Text as Data Problem Set 1

Ma Adelle Gia Arbo

14 October 2022

Contents

I. Getting and parsing texts

```
# loading packages
if (!require("pacman")) install.packages("pacman")
pacman::p_load(quanteda, tidyr, purrr, ggplot2,
               tidytext, httr, rvest, readr, xml2, reshape2,
               stringr, stringi, dplyr, tibble, lexicon)
```

First, let's read the text file *pg1934.txt* containing the collection of poems *Songs of Innocence and of Experience* by William Blake from Project Gutenberg. To make parsing easier in the next steps, I already split the text file wherein one line is one element of the list, *pg1934*.

```
p <- readr::read_file("./data/raw/pg1934.txt")
pg1934 <- str_split(p, "\n")[[1]]
```

Next, from the list that I created above, I obtained the table of contents to get a list of titles of the 47 poems, 19 of which are from the Songs of Innocence and 28 are from the Songs of Experience .

```
# table of contents
soi_lines <- str_which(pg1934, pattern = fixed("SONGS OF INNOCENCE"))
soe_lines <- str_which(pg1934, pattern = fixed("SONGS OF EXPERIENCE"))

toc_soi <- pg1934[(soi_lines[3]+1):(soe_lines[1]-1)] %>%
            str_subset(".+") %>%
            str_trim()

toc_soe <- pg1934[(soe_lines[1]+1):(soi_lines[4]-1)] %>%
            str_subset(".+") %>%
            str_trim() %>%
            str_replace("My Pretty Rose-Tree", "My Pretty Rose Tree")

toc <- append(toc_soi, toc_soe)

toc
```

```
 [1] "Introduction"                  "The Shepherd"
 [3] "The Echoing Green"             "The Lamb"
 [5] "The Little Black Boy"          "The Blossom"
 [7] "The Chimney-Sweeper"           "The Little Boy Lost"
 [9] "The Little Boy Found"          "Laughing Song"
[11] "A Cradle Song"                 "The Divine Image"
[13] "Holy Thursday"                 "Night"
[15] "Spring"                        "Nurse's Song"
[17] "Infant Joy"                    "A Dream"
[19] "On Another's Sorrow"           "Introduction"
[21] "Earth's Answer"                "The Clod and the Pebble"
[23] "Holy Thursday"                 "The Little Girl Lost"
[25] "The Little Girl Found"         "The Chimney-Sweeper"
[27] "Nurse's Song"                  "The Sick Rose"
[29] "The Fly"                       "The Angel"
[31] "The Tiger"                     "My Pretty Rose Tree"
[33] "Ah, Sunflower"                 "The Lily"
[35] "The Garden of Love"            "The Little Vagabond"
[37] "London"                        "The Human Abstract"
[39] "Infant Sorrow"                 "A Poison Tree"
[41] "A Little Boy Lost"             "A Little Girl Lost"
[43] "A Divine Image"                "A Cradle Song"
[45] "To Tirzah"                     "The Schoolboy"
[47] "The Voice of the Ancient Bard"
```

For this part, I extracted the main body of the file containing all of the 47 poems.

```
# main body
start <- soi_lines[4]
end <-  str_which(pg1934, pattern = fixed("*** END OF THE PROJECT GUTENBERG EBOOK SONGS OF INNOCENCE AN
body <- pg1934[(start):(end-1)] #%>%
          #stri_remove_empty() %>%
          #str_subset(".+") %>%
          ##str_trim()
body <- append(body, "") %>%
        str_replace("SONGS OF EXPERIENCE", "")
```

```
head(body, 33)
```

```
 [1] "SONGS OF INNOCENCE\r"
 [2] "\r"
 [3] "\r"
 [4] "\r"
 [5] "\r"
 [6] "INTRODUCTION\r"
 [7] "\r"
 [8] "\r"
 [9] "Piping down the valleys wild,\r"
[10] "    Piping songs of pleasant glee,\r"
[11] "On a cloud I saw a child,\r"
[12] "    And he laughing said to me:\r"
[13] "\r"
[14] "'Pipe a song about a Lamb!'\r"
[15] "    So I piped with merry cheer.\r"
[16] "'Piper, pipe that song again.'\r"
[17] "    So I piped: he wept to hear.\r"
[18] "\r"
[19] "'Drop thy pipe, thy happy pipe;\r"
[20] "    Sing thy songs of happy cheer!'\r"
[21] "So I sung the same again,\r"
[22] "    While he wept with joy to hear.\r"
[23] "\r"
[24] "'Piper, sit thee down and write\r"
[25] "    In a book, that all may read.'\r"
[26] "So he vanished from my sight;\r"
[27] "    And I plucked a hollow reed,\r"
[28] "\r"
[29] "And I made a rural pen,\r"
[30] "    And I stained the water clear,\r"
[31] "And I wrote my happy songs\r"
[32] "    Every child may joy to hear.\r"
[33] "\r"
```

Using the main body, I retrieved for the indices where the titles in the tables of contents appear in the body.

```
# index of poem titles
index <- list()
for (i in 1:length(toc)) {
  index[[i]] <- str_which(body, pattern = toupper(toc[i]))
}
index<- unlist(index) %>%
        unique() %>%
        sort(decreasing = F)
index <- append(index, length(body))

index
```

```
 [1]    6   37   53   92  121  162  182  218  234  250  271  317  348  369  429
[16]  465  491  511  542  598  628  664  685  711  782  853  874  890  906  937
```

```
[31]  963  999 1015 1031 1042 1063 1089 1115 1151 1167 1193 1229 1276 1292 1318
[46] 1344 1386 1404
```

With the list of indices, I can now locate the start and end of each of the 47 poems using a specific pattern, as seen in the code below. Then, I saved each poem and appended it into a list.

```
# extract each poem
poems <- list()
for (i in 1:length(toc)) {
  poems[[i]] <- body[((index[i]+3):(index[i+1]-5))]
}

head(poems, 1)
```

```
[[1]]
 [1] "Piping down the valleys wild,\r"
 [2] "    Piping songs of pleasant glee,\r"
 [3] "On a cloud I saw a child,\r"
 [4] "    And he laughing said to me:\r"
 [5] "\r"
 [6] "‘Pipe a song about a Lamb!’\r"
 [7] "    So I piped with merry cheer.\r"
 [8] "‘Piper, pipe that song again.’\r"
 [9] "    So I piped: he wept to hear.\r"
[10] "\r"
[11] "‘Drop thy pipe, thy happy pipe;\r"
[12] "    Sing thy songs of happy cheer!’\r"
[13] "So I sung the same again,\r"
[14] "    While he wept with joy to hear.\r"
[15] "\r"
[16] "‘Piper, sit thee down and write\r"
[17] "    In a book, that all may read.’\r"
[18] "So he vanished from my sight;\r"
[19] "    And I plucked a hollow reed,\r"
[20] "\r"
[21] "And I made a rural pen,\r"
[22] "    And I stained the water clear,\r"
[23] "And I wrote my happy songs\r"
[24] "    Every child may joy to hear.\r"
```

