Regulatory documents via LDA

Sebastian Knigge

18 8 2019

1 Setup

Following libraries are used in the code:

```
library(dplyr)
library(tidytext)
library(pdftools)
library(tidyr)
library(stringr)
library(tidytext)
library(udpipe)
library(topicmodels)
library(ggplot2)
library(wordcloud)
library(tm)
library(SnowballC)
library(RColorBrewer)
library(RCurl)
library(XML)
```

2 Import data

In this code reulatory documents are red in and processed via LDA. This first part focusses on reading in the pdf documents.

```
# getting the right directiory
library(here)
setwd("../")
path <- getwd() %>%
  file.path("TextDocs")
documents <- list.files(path)
```

Following functions are used to set up and analyze the pdfs.

```
read_pdf_clean <- function(document){
  # This function loads the document given per name
  # and excludes the stop words inclusive numbers
  pdf1 <- pdf_text(file.path(path, document)) %>%
    strsplit(split = "\n") %>%
    do.call("c",.) %>%
    as_tibble() %>%
    unnest_tokens(word,value) %>%
    # apply a filter for ^
    filter(!grepl("^",word))
  # load stopword library
  data(stop_words)
```

```
  # add own words to stop word library - here the numbers from 1 to 10
  new_stop_words <- tibble(word=as.character(0:9),
                           lexicon=rep("own",10)) %>%
    bind_rows(stop_words)

  pdf1 %>%
    anti_join(new_stop_words)
}

plot_most_freq_words <- function(pdf, n=5){
  # plots a bar plot via ggplot
  pdf %>% count(word) %>% arrange(desc(n)) %>% head(n) %>%
    ggplot(aes(x=word,y=n)) +
    geom_bar(stat="identity")+
    # no labels for x and y scale
    theme(axis.title.y=element_blank(),
          axis.title.x=element_blank())
}
```

Now we can read in all documents in a for loop:

```
# inital set up for the corpus
pdf1 <- read_pdf_clean(documents[1])
corpus <- tibble(document=1, word=pdf1$word)
# adding the documents iteratively
for (i in 2:length(documents)){
  pdf_i <- read_pdf_clean(documents[i])
  corpus <- tibble(document=i, word=pdf_i$word) %>% bind_rows(corpus,.)
}
```

3 LDA

The LDA model is applied. First the document term matrix has to be set up.

```
dtm <- corpus %>% count(document, word, sort = TRUE) %>%
  select(doc_id=document, term=word, freq=n) %>%
  document_term_matrix()
dim(dtm)
```

```
## [1]    28 12992
```

Using the function LDA sets up the model and prediction/evaluation is done via predict(). But first of all it shall be verified whether the Predict function actually delivers the same classification as the export of the gamma matrix directly from the LDA model. Therefore both gamma matrices of the single functions are compared. Table 1 displays the output of the gamma matrix received by the predict() function and Table 2 displays the gamma matrix returned by the LDA model itself.

```
set.seed(123)
documents_lda <- LDA(dtm,
                     k = 5, control = list(seed = 1234))
```

```
prediction5 <- predict(documents_lda, newdata=dtm, type="topic")

prediction5 %>%
  select(doc_id,topic_001,topic_002,topic_003,topic_004,topic_005) %>%
```

```r
  mutate_each(funs(as.numeric), doc_id,topic_001,topic_002,topic_003,topic_004,topic_005) %>%
  arrange(desc(-doc_id)) %>%
  round(2) %>%
  stargazer(summary=F, rownames = F, header = F, title="Gamma matrix for predict function", label="predi
```

