Section 08: Backpropagation

1. The Chain Rule

(a) Let $f : \mathbb{R}^n \to \mathbb{R}^m$, $g : \mathbb{R}^\ell \to \mathbb{R}^n$. Write the Jacobian of $f \circ g$ in terms of the Jacobian $\frac{\partial f}{\partial y}$ of f and the Jacobian $\frac{\partial g}{\partial x}$ of g. Make sure the matrix dimensions line up. What conditions must hold in order for this formula to make sense?

(b) What if instead the input of g is a matrix $W \in \mathbb{R}^{p \times q}$? Can we still write the derivative $\frac{\partial g}{\partial W}$ of g as a matrix?

2. Backpropagation Resources

Many people have written tutorials about backpropagation. We particularly recommend slides 24-36 of this first link below.

Joseph Redmon's Deep Learning slides, 03: `https://docs.google.com/presentation/d/1TW0oO7msvZqRZEGm7XkPcvr4CCTm8EHotY edit#slide=id.g435fffc3de_0_5`

Joseph Redmon's Deep Learning slides, 02: `https://docs.google.com/presentation/d/178I5I_3AK4Z6hZbDQ0R3nD2Roj2ZUJwyef edit#slide=id.g435fffc3de_0_406`

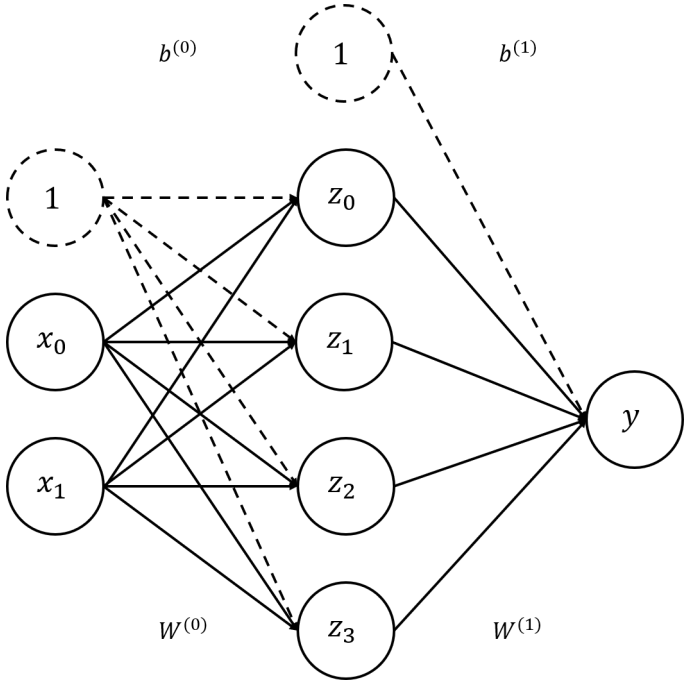
Section 8 notes from a previous quarter: `https://courses.cs.washington.edu/courses/cse446/20wi/Section8/Section_8.pdf`

colah's blog: `https://colah.github.io/posts/2015-08-Backprop/`

3. 1-Hidden-Layer Neural Network Gradients and Initialization

3.1. Forward and backward pass

Consider a 1-hidden-layer neural network with a single output unit. Formally the network can be defined by the parameters $W^{(0)} \in \mathbb{R}^{h \times d}$, $b^{(0)} \in \mathbb{R}^h$, $W^{(1)} \in \mathbb{R}^{1 \times h}$ and $b^{(1)} \in \mathbb{R}$. The input is given by $x \in \mathbb{R}^d$. We will use sigmoid activation for the first hidden layer z and no activation for the output y. Below is a visualization of such a neural network with $d = 2$ and $h = 4$.



(a) Write out the forward pass for the network using $x, W^{(0)}, b^{(0)}, z, W^{(1)}, b^{(1)}, \sigma$ and y.

Hint: Write $z = \ldots$ and $y = \ldots$

(b) Find the partial derivatives of the output with respect $W^{(1)}$ and $b^{(1)}$, namely $\frac{\partial y}{\partial W^{(1)}}$ and $\frac{\partial y}{\partial b^{(1)}}$.

(c) Now find the partial derivative of the output with respect to the output of the hidden layer z, that is $\frac{\partial y}{\partial z}$

(d) Finally find the partial derivatives of the output with respect to $W^{(0)}$ and $b^{(0)}$, that is $\frac{\partial y}{\partial W^{(0)}}$ and $\frac{\partial y}{\partial b^{(0)}}$.

