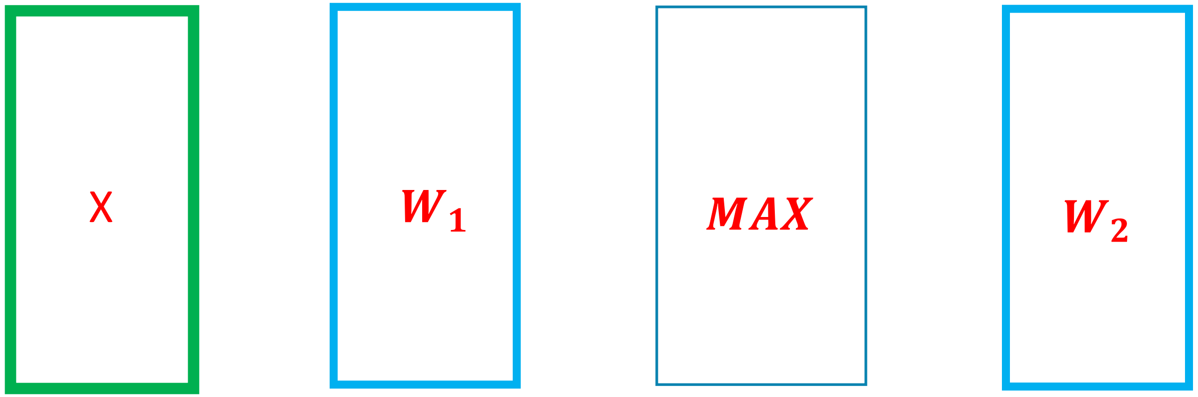
Neural Networks

Linear Score function

$$f = Wx$$

2 Layer Neural Net: Linearity + Max

$$f = W_2 max(0, W_1 x)$$



X, W_1, W_2 **are just matrices**

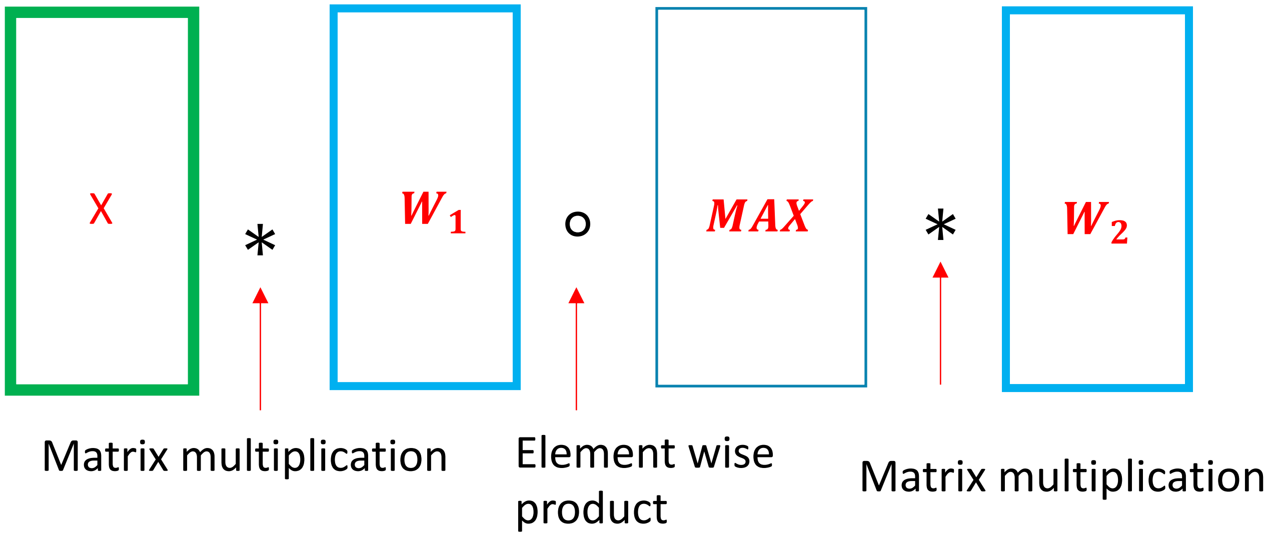
Neural Networks :Operators

Linear Score function

$$f = Wx$$

2 Layer Neural Net: Linearity + Max

$$f = W_2 max(0, W_1 x)$$



W_1, W_2 **are just matrices**

Neural Networks : Shapes

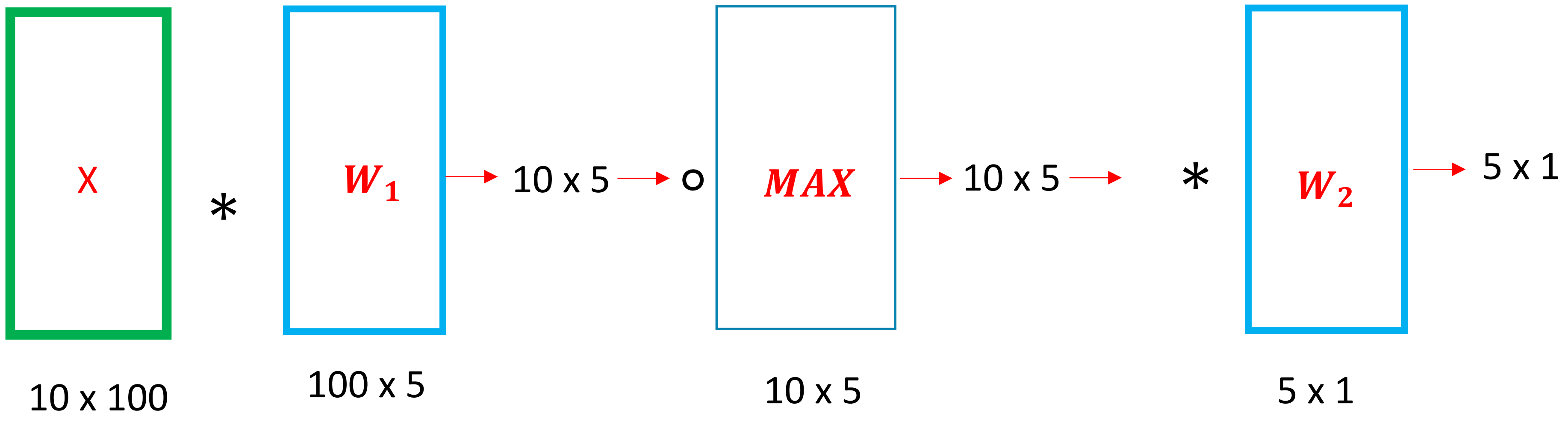
Linear Score function

$$f = Wx$$

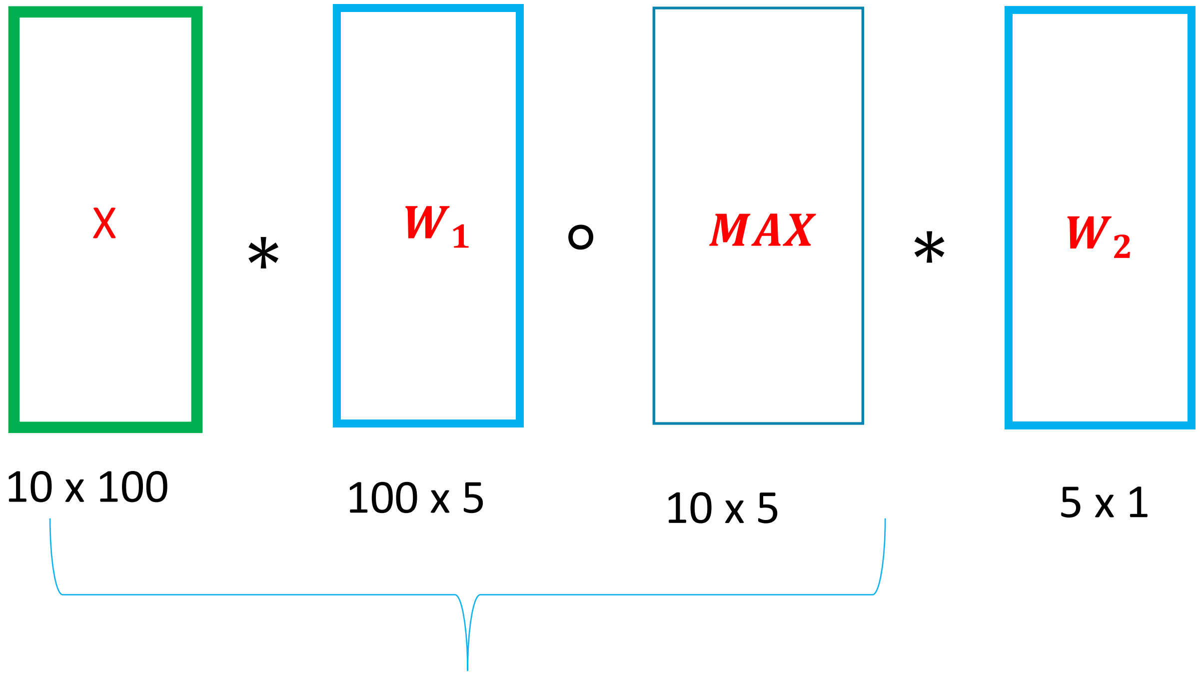
2 Layer Neural Net: Linearity + Max

$$f = W_2 max(0, W_1 x)$$

W_1, W_2 **are just matrices**

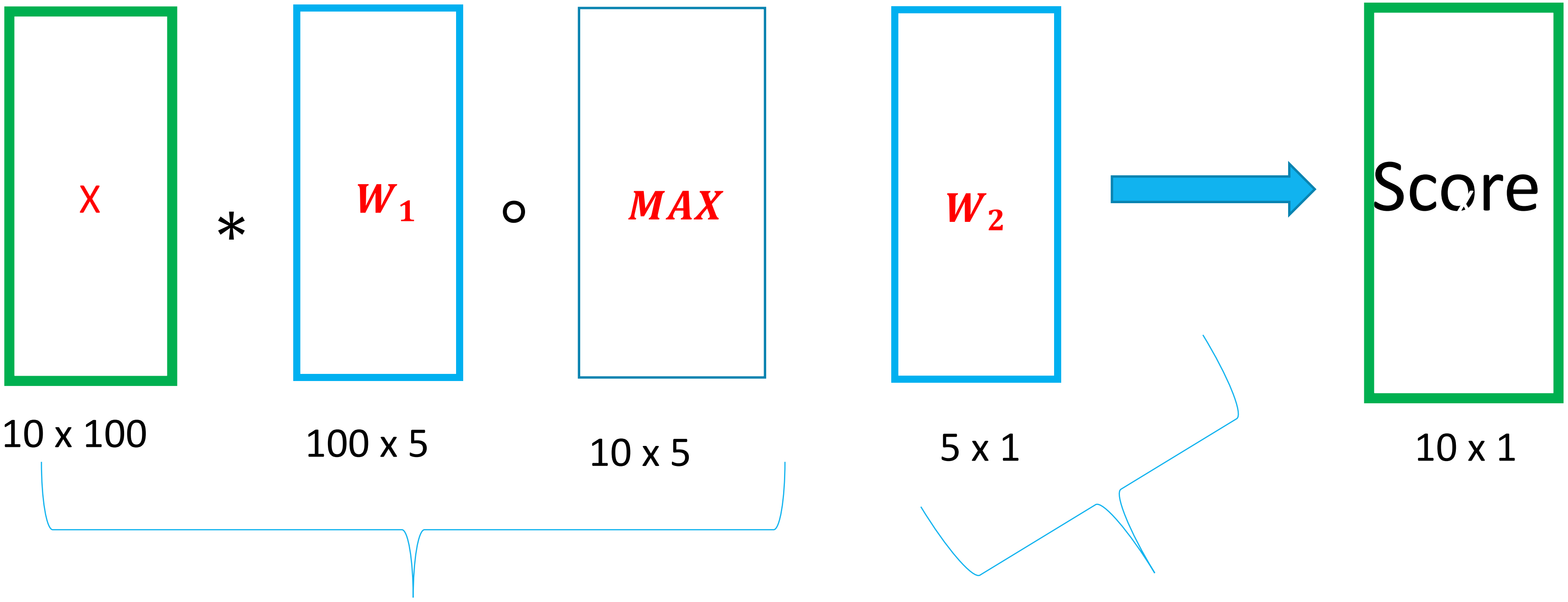


Neural Networks : Layers



Layer1 or Input Layer:
Number of units=5
Activation : Max
Dense(units=5, activation=‘max’)

Neural Networks : Layers



Layer1 or Input Layer:
Number of units=5
Activation : Max

Layer2 or Output Layer:
Number of units=1
Activation : None or Linear

Summary Until now

1. Neural Nets are simply **matrices stacked on top of each other with non linearities** in between
2. Multiple stages of heirarchial computations