Table 1: Gamma matrix for predict function

doc_id	topic_001	topic_002	topic_003	topic_004	topic_005
1	0	1	0	0	0
2	0	0.930	0	0.070	0
3	0.040	0	0.240	0.710	0.010
4	0	0	0.280	0.720	0
5	0	0	1	0	0
6	0.310	0.010	0.680	0	0
7	0	0	1	0	0
8	0	0	1	0	0
9	0	0	1	0	0
10	0	0	1	0	0
11	0.040	0	0.960	0	0
12	0	0	1	0	0
13	0	0.060	0	0.940	0
14	0	0	0.290	0.710	0
15	0	0	0.460	0.540	0
16	0.980	0	0	0	0.020
17	1	0	0	0	0
18	1	0	0	0	0
19	1	0	0	0	0
20	1	0	0	0	0
21	0	0	0	1	0
22	0.990	0	0	0.010	0
23	0	0	0	0	1
24	0.990	0	0	0.010	0
25	0.800	0	0.020	0.160	0.010
26	0	0.170	0	0	0.830
27	0	0.010	0	0.890	0.090
28	0	0.010	0	0.090	0.910

```r
ext_gamma_matrix <- function(model=documents_lda){
  # get gamma matrix for chapter probabilities
  chapters_gamma <- tidy(model, matrix = "gamma")
  # get matrix with probabilities for each topic per chapter
  spreaded_gamma <- chapters_gamma %>% spread(topic, gamma)
  spreaded_gamma %>%
    mutate_each(funs(as.numeric), document,1,2,3,4,5) %>%
  arrange(desc(-document))
}

ext_gamma_matrix(documents_lda) %>%
  round(2) %>%
  stargazer(summary=F, rownames = F, header=F, title="Gamma matrix extracted from model", label="extract
```

The tables below summarize which document refers to which topic, according to the LDA model.

Table 2: Gamma matrix extracted from model

document	1	2	3	4	5
1	0	1	0	0	0
2	0	0.93	0	0.07	0
3	0.04	0	0.24	0.71	0.01
4	0	0	0.28	0.72	0
5	0	0	1	0	0
6	0.31	0.01	0.68	0	0
7	0	0	1	0	0
8	0	0	1	0	0
9	0	0	1	0	0
10	0	0	1	0	0
11	0.04	0	0.96	0	0
12	0	0	1	0	0
13	0	0.06	0	0.94	0
14	0	0	0.29	0.71	0
15	0	0	0.46	0.54	0
16	0.98	0	0	0	0.02
17	1	0	0	0	0
18	1	0	0	0	0
19	1	0	0	0	0
20	1	0	0	0	0
21	0	0	0	1	0
22	0.99	0	0	0.01	0
23	0	0	0	0	1
24	0.99	0	0	0.01	0
25	0.8	0	0.02	0.16	0.01
26	0	0.17	0	0	0.83
27	0	0.01	0	0.89	0.09
28	0	0.01	0	0.09	0.91

Table 3: Documents for Topic 1

Group	Doc
1	5
1	6
1	10
1	16
1	21
1	23
1	27
1	28

Table 4: Documents for Topic 2

Group	Doc
2	1
2	4

Table 5: Documents for Topic 3

Group	Doc
3	12
3	13
3	15
3	19
3	20
3	22
3	24
3	26

Table 6: Documents for Topic 4

Group	Doc
4	3
4	7
4	9
4	11
4	14
4	17
4	25

Table 7: Documents for Topic 5

Group	Doc
5	2
5	8
5	18

4 Wordclouds

To check what topics tackle which context, we produce wordclouds using the TFIDF and the TF itself.

