2017 UN General Debate: Topic Modelling

Jeniffer V. Scurrel, Thomas Asikis

11/18/2019

#Introduction In this code tutorial we are going to show how topic modelling can be done on political texts using Latent Dirichlet Allocation. Since we already showcased how to load data from dataverse or locally, now we are going to use a `quanteda` corpus from the 2017 UN General Debate. The goal of the current study is twofold: (i) determine the topics that were mostly discussed during the debate and (ii) set the foundations for creating and expert dictionary.

Data loading

`quanteda` is a widely used library for natural language processing tasks. It doesn't have a rich collection of corpora as dataverse but it can be used to process local corpora as well. First we load the data_corpus_ungd2017 from \texttt{data_corpus_ungd201}.

```
#devtools::install_github("quanteda/quanteda.corpora")
library(quanteda.corpora)
library(quanteda)
```

```
## Package version: 1.5.1
```

```
## Parallel computing: 2 of 8 threads used.
```

```
## See https://quanteda.io for tutorials and examples.
```

```
##
## Attaching package: 'quanteda'
```

```
## The following object is masked from 'package:utils':
##
##     View
```

```
data_corpus_ungd2017
```

```
## Corpus consisting of 196 documents and 7 docvars.
```

```
data_corpus_ungd2017_wd <- data_corpus_ungd2017[1:20,]
```

Now we construct a document-feature matrix, i.e. a matrix with each document as row index and the extracted term after preprocessing for each words. Each element of the matrix represents the number of appearances of the preprocessed terms in each document. In our current example the words are normalized to lower case and stemmed. Punctuation, numbers and english stopwords are removed. Finally, we use only single words as the elements of our model.

```
ungd2017_dfm <- quanteda::dfm(data_corpus_ungd2017_wd,
              tolower=TRUE,
              stem=TRUE,
              remove_punct = TRUE,
              # removeNumbers = TRUE, # laptop r version is older and may not support this
              remove = stopwords("english"),
              ngrams=1,
              verbose=TRUE)
```

```
## Creating a dfm from a character input...
##    ... lowercasing
##    ... found 20 documents, 5,036 features
##    ... removed 116 features
##    ... stemming features (English)
## , trimmed 1564 feature variants
##    ... created a 20 x 3,356 sparse dfm
##    ... complete.
## Elapsed time: 0.168 seconds.
```

Now it is not possible to load the quanteda matrix in the `topicmodels` package methods. Therefore we convert it to the appropriate object structure. We refer to it as document term matrix (dtm) to distringuish the input for two packages.

```
dtm <- quanteda::convert(ungd2017_dfm, to = "topicmodels")
```

LDA Model

Let's now provide our dta to an lda model.

Choosing the number of topics

LDA high-level parameters are pretty straight forward. The most import one is the number of topics. Deciding the number of topics arbitarily may lead to problems, such as having a high number of topics may reduce interpretability. The best way to pick which and how many topics need to be used is to check them and pick them manually after running the process for a different number of topics. Several evaluation metrics have been developed to determine a good number of topics automatically. Below we do a parameter search for $1, 2, ..10$ topics. We use Gibbs sampling to determine the prior of the LDA, which usually leads to higher performance.