3. Lot of different non linear computations to choose from
4. The output of every layer is simply **weighted sum of inputs with our parameters (weight matrices)** followed by some **non linear operation**

Next Steps

1. Tensorflow
2. Hands on Lab

Introduction to Tensorflow 2.x

Agenda

1. Brief History of Tensorflow
2. Overview of Tensorflow components (google colab)

Brief history of Tensorflow

1st Generation : DistBelief

1. Born as DistBelief as proprietary in 2011
2. Google search, translate, photos

2nd Generation : Tensorflow 2015

1. Main Developer Jeff Dean
2. Before Tensorflow 2.0 it was shit
3. They realized Keras API is the way to go and integrated in Tensorflow 2.0

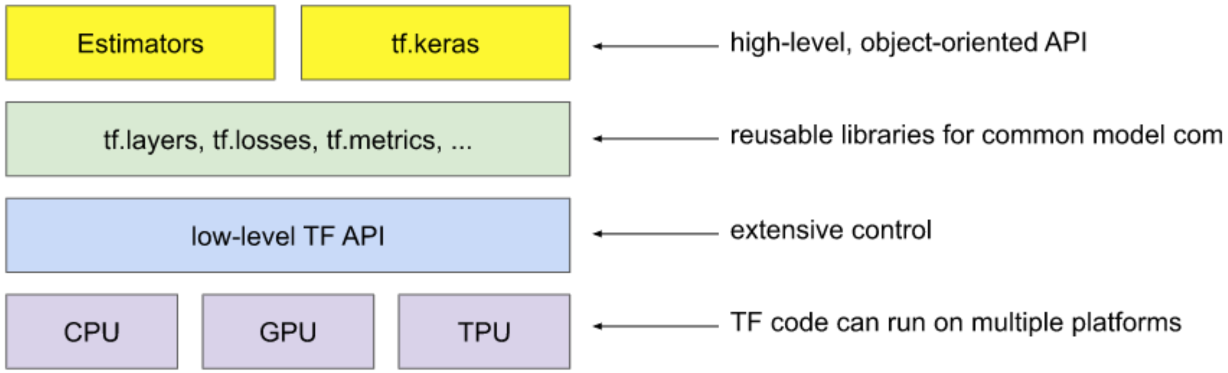
The Big Idea

Tensorflow is a framework composed of:

1. Library defining how to build **Computational Graphs**
2. A runtime for executing these graphs on different hardwares : **CPU,GPU,Microcontrollers**

Tensorflow

1. So many methods, classes and functions.
2. High level APIs : keras and sonnet
3. Lot of packages
4. They even have their own numpy : tf.numpy()



