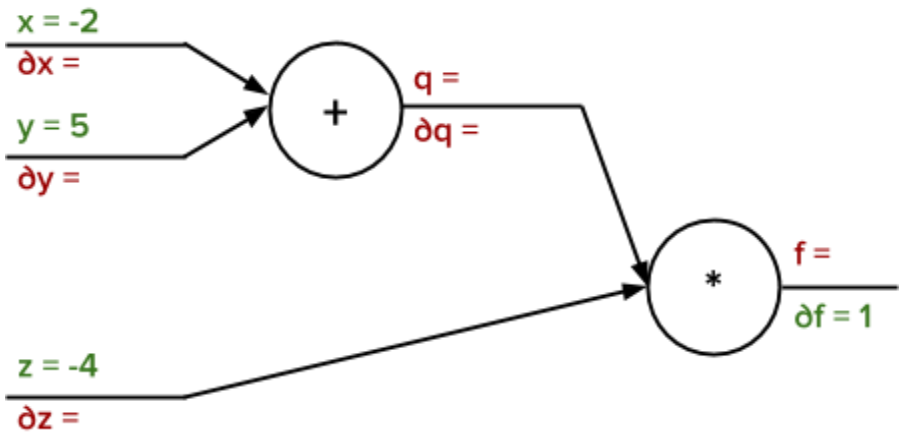
Section 3: Gradient Descent & Backpropagation
Practice Problems

Problem 1. Computation Graph Review

Let's assume we have a simple function $f(x, y, z) = (x + y)z$. We can break this up into the equations $q = x + y$ and $f(x, y, z) = qz$. Using this simplified notation, we can also represent this equation as a computation graph:



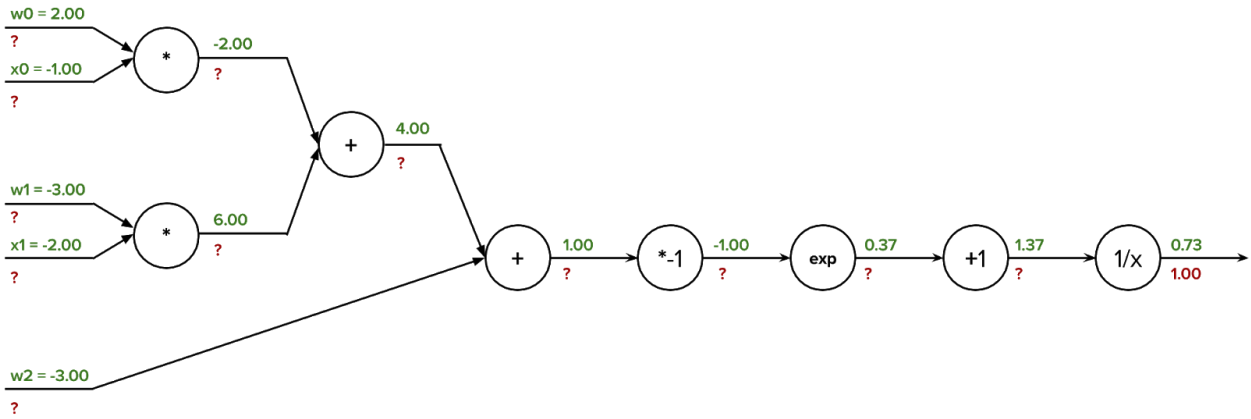
Now let's assume that we are evaluating this function at ***x = -2***, ***y = 5***, and ***z = -4***. In addition let the value of the upstream gradient (gradient of the loss with respect to our function, $\partial L/\partial f$) equal **1**. These are filled out for you in the computation graph.

Solve for the following values, both symbolically (without plugging in specific values of x/y/z), and evaluated at ***x = -2***, ***y = 5***, ***z = -4***, and ***∂L/∂f = 1***:

Symbolically	Evaluated:
1. $\partial f / \partial q =$	$\partial f / \partial q =$
2. $\partial q / \partial x =$	$\partial q / \partial x =$
3. $\partial q / \partial y =$	$\partial q / \partial y =$
4. $\partial f / \partial z =$	$\partial f / \partial z =$
5. $\partial f / \partial x =$	$\partial f / \partial x =$
6. $\partial f / \partial y =$	$\partial f / \partial y =$

Problem 2. Computation Graphs on Steroids

Now let's perform backpropagation through a single neuron of a neural network with a sigmoid activation. Specifically, we will define the pre-activation $z = w_o x_o + w_1 x_1 + w_2$ and we will define the activation value $\alpha = \sigma(z) = 1 / (1 + e^{-z})$. The computation graph is visualized below:



In the graph we've filled out the **forward activations**, on the top of the lines, as well as the upstream gradient (gradient of the loss with respect to our neuron, $\partial L / \partial \alpha$). Use this information to compute the rest of the gradients (labelled with **question marks**) throughout the graph.

Hint: A calculator may be helpful here.

