**Topic 3**

**Automatic annotation of a text file and creation of frequency lists**

23.04.2019

Aims of this class:

1) Learn how to read a text file in R

2) Practice how to annotate the text file and understand the output

3) Learn how to obtain the number of tokens and types

4) Make frequency lists

Data: the Universal Declaration of Human Rights, file UDHR.txt. Download it from GitHub and save locally, noting the location.

**1. Reading a text file in R**

To read a text file, you need to save it in .txt format (option: save as text file). A tip: use Notepad++ for processing your text. After that, you can use the function scan() in R:

**declaration <- scan(file = file.choose(), what = "character", sep = "\n", comment.char = "#", encoding = "UTF-8")**

You will be able to choose the file interactively with the help of the *Select File* window.

**head(declaration)**

[1] "Article 1."

[2] "All human beings are born free and equal in dignity and rights. They are endowed with reason and conscience and should act towards one another in a spirit of brotherhood."

[3] "Article 2."

… and so on…

The function head shows the first six items. These are the individual paragraphs in the text file.

**2. Annotation of text**

Now we are ready to annotate the file using the udpipe package, as we did in the previous session. First, we load the package and the language module:

**library(udpipe)**

**ud\_eng <- udpipe\_load\_model("english-ewt-ud-2.3-181115.udpipe")**

If you haven’t installed the package or haven’t loaded the English module, you can find the instructions in Topic 2.

**declaration\_ud <- udpipe(declaration, ud\_eng)**

**head(declaration\_ud)**

#output omitted

The object is a dataframe. It is like a huge spreadsheet with many columns and rows. The columns are different types of information. Some of them are more useful, some are less.

**colnames(declaration\_ud)**

[1] "doc\_id" "paragraph\_id" "sentence\_id" "sentence"

[5] "start" "end" "term\_id" "token\_id"

[9] "token" "lemma" "upos" "xpos"

[13] "feats" "head\_token\_id" "dep\_rel" "deps"

[17] "misc"

Now let’s examine the horizontal structure. Below are the first, second and third sentences (only some columns are shown for reasons of space):

**declaration\_ud[1:3, c(1, 3, 8, 9)]**

doc\_id sentence\_id token\_id token

1 doc1 1 1 Article

2 doc1 1 2 1

3 doc1 1 3 .

**declaration\_ud[4:16, c(1, 3, 8, 9)]**

doc\_id sentence\_id token\_id token

4 doc2 1 1 All

5 doc2 1 2 human

6 doc2 1 3 beings

7 doc2 1 4 are

8 doc2 1 5 born

9 doc2 1 6 free

10 doc2 1 7 and

11 doc2 1 8 equal

12 doc2 1 9 in

13 doc2 1 10 dignity

14 doc2 1 11 and

15 doc2 1 12 rights

16 doc2 1 13 .

**declaration\_ud[17:35, c(1, 3, 8, 9)]**

doc\_id sentence\_id token\_id token

17 doc2 2 1 They

18 doc2 2 2 are

19 doc2 2 3 endowed

20 doc2 2 4 with

21 doc2 2 5 reason

22 doc2 2 6 and

23 doc2 2 7 conscience

24 doc2 2 8 and

25 doc2 2 9 should

26 doc2 2 10 act

27 doc2 2 11 towards

28 doc2 2 12 one

29 doc2 2 13 another

30 doc2 2 14 in

31 doc2 2 15 a

32 doc2 2 16 spirit

33 doc2 2 17 of

34 doc2 2 18 brotherhood

35 doc2 2 19 .

In this annotation, each paragraph of our text file is represented in doc\_id, the sentences within the paragraphs are shown in sentence\_id, and the words within each sentence can be identified by token\_id.

**3. A first impression: the number of types and tokens**

The code below will show the number of rows. Each row is a token (a word, number or a punctuation mark). This is the number of **tokens**.

**nrow(declaration\_ud)**

[1] 1684

The number of types depends on what types we need: wordforms, lemmas or something else. We can use the function table() to obtain the frequencies of each individual type. Let’s first do it for the wordforms (note that the upper- and lowcase variants are treated as different).

**table(declaration\_ud$token)**

#output omitted

**length(table(declaration\_ud$token))**

[1]481

Therefore, we have 481 different wordforms (**types**).

For lemmas, we need to choose the column ‘lemma’:

**table(declaration\_ud$lemma)**

**length(table(declaration\_ud$lemma))**

[1] 443

Not surprisingly, the number of distinct lemmas is slightly less than the number of distinct wordforms.

**4. Making frequency lists**

In fact, we have already created frequency lists with the help of table(). Now, let us learn to sort them alphabetically and analyse the top 20. We can begin with the lemmas:

**freq\_lemma <- table(declaration\_ud$lemma)**

**sort(freq\_lemma, decreasing = T)**

#output omitted – too many words

**sort(freq\_lemma, decreasing = T)[1:20]**

the . and to , of be

95 90 80 74 70 65 48

right in ( ) article everyone have

34 33 32 32 30 30 30

or he shall a any 1

30 25 25 19 18 14

**Exercises**

1. Find the top 20 most frequent wordforms (tokens). What are the differences in comparison with the lemmas?

2. Create a frequency list of parts of speech (all, not only top 20).

3. Create a frequency list of the syntactic dependencies (all).

It can also be revealing to select specific parts of speech and make frequency lists only for them. For example, we can find the frequencies of all nouns in the text:

**freq\_nouns <- table(declaration\_ud$lemma[declaration\_ud$upos == "NOUN"])**

**sort(freq\_nouns, decreasing = T)[1:20]**

right article freedom rights

34 30 14 11

one law protection country

9 7 7 6

education family religion act

5 5 5 4

declaration discrimination limitation nationality

4 4 4 4

offence person society state

4 4 4 4

**Exercise**

What is the most frequent subject in the text? Make a frequency list of all subjects. Is your guess correct?

**Individual practice**

Create your own small corpus and parse it. Make a frequency list. Interpret the results.