useNews

Mario Haim, University of Leipzig Cornelius Puschmann, University of Bremen

The *useNews* dataset is a free-to-use (by citing/referencing it) compilation of three different sources of data over the course of two years. It subsumes representative survey data on the use of news, respective news content, and engagement metrics for the same news content. In total, data is included for twelve countries. This notebook provides detailed information on how the data looks like and it illustrates a series of queries that can be run against the useNews dataset.

The Dataset

The data originates from both the 2019 and the 2020 Reuters Digital News Report, media content from MediaCloud for 2019 and 2020 from all news outlets that have been used most frequently in the respective year according to the survey data, and engagement metrics for all available news-article URLs through CrowdTangle.

To start using the data, a total of eight data objects exist, namely one each for 2019 and 2020 for the survey, news-article meta information, news-article DFM's, and engagement metrics. To make your life easy, we've provided several packaged download options:

- survey data for 2019 only, for 2020 only, or both (also available in CSV format for 2019 and 2020)
- news-article meta data for 2019 only, for 2020 only, or both (also available in CSV format for 2019 and 2020)
- news-article DFM's for 2019 only, for 2020 only, or both
- engagement data for 2019 only, for 2020 only, or both (also available in CSV format for 2019 and 2020)
- all of 2019
- all of 2020

Note that all .rds files are .xz-compressed, which shouldn't bother you when you are in R. You can import all the .rds files through `variable_name <- readRDS('filename.rds')`, .RData (also .xz-compressed) can be imported by simply using `load('filename.RData')` which will load several already named objects into your R environment.

To import data through other programming languages, we also provide all data in respective CSV files. These files are rather large, however, which is why we have also .xz-compressed them. DFM's, unfortunately, are not available as CSV's due to their sparsity and size.

Survey Data from the Reuters Digital News Report

The first datasets represent the *raw survey data* from the 2019 and the 2020 Reuters Digital News Report, including *24,190 representative respondents* from *12 countries* in 2019 and *24,628 representative respondents* from the same countries in 2020. Both datasets contains the following variables:

- uid is a unique identifier per survey respondent
- country holds the country code from which the respondent originates (one of 'UK' (United Kingdom), 'US' (United States of America), 'DE' (Germany), 'AT' (Austria), 'NL' (Netherlands), 'NO' (Norway), 'KR' (South Korea), 'ES' (Spain), 'AU' (Australia), 'JP' (Japan), 'BR' (Brazil), 'RO' (Romania))
- weight is the country-specific representative weight as assigned by YouGov; this variable is *very important to consider* in order not to distort results

- gender holds the answer to the question "Which gender a respondent identifies with" (different language versions apply), allowing for 'f', 'm', and other
- age holds the answer to the question "What age a respondent has" (the survey was conducted in January/February 2019)
- education holds the answer to the question "What is your highest level of education? If you are currently in full-time education please put your highest qualification to date" allowing for country-specific answers pointing to one of 'none', 'early childhood', 'primary', 'lower secondary', 'upper secondary', 'post secondary', 'short-cycle tertiary', 'bachelors or equivalent', 'masters or equivalent', 'doctoral or equivalent'
- income holds the answer to the question "Household income" as country-specified sums translated into one of the following country-unspecific levels: 'low', 'medium', 'high'
- use_internet_general holds the answer to the question "How often do you access the Internet for any purpose (i.e. for work/leisure etc.)? This should include access from any device (desktop, laptop, tablet or mobile) and from any location (home, work, internet café or any other location)", allowing for one of '10+ times a day', '6-10 times a day', '2-5 times a day', 'once a day', '4-6 days a week', '2-3 days a week', 'once a week', 'less than once a week', 'don't know'
- use_news_general holds the answer to the question "Typically, how often do you access news? By news we mean national, international, regional/local news and other topical events accessed via any platform (radio, TV, newspaper or online)" (same response levels as for use_internet_general)
- use_news_main holds the answer to the question "You say you've used these sources of news in the last week , which would you say is your MAIN source of news?" allowing for 'Television news bulletins or programmes', '24 hour news television channels', 'Radio news programmes or bulletins', 'Printed Newspapers', 'Printed Magazines', 'Websites/apps of Newspapers', 'Websites/apps of news magazines', 'Websites/apps of TV and Radio companies', 'Websites/apps of other news outlets', 'Social media', 'Blogs', 'Online communities'
- use_news_avoidance holds the 2019 answer to the question "Do you find yourself actively trying to avoid news these days?" allowing for the following answers: 'often', 'sometimes', 'occasionally', 'never', 'don't know'; this question has not been posed in 2020
- use_news_worn_out holds the 2019 answer to the question "Please indicate your level of agreement with the following statement. 'I am worn out by the amount of news there is these days.' " with the possible answers 'strongly disagree', 'tend to disagree', 'neither agree nor disagree', 'tend to agree', 'strongly agree'; this question has not been posed in 2020
- use_news_tvshows thru use_news_none holds the answers to the questions "Have you used _____ in the last week as a source of news?" allowing for yes/no answers; note that the two items blogs and online communities have not been posed in 2020
- interest_in_news holds the answer to the question "How interested, if at all, would you say you are in news?" with the possible answers 'extremely interested', 'very interested', 'somewhat interested', 'not very interested', 'not at all interested', 'don't know'
- interest_in_politics holds the answer to the question "How interested, if at all, would you say you are in politics ?" (same response levels as for interest_in_news)
- political_orientation holds the answer to the question "Some people talk about 'left', 'right' and 'centre' to describe parties and politicians. (Generally, socialist parties would be considered 'left wing' whilst conservative parties would be considered 'right wing'). With this in mind, where would you position yourself?" with the possible response levels of 'very left-wing', 'fairly left-wing', 'slightly left-of-centre', 'centre', 'slightly right-of-centre', 'fairly right-wing', 'very right-wing', 'don't know'
- weekly_use_*id* is a set of questions, all adhering to the list of top brands per country (see the Reuters Digital News Report country portraits, such as Romania on page 104/105), for which people were presented with a list of national media and asked how often they used it on average, answers including (but not limited to) weekly and more than 3 days per week; the variables presented here report '1' (respondent answered to use this media outlet 'weekly') and '0' (respondent did answer otherwise); calculated weighted shares of these variables yield the light-orange part of the top-brands charts in the report; importantly, the last part of the variable name, the *id* refers to the *media_id* as specified in the *mediacloud* dataset
- heavier_use_*id* is very similar to weekly_use_*id* but reports '1' if a respondent said 'more than 3 days