Now that I have a list of poems, wherein each poem consists a of list of its lines, I can split this further into stanzas by joining the all lines of the poem then splitting it based on a specific pattern, as coded below.

```
# extract stanzas of each poem
stanzas <- list()
for (i in 1:length(poems)) {
  stanza=str_c(poems[[i]], collapse="\r")
  stanzas[i] = str_split(stanza, "\r\r\r\r")
}

head(stanzas, 2)
```

```
[[1]]
```

```
[1] "Piping down the valleys wild,\r\r    Piping songs of pleasant glee,\r\rOn a cloud I saw a child,\r
[2] "‘Pipe a song about a Lamb!’\r\r    So I piped with merry cheer.\r\r‘Piper, pipe that song again.’\
[3] "‘Drop thy pipe, thy happy pipe;\r\r    Sing thy songs of happy cheer!’\r\rSo I sung the same again
[4] "‘Piper, sit thee down and write\r\r    In a book, that all may read.’\r\rSo he vanished from my sig
[5] "And I made a rural pen,\r\r    And I stained the water clear,\r\rAnd I wrote my happy songs\r\r

[[2]]
[1] "How sweet is the shepherd’s sweet lot!\r\rFrom the morn to the evening he strays;\r\rHe shall follo
[2] "For he hears the lambs’ innocent call,\r\rAnd he hears the ewes’ tender reply;\r\rHe is watchful wh
```

I tidied the list of table of contents into a dataframe (toc_df) and created columns book_title, poem_title, and the poem id that is poem_number. I also tidied the list of stanzas into a dataframe (stanzas_df) from which I can extract the lines of each stanza per poem. In this way, I can create a dataframe with rows as lines of each poem, and it is easier to create the stanza number of each poem using group_by.

```
# convert toc list to df
toc_df <- enframe(toc) %>%
          mutate(book_title = ifelse(name<=19, "Songs of Innocence", "Songs of Experience")) %>%
          rename(poem_title = value, poem_number = name)

# convert stanzas list to df
stanzas_df <- enframe(stanzas) %>%
              rename(stanza = value) %>%
              unnest(stanza) %>%
              rename(poem_number=name)

# extract lines per stanza
lines_df <- stanzas_df %>%
              mutate(id = 1:nrow(stanzas_df)) %>%
              group_by(poem_number) %>%
              mutate(stanza_number = row_number(id)) %>%
              mutate(line = str_split(stanza, "\r\r")) %>%
              unnest(line) %>%
              subset(!line %in% c("\r","\r\r", ""))
```

Finally, I merged toc_df and lines_df to create the clean dataset with columns:

- poem_number
- poem_title
- stanza_number
- line
- line_number

I also saved the final dataframe into a csv format.

```
# merge data
df <- merge(toc_df, lines_df, by = "poem_number")
df$line <- str_trim(df$line)
df$idl <- 1:nrow(lines_df)

df <- df %>%
  group_by(poem_number) %>%
  mutate(line_number = row_number(idl)) %>%
```

```
  ungroup() %>%
  select(-c(stanza,id,idl))

# save dataset
write.csv(df, "./data/tidy/Songs_of_Innocence_Experience.csv")
```

```
head(df, 10)
```

```
# A tibble: 10 x 6
   poem_number poem_title   book_title         stanza_number line        line_~1
         <int> <chr>        <chr>                      <int> <chr>         <int>
 1           1 Introduction Songs of Innocence             1 Piping dow~       1
 2           1 Introduction Songs of Innocence             1 Piping son~       2
 3           1 Introduction Songs of Innocence             1 On a cloud~       3
 4           1 Introduction Songs of Innocence             1 And he lau~       4
 5           1 Introduction Songs of Innocence             2 'Pipe a so~       5
 6           1 Introduction Songs of Innocence             2 So I piped~       6
 7           1 Introduction Songs of Innocence             2 'Piper, pi~       7
 8           1 Introduction Songs of Innocence             2 So I piped~       8
 9           1 Introduction Songs of Innocence             3 'Drop thy ~       9
10           1 Introduction Songs of Innocence             3 Sing thy s~      10
# ... with abbreviated variable name 1: line_number
```

II. Visualising text data

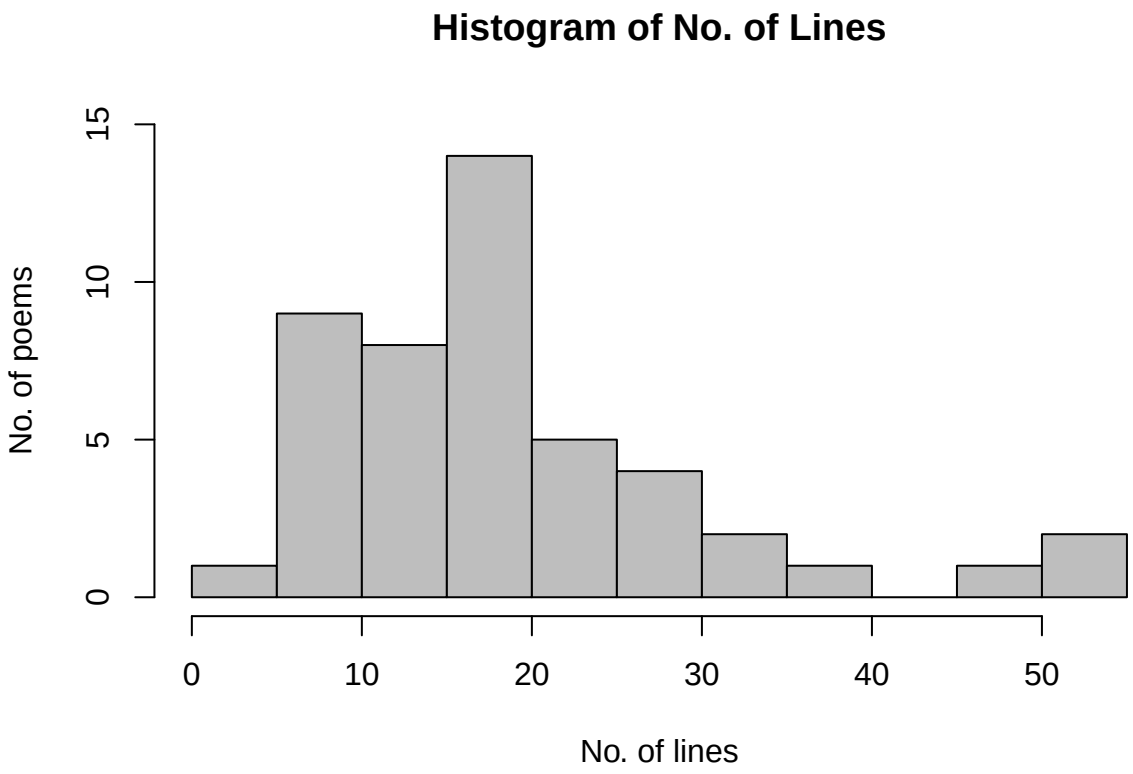
Create a histogram showing the number of lines per poem

