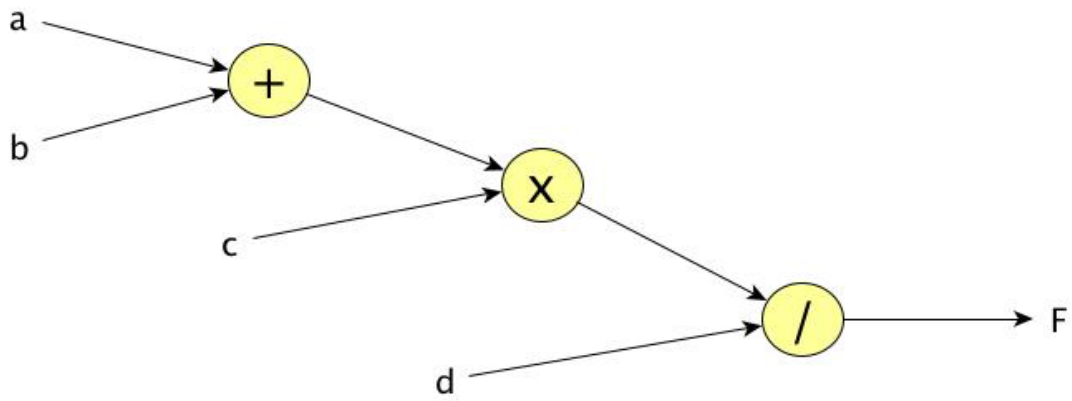
BACKPROPAGATION BLOG

This is part 1 of a series of blogs where we discuss about backpropagation. In today's blog, we will discuss about computational graphs and backpropagation with the help of a simple example. In the next blog, we will discuss a more complex example along with its python implementation.

To understand this blog, the readers must have some preknowledge of calculus. Before we discuss about the backpropagation algorithm, let's discuss what computational graphs are.

A computational graph can be used to represent any mathematical function in the form of a directed graph where the nodes represent the steps of computation. Let us see an example of a computational graph. Here, is a F: $\mathbb{R}^4 \rightarrow \mathbb{R}$ is a function where $F(a, b, c, d) = ((a + b) \times c) \div d$



What is backpropagation?

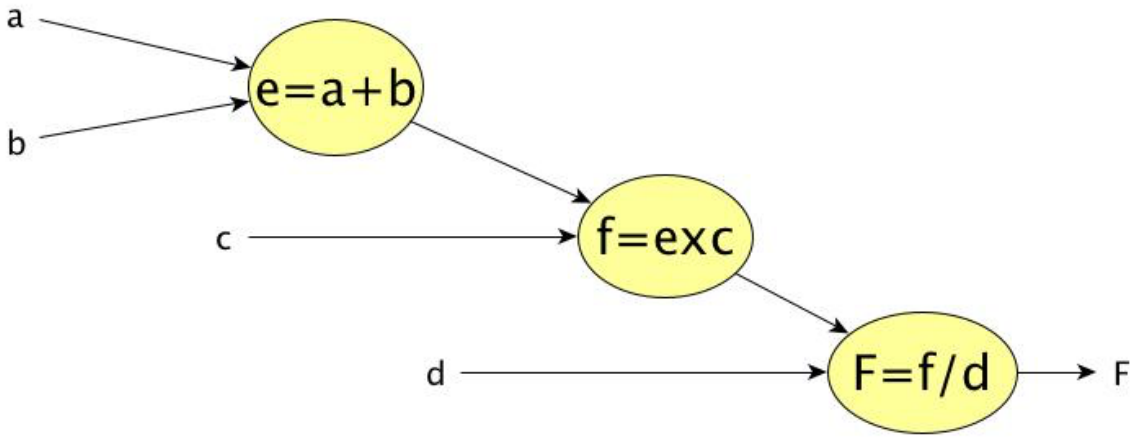
Backpropagation is a method used in artificial neural network to calculate gradients to be used in the calculation of weights. Backpropagation is also known as "the backward propagation of errors," since an error is computed at the output and distributed backwards through the layers of the network. It is a special case of a technique called as automatic differentiation.