per week'; calculated weighted shares of these variables yield the dark-orange part of the top-brands charts in the report

Media-Content Meta Data

The second datasets contain a data frame holding the metadata for *1.74 mio. online news items* from *76 different sources* (collected through MediaCloud) in 2019 as well as the metadata for *1.25 mio. online news items* from *81 different sources* in 2020. Articles were collected once per week over the course of one year, namely from September 2018 until (including) August 2019 for the 2019 dataset and from September 2019 until (including) August 2020 for the 2020 dataset. Meta data includes the following:

- stories_id holds the internal Mediacloud ID for the story and also references to the respective texts (third file)
- processed_stories_id is an incremental internal Mediacloud ID to help paginate through the results; typically not needed outside the Mediacloud API's
- collect_date represents the date when MediaCloud actually fetched the story
- guid is a generally acknowledged story ID that is being replaced by the story's URL if it is empty
- title is the title of a story as published through the RSS feed (i.e., no A/B test checking)
- publish_date is the story's publish date as maintained by the media outlet through meta information
- url is the story's original URL and also the link to the Crowdtagle dataset (fourth file)
- language represents the language of the story as detected by the chromium compact language detector library inside Mediacloud
- ap_syncidated indicates whether Mediacloud's detection algorithm thinks that this is an English language syndicated AP story
- media_id is the internal Mediacloud ID for the media source to which the story belongs; this is referenced in the first Reuters-DNI dataset
- media_name is the name of the media source to which the story belongs
- media_url is the main URL of the media source to which the story belongs

Actual Media Content

The third datasets contain the texts of these story items in the form of *Quanteda DFMs*. That is, neither Mediacloud nor we have the rights to publish original stories at this point; however, document-feature matrices are not the stories as one cannot reproduce the original stories from these DFM's, particularly since Mediacloud has already removed stopwords. Analysis herewith is thus limited, of course, as this is bag-of-words data only. The variable is a large list (very large, be careful!) with Quanteda DFMs on each position. You can identify individual stories through their stories_id as presented in the second file, for example through `mediacloud.wordmatrix2019[[1]]["969897634"]` where 969897634 is the stories_id.

Engagement Metrics

The fourth set of files contains the associated metadata of *1.71 mio. Facebook posts* from 2019 and *2.29 mio. Facebook posts* from 2020 that reference one of the news items as collected from CrowdTangle. Since Facebook is rather selective and nifty with their data, this is the dataset to handle with most care. It is built on public (!) posts of comparably influential users (i.e., media outlets, NGO's, politicians) only. The dataset includes a column `link` which refers to Mediacloud's `url`.

Load libraries

```
library(tidyverse)
library(quanteda)
library(quanteda.textplots)
library(quanteda.textstats)
library(scales)
```

```
library(corrgram)
library(lubridate)
theme_set(theme_minimal())
```

Load data

```
dni2019 <- read_rds('usenews.reutersdni.2019.rds')
dni2020 <- read_rds('usenews.reutersdni.2020.rds')
mediacloud2019 <- read_rds('usenews.mediacloud.2019.rds')
mediacloud2020 <- read_rds('usenews.mediacloud.2020.rds')
mediacloud.wordmatrix2019 <- read_rds('usenews.mediacloud.wm.2019.rds')
mediacloud.wordmatrix2020 <- read_rds('usenews.mediacloud.wm.2020.rds')
crowdtangle2019 <- read_rds('usenews.crowdtangle.2019.rds')
crowdtangle2020 <- read_rds('usenews.crowdtangle.2020.rds')
```