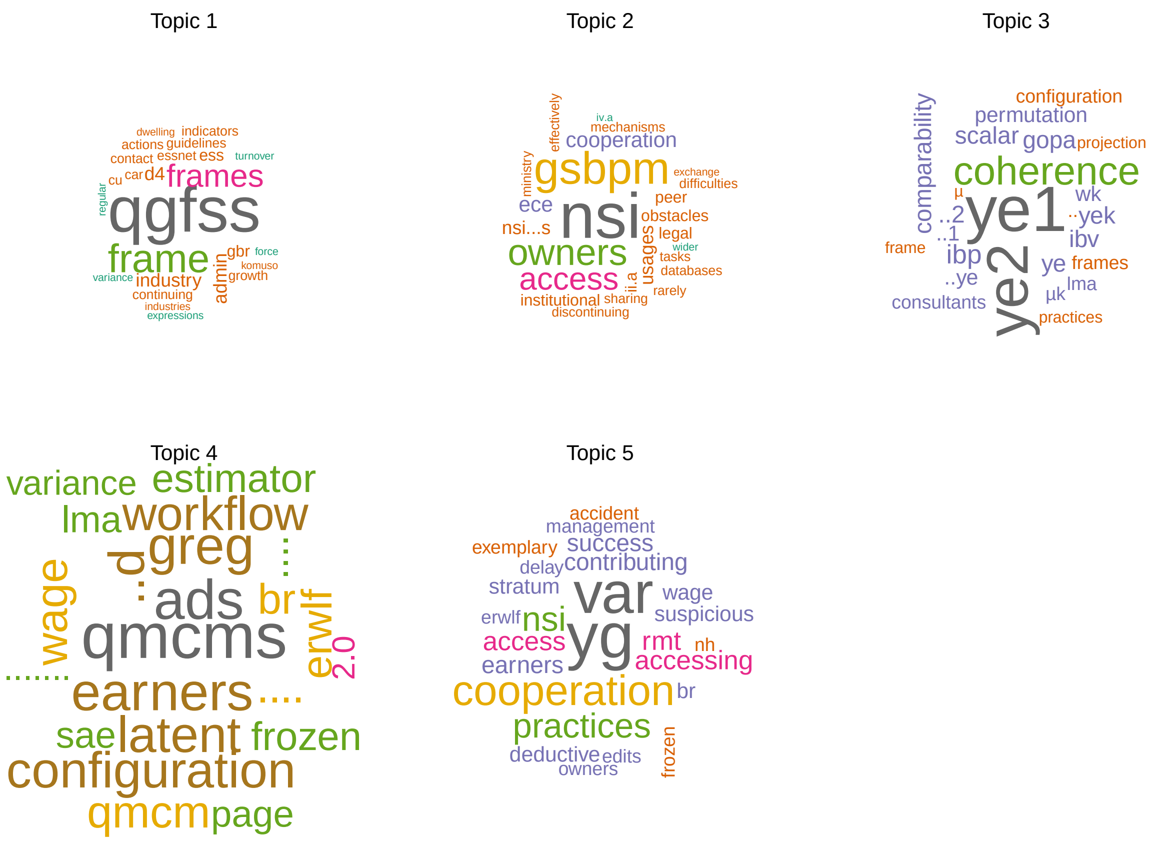
```
plot_wordcloud <- function(corpus, selection="ALL", max.words=25, i, freq="tfidf"){
  # setting up a tibble which returns tfidf and tf and frequency for
  # the whole corpus
  tfidf <- corpus %>% count(document, word, sort = TRUE) %>%
    bind_tf_idf(word, document, n)
  # include all documents for selection if selection="ALL"
  if (all(selection=="ALL")) {
    selection <- corpus %>%
      select(document) %>%
      unique() %>%
      unlist() %>%
      sort()}
  # filter for all selected documents
  # use either ft or tfidf
  if (freq=="tfidf"){
    dtm_selected <- tfidf %>% filter(document%in%selection) %>%
      select(word, tf_idf) %>% count(word, wt=tf_idf, sort=TRUE)
  } else {
    dtm_selected <- tfidf %>% filter(document%in%selection) %>%
      select(word, tf) %>% count(word, wt=tf, sort=TRUE)
  }
  wordcloud(words = dtm_selected$word, freq = dtm_selected$n, min.freq = 1,
            max.words=max.words, random.order=FALSE,
            colors=brewer.pal(8, "Dark2"), scale=c(3,0.2),
            main="Title", use.r.layout = TRUE)
  text(x=0.5, y=1, paste("Topic", i))
}
```

For getting specific and more individual words for each cloud, we use the TFIDF in the first step.

```
# compare topic 1 with topic 2, 3, 4 and 5
ind1 <- which(prediction5$topic==1)
ind2 <- which(prediction5$topic==2)
ind3 <- which(prediction5$topic==3)
ind4 <- which(prediction5$topic==4)
ind5 <- which(prediction5$topic==5)
```