I first summarized the data to visualize the histogram and the bar plots.

```
df_sum <- df %>%
  group_by(book_title, poem_title, poem_number) %>%
  summarise(n = n(), .groups = 'drop') %>%
  arrange(desc(n))
```

Below is the histogram showing the distribution of the number of lines across the 47 poems. It looks like most of the poems consists of 15-20 lines. I also saved all plots into a folder.

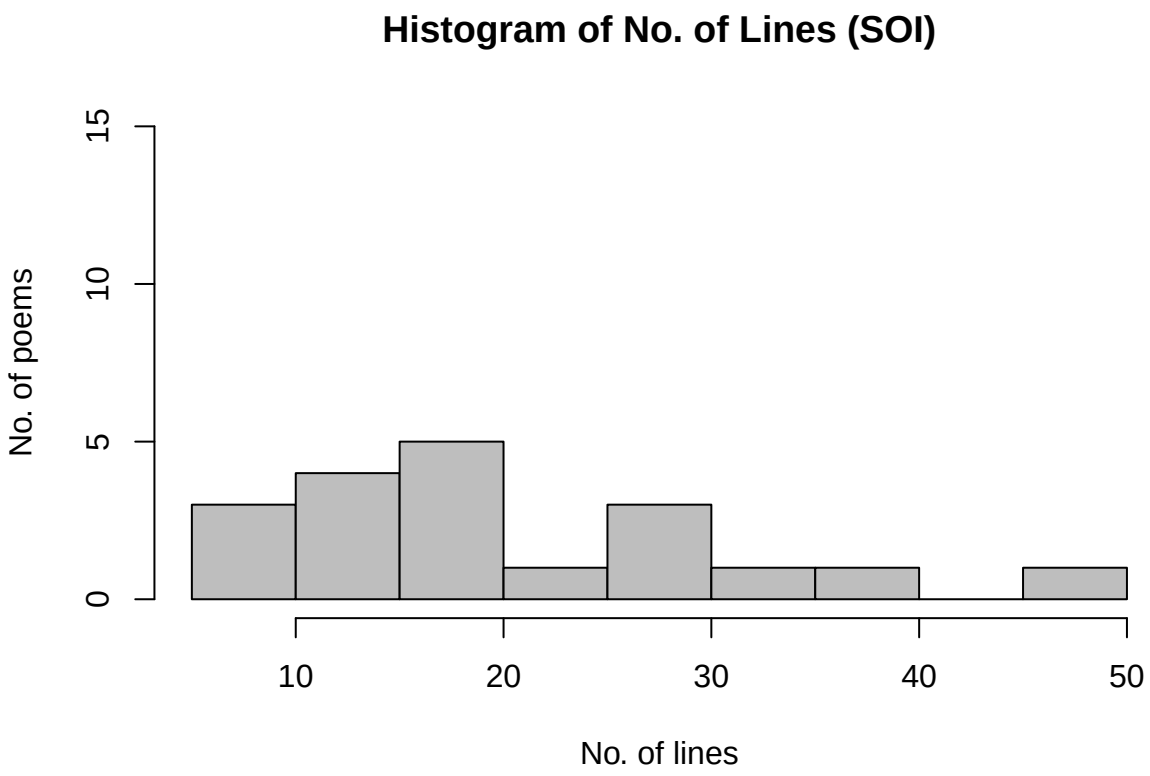
```
hist(df_sum$n, col="gray", main="Histogram of No. of Lines",
     xlab="No. of lines", ylab="No. of poems", ylim=c(0, 15))
```



