**Introduction to Corpus Linguistics**

WiSe 2018-2019

by Dr. Natalia Levshina ©

**Session 6. Distinctive slot fillers**

The goals of our today’s session is to learn how to find distinctive lexemes that perform a specific syntactic function in two different registers, text types or genres (I’ll use these notions as synonyms here).

An example: adverbial modifiers in Wikipedia and in news. Which adverbs would be more prominent in which text type?

For computing different measures of distinctiveness, we need the following frequencies for each lexeme:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Text Type 1** | **Text Type 2** | **Marginal row frequencies** |
| **Word 1** | a | b | a + b (the sum frequency of Word 1 in both text types) |
| **Not Word 1** | c | d | c + d (the sum frequency of all words other than Word 1 in both text types) |
| **Marginal column frequencies** | a + c  (the total frequency of all words in text type 1) | b + d (the total frequency of all words in text type 2) | N = a + b + c + d |

Note that this approach is very similar to **keyword analysis**, but here we compare the words that perform a specific syntactic function.

What does distinctiveness mean? A **distinctive** lexeme will more frequently occur in one register, and rarely in the other. This asymmetry tells us something about the communicative properties of the registers.

**Data**

We’ll use a corpus compiled from online Wikipedia articles, which is freely available from the Leipzig Corpora Collection, as well as a corpus from online news. The corpora have already been parsed by myself (in order to save time!). Please save them locally first.

The corpora are available from GitHub. Their names are eng\_wiki\_ud.txt and eng\_news\_ud.txt. We have already investigated the first one.

**R practice**

1. Open the corpora in R.

wiki <- read.table(file = file.choose(), header = T, sep = "\t", quote = "")

news <- read.table(file = file.choose(), header = T, sep = "\t", quote = "")

2. Extract all lemmas that play the role of adverbial modifiers.

adv\_wiki <- factor(wiki$lemma[wiki$dep\_rel == "advmod"])

adv\_news <- factor(news$lemma[news$dep\_rel == "advmod"])

3. Compute all frequencies of lemmas.

Again, we’ll use the familiar function table() to compute the frequencies. For technical reasons, we need to transform the numeric vector with frequencies into a data frame.

adv\_wiki\_freq <- table(adv\_wiki)

adv\_news\_freq <- table(adv\_news)

sort(adv\_news\_freq, decreasing = TRUE)[1:10]

adv\_news

not also so just more now how only even back

1175 369 287 278 247 230 202 189 180 163

sort(adv\_wiki\_freq, decreasing = TRUE)[1:10]

adv\_wiki

not also only however more often so then most even

858 587 282 250 229 182 175 170 165 148

adv\_wiki\_df <- data.frame(adv = names(adv\_wiki\_freq), freq\_wiki = as.numeric(adv\_wiki\_freq))

adv\_news\_df <- data.frame(adv = names(adv\_news\_freq), freq\_news = as.numeric(adv\_news\_freq))

The first impression is that the top most frequent adverbials are very similar.

4. Put together all frequencies a and b and compute the frequencies c and d:

adv\_both <- merge(adv\_wiki\_df, adv\_news\_df, by = "adv", all = TRUE)

adv\_both[is.na(adv\_both)] <- 0

adv\_both$freq\_wiki\_other <- sum(adv\_both$freq\_wiki) - adv\_both$freq\_wiki

adv\_both$freq\_news\_other <- sum(adv\_both$freq\_news) - adv\_both$freq\_news

5. Compute the expected frequencies for each lexeme.

It does not matter if we compute the expected frequency for one register or for the other. To compute the expected frequency in the Wikipedia corpus, we can use the following formula:

freq\_wiki\_exp = (a + c)\* (a + b)/(a + b + c + d) = total\_wiki\*(a + b)/total

where total\_wiki is the sum frequency of all adverbials, and total is the sum frequency of all adverbials in both corpora.

