Project 2: Named Entity Recognition (50 points)

CS 6320: Natural Language Processing

Due date: 4/27/2020 02:30pm

Problem Definition and Data

For Project-2, you will implement a feature-driven model that extracts named entities (NEs) from a document. The goal is to extract relevant concepts from a document such as names of persons, locations, geo-political entities, etc. To motivate the idea, consider the document given below:

In ancient Rome GPE , some neighbors live in three CARDINAL adjacent houses. In the center is the house of Senex GPE , who lives there with wife Domina PERSON , son Hero PERSON , and several slaves, including head slave Hysterium and the musical's main character Pseudolus GPE . A slave belonging to Hero PERSON , Pseudolus GPE wishes to buy, win, or steal his freedom. One CARDINAL of the neighboring houses is owned by Marcus Lycus ORG , who is a buyer and seller of beautiful women; the other belongs to the ancient Erronius PERSON , who is abroad searching for his long-lost children (stolen in infancy by pirates). One day DATE , Senex GPE and Domina PERSON go on a trip and leave Pseudolus in charge of Hero PERSON . Hero PERSON confides in Pseudolus GPE that he is in love with the lovely Philia GPE , one of the courtesans in the House of Lycus ORG (albeit still a virgin).

To train your NER model, you are provided with the CoNLL-2003 English dataset [1]. This corpus describes four kinds of NE's: Locations (LOC), Organizations (ORG), Persons (PER) and Miscellaneous (MISC). The figure given below is a snapshot of how data is arranged in the corpus:

```
Peter           B-PER
Blackburn       I-PER

BRUSSELS        B-LOC
1996-08-22      O

The             O
European        B-ORG
Commission      I-ORG
said            O
on              O
Thursday        O
it              O
disagreed       O
with            O
German          B-MISC
advice          O
to              O
```

As you can see in the figure, each new line in the corpus represents a token and its associated label. This corpus follows the BIO notation for representing labels where B marks the beginning of a tag, I marks the continuation of a previous tag and O marks that the given token is a named entity. Also note that a new line marks the beginning of a new sentence in the document.

Task - 1: Data preprocessing (5 points)

As an initial pre-processing step, carry out the following:

- Extract all sentences, tokens and the associated NER tags from the corpus. Identify how many sentences, unique tokens and tags are present in this corpus.
- Convert all tokens to lowercase.
- Replace each tag by a unique identifier integer.

Task - 2: Feature engineering (15 points)

Additionally, for each token, extract its lemma and POS tag by consulting an API like NLTK [2]. Note that to extract these features, you must pass the entire sentence as an argument to the associated functions as the POS tag of a word depends on its context. For example, the POS tag of the word 'race' depends on the context in which it is used. In the sentence "The horse is expected to race tomorrow", its POS tag is VB and in the sentence "The race for outer space", its POS tag is NN.

Represent the lemma and POS tag of each token in your corpus as two one-hot vectors. Recall that a one-hot vector is a binary vector which is used to distinguish each word or tag in a vocabulary or tagset from every other word in the vocabulary or tagset. The vector consists of 0s in all cells with the exception of a single 1 in a cell used uniquely to identify that word. Thus if there are V words in your vocabulary and t tags in your tagset, you will create a V - dimensional vector for the lemma of your token and a t - dimensional vector for the token's POS tag. Concatenate the two representations to create a $(V + t)$ - dimensional vector for each token.

Task - 3: Learning (10 points)

Use your favourite machine learning (not deep learning) algorithm to train a system that recognizes NER tags for a given word. Specifically, you will use the feature vectors created in Task-2 as inputs to the machine learning model. The output of the model will be the tag associated with that token.

Note that you may find some words in the test set which were not present in the training set. This is also known as the OOV (out-of-vocabulary) problem in NLP. To handle OOV words, you must create a special 'UNK' token in the vocabulary you created in Task-1. The lemmas of all OOV words must be replaced by this UNK token.

Task - 4: Model performance (10 points)

Report the model's performance on the supplied test set. Specifically, you will report the accuracy, precision, recall and F-score on the test set. Additionally, report the throughput of your system at inference time (you must report both the time taken as well as the throughput in kbps)

To Submit

Submit the following:

1. Your source code (either as a notebook or regular code file)
2. Instructions on how to run the code
3. A text file outlining your model's performance on the test set

External Links

1. NLTK: An easy-to-use API for NLP
2. SpaCy: An industrial-strength NLP tool
3. CoreNLP: Java library released by Stanford for common NLP tasks
4. MITIE: A C++ library for Information Extraction
5. Scikit-learn: Machine Learning with Python
6. OpenML with Java
7. A very simple tutorial for NER with scikit-learn

References

[1] Erik F Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*, 2003.

[2] Edward Loper Bird, Steven and Ewan Klein. Natural language processing with python. *O'Reilly Media Inc.*, 2009.