Assignment 1 : Neural Network

Due: 9/30

Submission Instructions

- Zip all your files (code, README, written answers, etc.) in a zip file named $\{firstname\}\_\{lastname\}\_CS6000\_HW1.zip$ and upload it to Blackboard
- Submit your model's prediction in Kaggle

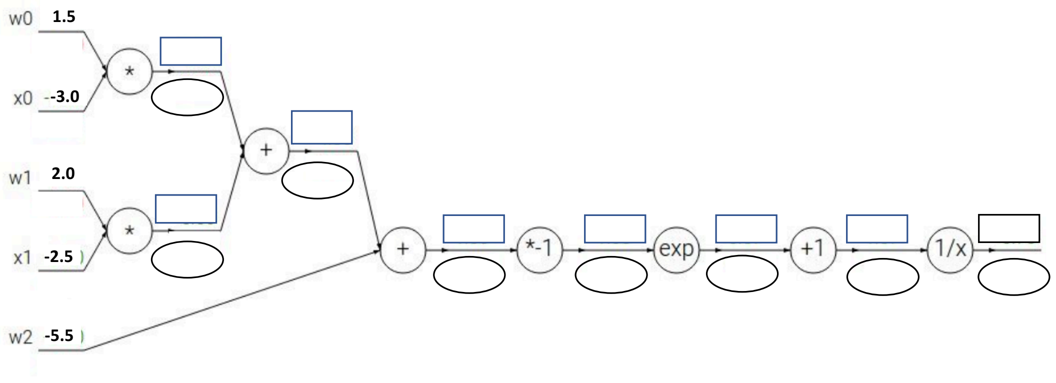
This assignment is split into two sections: Math and Neural Network. The first one entirely consists of written, analysis questions, whereas the second is primarily coding and implementation focused. If you get stuck on the first section, you can always work on the second. If you ever get stuck along the way, please come to Office Hours so that the TA can support you.

1. Chain Rule Practice (20 points)

   (a) Apply chain rule to differentiate the following functions:

   - $y = \sqrt{13x^2 - 5x + 8}$
   - $y = log(4 + (x-1)^2)$
   - $y = \frac{e^{-(x+5)}}{k!e^{3x}}$ for some integer k

   (b) Calculate one forward (square boxes) and backward (oval boxes) iteration for the given network:



2. (10 points) A fully connected Neural Network has $L = 2$, $d(0) = 4$, $d(1) = 2$, $d(2) = 1$. If only products of the form $w_{ij}^{(l)} x_i^{(l-1)}$, $w_{ij}^{(l)} \delta_j^{(l)}$, and $x_i^{(l-1)} \delta_j^{(l)}$ count as operations (even for $x_0^{(l-1)} = 1$), without counting anything else, what is the total number of operations in a single iteration of back-propagation (using SGD on one data point)?

3. (20 points) Let us call every "node" in a Neural Network a unit, whether that unit is an input variable or a neuron in one of the layers. Consider a Neural Network that has 12 input units (the constant $x_0^{(0)}$ is counted here as a unit), one output unit, and 40 hidden units (each $x_0^{(l)}$) is also counted as a unit). The hidden units can be arranged in any number of layers $l = 1, \ldots, L-1$, and each layer is fully connected to the layer above it.

   (a) Compute the minimum possible number of weights such a network can have.

