gesis Leibniz Institute for the Social Sciences

Automatic Sampling and Analysis of YouTube Data

Excursus: Retrieving Video Subtitles

Johannes Breuer, Annika Deubel, & M. Rohangis Mohseni

February 15th, 2023

Leibniz Association

Retrieving *YouTube* Video Subtitles

Instead of manually transcribing a video, you can retrieve its subtitles via the *YouTube* API.

What research could/would you conduct with video subtitles?

Types of *YouTube* Subtitles

- Videos with automatically created subtitles (*ASR*)
  - Always in English, even if video language is not English
  - Can be downloaded, but text quality can be bad (especially if translated)
- Videos without any subtitles?
  - There always seems to be an *ASR*
- Videos with more than one set of subtitles
  - Examples: *ASR* and regular subtitles, more than one language, more than one subtitle for the same language
  - Can be downloaded, but subtitle for analysis must be selected

Disclaimer

Due to a change in the *YouTube* API, the `tuber` function for retrieving video subtitles only works for videos that were created with the same account as the app used for the API access (see this closed `tuber` issue on GitHub). We will still discuss this function because it has other useful features, but recommend that you use the `youtubecaption` package for collecting subtitles for videos that you have not created yourself.

We will still briefly show you an example of collecting comments with `tuber` here as using `youtubecaption` requires a working installation of *Anaconda* on your computer.

Retrieving Video Subtitles with `tuber`

First, we need to get the list of subtitles for a video.

```
library(tuber)

caption_list <- list_caption_tracks(video_id = "nI_OfkQOG6Q")
```

Note: The `tuber` function `list_caption_tracks()` has an API quota cost of ~ 50.

Retrieving Video Subtitles with `tuber`

Next, we need to get the ID of the subtitles we want to collect.

```
ID <- caption_list[1,"id"]
```

Note: You can adapt the number to select the subtitle that you want (ASR = automatic sub)

Retrieving Video Subtitles with `tuber`

After that, we need to retrieve the subtitles and convert them from raw to char.

```
text <- rawToChar(get_captions(id = ID, format = "sbv"))
```

Now we can save the subtitles to a subtitle file.

```
write(text, file = "Captions.sbv", sep="\n")
```

Converting Subtitles

- Subtitles come in a special format called SBV
- The format contains time stamps etc. that we do not need for text analysis
- We can read the format with the package `subtools`

Converting Subtitles

```
library(subtools)

subs <- read_subtitles("Captions.sbv", format = "subviewer")
```

With `subtools`, we can also retrieve the text from the subtitles.

```
subtext <- get_raw_text(subs)
```

Now the text is ready for further analysis (see the previous sessions for examples).

Retrieving Video Subtitles with `youtubecaption`

- **Pros**:
  - No credentials necessary, therefore no quota reduction
  - Subtitles are automatically converted into a dataframe including texts and timestamps, so no manual conversion is needed
- **Cons**:
  - If there is more than one subtitle version per language, there is no way to select a specific one
  - You need to install *Anaconda*

Time for a Short Live Demo

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function | Addins | Project: (None)

YouTube Subtitles.R

Source on Save | Run | Source

```r
install.packages("stm") # topic modelling
install.packages("tuber") # YouTube data
install_github("fkeck/subtools") # subtitle processing

library(stm)
library(subtools)
library(tuber)


appID <- "" # Insert your own app Id here (OAuth 2.0 Client ID created for your project via the Google Developers platform)
appSecret <- "" # insert your own app Secret here (Client key created for your project via the Google Developers platform)
# NB: Be careful to NOT share these credentials with anyone else whom you don't want to be able to use this account.

## Upon running this line, there will be a prompt in the console asking you to save the access token in a file
## select "No" by entering 2 in the console and hitting enter.
## Afterwards, a browser window should open, prompting you to log in with your Google account
## After logging in, you can close the browser and return to R
yt_oauth(appID,appSecret)

                              #### Extracting Subtitles ####  YdOdTaepyyY PjpS9CftpVo nI_OfkQOG6Q COAY3AN5kI4
# gets video info including the subtitles id
#Caption_list <- list_caption_tracks(part = "snippet", video_id = "nI_OfkQOG6Q") # Video with 1 sub
#Caption_list <- list_caption_tracks(part = "snippet", video_id = "COAY3AN5kI4") # video with 1 automatically created sub
#Caption_list <- list_caption_tracks(part = "snippet", video_id = "3TNkWTRNNYE") # video with 2 subs -> 400 error
#Caption_list <- list_caption_tracks(part = "snippet", video_id = "j93NnRLyym0") # ASR exists, but not selectable
#Caption_list <- list_caption_tracks(part = "snippet", video_id = "YgWgHRjv_Ps") # ARS exists, but no subs displayed
Caption_list <- list_caption_tracks(part = "snippet", video_id = "OuO135_5Rxk") # ARS exists, but no subs displayed

ID <- toString(Caption_list[1,"id"]) # extracts the id and converts it to string
Text <- rawToChar(get_captions(id = ID, format = "sbv")) # retrieves the subtitles (format=raw)

# Known Issues
```

3:55 (Top Level) R Script

Console

Note: You can find the code for collecting and processing subtitles for *YouTube* videos in the `YouTubeSubtitles.R` file in the folder `content\R` within the workshop materials.

Retrieving Video Subtitles with YouTube Summary with ChatGPT

You can also manually retrieve subtitles with the Chrome plugin YouTube Summary with ChatGPT

- **Pros**:
  - Easy to install, easy to use
  - Desired subtitle can be selected
- **Cons**:
  - Manual copy & paste for each video (no automatization)
  - Subtitles are not in standard format; need to be processed

Retrieving Video Subtitles with `YouTube Summary with ChatGPT`

We have created an `R` script named `parse_video_transcript.R` that you can use to import video transcripts/subtitles collected using the Chrome plugin YouTube Summary with ChatGPT.

Once you have sourced the function, you can use it to import all .txt files from a given directory (provided as the `directory` argument) and process them into a dataframe with different columns/variables for the timestamp, text, video name, video URL, and video ID.

Of course, before you can use the function, you need to create at least one `.txt` file that containing the transcript copied from the YouTube Summary with ChatGPT Chrome plugin. The name(s) of the `.txt` file(s) do not matter, but the transcript/subtitles from each video should be

Retrieving Video Subtitles with `YouTube Summary with ChatGPT`

As a side note: You could, potentially, automate the transcript collection via a web scraping approach using `RSelenium` (for an introduction to `RSelenium`, see, e.g., this tutorial). Note, however, that we have not tested this and using web scraping with a browser plugin may be tricky.

Any (further) questions?