CSE-5368 Neural Networks Fall 2023 Exam 2_

Last Name	First Name	Student ID Number

Prob #	**1**	**2**	**3**	**4**	**Total**
Points	**25**	**25**	**20**	**30**	

Time: 80 Minutes

Last Name	First Name	Student ID Number

$$F(\mathbf{x}) = F(\mathbf{x}^*) + \nabla F(\mathbf{x})^T\Big|_{\mathbf{x}=\mathbf{x}^*}(\mathbf{x}-\mathbf{x}^*)$$

$$+\frac{1}{2}(\mathbf{x}-\mathbf{x}^*)^T \nabla^2 F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}^*}(\mathbf{x}-\mathbf{x}^*) + \cdots$$

$$\frac{\mathbf{p}^T \nabla F(\mathbf{x})}{\|\mathbf{p}\|} \qquad \frac{\mathbf{p}^T \nabla^2 F(\mathbf{x})\mathbf{p}}{\|\mathbf{p}\|^2} \qquad \alpha_k = -\frac{\mathbf{g}_k^T \mathbf{p}_k}{\mathbf{p}_k^T \mathbf{A} \mathbf{p}_k}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g}_k \qquad \mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$$

$$L_i = \sum_{j \neq i} max(0, y_j - y_i + \Delta)$$

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$$

$$H(p,q) = -\sum_x p(x) log(q(x))$$

$$L_i = -log\left(\frac{e^{y_i}}{\sum_j e^{y_j}}\right)$$

$$\text{PE}(pos, 2i) = sin\left(\frac{pos}{10000^{2i/d_{model}}}\right),$$

$$\text{PE}(pos, 2i+1) = cos\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$

Last Name	First Name	Student ID Number

1. Consider a multi-layer neural network with three nodes at the last layer. For a given input, the true/desired probabilities for each class of a particular example are **[0.2,0.6,0.2]** and actual outputs are **[0.5, 0.1, 0.2]**
   Calculate the cross entropy loss. **Use natural log.**

Last Name	First Name	Student ID Number

Problem 1 Continued

Last Name	First Name	Student ID Number

2. Consider the expression:

$$\boldsymbol{f(x,y) = \frac{xy}{min(xy, y^2) - 6}}$$

given the inputs: $\boldsymbol{x = 5,\ \ y = 4}$

Draw the computational graph and calculate the $\frac{\delta f(x,y)}{\delta x}$ and $\frac{\delta f(x,y)}{\delta y}$

You **MUST SHOW** all the numerical values **for each node** as they flow in the forward and backward path in the computational graph.

Last Name	First Name	Student ID Number

Problem 2 Continued

Last Name	First Name	Student ID Number

3. Write a Python function to calculate positional encodings for transformers.

```
import numpy as np
def positional_encoding(embeddings):
"""
Args:
embeddings: Input tensor of shape (sequence_length, embedding_dim).
Returns:
positional_encodings: Matrix of positional encodings.
"""
# Your code here
```

Last Name	First Name	Student ID Number

Problem 3 Continued

Last Name	First Name	Student ID Number

4. Complete the following function. This function should create and train an autoencoder using tensorflow. **DO NOT USE keras.**

Biases should be included in the weight matrices as the first row of the weight matrix.

```
import numpy as np
import tensorflow as tf
def create_and_train_autoencoder(train,encoder_layers, epochs, alpha):
# train: numpy array of input [nof_train_samples,input_dimensions]
# encoder_layers: list of integers representing number of nodes in
# each layer of the encoder. The last number represents the dimension of
latent space
# return: List of weight matrices
# Notes:
# Initialize all the weights and biases to zeros.
# Assume all layers are fully connected.
# Assume the activation function for every layers is RELU.
# Use MSE to calculate loss. Use gradient descent to adjust weights.
# Assume batch size is equal to one.
# Assume the decoder has the same layer structure as the encoder in the
reverse order.
```

Last Name	First Name	Student ID Number

Problem 4 Continued