**Introduction to Corpus Linguistics**

SoSe 2019

by Dr. Natalia Levshina ©

**Topic 6. N-grams**

The goals of our today’s session is to learn how to find n-grams in a corpus and to compute their frequencies. N-grams are sequences of n words (e.g. 2, 3 or 4). They are widely used in linguistics. For example, they serve for predicting the next word given the previous one, which is important in psycholinguistics (so-called transition probabilities). They are also important for identification of collocations (words that co-occur together more frequently than one can expect by chance, which is a topic of the next session). One can also use them to trace changes in language and culture over time, e.g. the Google Books Ngram Viewer: <https://books.google.com/ngrams>

**Data**

We’ll use a section from Donald Trump’s Twitter corpus. Please download the file trump\_big.txt.

**R practice**

1. Open the corpus in R. We’ll treat each new tweet as a separate unit:

**trump <- scan(file = file.choose(), what = "character", sep = "\n", quote = "", encoding = "UTF-8")**

**head(trump)**

[1] "RT @GOP: “Tomorrow the voters of this state will cast their ballots in one of the most important Senate elections of your lives—of all of…"

[2] "RT @GOP: “Only with a strong Senate GOP majority can we defend your tax cuts defend your Second Amendment protect your Medicare and Socia…"

[3] "RT @GOP: “I need the great people of Mississippi to send a message to… the radical Democrats by electing @cindyhydesmith.” -@realDonaldTrum…"

[output omitted]

2. Remove all units that are retweets and include the string “RT @”:

First, get the indices of those tweets.

**rt <- grep("RT @", trump)**

**head(rt)**

[1] 1 2 3 25 26 41

Next, remove all tweets with those indices:

**trump <- trump[-rt]**

**head(trump)**

[1] "....starts today election is on December 4th. @VoteBradRaff is tough on Crime and Borders Loves our Military and Vets. He will be great for jobs!"

[output omitted]

Now you see that the first retweets are gone.

3. Extract all tokens

First, we should split each of the tweets into tokens. The result is a list of 34927 elements (tweets), each with nested tokens. We will merge all of them with the help of function unlist(). As a result, we have 634080 tokens.

**trump\_tokens <- strsplit(trump, split = " ")**

**length(trump\_tokens)**

[1] 34927

**head(trump\_tokens)**

[[1]]

[1] "....starts" "today" "election" "is" "on" "December"

[7] "4th." "@VoteBradRaff" "is" "tough" "on" "Crime"

[output omitted]

**trump\_tokens <- unlist(trump\_tokens)**

**head(trump\_tokens)**

[1] "....starts" "today" "election" "is" "on" "December"

**length(trump\_tokens)**

[1] 634080

4. Clean up the data

As usual, we remove all punctuation, turn the capital letters into lowercase and remove all empty strings.

**trump\_tokens <- tolower(trump\_tokens)**

**trump\_tokens <- gsub("[[:punct:]]", "", trump\_tokens)**

**trump\_tokens <- trump\_tokens[nchar(trump\_tokens) > 0]**

**head(trump\_tokens)**

[1] "starts" "today" "election" "is" "on" "december"

5. Create bigrams and compute their frequencies

Bigrams are n-grams that consists of two words. We’ll need to extract the first elements (grams1), which will have all but one last word, and the second elements (grams2), which have all but one first word in the data. Next, we paste the 1st and 2nd elements and compute their frequencies:

**final <- length(trump\_tokens)**

**grams1 <- trump\_tokens[-final]**

**grams2 <- c(trump\_tokens[-1])**

**bigrams <- paste(grams1, grams2, sep = " ")**

**head(bigrams)**

[1] "starts today" "today election" "election is" "is on" "on december"

[6] "december 4th"

**bigrams\_freq <- table(bigrams)**

**sort(bigrams\_freq, decreasing = TRUE)[1:50]**

bigrams

thank you of the will be in the is a

2157 2140 1819 1575 1240

a great for the to the donald trump on the

1120 1034 940 907 880

to be for president i will is the at the

872 791 777

[output omitted]

What can you say about these bigrams?

6. Create trigrams and compute their frequencies

**penultimate <- final - 1**

**grams1 <- trump\_tokens[-c(penultimate, final)]**

**grams2 <- trump\_tokens[-c(1, final)]**

**grams3 <- c(trump\_tokens[-c(1, 2)])**

**trigrams <- paste(grams1, grams2, grams3, sep = " ")**

[1] "starts today election" "today election is" "election is on"

[4] "is on december" "on december 4th" "december 4th votebradraff"

**trigrams\_freq <- table(trigrams)**

**sort(trigrams\_freq, decreasing = TRUE)[1:50]**

trigrams

america great again make america great i will be

471 461 417

run for president one of the the united states

374 274 260

thank you for looking forward to the white house

236 210

[output omitted]

**Exercise**

Compute four-grams and interpret them.

**Exercise**

Take the data from Pottermore or another corpus of your choice and extract n-grams.