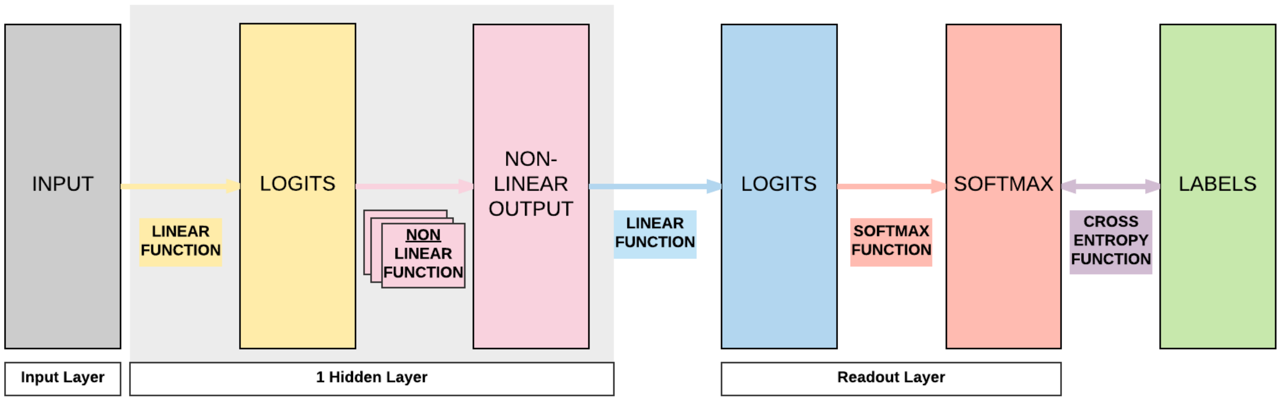
   (b) If we restrain the network to have only 2 hidden layers, compute the maximum possible number of weights such a network can have.

4. (50 points) Power of Neural Network (MLP)

   In this exercise, you will implement a two-layer neural network in Tensorflow 2.0 or Keras. You will use the data-set MNIST data-set to classify digits from "0" to "9". General speaking, it is ten-class classification problem.

   Your network will have two layers with 784, 256 units in hidden layer 1 and 2 respectively. The output layer will have 10 units representing the probabilities of input being "5" , "9" or other digit.

   For example, an output vector of $[\hat{y}_0, \hat{y}_1, \ldots \hat{y}_9]$ represents a probability of being digit "0" to digit "9". Your model is correct if the output unit with the *highest* probability



   (a) Build your own neural network :
   Cross - entropy as our objective function:

$$E_{\text{entropy}} = -\sum_{c=1}^{C} y \ln \hat{y} \tag{1}$$

i. Building your own Neural Net(Only Numpy can be used in modeling) and use the model to predict the class of test file mnist.test.npy .

ii. Save the prediction in CSV file and submit it in your Kaggle competition. The sample submission CSV file has been provided in Kaggle .

(b) Build neural network with Package(Tensorflow or Keras): In this part, please build a neural network with same architecture (784, 256,10) above. In part (b) default optimizer should be sgd and activation function should be tanh.

i. Regulation is a critical method to prevent over-fitting problem. Dropout is a regularization technique that has been widely used in deep learning. During training, dropout randomly sets units in the hidden layer h to zero with probability p_{drop} (dropping different units each minibatch).
1. Why should we apply dropout during training but not during evaluation?
2. Implement Dropout in your model, try to find the best dropout probability from $P \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$.

- What is the best dropout value of your model and compare the classification performance with the model have no Dropout.
- For both of models, showing the overall accuracy score and accuracy of each class.

ii. Batch Normalization: It is called because during training, we normalize each layer's inputs by using the mean and variance of the values in the current batch.
1. Why Batch Normalization can improve model speed and performance?
2. Based on the model with no dropout, create identical neural networks with and without batch normalization to compare

- The accuracy (overall and each class)
- Total running time
- Loss (training loss and validation loss) in 20 epochs.
- Plotting the learning curve of both models.

iii. Optimization matters! Different optimizing rule would generate A wide spectrum of results. Try to create an identical neural network without dropout and batch normalization. Try to use 4 optimizer : Adam, SGD, Momentum, RMSprop with same learning rate 0.01 and training 20 epochs.

- What is the best optimizer for the network
- Plotting the learning curve of these 4 methods.

For your Tensorflow or Keras modeling, please using the following Python code to let your result reproducible:

```
def seed_everything(SEED):
    np.random.seed(SEED)
    tf.set_random_seed(SEED)
    random.seed(SEED)

seed_everything(6300)
```

(c) Discuss effective methods to select the Hyperparameters. Optimer hyperparameters like learning rate, mini-batch size and number of iteration. And model hyperparameter like the number of hidden layers/units. You are allowed to discuss among your classmates or consult Internet resources for this question, but you need to document the sources.

5. (10 points) Extra Credit : Writing your solution with Python Class type. And guideline provided in nnclass.py file.