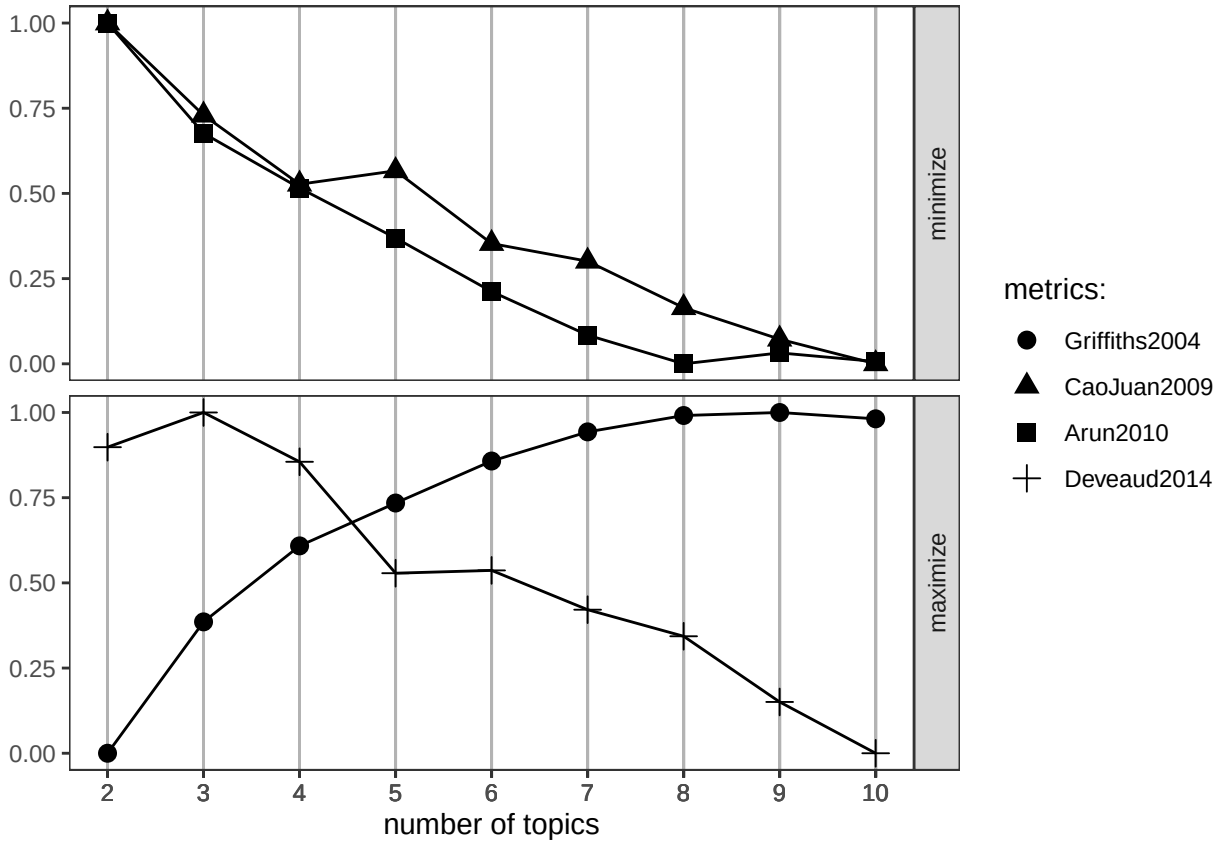
```
#install.packages("ldatuning")
library(ldatuning)

result <- FindTopicsNumber(
  dtm,
  topics = seq(from = 2, to = 10, by = 1),
  metrics = c("Griffiths2004", "CaoJuan2009", "Arun2010", "Deveaud2014"),
  method = "Gibbs",
  control = list(seed = 77),
  mc.cores = 4L,
  verbose = TRUE
)
```

```
## fit models... done.
## calculate metrics:
##   Griffiths2004... done.
##   CaoJuan2009... done.
##   Arun2010... done.
##   Deveaud2014... done.
```

```
knitr::kable(result)
```

topics	Griffiths2004	CaoJuan2009	Arun2010	
10	-141556.5	0.1219636	31.64286	
9	-141338.0	0.1343928	32.10966	
8	-141441.9	0.1502939	31.52026	
7	-142005.9	0.1737602	33.06719	
6	-143009.0	0.1826968	35.41838	
5	-144454.6	0.2194423	38.30485	
4	-145929.6	0.2126502	41.00609	
3	-148545.5	0.2475362	43.95796	
2	-153067.2	0.2940568	49.90729	
As litera	ture suggests (y	ou can also ch	eck the pac	kage vignette), we check the plot below and pick a number
At the sa	me time we try t	o pick the num	ber of topi	
The selec	tion might not a	lways be clear	and we mig	
In our ca	se a high number	of topics (e.	g. 9) seems	

```
FindTopicsNumber_plot(result)
```



Fitting the final LDA Now that number of topics is known, we can fit the final LDA model. Again, as the model is probabilistic we define a seed so that we get the same results across runs. The more data we have and the more efficient our sampling is, the less the randomness in genreal.

```
library(topicmodels)
ungd2017_lda <- LDA(dtm, k = 9, control = list(seed = 1234))
ungd2017_lda
```

```
## A LDA_VEM topic model with 9 topics.
```

