Geopolitical Changes in the UNSC – Is China's growing global ambitions reflected in its communication in the UNSC?

Erik Valentin Schulte

July 7th 2022

Introduction

References are to be made as follows: Fama and French (1997, 33) and Grinold and Kahn (2000) Such authors could also be referenced in brackets (Grinold and Kahn 2000) and together Grinold and Kahn (2000). Source

Theory Part…

Based on the theory of China's potential shift in policy objectives in the UNSC, I now conduct the empirical evaluation. For this purpose, I first describe the data preparation process. Secondly, I conduct the actual computer assisted text analysis which presents several measures and examines the hypotheses.

Link between Machine Learning and Text Analysis

Data

First of all, I present all relevant data sources and explain essential features to get a proper overview. Afterwards, I explain the process of loading the data into R and how we process the raw text corpus, to obtain meaningful results.

The dataset comprises UN Security Council debates between January 1995 and December 2020 and was downloaded from Schoenfeld et al. (2019). The official meeting protocols are split into distinct speeches. For every speech, metadata regarding the speaker, the speaker's nation or affiliation, and the speaker's role in the meeting is given. The topic of the meeting is also given. In total, the corpus contains 82,165 speeches extracted from 5,748 meeting protocols. Schoenfeld et al. (2019) also provide a codebook with the exact description of their workflow how to set-up the database.

The data are split in to a file containing all information about the UNSC meetings, one file about the metadata of the speeches and one file containing the whole text of the speeches.

Data Sources

For our empirical analysis we use three main data sources. The first two data sources are used for the text analysis and the building of the subgroups. Data on the speeches of the UNSC were obtained from Schönfeld et al. (2019). It contains 82165 speeches over the years 1995 until 2017.

The other two data sources are word dictionaries for the sentiment analysis. It is crucial for the sentiment analysis, that the dictionaries are suitable for the specific context of analysis. For example, a word list developed for sociology may not be a good fit to analyze text data from for finance, and vice versa. The selection of the right word lists is therefore of uttermost importance to get meaningful and robust results Grimmer and Stewart (2013, 274–75). Hence, I carefully choose only certain categories from the dictionaries which are less prone to missclassification. Two wordlists namely Military and Cooperation are from the most widely used standard dictionary, the Harvard General Inquirer (see Hall (2019)). The other four wordlists: Positive, Uncertainty, Strong Modal and Weak Modal are from "Documentation for the LoughranMcDonald_MasterDictionary". Word categories that were specially adapted to the financial context by Loughran and McDonald (2011), such as the negative wordlist, are neglected in my analysis.

Data preparation

In the next step, I import the speech data into R from the data folder of my project file. To be able to process larger amounts of text, I need special packages in R. These include the quanteda, quanteda.textplot and quanteda.textstats. Other libraries like tidyverse, magrtittr, dyplr, ggplot2, readtext, and kableExtra are used to manipulate the data and plot it. We also set a seed, so that our results are reproducible.

```
#load required packages
library(tidyverse)
library(readtext)
library(quanteda)
library(quanteda.textplots)
library(quanteda.textstats)
library(ggplot2)
library(magrittr)
library(gdata)
library(kableExtra)
library(tidytext)
library(dplyr)

#load the UN Security Council Debates dataset
load("data/dataset.RData")
load("data/docs.RData")

#set a seed for reproducability
set.seed(2333)
# View data
#head(meta_meetings)
#head(meta_speeches)
```

I then rename the filename column of the data frame containing the metadata of the speeches into doc_id in order to have a common identifier for merging with the actual text data from the speeches (raw_docs).

```
##rename common column to merge the content of the speeches
meta_speeches <- meta_speeches %>%
  rename(doc_id = filename) |>
   filter(participanttype != "The President")

## merge dataset raw with speeches
meta_speeches <- merge(meta_speeches, raw_docs, by = 'doc_id')
#*
##how to do it after a specific column???
#add_column(meta_speeches, raw_docs[c("text")], .after = "filename")
# meta_speeches |>
# summarize(length_speeches = mean(sentences))
# summarize(mean_tokens = mean(tokens))
# summarize(mean_unique_tokens = mean(types))
# meta_speeches |>
#     group_by(country)
#     count(as.numeric(meta_speeches$sentences))
#     arrange(desc(n))
#
# str(meta_speeches)
```

In the next step I want to make a corpus from my data. The command corpus comes from the quanteda package. Hence, the text column of the meta_speeces dataframe are now a corpus. I filter out speeches from the president of the UNSC, as he or she mostly speaks on behalf of the organization and not for their respective country. This reduces the number of speeches by about 30,000 to 50,933.

```
#as_corpus_frame(meta_speeches$text, filter = NULL, row.names = NULL)
corp_meta_speeches <- corpus(meta_speeches, text_field = "text")
summary(corp_meta_speeches, 1)
```

```
## Corpus consisting of 50933 documents, showing 1 document:
##
##                             Text Types Tokens Sentences speech
##  UNSC_1995_SPV.3487_spch002.txt   575   1746        45      2
##                  country      speaker participanttype role_in_un  spv
##  Bosnia And Herzegovina Mr. Sacirbey           Guest            3487
##           basename
##  UNSC_1995_SPV.3487
##                                                                topic
##  Federal Republic of Yugoslavia (Serbia and Montenegro) - Sanctions
##             date year month day types tokens sentences
##  12 January 1995 1995     1  12   575   1746        45
##                                                    topic2
##  Items relating to the situation in the former Yugoslavia
##                 subtopic agenda_item1      agenda_item2           agenda_item3
##  Bosnia and Herzegovina       Europe Former Yugoslavia Bosnia and Herzegovina
##  decision
##      <NA>
```

```
corp_meta_speeches <- corpus_subset(corp_meta_speeches, participanttype != "The President")
#*
#is_corpus_frame(corp_meta_speeches$text)
```

I also create a unique corpus just for the speeches held by China. I do this using the corpus_subset command from my previously defined corpus. I also take a subset of the meta_speeches dataframe with only the speeches from China. In total there are 3564 speeches by China in the dataframe, a total share of 3.82 % of all speeches. So there appear to be 1619 speeches from China as participant type president. So, the speeches where china speaks on behalf of the president are excluded.

```
#How do I assign the speeches to the columns (now the speeches are in the same order as in the document

corp_China <- corpus_subset(corp_meta_speeches, country == "China")
Chinaspeeches <- subset.data.frame(meta_speeches, country == "China")
share_CHN_speeches <- 1945/50933
```

```r
#*
#only creates value
#Germany <- meta_speeches$country == "Germany"
##make country names small (something wrong)
#chinaspeeches_lower <- corpus_subset(corp_meta_speeches, tolower(country) %in% tolower(corp_China))

## access to document level variables of the China corpus
head(docvars(corp_China))
```

```
##   speech country          speaker participanttype role_in_un             spv
## 1     30   China Mr. Wang Xuexian       Mentioned                       3487
## 2      4   China Mr. Wang Xuexian       Mentioned                       3489
## 3      7   China Mr. Wang Xuexian       Mentioned                       3492
## 4     11   China  Mr. Li Zhaoxing       Mentioned                       3494
## 5     13   China  Mr. Li Zhaoxing       Mentioned                       3496
## 6     16   China Mr. Wang Xuexian       Mentioned            3499Resumption1
##                        basename
## 1           UNSC_1995_SPV.3487
## 2           UNSC_1995_SPV.3489
## 3           UNSC_1995_SPV.3492
## 4           UNSC_1995_SPV.3494
## 5           UNSC_1995_SPV.3496
## 6 UNSC_1995_SPV.3499Resumption1
##                                                                   topic
## 1 Federal Republic of Yugoslavia (Serbia and Montenegro) - Sanctions
## 2                                                               Liberia
## 3                                                      Agenda for Peace
## 4                                                            Mozambique
## 5                                                                 Haiti
## 6                                                                Angola
##               date year month day types tokens sentences
## 1 12 January 1995 1995     1  12   287    707        22
## 2 13 January 1995 1995     1  13   202    432        16
## 3 18 January 1995 1995     1  18   670   1948        69
## 4 27 January 1995 1995     1  27   261    553        18
## 5 30 January 1995 1995     1  30   197    373        14
## 6 8 February 1995 1995     2   8   358    902        28
##                                                      topic2
## 1 Items relating to the situation in the former Yugoslavia
## 2                                The situation in Liberia
## 3                   Items relating to an Agenda for Peace
## 4                             The situation in Mozambique
## 5                           The question concerning Haiti
## 6                                 The situation in Angola
##                                subtopic agenda_item1      agenda_item2
## 1                Bosnia and Herzegovina       Europe Former Yugoslavia
## 2              The situation in Liberia       Africa           Liberia
## 3 Items relating to an Agenda for Peace     Thematic  Agenda for peace
## 4           The situation in Mozambique       Africa        Mozambique
## 5         The question concerning Haiti     Americas             Haiti
## 6               The situation in Angola       Africa            Angola
##            agenda_item3 decision
## 1 Bosnia and Herzegovina     <NA>
## 2                 Liberia     <NA>
## 3                Thematic     <NA>
## 4              Mozambique     <NA>
## 5                   Haiti     <NA>
## 6                  Angola     <NA>
```

```
##extract them the document level variables
#docvars(corp_China, field = "speaker")

#corp_China$year

##create new variables (does not work yet)
#corp_China$avg_speech_length <- (mean(corp_China$sentences))
#avg_speech_length
```

As a next step, I separate the whole speeches into sentences in tidy format. One column for every sentence of a speech. For that I use the unnest_tokens function from the tidytext package. This lets the the number of observations grow to 1,767,696 million. The problem here is that now after every salutation like "Mr." so after every point the function creates a new sentence. When we create a dataframe for every word as an observation, the number of observation grows to 43,534,652 million. Therefore, we must clean the data further before using them for analysis.

