IHLT Laboratory

Jordi Turmo
TALP Research Center
turmo@cs.upc.edu

Session 7

Lexical level in nltk library: POS models

Different options:

- Use the default POS tagger (averaged perceptron) or a predefined one
- Learn a POS tagger
  - Statistical: HMM, TnT, perceptron, CRF (requires `pip3 install python-crfsuite`)
  - Rule based: Brill
- Use third-parties' code

  Senna, Stanford, hunpos

Lexical level in nltk library

Example: HMM

```
from nltk.tag.hmm import HiddenMarkovModelTrainer
from nltk.corpus import treebank

train_data = treebank.tagged_sents()[:30]
test_data = treebank.tagged_sents()[3000:]

trainer = HiddenMarkovModelTrainer()
HMM = trainer.train_supervised(train_data)
print(HMM.evaluate(test_data))
print(HMM.tag(['the','men','attended','to','the',
'meetings']))
```

```
0.10628102741204404
[('the','DT'),('men','NNP'),('assisted','NNP'),('to','NNP'),
('the','NNP'),('meetings','NNP')]
```

Lexical level in nltk library

Example: TnT

```
import nltk;
from nltk.corpus import treebank

train_data = treebank.tagged_sents()[:30]
test_data = treebank.tagged_sents()[3000:]

from nltk.tag import tnt
TnT = tnt.TnT()
TnT.train(train_data)
print(TnT.evaluate(test_data))
print(TnT.tag(['the','men','attended','to','the',
'meetings']))
```

```
0.4571551910209367 7
[('the','DT'),('men','NNS'),('attended','Unk'),('to','TO'),
('the','DT'),('meetings','Unk')]
```

Lexical level in nltk library

Example: Perceptron

```
import nltk
from nltk.tag.perceptron import PerceptronTagger
from nltk.corpus import treebank

train_data = treebank.tagged_sents()[:30]
test_data = treebank.tagged_sents()[3000:]

PER= PerceptronTagger(load=False)
PER.train(train_data)
print(PER.evaluate(test_data))
print(PER.tag(['the','men','attended','to','the',
'meetings']))
```

```
0.6549967623570041
[('the','DT'),('men','NN'),('attended','VBG'),('to','TO'),
('the','DT'),('meetings','NN')]
```

Lexical level in nltk library

Example: CRF

```
import nltk
from nltk.tag import CRFTagger

train_data = treebank.tagged_sents()[:30]
test_data = treebank.tagged_sents()[3000:]

CRF=CRFTagger()
CRF.train(train_data,'crf_tagger_model')
print(CRF.evaluate(test_data))
print(CRF.tag(['the','men','attended','to','the',
'meetings']))
```

```
0.6848694150658321
[('the','DT'),('men','NN'),('attended','VBD'),('to','TO'),
('the','DT'),('meetings','NNS')]
```

Lexical level in nltk library: POS models

Save/Load a learned model:

- CRF uses their own.
  - Training and save: `CRF.train(train_data,"file_name")` as saw before
  - Load: `CRF.set_model_file("file_name")`
- HMM, Perceptron and TnT can use `dill`. Perceptron and TnT can also use `pickle` using, in both cases, `dump` and `load` functions.

Example:

```
import dill
f = open("tnt_treebank_pos_tagger", "wb")
dill.dump(TnT, f)
f.close()

f1 =open("tnt_treebank_pos_tagger", "rb")
c=dill.load(f1)
f1.close()
```

Session 7

Lexical level in nltk library: NERC models

Default model

- maximum entropy model (PERSON, LOCATION, ORGZATION) trained with ACE corpus

```
from nltk import ne_chunk
ne_chunk(POS_tagged_sent, [binary=True/False])
```

- binary: True is used just to recognize NEs, without classifying them into the three NE classes. By default, it is set to False
- Output: tree format by default. Use `nltk.chunk.tree2conllstr(ner_output)` to get CoNLL format

- NLTK doesn't have an English corpus for NERC (CoNLL 2002 corpus for Spanish and Dutch)

Lexical level in nltk library: NERC models

Example

```
import nltk
from nltk import word_tokenize, pos_tag, ne_chunk

sentence = "Mark Pedersen and John Smith are working at
Google since 1994 for $1000 per week."
x=pos_tag(word_tokenize(sentence))
print(ne_chunk(x),"\n")
print (ne_chunk(x, binary=True))
```

```
(S
  (PERSON Mark/NNP)
  (ORGANIZATION Pedersen/NNP)
  and/CC
  (PERSON John/NNP Smith/NNP)
  are/VBP
  working/VBG
  ...)
```

```
(S
  (NE Mark/NNP Pedersen/NNP)
  and/CC
  (NE John/NNP Smith/NNP)
  are/VBP
  working/VBG
  at/IN
  ...)
```

Lexical level in nltk library: NERC models

Third party model: Stanford CoreNLP server

- CRFs and rule models (PER, LOC, ORG, DATE, TIME, MONEY, ...)

```
from nltk.tag.stanford import CoreNLPNERTagger

CoreNLPNERTagger(url='http://localhost:9000').tag(tokenized_sent)
```

  make sure you have installed at least nltk version 3.2.5

- Previously, install CoreNLP:

  `https://stanfordnlp.github.io/CoreNLP/download.html`

- and execute CoreNLP server: `java -mx4g -cp "whole_path/stanford-corenlp-full-2017-06-09/*" edu.stanford.nlp.pipeline.StanfordCoreNLPServer -port 9000 -timeout 15000`

  make sure you have intalled the later java version

Lexical level in nltk library: NERC models

Example

```
import nltk
from nltk import word_tokenize
from nltk.tag.stanford import CoreNLPNERTagger
sentence = "Mark Pedersen and John Smith are working at
Google since 1994 for $1000 per week."
print(CoreNLPNERTagger(url='http://localhost:9000').tag(word
_tokenize(sentence)))
```

```
[('Mark','PERSON'),('Pedersen','PERSON'),('and','O'),
('John','PERSON'),('Smith','PERSON'),('are','O'),
('working','O'),('at','O'), ('Google','ORGANIZATION'),
('since','O'),('1994','DATE'),('for','O'),
('$','MONEY'),('1000','MONEY'),('per','O'),
('week','DURATION'),('.','O')]
```

Exercises

A) Consider Treebank corpus. Train HMM, TnT, perceptron and CRF models using the first 500, 1000, 1500, 2000, 2500 and 3000 sentences. Evaluate the resulting 24 models using sentences from 3001.

Provide a figure with four learning curves, each per model type (X=training set size; Y=accuracy). Which model would you select? Justify the answer.

B) Read the three first pair of sentences of the training file within the evaluation framework of the project. Compute their similarities by considering the following approaches:

- words and Jaccard coefficient (same as in Session 5)

  ex: `words=['John','Smith','is','working']`
- words plus NEs and Jaccard coefficient

  ex: `word_and_NEs=['John Smith','is','working']`

Print the results. Do you think it could be relevant to use NEs to compute the similarity between two sentences? Justify the answer.