MSDS - Data Wrangling - Final Project Report

Devis Shehu

5/5/2022

Introduction

This report summarize the key concepts and features from the Final Project for the Data Wrangling class, part of the MSDS program at Rutgers.

For this project, explored collecting, parsing and processing news articles from Google Search and News API. This deliverable can also been seen as a 1st effort to create a project GutenbergeR for News. Project places a heavy emphasis on data wrangling (parsing, processing) and data exploration. Analysis was limited given the time constraints.

Also included is Shiny app and visualization as well as **SpacyR** package use to extract root words, entities and some noun-phrases. This package is useful for extracting additional context and meaning beyond the classical text mining techniques. The Shiny app can be found under "app" sub-directory.

The goals and high level architecture are documented in the *./final_project_presentation.pdf*. Essentially this is v0.0.1 of building a knowledge base from a collection of documents.

A secondary goal is to reinforce learnings from Data Wrangling project including transforming data into Tidy format, using Tidy Text techniques such as Sentiment Analysis and Topic Modeling, Web scraping using RVest and RSelenium, Reactive App development with Shiny, etc.

To facilitate the portability of the Shiny app and this markdown file, the pre-processed meta data and datasets have been uploaded to a public S3 bucket.

https://s3.amazonaws.com/s3.msds.rutgers.org/data_wrangling/

The processed data can also be downloaded from Github, see the *./data/* dir.

https://github.com/d-shehu/msds-data-wrangling-final-project

Dependencies and Packages

In addition to the packages listed above, which are required by the pre-processing scripts, this project uses **R Selenium** and **R Vest**. See *scripts* subdir.

The data collection and scraping requires running Dockerized containers on Ubuntu 20.04 or comparable Debian distro. The script for building and running the image is in *./build.sh* and *./run.sh*. It should also be possible to run the *./collector.r* and *./process.r* from R CLI or using Rscript. The latter is in fact how Docker container invokes the code. This script assumes dependencies are installed.

Package Installation:

install.packages(c('digest', 'dplyr', 'lubridate', 'httr', 'RCurl', 'rvest', 'RSelenium', 'textdata', 'tidyr', 'tidytext', 'tokenizers', 'urltools', 'uuid'));

install.packages('arrow', repos = 'https://packagemanager.rstudio.com/all/__linux__/focal/latest');

Data Collection

As shown in the *final_project_presentation*, there are a series of scripted and semi-manual steps that need to be run in order to collect and format the data. There are 2 logic paths, Google news and News API. The former lives in *scripts/googleNews.r* while the latter is in *newsAPI.r*.

Google News

Processing consists of a series of functions. Using RSelenium, Google Search APIs are invoked for the given search terms, eg. *Ukraine + War* and **time range**. For this project, this spanned from end of January, just prior to the start of the war, to mid April, near the completion of the project.

However, the logic is generalized and can work with any search term and timeframe. Articles are limited to English speaking sources which simplifies text mining and semantic processing. To minimize issues with Google's throttling, the code invokes Google search per day in the given range, pulling up to 10 pages of search results via Google's pagination at a time (~100 articles).

Throttling and IP Blocking Despite the built-in delays and throttling in the process, from time to time, it was observed that Google would block requests unexpectedly, especially if the program pulled more than 5 days of results consecutively and/or was run for more than 10-15 minutes. So an additional cooldown period of 30 minutes was introduced after each batch.

Short-term blocking of IPs was nevertheless observed with Google aswell NY Times which required even more cooldowns. These along with memory issues with R Selenium complicated data collection and slowed down early development.

Basic Parsing For each article, it was necessary to use Curl util to get the source's original URL from Google's redirect ULR. Article title had to be parsed using rvest scanning for token **.yuRUbf** in HREF tag's CSS class. Some additional processing was needed to extract the source from the original URL and to convert the links into a **tidy** dataframe. Each article was assigned a dynamically generated uuid which is guaranteed to be unique and globally consistent. uuid is used to reference an article's raw data (HTML) and in subsequent operations such as joins.

Once the links were obtained, a separate collector process is run for Google news as well as News API. *./collector.r*. The collector, takes the URL and uses Selenium to fetch the HTML file and store it in the *parsed* director. This is useful in case the article is later altered, moved or otherwise becomes unavailable. It also helps speed up development, testing, and cuts down on throttling.

News API

For **NewsAPI.org**, obtained an API key for a free account. The free account limits user to 1 month of data use getForm and 100 requests per day. See URL below for more details. API key will be provided with project deliverable.

https://newsapi.org/pricing

Throttling and IP blocking was not an issue with this API. However, the 1 month limit required periodic fetching to minimize gaps in the data. The logic can be seen in *./scripts/newsAPI.r*. The flow is as follows

- Call API using RCurl's *getForm* function. Extract available meta data including source, URL, author, date published. Up to 100 articles per day were fetched which was the maximum allowed. Language code was also specified.
- Convert meta to data frame. See below for data format. This was straightforward
  unlike with Google news. However, additional work was needed to extract author. Duplicates (titles) were removed during processing.

Cleaning up Authors For some newsAPI sources, the author was given. In these cases, the logic as shown in the function **fnCleanupAuthors**, attempted to extract a name. However, the data in newsAPI is inconsistent and/or contains error. In addition to various special characters, phone numbers and emails there is a great deal of formatting inconsistency and bogus information.

Some sources often attribute authorship to 3rd parties, such as AP/Wire. While others may simply indicate the parent news organization as the originator. Common example is with national and local affiliates. In such cases, the code will note that the article has been attributed to another sources.

There are many differences in spellings as well as generic entries such as *staff*, or *editorial board*. The author field is quite convoluted and messy and given the many gaps seems less than useful at this time. However, logic makes a best effort to handle as many special cases as possible and some useful data was extracted.

One potential improvement would be to scan through the bylines of the HTML metadata to extract the author. Many sources, such as CNN, use a byline attribute or tag. However, the coding is not always consistent in bylines and would require significant work to research and harmonize. Bylines also have similar issues as observed in the NewsAPI author meta data. Though this would allow for more articles to be assigned to specific authors which might yield insights (top creators, associations of authors, nationalitys, organizations, etc.)

Data Format

There are 2 versions of the data set, **small** and **large**. The former is a sample of 1000 articles used during development. The latter is the full set of articles of ~10,000 articles. Each set is made up of a collection of files as shown below.

The file format is **Apache Parquet**, which is a *columnar format* that is convenient way to store data frames and can be used to fetch specific column without the need for iterating over rows. For the sake of simplicity and given the size of the data, the data frames are loaded into memory.

However, the performance of the app could be significantly improved with the addition of index in Parquet or by using database such as SQL Lite

https://blog.cloudera.com/speeding-up-select-queries-with-parquet-page-indexes/

Meta Data

Meta data is stored in files named _meta.parquet where the provider is google_news and news_api. data.r contains the data fetching logic and data filtering, by time-range and source.

The snippet below shows how to read the meta data and format. For this analysis the full dataset was used as it contains all 10K articles. Please note that in the Shiny app the user can select either small or large. Large data set can be quite slow with queries and visualizations taking up to 1 minute to load.

```
## Rows: 10,390
## Columns: 6
## $ id        <chr> "c6f5fb44-3cd6-4065-93b1-064ed8aff0d7", "1ec76405-b83f-4bb4-~
## $ title     <chr> "Ukraine crisis: Why is Germany out of step with the US, Eur~
## $ published <date> 2022-01-31, 2022-01-31, 2022-01-31, 2022-01-31, 2022-01-31,~
## $ source    <chr> "aljazeera", "bbc", "apnews", "bbc", "aljazeera", "reuters",~
## $ sourceURL <chr> "https://www.aljazeera.com/news/2022/1/31/ukraine-crisis-que~
## $ authors   <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
```

Data Set

Each data set is made of a series of data frames as shown below. The data frames were generated as part of the data collection process as defined in *./scripts/processNews.r*.

All Terms: the unique words from each of the articles. The data frame tracks article id, paragraph and sentence for each token. It's in Tidy format with the removal of stop words.

```
## # A tibble: 3 x 4
##   word        n id                                   published
##   <chr>   <int> <chr>                                <date>
## 1 germany    16 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 2 ukraine    16 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 3 russia     14 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
```

Also stored are the original **article text** which has been sanitized to cleanup stop words.

trigrams are stored separately. The 3-word pair was found to be most useful in extracting word-pair associations such as concepts such the *Nord Stream 2* pipeline and *Weak European Cohesion*.

Surprisingly, classical text mining techniques are able to surface many concepts from the corpus. So the original text is not used directly in the analysis or app.

```
## # A tibble: 5 x 4
##   trigram                     n id                                   published
##   <chr>                   <int> <chr>                                <date>
## 1 nord stream 2               6 c6f5fb44-3cd6-4065-93b1-064ed8aff0~ 2022-01-31
## 2 told al jazeera             5 c6f5fb44-3cd6-4065-93b1-064ed8aff0~ 2022-01-31
## 3 german foreign minister     2 c6f5fb44-3cd6-4065-93b1-064ed8aff0~ 2022-01-31
## 4 weakens european cohesion   2 c6f5fb44-3cd6-4065-93b1-064ed8aff0~ 2022-01-31
## 5 100,000 russian soldiers    1 c6f5fb44-3cd6-4065-93b1-064ed8aff0~ 2022-01-31
```

Stats and Semantics

Per article stats are also calculated during data processing which helps speed up the Shiny app and makes it easier to drill-down and filter on specific sources or time ranges.