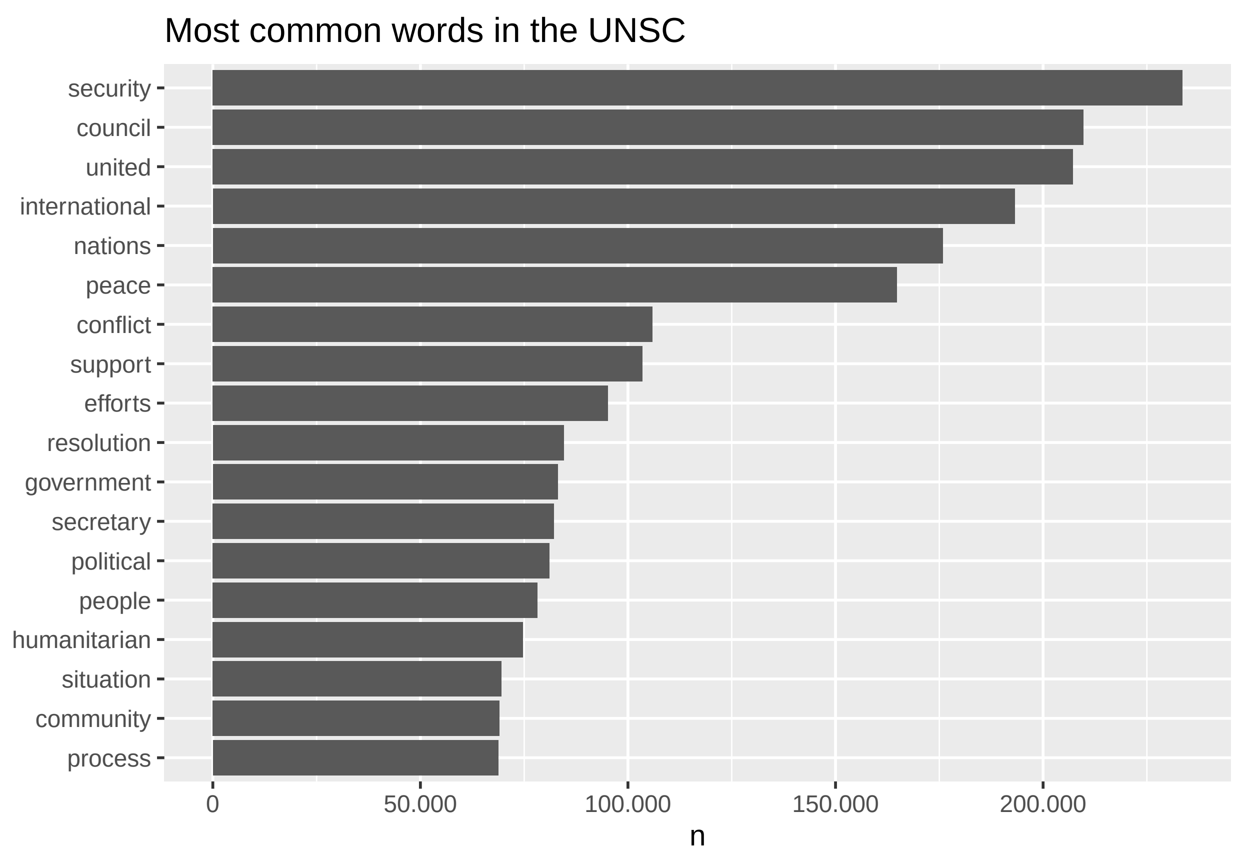
```
##separates by default into words all speeches from raw docs (here by sentences) variable after tokens
tidy_raw_sentences <- meta_speeches %>%
  unnest_tokens(sentences_content, text, token = "sentences" )

##separate into words a new variable each word per speech (gives us 47,4 Mio obs.)
tidy_raw_words <- meta_speeches |>
    unnest_tokens(word, text)
```

The tidytext package also come with a list of stopwords – words that are not meaningful and that we want to exclude from the analysis. The stopwords list contains 1149 words. Our dataframe reduces to 19,623,316 million observations – by more than a half. This allows me to start with a first analysis. I can plot the most frequent words used in the speeches of the UNSC. For that, I plot the most common words that were used more than 80,000 times in total. The threshold of 65,000 is subjectively chosen and is based on the graphical aestethics.

```
##remove redundant words (reduces from 47,6 Mio words to 21,4 Mio words)
data(stop_words)  ##this comes with a package (a list of unuseful words)

tidy_raw_words <- tidy_raw_words %>%
  anti_join(stop_words)

##count words, after redundant words were removed (only meaningful words)
# tidy_raw_words %>%
#   count(word, sort = TRUE)
```

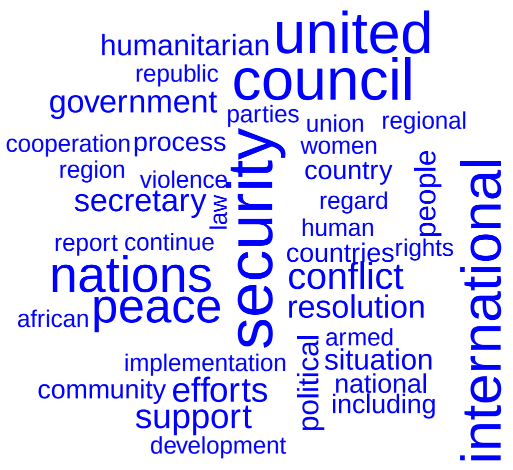
A first visual inspection of the most commonly used words

We can see from the graph that council, security and united are the most frequently used words. This gives are first overview of the most used words.