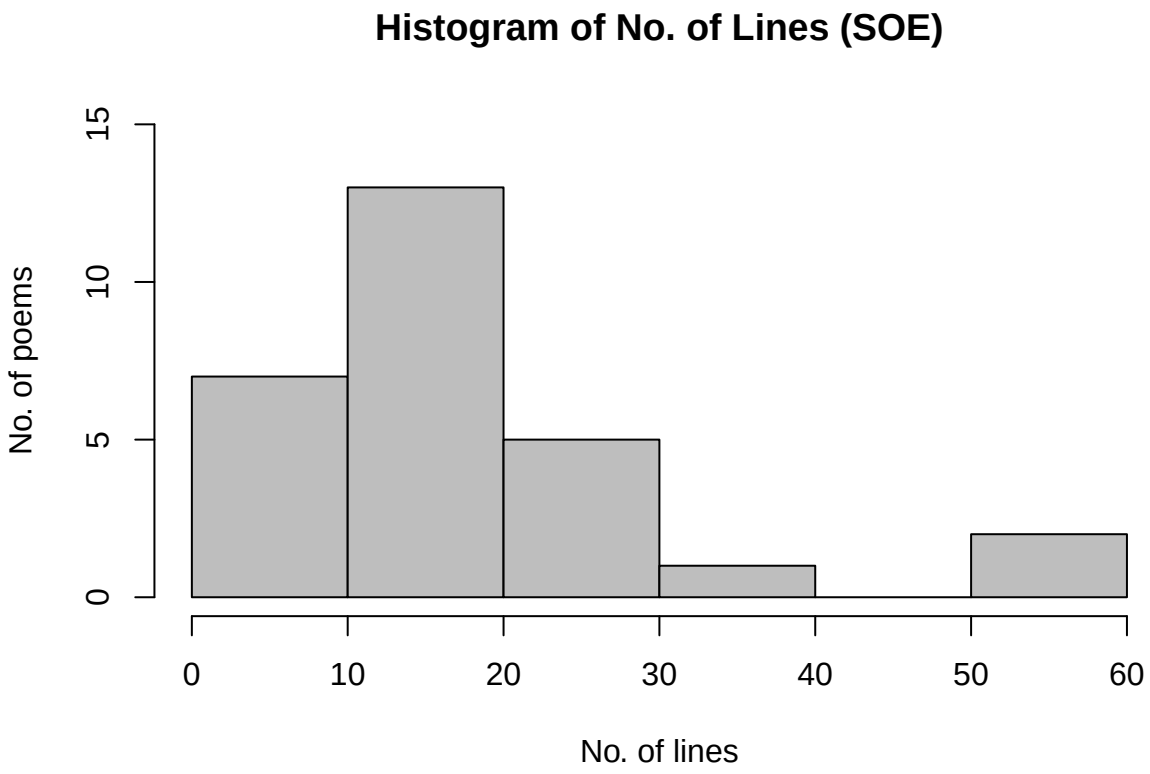

```
png("./plots/histogram.png")
hist(df_sum$n, col="gray", main="Histogram of No. of Lines",
     xlab="No. of lines", ylab="No. of poems", ylim=c(0, 15))
dev.off()
```

```
pdf
  2
```

```
df_sum_soi<-df_sum %>% filter(book_title=="Songs of Innocence") %>% select(n)
hist(df_sum_soi$n,col="gray", main="Histogram of No. of Lines (SOI)",
     xlab="No. of lines", ylab="No. of poems", ylim=c(0, 15))
```




```
df_sum_soe<-df_sum %>% filter(book_title=="Songs of Experience") %>% select(n)
hist(df_sum_soe$n,col="gray", main="Histogram of No. of Lines (SOE)",
     xlab="No. of lines", ylab="No. of poems", ylim=c(0, 15))
```

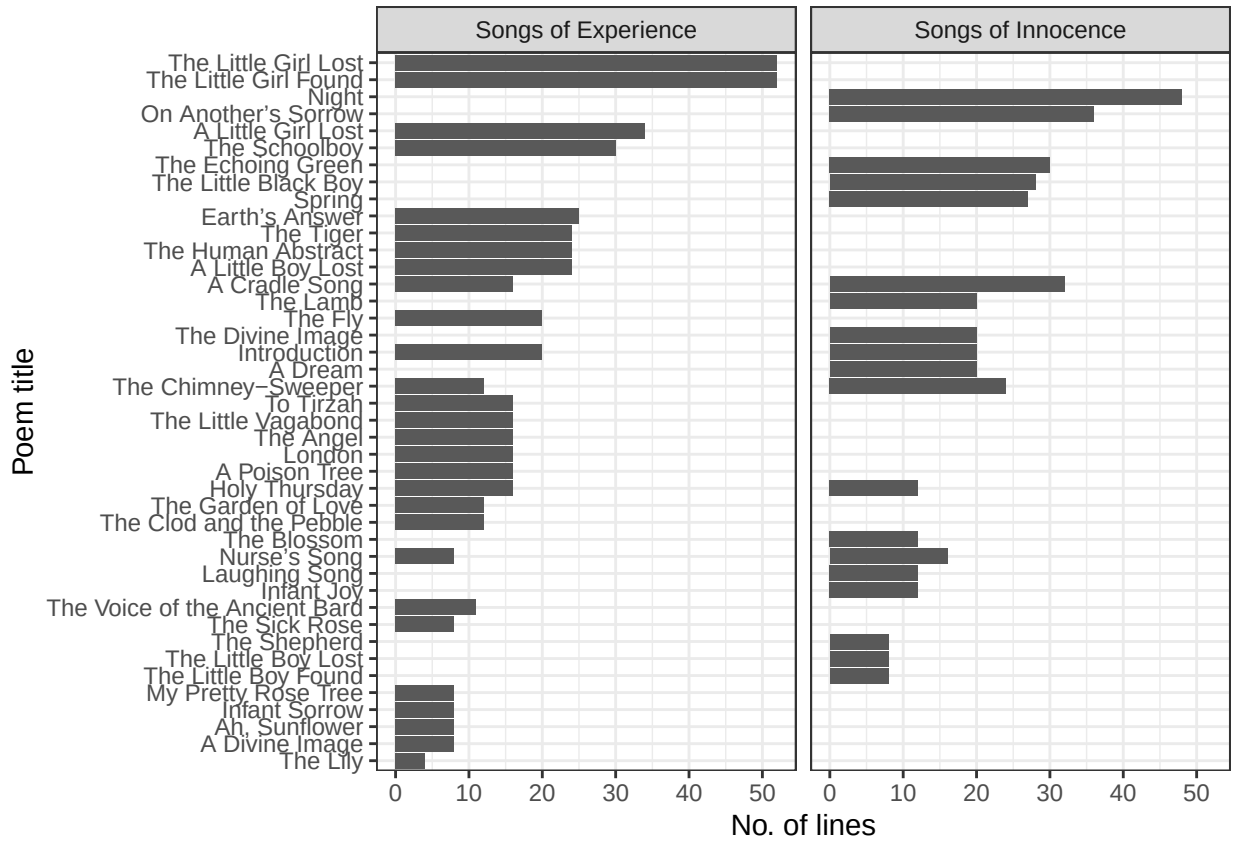



```
df %>% group_by(book_title) %>%
  summarise(n())
```

```
# A tibble: 2 x 2
  book_title          `n()`
  <chr>               <int>
1 Songs of Experience   516
2 Songs of Innocence    393
```