4.1 Wordclouds using tfidf

```
par(mfrow=c(2,3))
par(mar=c(1,1,0.5,1))
plot_wordcloud(corpus, selection=ind1, i=1)
plot_wordcloud(corpus, selection=ind2, i=2)
plot_wordcloud(corpus, selection=ind3, i=3)
plot_wordcloud(corpus, selection=ind4, i=4)
plot_wordcloud(corpus, selection=ind5, i=5)
```



4.2 Wordclouds using tf

The same can be done using the regular term frequency.

```
par(mfrow=c(2,3))
par(mar=c(1,1,0.5,1))
plot_wordcloud(corpus, selection=ind1, i=1, freq="tf")
plot_wordcloud(corpus, selection=ind2, i=2, freq="tf")
plot_wordcloud(corpus, selection=ind3, i=3, freq="tf")
plot_wordcloud(corpus, selection=ind4, i=4, freq="tf")
plot_wordcloud(corpus, selection=ind5, i=5, freq="tf")
```

5 Embedding via tfidf

Now it's interesting to see if embedding with tfidf will cluster other groups or the same. So we will reduce the Document Term Matrix to 10000 words which is a reduction by approx. 20%.

```
dtm_50 <- dtm %>% dtm_remove_tfidf(top=10000)
set.seed(123)
documents_lda_2 <- LDA(dtm_50,
                     k = 5, control = list(seed = 1234))
```

```
prediction5_2 <- predict(documents_lda_2, newdata=dtm_50, type="topic")
# compare topic 1 with topic 2, 3, 4 and 5
ind1_2 <- which(prediction5_2$topic==1)
ind2_2 <- which(prediction5_2$topic==2)
ind3_2 <- which(prediction5_2$topic==3)
ind4_2 <- which(prediction5_2$topic==4)
ind5_2 <- which(prediction5_2$topic==5)
```

```
ext_gamma_matrix(documents_lda_2) %>%
  round(2) %>%
  stargazer(summary=F, rownames = F, header=F, title="Gamma matrix extracted from model for embedding wi
```

