AI/ML lab (CS 335) **End Sem**
Computer Science and Engineering **10-11-2022 Friday, Total time 2 hours 30 mins**
Indian Institute of Technology Bombay

Instructions

1. It is an OPEN BOOK and OPEN INTERNET examination.
2. This Lab has one question. The maximum marks is 20.
3. You need to upload your code solution and output files in Moodle before the deadline.
4. Be sure to follow the upload instructions.
5. Total time for the examination is 2 hours 30 minutes.
6. This is an open-ended assignment. You are free to use any API from the following libraries to solve the problems: `pytorch, numpy, scipy, matplotlib, time`.
7. **Note that you are not supposed to use `sklearn`**

1. **Set Retrieval Task** In this question we will train a model for set retrieval, *i.e.* the model ranks the items in a corpus for a given query. To this end, we assume that each query and each corpus is a set of items. For example, a query can be the set of keywords you type in google search bar and corpus is the set of words in the webpage that google returns. Google presents us with a list of corpus in a ranked order. To simplify the problem, we will work with a synthetic dataset where corpus contains a list of 63 webpages. Each webpage can contain different number of words in it. For simplicity, we consider that each query contains a set of exactly 3 keywords in it. Further, we represent words using pre-trained embeddings $\in \mathbb{R}^5$. Given a query q, the task is to assign score to the 63 corpus items such that the relevant corpus receive higher scores.

Dataset Description For this problem, we are given with a dataset consisting of the following:

- List of training queries: We provide 50 training queries in the form of a list. Each query is a set of items of fixed length, where each item is represented by a feature vector in $\mathbb{R}^5$
- List of corpus: We provide 63 corpus in the form of a list. Each corpus is a set of items of variable length $\in [6]$, where each item is represented by a feature vector in $\mathbb{R}^5$
- Training ground truth relevance labels: We provide binary relevance labels in the form of a tensor of shape $(50, 63)$. The entry (i, j) contains 1 if the j^{th} corpus set is relevant the the i^{th} query set, and 0 otherwise.
- List of test queries: We provide 10 test queries in the form of a list. Each query is a set of items of fixed length, where each item is represented by a feature vector in $\mathbb{R}^5$

Implementation Guidelines You can train any model of your choice. However, you have to adhere to the provided template which requires you to implement the following:

1.a function `set_embed` in class `Model`: You can implement any set embedding model that you want, as long as it meets the input and output shape criteria mentioned in the code.

1.b function `ranking loss`: Implement the following:

$$\sum_{q \in \text{queries}} \sum_{\substack{c_{\checkmark} \in \text{Corpus relevant to q,} \\ c_{\text{✗}} \in \text{Corpus irrelevant to q}}} \text{ReLU}[\text{score}(q, c_{\text{✗}}) - \text{score}(q, c_{\checkmark}) + \text{margin}]$$

1.c function `mean_average_precision` : Implement the mean average precision (**mAP**) score as described in the following link: [LINK]. You can read till the end of section 2 in the link. You should not use `sklearn`.

1.d `score`: Given the query set embedding $q \in \mathbb{R}^d$ and corpus set embedding $c \in \mathbb{R}^d$, compute the following relevance score:

$$\text{score}(q, c) = -\sum_{i=1}^{d} \text{ReLU}[(q - c)_i]$$

Note that you will have to compute the pairwise scores between all available query and corpus embeddings. Make sure to implement a tensorized code.

1.e You will also need to add code for training your models, in the main function.

Evaluation We will be evaluating the following:

1.a correctness of `ranking loss`

1.a	/3	

1.b correctness of `score`

1.b	/2	

1.c correctness of `mean_average_precision`

1.c	/5	

1.d evaluation of test set predictions (with respect to hidden test ground truth): **We will only refer to the uploaded output.pkl**. We will use the uploaded model files to check the consistency between the predicted scores in `output.pkl` and the model predictions.

1.d	/10	

1 Submission instructions

Complete the functions in `assignment.py`. Do not modify the function signatures. Keep the file in a folder named `<ROLL_NUMBER>_exam` and compress it to a tar file named `<ROLL_NUMBER>_exam.tar.gz` using the command

```
tar -zcvf <ROLL_NUMBER>_exam.tar.gz <ROLL_NUMBER>_exam
```

Submit the tar file on Moodle. The directory structure should be -

```
<ROLL_NUMBER>_exam
| - - - - assignment.py
| - - - - output.pkl
| - - - - model.pkl
```

Replace `ROLL_NUMBER` with your own roll number. If your Roll number has alphabets, they should be in *"small"* letters.

Total: 20