Let us see how backpropagation is carried out using the above example. Let a = 5, b = 1, c = -2, d = -3

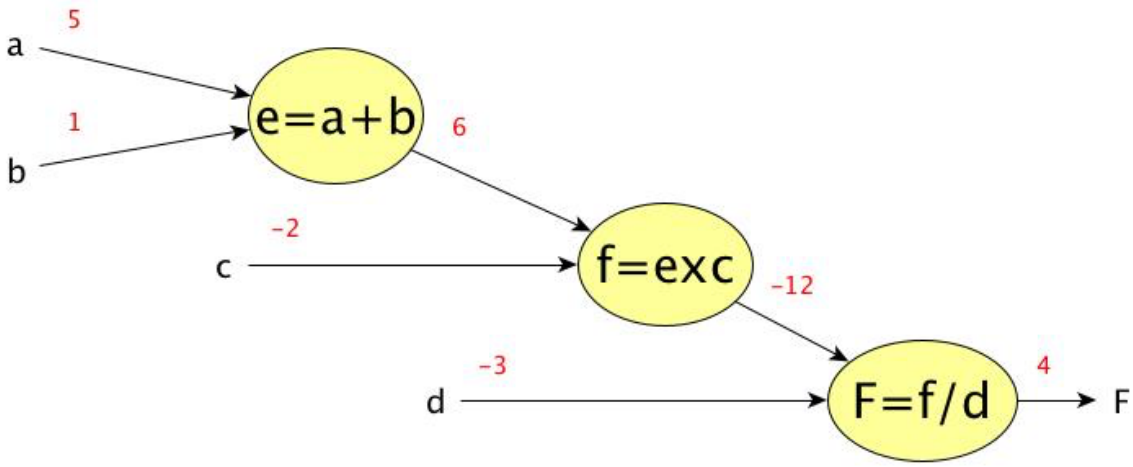
Step 1: Represent the function using a computational graph.
The above function has three operations: +, × and ÷. We can convert the function in a way that every node's output has a variable.

$$e = a + b$$
$$f = e \times c$$
$$g = f \div d$$

The above computation graph is modified as:



Step 2: Perform forward pass on the computational graph
Here, given the value of the variable, we will calculate the values of all the intermediate nodes and the final output. The value in red color represents these values.



The advantage of computational graph is that we can use backpropagation algorithm that recursively uses the chain rule of differentiation to compute the derivative with respect to every variable in the computational graph.

Step 3: Perform backpropagation algorithm
We are going to start at the end of the computational graph and calculate the derivative of F with respect to all input and intermediate variables. This is done by recursively moving backwards through the computational graph.

1) **Calculation of** $\frac{\partial F}{\partial F}$

This calculation is very trivial. The answer is 1.

2) Calculation of $\frac{\partial F}{\partial f}$

We know by differentiation, that $\frac{d(Ax)}{dx} = A$. (Rule 1)

Here, $\frac{\partial F}{\partial f} = \frac{\partial(\frac{f}{d})}{\partial f} = \frac{1}{d} =$ -3

3) Calculation of $\frac{\partial F}{\partial d}$

We know by differentiation, that $\frac{d(x^n)}{dx} = nx^{n-1}$. (Rule 2)

Here, $\frac{\partial F}{\partial d} = \frac{\partial(\frac{f}{d})}{\partial d} = -\frac{f}{d^{-2}} = -1 \times \frac{-12}{-3^{-2}} = 108$

4) Calculation of $\frac{\partial F}{\partial e}$ and $\frac{\partial F}{\partial c}$

Here, e and c are not directly connected to F. We need to apply chain rule to calculate $\frac{\partial F}{\partial e} and \ \frac{\partial F}{\partial c}$. As e and c are used in the calculation of f, we will first calculate the following gradients: $\frac{\partial f}{\partial e} and \ \frac{\partial f}{\partial c}$.

