Lemmatization and Stemming Methods in CL: Lab Session - Solutions

Diego Frassinelli

November 27 2019

The main goal of this lab session is to make you familiarize with the concepts of *lemmatization*, and *stemming*. In this session, you will use a set of tools to lemmatize and stem a short text. In this handout there are **3 tasks** to perform.

Practicalities

The participation in this lab session is voluntary: you do not have to submit any result. If you want to get feedback, please ask questions during today's session. Moreover, we will discuss the results of this handout in the next lab session.

If you find the programming part too hard, share a computer with a more experienced student (but do not be passive) and try to understand what (s)he is doing and why.

To solve the tasks, you can use the programming language you prefer (e.g., shell, python). If not clearly stated, avoid to solve the problem manually. Feel free to work in small groups (max 3 people) and discuss the outcome of the different steps together.

In this handout you will find a series of review questions (**Review**) about various topics we covered in the previous lectures. Try to answer to these questions and discuss your answers with the other students you are working with.

Task 1

In the first Lab session, you've learned how to tokenise a file. Now you will use the Stanford pipeline to lemmatize the same text.

1. Using the examples from the Stanford webpage, and the code you already used to tokenize the text, lemmatize the text from `guardian.html`.

```
java -cp "/mount/studenten/MethodsCL/2019/NLP/tools/stanford-corenlp/*"
  -Xmx2g edu.stanford.nlp.pipeline.StanfordCoreNLP -annotators
  tokenize,cleanxml,ssplit,pos,lemma -file guardian.html -outputFormat conll
```

2. Open the output file and check the lemmatization results. Do you see any strange behaviour?
   - Xi, ASX200→ case folding → NN

3. Extract the lemmas from the output file and generate a new file (`guardian.lemma`) with them.

```
awk -F '\t' '{print $3}' guardian.html.conll > guardian.lemma
```

4. Count the number of tokens in the lemmatized file: 538 tokens.

```
wc -w guardian.lemma
```

5. Generate an ordered (decreasing) frequency list of the word types and save it in `guardian.lemma.freq`.

```
grep -v '^\s*$' guardian.lemma | sort| uniq -c | sort -nr > guardian.lemma.freq
```

6. Count the number of word types: 266 types.

```
wc -l guardian.lemma.freq
```

7. Look into the list and check the nature of the words that are more frequent and those that are less frequent. Can you see any pattern?

```
head guardian.lemma.freq
tail guardian.lemma.freq
```

Task 2

The Porter Stemmer is a very famous stemmer for English.

8. **Review:** What does a stemmer do?

   A stemmer deletes word endings following a series of rules.

9. **Review:** What are the differences between lemmatization and stemming?

Lemmatization	**Stemming**
Language Dependent	Language Dependent
Reduces Inflexional forms	Reduces Inflexional and Derivational forms
Principled Reduction	Crude Heuristic
A lot of Linguistic Knowledge	Not much Linguistic Knowledge

In the folder:

`/mount/studenten/MethodsCL/2019/NLP/tools/porterStemmer`

there is a python implementation of the Porter Stemmer (`stemmer.py`). To know more about it, just visit this webpage: `http://tartarus.org/martin/PorterStemmer/`. This script requires in input a file with a verticalized list of words. Note that you need an empty line at the end of the list.

10. Take `guardian.token` from Lab1, run the stemmer script on it, and save the output in `guardian.stem`.
11. Generate an ordered (decreasing) frequency list of the word types and save it in `guardian.stem.freq`.

```
sort guardian.stem | uniq -c | sort -nr > guardian.stem.freq
```

12. Count the number of word types: 271 types.

```
wc -l guardian.stem.freq
```

13. Look into the list and check the nature of the words that are more frequent and those that are less frequent. Can you see any pattern?

```
head guardian.stem.freq
tail guardian.stem.freq
```

Task 3

In Lab1 and Lab2 you have processed the input text in different ways: tokenization, lemmatization and stemming.

14. Compare and discuss the outcomes of these three processes. In particular consider:

    - the number of types;
    - the number of tokens;
    - the top and bottom lines in the frequency lists.