BIOF 510 - Project #1 (35 points)

Learning Objectives:

- Increase familiarity with pytorch
- Learn how to use an LSTM model
- Learn how to integrate Deep learning models
- Learn how to process/handle a new type of data
- Practice neural network optimization

Assignment:

1. On canvas, I have provided you two text datasets
    a. train.text
    b. test.text

    Using pytorch, You will develop a POS tagger that uses both a bi-directional LSTM and a CNN for maximum accuracy. You can use the pytorch LSTM example and the CNN tutorial as a reference point.

    Required Structural Components: a word embedding, a CNN, a bi-directional LSTM, and linear layers.

    Tips/Hints:

    1. You will need to organize/process the data into python data structures that are organized/easy to use later on.

    2. You will need to convert words/characters/POS tags into a numeric representation (Pytorch cannot learn from text). We have seen an example of this….

    3. You will need to make sure all the inputs to both the CNN and the LSTM are the same length. What length should they be? How will we prevent our network from becoming confused? Hint: Look up the ignore_index parameter for the nn.CrossEntropyLoss function. What would you use that for?

    4. For your CNN, you will need to use conv1d instead of conv2 (words are 1-dimensional)

See next page

2. **Extra Credit -** For 5 points of extra credit, use a pre-trained transformer model to do the part-of-speech tagging. You can do this in Pytorch with the information provided in this link: https://pytorch.org/hub/huggingface_pytorch-transformers/. You can use the same data format/organization that you used for your LSTM-CNN model.

   HINT: the encoded outputs from your transformer model can be used as inputs into a basic ANN. The ANN will optimize the feature-weight combinations for better predictions (See basic ANN tutorial).

   You can use the pre-trained model directly on test.txt. However, for better accuracy, fine-tune the model on train.txt before generating labels on the test data.

Submission Information for Question 1: You should submit a .py file or a Jupyter notebook (with documentation) that will allow me to train your model on train.txt and test your model on test.txt. This assignment will be due on April 15th, 2021, at 11:59pm ET.

Grading Information:

25 points for implementation (code)

10 points for accuracy (graded on a scale relative to the overall performance in the class). **When I run your code, the accuracy values for the test dataset should be printed.** Especially clever ways of improving accuracy will be given 1 point of extra credit (Hint: punctuation).

6 points of total extra credit are possible

DO NOT COPY CODE FROM ONLINE