Word Meaning Comparison

(A3: Part A)

Words often have multiple meanings associated with them and the surrounding context often determines the specific meaning of the word which is used. The goal of this assignment is to develop deep neural models that can identify whether a particular word used in a sentence pair has the same meaning in both sentences or has a different meaning in each of the sentences.

For example,

S1: *We often used to play on the* ***bank*** of the *river*
S2: *We lived along the* ***bank*** *of the Ganges.*
S3: *He cashed a check at the* ***bank***

S1 and S2 use the same meaning of the word *bank* (river bed) while S1 and S3 use different meanings of the word *bank* (river bed vs. financial institution)

We call this task *Word Meaning Comparison* and frame it as a classification problem. The NLP model must classify the input sentence pair (*X*) and the word (*W)* into a label **T** if W has the same meaning in both sentences of *X* and **F** if *W* has different meanings.

X, W	Ground Truth Label
(S1, S2), bank	**T**
(S1, S3), bank	**F**
(S2, S3), bank	**F**

Instructions:
This assignment is divided into two parts. In both parts, you are only allowed to use deep neural models and so you cannot use sklearn.

In the current Part (A), you have to build an LSTM/CNN/Attention-based (or similar) model using only Pytorch and Word2Vec/FastText/Glove embeddings.

You are not allowed to use any pre-trained models (such as BERT and GPT-2) or other publicly available deep learning libraries. You will be given more flexibility in Part (B), for which we will release a separate set of instructions.

You will have to use Colab notebooks for training and testing your models. You need to create two notebboks - one for training and another for testing the models. You should start by making a copy of training notebook and testing notebook. To help you get started, it also contains some

boilterplate code, which you can use. It has the basic training loop in pytorch and some utility functions. You need to add your own data loaders and model files in order to get it to work.

Data:

You can download the training and validation data from data link. The *.data.txt correspond to the input file and *.gold.txt correspond to the final labels.

You are not allowed to use the validation data for training the model. You will have to report your final models' validation performance in a Google form. These will be available for the entire class to view and compare with their own. You can make multiple submissions to the form. The last one before the deadline will be considered final. We should get the same score on re-training your model (discussed further under ***Submission***).

Evaluation:

The performance on the task will be evaluated using binary accuracy.

The testing notebook contains the format of the test file, expected format of the final predictions and code for computing the performance. You have to report the score computed with this as the final validation performance.

Submission:

You have to name your Colab notebooks as USERID_A3_A_train.ipynb and USERID_A3_A_test.ipynb, where the format of USERID is 2017CSZ8058 and share them with keshavkolluru@gmail.com, vishalsaley114@gmail.com and kartikeya.badola@gmail.com. You will also have to share your final saved model. We will float a Google form where you will have to paste the view-only shareable links for the notebooks as well as the saved model.

For the notebooks and saved model, we will check the revision history to ensure that no changes have been made beyond the assignment deadline.

We will run the final model on an unseen test set and use the generated predictions to evaluate the final performance. Your assignment will be graded based on the relative ranking in the class.

In order to ensure integrity, we will randomly re-train your model with the original training/validation split and verify that the model achieves the same performance as reported in the validation scores Google form. Some minor discrepancies are acceptable due to randomness innate in Pytorch. However, significant differences may be treated as model plagiarism (such as copying model weights from another student) and will be dealt with accordingly.

We will do ‘Runtime → Run All’ in both the notebooks (after creating a copy) for training and testing your model.

We will allow a maximum runtime of 5 hours for training and 30 mins for testing on a K80 GPU (12 GB).

What is allowed? What is not?

General

1. The assignment is to be done individually.
2. You must use Python for this assignment.
3. You must not discuss this assignment with anyone outside the class. Make sure you mention the names in your write-up in case you discuss with anyone from within the class outside your team. Please read academic integrity guidelines on the course home page and follow them carefully.
4. We will run plagiarism detection software. Any team found guilty will be awarded a suitable penalty as per IIT rules.
5. Your code will be automatically evaluated. You get significant penalty if it is does not conform to output guidelines. Make sure it satisfies the format checker before you submit.

Specific:

6. This assignment is divided into two parts. In both parts, you are only allowed to use deep neural models and so you are not allowed to use sklearn.
7. In the current Part (A), you are only allowed to use Pytorch and Word2Vec, FastText, or Glove embeddings. You are not allowed to use any pre-trained models (such as BERT and GPT-2) or other publicly available deep learning libraries. You will be given more flexibility in Part (B), for which we will release a separate set of instructions.
8. We will accept case-by-case exceptions to certain libraries that may be used for the data processing pipeline. You can raise requests on Piazza and we will maintain a public list of acceptable external libraries.
9. Apart from the data already provided, you are not allowed to use any external resources such as WordNet or any other training data you may find online or you may decide to annotate on your own.

All the best!