Components of a computational graph in Tensorflow

1. **Tensors** : Representing some data
2. **Variables** : Representing some weight
3. **GradientTape** : Training those weights
4. **Modules** : Building a model
5. **Tf.function** : accelarate the training

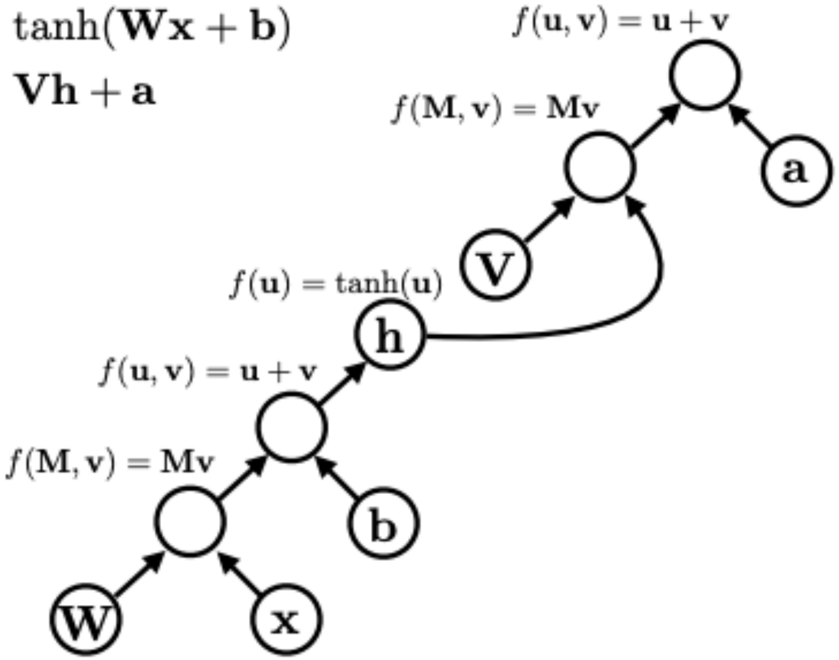
Next Steps

Now we try to understand these components of a computational graph directly in tensorflow.

We will be covering **Tensors, Variables and Automatic Differentiation**

The MLP

$$\mathbf{h} = \tanh(\mathbf{W}\mathbf{x} + \mathbf{b})$$
$$\mathbf{y} = \mathbf{V}\mathbf{h} + \mathbf{a}$$



1. Introduction to Computational Graphs
2. Overview of Tensorflow components
3. Tensorflow Standard API
4. Hands on Tensorflow Lab
    1. Calculus using Tensorflow
    2. Learning to sort using an MLP

What is a computational graph?

Computational Graph is a way to represent mathematical expressions as a directed graph data structure. The nodes represent mathematical operations and the edges represent function argument/data dependency.

$$\boldsymbol{e} = (\boldsymbol{a} + \boldsymbol{b}) * (\boldsymbol{b} + \boldsymbol{1})$$

How to represent the expression as a computational graph ?

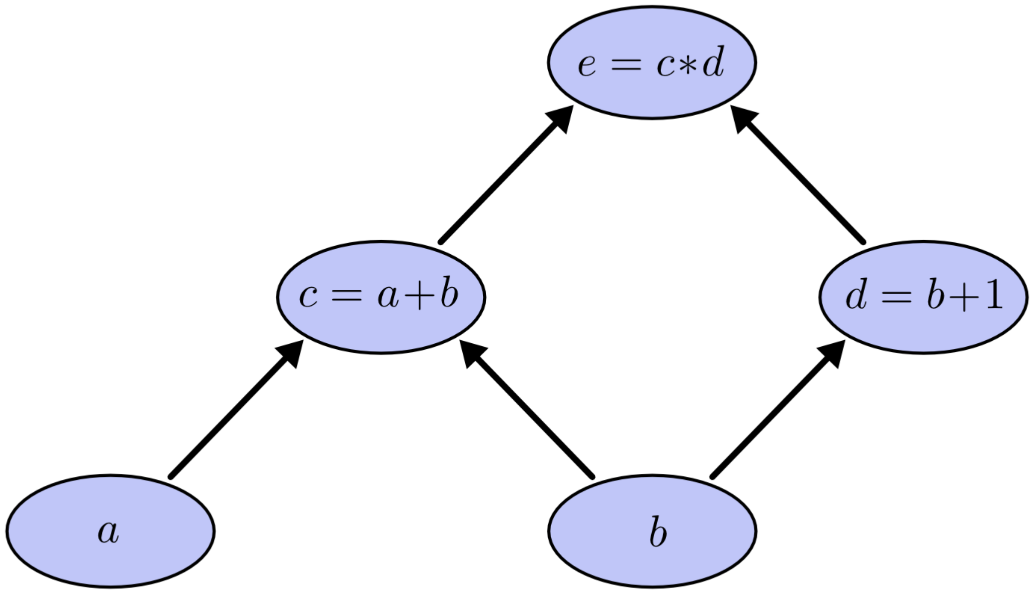
1. 3 Ops : 2 Additions and 1 Multiplication
2. Break down the expression into smaller parts
3. Write c = a+b and d = b+1

What is a computational graph?

Computational Graph is a way to represent mathematical expressions as a directed graph data structure. The nodes represent mathematical operations and the edges represent function argument/data dependency.

$$\boldsymbol{e} = (\boldsymbol{a} + \boldsymbol{b}) * (\boldsymbol{b} + \boldsymbol{1})$$

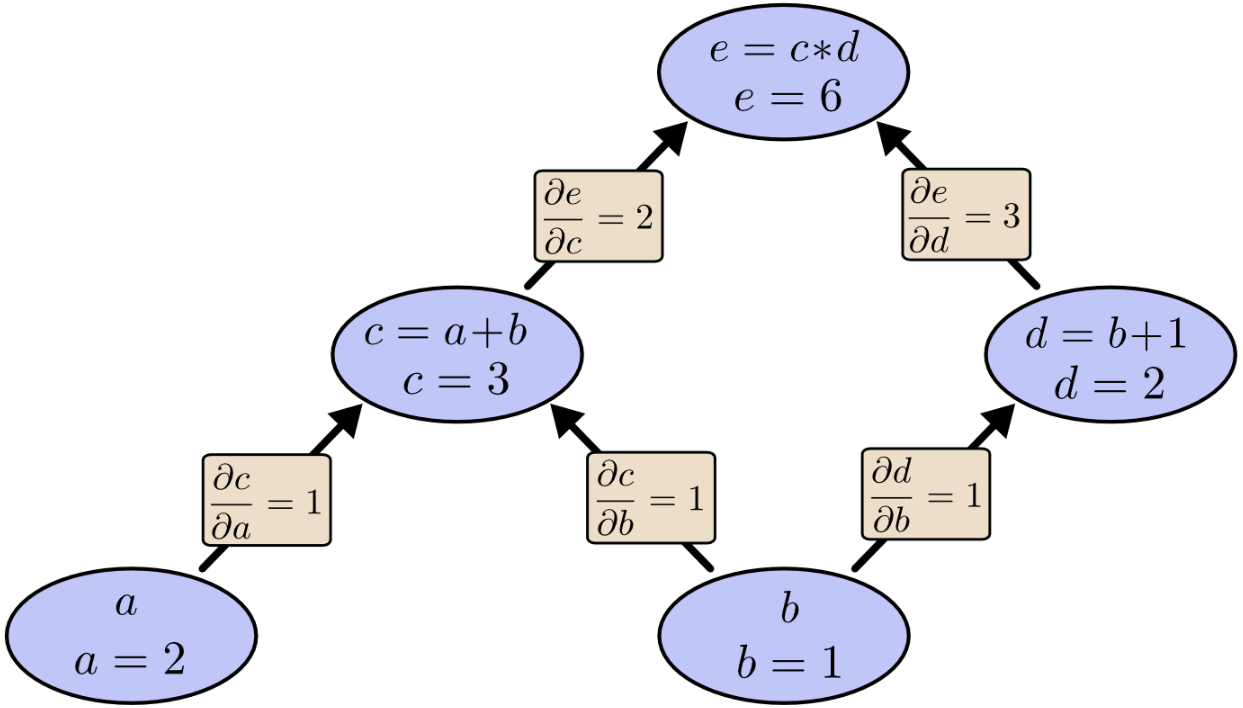
$$\boldsymbol{c} = \boldsymbol{a} + \boldsymbol{b}$$
$$\boldsymbol{d} = \boldsymbol{b} + \boldsymbol{1}$$
$$\boldsymbol{e} = \boldsymbol{c} * \boldsymbol{d}$$



