Homework 1: Named-Entity Recognition

Due dates: part1 due February 8th, part2 due February 12, both are due @ 2:00pm CDT

Named-entity recognition (NER) is a subtask of information extraction that seeks to classify atomic elements in text into predefined categories such as the names of persons, organizations, places and more. The goal of this assignment is for you to learn how to **detect** and **classify** novel, emerging, singleton named entities in noisy and unstructured text.

Task

Your task is to implement a named entity recognizer using Conditional Random Fields (CRFs). Your code should be implemented in Python 3 and you are allowed to use available libraries such as scikit-learn.[1] The main steps you need to follow to achieve the goal of this assignment are: 1) **Data preprocessing** to transform raw data into a format suitable for input to the CRF model, 2) **Feature extraction** to discover what latent signals might be helpful to identify entities, 3) **Training** of your CRF model with your features, and 4) **Classification** using Conditional Random Field (CRF). For more information regarding the task, you can read J&M, Chapter 17.1. Recommended readings about CRFs: Esienstein, 7.5.3, Sutton and McCullum, CRFs.

Data

For this assignment, training and development sets come from WNUT corpus, and test data is from an anonymous social media platform. The format of the data is the same across all three different sets. Each row includes one word and the entity tag which is assigned to that word, and posts are separated with blank lines. The data uses IOB encoding, where each named entity tag is prefixed with a B-, which means beginning, or an I-, which means inside. The O tag is for non-entities. To make it more clear, please look at the following example:

```
Michael     B-person
Jordan      I-person
is          O
a           O
professor   O
at          O
Berkeley    B-location
.           O
```

Note that you do not have access to the tags for the test set. Data can be downloaded via the following links:

- Training set

[1] https://scikit-learn.org/stable/

- Development set
- Test set (no tag)

You should use training data for building your model, and development set for fine-tuning the hyper-parameters of the classifier or improving the feature set. You can also use your development set for evaluating and analyzing your model. Test data must only be used for the final evaluation of your model.

Baseline

We ran a simple baseline to give you some idea regarding the performance of the model. The baseline includes gazateer features generated from several lexicons and uses CRF on top for the final classification. This model was trained on training set and evaluated on development data.

Here are the results:

```
accuracy:  93.78%; precision:  38.74%; recall:  32.53%; F1:  35.36%
company            precision:  29.41%; recall:  25.64%; F1:  27.40%
group              precision:  16.67%; recall:  2.70%;  F1:  4.65%
location           precision:  42.86%; recall:  44.81%; F1:  43.81%
other              precision:  32.94%; recall:  21.21%; F1:  25.81%
person             precision:  47.22%; recall:  59.65%; F1:  52.71%
product            precision:  9.38%;  recall:  8.11%;  F1:  8.70%
title              precision:  0.00%;  recall:  0.00%;  F1:  0.00%
```

Deliverables

This assignment is to be completed individually. You will need to turn in the following:

1. **Part1 (prediction files):** This part is due on **February 8th, 2:00 PM (CDT)**. You need to deliver your test predictions in a txt file: test_prediction_1.txt. The format of the data in each row of this file must be:
   ```
   word1    prediction1
   word2    prediction2
   word3    prediction3
   ```
   Posts should be separated with a blank line You should submit this part via BlackBoard. Every one can submit up to three different prediction files using three best models that are trained on training data and tested on development set. Put all your prediction files (up to three) into a single zip file:
   `lastname_COSC6336_hw1_part1.zip`
   and include the txt file with the predictions and a readme file that briefly describes the specific features/model you have used for generating that specific output. For the submissions, please take into account the following requirements:

   - Keep the order of the documents the same as the original test set.
   - Use exactly the same format for your submissions as what we described above.
   - Use a specific **pseudonym** for all your submissions (add this name to your readme file). We will evaluate your models using seqeval[2] python library. Then, we will compare the performance of your best submitted model to other students' and rank your system based on that. The final ranking will be posted to Piazza on **February 10th by 10:00 AM**, and will be anonymized using the pseudonym you have provided.

[2] https://pypi.org/project/seqeval/

NOTE: Make sure you follow the format specifications; if the evaluation script breaks with your input you will loose any points associated with this aspect.

2. **Part2 (code & report):** This part is due on **February 12th, 2:00 PM (CDT)**, and you need to deliver the following items:

   - **Code:** should be written in python 3.
   - **Report:** you have to provide a technical report in pdf format describing your system, relevant experiments, and analyzing the results. The report should be written in ACL style format. You can either download the template here or open it on Overleaf. Do not include code on your report.

   You need to submit this part as a zip file named `lastname_COSC6336_hw1_part2.zip` to BlackBoard.

Grading Policy

For this assignment, we will use the following grading percentages:

- Overall performance of your model: 10%
- Originality of the model: 30%
- Analysis of results and findings: 30%
- Overall quality of write-up: 30%

If you have any questions regarding the assignment, please post it on Piazza or see us during office hours.