Topic 5: Word Relationships EPA Reports on EJ

Halina Do-Linh

05/02/2022

Here I am creating an initial corpus, amending stop words, and converting the data into tidy format.

```r
# intial corpus
epa_corpus <- corpus(x = ej_pdf,
                     text_field = "text")
summary(epa_corpus)
```

```
## Corpus consisting of 6 documents, showing 6 documents:
##
##              Text Types Tokens Sentences type year docvar3
##  EPA_EJ_2015.pdf  2136   8944       263  EPA   EJ    2015
##  EPA_EJ_2016.pdf  1599   7965       176  EPA   EJ    2016
##  EPA_EJ_2017.pdf  3973  30564       653  EPA   EJ    2017
##  EPA_EJ_2018.pdf  2774  16658       447  EPA   EJ    2018
##  EPA_EJ_2019.pdf  3773  22648       672  EPA   EJ    2019
##  EPA_EJ_2020.pdf  4493  30523       987  EPA   EJ    2020
```

```r
# amending stop words
more_stops <-
  c("2015",
    "2016",
    "2017",
    "2018",
    "2019",
    "2020",
    "www.epa.gov",
    "https")
add_stops <- tibble(word = c(stop_words$word, more_stops))
# use stop vector with quanteda tools
stop_vec <- as_vector(add_stops)
```

```r
# tidy format
tidy_text <- tidy(epa_corpus)

# adding stop words
words <- tidy_text %>%
  mutate(year = as.factor(docvar3)) %>%
  unnest_tokens(word, text) %>%
  anti_join(add_stops,  by = 'word') %>%
  select(-docvar3)
```

Here I am creating data objects so I can do analysis.

```
# # most frequent words across all docs
# words_freq <- words %>%
#   count(year, word, sort = TRUE)
#
# # number of total words by doc per year
# words_total <- words_freq %>%
#   group_by(year) %>%
#   summarize(total = sum(n))
#
# # join words_freq and words_total
# words_report <- left_join(words_freq, words_total)
```

```
# quanteda word relationship tools
tokens <- tokens(epa_corpus,
                 remove_punct = TRUE)
tokens_1 <- tokens_select(tokens,
                          min_nchar = 3)
tokens_1 <- tokens_tolower(tokens_1)
tokens_1 <- tokens_remove(tokens_1,
                          pattern = (stop_vec))
# create document feature matrix
dfm <- dfm(tokens_1)

tstat_freq <- textstat_frequency(dfm, n = 5, groups = year)
head(tstat_freq, 10)
```

```
##         feature frequency rank docfreq group
## 1 environmental      1088    1       6    EJ
## 2   communities       940    2       6    EJ
## 3           epa       929    3       6    EJ
## 4     community       744    4       6    EJ
## 5       justice       606    5       6    EJ
```

1. What are the most frequent trigrams in the dataset? How does this compare to the most frequent bigrams? Which n-gram seems more informative here, and why?

```
# most freq trigrams
tokens_3 <- tokens_ngrams(tokens_1, n = 3)
dfm3 <- dfm(tokens_3)
dfm3 <- dfm_remove(dfm3, pattern = c(stop_vec))
freq_words3 <- textstat_frequency(dfm3, n = 20)
freq_words3$token <- rep("trigram", 20)

tstat_freq3 <- textstat_frequency(dfm3, n = 5, groups = year)
head(tstat_freq3, 10)
```

```
##                          feature frequency rank docfreq group
## 1        justice_fy2017_progress        51    1       1    EJ
## 2         fy2017_progress_report        51    1       1    EJ
## 3      environmental_public_health        50    3       6    EJ
## 4     environmental_justice_fy2017        50    3       1    EJ
## 5 national_environmental_justice        37    5       6    EJ
```

```r
# most freq bigrams
tokens_2 <- tokens_ngrams(tokens_1, n = 2)
dfm2 <- dfm(tokens_2)
dfm2 <- dfm_remove(dfm2, pattern = c(stop_vec))
freq_words2 <- textstat_frequency(dfm2, n = 20)
freq_words2$token <- rep("bigram", 20)

tstat_freq2 <- textstat_frequency(dfm2, n = 5, groups = year)
head(tstat_freq2, 10)
```

```
##                  feature frequency rank docfreq group
## 1 environmental_justice       556    1       6    EJ
## 2  technical_assistance       139    2       6    EJ
## 3        drinking_water       133    3       6    EJ
## 4         public_health       123    4       6    EJ
## 5       progress_report       108    5       6    EJ
```

Answer: The most frequent trigrams do not seem more informative than the most frequent bigrams. One of the top trigrams is `fy2017_progress_report` which is not informative at all. Because of this, I would say the bigrams are the more informative n-grams.

2. Choose a new focal term to replace "justice" and recreate the correlation table and network (see corr_paragraphs and corr_network chunks). Explore some of the plotting parameters in the cor_network chunk to see if you can improve the clarity or amount of information your plot conveys. Make sure to use a different color for the ties!

