CSE 152 Assignment 3
Winter 2019
Due Thu, Mar 14, 5pm PST

Instructions :

- Attempt all questions.
- Please comment all your code adequately.
- In this assignment, you are required to use iPython notebook (`https://jupyter.org/`) to run your code.
- You may use the ieng6 cluster to obtain GPU resources (instructions on Piazza).
- There is one .ipynb file (`hw3.ipynb`) which contain codes for problems 2 and 3, respectively.
- You must submit (i) a ZIP file containing the solution .ipynb file and (ii) a PDF file containing your iPython notebook codes and results. You have to export iPython notebooks to a PDF which can demonstrate your code and figures clearly.
- You must submit both files (.pdf and .zip) on Gradescope. You must mark each problem on Gradescope in the PDF.
- Please write your code at the "WRITE YOUR CODE HERE" prompt in the .ipynb file.

1 Backpropagation

We will study the backpropagation behavior for an exponential neuron, given by

$$f(z) = e^{2z} - 1. \tag{1}$$

Consider a two-dimensional input given by $\mathbf{x} = (x_1, x_2)^\top$. A weight vector $\mathbf{w} = (w_1, w_2)^\top$ and a bias b act on it. Thus, the output of the neuron is given by $f(x_1, x_2) = e^{2(w_1x_1+w_2x_2+b)} - 1$.

(a) Draw the computation graph for the neuron in terms of elementary operations (addition, subtraction, multiplication, division, exponentiation), as seen in class. **[2 points]**

(b) Consider inputs $x_1 = 0.4$, $x_2 = 0.5$, weights $w_1 = 0.2$, $w_2 = 0.3$ and bias $b = 0.1$. In the same figure, show the values at each node of the graph during forward propagation. **[2 points]**

(c) Use backpropagation to determine the gradients $\frac{\partial f}{\partial w_1}$, $\frac{\partial f}{\partial w_2}$ and $\frac{\partial f}{\partial b}$. Also illustrate in the same figure the intermediate gradients at each node of the computation graph. **[4 points]**

(d) Explain the process of backpropagation you used to compute partial derivatives. **[2 points]**

2 Training a small CNN for MNIST digit classification

In this problem, we will train a small convolutional neural network for image classification, using PyTorch.[1] We will use the MNIST dataset of digits.[2] You will define the network structure, cost functions and optimization methods, as well as analyze the outputs. Please refer to the Jupyter notebook file and enter your code in places indicated by "YOUR CODE HERE". **[15 points]**

3 Transfer learning

You will now visualize the effects of transfer learning by performing experiments using the CIFAR-10 dataset.[3] Note that this is just to understand how transfer learning works, in practice it is generally used with very large datasets and complex networks. Please refer to the Jupyter notebook file and enter your code in places indicated by "YOUR CODE HERE". **[20 points]**

[1] `https://pytorch.org/`

[2] Available for download here: `http://yann.lecun.com/exdb/mnist/`

[3] Available for download here: `https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz`