Assignment 1 Rubrics

Task 1 - WordPiece Tokenizer - [30 Marks]

1. Code - **[15 Marks]**

   a. **Preprocessing and Pair Score**
   Functions implementing the following functionally must be present.

      i. It includes splitting the data and splitting the word. For example: “**token” should get processed to t, ##o, ##k, ##e, ##n**. **[2.5 Marks]**

      ii. Pair Score should be computed for **every token** in the **current vocabulary**. Check for the logic. It should be based on probability. Refer to the following formula. **[2.5 Marks]**

      Score = freq(element1 || element2) / [freq(element1) * freq(element2)]

   b. **Pair merging and vocabulary construction**
   Functions implementing the following functionality must be present.

      i. Logic to **merge pairs** appropriately should be there. For example: **[i, ##n] should be merged to “in”** so logic to **remove ##** should be there. **[2.5 Marks]**

      ii. Logic for **vocabulary construction** should be appropriately written. There will be a while loop which should run until the provided vocabulary is not reached. **[2.5 Marks]**

   After vocabulary construction the vocabulary should be saved for further use in the subsequent tasks.

   c. **Tokenize method**
   Functions implementing the following functionality must be present.

      i. There should be a logic to **tokenize** a given **input sentence**. It should **generate** the **tokens** themselves. **[2.5 Marks]**

      ii. There should be a function to **tokenize** a given **input sentence** and **return** the **token IDs** for the tokens generated. **[2.5 Marks]**

2. Evaluation - **[10 Marks]**

   Ask them to generate vocabulary and train their tokenizer on **4937 tokens**, before further evaluation. Make sure you don't use the generated files.

   a. **JSON Files for tokenization** **[5 Marks]**

      i. Use the given **gen_json.py** to generate a JSON containing the sentences. And then provide it to students.
      ii. They will generate a **tokenized_{Group_No.}.json.** Make sure the naming convention is correct. Thereafter save this file locally on your system.

   b. **Vocabulary File -** Save their generated **vocabulary_{Group_No.}.txt** on your system. **[5 Marks]**

3. Viva - **[5 Marks]**

   a. **Individual Viva** will be conducted. Marks for the viva will be based on the individual performance of group members.
   b. You have to ask **questions related to concepts covered in the task** and **questions related to the code** they have implemented.

Task 2 - Word2Vec CBOW - [25 Marks]

1. Code - **[15 Marks]**

   a. **Dataset Class** - **[5 Marks]**

      They should load the provided “corpus.txt” for training the model.

      i. **__init__()** - Initialize necessary attributes such as data paths, preprocessing on the corpus, and any other parameters. **[0.5 Marks]**

      ii. **__len__()** - Return the total number of samples in the dataset. **[0.5 Marks]**

      iii. **__getitem__()** - Retrieve a sample from the preprocessed dataset based on the index. It should **return a tuple of (context tokens, target token).** **[0.5 Marks]**

iv. **Load and preprocess the corpus** - [3.5 Marks]
   1. Preprocessing for getting context words for target words should be there.
   2. Add with [PAD] tokens on either or both sides if not enough context words are present.
   3. Tokenization should be done based on IDs not actual tokens.

It is possible that they have implemented the logic for token IDs outside this class in that case actual tokens are returned instead of token IDs - **Deduct 1.5 Marks**

b. **Model Class** - **[5 Marks]**

   i. **__init__()** - Define and initialize the layers and parameters of the neural network. **[3 Marks]**
   ii. **forward()** - This function will compute a forward pass on the network. And then will return the output tensor. **[2 Marks]**

c. **Train function** - **[5 Marks]**

   i. **Utilities -** They should create an **optimizer** and a **dataloader** using their custom implemented dataset class **[2 Marks]**

   ii. **Training and Validation loop -** Training loop should be there. And for **every epoch** they should **print the training and validation loss**. Check for the following **[2 Marks]**
      1. Loss is computed with the model output and the target token.
      2. Check for **optimizer.zero_grad(), loss.backward(), and optimizer.step().**

   iii. **Checkpoint -** After the completion of training they should save the checkpoint of the trained word2vec model. **[1 Marks]**

2. Evaluation - **[5 Marks]**

   a. **Quality and correctness of triplets -** There must be two triplets and among the triplets two tokens must have high cosine similarity and two must have low. Quality of the tokens is based on subjective understanding of TAs. **[3 Marks]**
   b. **Train and Validation Loss Plots** - There should be plots of Training/Validation loss V/S Epochs. Check for the **continuous decrease** in the loss and the

convergence of the model. **[1+1 Marks]**

3. Viva - **[5 Marks]**

    a. **Individual Viva** will be conducted. Marks for the viva will be based on the individual performance of group members.
    b. You have to ask **questions related to concepts covered in the task** and **questions related to the code** they have implemented.

Task 3 - Neural LM - [45 Marks]

4. Code - **[25 Marks]**

    a. **Dataset Class** - **[5 Marks]**

    They should load the provided "corpus.txt" for training the model.

    i. **__init__()** - Initialize necessary attributes such as data paths, preprocessing on the corpus, and any other parameters. **[0.5 Marks]**

    ii. **__len__()** - Return the total number of samples in the dataset. **[0.5 Marks]**

    iii. **__getitem__()** - Retrieve a sample from the preprocessed dataset based on the index. **[0.5 Marks]**

    iv. **Load and preprocess the corpus** - Preprocessing should be done based on the **next word prediction task**. And it should be preprocessed so that the model can be trained in an autoregressive fashion. **[3.5 Marks]**

    b. **Model Class** - **[3 X 5 Marks]**

    There should be **three different implementations** of the model class. They should be different on a **logical basis** and the **difference must not be repetitive.** All the three implementations must contain the following

    i. **__init__()** - Define and initialize the layers and parameters of the neural network. **[3 X 3 Marks]**
    ii. **forward()** - This function will compute a forward pass on the network. And then will return the output tensor. **[2 X 3 Marks]**

    If the changes are not logical do not award any marks for that model

c. **Train function** - **[5 Marks]**

   i. **Utilities -** They should create an **optimizer** and a **dataloader** using their custom implemented dataset class **[2 Marks]**

   ii. **Training and Validation loop -** Training loop should be there. And for **every epoch** they should **print the training and validation loss**. Check for the following **[2 Marks]**
      1. Loss is computed with the model output and the target token.
      2. Check for **optimizer.zero_grad(), loss.backward(), and optimizer.step().**

   iii. **Checkpoint -** After the completion of training they should save the three checkpoints of the trained neuralLM models.**[1 Marks]**

5. Evaluation - **[10 Marks]**

   a. **Accuracy -** Accuracy on both training and validation set. **[2.5 Marks]**
   b. **Perplexity -** Perplexity on both training and validation set. **[2.5 Marks]**
   c. **Test LM** - Use the provided **gen_test_sentences.py** to generate a **"test.txt"** on which they will generate the **next 3-5 tokens**. Based on your understanding and their quality of tokens award marks. There will be **5 sentences** in the file. **[5 X 1 Marks]**

6. Viva - **[10 Marks]**

   a. **Individual Viva** will be conducted. Marks for the viva will be based on the individual performance of group members.
   b. You have to ask **questions related to concepts covered in the task** and **questions related to the code** they have implemented.

If the Report is not submitted deduct 10 Marks

NOTE

They have to submit a complete and comprehensive Report. Every explanation they are providing must be documented and logical. Other than that it is upto the subjective understanding of TAs.Assignment 1 Rubrics