Named Entity Recognition (NER)

In the last assignment, you tried out using statistical models for NLP tasks.

In this assignment you'll be experimenting with deep learning based models for a sequence labelling task: Named Entity Recognition

Dataset and Task overview

First we provide a helper function which you can use for reading the data given.

Note: Feel free to augment the helper functions given in this notebook as per your need. As long as the overall objective is being met this shouldn't be an issue.

```python
def read_file(filename):
    with open(filename, "r") as file:
        text = file.readlines()
    return text

def process_text(text):
    X = []
    Y = []
    sentenceX = []
    sentenceY = []
    for line in text:
        split = line.split(" ")
        if len(split) > 1:
            sentenceX.append(split[0])
            sentenceY.append(split[1].replace("\n", ""))
        else:
            X.append(sentenceX)
            Y.append(sentenceY)
            sentenceX = []
            sentenceY = []
    return X, Y

text = read_file("data/train.txt")
X, Y = process_text(text)
```

Following is an example to visualize what is happening here:

```python
for i in range(len(X[1])):
    print(X[1][i], Y[1][i])
```

```
Chancellor O
of B-PP
the B-NP
Exchequer I-NP
Nigel B-NP
```

```
Lawson I-NP
's B-NP
restated I-NP
commitment I-NP
to B-PP
a B-NP
firm I-NP
monetary I-NP
policy I-NP
has B-VP
helped I-VP
to I-VP
prevent I-VP
a B-NP
freefall I-NP
in B-PP
sterling B-NP
over B-PP
the B-NP
past I-NP
week I-NP
. O
```

Notice how every token in the sentence has been assigned a label.

These labels here are being referred to as the named entity.

Note that we are following a BIO Tagging scheme here.

Basically, every token is either the beginning (B) of a chunk, the continuity of a chunk (I) or outside the chunk (O). E.g. "Barack Obama went to Greece today" -> “Barack B-PER Obama I-PER went O to O Greece B-LOC today O." Of course, there are other types of tagging schemes also possible like simply BO tagging, where I- is not explicitly tagged, and all contiguous tokens of the same type are combined to extract one entity. In such a schedule, the tagging will be “Barack PER Obama PER went O to O Greece LOC today O."

You can read more about it here.

Modelling

Your task in this assignment is to train a Deep Learning based model using the train set (feel free to split it into train and dev sets) and test out your models performance on a held out dataset (test set). A few points to note are the following:

- You need to use deep learning for this assignment. The allowed models are CNN, LSTM. **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. To avoid confusion, you will be evaluated on the perfomance on one of your models. It would be good if you can show a comparison between the various settings you have tried however implementing one model completely would be sufficient as well.

- 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. (This step is optional and meant for your learning).
- You are welcome to use probabilistic models like CRF on top of deep learning models. Example, read up on BiLSTM-CRF models for the task of sequence labeling.

Note: Use of only PyTorch or Tensorflow is allowed for the assignment. PyTorch is recommended. Don't use SpaCy

Evaluation

```
# Here we have assumed that the predictions and the true labels are
contained in a 1D array as shown below.
# If you have a 2D array containing predictions of each sentence in a
different array then please first flatten the array so that
predictions are contained sequentially.

predY_eg = ["B-NN", "O", "B-PP", "I-PP"]
trueY_eg = ["B-NN", "O", "B-PP", "B-PP"]
```

THe metrics we'll be using are Micro and Macro F1 scores.

You can make use of the following code for calculating the scores which we'll be using for evaluating the performance.

```
def get_scores(predY, trueY):
    from sklearn.metrics import f1_score
    trueY_O = [i for i, x in enumerate(trueY_eg) if x == "O"]
    predY = [predY[i] for i in range(len(predY)) if i not in trueY_O]
    trueY = [trueY[i] for i in range(len(trueY)) if i not in trueY_O]

    print("Micro F1 score: ", f1_score(trueY, predY, average="micro"))
    print("Macro F1 score: ", f1_score(trueY, predY, average="macro"))
    print("Average F1 score: ", (f1_score(trueY, predY,
average="micro") + f1_score(trueY, predY, average="macro")) / 2)

get_scores(predY_eg, trueY_eg)
```

```
Micro F1 score:  0.6666666666666666
Macro F1 score:  0.5555555555555555
Average F1 score:  0.611111111111111
```