Now that we have our topics, let's check what words are inside each topic: These will be the terms that are

assigned the higher β, also reffered to as "per-topic-per-word probabilities", value per topic, and we would like to plot the top 10.

```
library(tidytext)
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

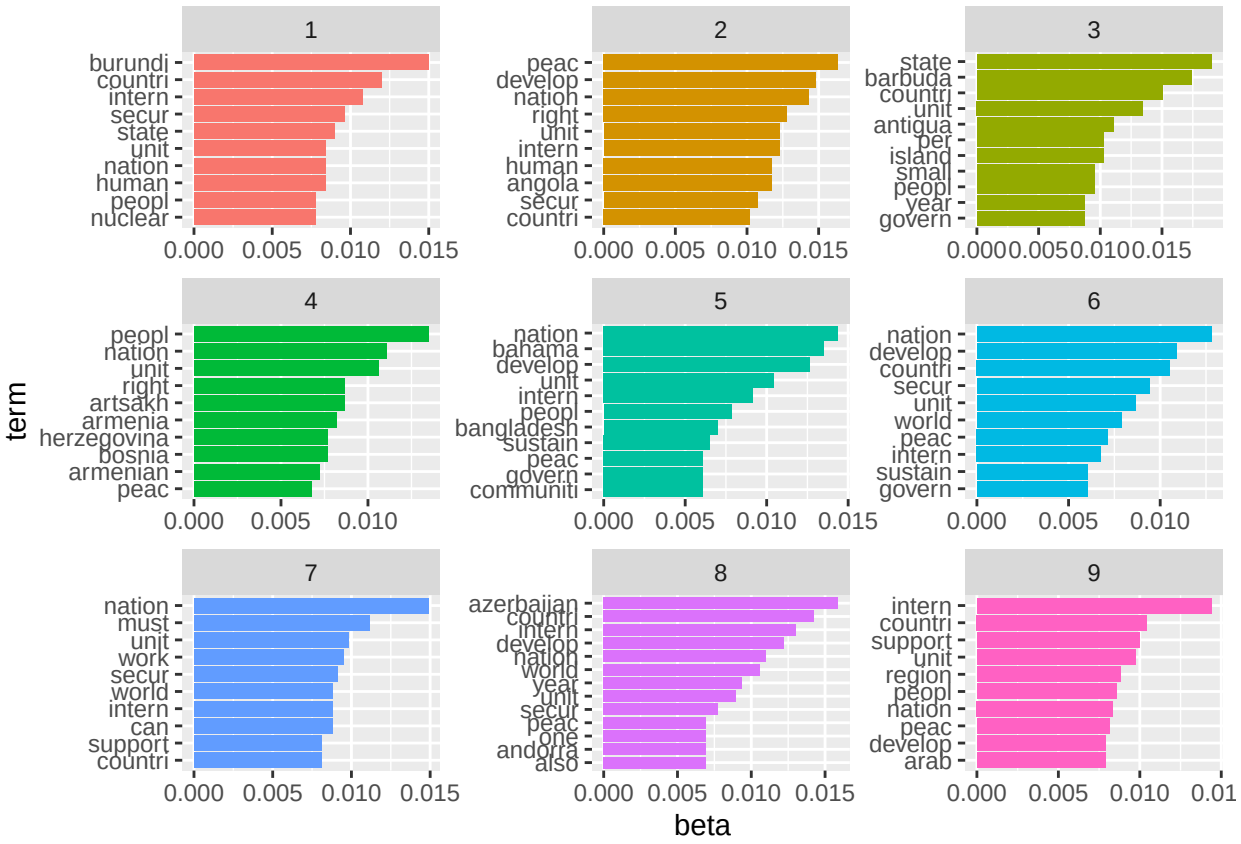
```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
ungd2017_topics <- tidy(ungd2017_lda, matrix = "beta")
ungd2017_top_terms <- ungd2017_topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)
```

We can use the top terms to plot them in a descending order according to beta:

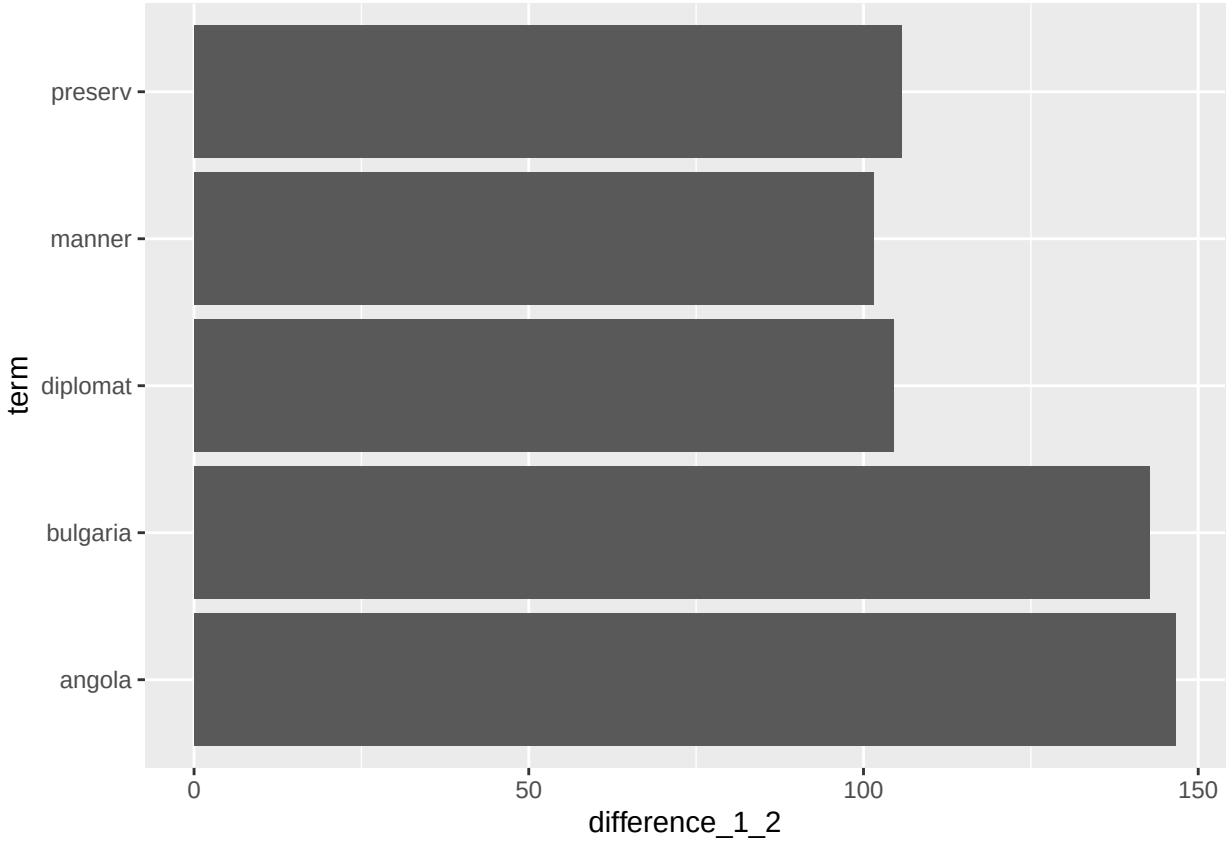
```
ungd2017_top_terms %>%
  mutate(term = reorder_within(term, beta, topic)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip() +
  scale_x_reordered()
```



Furthermore, we can see that the beta value per term and topic. This could be used as some kind of term representation or embedding. Still, there may be more optimal ways to produce LDA based embeddings...