Finally, report the symbolic gradients with respect to the input parameters, $x_o,\ x_1,\ w_0,\ w_1, w_2$:

1. $\partial \alpha / \partial x_0 =$
2. $\partial \alpha / \partial w_0 =$
3. $\partial \alpha / \partial x_1 =$
4. $\partial \alpha / \partial w_1 =$
5. $\partial \alpha / \partial w_2 =$

Problem 3. Backpropagation Basics: Dimensions & Derivatives

Let's assume we have a two layer neural network, as defined below:

$$z_1 = W_1 x^{(i)} + b_1$$
$$a_1 = ReLU(z_1)$$
$$z_2 = W_2 a_1 + b_2$$
$$\hat{y}^{(i)} = \sigma(z_2)$$
$$L^{(i)} = y^{(i)} * \log(\hat{y}^{(i)}) + (1 - y^{(i)}) * \log(1 - \hat{y}^{(i)})$$
$$J = \frac{-1}{m} \sum_{i=1}^{m} L^{(i)}$$

Note that $x^{(i)}$ represents a single input example, and is of shape $D_x \times 1$. Further $y^{(i)}$ is a single output label and is a scalar. There are *m* examples in our dataset. We will use D_{a_1} nodes in our hidden layer; that is, z_1 's shape is $D_{a_1} \times 1$.

1. What are the shapes of W_1, b_1, W_2, b_2? If we were vectorizing this network across multiple examples, what would the shapes of the weights/biases be instead? If we were vectorizing across multiple examples, what would the shapes of X and Y be instead?

2. What is $\partial J / \partial \hat{y}^{(i)}$? Refer to this result as ${\delta_1}^{(i)}$. Using this result, what is $\partial J / \partial \hat{y}$?

3. What is $\partial \hat{y}^{(i)} / \partial z_2$? Refer to this result as ${\delta_2}^{(i)}$.

Equations reproduced below for the remaining parts of the question:

$$z_1 = W_1 x^{(i)} + b_1$$
$$a_1 = ReLU(z_1)$$
$$z_2 = W_2 a_1 + b_2$$
$$\hat{y}^{(i)} = \sigma(z_2)$$
$$L^{(i)} = y^{(i)} * \log(\hat{y}^{(i)}) + (1 - y^{(i)}) * \log(1 - \hat{y}^{(i)})$$
$$J = \frac{-1}{m} \sum_{i=1}^{m} L^{(i)}$$

Note that $x^{(i)}$ represents a single input example, and is of shape $D_x \times 1$. Further $y^{(i)}$ is a single output label and is a scalar. There are *m* examples in our dataset. We will use D_{a_1} nodes in our hidden layer; that is, z_1 's shape is $D_{a_1} \times 1$.

4. What is $\partial z_2 / \partial a_1$? Refer to this result as $\delta_3{}^{(i)}$.

5. What is $\partial a_1 / \partial z_1$? Refer to this result as $\delta_4{}^{(i)}$.

6. What is $\partial z_1 / \partial W_1$? Refer to this result as $\delta_5{}^{(i)}$.

7. What is $\partial J / \partial W_1$? It may help to reuse work from the previous parts. Hint: Be careful with the shapes!

Problem 4. Bonus!

Apart from simple mathematical operations like multiplication or exponentiation, and piecewise operations like the max used in relu activations, we can also perform complex operations in our neural networks. For this question, we'll be exploring the **sort** operation in hopes of better understanding how to backpropagate gradients through a sort. This is applicable in a variety of real-world use-cases including a differentiable non-max suppression, for object detection networks.

For each of the following parts, assume you are given an input vector $x \in R^n$ and some upstream gradient vector $\partial L / \partial F$, and you want to calculate $\partial L / \partial x$ where *F* is a function of *x* that also returns a vector. You may assume all values in *x* are distinct. Note that x_0 is the first component in the vector *x:* ($x = [x_0, x_1, \ldots, x_{n-1}]$).

