AdvNLE Lab 7 Named Entity Recognition

Julie Weeds

February 25, 2021

1 Getting started

`nltk` provides an interface to the Stanford CoreNLP tools — which include amongst other things a n NER tagger. See https://github.com/nltk/nltk/wiki/Stanford-CoreNLP-API-in-NLTK for more information. You need a java JRE to be installed and you also need to download the software and models from Stanford (see the above website).

First, at the command line, make sure you have the most up-to-date version of nltk.

```
conda upgrade nltk
```

Then, from the command line, download the necessary CoreNLP packages

```
cd ~
wget http://nlp.stanford.edu/software/stanford-corenlp-full-2018-02-27.zip
unzip stanford-corenlp-full-2018-02-27.zip
cd stanford-corenlp-full-2018-02-27
```

Then, also from the command line, start the server

```
java -mx4g -cp "*" edu.stanford.nlp.pipeline.StanfordCoreNLPServer \
-preload tokenize,ssplit,pos,lemma,ner,parse,depparse \
-status_port 9000 -port 9000 -timeout 15000 &
```

You can use the tools interactively by navigating to http://localhost:9000 on any web browser. You can also interact with the server from python (e.g., in a Jupyter notebook):

```
from nltk.parse.corenlp import CoreNLPParser

ner_tagger=CoreNLPParser(url="http://localhost:9000",tagtype="ner")
sentence="John Smith is a student at Sussex University in the UK"
tokens=sentence.split(" ")
print(tokens)
tagged= ner_tagger.tag(tokens)
print(tagged)
```

You may need to set your JAVA_HOME environment variable. You can do this via Jupyter using the following code. Find where your java.exe is in your directory structure and make this the value of `javapath`

```
import os
java_path = "C:/Program Files/Java/jdk1.7.0_11/bin/java.exe" #edit this
os.environ["JAVA_HOME"] = java_path
```

See https://www.nltk.org/_modules/nltk/parse/corenlp.html for more information about the nltk interface to CoreNLP and https://stanfordnlp.github.io/CoreNLP/ for more information about the tools more generally.

2 Tasks

1. Try using different sentences as input to the tagger. What happens if the text has multiple sentences? Can you provide a tagging for a complete textfile. Write it out to file in the standardly employed CONLL column format.

   ```
   1 John PERSON
   ```

```
2 Smith PERSON
3 is O
4 a O
5 student O
6 at O
7 Sussex ORGANIZATION
8 University ORGANIZATION
9 in O
10 the O
11 UK LOCATION
12 . O
13 He O
14 likes O
15 Coldplay PERSON
16 . O
```

2. Look at the documentation for the CoreNLP named entity recogniser https://stanfordnlp.github.io/CoreNLP/ner.html.
   - Can you find out the set of tags used by the system?
   - Are there options to use models with different tag sets?
   - Unfortunately, the nltk interface to the Stanford CoreNLP server doesn't seem to allow the setting of properties which would allow the use of different models. However, it is possible to do this if you use the stanfordcorenlp python library. You will need to install this into your environment (using pip as it is not in the conda repository) and then you can run the following code.

```
from stanfordcorenlp import StanfordCoreNLP
nlp = StanfordCoreNLP("http://localhost", port=9000)
props={"annotators":
    "ner","pipelineLanguage":"en","outputFormat":"conll","ner.applyFineGrained":"false"}
print(nlp.annotate(sentence,props))
```

3. Find a short piece (about 10 sentences) of newswire text on the Internet e.g., from the BBC website. Provide a GOLD STANDARD named entity annotation of the text using the same tagset as one of the models for the Stanford NER tagger. It is **very** important that you generate the gold standard **without** running it through the tagging software. Use the same CONLL format for your gold standard as before.

4. Take the piece of text chosen by of your peers and also provide a gold standard annotation of this text. Compare the gold standard annotations for the different texts. Do the annotators agree?

5. Tag the text using the Stanford NER tagger. Evaluate the output against your gold standard in terms of tag accuracy (% of correct tags); precision and recall of each tag type and precision and recall of each entity type. What do you think of the results? What kind of errors is the system making? Are they recognition or classification errors? Were the errors on tags where you might expect there to be inter-annotator disagreement?

6. Repeat the experiment with a piece of text from a different domain e.g., a blog post or a product or movie review. How do the results compare?

3 Extension

Experiment with the NER tagger provided in the SpaCy library (see the documentation https://spacy.io/api). How does it compare to the Stanford NER tagger?