```
library(tidyr)

beta_spread <- ungd2017_topics %>%
  mutate(topic = paste0("topic", topic)) %>%
  spread(topic, beta)

whichpart <- function(x, n=30) {
  nx <- length(x)
  p <- nx-n
  xp <- sort(x, partial=p)[p]
  which(x > xp)
}


diff <-  log(beta_spread$topic1)/log(beta_spread$topic2)
abs_diff <- abs(diff)
beta_spread$difference_1_2 <- diff
largest_differences_ind <- whichpart(diff, n=5)
largest_differences <- beta_spread[largest_differences_ind, c('term', 'topic1', 'topic2', 'difference_1_

#library(plotly)
# a plotly equivalent
#p <- plot_ly(x =largest_differences$difference_1_2, y = largest_differences$term, type = 'bar', orient
#plotly::export(p)
```

```
p <- ggplot(largest_differences, aes(y=difference_1_2, x=term))+
  geom_col()
p + coord_flip()
```



Now we coud also chech the γ value, whici refers to "per-document-per-topic probabilities". We can use these probabilities and a thershold (assume $\gamma > 0.001$) to derive topic similarities between documents.

```
ungd2017_documents <- tidy(ungd2017_lda, matrix = "gamma")
most_probable_topics <- ungd2017_documents[ungd2017_documents$gamma> 0.001, ]
knitr::kable(t(most_probable_topics))
```

document	Austria	Burundi	Angola	Bulgaria	Antigua & Barbuda	Armenia	Bosnia & Herzegovina	B
topic	1	1	2	2	3	4	4	5
gamma	0.9998747	0.9999084	0.9999101	0.9999093	0.9999301	0.9998999	0.9999262	0

```
#t1: Burundi particiapted in the negotiations of the UN treaty on the Prohibition of Nucelar Weapons
# and voted in favour on in July 2017

#2: https://gadebate.un.org/en/72/angola

#3: GASTON ALPHONSO BROWNE, Prime Minister and Minister for Finance and
#Corporate Governance of Antigua and Barbuda,
#recalled how on 6 September, his two-island State was victim to the ferocity of Hurricane Irma.
```