Hint: First find $\frac{\partial z_i}{\partial W_i^{(0)}}$ and $\frac{\partial z_i}{\partial b_i^{(0)}}$. Then note that $\frac{\partial y}{\partial W_i^{(0)}} = \frac{\partial y}{\partial z_i}\frac{\partial z_i}{\partial W_i^{(0)}}$ and $\frac{\partial y}{\partial b_i^{(0)}} = \frac{\partial y}{\partial z_i}\frac{\partial z_i}{\partial b_i^{(0)}}$

3.2. Weight initialization

Suppose we initialize all weights and biases in the network to 0 before performing gradient descent.

(a) For all $x \in \mathbb{R}^d$, find z and y after the forward pass.

(b) Now find the values of the gradients $\frac{\partial y}{\partial W^{(1)}}$, $\frac{\partial y}{\partial b^{(1)}}$, $\frac{\partial y}{\partial W^{(0)}}$ and $\frac{\partial y}{\partial b^{(0)}}$. Note that some of the gradients will be in terms of x.

(c) Observe the values of each z_i and observe each $\frac{\partial y}{\partial W_i^{(1)}}$ and $\frac{\partial y}{\partial b_i^{(1)}}$. What do you notice? And what does this imply for the expressiveness of the network? (Note that there is nothing special about the value 0 here, it just simplifies the calculations. The same can be shown for initialization with any constant c)

3.3. Other ways of computing gradients

There are at least three other ways of computing the gradient of a complicated function $f(x) : \mathbb{R}^n \to \mathbb{R}$. For each of these, think about one possible disadvantage that might explain why people tend to prefer reverse-mode automatic differentiation (i.e. backprop) in practice.

(a) *Finite differencing.* We can approximate the gradient as $\frac{\partial f}{\partial x_i} \approx \frac{1}{\varepsilon}\big(f(x + \varepsilon e_i) - f(x)\big)$, where ε is some small positive number and e_i is the ith standard basis vector.

(b) *Analytic differentiation.* If we can write down a mathematical formula for f, we could derive a formula for ∇f using pencil and paper, then write code to compute it directly.

(c) *Forward-mode automatic differentiation.* You may be surprised to learn that there is there is an automatic differentiation technique that proceeds *forward* through the graph and does not require saving any intermediate results. In the example above, let $v = W^{(0)}x + b^{(0)}$ be the pre-activation output of the hidden layer. Above, we computed

$$\frac{\partial y}{\partial W^{(0)}}(x, W^{(0)}, b^{(0)}, W^{(1)}, b^{(1)}) = \left(\frac{\partial y}{\partial z}(z, W^{(1)}, b^{(1)}) \cdot \frac{\partial z}{\partial v}(v)\right) \cdot \frac{\partial v}{\partial W^{(0)}}(W^{(0)}, x, b^{(0)})$$

(Note that $\frac{\partial v}{\partial W^{(0)}}$ is one of those cases where we need to flatten the matrix $W^{(0)}$ in order for the notation to make sense.) We could instead have computed this gradient by multiplying in the other direction:

$$\frac{\partial y}{\partial W^{(0)}}(x, W^{(0)}, b^{(0)}, W^{(1)}, b^{(1)}) = \frac{\partial y}{\partial z}(z, W^{(1)}, b^{(1)}) \cdot \left(\frac{\partial z}{\partial v}(v) \cdot \frac{\partial v}{\partial W^{(0)}}(W^{(0)}, x, b^{(0)})\right)$$

This second method is called forward-mode automatic differentiation. Why do you think people usually use reverse-mode autodiff instead?

4. Auto-differentiation on a (non-neural-network) computational graph

(a) We explore how to construct a computational graph, as we learned for general auto-differentiation. Consider the toy function $p : \mathbb{R}^2 \to \mathbb{R}$.

$$p(x, y) = \frac{6\exp(-y)}{1 + x^2 + y^2} + 2x^3$$

For computational efficiency, we want create intermediate variables that have minimal dependency with each others. Write out the intermediate variables and draw the computational graph.

(b) Now, we use auto-differentiation. (Check the solution to part (a) first to ensure you're using the same definitions for the intermediate variables $z_1, \ldots, z_4$.) Calculate the following derivatives:

- $\frac{\partial z_4}{\partial z_3}$
- $\frac{\partial z_4}{\partial z_2}$
- $\frac{\partial z_4}{\partial z_1}$
- $\frac{\partial z_4}{\partial x}$
- $\frac{\partial z_4}{\partial y}$

To do so, suppose x, y, z_1, z_2, z_3, z_4 are already pre-computed so that your expression for $\frac{\partial z_4}{\partial z_t}$ should be written in terms of x, y, z_1, z_2, z_3, z_4 and the derivatives $\frac{\partial z_4}{\partial z_{t+1}}, \ldots, \frac{\partial z_4}{\partial z_4}$.