IE643 Deep Learning : Theory and Practice **July-Dec 2022**

Assignment 2: Due On 11^{th} September 2022 (11:59 PM IST)

1 Instructions

Answer all questions. Write your answers clearly. You can score a maximum of 50 marks in this assignment.

Make sure that your answers and plots are visible in the python notebook (.ipynb) file. Upload in moodle, the .ipynb files correspoding to the questions as a single zip file named as "IE643_rollno_assignment2.zip". All your files within the zip file should follow similar naming convention. There will be no extensions to the submission deadline.

The links for all related files used in Assignment 2 are provided in moodle.

2 Assignment Questions

1. **[Use only Python]** Use the code template for multi-layer perceptron (or feed-forward network) posted in moodle. Answer the following:

   (a) Recall that you might have implemented the $\sigma_{\tanh}$ activation function (called tanh sigmoid) given by $\sigma_{\tanh}(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$ in the code. Now, implement the python functions to compute the following activation function called **STANH** given by $\text{STANH}(z) = z\sigma_{\tanh}(z)$ and its gradient. [**2 marks**]

   (b) Recall that you might have implemented the ReLU activation function given by $\text{ReLU}(z) = \max\{z, 0\}$. Consider the following activation function called **a-ReLU** given by

   $$\text{a-ReLU}(z) = \begin{cases} az & \text{if } z < 0 \\ z & \text{else} \end{cases}$$

   for some positive $a \in (0, 1)$. Write python functions to compute the a-ReLU activation function and its (sub-)gradient. [**2 marks**]

   (c) Consider another activation function called **EXU** given by

   $$\text{EXU}(z) = \begin{cases} \beta z & \text{if } z \geq 0 \\ \beta\mu(\exp(z) - 1) & \text{else} \end{cases}$$

   with $\beta = 1.05, \mu = 1.67$. Write python functions to compute the EXU activation function and its (sub-)gradient. [**2 marks**]

   (d) Consider an appropriate neural network architecture where each hidden layer has only STANH activation functions and the output layer has a logistic sigmoid activation function. Illustrate the exploding gradient and vanishing gradient problems in this network. Justify the architecture you used, indicate how you checked the exploding gradient and vanishing gradient problems, and explain your observations. [**3 marks**]

(e) Consider an appropriate neural network architecture where each hidden layer has only a-ReLU activation functions and the output layer has a logistic sigmoid activation function. Illustrate the exploding gradient and vanishing gradient problems in this network. Justify the architecture you used, indicate how you checked the exploding gradient and vanishing gradient problems, and explain your observations. [**3 marks**]

(f) Consider an appropriate neural network architecture where each hidden layer has only EXU activation functions and the output layer has a logistic sigmoid activation function. Illustrate the exploding gradient and vanishing gradient problems in this network. Justify the architecture you used, indicate how you checked the exploding gradient and vanishing gradient problems, and explain your observations. [**3 marks**]

(g) Among the three networks considered in questions 1d, 1e, 1f, describe which networks were more prone to the vanishing gradient issue and which networks were more prone to the exploding gradient issue. Use appropriate justifications for your observations involving the quantities used during backpropagation. [**5 marks**]

(h) Suppose you had a linear activation function at the output layer in the three networks you considered in questions 1d, 1e, 1f. Compare and contrast the exploding gradient and vanishing gradient issues in the networks with linear activations at the output layer with that of networks with logistic sigmoid activation function at the output layer. Use appropriate justifications for your observations involving the quantities used during backpropagation. Using these observations, can you comment on the behavior if you had ReLU activation at the output layer (without actually running the experiments)? [**5 marks**]

2. **[Use only Python]** Use the code template for multi-layer perceptron (or feed-forward network) posted in moodle. Consider the RestaurantReviews data set posted in moodle. In RestaurantReviews.csv file, ignore the first column (which denotes the restaurant id), consider the last 5 columns as the labels and the other columns (except the id column) as features. Realize that the data corresponds to a multi-label setting.

(a) Write code to read the data into suitable `numpy` arrays. [**2 marks**]

(b) Write the required code to shuffle and split the data set into three sets S_1, S_2 and S_3 such that S_1 contains 65% of the data, S_2 contains 20% of the data and S_3 contains 15% of the data. [**2 marks**]

(c) Design a single feed forward neural network and a corresponding loss function to perform training on the RestaurantReviews data set. Justify the design choice of your neural network and loss function and implement the loss function in the code. [**3 marks**]

(d) Illustrate how you will carry out backpropagation for the new loss function in the last layer. Include its implementation in code. [**3 marks**]

(e) Recall that **accuracy** was used as a performance metric for binary and multi-class classification. Here, we will use the following performance metrics to measure the predictive capability of the classifier. Suppose the actual labels of a sample x is given by the set $y \subseteq \mathcal{Y}$ and the predicted labels are given by the set $\hat{y} \subseteq \mathcal{Y}$. Then we calculate the following metrics:
Precision $= \frac{|y \cap \hat{y}|}{|\hat{y}|}$ and **Recall** $= \frac{|y \cap \hat{y}|}{|y|}$, where the notation $|A|$ denotes the size or cardinality of set A.
Explain your understanding of the performance metrics given above. Implement in code, the computation of average Precision and average Recall computed over the data set samples. [**3 marks**]

(f) For the chosen loss function, choose the learning rates from the set $\{0.1, 0.01, 0.001, 10^{-4}, 10^{-5}\}$ and mini-batch sizes from $\{5, 10, 20, 30, 40\}$. For each (learning rate, mini-batch size) pair, run the mini-batch stochastic gradient descent algorithm on S_1, with 200 epochs. For every 5 epochs,

record the loss and average precision and average recall achieved on the sets S_1 and S_2. Now plot the loss for every 5 epochs for each (learning rate, mini-batch size) pair on S_2 (use a single plot and different colors for different pairs). Similarly plot the average precision for every 5 epochs for each (learning rate, mini-batch size) pair on S_2 (use a single plot and different colors for different pairs). Also plot the average recall for every 5 epochs for each (learning rate, mini-batch size) pair on S_2 (use a single plot and different colors for different pairs). Can you come up with a suitable selection procedure for the best (learning rate, mini-batch size) pair using the experiments conducted? Explain your selection procedure and justify. [**6 marks**]

(g) Using the best (learning rate, mini-batch size) pair identified above, conduct training using mini-batch SGD on the set $S_1 \bigcup S_2$ with max epochs set to 500. For every 5 epochs, record the loss and average precision and average recall achieved on the sets $S_1 \bigcup S_2$ and S_3. Include a stopping condition such that you can stop the training when the loss on the set $S_1 \bigcup S_2$ does not decrease significantly for p epochs with a suitable choice for p. Plot the loss on $S_1 \bigcup S_2$ and S_3 in a single plot and comment on the observations. Similarly plot average precision on $S_1 \bigcup S_2$ and S_3 in a single plot and comment on the observations. Also plot average recall on $S_1 \bigcup S_2$ and S_3 in a single plot and comment on the observations. [**6 marks**]