Generate basic statistics

```
# total survey respondents from 2019
dni2019 %>%
  count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1 24190
```

```
# the same for 2020
dni2020 %>%
  count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1 24628
```

```
# country count 2019
dni2019 %>%
  distinct(country) %>%
  count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1    12
```

```
# total mc items in 2019
mediacloud2019 %>%
  count()
```

```
##         n
## 1 1740761
```

```
# first/last article publication as per 2019 data
mediacloud2019 %>%
```

```
  summarise(min(publish_date),
            max(publish_date),
            .groups = 'drop')
```

```
##   min(publish_date) max(publish_date)
## 1        2018-09-01        2019-08-31
```

```
# total ct items in 2020
crowdtangle2020 %>%
  count()
```

```
## # A tibble: 1 x 1
##         n
##     <int>
## 1 2294819
```

```
# token count 2019
mediacloud.wordmatrix2019 %>%
  map(ntoken) %>%
  flatten_int() %>%
  sum()
```

```
## [1] 299310424
```

```
# media sources 2020
mediacloud2020 %>%
  distinct(media_id) %>%
  count()
```

```
##    n
## 1 81
```

```
# fb users 2019
crowdtangle2019 %>%
  distinct(account.id) %>%
  count()
```

```
## # A tibble: 1 x 1
##        n
##    <int>
## 1 148527
```

```
# total likes/shares/comments 2020
crowdtangle2020 %>%
  summarise(sum(statistics.actual.likeCount),
            sum(statistics.actual.shareCount),
            sum(statistics.actual.commentCount),
            .groups = 'drop')
```

```
## # A tibble: 1 x 3
##   `sum(statistics.actual.likeCount)` `sum(statistics.actual.shareCount)` `sum(statistics.actual.comme
##                                <int>                               <int>
## 1                          221513783                            93679394
```

Get the top-brands list from the Reuters Digital News Report for the Netherlands 2019

```
dni2019 %>%
  filter(country == 'NL') %>%
  mutate(n = n()) %>%
  mutate_at(vars(weekly_use_1:weekly_use_83354),
            ~.*weight) %>%
  summarise_at(vars(weekly_use_1:weekly_use_83354),
               ~100*sum(.)/first(n)) %>%
  pivot_longer(weekly_use_1:weekly_use_83354,
               names_to = 'media_id',
               values_to = 'share') %>%
  filter(share > 0) %>%
  mutate(media_id = as.integer(str_split_fixed(media_id, '_', 3)[,3])) %>%
  left_join(mediacloud2019 %>%
              select(media_id, media_name) %>%
              distinct(),
            by = 'media_id') %>%
  arrange(desc(share)) %>%
  print()
```

```
## # A tibble: 7 x 3
##   media_id share media_name
##      <int> <dbl> <chr>
## 1    55612 23.1  telegraaf.nl
## 2   119661 17.2  rtlnieuws.nl
## 3   623515  6.96 MSN - Brazil (Portuguese)
## 4     1094  4.25 BBC
## 5     1095  3.40 CNN
## 6   751082  2.28 Yahoo News - Latest News & Headlines
## 7    27502  2.05 Huffington Post
```

Determine which DFM is associated with which medium

```
mediacloud2019 %>%
  group_by(media_name) %>%
  summarise(n = n(),
            .groups = 'drop') %>%
  print()
```

```
## # A tibble: 76 x 2
##    media_name                          n
##    <chr>                           <int>
##  1 20minutos                       16923
##  2 9 News                           9505
##  3 abc                             52000
##  4 abcnyheter.no                      10
##  5 adevarul                        44889
##  6 aftenposten                     16627
##  7 antena3                          4192
##  8 antena3noticias                 22690
```

```
##  9 au.yahoo.com                          2
## 10 Australian Broadcast Company (ABC) 23654
## # ... with 66 more rows
```

```
# to include survey data (e.g., share of female weekly users among all weekly users), add a join
# note that since some outlets (e.g., HuffingtonPost) are consumed in several countries, more entries a
mediacloud2019 %>%
  group_by(media_name, media_id) %>%
  summarise(n = n(),
            .groups = 'drop') %>%
  left_join(dni2019 %>%
              pivot_longer(starts_with('weekly_use_')) %>%
              filter(value > 0) %>%
              mutate(media_id = as.integer(str_split_fixed(name, '_', 3)[,3])) %>%
              group_by(country, media_id) %>%
              summarise(female_weekly_users = sum(if_else(gender == 'f', weight, 0)),
                        weekly_users = sum(weight),
                        share = round(100*female_weekly_users/weekly_users),
                        .groups = 'drop') %>%
              select(media_id, country, female_weekly_users = share),
            by = 'media_id') %>%
  print()
```

```
## # A tibble: 137 x 5
##    media_name      media_id     n country female_weekly_users
##    <chr>              <int> <int> <fct>                 <dbl>
##  1 20minutos          40499 16923 ES                       54
##  2 9 News             68328  9505 AU                       52
##  3 abc                39848 52000 ES                       43
##  4 abcnyheter.no     123356    10 NO                       46
##  5 adevarul           39952 44889 RO                       46
##  6 aftenposten        41062 16627 NO                       46
##  7 antena3            41653  4192 ES                       57
##  8 antena3            41653  4192 RO                       43
##  9 antena3noticias    41255 22690 ES                       57
## 10 antena3noticias    41255 22690 RO                       43
## # ... with 127 more rows
```