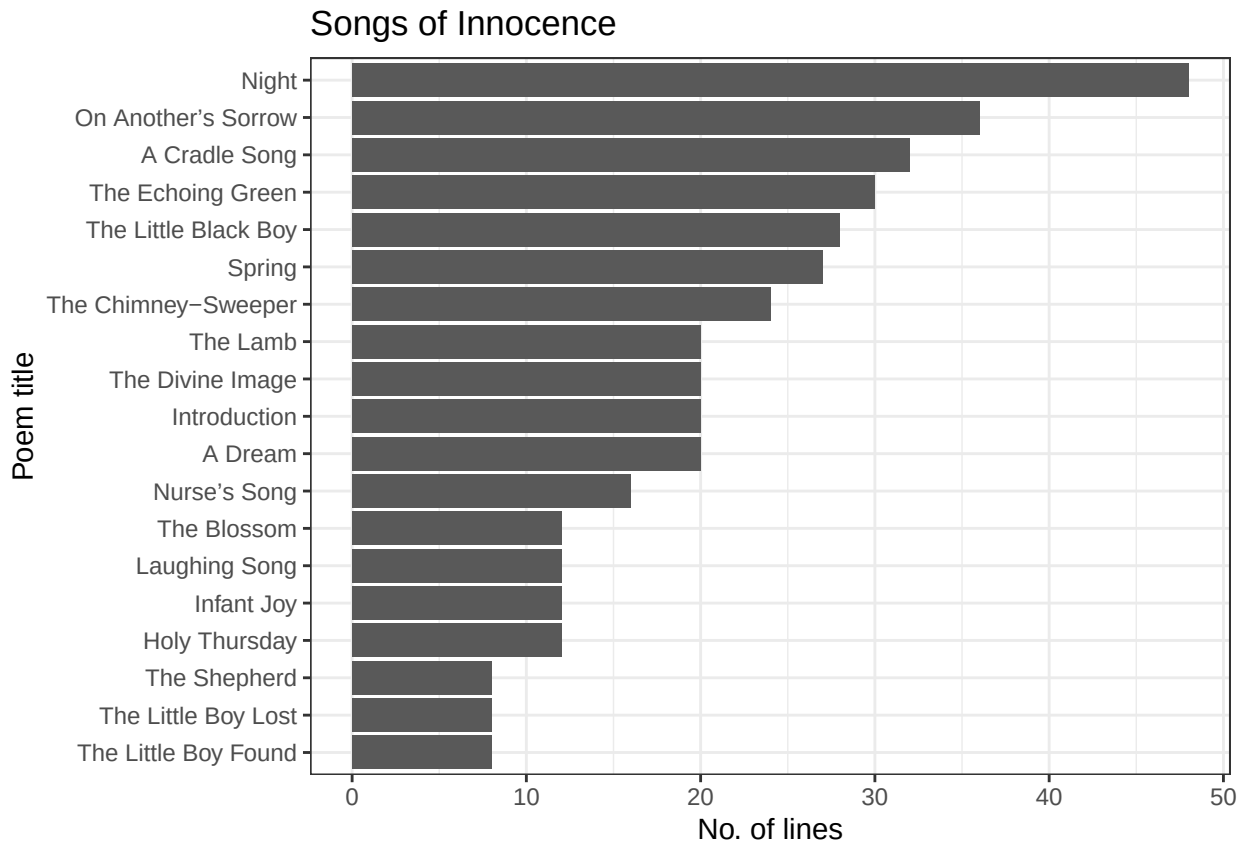
To further investigate this, let's plot a bar graph showing the number of lines per poem for each book. It is found that The Little Girl Lost and The Little Girl Found from the Songs of Experience are the outliers from the histogram with more than 50 lines, and Night from the Songs of Innocence with more than 45 lines.

```
plot_soi_soe <- df_sum %>%
  ggplot(aes(x=reorder(poem_title,n), y=n)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  facet_wrap(~book_title) +
  labs(y = "No. of lines", x = "Poem title") +
  theme_bw()
plot_soi_soe
```



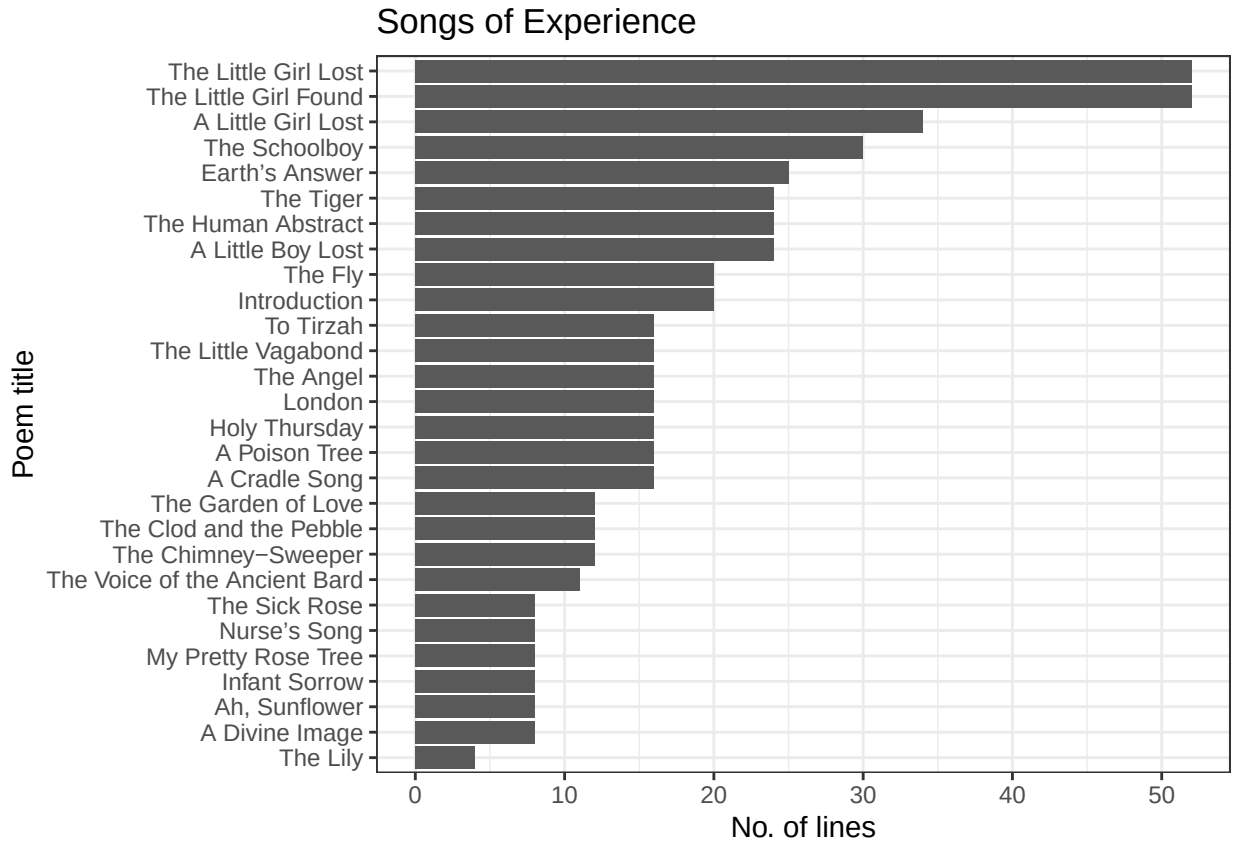

```
ggsave("./plots/plot_soi_soe.png")
```

```
# reorder x values
# SOI
plot_soi <- df_sum %>%
  filter(book_title == "Songs of Innocence") %>%
  ggplot(aes(x=reorder(poem_title,n), y=n)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  labs(y = "No. of lines", x = "Poem title", title = "Songs of Innocence") +
  theme_bw()
plot_soi
```




```
ggsave("./plots/plot_soi.png")
```

```
# SOE
plot_soe <- df_sum %>%
  filter(book_title == "Songs of Experience") %>%
  ggplot(aes(x=reorder(poem_title,n), y=n)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  labs(y = "No. of lines", x = "Poem title", title = "Songs of Experience") +
  theme_bw()
plot_soe
```




```
ggsave("./plots/plot_soe.png")
```