total\_wiki <- sum(adv\_both$freq\_wiki)

total\_wiki

#[1] 9242

total\_news <- sum(adv\_both$freq\_news)

total\_news

#[1] 9363

total <- total\_wiki + total\_news

total

#[1] 18605

adv\_both$freq\_wiki\_exp <- total\_wiki\*(adv\_both$freq\_wiki + adv\_both$freq\_news)/total

6. Compute PMI or any other measure of attraction (see previous class):

adv\_both$PMI <- log2(adv\_both$freq\_wiki/adv\_both$freq\_wiki\_exp)

7. Create a short version of the dataset only with the lexemes that occur more frequently than 5 times in both corpora taken together:

adv\_both\_short <- adv\_both[adv\_both$freq\_wiki + adv\_both$freq\_news > 5,]

8. Explore the top twenty (or more) lexemes.

adv\_both\_short[order(-adv\_both\_short$PMI), ][1:20,]

adv freq\_wiki freq\_news freq\_wiki\_other freq\_news\_other freq\_wiki\_exp PMI

545 notably 17 0 9225 9363 8.444719 1.0094134

379 i.e. 8 0 9234 9363 3.973985 1.0094134

402 independently 10 0 9232 9363 4.967482 1.0094134

573 onwards 7 0 9235 9363 3.477237 1.0094134

605 p. 6 0 9236 9363 2.980489 1.0094134

612 partly 11 0 9231 9363 5.464230 1.0094134

695 relative 10 0 9232 9363 4.967482 1.0094134

796 substantially 8 0 9234 9363 3.973985 1.0094134

794 subsequently 18 1 9224 9362 9.438216 0.9314109

922 widely 35 2 9207 9361 18.379683 0.9292430

151 commonly 37 3 9205 9360 19.869927 0.8969387

333 gradually 12 1 9230 9362 6.457726 0.8939362

714 roughly 12 1 9230 9362 6.457726 0.8939362

161 consequently 8 1 9234 9362 4.470734 0.8394884

223 e.g. 16 2 9226 9361 8.941467 0.8394884

744 sharply 8 1 9234 9362 4.470734 0.8394884

843 thus 75 10 9167 9353 42.223596 0.8288412

275 explicitly 7 1 9235 9362 3.973985 0.8167683

363 historically 14 2 9228 9361 7.947971 0.8167683

634 possibly 21 3 9221 9360 11.921956 0.8167683

adv\_both\_short[order(adv\_both\_short$PMI), ][1:20,]

adv freq\_wiki freq\_news freq\_wiki\_other freq\_news\_other freq\_wiki\_exp PMI

939 ’d 0 7 9242 9356 3.477237 -Inf

1125 hopefully 0 8 9242 9355 3.973985 -Inf

1152 kind 0 8 9242 9355 3.973985 -Inf

194 definitely 1 13 9241 9350 6.954475 -2.797942

240 else 1 12 9241 9351 6.457726 -2.691026

487 maybe 2 19 9240 9344 10.431712 -2.382904

646 pretty 3 28 9239 9335 15.399194 -2.359820

864 unanimously 1 9 9241 9354 4.967482 -2.312515

710 right 6 53 9236 9310 29.308143 -2.288267

51 allegedly 1 7 9241 9356 3.973985 -1.990587

687 really 21 115 9221 9248 67.557753 -1.685732

107 barely 2 10 9240 9353 5.960978 -1.575549

315 fourth 1 5 9241 9358 2.980489 -1.575549

461 little 1 5 9241 9358 2.980489 -1.575549

603 overseas 1 5 9241 9358 2.980489 -1.575549

313 forward 9 43 9233 9320 25.830906 -1.521101

72 anyway 3 14 9239 9349 8.444719 -1.493087

342 hardly 2 9 9240 9354 5.464230 -1.450018

45 ahead 4 17 9238 9346 10.431712 -1.382904

341 hard 4 17 9238 9346 10.431712 -1.382904

Note: The -Inf values are due to the fact that the log of 0 is Infinity. One can fix that by adding a small quantity to all frequencies.

How can you interpret the results?

**Exercises**

1. Compute another measure of association from the previous session. What are the differences?

2. Take another dependency or part of speech and compute the distinctive lexemes. Interpret the results.