$$\frac{\partial f}{\partial e} = c = -2$$

$$\frac{\partial f}{\partial c} = e = 6$$

We will now use chain rule as follows:

$$\frac{\partial F}{\partial e} = \frac{\partial F}{\partial f} \times \frac{\partial f}{\partial e} = -3 \times -2 = 6$$

$$\frac{\partial F}{\partial c} = \frac{\partial F}{\partial f} \times \frac{\partial f}{\partial c} = -3 \times 6 = -18$$

5) Calculation of $\frac{\partial F}{\partial a}$ and $\frac{\partial F}{\partial b}$

Here again a and b are not directly connected to F. We need to apply chain rule to calculate $\frac{\partial F}{\partial a} and \ \frac{\partial F}{\partial b}$. As a and b are used in the calculation of e, we will first calculate the following gradients: $\frac{\partial e}{\partial a} and \ \frac{\partial e}{\partial b}$.

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

$$\frac{\partial e}{\partial b} = 1$$

We will now use chain rule as follows:

$$\frac{\partial F}{\partial a} = \frac{\partial F}{\partial e} \times \frac{\partial e}{\partial a} = 6 \times 1 = 6$$

$$\frac{\partial F}{\partial b} = \frac{\partial F}{\partial e} \times \frac{\partial e}{\partial b} = 6 \times 1 = 6$$

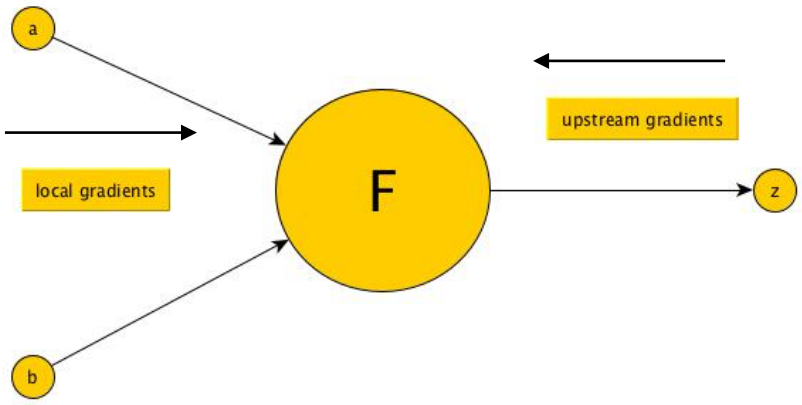
A more expanded form of $\frac{\partial F}{\partial a}$ is $\frac{\partial F}{\partial a} = \frac{\partial F}{\partial f} \times \frac{\partial f}{\partial e} \times \frac{\partial e}{\partial a}$

Similarly, for $\frac{\partial F}{\partial b}$ is $\frac{\partial F}{\partial b} = \frac{\partial F}{\partial f} \times \frac{\partial f}{\partial e} \times \frac{\partial e}{\partial b}$

Let's look at a more general case to understand how the chain rule works:

Every node has some inputs and outputs. Here, a and b are the inputs and z is the output after some operation F. Let us say we are calculating some final variable L. While performing backpropagation on this node, we will have to calculate two type of gradients:

1) Upstream gradient $\frac{\partial L}{\partial z}$
2) Local gradients $\frac{\partial z}{\partial a}$ and $\frac{\partial z}{\partial b}$



Hence to calculate $\frac{\partial l}{\partial a}$ and $\frac{\partial L}{\partial b}$, we will do so by the following formula:

$$\frac{\partial L}{\partial X} = UPSTREAM \times LOCAL$$

Hence, $\frac{\partial L}{\partial a} = \frac{\partial L}{\partial z} \times \frac{\partial z}{\partial a}$ and $\frac{\partial L}{\partial b} = \frac{\partial L}{\partial z} \times \frac{\partial z}{\partial b}$