Statistics can indicate if a source has contributed a significant amount to the corpus: e.g. total # of articles per source, (unique) words, etc. While this is not equivalent to unique or factually correct content, it's a 1st step in characterizing sources using objective measures.

```
## Rows: 9,032
## Columns: 11
## $ num_paragraphs    <int> 41, 51, 44, 53, 38, 53, 50, 15, 17, 21, 45, 36, 28, ~
## $ num_sentences     <int> 63, 62, 73, 68, 57, 82, 101, 16, 47, 49, 53, 55, 33,~
## $ num_words         <int> 685, 475, 693, 515, 516, 761, 895, 229, 422, 265, 39~
## $ num_unique_words  <int> 415, 342, 424, 403, 368, 496, 524, 178, 319, 153, 28~
## $ max_word_sentence <dbl> 30, 18, 32, 27, 21, 23, 30, 26, 34, 26, 22, 28, 23, ~
## $ avg_word_sentence <dbl> 10.873016, 7.661290, 9.493151, 7.573529, 9.052632, 9~
## $ sd_word_sentence  <dbl> 6.617406, 4.168357, 6.234143, 5.111638, 4.856590, 5.~
## $ max_word_len      <dbl> 15, 18, 15, 18, 15, 15, 14, 16, 16, 14, 18, 15, 14, ~
## $ avg_word_len      <dbl> 6.801460, 6.917895, 6.744589, 6.566990, 6.662791, 6.~
## $ sd_word_len       <dbl> 2.186147, 2.204616, 2.249532, 2.126559, 2.200320, 2.~
## $ id                <chr> "c6f5fb44-3cd6-4065-93b1-064ed8aff0d7", "1ec76405-b8~
```

And finally, there are several semantics dataframes extracted by using SpacyR. **terms**, **noun-phrases** and **entities**.

The latter include important details such as people, places, events and organizations. Given this enhanced data, it's possible to replace some of the simpler analysis based on words and trigrams and get even more meaningful results. This could be a future extension of the Shiny App where user is able to select either tidy text tokens or semantics and then run comparative, sentiment analysis and topic modeling.

```
## # A tibble: 6 x 4
##   word        n id                                   published
##   <chr>   <int> <chr>                                <date>
## 1 Germany    26 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 2 Ukraine    17 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 3 Russia     14 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 4 cookie     10 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 5 german     10 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 6 country     9 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
```

```
## # A tibble: 6 x 4
##   phrase          n id                                   published
##   <chr>       <int> <chr>                                <date>
## 1 Germany        18 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 2 Ukraine        15 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 3 Russia         14 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 4 you            10 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 5 it              8 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 6 the_country     8 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
```

```
## # A tibble: 6 x 5
## # Groups:   entity [6]
##   entity     entity_type     n id                                   published
##   <chr>      <chr>       <int> <chr>                                <date>
## 1 Germany    GPE            25 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 2 Ukraine    GPE            17 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 3 Russia     GPE            14 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 4 German     NORP           10 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 5 Al_Jazeera ORG             5 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 6 European   NORP            5 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
```

SpacyR entity types can be a bit confusing so a function has been provided to decypher the type. See below. Also note that noun-phrases do pick up some personal pronouns as well as individual words. This is mitigated in the R News Digest Shiny App through some additional filtering.

```
## Warning: The `x` argument of `as_tibble.matrix()` must have unique column names if `.name_repair` is
## Using compatibility `.name_repair`.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```

```
## # A tibble: 18 x 2
##    Type        Description
##    <chr>       <chr>
##  1 PERSON      People, including fictional.
##  2 NORP        Nationalities or religious or political groups.
```

```
##  3 FAC         Buildings, airports, highways, bridges, etc.
##  4 ORG         Companies, agencies, institutions, etc.
##  5 GPE         Countries, cities, states.
##  6 LOC         Non-GPE locations, mountain ranges, bodies of water.
##  7 PRODUCT     Objects, vehicles, foods, etc. (Not services.)
##  8 EVENT       Named hurricanes, battles, wars, sports events, etc.
##  9 WORK_OF_ART Titles of books, songs, etc.
## 10 LAW         Named documents made into laws.
## 11 LANGUAGE    Any named language.
## 12 DATE        Absolute or relative dates or periods.
## 13 TIME        Times smaller than a day.
## 14 PERCENT     Percentage, including "%".
## 15 MONEY       Monetary values, including unit.
## 16 QUANTITY    Measurements, as of weight or distance.
## 17 ORDINAL     "first", "second", etc.
## 18 CARDINAL    Numerals that do not fall under another type.
```

The Corpus

The set of articles is quite diverse with nearly 700 unique sources and 10,000 articles covering a time period of approximately 2 1/2 months. See stats.

```
## [1] "Number of unique sources: 679"
```

```
## [1] "Number of articles: 10390"
```

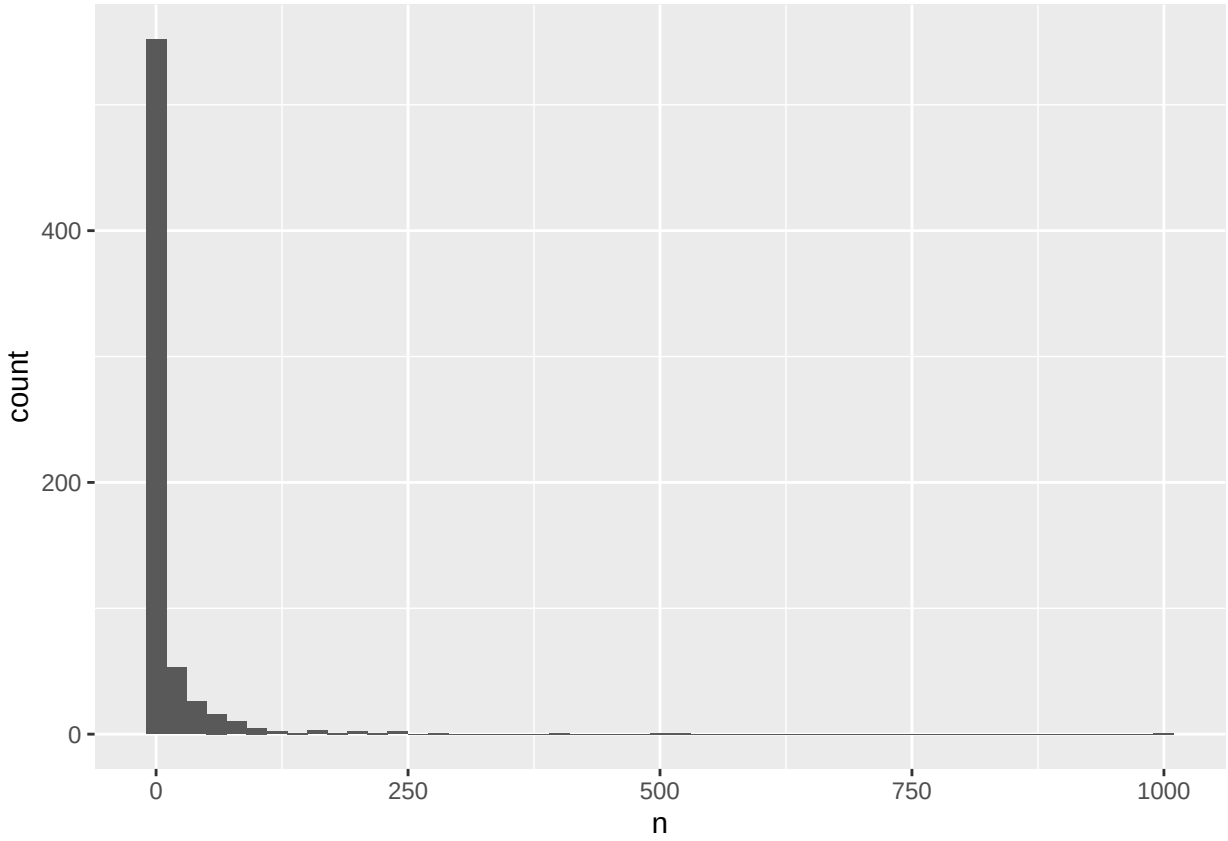
```
## [1] "From 2022-01-31 to 2022-04-16"
```

```
## [1] "Avg Article: # paragraphs:41.0834809565988, # words: 575.35119574845, # unique words: 344.78620
```

But only a handful of source contribute a significant number of articles on a consistent (daily, weekly) basis. This is shown by the histogram below. There is a huge disparity where most sources contribute only a couple of dozen articles while approximately 20 contributed hundreds each.

Given the time period, high visibility of this topic, and the 2 1/2 month period it's expected that any authoritative source should be contributing on the order of 100 or more articles per source. This corresponds to 1-2 articles published daily.

Note: did not observe a significant selection bias outside the expected results i.e. given the search criteria of English sources.