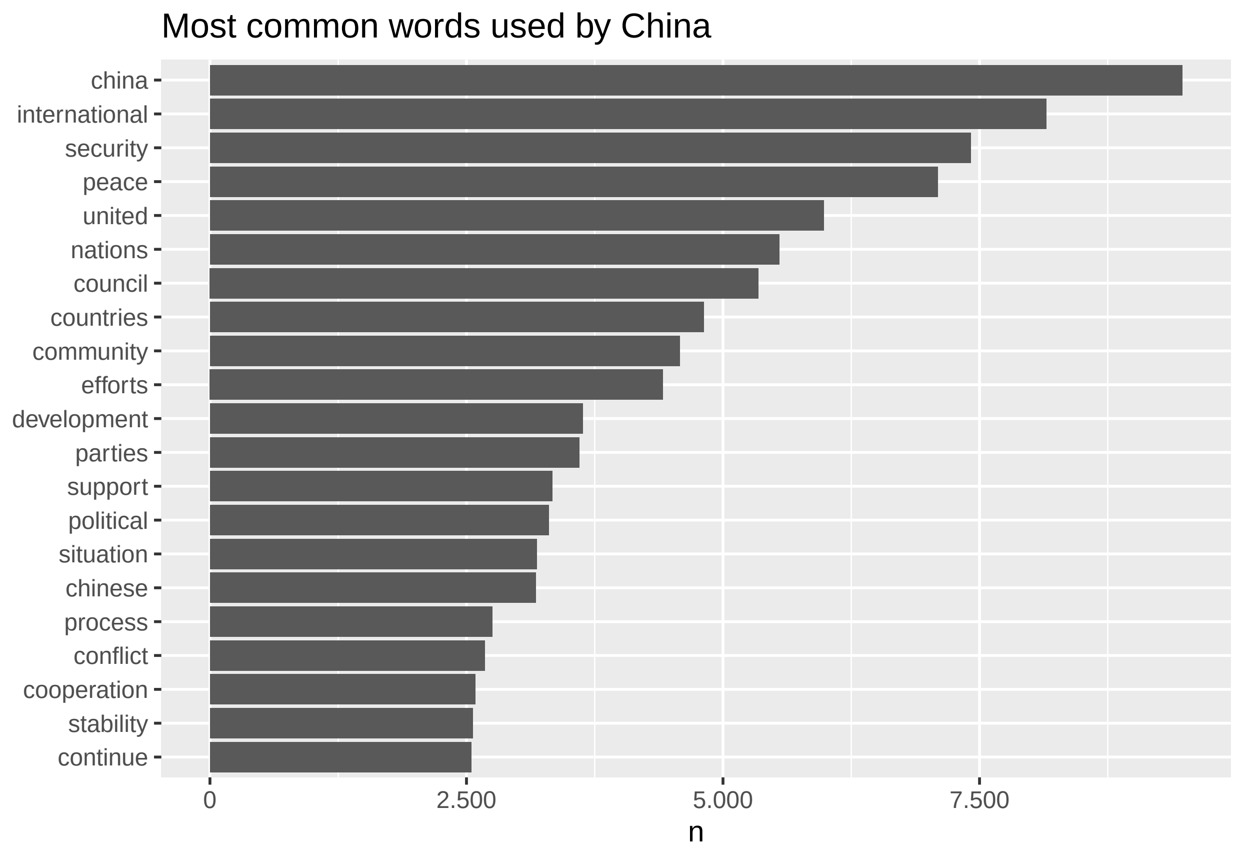


I also plot the 40 most common words from the UNSC in a wordcloud.

```r
# Load the wordcloud package
library(wordcloud)

# Compute word counts and assign to word_counts
word_counts_UNSC <- tidy_raw_words %>%
  count(word)

wordcloud(
  # Assign the word column to words
  word = word_counts_UNSC$word,
  # Assign the count column to freq
  freq =word_counts_UNSC$n,
  scale=c(2,.4),
  max.words = 40,
  colors = "blue"
)
```

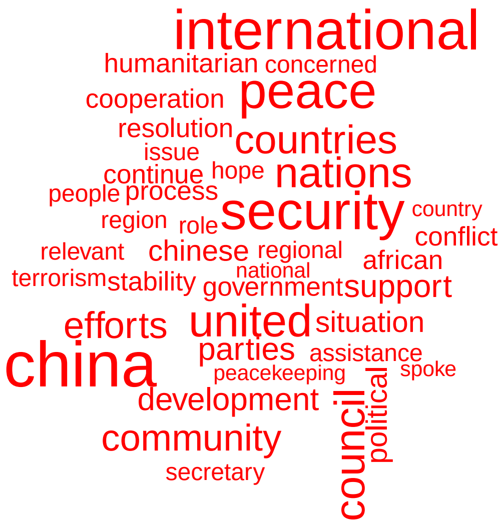


I then apply the same analysis for the data frame from the Chinese speeches.



Additionally, I visualize the 40 most common words in a wordcloud using the wordcloud package.

```
# Load the wordcloud package
library(wordcloud)

# Compute word counts and assign to word_counts
word_counts_CHN <- tidy_raw_words_CHN %>%
  count(word)

wordcloud(
  # Assign the word column to words
  word = word_counts_CHN$word,
  # Assign the count column to freq
  freq =word_counts_CHN$n,
  scale=c(2,.4),
  max.words = 40,
  colors = "red"
)
```



A central question in text mining and natural language processing is how to quantify what a document is about. One measure of how important a word may be is its term frequency (tf), how frequently a word occurs in a document.

Frequency terms

Next, I calculate frequency terms. The function get_freqs creates a list of the frequency of terms that appear in each speech. The function takes the respective corpus as an input and returns the frequencies in the form of a data frame. In the first step, it creates tokens from the text documents. Tokens are a sequence of elementary lexical components, in our case words. Punctuation, separators and stop words like "he," "do," or "if" are excluded as they have no meaning for the content and therefore represent no real added value for the analysis. The dfm() function is then applied to the tokens to create the document feature matrix (dfm). As the dfm is a very sparse matrix containing the documents as rows and the terms as columns, I aggregate the data over all documents and sort the frequency of the terms in a descending order using the textstat_frequency() command. In addition, a special feature of the get_freqs function is the weighting parameter, which is set to FALSE per default. If it is being activated the frequencies are weighted according to the term-frequency inverse document frequency (tf.idf) weighting scheme. This measures how important a word is to a document compared with a corpus of other documents, e.g. one short story in a collection of short stories.

$idf(\text{term}) = \ln\left(\frac{n_{\text{documents}}}{n_{\text{documents containing term}}}\right)$

In the end we apply the get_dfm function to the two corpora and save the result in the variable freqs_all and freqs_CHN respectively.

i have to remove ("interpretation from Chinese first actually)

```
#calculate the frequency of terms that appear in each speech. It also creates a document feature matrix
#frequency of each word in the corpus
# function takes the respective corpus as an input and returns the frequencies in the form of a data fr


get_freqs <- function(corp_meta_speeches, weighting = TRUE){
tokens <- tokens(corp_meta_speeches, remove_punct = T, remove_separators = T, include_docvars = T)
tokens <- tokens_remove(tokens, stopwords("en"))
dfm <- tokens %>% dfm()
freqs <- textstat_frequency(dfm)

if(weighting == TRUE){
freqs$term_frequency <- freqs$frequency / sum(freqs$frequency)
freqs$inverse_doc_freq <- log10( length(corp_meta_speeches) / freqs$docfreq )
freqs$frequency <- freqs$term_frequency*freqs$inverse_doc_freq
}
return(freqs)

}


freqs_CHN <- get_freqs(corp_China,weighting=TRUE)
freqs_all <- get_freqs(corp_meta_speeches,weighting = TRUE)

##why is the frequency in general a bit lower than in the tasks before with the plot where a plot a gra

#Mr. und Mrs. daran kann man vielleicht etwas zur Frauenquote sagen
```

We then get a new dataframe with each word, the frequency (percentage share of appearance of all words), the rank based on the frequency, the total document frequency of the words, the term_frequency and the inverse_term_frequency.

Creating a dataframe which counts every word and shows that we have 13494 unique words in the dataframe used by China after deleting the stopwords.

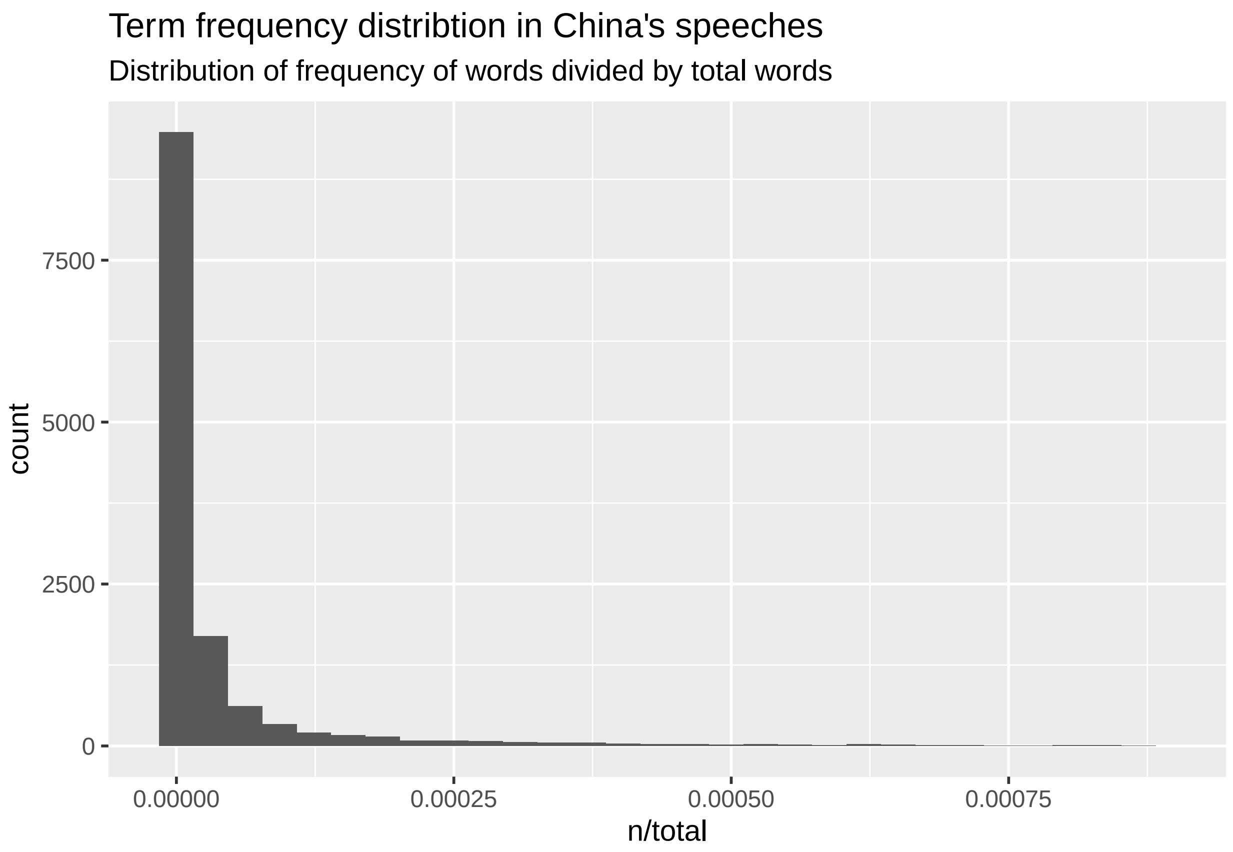
```
speech_words_CHN <- tidy_raw_words_CHN %>%
 # unnest_tokens(word, text) %>%
  count(word, sort = TRUE) |>
    mutate(total = sum(n))

speech_words_CHN_per_speech <- tidy_raw_words_CHN %>%
 # unnest_tokens(word, text) %>%
  count(speech, word, sort = TRUE) |> #mit speech dann ist es per speech sortiert....
    mutate(total = sum(n))
```