Create a document feature matrix (dfm) treating each line as a document

To create a document feature matrix treating each line as a document, I first created a corpus using of lines. From there, I extracted each word (one gram) of the lines as tokens, removed punctuations, and English stopwords. This led to 1168 features from the 909 documents with 99.73% sparse rate.

```
corp <- corpus(df$line)
dfmat_lines <- corp %>% tokens(remove_punc=TRUE) %>%
  tokens_remove(pattern=stopwords("en")) %>%
  dfm()
# %>%  dfm_trim(min_termfreq=20)
#  %>% dfm_tfidf() # we can also get the tfidf instead of count

dfmat_lines
```

```
Document-feature matrix of: 909 documents, 1,168 features (99.73% sparse) and 0 docvars.
       features
docs    piping valleys wild songs pleasant glee cloud saw child laughing
  text1      1       1    1     0        0    0     0   0     0        0
  text2      1       0    0     1        1    1     0   0     0        0
  text3      0       0    0     0        0    0     1   1     1        0
  text4      0       0    0     0        0    0     0   0     0        1
```

```
  text5      0       0    0     0        0    0     0   0     0        0
  text6      0       0    0     0        0    0     0   0     0        0
[ reached max_ndoc ... 903 more documents, reached max_nfeat ... 1,158 more features ]
```

- Create a separate document feature matrix treating each poem as a document

I first joined the lines of each poem so that each row will contain a poem. After that, the same is done above here to get the dfm with each poem as a document. Here, I did not trim anymore, and it gave 1168 features from the 47 documents with 95.85% sparse rate.

```
df_poems <- df %>%
  group_by(book_title, poem_title, poem_number) %>%
  summarise(poem = str_c(line, collapse="\n"), .groups = 'drop') %>%
  arrange(poem_number)

corp_poems <- corpus(df_poems$poem)
dfmat_poems <- corp_poems %>% tokens(remove_punc=TRUE) %>%
  tokens_remove(pattern=stopwords("en")) %>%
  dfm()
#  dfm_trim(min_termfreq=20)
#  %>% dfm_tfidf()

dfmat_poems
```

```
Document-feature matrix of: 47 documents, 1,168 features (95.85% sparse) and 0 docvars.
       features
docs    piping valleys wild songs pleasant glee cloud saw child laughing
  text1      2       1    1     3        1    1     1   1     2        1
  text2      0       0    0     0        0    0     0   0     0        0
  text3      0       0    0     0        0    0     0   0     0        0
  text4      0       0    0     0        0    0     0   0     2        0
  text5      0       0    1     0        0    0     3   0     1        0
  text6      0       0    0     0        0    0     0   0     0        0
[ reached max_ndoc ... 41 more documents, reached max_nfeat ... 1,158 more features ]
```

- Using one of these document feature matrices, create a plot that compares the frequency of words in each book. Comment on the features that are more or less frequent in one book than another.

For easier and cleaner visualization, I used dfmat_poems into a dfm with min_termfreq of 8 for Songs of Innocence (SOI), which gives me 19 documents with 27 features. For Songs of Experiences (SOE), I also used min_termfreq of 8, which gives me 20 features from the 28 documents. (Note: max_termfreq can also be used to trim the dfm if we want to compare the least occuring words in each book.)

Comparing the two plots of document feature matrix, it can be seen that the word “thee”, “little”, “sweet”, and “joy” are more frequently used in the poems of SOI than in SOE. Meanwhile, words like “thy”, “night”, and “sleep” have more occurrence in SOE than in SOI.