```
## [1] "Mean of # of articles per source: 15.3019145802651"
```

```
## [1] "Max of # of articles per source: 1009"
```

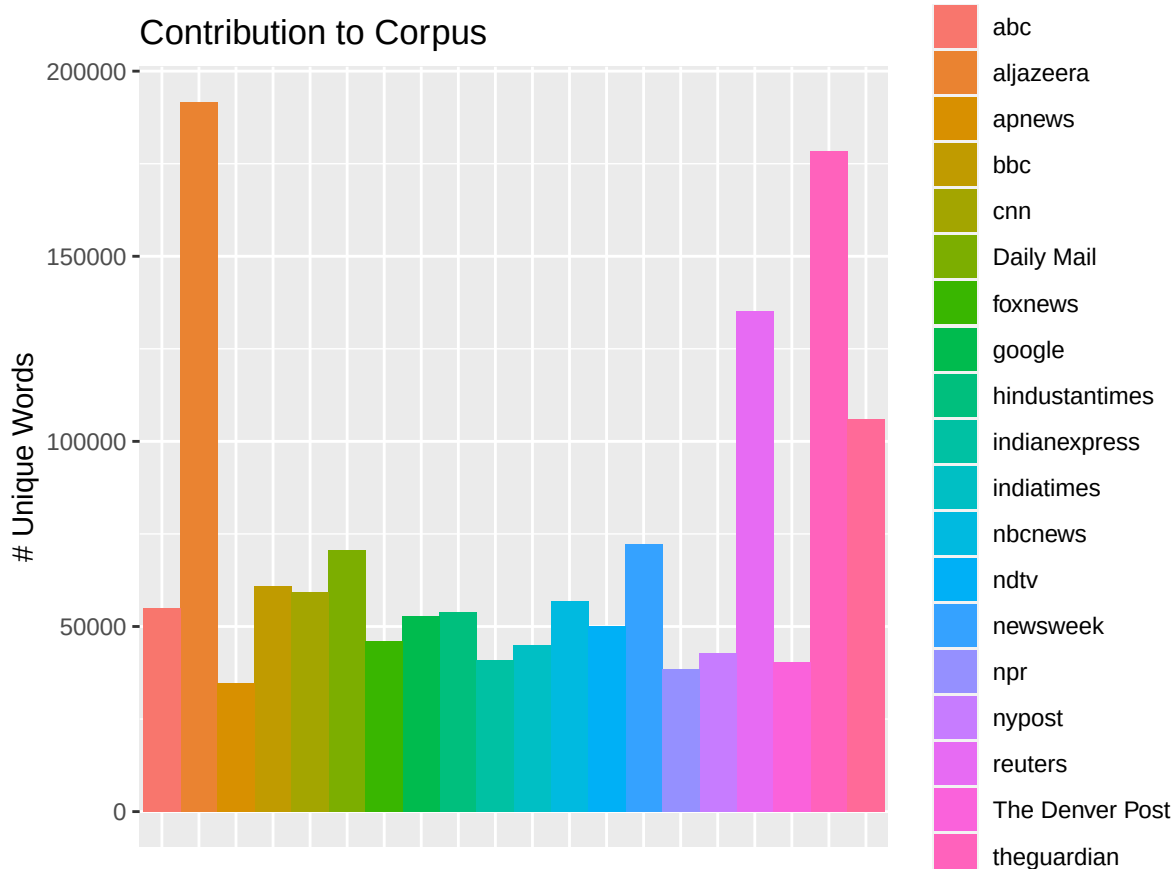
```
## [1] "Sources with > 100 articles:"
```

```
## [1] "abc, aljazeera, apnews, bbc, cnn, Daily Mail, Freerepublic.com, google, hindustantimes, Indepen
```

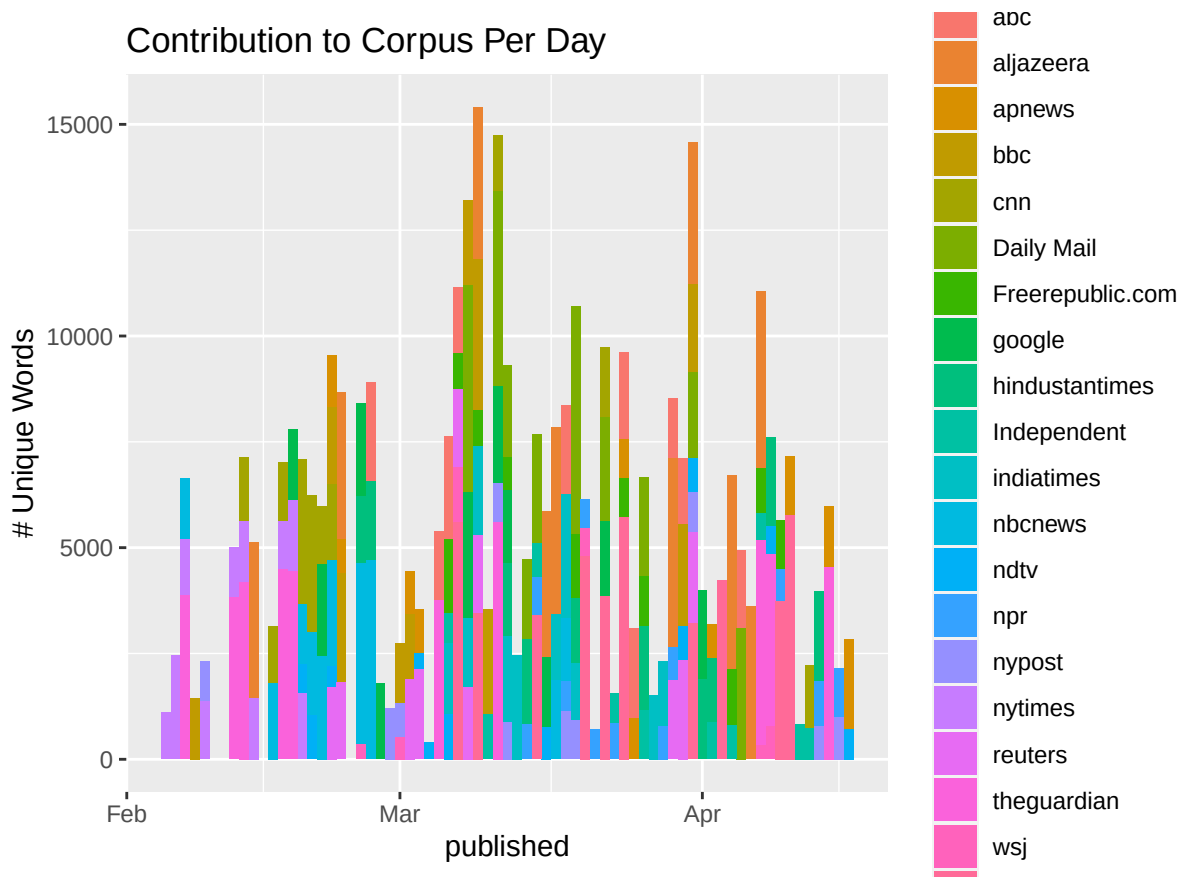
Using the *./app/summary.r* to plot top source and contribution over time. Note that the code is passing in the stats so the contribution reflect the total words and not just the article count.

A more sophisticated approach would drill down to the semantics and could look at the unique entities contribute by each source vs those found across the entire corpus. Unique entities could be verified by cross-referencing across 2 or more sources.

What's shown below is a simple way of isolating the most useful sources.




```
## `summarise()` has grouped output by 'source'. You can override using the
## `.groups` argument.
```

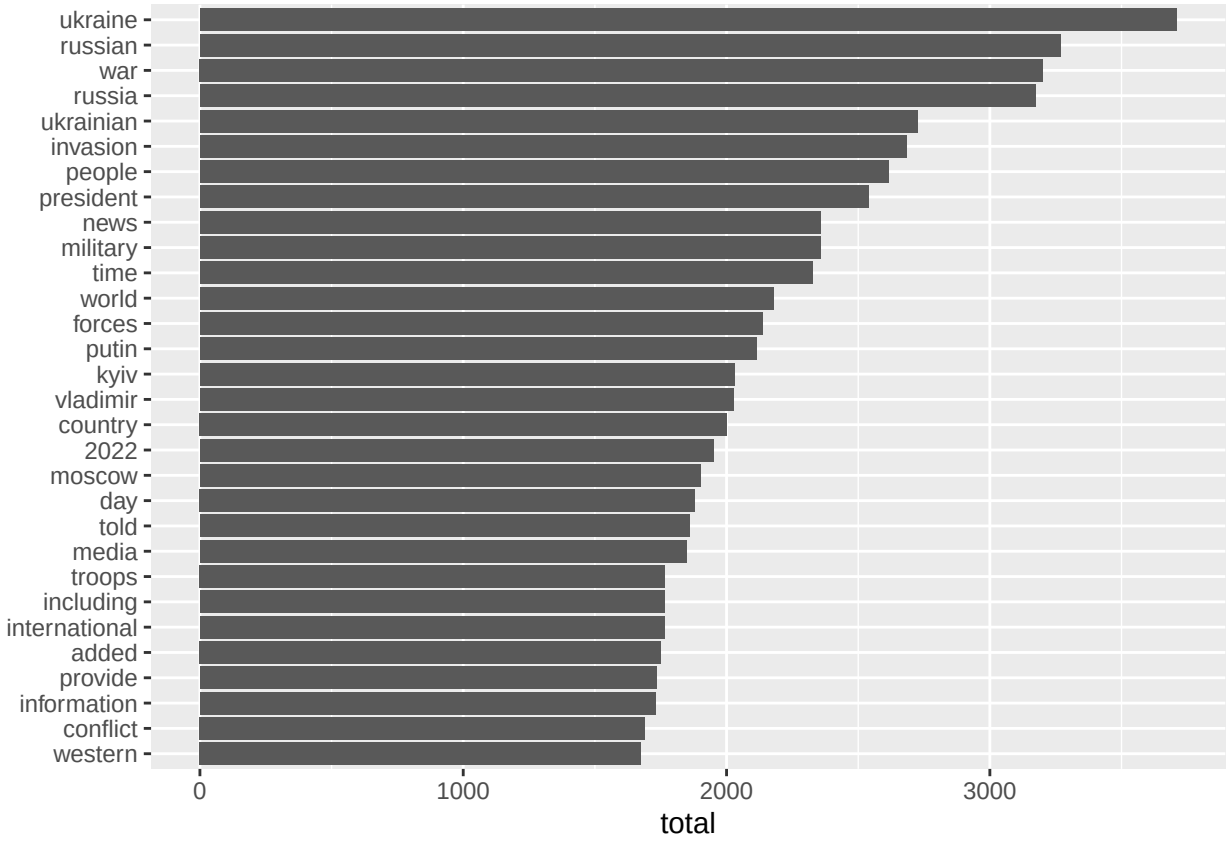


Extracting Terms & Concepts

Given the selection of sources, extract terms and concepts using text mining techniques from class. Please note that will be using the same functions as in the Shiny app when possible.

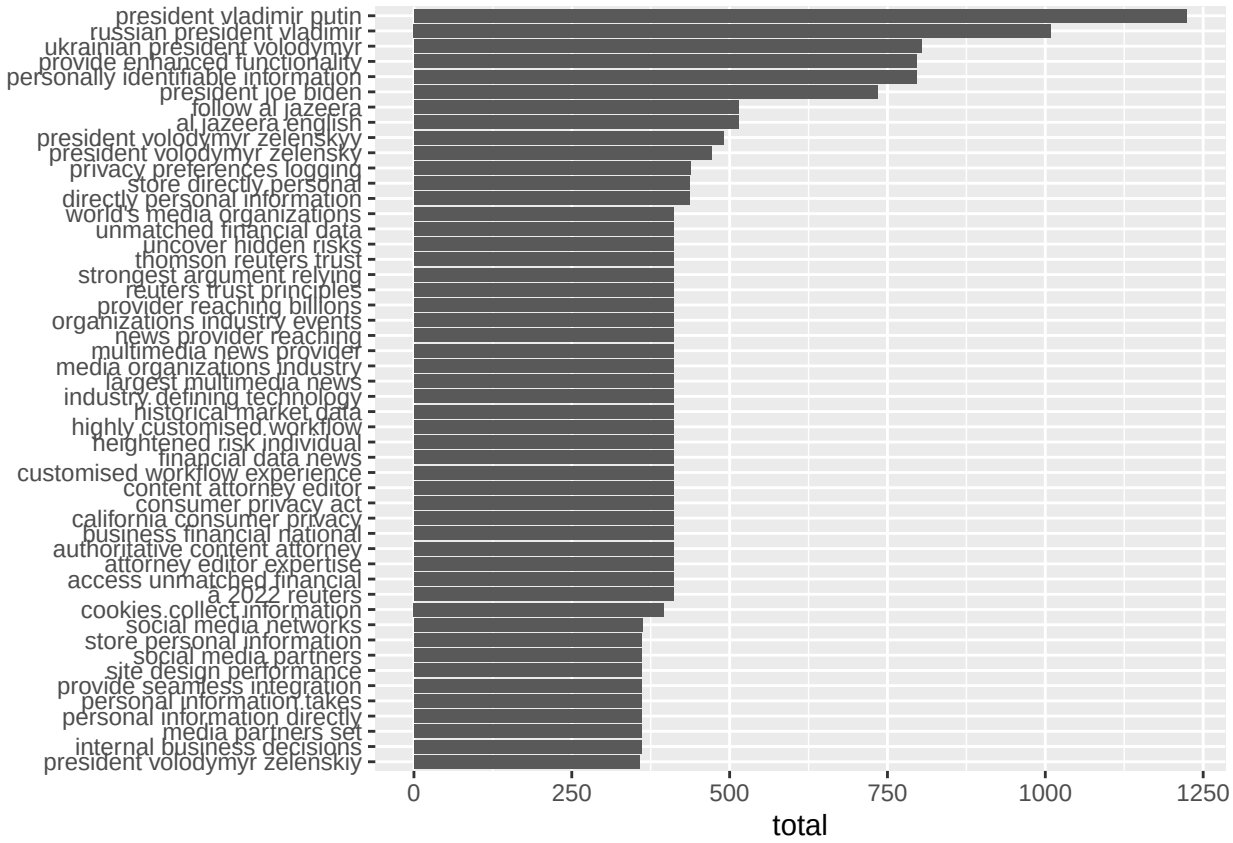
Given the topic what are the most common terms and ngrams over the entire period?