Visualizing term frequency in China's speeches

The graph shows the number of times a word appears in the speeches divided by the total number of terms (words) in the speeches. There are very long tails to the right for the Chinese speeches (those extremely rare words!) that I do not include in the plot. Many words occur rarely and few words occur frequently.

```
ggplot(speech_words_CHN, aes(n/total)) +
  geom_histogram(show.legend = FALSE) +
  xlim(NA, 0.0009) +
    labs(
        title = "Term frequency distribtion in China's speeches",
        subtitle = "Distribution of frequency of words divided by total words"
    )
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 195 rows containing non-finite values (stat_bin).

## Warning: Removed 1 rows containing missing values (geom_bar).
```



```
#  facet_wrap(~book, ncol = 2, scales = "free_y")
```

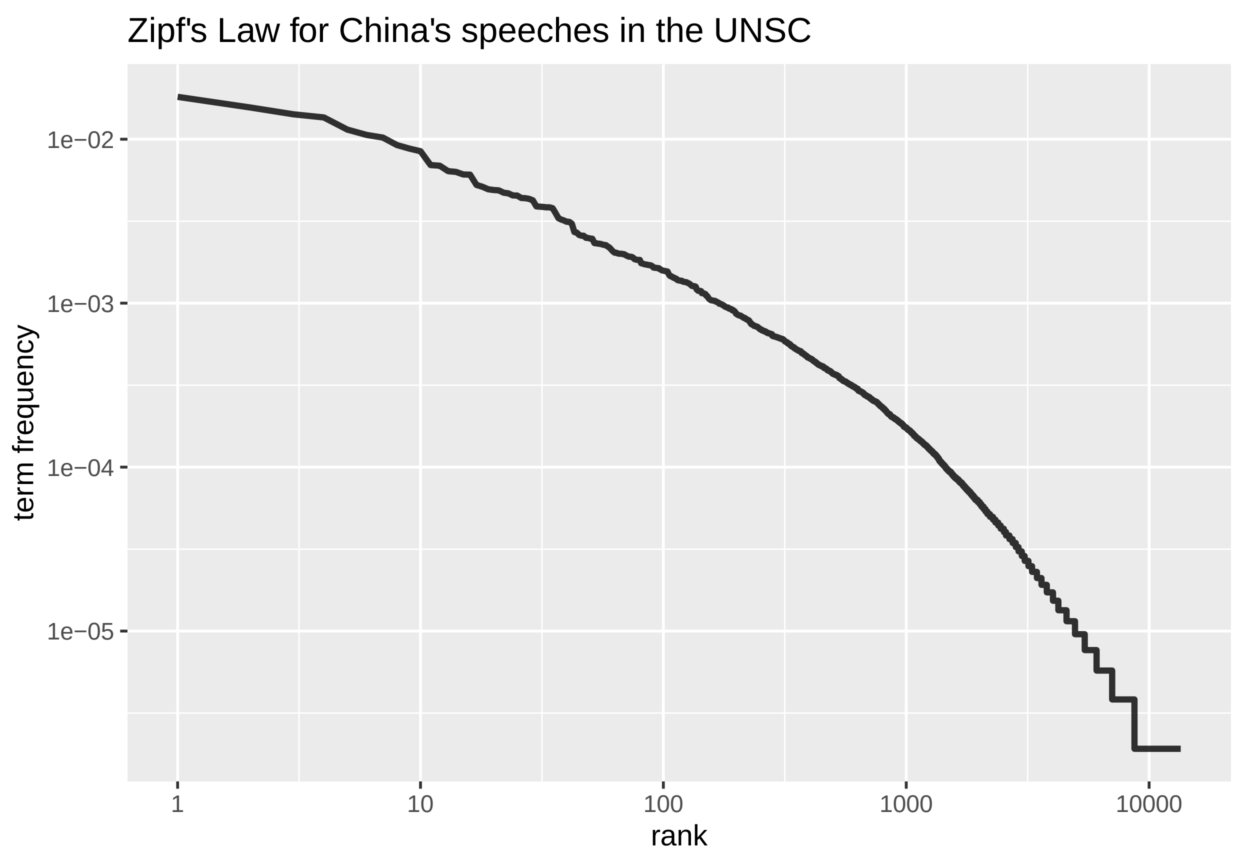
The distribution we can see here is common in language. These long-tailed distributions are very common in natural language (like books, a lot of text from a website, or spoken words). Hence, the relationship between the frequency that a word is used and its rank has been the subject of study. A famous version of this relationship is called Zipf's law, after George Zipf, a 20th century American linguist. Zipf's law states that the frequency that a word appears is inversely proportional to its rank Silge and Robinson (2017).

```r
freq_by_rank_CHN <- speech_words_CHN %>%
 # group_by(book) %>%
  mutate(rank = row_number(),
         `term frequency` = n/total) %>%
  ungroup()

head(freq_by_rank_CHN)
```

```
##            word    n  total rank term frequency
## 1         china 9476 522480    1     0.01813658
## 2 international 8150 522480    2     0.01559868
## 3      security 7414 522480    3     0.01419002
## 4         peace 7095 522480    4     0.01357947
## 5        united 5980 522480    5     0.01144541
## 6       nations 5550 522480    6     0.01062242
```

The rank column here tells us the rank of each word within the frequency table; the table was already ordered by n so we could use row_number() to find the rank. Then, we can calculate the term frequency in the same way we did before. Zipf's law is often visualized by plotting rank on the x-axis and term frequency on the y-axis, on logarithmic scales. Plotting this way, an inversely proportional relationship will have a constant, negative slope.

```r
freq_by_rank_CHN %>%
  ggplot(aes(rank, `term frequency`)) +
  geom_line(size = 1.1, alpha = 0.8, show.legend = FALSE) +
  scale_x_log10() +
  scale_y_log10() +
    labs(
        title = "Zipf's Law for China's speeches in the UNSC"
    )
```



```
#How could I plot the same for all speeches?
```