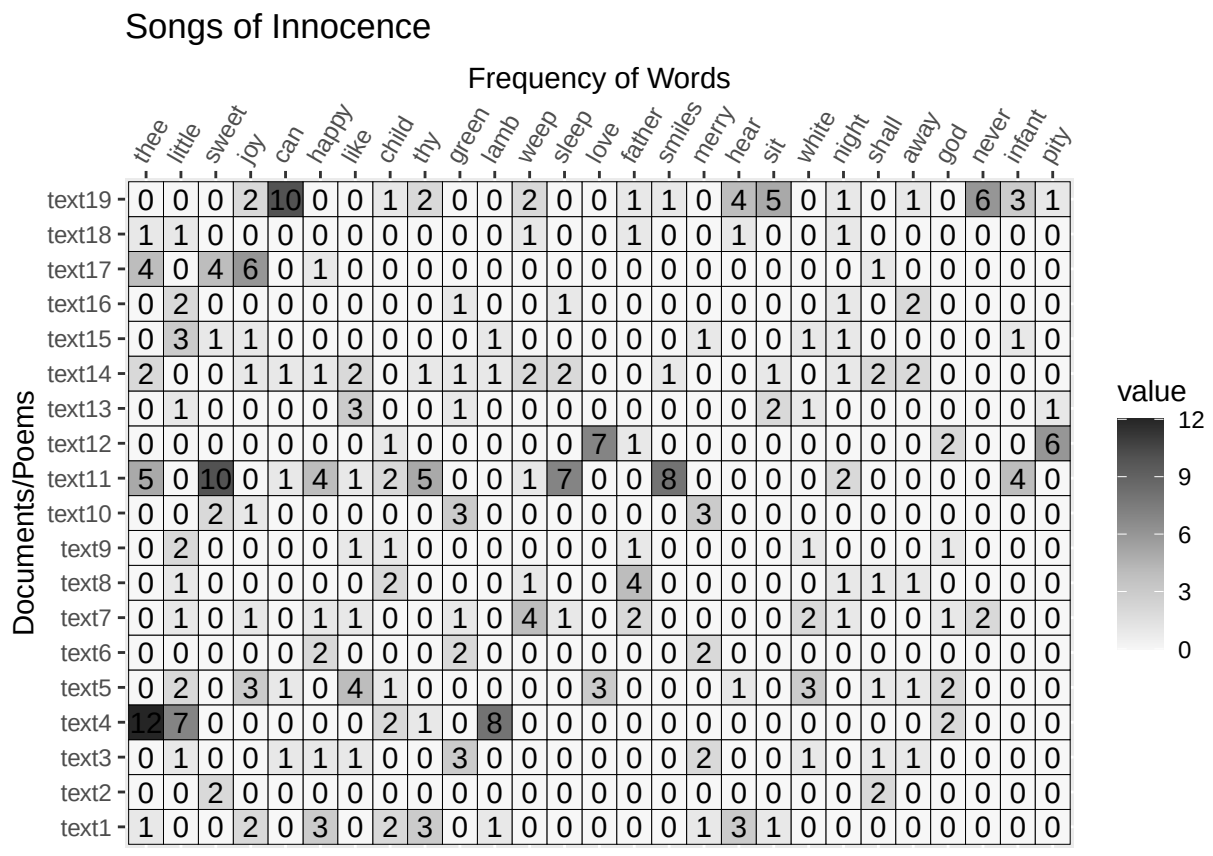
```
dfmat_soi <- dfmat_poems[1:19,] %>% dfm_trim(min_termfreq=8)
dfmat_soi
```

```
Document-feature matrix of: 19 documents, 27 features (71.73% sparse) and 0 docvars.
       features
```

```
docs    child lamb merry hear thy happy joy sit thee sweet
  text1     2    1     1    3   3     3   2   1    1     0
  text2     0    0     0    0   0     0   0   0    0     2
  text3     0    0     2    0   0     1   0   0    0     0
  text4     2    8     0    0   1     0   0   0   12     0
  text5     1    0     0    1   0     0   3   0    0     0
  text6     0    0     2    0   0     2   0   0    0     0
[ reached max_ndoc ... 13 more documents, reached max_nfeat ... 17 more features ]
```

```
# SOI - 1st to 19th poem
tidy_dfm_soi <- melt(as.matrix(dfmat_soi))

ggplot(tidy_dfm_soi, aes(reorder(features,-value), docs)) +
  geom_tile(aes(fill=value), color="black") +
  geom_text(aes(label=value)) +
  coord_fixed() +
  scale_fill_distiller(direction=1, palette="Greys", limits=c(0,12)) +
  scale_x_discrete(position="top") +
  theme(axis.text.x = element_text(angle=60, vjust=0.5, hjust=0)) +
  labs(x = "Frequency of Words",
       y = "Documents/Poems",
       title = "Songs of Innocence")
```



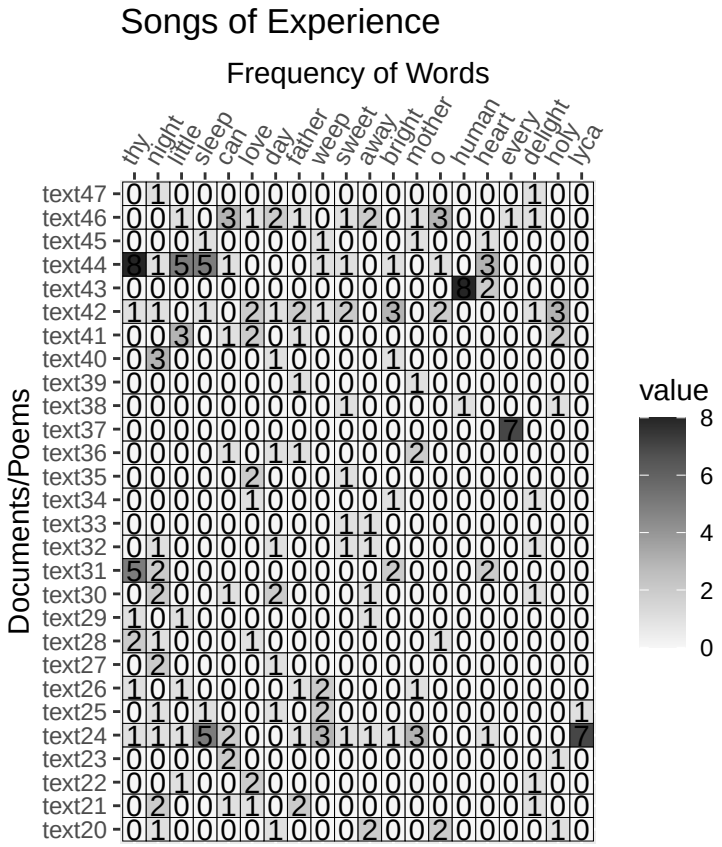

```
ggsave("./plots/dfmat_soi.png")
```

```
dfmat_soe <- dfmat_poems[20:47,] %>% dfm_trim(min_termfreq=8)
dfmat_soe
```

```
Document-feature matrix of: 28 documents, 20 features (77.32% sparse) and 0 docvars.
        features
docs     thy every sweet day away little can delight bright mother
  text20   0     0     0   1    2      0   0       0      0      0
  text21   0     0     0   0    0      0   1       1      0      0
  text22   0     0     0   0    0      1   0       1      0      0
  text23   0     0     0   0    0      0   2       0      0      0
  text24   1     0     1   0    1      1   2       0      1      3
  text25   0     0     0   1    0      0   0       0      0      0
[ reached max_ndoc ... 22 more documents, reached max_nfeat ... 10 more features ]
```

```
# SOE - 20th to 47th poem
tidy_dfm_soe <- melt(as.matrix(dfmat_soe))

ggplot(tidy_dfm_soe, aes(reorder(features,-value), docs)) +
  geom_tile(aes(fill=value), color="black") +
  geom_text(aes(label=value)) +
  coord_fixed() +
  scale_fill_distiller(direction=1, palette="Greys", limits=c(0,8)) +
  scale_x_discrete(position="top") +
  theme(axis.text.x = element_text(angle=60, vjust=0.5, hjust=0)) +
  labs(x = "Frequency of Words",
       y = "Documents/Poems",
       title = "Songs of Experience")
```




```
ggsave("./plots/dfmat_soe.png")
```

