LING439/539 - Statistical NLP
Submission of homework
Assignment #1

MITHUN PAUL

Due Thursday, September 8 2016 at 11:00AM

Prepare your corpus for POS tagging

1. download the following page from `http://tucson.com`
   - `http://tucson.com/business/tucson/south-tucson-motel-to-be-torn-down-for-new-development/article_f3305b92-238f-5a63-ad01-c474ccaddb56.html`
   - you can use a command `wget`

   Ans: I used CURL instead. Here is the command I used:

   ```
    curl http://tucson.com/business/tucson
   /south-tucson-motel-to-be-torn-down-for-new-development
   /articled01-c474ccaddb56.html > tucsonmotel
   ```

   The thus downloaded document is attached herewith as:tucsonmotel.
   Note: might have to use an editor like vi/emacs to see the html tags. Other editors try to act smart and interpret the html tags already.
2. Propose a regular expression to remove html tags (3pts)
   Ans: I tried using the below sed command:

   ```
   cat tucsonmotel_removedhtml.txt | sed  ‘s/<html>//g’
   ```

- **Qn 2.1 : it works ? explain the problem if any.**
- **Ans:** Right now, it picks only the entire $< html >$ tags. However, they are very few of them.
- Most of the tags have more data to the end of the word html. EG: $< htmllang = "en" >$
- Below are a few upgrades to the sed command I devised.

-
  ```
  sed -e 's/^<html//g' test > edited_test
  cat tucsonmotel_removedhtml.txt | sed 's/^<html//g'
  sed -e 's/^<html//g' test > edited_test
  ```
- I realized that “cat” is just printing it to tty(Terminal). And not to the file. So I am modifying the sed command to act on the file itself.

  ```
  sed -e 's/^<html//g' test > edited_test
  ```
- This deletes all the lines starting with html. however it still doesnt capture the below tag
- $<!DOCTYPEhtml>$
- Hence Modifying sed again.

-
  ```
  sed -e '/html>*$/d' tucsonmotel.txt > edited_tucsonmotel
  ```

- I thought atleast it would work this time and remove all lines ending with html. Apparently not. There is `"^M"` character at the end of each line.
-
- **Qn 2.2: Find other solutions in the Web (keyword: boilerplate) and describe it (optional)**
- Ans: Here is a boiler plate code I found from stackoverflow:
- `sed -e 's/<[^>]*>//g' file.html`
- `http://stackoverflow.com/questions/19878056/sed-remove-tags-from-html-file`
- `http://stackoverflow.com/questions/30817035/remove-boilerplate-content-from-html-page`
-
- **Qn 2.3: otherwise, you can always remove html tags manually...**
- Ans: done. Am attaching the hand edited file (edited_tucsonmotel) herewith. I just did a replace (ctrl+H) in a text editor.

3. Normalize symbols in your text (1pt)

Ans: I used an online tool `https://try.jsoup.org/` to normalize. This tool takes an html input, removes the unwanted characters and tags and returns an UTF8 encoded text.
In addition to that I used inputs from The Wolfram Language, which provides powerful knowledge-based tools for normalizing text in preparation for text analysis, visualization, etc. Also I manually removed some curly braces. The post normalized version is kept in the file(qn2Normalization)attached herewith.
Note: please use vi/emacs.

4. Detect sentence boundaries (3pts)
   Ans: I was going to start with a simple sed command to detect period/fullstop to earmark sentence boundaries. However, I realized that it gets complicated. The sentences can end with other punctutation marks like ! or etc. Hence I have decided to use ready made tools.
   - **Qn 4.1:** find existing solutions in the Web and describe it
   - **Ans:**
   - `http://text0.mib.man.ac.uk:`

8080/scottpiao/sent_detector

- I think what it does is to decide when single quotes are parts of words, when periods do an don't imply sentence boundaries, etc. Sentence splitting is a deterministic consequence of tokenization: a sentence ends when a sentence-ending character (., !, or ?) is found which is not grouped with other characters into a token (such as for an abbreviation or number), though it may still include a few tokens that can follow a sentence ending character as part of the same sentence (such as quotes and brackets).
- Other tools I found are:
- http://weblicht.sfs.uni-tuebingen.de/weblichtwiki/index.php/Tokenizer_and_Sentence_Boundary_Detector_Service
- http://textminingonline.com/tag/sentence-boundary-detection (This tool is built on top of NLTK)
- Also, Stanford NLP has its own versions of sentence boundary detectors.
- http://nlp.stanford.edu/software/tokenizer.shtml
-

  - **Qn 4.2** one sentence per line
  - I fed the text from a wikipedia page (`https://en.wikipedia.org/w/index.php?title=Natural_language_processing&printable=yes`) to this (`http://text0.mib.man.ac.uk:8080/scottpiao/sent_detector`) sentence boundary detector.
  - The resulted text was in one sentence per line. The resulting text can be found in the attached file titled:qn4.2.pdf

5. tokenize your text[1] (1pt)
   Ans: I used the online tokenizer given here `https://open.xerox.com/Services/fst-nlp-tools/Consume/Tokenization-175`. on the text mentioned above: The results are kept in the file: qn5.1.pdf attached herewith.
6. Calculate the number of tokens and sentences (1pt)
   Ans: I used the wc command on unix and fed it the aforementioned tokenized text. This can be found attached herewith as qn6.txt

   ```
   $ wc qn6.txt
   ```

```
     202     303    1637 qn6.txt
```

As per this there are 202 words- which are tokens in this case (since it is already tokenized).
To find the number of sentences i used another tool given here: `http://textmechanic.com/text-tools/basic-text-tools/count-characters-words-lines/`.
I fed it the NLP wikipedia page (attached herewith as nlp_wiki.txt). As per this tool there are 244 sentences in this document.
Note: I could have used wc -l but that just looks for a newline character.

7. Annotate your text with part-of-speech using the TreeTagger. The tagger is available at `http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger` (1pt).
   Ans: I used multiple methods for this

   7.1 I downloaded the POS tagger from muenchen as mentioned above. Installed it. Modified the path variables. Worked fine with the sample command as below.

```
 echo ‘hello world is here’ | cmd/tree-tagger-english
reading parameters ...
tagging ...
 finished.
hello NN hello
world NN world
is VBZ be
here RB here
```

Also had to go through the read me file to find the exact command.
The input file was the file tokenized earlier (attached herewith as tokenized.txt). Took some time to figure out how to input the exact command along with the english parameter file. Exact command used is as follows:

```
tree-tagger english-par-linux-3.2-utf8.bin tokenized.rtf
```

The output can be found in the attached file:taggerOutput.txt

7.2 Also alternately I used the POS tagger :
`https://open.xerox.com/Services/fst-nlp-tools/Consume/Part%20of%20Speech%20Tagging%20(Real%`

`20Time)\discretionary{-}{}{}181`

I fed it the Gettysburg address by Abraham Lincoln. The results are kept in a file titled qn7.pdf and attached herewith.

7.3 I also went one step further and used a Tagger we have developed in our lab. The resultant file can be found attached herewith , with the name:OdinBioRulesResultsVisualization.pdf The tagger used can be found here:`http://agathon.sista.arizona.edu:8080/odinweb/bio/enterText`

PostScript/NoteBena: Hope this fetches me some extra points and be pardoned for the late submission.

7.4

describe each step within 2 pages max and send it with the result file (TreeTagger tagged) to `jungyeul@email.arizona.edu` before 11:00AM on Thursday, September 8.