```
## `summarise()` has grouped output by 'word'. You can override using the `.groups`
## argument.
```



From a glance, some of the key players emerge such as NATO, the (US) President and (Vladimir) Putin. This list could be a good starting point for tags or as top-level categories for breaking up the corpus.

And the trigrams? Do they provide any additional context?



Observe that trigrams provide a clearer list of the players such as German Chancellor Olaf Scholz and French President Emmanuel Macron who have been involved in early diplomatic effort. Unfortunately, some irrelevant concepts sneak in such as *personally identifiable information* and *California Consumer Privacy* References to sources such as Al Jazeera and Reuters copyright can also be seen. These could be cleaned up with a bit more effort.

Also visible areglimpses of the progression, i.e. a rudimentary timeline. Early in the war, the diplomatic efforts dominated the news including the Munich conference.

References to 1917 and the Russian revolution highlight some of the historical context including Russia's losses in WW1 and the short-lived treaty of Brest-Litovsk as well the subsequent Polish-Soviet War which resulted in the partitioning of Ukraine and the establishment of the Ukraine S.S.R. These are often references in the context of the recent conflict as well as the conflict in Crimea in 2014.

As before some irrelevant concepts such as *highly customized workflow* and *attorney editor expertise* are mixed in. These too could be filtered.

```
## `summarise()` has grouped output by 'published'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 7 x 7
## # Groups:   published [7]
##   published  `1`                          `2`           `3`   `4`   `5`   `6`
##   <chr>      <chr>                        <chr>         <chr> <chr> <chr> <chr>
## 1 2022-04-16 ukrainian president volodymyr directly per~ pers~ priv~ prov~ stor~
## 2 2022-04-15 ukrainian president volodymyr personally i~ prov~ blac~ pres~ russ~
## 3 2022-04-14 1.5 million supporters       1917 russian~ civi~ conf~ cris~ econ~
```

```
## 4 2022-04-13 ukrainian president volodymyr personally i~ prov~ pres~ russ~ pres~
## 5 2022-04-12 president vladimir putin      ukrainian pr~ russ~ pers~ pres~ pres~
## 6 2022-04-11 ukrainian president volodymyr president vl~ russ~ pers~ prov~ pres~
## 7 2022-04-10 ukrainian president volodymyr president vl~ russ~ 1.5 ~ 1917~ chan~
```

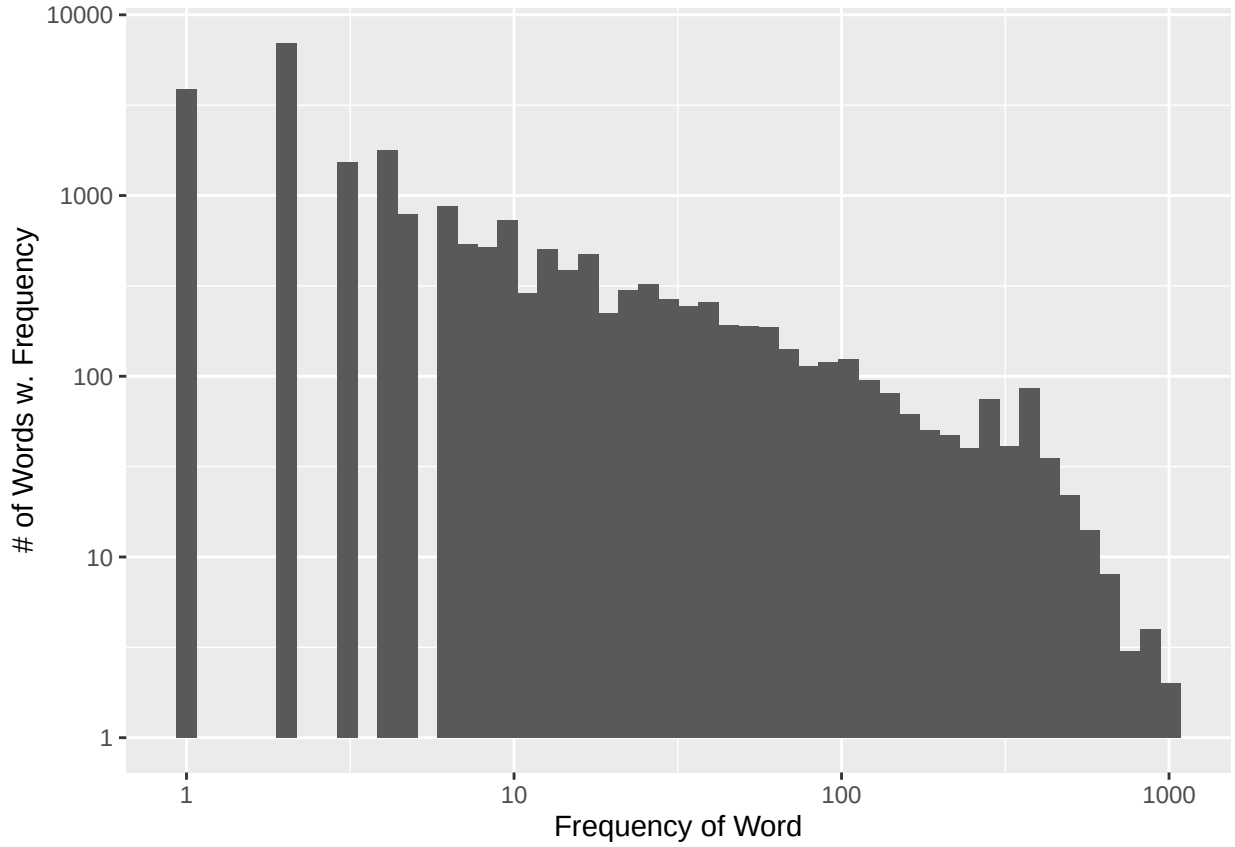
Comparative Analysis

Different sources may have different priorities, biases and criteria for selecting source material. While it's difficult to characterize individual sources based on a relatively small sample of articles, comparing credible sources such as *The Guardian* and *Al Jazeera* does show some differences in content and indicates if certain source are more useful with respect to the dimensions of the conflict, e.g. finance, politics, etc.

Once again, using the techniques from class, analyze word usage across different sources. In this snippet, will look at "Al Jazeera" vs "The Guardian". Plot.

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

