Estimations of Quantitative Measures

Chris Bentz

02/01/2023

Load libraries

If the libraries are not installed yet, you need to install them using, for example, the command: install.packages("ggplot2"). For the Hrate package this is different, since it comes from github. The devtools library needs to be installed, and then the install_github() function is used.

```
library(stringr)
library(ggplot2)
library(ggrepel)
library(plyr)
library(entropy)
library(ggExtra)
library(gsubfn)
```

```
## Loading required package: proto
```

```
#library(devtools)
#install_github("dimalik/Hrate")
library(Hrate)
```

List files

Create list with all the files in the directory "data".

```
# give file paths to the files to be processed
file.list <- list.files(path = "~/Github/NaLaFi/data/",
                        recursive = T, full.names = T)
head(file.list)
```

```
## [1] "/home/chris/Github/NaLaFi/data//generated/random/random_ran_10"
## [2] "/home/chris/Github/NaLaFi/data//generated/random/random_ran_11"
## [3] "/home/chris/Github/NaLaFi/data//generated/random/random_ran_12"
## [4] "/home/chris/Github/NaLaFi/data//generated/random/random_ran_13"
## [5] "/home/chris/Github/NaLaFi/data//generated/random/random_ran_14"
## [6] "/home/chris/Github/NaLaFi/data//generated/random/random_ran_15"
```

```
length(file.list)
```

```
## [1] 330
```

Estimations per file

```
# set counter
counter = 0
# set the maximal number of units (n) to be used for analysis
n = 1000
# initialize dataframe to append results to
estimations.df <- data.frame(filename = character(0), subcorpus = character(0),
                         code = character(0), huni.chars = numeric (0),
                         huni.strings = numeric(0), hrate.chars = numeric(0),
                         hrate.strings = numeric(0), ttr.chars = numeric(0),
                         ttr.strings = numeric (0), rm.chars = numeric(0))

# start time
start_time <- Sys.time()
for (file in file.list)
{
  try({ # if the processing failes for a certain file, there will be no output for this file,
  # but the try() function allows the loop to keep running

  # basic processing
  # loading textfile
  textfile <- scan(file, what = "char", quote = "", comment.char = "",
                   encoding = "UTF-8", sep = "\n" , skip = 7)
  # remove annotations marked by '<>'
  textfile <- gsub("<.*>","",textfile)
  # print(head(textfile))
  # get filename
  filename <- basename(file)
  #print(filename) # for visual inspection
  # get subcorpus category
  subcorpus <- sub("_.*", "", filename)
  # print(subcorpus) # for visual inspection
  # get the three letter identification code + the running number
  code <- substring(substring(filename, regexpr("_", filename) + 1), 1, 8)

  # Split into individual characters/signs
  # remove tabs and parentheses, as well as star signs `*' and plus signs `+´
  # note that this might have to be tuned according to the text files included
  textfile <- str_replace_all(textfile, c("\\\t" = "", "\\(" = "", "\\)" = "",
                                          "\\]" = "", "\\[" = "",  "\\}" = "",
                                          "\\{" = "", "\\*" = "", "\\+" = ""))
  # split the textfile into individual utf-8 characters. Note that white spaces are
  # counted as utf-8 characters here.
  chars <- unlist(strsplit(textfile, ""))
  chars <- chars[1:n] # use only maximally n units
  chars <- chars[!is.na(chars)] # remove NAs for vectors which are already shorter
  # than n
  # chars <- chars[chars != " "] # remove white spaces from character vector
  # Split the textfile into strings delimited by white spaces. The output of strsplit()
  # is a list, so it needs to be "unlisted"" to get a vector.
  strings <- unlist(strsplit(textfile, " "))
  strings <- strings[1:n] # use only maximally n units
```

```
  strings <- strings[!is.na(strings)] # remove NAs for vectors which are already
  # shorter than n

  # Unigram entropy estimation
  # calculate unigram entropy for characters
  chars.df <- as.data.frame(table(chars))
  # print(chars.df)
  huni.chars <- entropy(chars.df$Freq, method = "ML", unit = "log2")
  # calculate unigram entropy for strings (Maximum Likelihood method)
  strings.df <- as.data.frame(table(strings))
  # print(strings.df)
  huni.strings <- entropy(strings.df$Freq, method = "ML", unit = "log2")

  # entropy rate estimation
  # the values chosen for max.length and every.word will crucially
  # impact processing time. In case of "max.length = NULL" the length is n units
  hrate.chars <- get.estimate(text = chars, every.word = 1, max.length = NULL)
  hrate.strings <- get.estimate(text = strings, every.word = 1, max.length = NULL)

  # calculate type-token ratio (ttr)
  ttr.chars <- nrow(chars.df)/sum(chars.df$Freq)
  ttr.strings <- nrow(strings.df)/sum(strings.df$Freq)

  # calculate repetition measure according to Sproat (2014)
  # the overall number of repetitions is the sum of frequency counts minus 1.
  R <- sum(chars.df$Freq-1)
  # calculate the number of adjacent repetitions
  r = 0
  if (length(chars) > 1){
    for (i in 1:(length(chars)-1)){
      if (chars[i] == chars[i+1]){
        r = r + 1
      } else {
        r = r + 0
      }
    }
    # calculate the repetition measure
    rm.chars <- r/R
  } else {
    rm.chars <- "NA"
  }

  # append results to dataframe
  local.df <- data.frame(filename, subcorpus, code, huni.chars,
                         hrate.chars, huni.strings, hrate.strings,
                         ttr.chars, ttr.strings, rm.chars)
  estimations.df <- rbind(estimations.df, local.df)
  # counter
  counter <- counter + 1
  # print(counter)
  })
}
```

```
## Error in grepl(pattern = substr, x = fullstr) :
```

```
##   invalid regular expression '\bNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
## Error in grepl(pattern = substr, x = fullstr) :
##   invalid regular expression '\bNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
```

```r
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 1.589457 mins
```

```r
#estimations.df
```

Safe outputs to file

```r
write.csv(estimations.df, paste("~/Github/NaLaFi/results/estimation",
                                paste(n, "chars.csv", sep = ""), sep = ""), row.names = F)
```