5.1 Wordclouds

```
par(mfrow=c(2,3))
par(mar=c(1,1,0.5,1))
plot_wordcloud(corpus, selection=ind1_2, i=1)
```

Table 8: Documents for Topic 1

Group	Doc_embedding_0.5
1	5
1	6
1	10
1	12
1	16
1	22
1	24
1	26

Table 9: Documents for Topic 2

Group	Doc_embedding_0.5
2	8
2	21
2	23
2	27
2	28

Table 10: Documents for Topic 3

Group	Doc_embedding_0.5
3	1
3	4

Table 11: Documents for Topic 4

Group	Doc_embedding_0.5
4	2
4	7
4	14
4	18
4	25

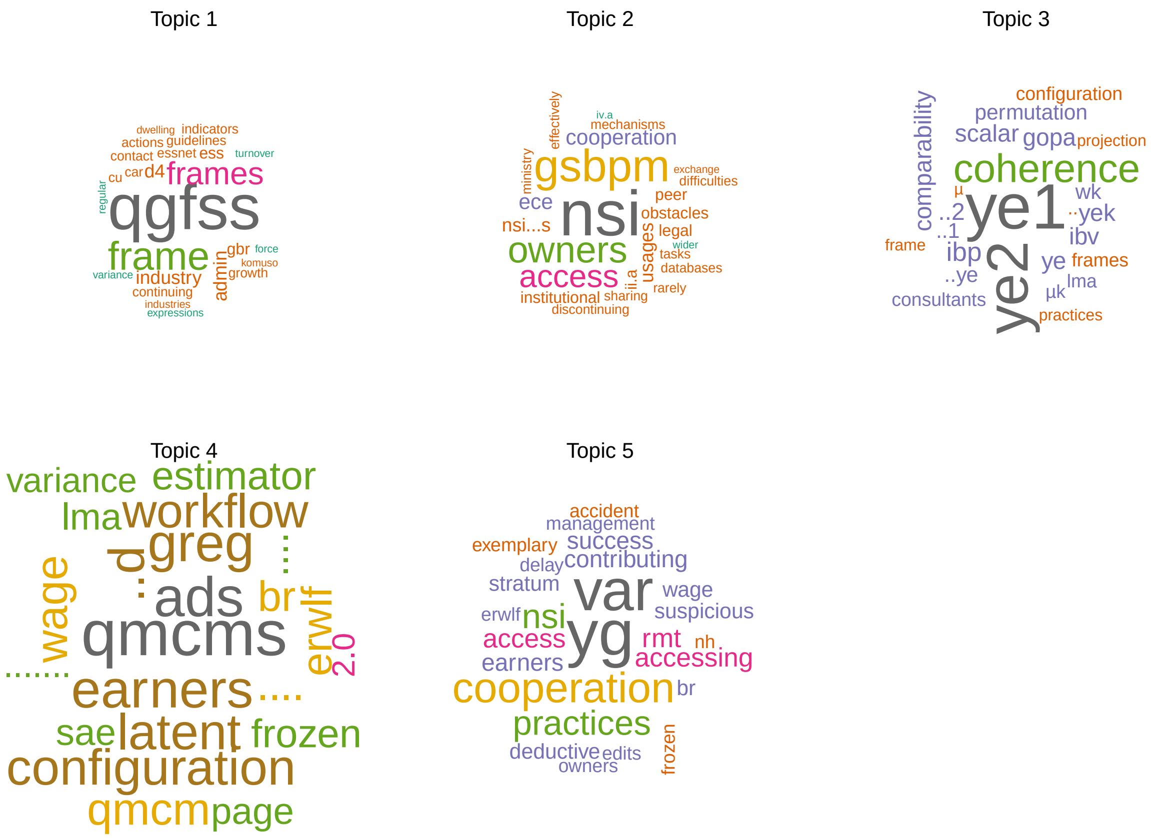
Table 12: Documents for Topic 5

Group	Doc_embedding_0.5
5	3
5	9
5	11
5	13
5	15
5	17
5	19
5	20

Table 13: Gamma matrix extracted from model for embedding with tfidf

document	1	2	3	4	5
1	0	0	1	0	0
2	0	0	0.95	0.05	0
3	0.09	0	0	0.08	0.83
4	0	0	0	0.02	0.98
5	0.61	0	0	0	0.39
6	0.16	0	0	0	0.84
7	0.74	0	0	0	0.26
8	0.4	0	0	0	0.6
9	0.5	0	0	0	0.49
10	0.44	0	0	0	0.56
11	0.55	0	0	0	0.45
12	0.4	0	0	0	0.6
13	0	0	0.18	0.26	0.55
14	0	0	0	0.01	0.99
15	0.07	0	0	0.93	0
16	1	0	0	0	0
17	0	1	0	0	0
18	0	1	0	0	0
19	0	1	0	0	0
20	0	1	0	0	0
21	0	0	0	0.97	0.03
22	0.99	0	0	0.01	0
23	0	0.89	0	0.11	0
24	0.99	0	0	0.01	0
25	0.86	0	0	0.14	0
26	0	0	0.12	0.88	0
27	0	0	0	1	0
28	0	0.05	0	0.95	0.01

```r
plot_wordcloud(corpus, selection=ind2_2, i=2)
plot_wordcloud(corpus, selection=ind3_2, i=3)
plot_wordcloud(corpus, selection=ind4_2, i=4)
plot_wordcloud(corpus, selection=ind5_2, i=5)
```



6 Explorative Analysis for ANN

```r
# number of words
l <- 10

tf_idf_matrix <- corpus %>%
  count(document, word, sort = TRUE) %>%
  bind_tf_idf(word, document, n)
# return the l most frequent words per document
freq_words <- lapply(1:28, function(x) tf_idf_matrix %>% filter(document==x) %>% arrange(desc(tf_idf)) %
                       .[1:l,2]) %>%
                        do.call("cbind",.)

colnames(freq_words) <- paste0("doc_",1:28)
```