Get all words in a specific document by stories_id

```
mediacloud.wordmatrix2019[[1]] %>%
  dfm_subset(stories_id == '969897634') %>%
  dfm_trim(min_termfreq = 1) %>%
  convert(to = 'data.frame')
```

```
##      doc_id correction sports mikita broadcast acetate misstated shellac bootleg kaepernick offensive
## 1 969897634          1      1      1         1       1         2       1       1          1         1
##   slovak savory stan collusion williams obituary misidentified damaging compensatory
## 1      1      1    1         1        1        1             1        2            2
```

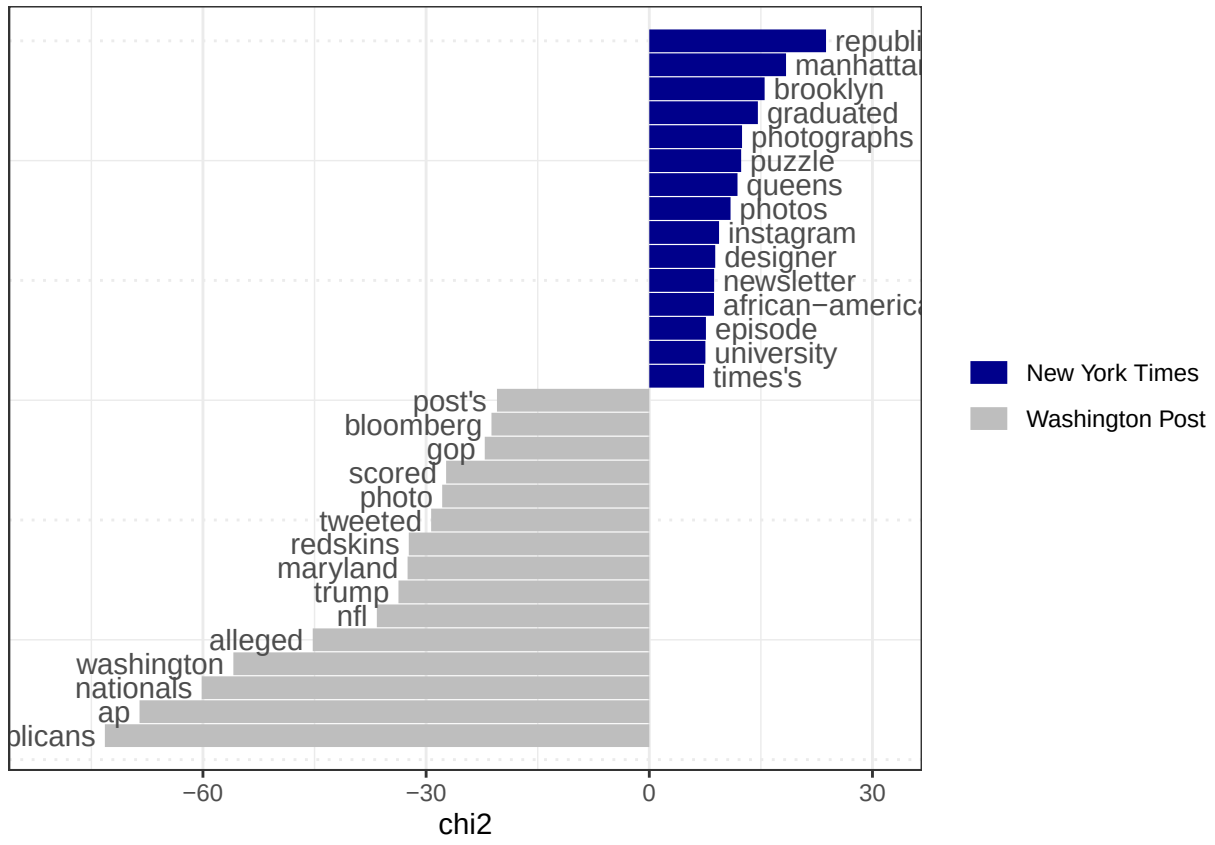
Compare word use in the *New York Times* (#1) and the *Washington Post* (#21)

```
nyt <-
  mediacloud.wordmatrix2019[[1]] %>%
  dfm_weight(scheme = 'prop')

wapo <-
  mediacloud.wordmatrix2019[[21]] %>%
  dfm_weight(scheme = 'prop')

usa <- rbind(nyt, wapo)
docvars(usa) <- rbind(docvars(nyt), docvars(wapo))

usa %>%
  dfm_group(groups = docvars(usa, 'media_name')) %>%
  textstat_keyness(target = 'New York Times') %>%
  textplot_keyness(n = 15)
```