Here I am tokenizing the paragraphs from my tidy corpus, and then tokenizing the paragraphs by words.

```r
# tokenize by paragraphs
paragraph_tokens <- unnest_tokens(tidy_text,
                                  output = paragraphs,
                                  input = text,
                                  token = "paragraphs")
# give each paragraph an id
paragraph_tokens <- paragraph_tokens %>%
  mutate(par_id = 1:n())
# tokenize paragraphs by words
paragraph_words <- unnest_tokens(paragraph_tokens,
                                 output = word,
                                 input = paragraphs,
                                 token = "words")
```

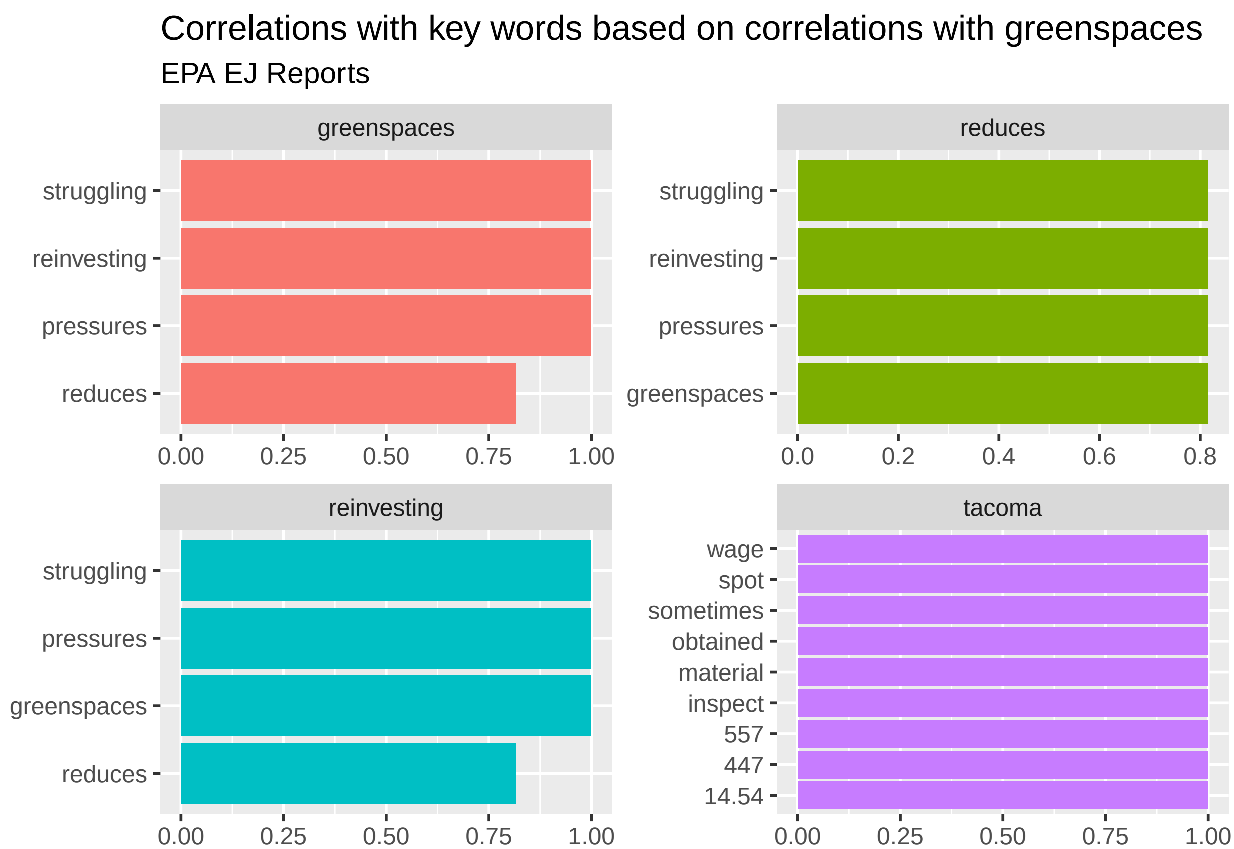
Here I am identifying which words tend to occur close together in the EPA reports, the word correlations, and chose "vegetation" as my focal word.

```r
# closely related pairs
word_pairs <- paragraph_words %>%
  pairwise_count(word, par_id, sort = TRUE, upper = FALSE) %>%
  anti_join(add_stops, by = c("item1" = "word")) %>%
  anti_join(add_stops, by = c("item2" = "word"))

# correlations
word_correlations <- paragraph_words %>%
```