Calculus on a computational graph

Refresher:

1. Sum rule : $\frac{\partial(a+b)}{\partial x} = \frac{\partial a}{\partial x} + \frac{\partial b}{\partial x}$
2. Product rule : $\frac{\partial(uv)}{\partial x} = u\frac{\partial v}{\partial x} + v\frac{\partial u}{\partial x}$
3. Chain rule : $\frac{\partial y}{\partial x} = \frac{\partial y}{\partial u} * \frac{\partial u}{\partial x}$



Partial derivatives :

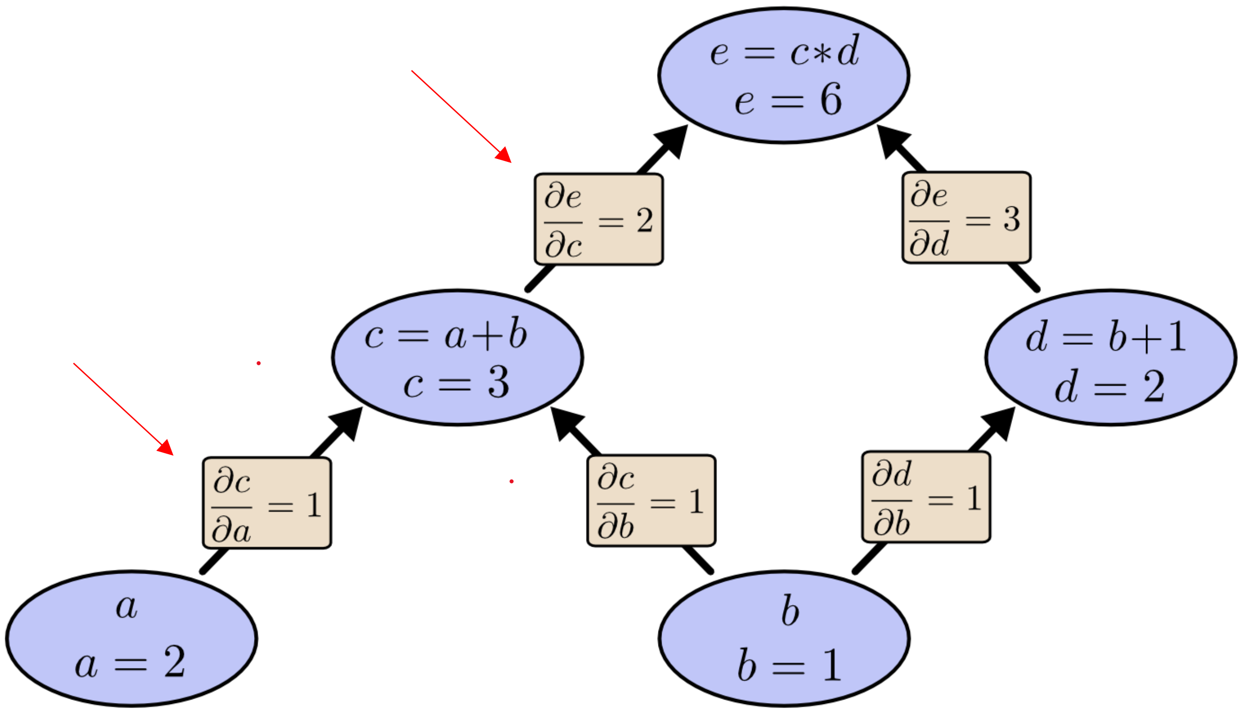
1. How does **e** change with respect to **a** ?
   Analytical solution

$$\frac{\partial e}{\partial a} = \frac{\partial e}{\partial c} * \frac{\partial c}{\partial a} = 2 * 1$$

Calculus on a computational graph

Refresher:

1. Sum rule : $\frac{\partial(a+b)}{\partial x} = \frac{\partial a}{\partial x} + \frac{\partial b}{\partial x}$
2. Product rule : $\frac{\partial(uv)}{\partial x} = u\frac{\partial v}{\partial x} + v\frac{\partial u}{\partial x}$
3. Chain rule : $\frac{\partial y}{\partial x} = \frac{\partial y}{\partial u} * \frac{\partial u}{\partial x}$



Partial derivatives :

1. How does **e** change with respect to **a** ?

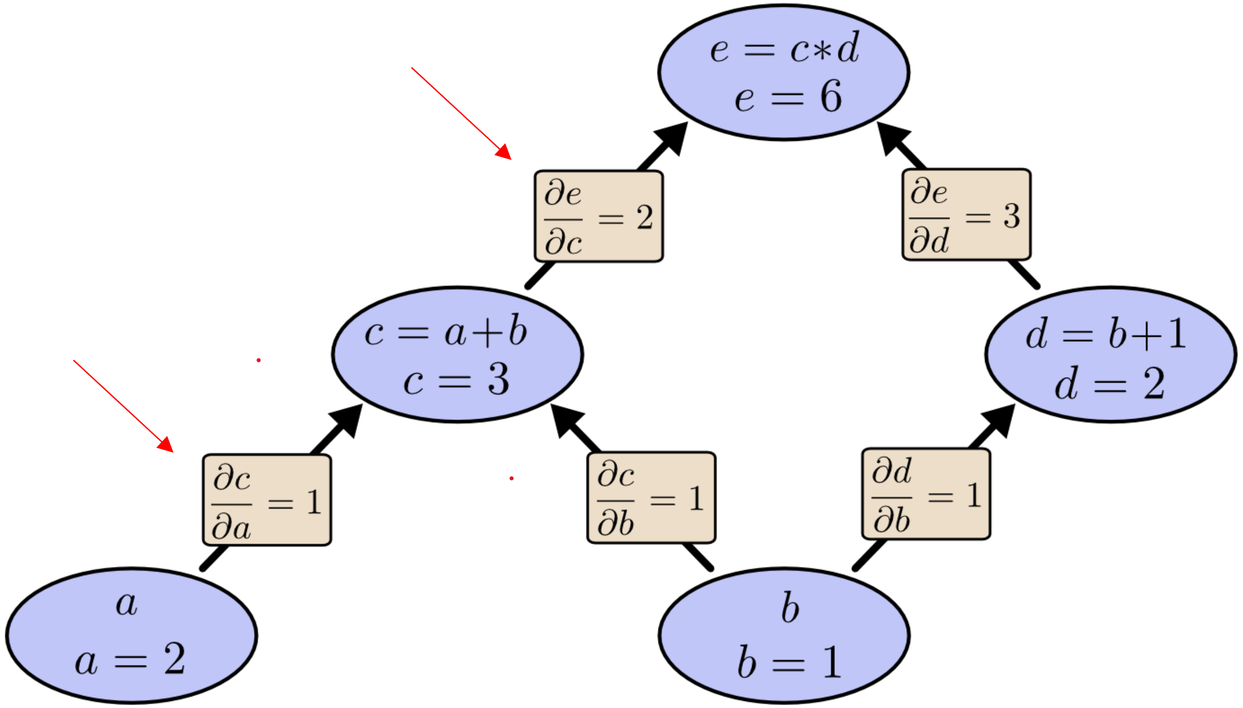
   Graph solution : Multiply the edges !

$$\frac{\partial e}{\partial a} = 2 * 1$$

Calculus on a computational graph

Refresher:

1. Sum rule : $\frac{\partial(a+b)}{\partial x} = \frac{\partial a}{\partial x} + \frac{\partial b}{\partial x}$
2. Product rule : $\frac{\partial(uv)}{\partial x} = u\frac{\partial v}{\partial x} + v\frac{\partial u}{\partial x}$
3. Chain rule : $\frac{\partial y}{\partial x} = \frac{\partial y}{\partial u} * \frac{\partial u}{\partial x}$



Partial derivatives :

1. How does **e** change with respect to **a** ?

   Graph solution : Multiply the edges !