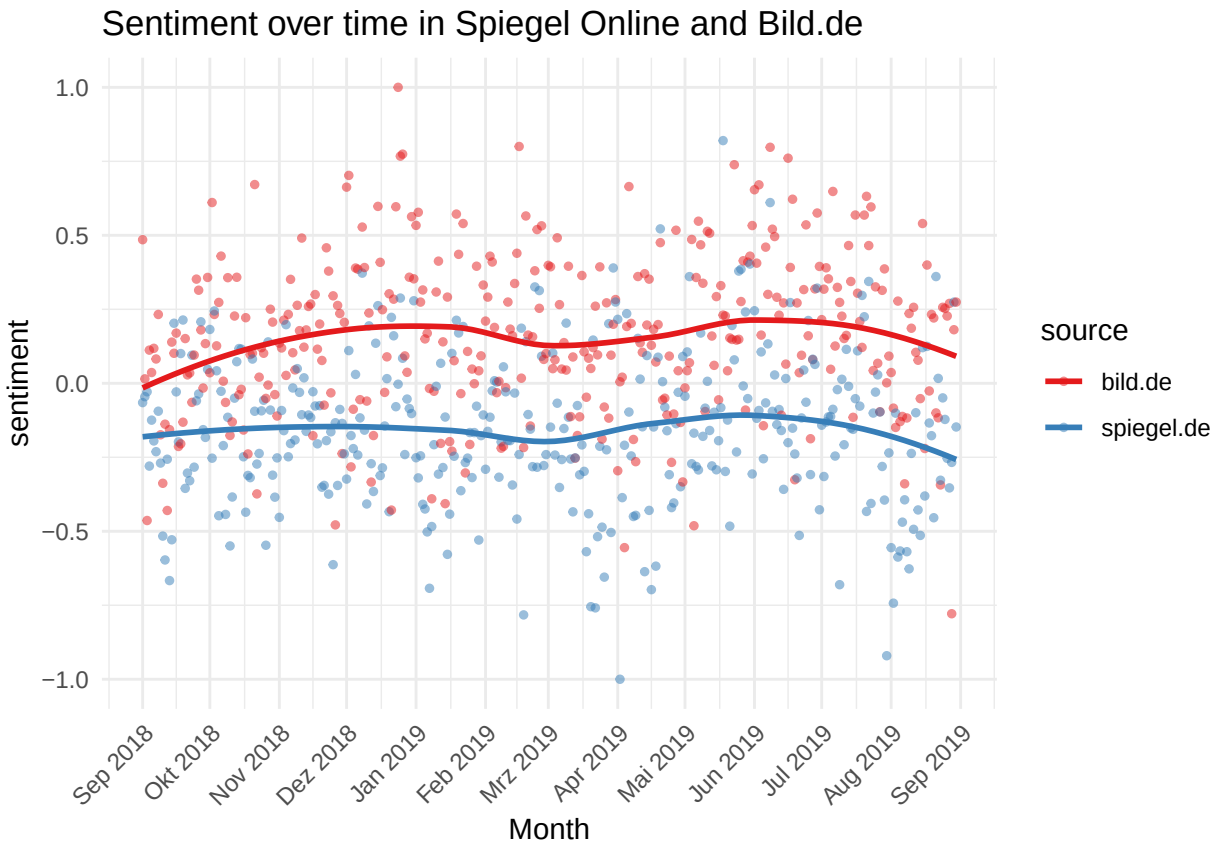
Compare sentiment in *Der Spiegel* (#20) and *Bild.de* (#26)

```
load('sentiWS.RData')
dict.senti <- dictionary(list(positive = positive.woerter.senti,
                              negative = negative.woerter.senti))
```

```
spiegel <- dfm_compress(mediacloud.wordmatrix2019[[20]])
bild <- dfm_compress(mediacloud.wordmatrix2019[[26]])

de <- rbind(spiegel, bild)
docvars(de) <- rbind(docvars(spiegel), docvars(bild))

de %>%
  dfm_lookup(dictionary = dict.senti) %>%
  dfm_group(groups = (docvars(de, c('publish_date', 'media_name')) %>%
                        mutate(grp = paste(publish_date, media_name)) %>%
                        pull(grp))) %>%
  dfm_weight(scheme = 'prop') %>%
  convert(to = 'data.frame') %>%
  mutate(sentiment = rescale(positive, to = c(-1,1)),
         published = as.Date(str_sub(doc_id, 1, 10)),
         source = str_sub(doc_id, 12)) %>%
  ggplot(aes(x = published,
             y = sentiment,
             color = source,
             group = source)) +
  geom_point(size = 1, alpha = 0.5) +
  geom_smooth(se = FALSE,
              method = 'loess',
              formula = y ~ x) +
  scale_colour_brewer(palette = 'Set1') +
  scale_x_date('Month', date_breaks = '1 month', date_labels = '%b %Y') +
  ggtitle('Sentiment over time in Spiegel Online and Bild.de') +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Who are some of the most active actors sharing news in 2020?

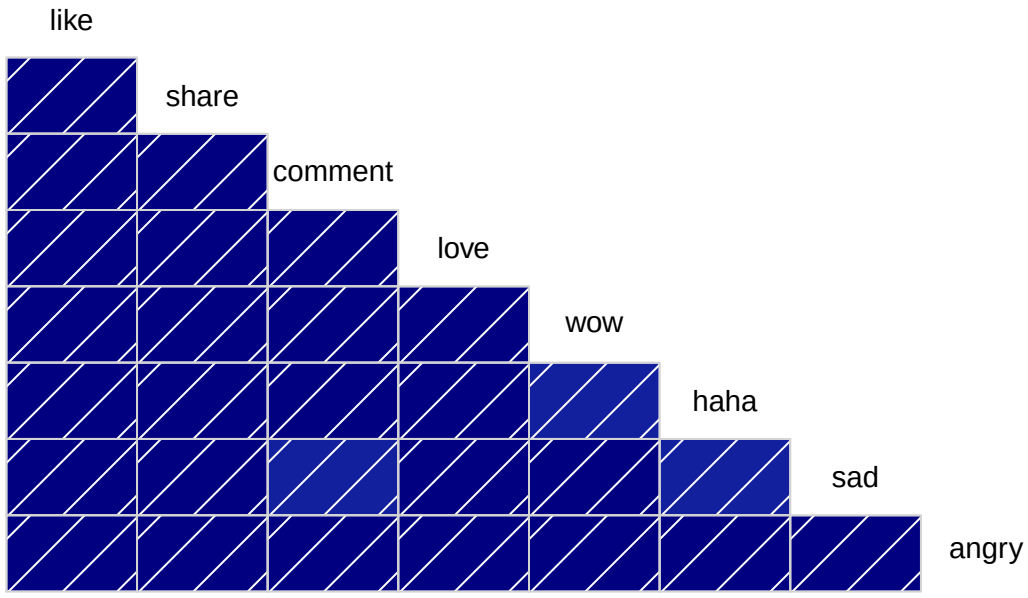
```
crowdtangle2020 %>%
  group_by(account.name) %>%
  summarise(subscriberCount = max(subscriberCount),
            posts = n(),
            like = sum(statistics.actual.likeCount),
            share = sum(statistics.actual.shareCount),
            comment = sum(statistics.actual.commentCount),
            love = sum(statistics.actual.loveCount),
            wow = sum(statistics.actual.wowCount),
            haha = sum(statistics.actual.hahaCount),
            sad = sum(statistics.actual.sadCount),
            angry = sum(statistics.actual.angryCount),
            .groups = 'drop') %>%
  arrange(desc(posts)) %>%
  print()
```

