Homework 4

Week 5-6: Neural Networks

Patthadon Phengpinij
Collaborators. ChatGPT

1 The Basics

In this section, we will review some of the basic materials taught in class. These are simple tasks and integral to the understanding of deep neural networks, but many students seem to misunderstand.

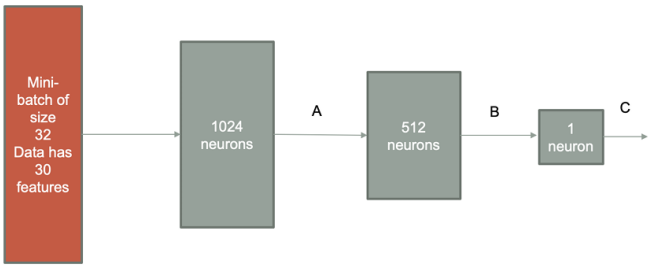
T1. Compute the forward and backward pass of the following computation. Note that this is a simplified residual connection.

$$\begin{aligned} x_1 &= ReLU(x_0 * w_0 + b_0) \\ y_1 &= x_1 * w_1 + b_1 \\ z &= ReLU(y_1 + x_0) \end{aligned}$$

Let $x_0 = 1.0$, $w_0 = 0.3$, $w_1 = -0.2$, $b_0 = 0.1$, $b_1 = -0.3$. Find the gradient of z with respect to w_0, w_1, b_0, and b_1.

Solution. The solution goes here.

T2. Given the following network architecture specifications, determine the size of the output A, B, and C.



Solution. The solution goes here.

T3. What is the total number of learnable parameters in this network? (Don't forget the bias term)

Solution. The solution goes here.

2 Deep Learning from (almost) scratch

In this section we will code simple a neural network model from scratch (numpy). However, before we go into coding let's start with some loose ends, namely the gradient of the softmax layer.

Recall in class we define the softmax layer as:

$$P(y=j) = \frac{exp(h_j)}{\sum_k exp(h_k)}$$

where h_j is the output of the previous layer for class index j.

The cross entropy loss is defined as:

$$L = -\sum_j y_j \log P(y=j)$$

where y_j is 1 if y is class j, and 0 otherwise.

T4. Prove that the derivative of the loss with respect to h_i is $P(y=i) - y_i$. In other words, find $\frac{\partial L}{\partial h_i}$ for $i \in \{0, \ldots, N-1\}$ where N is the number of classes.

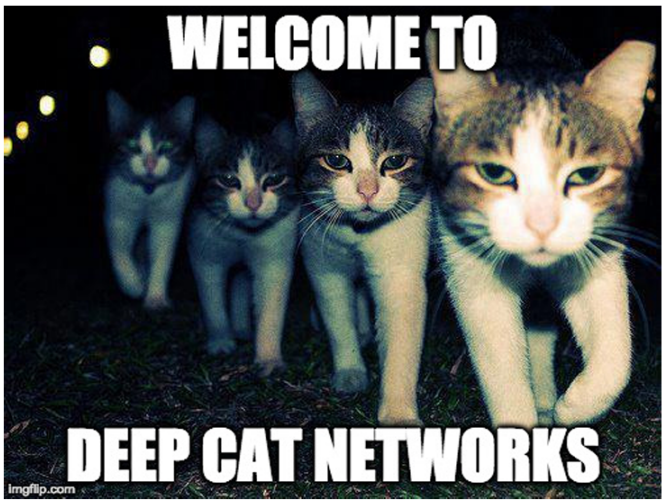
Hint: first find $\frac{\partial P(y=j)}{\partial h_j}$ for the case where $j = i$, and the case where $j \neq i$. Then, use the results with chain rule to find the derivative of the loss.

Solution. The solution goes here.

Next, we will code a simple neural network using numpy. Use the starter code `hw4.zip` on github. There are 8 tasks you need to complete in the starter code.

Hints: In order to do this part of the assignment, you will need to find gradients of vectors over matrices. We have done gradients of scalars (Traces) over matrices before, which is a matrix (two-dimensional). However, gradients of vectors over matrices will be a tensor (three-dimensional), and the properties we learned will not work. I highly recommend you find the gradients in parts. In other words, compute the gradient for each element in the the matrix/vector separately. Then, combine the result back into matrices. For more information, you can read this simple guide http://cs231n.stanford.edu/vecDerivs.pdf.

Happy coding.



All the tasks that are in the starter code, also the result and writeup solution, will be given in the **Appendix 1:** Simple Neural Network.