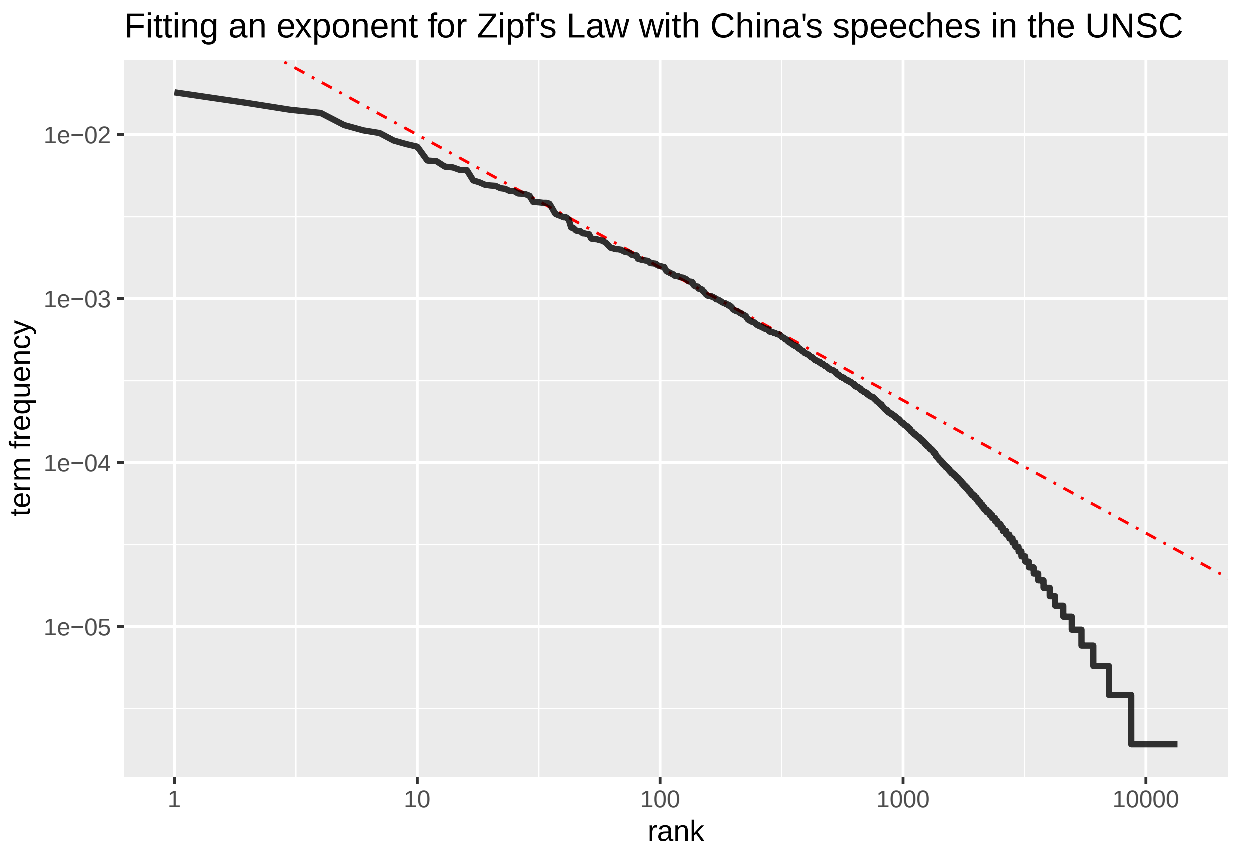
Notice that the Zipf's law figure is in log-log coordinates. We see that the relationship between rank and frequency does have a negative slope. It is not quite constant, though; perhaps we could view this as a broken power law. Let's see what the exponent of the power law is for the middle section (between ranks 10 and 500) of the rank range (Silge and Robinson 2017).

```
rank_subset <- freq_by_rank_CHN %>%
  filter(rank < 500,
         rank > 10)

lm(log10(`term frequency`) ~ log10(rank), data = rank_subset)
```

```
##
## Call:
## lm(formula = log10(`term frequency`) ~ log10(rank), data = rank_subset)
##
## Coefficients:
## (Intercept)  log10(rank)
##     -1.1924      -0.8191
```

Classi versions of Zipf's law: frequency $\propto \frac{1}{\text{rank}}$

```r
freq_by_rank_CHN %>%
  ggplot(aes(rank, `term frequency`)) +
  geom_abline(intercept = -1.19, slope = -0.81, #take these numbers from the regression above
              color = "red", linetype = 4) +
  geom_line(size = 1.1, alpha = 0.8, show.legend = FALSE) +
  scale_x_log10() +
  scale_y_log10() +
    labs(
        title = "Fitting an exponent for Zipf's Law with China's speeches in the UNSC")
```



We have found a result close to the classic version of Zipf's law for the corpus of China's UNSC speeches. The deviations we see here at high rank are not uncommon for many kinds of language; a corpus of language often contains fewer rare words than predicted by a single power law. The deviations at low rank are more less unusual compared to the deviations from the fitted line at high rank. China uses a lower percentage of uncommon words than many collections of language, which may indicate the complexity of the diplomatic language.

The bind_tf_idf() function

The idea of tf-idf is to find the important words for the content of each document by decreasing the weight for commonly used words and increasing the weight for words that are not used very much in a collection or corpus of documents, in this case, the speeches by China in the UNSC. Calculating tf-idf attempts to find the words that are important (i.e., common) in a text, but not too common.

The bind_tf_idf() function in the tidytext package takes a tidy text dataset as input. One column (word here) contains the terms/tokens, one column contains the documents (speech in this case), and the last necessary column contains the counts, how many times each speech contains each term (n in this example). I calculated a total for each speech for our explorations in previous sections, but it is not necessary for the bind_tf_idf() function; the table only needs to contain all the words in each speech.

```
CHN_tf_idf <- speech_words_CHN_per_speech %>%  #89955 variables...
  bind_tf_idf(word, speech, n)

head(CHN_tf_idf)
```

```
##   speech          word   n  total         tf        idf       tf_idf
## 1     12         china 689 522480 0.02115054 0.00000000 0.0000000000
## 2     11         china 658 522480 0.01835938 0.00000000 0.0000000000
## 3     13         china 652 522480 0.01825257 0.00000000 0.0000000000
## 4     14         china 597 522480 0.01729182 0.00000000 0.0000000000
## 5     10         china 581 522480 0.01923586 0.00000000 0.0000000000
## 6     13 international 576 522480 0.01612497 0.02247286 0.0003623741
```

Sentiment Analysis

For the sentiment analysis, I have to get dictionaries for my purpose into R. I choose to do a dictionary-based approach as I am a beginner in text analysis. In accordance with processing of the speech data, we lower all dictionary terms so that the matching function is case sensitive. Two wordlists namely Military and Cooperation are from the most widely used standard dictionary, the Harvard General Inquirer Hall (2019). I manually collected this word list. It was difficult to obtain due to a restricted access. I create the new dataframe dict with these two categories after making all words to lower case letters.

```
library(readxl)
GI_dic <- read_excel("~/ownCloud/Uni Göttingen/Stellenbosch University/Data Science Methods/Github Proj

GI_dic$military <- tolower(GI_dic$military) #make words to lower case in military columns
GI_dic$powercoop <- tolower(GI_dic$powercoop) #mark words to lower case in powercoop column
colnames(GI_dic) <- tolower(colnames(GI_dic))

dict <- GI_dic |>
    select(military, powercoop)
```

The second dictionary I am using is the one by "Documentation for the LoughranMcDonald_MasterDictionary". From this I append the wordlists "Positive,"Uncertainty", "Strongmodal" and "Weakmodal" to my dict.

```
loughran <- read_excel("~/ownCloud/Uni Göttingen/Stellenbosch University/Data Science Methods/Github Pr


colnames(loughran) <- tolower(colnames(loughran))
# Select the following 4 categories for my analysis, exclude negative inter alia
loughran <- loughran[c("positive","uncertainty","strongmodal","weakmodal")]
# make all words to lower case
loughran$positive <- tolower(loughran$positive)
loughran$uncertainty <- tolower(loughran$uncertainty)
loughran$strongmodal <- tolower(loughran$strongmodal)
loughran$weakmodal <- tolower(loughran$weakmodal)
```

```r
#cbindX = column-binds objects with different number of rows.
dict <- cbindX(dict, loughran)
colnames(dict) <- c("Military","Cooperation","Positive","Uncertainty","Strongmodal",
"Weakmodal")
```