```
## # A tibble: 248,180 x 11
##    account.name       subscriberCount posts     like    share comment    love      wow     haha      sad
##    <chr>                        <int> <int>    <int>    <int>   <int>   <int>    <int>    <int>    <int>
##  1 The New York Times        17417864 13329 17251742 5519792 6389478 2566531 1552275 2050665 3654699
##  2 Adevarul                    737297 10015   548674  230213  213801   24683   22698   90274   87297
##  3 20minutos.es               1178896  9599  1006694 1098085  610555  134749  175156  160378  351825
##  4 The Guardian               8402573  8899  4702253 1573854 2200580  537548  301572  595817  881457
```

```
##  5 El Confidencial            1031809  8868   888094  883766  588525   84805   94824  135091  169115
##  6 ABC.es                     1572587  7926  1857263 2209620  912012  263316  177776  264638  412442
##  7 La Vanguardia              4313636  7423  2930704 2720089 1496810  380208  402565  257591  942242
##  8 UOL                        8146450  7201  3051412  940345 1159865  317841   86519  526137  476083
##  9 okdiario.com                844816  6655  1707615 2355713 1437439  194229   83781  408397  145195
## 10 HuffPost Politics          2108517  6478  2454636 1414161 1895439  379135  191772  662179  297267
## # ... with 248,170 more rows
```

How are different Facebook Reactions correlated?

```
crowdtangle2020 %>%
  group_by(account.name) %>%
  summarise(like = sum(statistics.actual.likeCount),
            share = sum(statistics.actual.shareCount),
            comment = sum(statistics.actual.commentCount),
            love = sum(statistics.actual.loveCount),
            wow = sum(statistics.actual.wowCount),
            haha = sum(statistics.actual.hahaCount),
            sad = sum(statistics.actual.sadCount),
            angry = sum(statistics.actual.angryCount),
            .groups = 'drop') %>%
  corrgram(order = NULL,
           lower.panel = panel.shade,
           upper.panel = NULL,
           text.panel = panel.txt,
           main = 'Correlation of different Facebook Reactions')
```



Compare the most frequent words in articles with angry reactions with all other articles

```
angriest_news <-
  crowdtangle2020 %>%
  filter(account.name %in% c('The Guardian', 'HuffPost')) %>%
  group_by(account.name, link) %>%
  summarise(angry = sum(statistics.actual.angryCount),
            .groups = 'drop')

mediacloud.wordmatrix2020[[32]] %>%
  dfm_subset(url %in% angriest_news$link) %>%
  topfeatures(n = 20)
```

```
##        worm         lbc      farage       nigel     epstein      andrew      prince       media    in
##           8           8           6           5           5           4           4           4
##      berman      tonsil     british      global   company's    protests investigate        fbi
##           3           3           2           2           2           2           2           2
```

```
mediacloud.wordmatrix2020[[27]] %>%
  dfm_subset(url %in% angriest_news$link) %>%
  topfeatures(n = 20)
```

```
##       ortberg      election         trump         elder    democratic         morse         menlo
##            26            24            21            15            14            14            14
```

```
##       mail-in massachusetts        pastor        lavery       ballots         tweet     degeneres
##            10             9             9             9             9             8             8
```

Select a single article from the New York Times

```
crowdtangle2019 %>%
  filter(str_detect(link, fixed('nytimes.com'))) %>%
  slice_sample(n = 20)
```