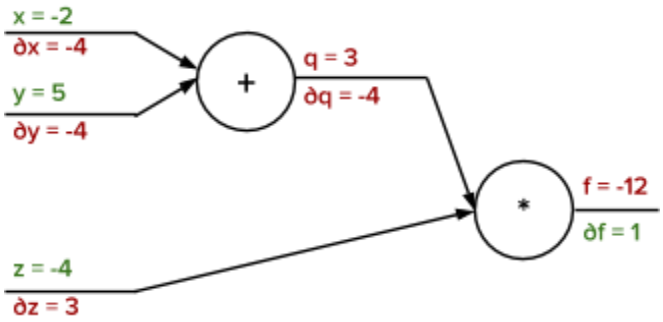
1. F(x) = x_0 * x

2. F(x) = sort(x)

3. F(x) = x_0 * sort(x)

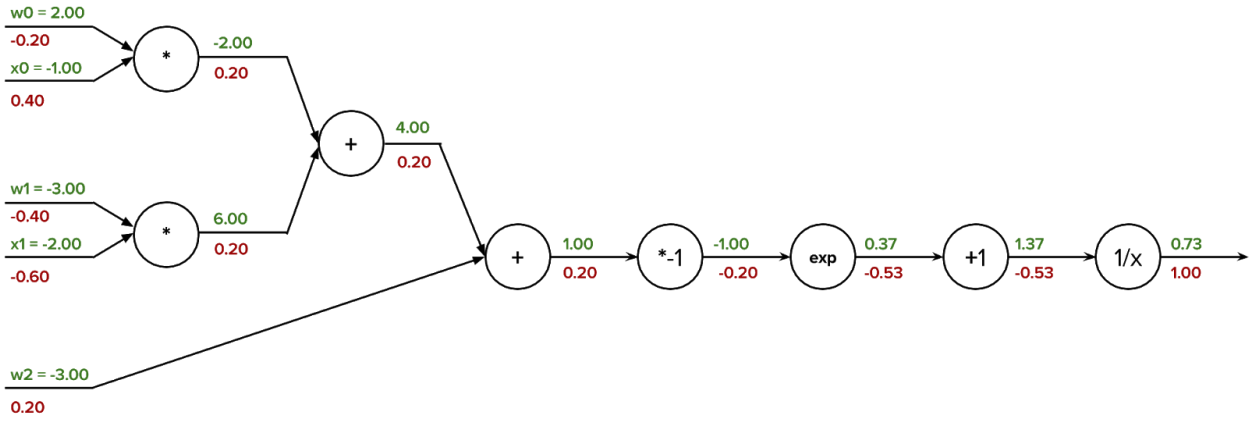
Section 3 Solutions

Problem 1. Computation Graph Review



1. $\partial f / \partial q = z = -4$
2. $\partial q / \partial x = 1$
3. $\partial q / \partial y = 1$
4. $\partial f / \partial z = q = x + y = 3$
5. $\partial f / \partial x = z * 1 = z = -4$
6. $\partial f / \partial y = z * 1 = z = -4$

Problem 2. Computation Graphs on Steroids



1. $\partial \alpha / \partial x_0 = \sigma(z)\,(1 - \sigma(z))\,w_0$
2. $\partial \alpha / \partial w_0 = \sigma(z)\,(1 - \sigma(z))\,x_0$
3. $\partial \alpha / \partial x_1 = \sigma(z)\,(1 - \sigma(z))\,w_1$
4. $\partial \alpha / \partial w_1 = \sigma(z)\,(1 - \sigma(z))\,x_1$
5. $\partial \alpha / \partial w_2 = \sigma(z)\,(1 - \sigma(z))$

Problem 3. Backpropagation Basics: Dimensions & Derivatives

1. $W_1 \in R^{D_{a_1} \times D_x}$, $b_1 \in R^{D_{a_1} \times 1}$, $W_2 \in R^{1 \times D_{a_1}}$, $b_2 \in R^{1 \times 1}$. The shapes of the weights/biases would be the same after vectorizing. $X \in R^{D_x \times m}$, $Y \in R^{m \times 1}$ after vectorizing.
2. $\delta_1^{(i)} = y^{(i)} / \hat{y}^{(i)} - (1 - y^{(i)}) / (1 - \hat{y}^{(i)})$. $\delta J / \delta \hat{y} = -\frac{1}{m} \sum_i \delta_1^{(i)}$
3. $\delta_2^{(i)} = \sigma(z_2)\,(1 - \sigma(z_2))$
4. $\delta_3^{(i)} = W_2$
5. $\delta_4^{(i)} = 0 \; if \; z_1 < 0, \; 1 \; if \; z_1 >= 0$
6. $\delta_5^{(i)} = x^{(i)^T}$
7. $\delta_6^{(i)} = -\frac{1}{m} \sum_i \delta_1^{(i)} * \delta_2^{(i)} * (\delta_3^{(i)} \circ \delta_4^{(i)}) * \delta_5^{(i)}$

Problem 4. Bonus!

1. As an example, say $x = [x_0, x_1, x_2]$. Then $F(x) = [x_0 * x_0, x_0 * x_1, x_0 * x_2]$. Then $\partial L / \partial x$ will be a vector. For component *i*, where *i* is not 0, it is $\partial L / \partial F_i * x_0$. For the 0th component, it will be $2 * x_0 * \partial L / \partial F_0 + \sum_{i \neq 0} \partial L / \partial F_i * x_i$.
2. Sorting will simply reroute the gradients. As an example, say $x = [x_0, x_1, x_2]$, we have upstream gradients $\partial L / \partial F = [\partial_0, \partial_1, \partial_2]$, and $F(x) = [x_1, x_2, x_0]$. Then, $\partial L / \partial x = [\partial_2, \partial_0, \partial_1]$ (move gradients to reverse the transformation from *x -> F(x)*).
3. This can be viewed as a computation graph where the multiplication happens first and then the sorting happens. As such, it simply requires rerouting the gradients to account for the sort as in #2, and then performing the multiplicative rules as in #1 to account for multiplying by x_0.