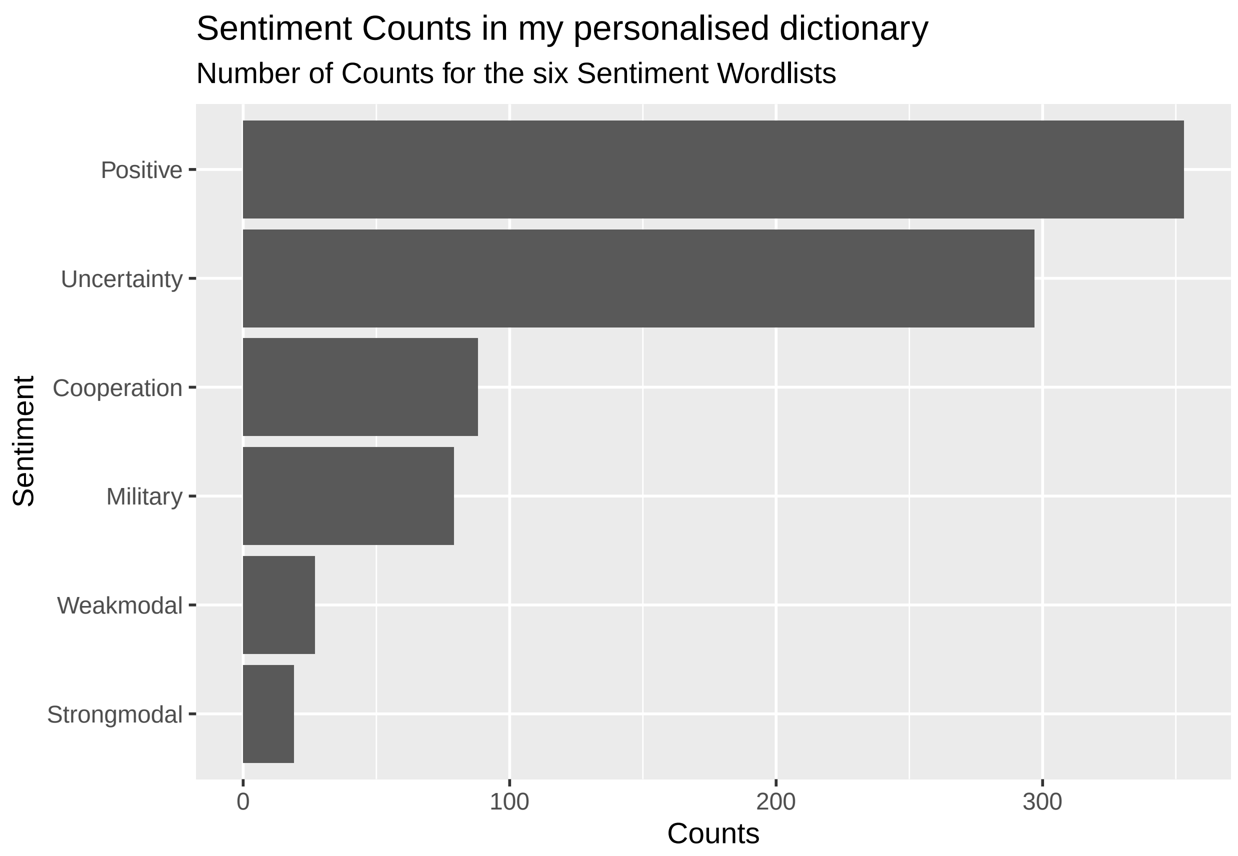
The second important function called sentiment_analysis uses this dict data frame together with the previously created term frequency tables as inputs. It then calculates a proportional count of each category to measure the tone of the language and outputs the result as a data frame. To do so, the function initializes a storage table for our results named data. The column names of the result table are set to the categories contained in the dictionary table. We then iterate through the column names (i.e. the categories) to match the words of each category with the term frequency tables. The resulting data frame join_1 contains only the words, that are in both data frames (this is called an inner join). We then calculate the share of words for each category from the overall frequency. The same is applied to the second term frequency table and then saved to the result table.

```r
sentiment_analysis <- function (freqs_all, freqs_CHN, dict){
data <- data.frame(matrix(NA,2,dim(dict)[2]))  #data is the storage table
colnames(data) <- colnames(dict)

for (i in colnames(data)){
join_1 <- freqs_all %>% inner_join(dict, by= c("feature" = i))
a <- sum(join_1$frequency)/sum(freqs_all$frequency)*100

join_2 <- freqs_CHN %>% inner_join(dict, by= c("feature" = i))
b <- sum(join_1$frequency)/sum(freqs_CHN$frequency)*100

data[i] <- rbind(a,b)
}
return(data)

}
```

My self-created tidy dictionary contains two columns with 863 words and 5 sentiment categories in total. 88 words belong to the Cooperation wordlist (e.g. "co-opeation"), 79 to the Military wordlist (e.g. "army"), 353 to the Positive wordlist (e.g. "accomplishment"), 19 to the strong modal wordlist (e.g. "undisputed"), 297 to the uncertainty wordlist (e.g. "ambiguity") and 27 to the weak modal wordlist (e.g. "apparently."

```
## # A tibble: 863 x 2
##    Sentiment word
##    <chr>     <chr>
##  1 Military  air
##  2 Military  ambush
##  3 Military  ammunition
##  4 Military  arm
##  5 Military  armed
##  6 Military  armistice
##  7 Military  army
##  8 Military  arrow
##  9 Military  battle
## 10 Military  blockade
## # ... with 853 more rows
```

```r
sentiment_counts <- tidy_dict %>%
  count(Sentiment) %>%
 mutate(sentiment2 = fct_reorder(Sentiment, n))
ggplot(sentiment_counts, aes(x = sentiment2, y = n)) +
  geom_col() +
  coord_flip() +
  labs(
    title = "Sentiment Counts in my personalised dictionary",
    subtitle = "Number of Counts for the six Sentiment Wordlists",
    x = "Sentiment",
    y = "Counts"
  )
```



Join the dictionary with the speech data

I create a new dataframe using the inner_join function that appends the my sentiment dictionary in tidy format to the dataframe from China containing all words. I can then see that out of the 522,480 words used by China, 44,588 match to my dictionary (8.53%). I then mutate a new column which calculates the share of the sentiment used. For China, we can see that 38.07% of the sentiments fall into the cooperation wordlist, 10% in the military wordlist, 47.86% in the Positive sentiment wordlist, 3.28% in the uncertainty wordlist and very small shares in the strongmodal and weakmodal categories. What is apparent at first is that, altough the uncertainty wordlist has a multiple times higher share on the words in my dictionary, it is the reverse in the share of sentiments in China's speeches.

```r
sentiment_review_CHN <- tidy_raw_words_CHN |>
    inner_join(tidy_dict)
```

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

```r
sentiment_review_CHN |>
    count(Sentiment) |>
    mutate(share_sentiment_category = n/44588)
```

```
##     Sentiment     n share_sentiment_category
## 1 Cooperation 16976              0.380730241
## 2    Military  4463              0.100094196
## 3    Positive 21339              0.478581681
## 4 Strongmodal   222              0.004978918
## 5 Uncertainty  1461              0.032766664
## 6   Weakmodal   127              0.002848300
```

We can further see the most words that appear both in my dictionary and in China's speeches. The high share of the cooperation sentiment is driven by the very frequent use of the word "peace" (7095 times). This is followed by the words cooperation, stability, strenghten, achieve and progress in descending order.

```r
sentiment_review_CHN |>
    count(word, Sentiment) |>
    arrange(desc(n)) |>
    head()
```

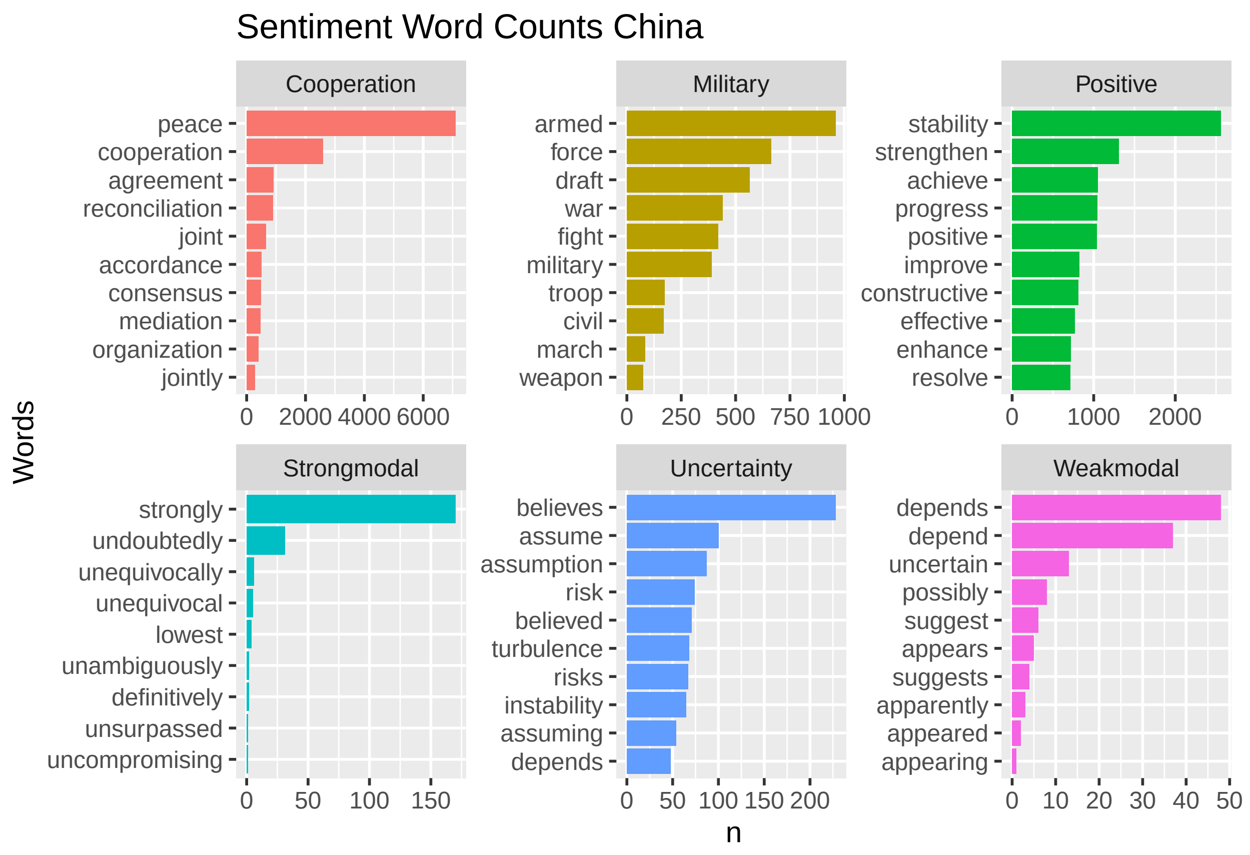
```
##          word   Sentiment    n
## 1       peace Cooperation 7095
## 2 cooperation Cooperation 2587
## 3   stability    Positive 2560
## 4  strengthen    Positive 1308
## 5     achieve    Positive 1050
## 6    progress    Positive 1046
```

visualizing sentiments

```r
#sentiment_review_CHN2 <- sentiment_review %>%
 # filter(sentiment %in% c("positive", "negative"))
word_counts_CHN2 <- sentiment_review_CHN %>%
  count(word, Sentiment) %>%
  group_by(Sentiment) %>%
  top_n(10, n) %>%
  ungroup() %>%
  mutate(
    word2 = fct_reorder(word, n)
  )
```