```
## # A tibble: 20 x 52
##          id platformId  platform date   updated  type  title   caption description    message    link
##       <dbl> <chr>       <chr>    <chr>  <chr>    <chr> <chr>   <chr>   <chr>          <chr>      <chr
##  1  6.73e10 1369565898~ Facebook 2019-~ 2019-12~ link  They'r~ nytime~ Among older w~ "Just FYI~ http
##  2  6.70e10 5363469031~ Facebook 2019-~ 2019-12~ link  Opinio~ nytime~ A septuagenar~ "A litera~ http
6.27                5
##  3  5.36e10 2768411356~ Facebook 2019-~ 2019-12~ link  A Coll~ nytime~ About 26,000 ~ "Betsy De~ http
##  4  6.14e10 1004337067~ Facebook 2019-~ 2019-07~ link  German~ nytime~ Mesut Ozil wh~  <NA>      http
10                  0
##  5  6.51e10 7307983751~ Facebook 2019-~ 2019-11~ link  7 Thin~ nytime~ Our guide to ~ "The New ~ http
0.706               2
##  6  6.34e10 1405837894~ Facebook 2019-~ 2019-07~ link  Southo~ nytime~ In this quiet~ "https://~ http
5.2                 3
##  7  4.28e10 3290221471~ Facebook 2018-~ 2019-12~ link  The Ne~ <NA>    Something ver~ "Kavli In~ http
1.95               41
##  8  6.86e10 1682269835~ Facebook 2019-~ 2019-08~ link  Opinio~ nytime~ The blame lie~ "This art~ http
6.3                 3
##  9  5.14e10 1773373712~ Facebook 2019-~ 2019-12~ link  A Bord~ nytime~ The A.C.L.U. ~  <NA>      http
1.94               11
## 10  5.78e10 1700045797~ Facebook 2019-~ 2019-12~ link  Opinio~ nytime~ The very rich~  <NA>      http
1.79               15
## 11  4.63e10 2093564483~ Facebook 2018-~ 2019-12~ link  U.S., ~ nytime~ The Trump adm~  <NA>      http
2.67                9
## 12  5.43e10 1052030531~ Facebook 2019-~ 2019-12~ link  Opinio~ nytime~ It happens th~ "#TellThe~ http
2.22                6
## 13  7.37e10 1730440387~ Facebook 2019-~ 2019-12~ link  Austra~ nytime~ A teenager in~  <NA>      http
2.5                 4
## 14  6.77e10 1218597270~ Facebook 2019-~ 2019-12~ link  A Quar~ nytime~ Around the wo~ "Around t~ http
## 15  6.69e10 5281959998~ Facebook 2019-~ 2019-12~ link  Baltim~ nytime~ When the pres~ "Presiden~ http
## 16  6.31e10 9531416060~ Facebook 2019-~ 2019-07~ link  Thomas~ nytime~ At TAK Room a~ "The retu~ http
1.4                18
## 17  7.37e10 3849347348~ Facebook 2018-~ 2019-12~ link  Opinio~ nytime~ Total victory~ "WE MUST ~ http
0.25                4
## 18  5.15e10 4109375123~ Facebook 2019-~ 2019-12~ link  'It Is~ nytime~ The crisis ov~ "The Cath~ http
8.55               28
## 19  5.34e10 1945991268~ Facebook 2019-~ 2019-12~ link  Opinio~ nytime~ The Minnesota~ "THIS <U+27A1>
## 20  6.19e10 4082530660~ Facebook 2019-~ 2019-12~ link  18 Ske~ nytime~ The human rem~ "https://~ http
## # ... with 37 more variables: statistics.actual.shareCount <int>, statistics.actual.commentCount <int
## #   statistics.actual.wowCount <int>, statistics.actual.hahaCount <int>, statistics.actual.sadCount
## #   statistics.actual.thankfulCount <int>, statistics.expected.likeCount <int>, statistics.expected.
## #   statistics.expected.commentCount <int>, statistics.expected.loveCount <int>, statistics.expected
## #   statistics.expected.sadCount <int>, statistics.expected.angryCount <int>, statistics.expected.tha
## #   account.name <chr>, account.handle <chr>, account.profileImage <chr>, account.subscriberCount <in
## #   account.platformId <chr>, account.verified <lgl>, videoLengthMS <int>, brandedContentSponsor.id
```

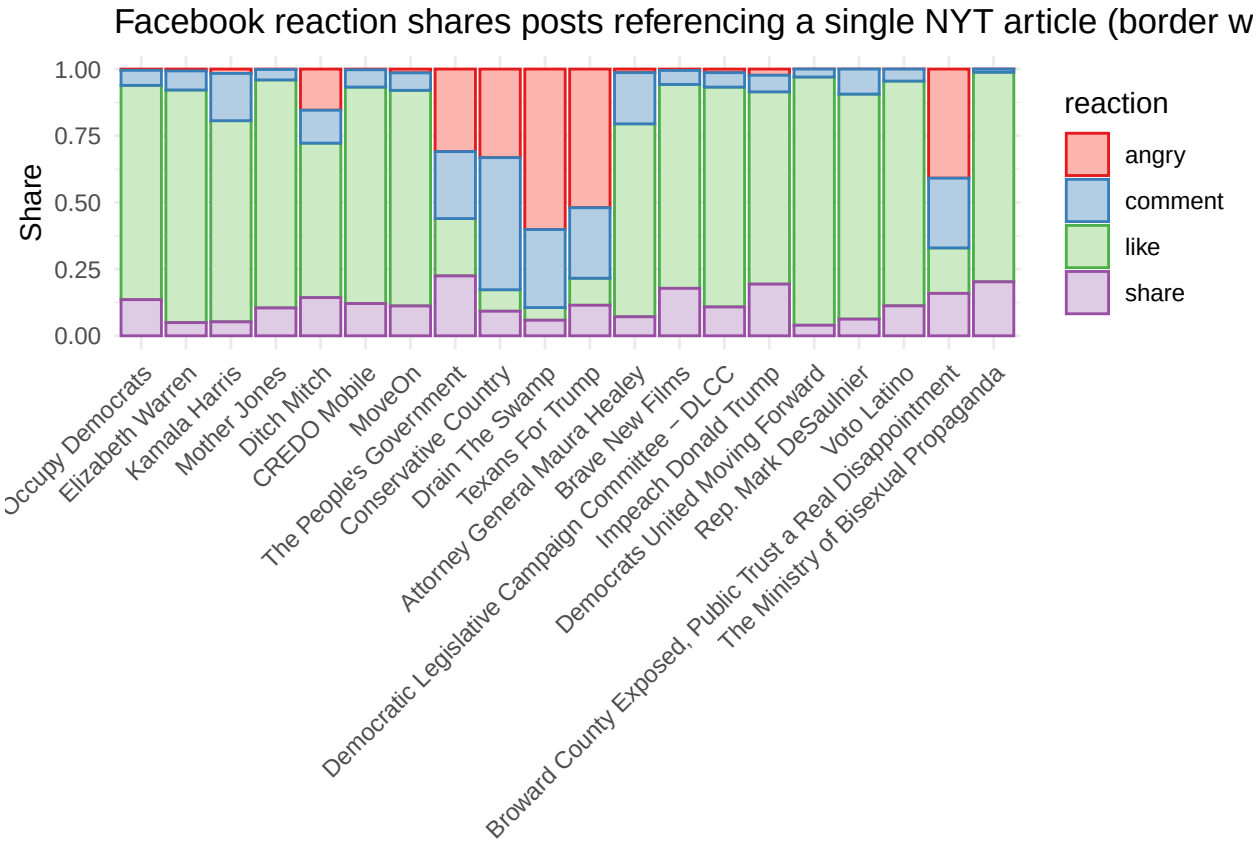
```
## #   brandedContentSponsor.handle <chr>, brandedContentSponsor.profileImage <chr>, brandedContentSpons
## #   brandedContentSponsor.url <chr>, brandedContentSponsor.platform <chr>, brandedContentSponsor.pla
## #   liveVideoStatus <chr>
```

