ASSIGNMENT 2: NAMED ENTITY RECOGNITION

Motivation: The motivation of this assignment is to get practice with deep learning models for sequence labeling tasks such as Named Entity Recognition (NER).

Problem Statement: The goal of the assignment is to build a NER system for Bio-medical text extracted from scientific documents/articles. The input of the code will be a set of tokenized sentences and the output will be a label for each token in the sentence. Labels will be from 4 classes i.e. Chemical_Compound, Biological_Molecule, Species, and Others.

Labeled Training Data: We are sharing training and a dev dataset, named `train.txt' and `dev.txt' respectively. In the train/dev file, each line contains one token of a sentence, followed by a tab ("\t") and its token label. Sentences are separated by the blank lines. There are 38,769 labeled sentences (examples) in `train.txt' and 8,970 ones in `dev.txt'. We also provide the sample test input and output files to you. The labels are encoded using IOBES scheme.

IOBES Tagging Scheme: A further extension to the BIO labeling scheme((Read https://en.wikipedia.org/wiki/Inside%E2%80%93outside%E2%80%93beginning_(tagging))). It adds two new tags types. E is used to label the token that is the last item of a span. The new S prefix is used for span that only include a single token.

The Task: You need to write a sequence tagger that labels the given instructions in a tokenized test file. The tokenized test file follows the same format as training data except that it contains only a sentence token in each line immediately followed by newline character `\n' (i.e. no token labels in test file will be given at inference time). You should label the test file in the same format as the training data. The format of your output file will be the label (of string type) for each corresponding token of the test file in one line immediately followed by `\n' (do not output the token itself in output file but only the labels). So, the output will have the same no. of lines as the text file with matching blanks marking the end of sentence. A few points to note are following-

1. You need to use deep learning for this assignment. The allowed models are CNN, LSTM, Transformers (or any extension of these, including attention or custom-built models). However, you are NOT allowed to use any pretrained Language Models such as BERT, ELMO, GPT. You must train all models from scratch. You are allowed to use pre-trained word vectors from word2vec, Glove or FastText. If you wish to use any other pre-trained information, you should ask on Piazza.

2. You may like to create additional features for each token, e.g. whether the token is capitalized or not, whether it's a number or not etc. You may also try features from lower level syntactic processing like POS tagging or shallow chunking. You may use off-the-shelf POS taggers with proper reference/citation in the writeup file. However, such taggers (or other codes) cannot also use any pre-trained language models.

3. Defining task-specific features such as specific regular expressions indicative of specific types could be useful.

4. You may like to create your own tokenizer for the task. Please read the data carefully to design one.

5. You can use any off-the-shelf tool/code, but only for feature engineering (but not for making the NER model itself), as long as it does not use any pre-trained language model.

6. You are welcome to use probabilistic models like CRF on top of deep learning models. Example, read up on BiLSTM-CRF or Transformer-CRF models for the task of sequence labeling.

Submission Format:
The submission deadline for the assignment is 21st March 11:59 PM. We will follow a similar format as in A1, with changes in output format etc. for this problem. Please read below:

1. Submit a zip file on moodle with name <kerberos_id.zip> (E.g. `csz198394.zip’). Unzipping this should generate a directory with name <kerberos_id> (E.g. `csz198394’) having 3 files – `install_requirements.sh’, `run_model.sh’ and `writeup.txt’.

2. The command for training is - ` bash run_model.sh train <train_file_path> <val_file_path>’ . This should generate a tagging model named `<kerberos_id>_model’ (E.g. `csz198394_model’) along with possibly other files needed for inference.

3. Command for inference - `bash run_model.sh test <test_file_path> outputfile.txt’ . This should generate `outputfile.txt’ having predicted label in string format (`B-Action’, `I-Action’ etc.) followed by `\n’ in each line for the token in the corresponding line of test file.

Note:- Please see the sample input and output files provided to you. We will follow the exact same format for test input and expected output.

Evaluation Criteria:
1. This assignment is worth 100 points.
2. We will take the mean of micro-F1 and macro-F1 scores based on all NER labels excluding Other.
3. Bonus points awarded for outstanding performers.
4. There may be a secret bonus on an extra metric which will be announced after the submission deadline. (Hint: Data says everything :))

What is allowed? What is not?
1. The assignment is to be done individually.
2. You should use Python 3.7 and PyTorch 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. Please read academic integrity guidelines on the course home page and follow them carefully.
4. Feel free to search the Web for papers or other websites describing how to build named entity recognizers. Cite the references in your writeup.
5. As mentioned earlier, you are NOT allowed to use any pretrained LMs such as BERT, ELMO, GPT etc. or any software based on these.
5. We will run plagiarism detection software (amongst submissions + commonly available public repositories for NER). Any team found guilty will be awarded a suitable penalty as per IIT rules.
6. Your code will be automatically evaluated. You will get a significant penalty if it does not conform to output guidelines