Tokenization
Methods in CL: Lab Session - Solutions

Diego Frassinelli

November 25 2019

The main goal of this lab session is to make you familiarize with the concepts of *tokenization*. In this session, you will use a set of tools to tokenize a short text. In this handout there are **4 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.

Before you Start

Before starting with this tutorial, open the terminal and create a new directory in your personal space called MethodsCL. Use it to store the files you will create in all the lab sessions:

```
mkdir -p ~/MethodsCL/lab1
```

This code will create (`mkdir`) the directory `MethodsCL` and its subdirectory `lab1` in your home folder (∼). The `-p` parameter allows you to create parent directories (e.g., MethodsCL) if they do not exist yet. Today, you will work in the subfolder `lab1`: download from ILIAS the file `guardian.html` and `guardian.html.original`.

Task 1

In `guardian.html.original` you can find an article downloaded from `https://www.theguardian.com`. Open this file and have a look at it. As we discussed in class, using text from the web has clear advantages.

1. **Review:** Do you remember any of these advantages?

- Huge amount of linguistic data
- Data already digitised and ready to be processed
- Language in use from different domains

However, we also know that the data from the web is more noisy and needs more pre-processing.

2. Which type of noise is present in this text?

- Html Tags
- Style info
- Intervening text (e.g., figures)

Task 2

Stanford University provides a set of free tools to perform various NLP tasks.

I have downloaded for you the Stanford CoreNLP from: `https://stanfordnlp.github.io/CoreNLP/`. Please, visit this webpage and read the "About" and the "Usage" sections. Spend some time trying to understand the use of this tool because we will use it multiple times!

You can find the tools in the following directory:

`/mount/studenten/MethodsCL/2019/NLP/tools/stanford-corenlp`

Today, you will use some of these tools to process the content of `guardian.html`. Please, open the file and look at its content. This represents a partially pre-cleaned version of the original file `guardian.html.original`. First of all, you have to tokenize the text.

3. Using the terminal go to your `lab1` directory (where you should see only `guardian.html` and `guardian.html` → type "ls" to check the content of the directory).

4. Once there, start the tokenization process typing (or copying) the following command (everything on one line!):

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

5. Using the documentation available online, try to understand the syntax of this command. In particular the -annotators and -outputFormat parameters.

6. At the end of the processing, you should see (type "ls") the new file `guardian.html.conll` (Tab separated file). Which information does the file contain?

   - Word order in the sentence
   - Tokens
   - Sentence split → empty line

7. In class we discussed many tokenization problems: how does the CoreNLP tools deal with them?

   - face-to-face, export-oriented

- China - 's
- 3.3 -
- $ - 1.30

8. Extract (manually or automatically) only the verticalized list of tokens, add an empty line at the end (needed for further processing) and save it into a new file `guardian.token`.

```
awk -F '\t' '{print $2}' guardian.html.conll > guardian.token
```

Task 3

Now that you have tokenized the text, you can compute some statistics on it.

9. Count the number of tokens in the document (`guardian.token`): 538 tokens.

```
## we have to remove empty lines from the count of tokens:
grep -v '^\s*$' guardian.token | wc -l
## -v output everything except for match

## or we can use
wc -w guardian.token
```

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

```
grep -v '^\s*$' guardian.token | sort | uniq -c | sort -nr > guardian.freq
## -c count number of duplicates
## -n numeric
## -r reverse
```

11. 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.freq
tail guardian.freq
```

12. Count the number of word types: 289 types.

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

13. Plot (if needed, use OpenOffice) the frequency distribution of the word types. The plot will have on the x-axis the rank (order in the list: e.g. type1 would have rank1) and on the y-axis the frequency. Already with such a small corpus, you should see some interesting results. Does this distribution look familiar to you?

```
gnuplot
gnuplot> plot 'guardian.freq' with impulses
```

14. **Review:** What does the Zipf's "Law" state? Why is it distribution important in NLP?

- The frequency of any word is inversely proportional to its rank
  (a) Few very common words
  (b) Middle number of medium frequency words
  (c) and many low frequency words

Task 4 (optional)

Procede to this task only if you are done with the previous 3 tasks and you still have some time left.

15. Open the file `guardian.html.original` and identify all the noise in this html file.
16. Write a script that removes all the noise from this file. The output should be the text of the article that could be processed by the CoreNLP tools without the use of the "cleanxml" parameter.

Super cool resource: `http://bootcat.dipintra.it/`