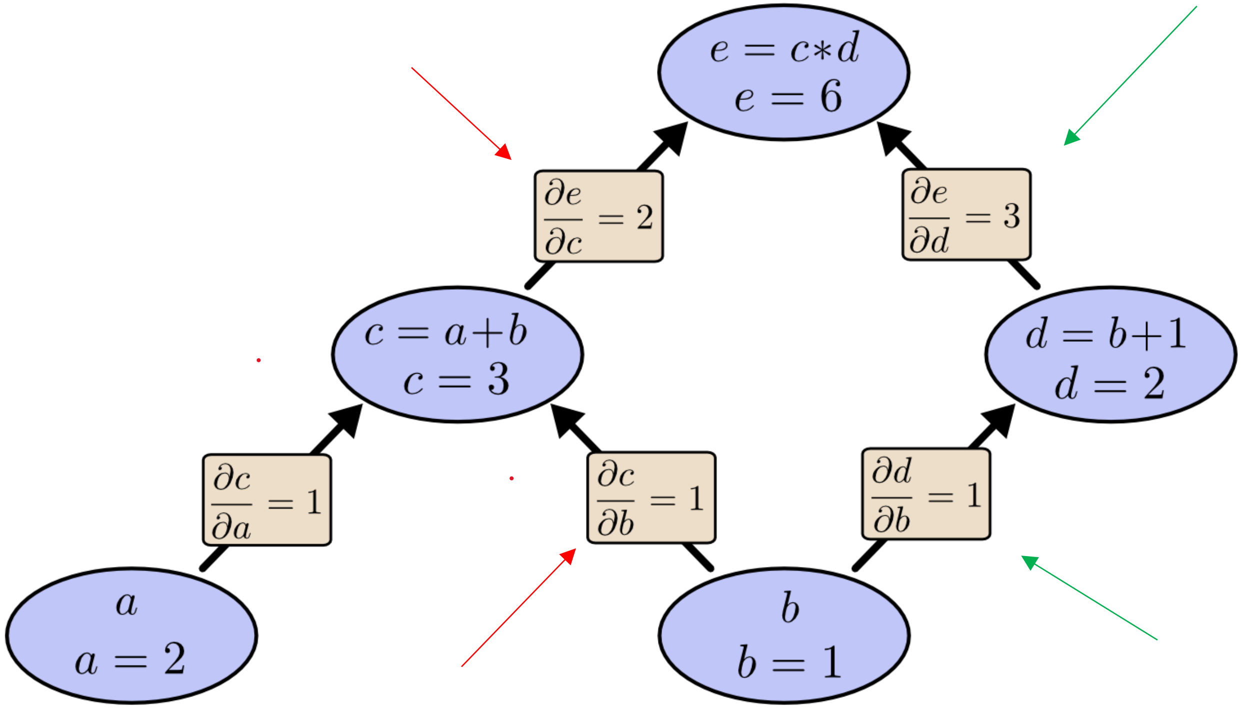
$$\frac{\partial e}{\partial a} = 2 * 1$$

General rule: Sum over all possible paths from one node to the other, multiplying the derivatives on each edge of the path together

Calculus on a computational graph

Refresher:

1. Sum rule : $\frac{\partial(a+b)}{\partial x} = \frac{\partial a}{\partial x} + \frac{\partial b}{\partial x}$
2. Product rule : $\frac{\partial(uv)}{\partial x} = u\frac{\partial v}{\partial x} + v\frac{\partial u}{\partial x}$
3. Chain rule : $\frac{\partial y}{\partial x} = \frac{\partial y}{\partial u} * \frac{\partial u}{\partial x}$



Partial derivatives :

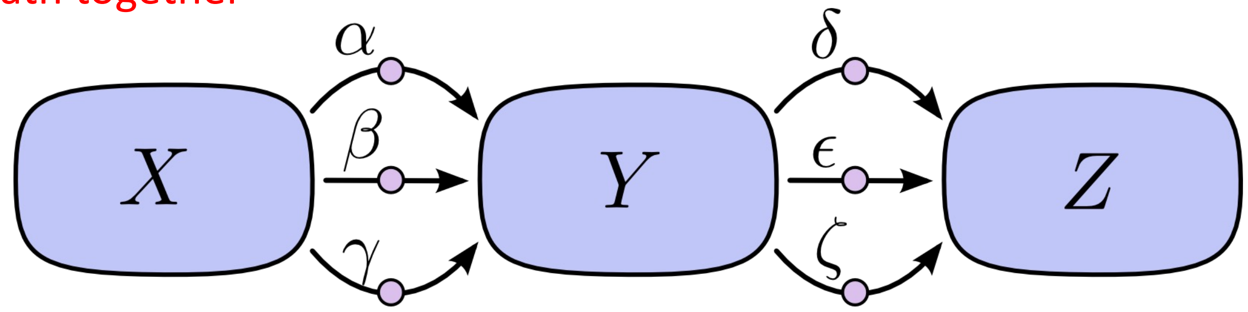
1. How does **e** change with respect to **b** ?

   Graph solution : Multiply the edges !

$$\frac{\partial e}{\partial b} = 2 * 1 + 3 * 2$$

Problems

General rule: Sum over all possible paths from one node to the other, multiplying the derivatives on each edge of the path together

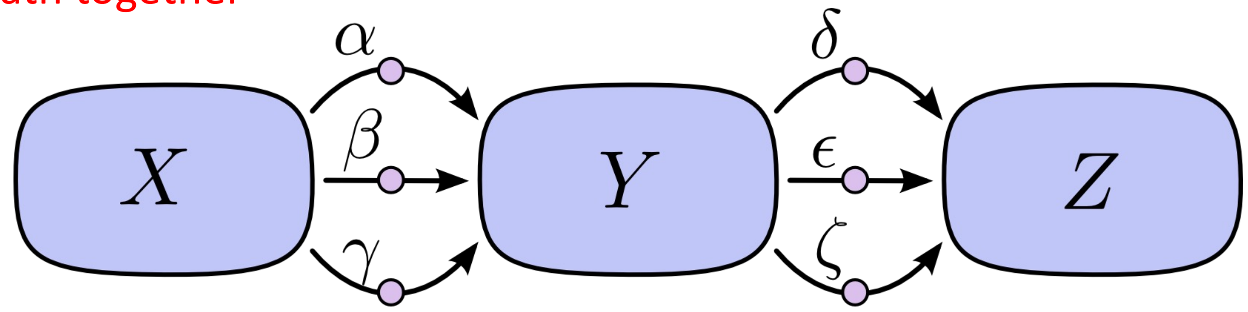


1. Too many possible paths to sum over
   1. X to Y : 3 paths
   2. Y to Z : 3 paths
   3. Total number of paths : 3x3 = 9 paths
   4. For n nodes each with 3 paths : 3^n

$$\frac{\partial \boldsymbol{Z}}{\partial \boldsymbol{X}} = \boldsymbol{\alpha\delta} + \boldsymbol{\alpha\epsilon} + \boldsymbol{\alpha\zeta} + \boldsymbol{\beta\delta} + \boldsymbol{\beta\epsilon} + \boldsymbol{\beta\zeta} + \boldsymbol{\gamma\delta} + \boldsymbol{\gamma\epsilon} + \boldsymbol{\gamma\zeta}$$