```
url.lawsuits <- 'https://www.nytimes.com/2019/02/18/us/politics/national-emergency-lawsuits-trump.html'
mediacloud2019 %>%
  filter(url == url.lawsuits) %>%
  print()
```

```
##   stories_id processed_stories_id collect_date
## 1 1201232491           1492340774   2019-02-18 https://www.nytimes.com/2019/02/18/us/politics/nationa
emergency-lawsuits-trump.html
##                                                                                 title publish_date
## 1 16 States Sue to Stop Trump's Use of Emergency Powers to Build Border Wall   2019-
02-19
##                                                                                                 url language ap_s
## 1 https://www.nytimes.com/2019/02/18/us/politics/national-emergency-lawsuits-trump.html       en
```

Select FB posts referencing the article and plot reactions

```
crowdtangle2019 %>%
  filter(link == url.lawsuits) %>%
  select(account.name,
         like = statistics.actual.likeCount,
         angry = statistics.actual.angryCount,
         share = statistics.actual.shareCount,
         comment = statistics.actual.commentCount) %>%
  mutate(total = like + angry + share + comment,
         like = like/total,
         angry = angry/total,
         share = share/total,
         comment = comment/total) %>%
  slice_max(total, n = 20) %>%
  mutate(rank = row_number()) %>%
  pivot_longer(like:comment,
               names_to = 'reaction',
               values_to = 'count') %>%
  ggplot(aes(x = reorder(account.name, rank),
             y = count,
             color = reaction,
             fill = reaction)) +
  geom_col() +
  scale_x_discrete(NULL) +
  scale_y_continuous('Share') +
  scale_color_brewer(palette = 'Set1') +
  scale_fill_brewer(palette = 'Pastel1') +
  ggtitle('Facebook reaction shares posts referencing a single NYT article (border wall)') +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



What is the mean number of angry reactions for a given source?

```
crowdtangle2020 %>%
  mutate(outlet = case_when(str_detect(link, fixed('spiegel.de')) ~ 'spiegel.de',
                            str_detect(link, fixed('bild.de')) ~ 'bild.de',
                            TRUE ~ NA_character_)) %>%
  filter(!is.na(outlet)) %>%
  group_by(outlet) %>%
  summarise(mean_angry = mean(statistics.actual.angryCount),
            .groups = 'drop')
```

```
## # A tibble: 2 x 2
##   outlet     mean_angry
##   <chr>           <dbl>
## 1 bild.de          36.1
## 2 spiegel.de       13.3
```