In this graph, I visualized the sentiments of each of the six categories, so that we can see the most common words for each category.

```
ggplot(word_counts_CHN2, aes(x = word2, y = n, fill = Sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ Sentiment, scales = "free") +
  coord_flip() +
  labs(
    title = "Sentiment Word Counts China",
    x = "Words"
  )
```



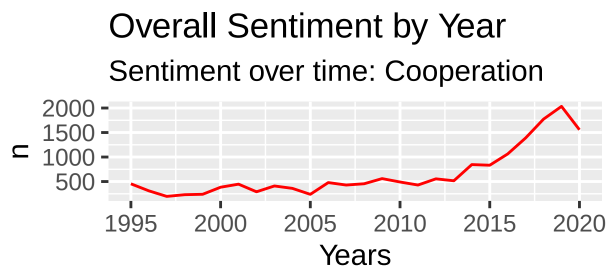
can we count sentiments by year?

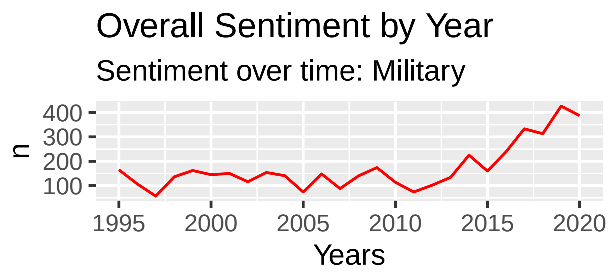
The following gives a table of the counts of the sentiments by Year.

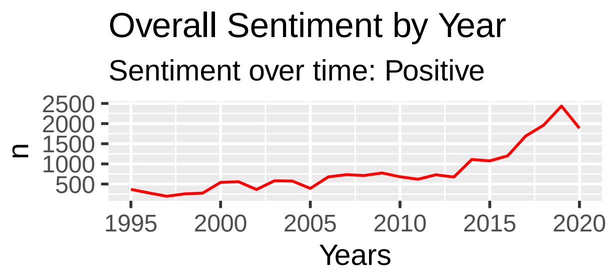
```
sentiment_over_time_CHN <- sentiment_review_CHN |>
    count(year, Sentiment) |>  #see how genius the spread command is
    spread(Sentiment, n)
```

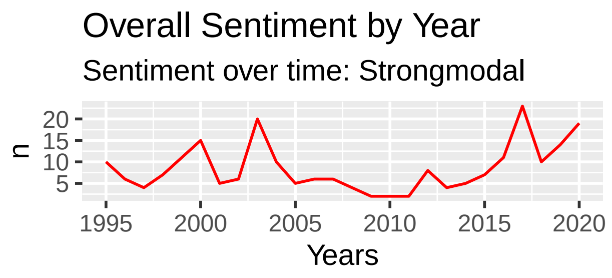
We can also plot this nicely in a graph, which shows the number of the sentiment words used over time for each of the six sentiment categories.

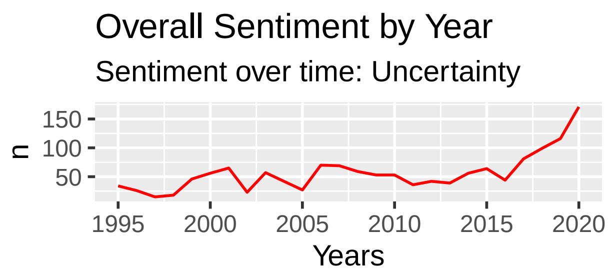
```
#one line of code to put the pictures in one graph
library(gridExtra)
CHN_sentiment_development <- grid.arrange(Cooperation_CHN, Military_CHN, Positive_CHN, Strongmodal_CHN,
```

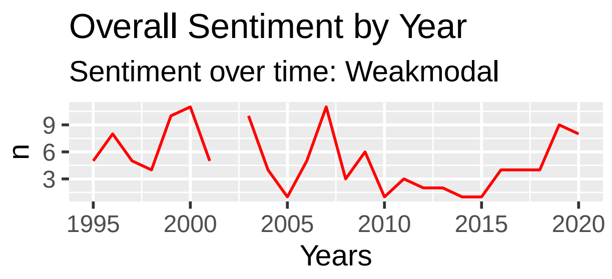












Topic Modelling

For topic modelling we need to create document term matrices (dtm). Sparsity of a matrix refers to how many zeros it contains. It should contain one row for each speech given by China. With the cast_dtm commmand from the tidytext package, we can count each word in each speech and then creates a dtm from the word counts per speech.

```
dtm_review_CHN <- tidy_raw_words_CHN %>%    #assign dtm to tidy_CHN_dataframe
  count(word, speech) %>%
  cast_dtm(speech, word, n) |>     #Cast the word counts by speech into a DTM
    as.matrix()

dtm_review_CHN[1:4, 2000:2008] #this shows output for rows 1-4 and columns 2000-2008
```

```
##     Terms
## Docs barbaric barberi bardales bare bargained bargaining baroness barriers
##   10        1       0        0    0         0          0        2        0
##   20        0       0        0    0         0          0        0        0
##   2         2       0        0    0         0          0        0        0
##   4         0       1        0    0         1          0        0        0
##     Terms
## Docs barring
##   10       0
##   20       0
##   2         0
##   4         0
```

using Latent Dirichlet Allocation (LDA): 2 topics

```
library(topicmodels)
lda_out_CHN <- LDA(
  dtm_review_CHN,
  k = 2,
  method = "Gibbs",
  control = list(seed = 42)
)

glimpse(lda_out_CHN)
```

```
## Formal class 'LDA_Gibbs' [package "topicmodels"] with 16 slots
##   ..@ seedwords      : NULL
##   ..@ z              : int [1:522480] 1 1 1 2 1 1 2 2 2 2 ...
##   ..@ alpha          : num 25
##   ..@ call           : language LDA(x = dtm_review_CHN, k = 2, method = "Gibbs", control = list(seed
##   ..@ Dim            : int [1:2] 45 13494
##   ..@ control        :Formal class 'LDA_Gibbscontrol' [package "topicmodels"] with 14 slots
##   ..@ k              : int 2
##   ..@ terms          : chr [1:13494] "_capacity" "0" "06" "07" ...
##   ..@ documents      : chr [1:45] "10" "20" "2" "4" ...
##   ..@ beta           : num [1:2, 1:13494] -12.1 -15 -12.1 -15 -14.5 ...
##   ..@ gamma          : num [1:45, 1:2] 0.4 0.277 0.68 0.533 0.341 ...
##   ..@ wordassignments:List of 5
##   .. ..$ i   : int [1:86955] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..$ j   : int [1:86955] 1 7 8 15 21 22 24 29 32 40 ...
##   .. ..$ v   : num [1:86955] 1 1 1 2 1 2 2 1 1 2 ...
##   .. ..$ nrow: int 45
##   .. ..$ ncol: int 13494
##   .. ..- attr(*, "class")= chr "simple_triplet_matrix"
##   ..@ loglikelihood  : num -3647091
##   ..@ iter           : int 2000
##   ..@ logLiks        : num(0)
##   ..@ n              : int 522480
```

This gives a tibble of 26988 words that are assigned to two topics. beta gives the word probabilities…

```
lda_topics_CHN <- lda_out_CHN %>%
 tidy(matrix = "beta")
lda_topics_CHN %>%
  arrange(desc(beta))
```

```
## # A tibble: 26,988 x 3
##    topic term           beta
##    <int> <chr>         <dbl>
##  1     1 china         0.0217
##  2     2 international 0.0191
##  3     1 council       0.0179
##  4     2 peace         0.0163
##  5     2 china         0.0159
##  6     1 security      0.0146
##  7     2 security      0.0139
##  8     2 nations       0.0128
##  9     2 countries     0.0127
## 10     2 united        0.0126
## # ... with 26,978 more rows
```