Solution

General rule: Sum over all possible paths from one node to the other, multiplying the derivatives on each edge of the path together



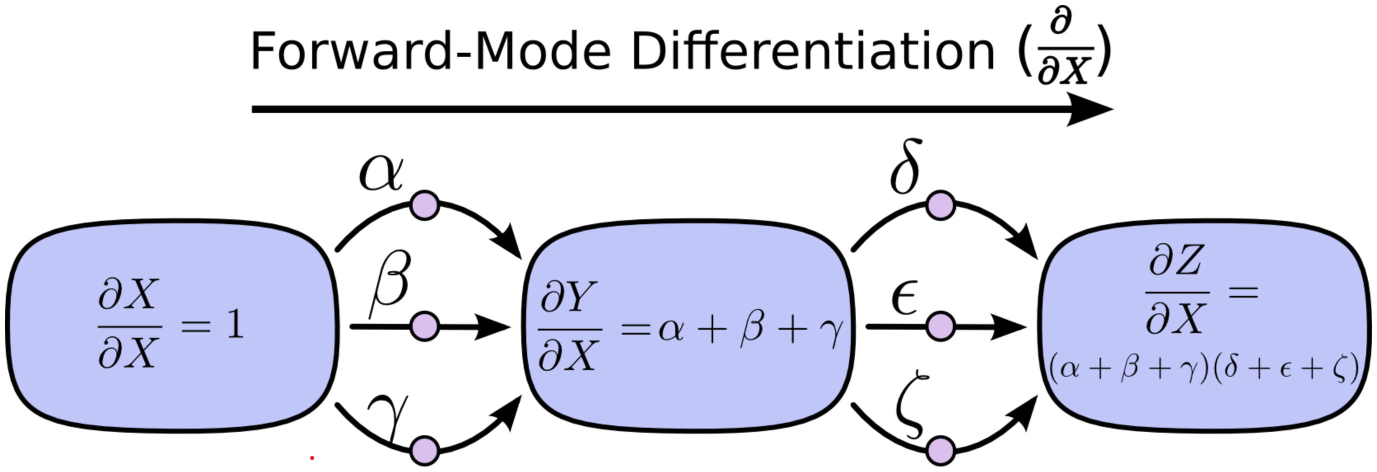
1. Too many possible paths to sum over
    1. X to Y : 3 paths
    2. Y to Z : 3 paths
    3. Total number of paths : 3x3 = 9 paths
    4. For n nodes each with 3 paths : 3^n

Refactor the paths !

$$\frac{\partial \boldsymbol{Z}}{\partial \boldsymbol{X}} = (\boldsymbol{\alpha} + \boldsymbol{\beta} + \boldsymbol{\gamma})(\boldsymbol{\delta} + \boldsymbol{\epsilon} + \boldsymbol{\zeta})$$

Forward mode differentiation

1. Calculate the derivative of each node with respect to node X
2. That is apply the operator $\frac{\partial}{\partial X}$ to every node efficiently
3. Merge all the incoming paths of a node



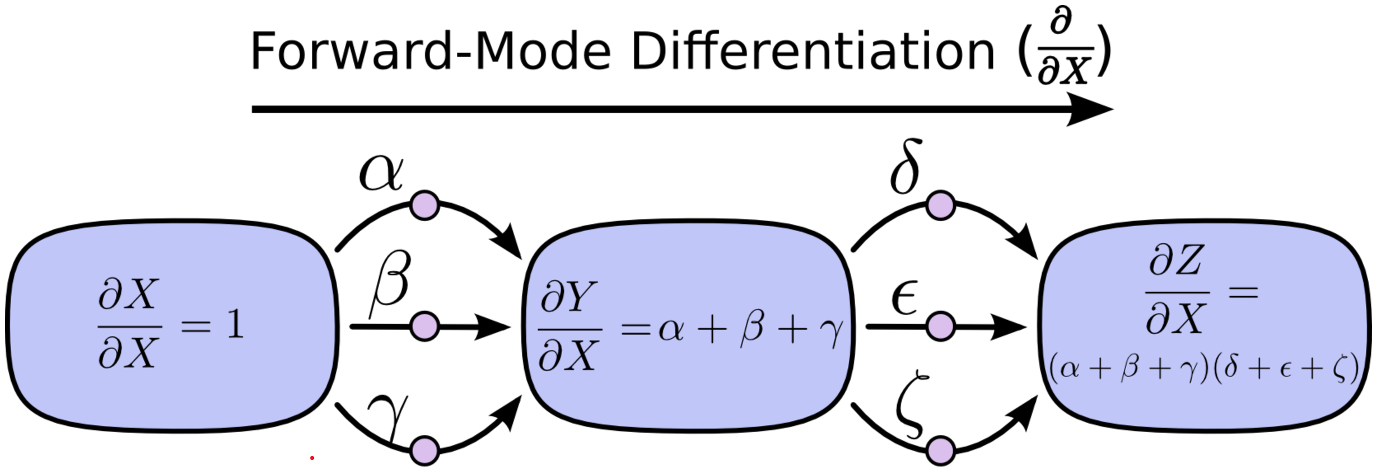
How to do it ?

1. Express each node as sum of incoming paths
   1. $\frac{\partial X}{\partial X} = 1$ because no incoming edges
   2. $\frac{\partial Y}{\partial X} = \alpha + \beta + \gamma$ because three incoming edges from X to Y
   3. $\frac{\partial Z}{\partial Y} = \delta + \epsilon + \zeta$ because three incoming edges from Y to Z

$$\frac{\partial Z}{\partial X} = ??$$

Forward mode differentiation

1. Calculate the derivative of each node with respect to node X
2. That is apply the operator $\frac{\partial}{\partial X}$ to every node efficiently
3. Merge all the incoming paths of a node
4. Start from input nodes and move towards output



How to do it ?

1. Express each node as sum of incoming paths
   1. $\frac{\partial X}{\partial X} = 1$ because no incoming edges
   2. $\frac{\partial Y}{\partial X} = \alpha + \beta + \gamma$ because three incoming edges from X to Y
   3. $\frac{\partial Z}{\partial Y} = \delta + \epsilon + \zeta$ because three incoming edges from Y to Z

$$\frac{\partial Z}{\partial X} = \frac{\partial Z}{\partial Y} * \frac{\partial Y}{\partial X} = (\delta + \epsilon + \zeta)(\alpha + \beta + \gamma)$$