```
## Warning: Removed 8 rows containing missing values (geom_bar).
```



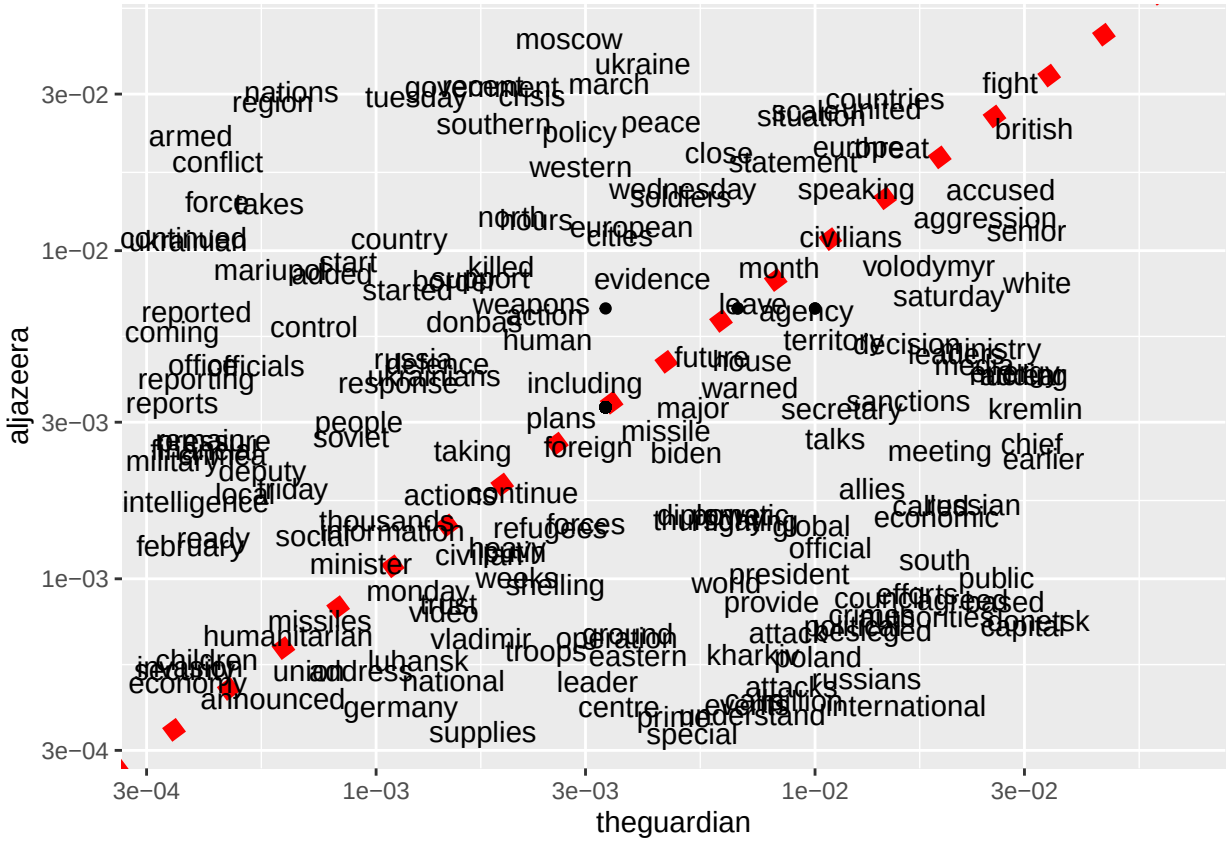
Word Usage

As we can see the plot above, word usage is highly skewed so will focus on only the most significant terms, top 300. In the plot below, we can see some differentiation in the most common terms. With the *Al Jazeera*, there is an international perspective with French contribution to diplomatic efforts and US efforts to publicize the

intelligence. Crimea figures prominently in a number of articles as both foreshadowing the current conflict and a missed opportunity for the West to potentially curb Russian aggression earlier.

```
## Warning: Removed 224 rows containing missing values (geom_point).
```

```
## Warning: Removed 224 rows containing missing values (geom_text).
```



Word Usage

The Guardian features peace conference as well as British involvement more prominently. Both sources highlight peace efforts at the start of the conflict though The Guardian is more pronounced in showing Germany's reluctance to provide weapons and take a harder line. This is even more visible in the trigrams which retain more of the context as opposed to individual teerms. So a comparative analysis of trigrams is a logical next step.

```
##    aljazeera_only theguardian_only
## 1       zelenskyy        zelenskiy
## 2           wrote              won
## 3        websites               we
## 4         website            vital
## 5      washington         upheaval
## 6         warning      unthinkable
## 7       voiceless           unlike
## 8           voice      uncertainty
```

```
## 9         updates         truthful
## 10         unique            truth
## 11        twitter      transformed
## 12          tools            track
## 13       tensions            times
## 14      targeting          telling
## 15        systems            tanks
## 16         switch         sustains
## 17         sunday       supporters
## 18      strategic           supply
## 19          store       substitute
## 20      stability     spokesperson
```

When comparing the most common words that are unique to one source or the other, we see some oddities such as different spellings of *Zelenskyy*. While seemingly a minor data consistency issue, it's an example of transliteration issues and lack of common standards when translating names from Cyrillic to English. Thus it's relevant to the mechanics of generating unique concepts as well as the topic itself.

From this small set, we can see some differences in terminology. For *Al Jazeera* terms such as territory, strategy, supplies, store emphasize the practical and objective side of the story. While *The Guardian* includes perhaps a slightly more subjective assessment, with emotional terms being more evident. For example" *unthinkable*, *vital*, *upheaval*.

https://www.cnn.com/2022/03/17/politics/how-to-spell-volodymyr-zelenskyy/index.html

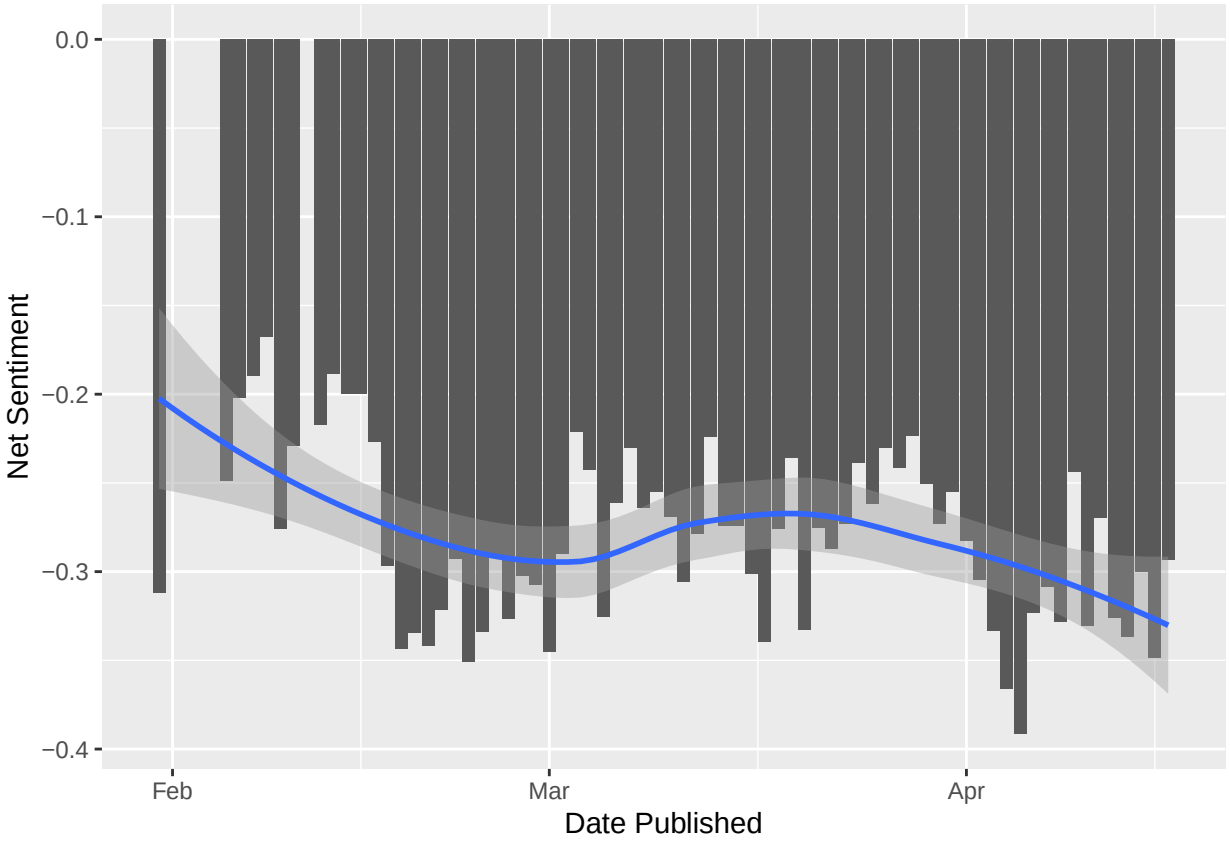
Sentiment Analysis

Sentiment analysis can have some useful practical applications beyond simply assessing the emotional state. Pronounced shifts in emotion might indicate transitions in the narrative and could be useful with segmentation and partitioning of the corpus.

In the Ukraine conflict, there was the lead up to the conflict with initial expectation of a diplomatic solution, to the early low-intensity fight and more recently the the escalation as Russian was fustrated in achieving a quick victory.

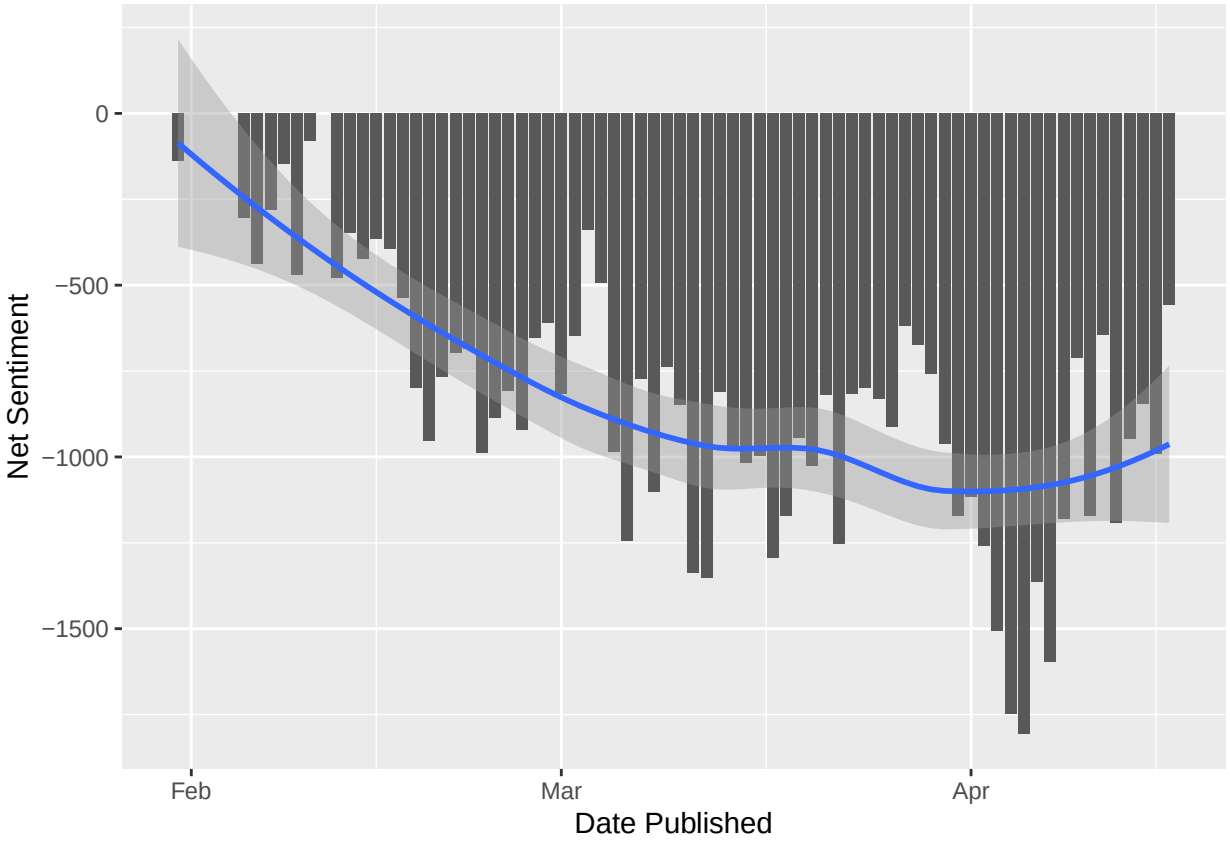
The code evaluate sentiment to see if these shifts are especially pronounced. First, usingthe **Bing** dictionary to categorize terms as positive vs negative and aggregated by day.

