LDA Documentation

Sebastian Knigge

6 8 2019

Initial Setup

This is the essential step for setting up the LDA model. These functions include the sampling procedure from the *gutenbergr* library

```
# loading packages
library(gutenbergr)
library(dplyr)
library(tidyr)
library(stringr)
library(tidytext)
library(udpipe)
library(topicmodels)
library(ggplot2)

sampling_books <- function(seed=1234, n=20){
  # sample n books from the whole library
  set.seed(seed)
  gutenberg_works() %>%
    # select works with title
    dplyr::filter(!is.na(title)) %>%
    # set the sample sitze
    sample_n(n) %>%
    # set a special download link
    gutenberg_download(
      mirror = "http://mirrors.xmission.com/gutenberg/")
}
```

- good seperation for 4 topics:
  - seed=12345
  - seed=54321
- for 6 books:
  - seed=222
  - seed 101
- for 10 books:
  - seed=54321
  - seed=123456

```
set_up_books <- function(n_books=4, seed=1992){
  # initial book sample
  books <- sampling_books(n=n_books, seed=seed)
  by_chapter <- books %>%
    group_by(gutenberg_id) %>%
    # split in chapters
    mutate(chapter = cumsum(str_detect(text, regex("^chapter ", ignore_case = TRUE)))) %>%
    ungroup() %>%
    # exclude books without chapters
```

```
    dplyr::filter(chapter > 0)
  return(by_chapter)
}

shorten_titles <- function(titles){
  # shorten very long book titles by setting
  # a subset of characters of the first line
  # of the title
  sub_inds <- titles %>%
    regexpr(pattern="\\n|\\r")-1
  sub_inds[sub_inds<0] <- nchar(titles)[sub_inds<0]
  titles %>%
    substr(1,sub_inds)
}

get_titles <- function(x, n_books){
  # get the sampled gutenberg_ids
  unique_ids <- x %>%
    select(gutenberg_id) %>%
    unique() %>% unlist()
  # get the titles
  titles <- gutenberg_works() %>%
    dplyr::filter(gutenberg_id %in% unique_ids) %>%
    select(gutenberg_id, title) %>%
    mutate(title=shorten_titles(title))
  # get the number of gutenberg ids
  len <- nrow(titles)
  if(n_books!=len) warning(paste("--- ",n_books-len,
                                 " books have 0 chapters --- "))
  # the output as a list
  ret <- list(
    titles=titles,
    len=len
  )
  return(ret)
}

append_by_chapter <- function(x=by_chapter, n_books, seed_index=1){
  # append the books matrix until
  # we get the desired number of books n_books

  titles <- get_titles(x, n_books)
  n <- titles$len
  while (n<n_books) {
    book2add <- sampling_books(n=1, seed=seed_index)
    by_chapter_add <- book2add %>%
      group_by(gutenberg_id) %>%
      # split in chapters
      mutate(chapter = cumsum(str_detect(text, regex("^chapter ", ignore_case = TRUE)))) %>%
      ungroup() %>%
      # exclude books without chapters
      dplyr::filter(chapter > 0)
    titles2add <- get_titles(by_chapter_add, 1)
```

```
    # adding the book to by_chapter if there are chapters in the
    # book plus it is not in the data already
    if (titles2add$len==1) if(!titles2add$titles$gutenberg_id%in%titles$titles$gutenberg_id) {
      x <- bind_rows(x, by_chapter_add)
    }
    n<-get_titles(x, n)$len
    seed_index <- seed_index+1
  }
  return(x)
}

exclude_stop_words <- function(x){
  # unite chapter and document title
  by_chapter_word <- x %>%
    unite(document, gutenberg_id, chapter) %>%
    # split into words
    unnest_tokens(word, text)
  # import tibble stop words
  data(stop_words)
  # find document-word counts
  word_counts <- by_chapter_word %>%
    # exclude stop words
    anti_join(stop_words) %>%
    # count each word by chapter
    count(document, word, sort = TRUE) %>%
    ungroup()
  return(word_counts)
}

convert_to_dtm <- function(x, minfq=2){
  # get into a format lda can handle
  chapters_dtm <- x %>%
    select(doc_id=document, term=word, freq=n) %>%
    document_term_matrix() %>%
    # reduce by low frequencies
    dtm_remove_lowfreq(minfreq = minfq)
  return(chapters_dtm)
}

convert_to_dtm_2 <- function(x, n=n, minfq = 2, top=10000){
  # get into a format lda can handle
  chapters_dtm <- x %>%
    select(doc_id=document, term=word, freq=n) %>%
    document_term_matrix() %>%
    # reduce by low frequencies
    dtm_remove_tfidf(top=top)
  return(chapters_dtm)
}
```

Now we can use all these functions to get to the initial corpus sample.

```
n_books <- 6
by_chapter <- set_up_books(n_books=n_books, seed=222)
appended_by_chapter <- append_by_chapter(x=by_chapter, n_books = n_books)
```

```
word_counts <- exclude_stop_words(appended_by_chapter)
```

```
## Joining, by = "word"
```

These are the sampled titles for the book sample with the seed 222.

```
titles <- get_titles(appended_by_chapter, n_books)
titles$titles %>% stargazer(summary=FALSE, font.size = "footnotesize", header=FALSE, title="Book-titles
```

Table 1: Book-titles

gutenberg_id	title
11	Alice's Adventures in Wonderland
3096	Beatrice
7705	"My Novel" — Volume 04
25603	Detailed Minutiae of Soldier life in the Army of Northern Virginia, 1861-1865
47402	Along Alaska's Great River
49675	Hawkins Electrical Guide v. 5 (of 10)

In the set up we have another parameter to adjust. The minimum frequency for the bag of words dictionary. Let us set it to 2 in this case, meaning that we include a word only if the frequency is 2 or more.

```
chapters_dtm <- convert_to_dtm(word_counts, minfq=2)
ncol(chapters_dtm)
```

```
## [1] 10717
```

Let us compare it to the case if we include all words.

```
chapters_dtm_all <- convert_to_dtm(word_counts, minfq=0)
ncol(chapters_dtm_all)
```

```
## [1] 18016
```

We also want to compare this to a reduction of the word dictionary by the tfidf. For the sake of comparison the reduction is made to the same value as used above via minfreq=2 (i.e. 10685 words).

```
chapters_dtm_tfidf <- convert_to_dtm_2(word_counts, top=10685)
ncol(chapters_dtm_tfidf)
```

```
## [1] 10685
```