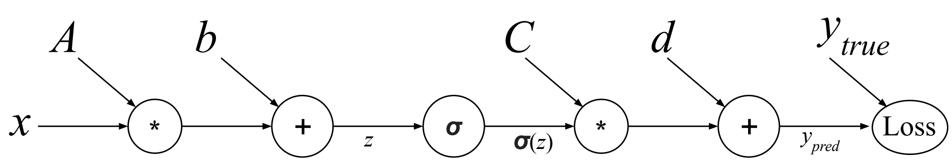
EECS 182 Deep Neural Networks
Spring 2023 Anant Sahai

Midterm Review: Basics

Consider the simple neural network that takes a scalar real input, has 1 hidden layer with k units in it and a sigmoid nonlinearity for those units, and an output linear (affine) layer to predict a scalar output. We can algebraically write any function that it represents as

$$y_{pred} = C\sigma(Ax + \mathbf{b}) + d$$

The $\sigma(.)$ represents an arbitrary nonlinearity, with derivative $\sigma'(.)$ Where $x \in \mathbb{R}$, $A \in \mathbb{R}^{k\times 1}$, $\mathbf{b} \in \mathbb{R}^{k\times 1}$, $C \in \mathbb{R}^{1\times k}$, $d \in \mathbb{R}$, and $y_{pred} \in \mathbb{R}$ We can write it as $y_{pred} = C\sigma(\mathbf{z}) + d$, where $z = Ax + \mathbf{b}$ and the nonlinearity is applied element-wise. We have the true label y_{true} for each x, and we use the L2 Loss $L(y_{true}, y_{pred}) = (y_{true} - y_{pred})^2$.



1. (a) Consider the sigmoid nonlinearity function $\sigma(z) = \frac{1}{1+e^{-z}}$. Show that $\frac{d}{dz}\sigma(z) = \sigma(z)(1 - \sigma(z))$

Solution:

$$\sigma(z) = (1 + \exp(-z))^{-1}$$
$$\sigma'(z) = -(1 + \exp(-z))^{-2}(-\exp(-z))$$
$$\sigma'(z) = \frac{1}{1 + \exp(-z)} \cdot \frac{\exp(-z)}{1 + \exp(-z)}$$
$$\sigma'(z) = \sigma(z)(1 - \sigma(z))$$

(b) Calculate $\frac{\partial L}{\partial d}$

Solution: $\frac{\partial L}{\partial d} = \frac{\partial L}{\partial y_{pred}} = 2 * (y_{pred} - y_{true})$

(c) Calculate $\frac{\partial L}{\partial C_i}$

Solution: $\frac{\partial L}{\partial C_i} = 2 * [\sigma(Ax + \mathbf{b})]_i * (y_{pred} - y_{true})$

(d) Calculate $\frac{\partial L}{\partial b_i}$

Solution: $\frac{\partial L}{\partial b_i} = 2 * C_i * [\sigma'(Ax + \mathbf{b})]_i * 1 * (y_{pred} - y_{true})$

(e) Calculate $\frac{\partial L}{\partial A_i}$

Solution: $\frac{\partial L}{\partial A_i} = 2 * C_i * [\sigma'(Ax + \mathbf{b})]_i * x * (y_{pred} - y_{true})$

(f) Write the gradient-descent update rule for $\mathbf{b}^{(t+1)}$ with learning rate α.

Solution: $\mathbf{b}^{(t+1)} = \mathbf{b}^{(t)} - \alpha \frac{\partial L}{\partial \mathbf{b}^{(t)}}^T$

2. Given the Regularized Objective function:

$$\underset{\mathbf{x}}{\operatorname{argmin}} \|A\mathbf{x} - \mathbf{b}\|^2 + \lambda \|\mathbf{x}\|^2$$

Use vector calculus to find the closed form solution for $\mathbf{x}$. Interpret what this means in terms of the singular values.

Solution: Expand out objective

$$f(\mathbf{x}) = \mathbf{x}^T A^T A \mathbf{x} - 2\mathbf{x}^T A \mathbf{b} + \mathbf{b}^T \mathbf{b} + \lambda \mathbf{x}^T \mathbf{x}$$

Take gradient wrt $\mathbf{x}$ and set it to zero:

$$\nabla_{\mathbf{x}} f = 2A^T A \mathbf{x} - 2A^T \mathbf{b} + 2\lambda \mathbf{x} = \mathbf{0}$$

Solve for $\mathbf{x}$:

$$(A^T A + \lambda I)\mathbf{x} = A^T \mathbf{b}$$
$$\mathbf{x} = (A^T A + \lambda I)^{-1} A^T \mathbf{b}$$

Interpretation: "Hack" of shifting the singular values of $A^T A$ away from zero, so that they don't blow up when the matrix is inverted.

3. Consider a simple neural network that spits out 1-dim values after a nonlinearity. These values for a batch are $\{1, 7, 7, 9\}$. What is the output of running batchnorm with this data and $\gamma = 1$ and $\beta = 0$. In other words, standardize the data to have mean 0 and variance 1.

Solution: We calculate the mean of the batch as $\mu = \frac{1+7+7+9}{4} = \frac{24}{4} = 6$

We calculate the variance as $\sigma^2 = \frac{(1-6)^2+(7-6)^2+(7-6)^2+(9-6)^2}{4} = \frac{5^2+1^2+1^2+3^2}{4} = \frac{36}{4} = 9.$

So the standard deviation is $\sigma = \sqrt{9} = 3.$

To standardize the batch, we subtract out the mean and divide by the standard deviation. so our batch becomes $\{\frac{1-6}{3}, \frac{7-6}{3}, \frac{7-6}{3}, \frac{9-6}{3}\} = \{\frac{-5}{3}, \frac{1}{3}, \frac{1}{3}, 1\}$

4. Consider a simplified batchnorm layer where we don't actually divide by standard deviation, instead we just de-mean our data before scaling it by γ, then passing it to the next layer. That is, we calculate our mini-batch mean μ, then simply let $\hat{x}_i = x_i - \mu$, and $y_i = \gamma \hat{x}_i$ is passed onto the next layer. Assume batchsize of m. If our final loss function is L, Calculate $\frac{\partial L}{\partial x_i}$ in terms of $\frac{\partial L}{\partial y_j}$ for $j = 1, ...m$, γ, and m.

Solution:

Note that since values for a given x_i affects all the values of y_j we need to sum over partial derivatives with respect to all the different y_j.

$$\frac{\partial L}{\partial x_i} = \sum_{j=1}^{m} \frac{\partial L}{\partial y_j} * \frac{\partial y_j}{\partial x_i}$$

Now, let's calculate $\frac{\partial y_j}{\partial x_i}$. Note that we can write y_j as

$$y_j = \gamma \left(x_j - \frac{1}{m} \sum_{k=1}^{m} x_k \right)$$

So we can calculate it's derivative wrt x_i as

$$\frac{\partial y_j}{\partial x_i} = \begin{cases} \gamma * \left(1 - \frac{1}{m}\right) \text{, if } i = j \\ \gamma * \left(-\frac{1}{m}\right) \text{, if } i \neq j \end{cases}$$

We can combine the sum to get

$$\frac{\partial L}{\partial x_i} = \left[\sum_{j=1, j\neq i}^{m} \gamma * -\frac{1}{m} * \frac{\partial L}{\partial y_j}\right] + \left[\gamma * (1 - \frac{1}{m}) * \frac{\partial L}{\partial y_i}\right]$$

$$= \gamma \left[\frac{\partial L}{\partial y_i} - \frac{1}{m}\sum_{j=1}^{m} \frac{\partial L}{\partial y_j}\right]$$