```
## Joining, by = "word"
## `summarise()` has grouped output by 'published'. You can override using the `.groups` argument.
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Comparing this with the **Afinn** dictionary reveals similar patterns though the effect is less pronounced in the former. This might be a result of Afinn assigning a numerical value to each word as opposed to a simple negative/positive rating. A cusrsory inspection of the upswing in mid April seems to suggest that the sentiment is exagaerated. It's unclear if the sources are (over)reacting to potential for the conflict to end or as a result of the analysis itself.

```
## Joining, by = "word"
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



At the time of this sentiment shift, Russia's 1st operational phase, which included capturing Kyiv had essentially stalled. By 23rd of March Ukraine had successfully pushed back the front lines east of Kyiv. That combined with increasing assistance from the west, especially UK and US, suggests a slight improvement in expectations on the outcome in favor of Ukraine.

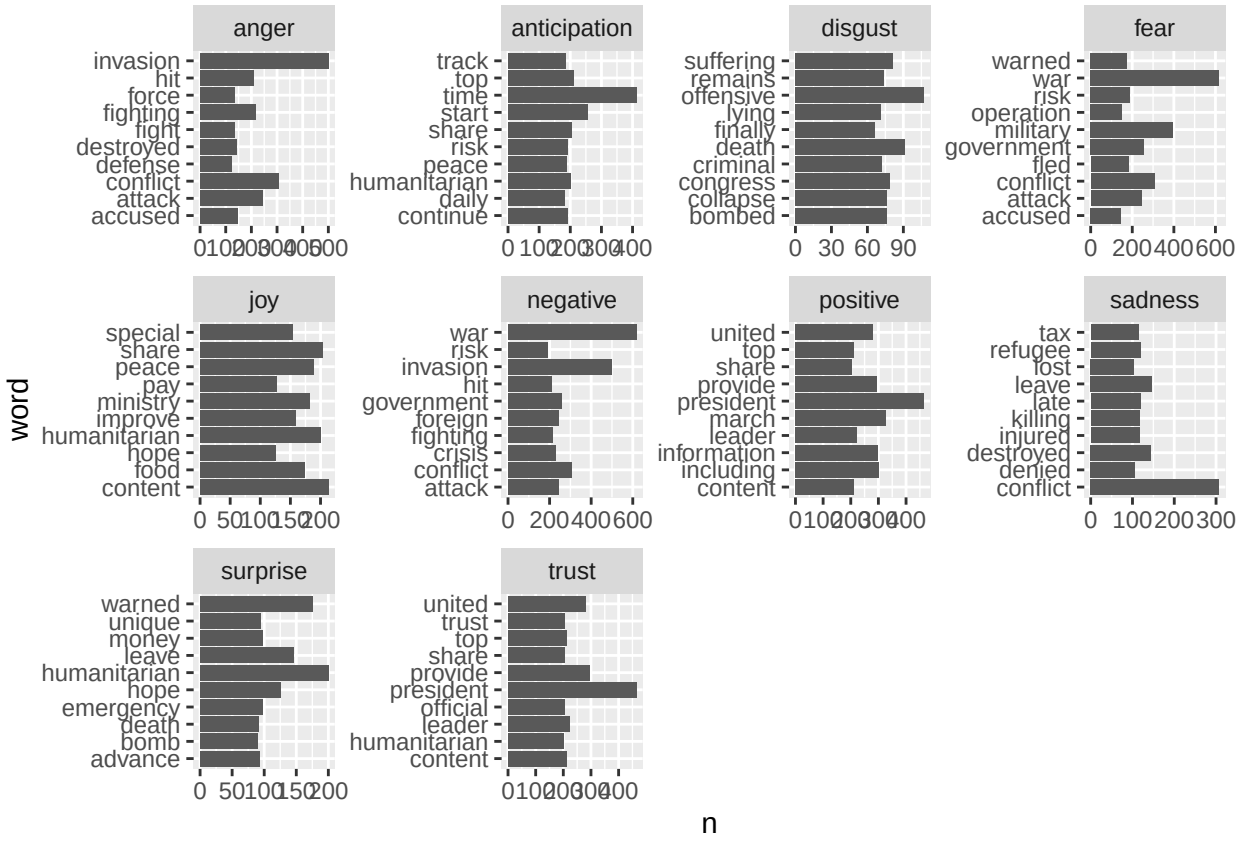
https://en.wikipedia.org/wiki/Timeline_of_the_2022_Russian_invasion_of_Ukraine#Prelude

Looking at the top words by sentiment/emotion for the time period reveals the expected pattern. Words such as *invasion*, *destroyed* and *conflict* are associated with **anger**. From the terms associated with **disgust**, it's possible to see references to *criminal* and *bombed* behavior, likely with regard to war crimes. On the other hand *humanitarian* and *peace* initiatives contributed to the *anticipation* and *joy* sentiments.

However, *trust* seems less credible since the terms such as *president*, *leader* and *official* are probably taken out of context and can be associated with untrustworthy actors.

Likewise *humanitarian* can refer to the humanitarian disaster just as easily as to the overwhelmingly positive response from host nations such as Poland and Romania.

