EECS 182 Deep Neural Networks
Spring 2023 Anant Sahai

Discussion 2

1. Visualizing Derivatives

Consider a simple neural network that takes a scalar real input, has 1 hidden layer with k units in it and a ReLU nonlinearity for those units, and an output linear (affine) layer.

We can algebraically write any function that it represents as

$$y = W^{(2)}(\max(\mathbf{0}, W^{(1)}x + \mathbf{b^{(1)}})) + b^{(2)}$$

Where $x, y \in \mathbb{R}$, $W^{(1)} \in \mathbb{R}^{k\times 1}$, $W^{(2)} \in \mathbb{R}^{1\times k}$, and $\mathbf{b}^{(1)} \in \mathbb{R}^{k\times 1}$, and $b^{(2)} \in \mathbb{R}$. The superscripts are indices, not exponents and the $\max$ given two vector arguments applies the max on corresponding pairs and returns a vector.

For each part, **calculate the partial derivative and sketch a small representative plot of the derivative as a function of** x. Make sure to clearly label any discontinuities, kinks, and slopes of segments. The subscript i refers to the i-th element of a vector.

(a) $\frac{\partial y}{\partial b^{(2)}}$

(b) $\frac{\partial y}{\partial w_i^{(2)}}$

(c) $\frac{\partial y}{\partial b_i^{(1)}}$

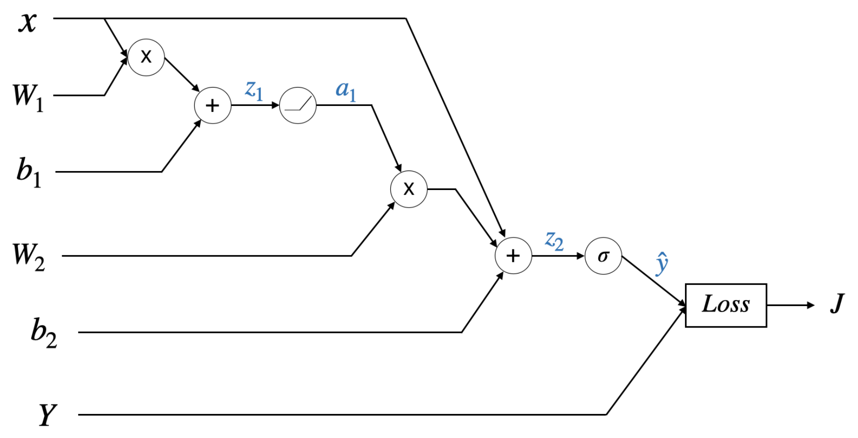
(d) $\frac{\partial y}{\partial w_i^{(1)}}$

2. Computation Graph Review

One of the reasons why modern neural networks are growing much larger than ten years ago is due to the advent of automatic differentiation softwares, also known as *Autodiff*. Autodiff softwares can compute the gradients of any differentiable function by constructing a *computation graph* of such functions, which allows researchers to simply focus on building models with effective information propagation and not to worry about the complex back prop.

Here is the computation graph of an one hidden layer MLP with *skip connection*, which basically means adding the input to the output of some later layers (this is a commonly used design that we will cover in detail later in the course). σ denotes the sigmoid function and *J* is the final loss. Assume that we use binary cross-entropy as our loss function with a ground truth label y, ie. $loss = -y\log\hat{y} - (1-y)log(1-\hat{y})$. You may find this quantity useful:

- $\frac{\partial J}{\partial z_2} = \hat{y} - y$



(a) **Express $\hat{y}$ as a function of *x*, *W*$_1$, *b*$_1$, *W*$_2$ and *b*$_2$.**

(b) **Compute the gradient** $\frac{\partial J}{\partial W_2}$, $\frac{\partial J}{\partial b_2}$.

(c) **Compute the gradient** $\frac{\partial J}{\partial W_1}$, $\frac{\partial J}{\partial b_1}$, $\frac{\partial J}{\partial x}$.

(d) What **intermediate variables do we need to cache** in this case? **Why is it more efficient to have an additional backward pass** on top of the forward pass for gradient computation?

3. MAP Interpretation of Ridge Regression

Consider the Ridge Regression estimator,

$$\underset{\mathbf{w}}{\operatorname{argmin}} \|X\mathbf{w} - y\|_2^2 + \lambda\|\mathbf{w}\|^2$$

We know this is solved by

$$\hat{\mathbf{w}} = (X^T X + \lambda I)^{-1} X^T \mathbf{y} \tag{1}$$

One interpretation of Ridge Regression is to find the Maximum A Posteriori (MAP) estimate on **w**, the parameters, assuming that the prior of **w** is $\mathcal{N}(0, I)$ and that the random **Y** is generated using

$$\mathbf{Y} = X\mathbf{w} + \sqrt{\lambda}N$$

Note that each entry of vector N is zero-mean, unit-variance normal. **Show that Equation 1 is is indeed the MAP estimate for w given an observation on Y=y.**

Contributors:

- Saagar Sanghavi.
- Anant Sahai.
- Kevin Li.