III. Parsing XML text data

I used the xml data from the Minutes of the 57th meeting on Thursday, 29 September 2022.

```
# parse XML data
data <- read_html("https://www.bundestag.de/resource/blob/913074/a5fc2d586c98777a4550bda3f3740d77/20057
speeches_xml <- data %>% html_elements("rede")
speeches <- as_tibble(do.call(rbind, html_attrs(speeches_xml)))
speeches$text <- speeches_xml %>% html_text()
```

```
# extract politicians info
politicians <- tibble(id = character(), name=character(), surname=character(), full_name=character(),
                      party=character(), role=character())
for (i in 1:length(speeches_xml)) {
  id = xml_attr(speeches_xml[i][[1]], "id")
  name = speeches_xml[i][[1]] %>% html_element("vorname") %>% html_text()
  surname = speeches_xml[i][[1]] %>% html_element("nachname") %>% html_text()
  full_name = paste0(name, " ", surname)
  party = speeches_xml[i][[1]] %>% html_element("fraktion") %>% html_text()
  role = speeches_xml[i][[1]] %>% html_element("rolle") %>% html_text()

  politicians <- politicians %>% add_row(id=id, name=name,surname=surname,
```

```
                                        full_name=full_name,party=party,role=role)
}

# assuming the full names of the politicians are unique, assign unique id
politicians_df <- politicians %>%
  group_by(full_name) %>%
  mutate(pol_id = cur_group_id()) %>% # 167 pol_ids
  arrange(pol_id, id)
```

```
# extract speeches only, remove name info in start, remove "(Beifall..."
speeches_df <- speeches %>%
  mutate(txt = sub("^[^:]*:", "", text)) %>%
  select(-text) %>%
  mutate(text = str_remove_all(txt, '\\(Beifall.*\\)')) %>%
  select(-txt)
```

```
#  compile data
speeches_df <- merge(politicians_df, speeches_df, by = "id")

# save data
write.csv(speeches_df, "./data/tidy/session_minutes_290922.csv")
```

```
print(sprintf("There are %s speeches delivered by %s politicians during the 57th meeting on Thursday, 29
```

```
[1] "There are 194 speeches delivered by 167 politicians during the 57th meeting on Thursday, 29 Septeml
```

I chose the politician named Alexander Radwan with pol_id == 3. Below, I show the number of speeches and the content of the first speech.

```
pol_3 <- speeches_df %>% filter(pol_id==3)
full_name <- pol_3[1,]$full_name

print(sprintf("%s deliverd %s speeches during the 57th meeting on Thursday, 29 September 2022", pol_3[1
```

```
[1] "Alexander Radwan deliverd 3 speeches during the 57th meeting on Thursday, 29 September 2022"
```

Below is the first speech that Alexander Radwan gave:

```
writeLines(pol_3[1,]$text)
```

```
Frau Präsidentin! Liebe Kolleginnen und Kollegen! Ich spreche jetzt hier nicht, weil ich
Das betrifft uns in Bayern bei den Almen, bei den Alpen, aber auch in Tirol. Und die Mel
Gerade Almbauern arbeiten mit sehr viel Idealismus, mit sehr viel Leidenschaft auf ihrer
Vizepräsidentin Aydan Özoguz:Herr Kollege, gestatten Sie eine Zwischenfrage oder -bemerk

    AlexanderRadwanCDU/CSUAlexander Radwan (CDU/CSU):
Ich möchte das jetzt einfach mal zu Ende führen; ich habe ja eh nur drei Minuten.
Vizepräsidentin Aydan Özoguz:Also: Nein.
```

```
    AlexanderRadwanCDU/CSUAlexander Radwan (CDU/CSU):
Besten Dank. - Dann ist es umso erstaunlicher, wenn bestimmte Ziele, die immer hochgehal
(Zuruf des Abg. Dr. Jan-Niclas Gesenhues [BÜNDNIS 90/DIE GRÜNEN])Meine Damen und Herren
Und der Höhepunkt war vor einem Jahr. Da haben sich die Bauern die Mühe gemacht, mal so
Meine Damen und Herren, in manchen Regionen ist eine Koexistenz nicht möglich. Da darf
Besten Dank, meine Damen und Herren.
Vizepräsidentin Aydan Özoguz:Für eine Kurzintervention erhält das Wort der Kollege Lenk
```

IV. Using regular expressions

```
speech_id <- str_which(speeches_df$text, pattern="\\w*[Kk]ohle\\w*")

sum_id <- sum(str_detect(speeches_df$text, pattern="\\w*[Kk]ohle\\w*"))

print(sprintf("There are %s speeches that mentioned 'kohle' during the 57th meeting on Thursday, 29 Sep
```

```
[1] "There are 11 speeches that mentioned 'kohle' during the 57th meeting on Thursday, 29 September 202
```

```
speeches$id[speech_id]
```

```
 [1] "ID205700100" "ID205700300" "ID205701600" "ID205702000" "ID205703200"
 [6] "ID205705900" "ID205713300" "ID205713500" "ID205713700" "ID205714300"
[11] "ID205715900"
```

```
kohle <- unlist(unique((str_extract_all(speeches_df$text, pattern="\\w*[Kk]ohle\\w*"))))

kohle_tab <- as.tibble(table(kohle)) %>% arrange(desc(n))

print(sprintf("There are %s words that contain 'kohle'.", nrow(kohle_tab)))
```

```
[1] "There are 7 words that contain 'kohle'."
```

```
kohle_tab
```

```
# A tibble: 7 x 2
  kohle               n
  <chr>           <int>
1 Kohle               4
2 Kohlekraftwerke     2
3 Kohleausstieg       1
4 Kohlekraft          1
5 Kohlekraftwerken    1
6 Kohlestrom          1
7 Kohleverstromung    1
```