```
## Joining, by = "word"
## `summarise()` has grouped output by 'sentiment'. You can override using the `.groups` argument.
```

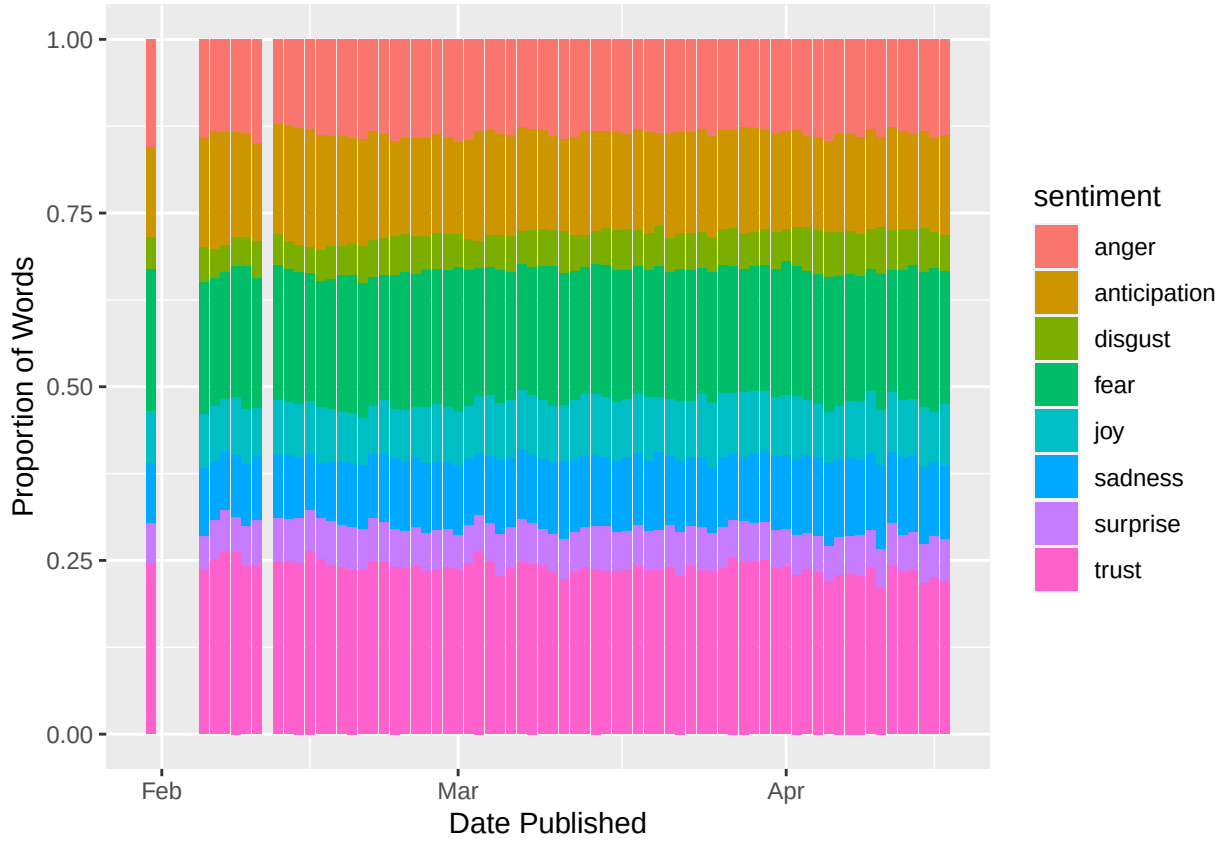


Sentiment analysis seems less useful in this application without extracting and associated relavent semantics. When dealing with complex and nuanced subject matter, assigning a semantic score to each word does not reveal any additional meaning and might actually be misleading.

What was surprising is the level of disgust, fear and sadness did not increase as much as expected especially in April as the scale of atrocities grew and coverage intensified across sources. As shown in the plot below, it's clear that all grew modestly while anticipation shrank, possible as it became clear this was not going to be a short war.

Given the questions on the usefulness of "trust" sentiment, it's not surprising there wasn't a significant shift here. However, Western governments have shared a great deal intelligence so trust levels in the West should be higher. However, it's unlikely that the sentiment analysis would pick up on this nuance.

```
## Joining, by = "word"
## `summarise()` has grouped output by 'published'. You can override using the `.groups` argument.
```

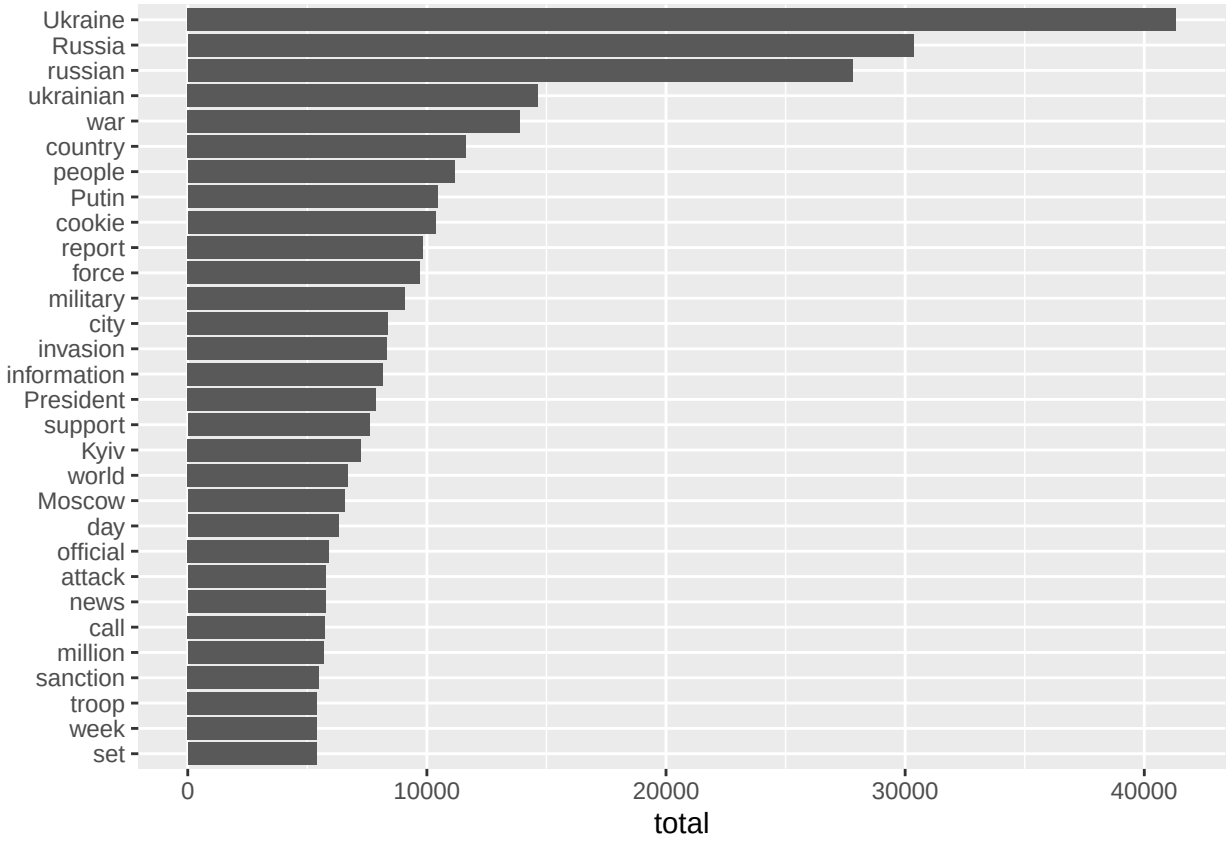


Semantic Analysis

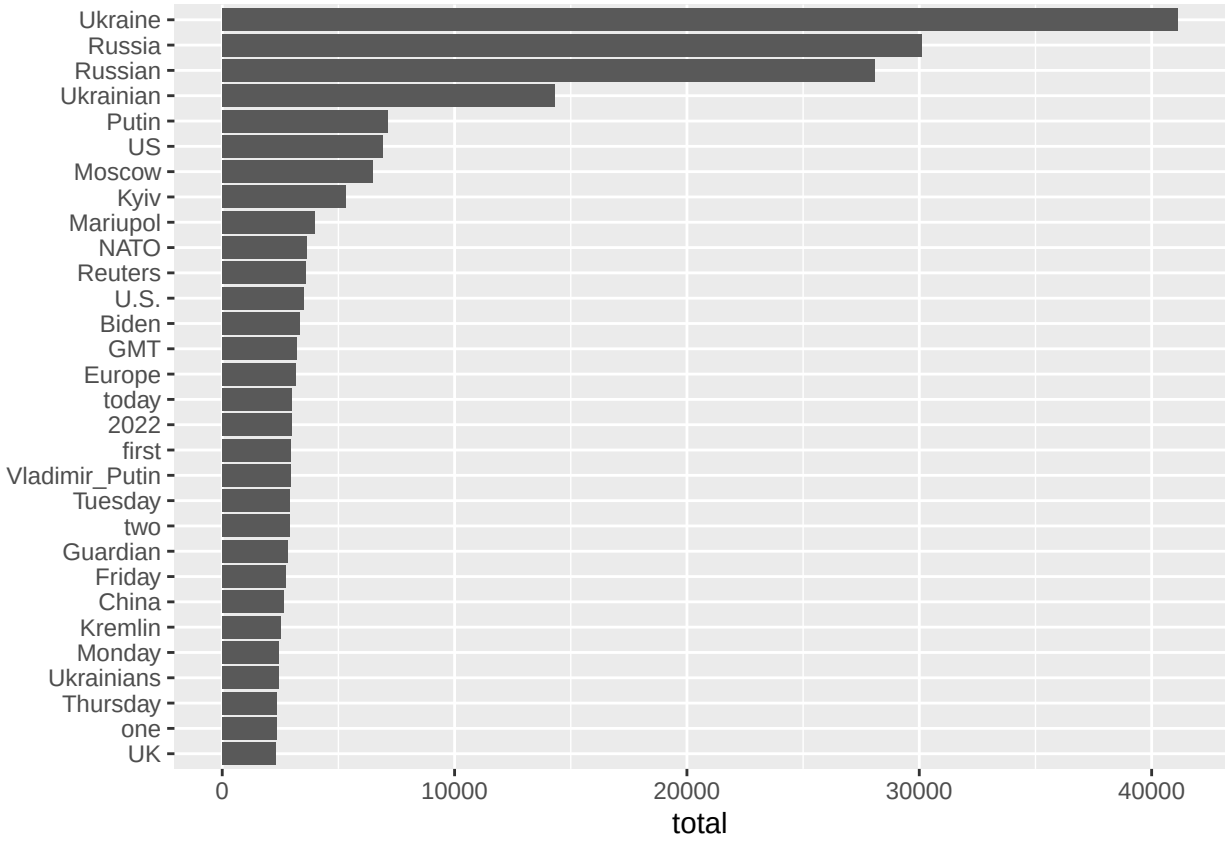
This project took the first step with semantic analysis by using the SpacyR package to extract **lemmas**, **entities** and **noun-phrases**. The goal was to supplement or supplant terms, trigrams from the classic text mining with the expectation that more meaningful results could be obtained.

As shown below, we can see that the top terms are quite meaningful and relevant to the Ukraine War.