Just multiply nodes in the path
X*Y*Z

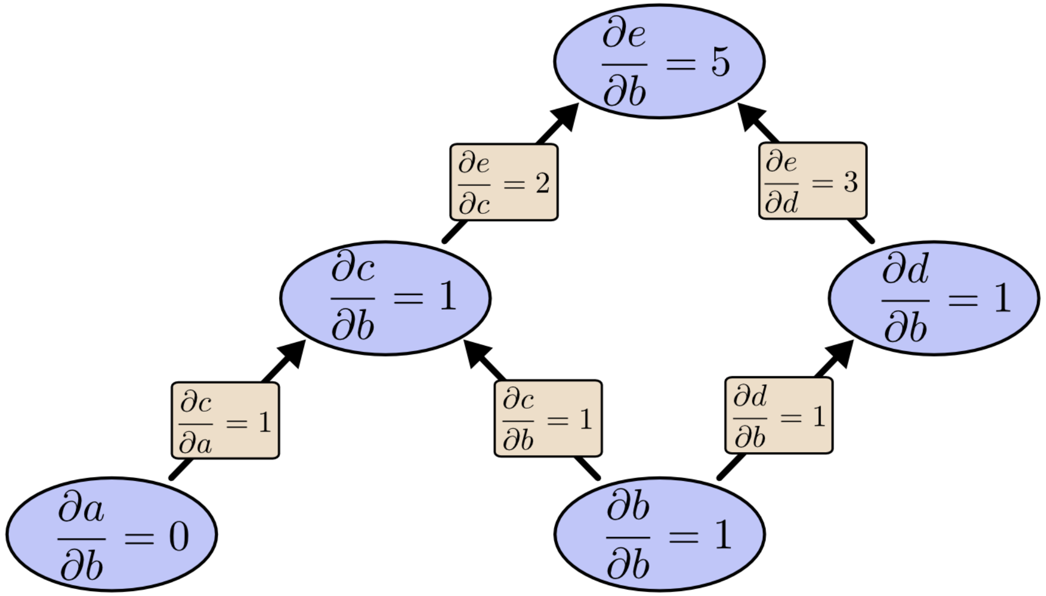
CGnal

Example

$$\boldsymbol{Operator = \frac{\partial(all\,nodes)}{\partial b}}$$

$$\frac{\partial e}{\partial b} = \frac{\partial e}{\partial c} * \frac{\partial c}{\partial d} + \frac{\partial c}{\partial d} * \frac{\partial d}{\partial b} = 2 * 1 + 3 * 1$$

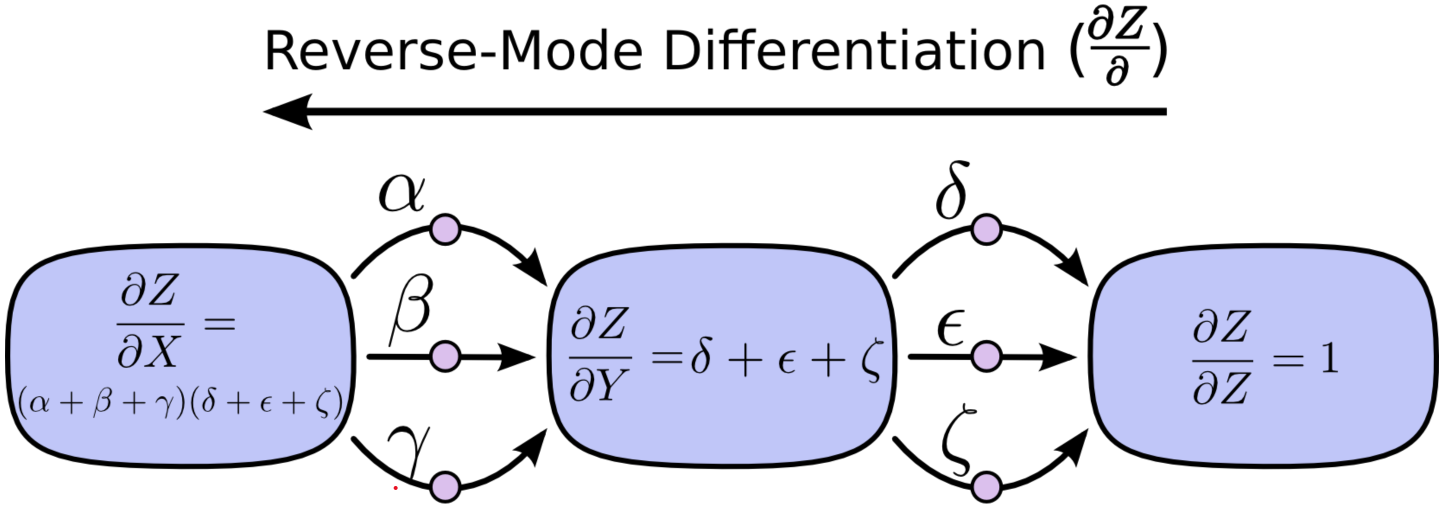
$$\frac{\partial c}{\partial b} = 1$$
$$\frac{\partial d}{\partial b} = 1$$
$$\frac{\partial a}{\partial b} = 0$$



We get derivative of our output ***e*** with respect to only a single input ***b***
For ***n*** inputs we must traverse the tree ***n*** number of times!

Reverse mode differentiation

1. Calculate the derivative of output node Z with respect to node nodes
2. That is apply the operator $\frac{\partial Z}{\partial}$ to every node efficiently
3. Merge all the outgoing paths of a node
4. Start from output node and move towards input node

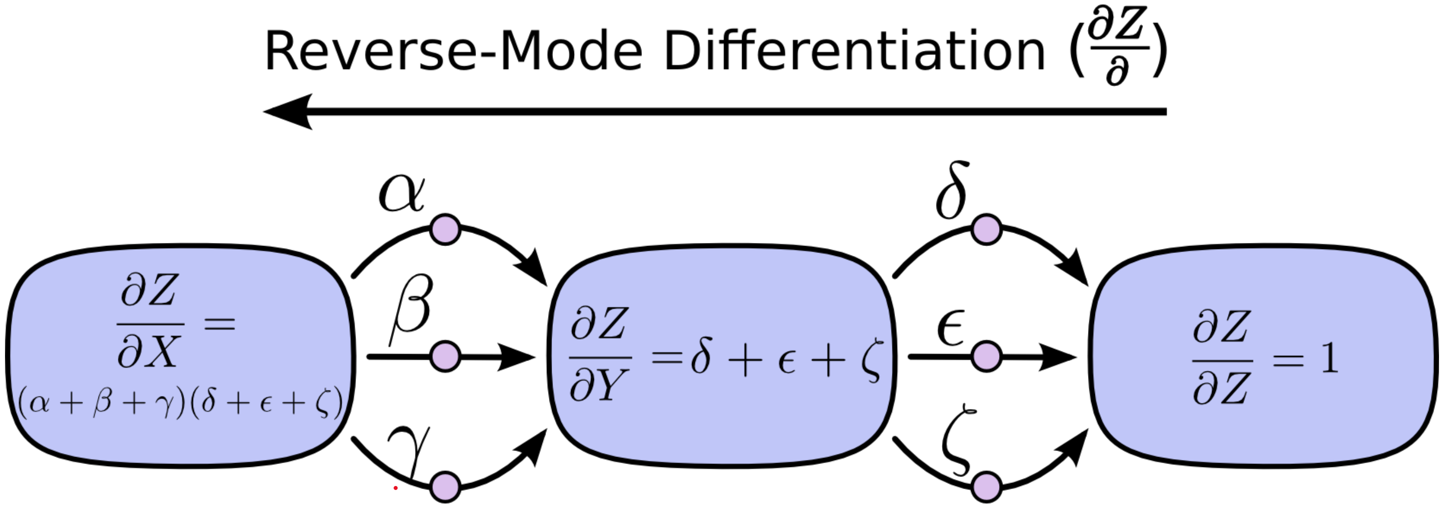


How to do it ?

1. Express each node as sum of outgoing paths
   1. $\frac{\partial Z}{\partial Z} = 1$ because no outgoing edges
   2. $\frac{\partial Z}{\partial Y} = \delta + \epsilon + \zeta$ because three outgoing edges from Y to Z
   3. $\frac{\partial Y}{\partial X} = \alpha + \beta + \gamma$ because three outgoing edges from X to Y

$$\frac{\partial Z}{\partial X} = ?$$

Reverse mode differentiation



1. Calculate the derivative of output node Z with respect to node nodes
2. That is apply the operator $\frac{\partial Z}{\partial}$ to every node efficiently
3. Merge all the outgoing paths of a node
4. Start from output node and move towards input node

How to do it ?