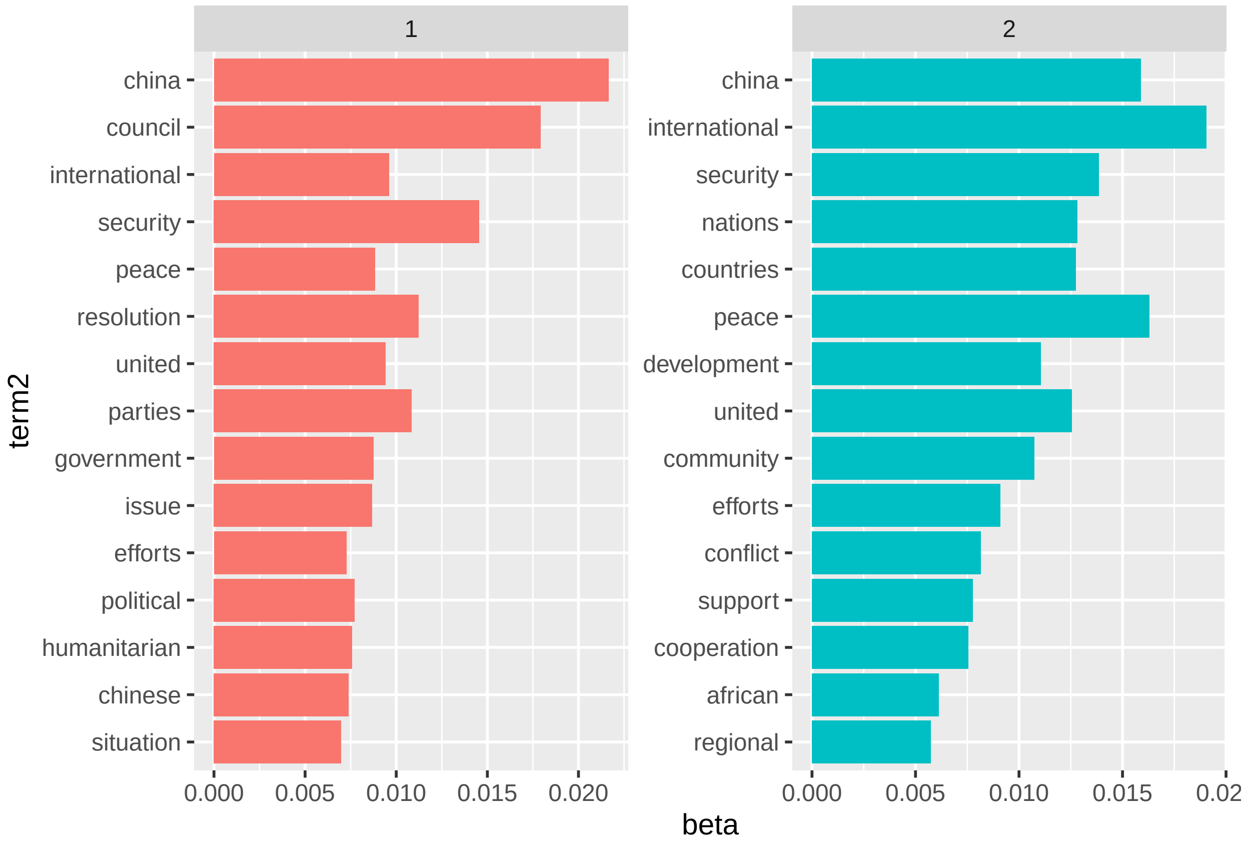
```
lda_out_CHN %>%
 tidy(matrix = "beta")
```

```
## # A tibble: 26,988 x 3
##    topic term             beta
##    <int> <chr>           <dbl>
##  1     1 _capacity 0.00000559
##  2     2 _capacity 0.000000304
##  3     1 0         0.00000559
##  4     2 0         0.000000304
##  5     1 06        0.000000508
##  6     2 06        0.0000125
##  7     1 07        0.0000412
##  8     2 07        0.000000304
##  9     1 08        0.00000559
## 10     2 08        0.0000277
## # ... with 26,978 more rows
```

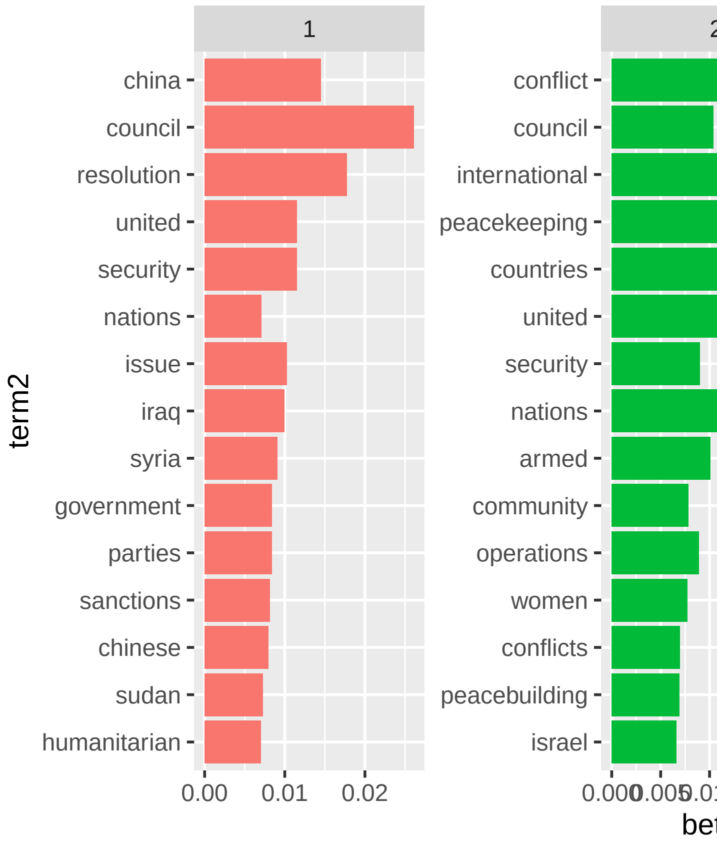
```
word_probs_CHN <- lda_topics_CHN %>%
  group_by(topic) %>%  # Keep the top 15 highest word probabilities by topic
  top_n(15, beta) %>%
  ungroup() %>%
  mutate(term2 = fct_reorder(term, beta)) #this is a factor ordered by word probability
```

I obtain a graph that plots per topic the 15 words with the highest word probability (beta). For the first topic I would assing the topic model “problem-solving.” For the second topic, I suggest the topic

```
# Plot term2 and the word probabilities
ggplot(word_probs_CHN, aes(term2, beta, fill = as.factor(topic))) +
  geom_col(show.legend = F) +
  # Facet the bar plot by topic
  facet_wrap(~ topic, scales = "free") +
  coord_flip()
```


1
2
term2
china
council
international
security
peace
resolution
united
parties
government
issue
efforts
political
humanitarian
chinese
situation
china
international
security
nations
countries
peace
development
united
community
efforts
conflict
support
cooperation
african
regional
0.000
0.005
0.010
0.015
0.020
0.000
0.005
0.010
0.015
0.02
beta

3 topics



Now, we can see the results when we run three topic models. . .

References

"Documentation for the LoughranMcDonald_MasterDictionary." https://www3.nd.edu/~mcdonald/Word_Lists_files/Documentation/Documentation_LoughranMcDonald_MasterDictionary.pdf.

Fama, Eugene F, and Kenneth R French. 1997. "Industry Costs of Equity." *Journal of Financial Economics* 43 (2): 153–93.

Grimmer, Justin, and Brandon M Stewart. 2013. "Text as Data: The Promise and Pitfalls of Automatic Content Analysis Methods for Political Texts." *Political Analysis* 21 (3): 267–97.

Grinold, Richard C, and Ronald N Kahn. 2000. "Active Portfolio Management."

Hall, William James. 2019. "Harvard Iv-4 General Inquire Dictionary." 2019. http://www.wjh.harvard.edu/~inquirer/homecat.htm.

Schoenfeld, Mirco, Steffen Eckhard, Ronny Patz, Hilde van Meegdenburg, and Antonio Pires. 2019. "The UN Security Council Debates." Harvard Dataverse. https://doi.org/10.7910/DVN/KGVSYH.

Schönfeld, Mirco, Steffen Eckhard, Ronny Patz, and Hilde van Meegdenburg. 2019. "The UN Security Council Debates 1995-2017." *arXiv Preprint arXiv:1906.10969*.

Silge, Julia, and David Robinson. 2017. *Text Mining with r: A Tidy Approach.* " O'Reilly Media, Inc.".