```
## # A tibble: 1,195,020 x 4
##    word        n id                                   published
##    <chr>   <int> <chr>                                <date>
##  1 Germany    26 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
##  2 Ukraine    17 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
##  3 Russia     14 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
##  4 cookie     10 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
##  5 german     10 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
##  6 country     9 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
##  7 Al          7 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
##  8 Jazeera     7 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
##  9 provide     7 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## 10 website     7 c6f5fb44-3cd6-4065-93b1-064ed8aff0d7 2022-01-31
## # ... with 1,195,010 more rows
```

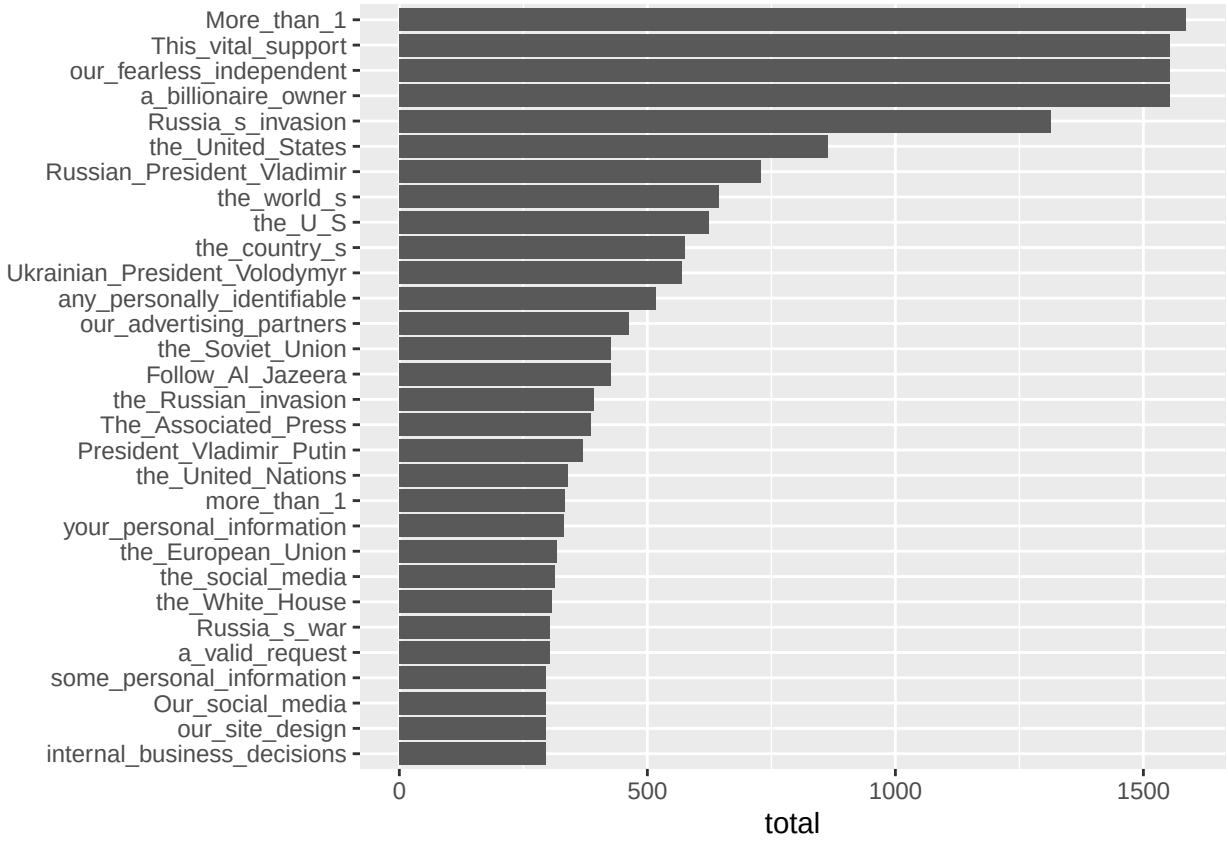


Likewise, when reviewing the list of top entities, with the exception of a few examples such as *GMT* and *one*, it clearly has fewer irrelevant terms as compared to trigrams extracted earlier.




```
## Warning: Expected 3 pieces. Additional pieces discarded in 70973 rows [40, 42,
## 47, 49, 53, 57, 64, 72, 77, 89, 98, 101, 102, 103, 104, 105, 107, 108, 109,
## 111, ...].
```

```
## Warning: Expected 3 pieces. Missing pieces filled with `NA` in 369225 rows [1,
## 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

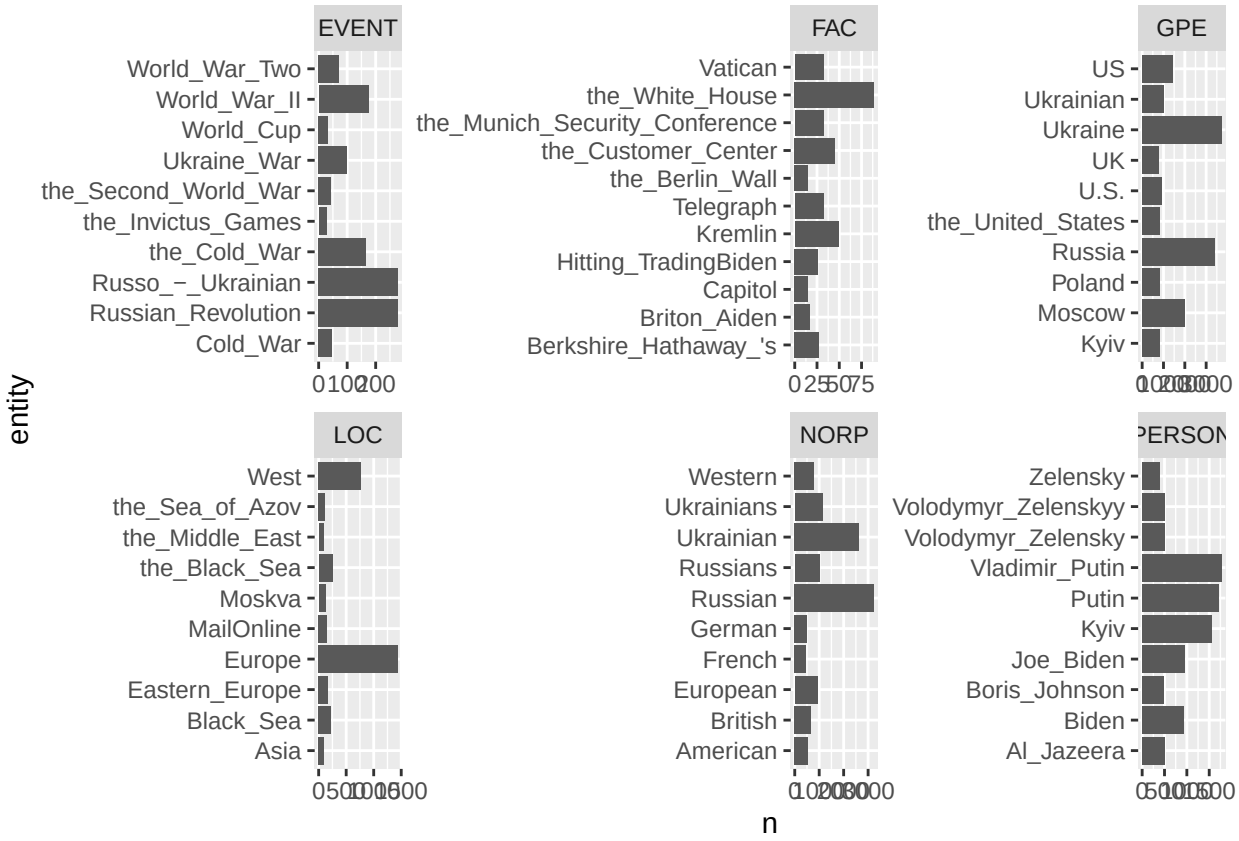


Entity extraction when combined with custom parsing via SpacyR *nlp* function could be the next step in extract meaningful phrases using the *SVO* (subject, verb, object) format. See this article.

https://suttipong-kull.medium.com/how-to-extract-subject-verb-and-object-by-nlp-4149323a7d7d

While a bit noisy, the entity and types are clearly relavent and figure prominently in the conflict. So the semantic extraction seems to be working well. Additional filtering could be useful in combination with temporal analysis.

```
## `summarise()` has grouped output by 'entity_type'. You can override using the
## `.groups` argument.
```



Please see this legend for a description of each of the entity types.

```
## # A tibble: 18 x 2
##    Type        Description
##    <chr>       <chr>
##  1 PERSON      People, including fictional.
##  2 NORP        Nationalities or religious or political groups.
##  3 FAC         Buildings, airports, highways, bridges, etc.
##  4 ORG         Companies, agencies, institutions, etc.
##  5 GPE         Countries, cities, states.
##  6 LOC         Non-GPE locations, mountain ranges, bodies of water.
##  7 PRODUCT     Objects, vehicles, foods, etc. (Not services.)
##  8 EVENT       Named hurricanes, battles, wars, sports events, etc.
##  9 WORK_OF_ART Titles of books, songs, etc.
## 10 LAW         Named documents made into laws.
## 11 LANGUAGE    Any named language.
## 12 DATE        Absolute or relative dates or periods.
## 13 TIME        Times smaller than a day.
## 14 PERCENT     Percentage, including "%".
## 15 MONEY       Monetary values, including unit.
## 16 QUANTITY    Measurements, as of weight or distance.
## 17 ORDINAL     "first", "second", etc.
## 18 CARDINAL    Numerals that do not fall under another type.
```

The R News Digest app includes many of these graphs and controls for filtering out data. It also includes a configurable *Topic Modeling* for deriving categories from the corpus.

Please review the application for additional insights. Thanks for your attention!