1. Express each node as sum of outgoing paths
   1. $\frac{\partial Z}{\partial Z} = 1$ because no outgoing edges
   2. $\frac{\partial Z}{\partial Y} = \boldsymbol{\delta + \epsilon + \zeta}$ because three outgoing edges from Y to Z
   3. $\frac{\partial Y}{\partial X} = \boldsymbol{\alpha + \beta + \gamma}$ because three outgoing edges from X to Y

$$\frac{\partial Z}{\partial X} = \frac{\partial Z}{\partial Y} * \frac{\partial Y}{\partial X} = (\boldsymbol{\delta + \epsilon + \zeta})(\boldsymbol{\alpha + \beta + \gamma})$$

Just multiply nodes in the path
Z*Y*X

Example

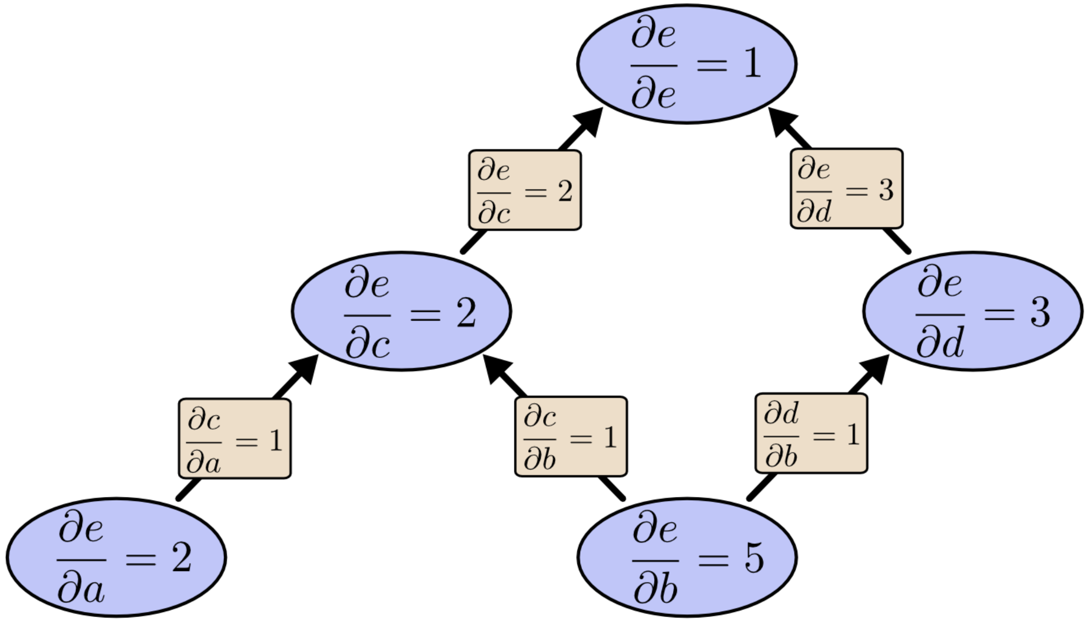
$$\boldsymbol{Operator} = \frac{\partial \boldsymbol{e}}{\partial(\boldsymbol{all\, nodes})}$$

$$\boldsymbol{Node\ c}: \frac{\partial \boldsymbol{e}}{\partial \boldsymbol{c}} = \mathbf{2}$$

$$\boldsymbol{Node\ d}: \frac{\partial \boldsymbol{e}}{\partial \boldsymbol{d}} = \mathbf{3}$$

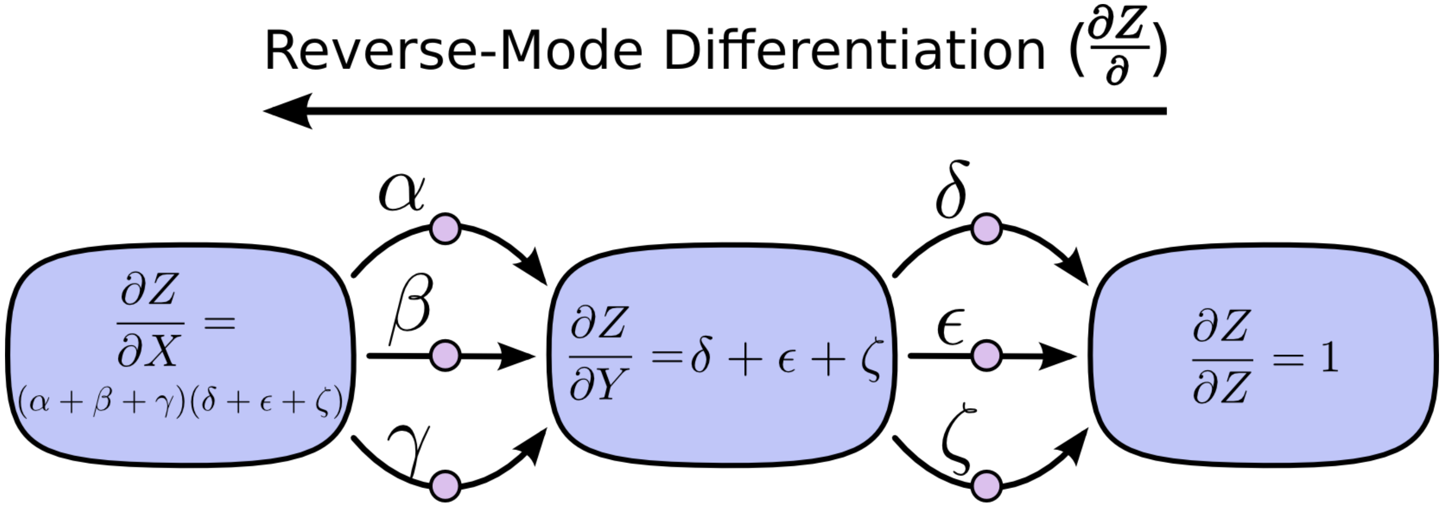
$$\boldsymbol{Node\ a}: \frac{\partial \boldsymbol{e}}{\partial \boldsymbol{a}} = \frac{\partial \boldsymbol{e}}{\partial \boldsymbol{c}} * \frac{\partial \boldsymbol{c}}{\partial \boldsymbol{a}} = \mathbf{2} * \mathbf{1}$$

$$\boldsymbol{Node\ b}: \frac{\partial \boldsymbol{e}}{\partial \boldsymbol{b}} = \frac{\partial \boldsymbol{e}}{\partial \boldsymbol{c}} * \frac{\partial \boldsymbol{c}}{\partial \boldsymbol{b}} + \frac{\partial \boldsymbol{e}}{\partial \boldsymbol{d}} * \frac{\partial \boldsymbol{d}}{\partial \boldsymbol{b}} = \mathbf{2} * \mathbf{1} + \mathbf{3} * \mathbf{1}$$



We get derivative of our output ***e*** with respect to all the input nodes ***a & b in a single traversal***
For ***n*** inputs we must traverse the tree just ***1*** time!
For million inputs and a single output (Neural Net) : 1 million times faster than Forward mode differentiation

So what about it ?



1. Reverse mode differentitation is backpropagation !
2. Backpropagation is just the chain rule !
3. Calculating gradients on big computational graphs with million input nodes and 1 output node is simply chain rule and damn smart.
4. Calculating derivatives is really cheap. That’s the inherent nature of calculus.

Other Resources

1. Colah's blog : https://colah.github.io/posts/2015-08-Backprop/
2. http://neuralnetworksanddeeplearning.com/chap2.html
3. Andrew Ng's lectures : https://youtu.be/yXcQ4B-YSjQ

CGnal