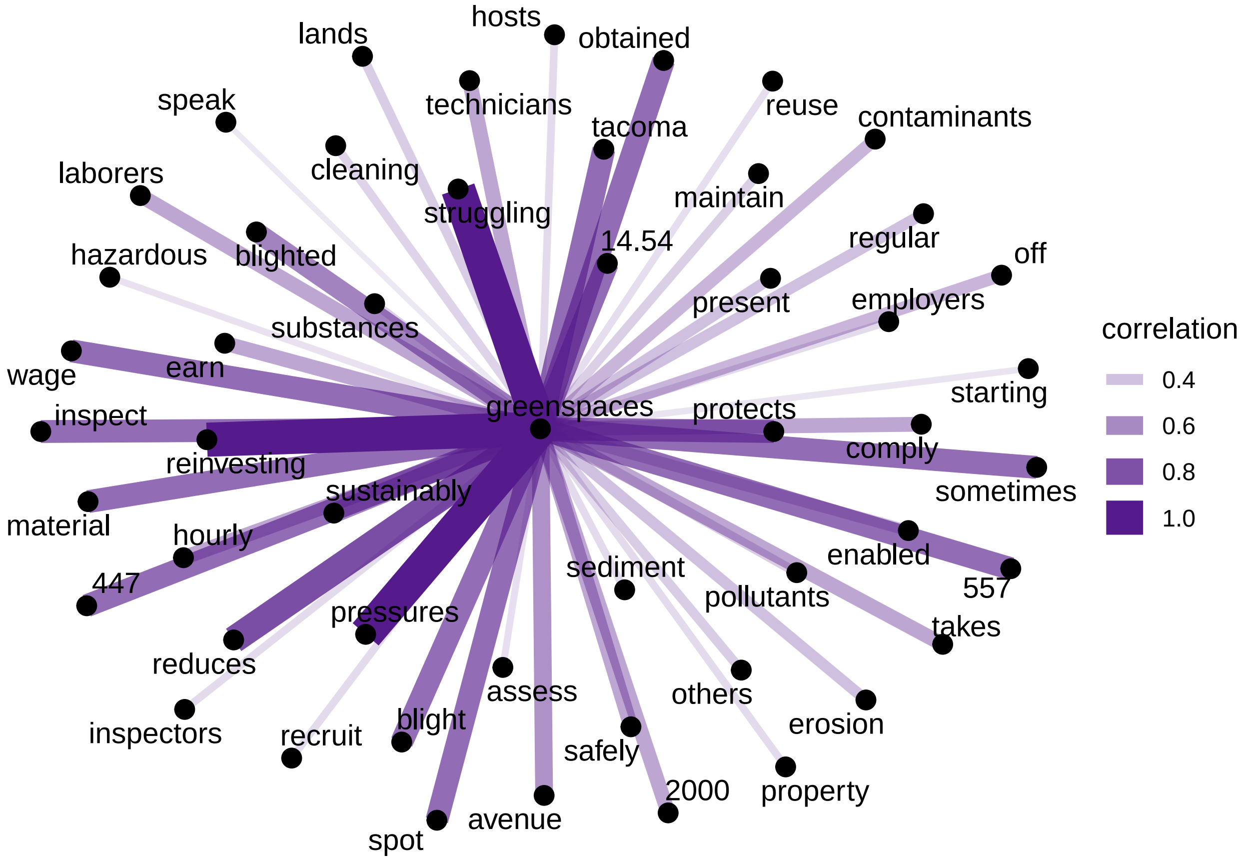
```r
  add_count(par_id) %>%
  filter(n >= 50) %>%
  select(-n) %>%
  pairwise_cor(word, par_id, sort = TRUE)

# focal word
greenspaces_correlations <- word_correlations %>%
  filter(item1 == "greenspaces") %>%
  mutate(n = 1:n())
```

Here I am recreating the correlation table and network.

```r
# correlations
word_correlations %>%
  filter(item1 %in% c("greenspaces",
                      "reinvesting",
                      "reduces",
                      "tacoma")) %>%
  group_by(item1) %>%
  top_n(4) %>% # top 4 words
  ungroup() %>%
  mutate(item1 = as.factor(item1),
         name = reorder_within(item2, correlation, item1)) %>%
  ggplot(aes(y = name, x = correlation, fill = item1)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~item1, ncol = 2, scales = "free")+
  scale_y_reordered() +
  labs(y = NULL,
         x = NULL,
         title = "Correlations with key words based on correlations with greenspaces",
         subtitle = "EPA EJ Reports")
```




```r
# network
greenspaces_correlations  %>%
  filter(n <= 50) %>%
  graph_from_data_frame() %>%
  ggraph(layout = "fr") +
  geom_edge_link(aes(edge_alpha = correlation, edge_width = correlation), edge_colour = "purple4",
                 size = 2) +
  geom_node_point(size = 3) +
  geom_node_text(aes(label = name), repel = TRUE,
                 point.padding = unit(0.2, "lines")) +
  theme_void()
```



3. Write a function that allows you to conduct a keyness analysis to compare two individual EPA reports (hint: that means target and reference need to both be individual reports). Run the function on 3 pairs of reports, generating 3 keyness plots.

```
# keyness <- function(){
#
#
# }
```

4. Select a word or multi-word term of interest and identify words related to it using windowing and keyness comparison. To do this you will create to objects: one containing all words occurring within a 10-word window of your term of interest, and the second object containing all other words. Then run a keyness comparison on these objects. Which one is the target, and which the reference? Hint