Package ‘ngramr’

January 16, 2023

Type Package
Title Retrieve and Plot Google n-Gram Data
Version 1.9.3
Date 2023-01-16
Maintainer Sean Carmody <seancarmody@gmail.com>
Description Retrieve and plot word frequencies through time from the "Google Ngram Viewer" <https://books.google.com/ngrams>.
Depends R (>= 4.0.0)
Imports httr, rlang, curl, dplyr (>= 1.0.3), cli, tibble, tidyr, rjson, stringr, ggplot2, scales, xml2, textutils
URL https://github.com/seancarmody/ngramr
BugReports https://github.com/seancarmody/ngramr/issues
License MIT + file LICENSE
RoxygenNote 7.2.2
Encoding UTF-8
Suggests testthat
Language en-AU
NeedsCompilation no
Author Sean Carmody [aut, cre, cph]
Repository CRAN
Date/Publication 2023-01-16 11:30:02 UTC

R topics documented:

chunk ........................................................................... 2
corporuses ................................................................... 3
ggram ......................................................................... 3
hacker ......................................................................... 5
ngram ......................................................................... 5
chunk

Description

chunk takes a vector (or list) and returns a list of chunks which all have lengths (approximately) equal to a specified value.

Usage

chunk(x, len = NULL, n = NULL)

Arguments

x vector of list
len target length of chunks
n number of chunks

Details

If n is specified, len is ignored and chunk returns a list of length n of "chunks" of x. Otherwise n is calculated to break the vector into chunks which are each approximately of length len. If both len and n are unspecified, chunk simply returns x.

Examples

chunk(letters, 10)
chunk(LETTERS, n = 3)
corpuses

Google n-gram corpus information

Description
Details of the various corpuses available through the Google n-gram tool

Usage
corpuses

Format
a 33 x 6 ngram data frame

ggram

Plot n-gram frequencies

Description
ggram downloads data from the Google Ngram Viewer website and plots it in ggplot2 style.

Usage
ggram(
  phrases,
  ignore_case = FALSE,
  code_corpus = FALSE,
  geom = "line",
  geom_options = list(),
  lab = NA,
  google_theme = FALSE,
  ...
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phrases</td>
<td>vector of phrases. Alternatively, phrases can be an ngram object returned by ngram or ngrami.</td>
</tr>
<tr>
<td>ignore_case</td>
<td>logical, indicating whether the frequencies are case insensitive. Default is FALSE.</td>
</tr>
<tr>
<td>code_corpus</td>
<td>logical, indicating whether to use abbreviated corpus 'codes or longer form descriptions. Default is FALSE.</td>
</tr>
<tr>
<td>geom</td>
<td>the ggplot2 geom used to plot the data; defaults to &quot;line&quot;</td>
</tr>
<tr>
<td>geom_options</td>
<td>list of additional parameters passed to the ggplot2 geom.</td>
</tr>
</tbody>
</table>
lab y-axis label. Defaults to "Frequency".
google_theme use a Google Ngram-style plot theme.
... additional parameters passed to ngram

Details

Google generated two datasets drawn from digitised books in the Google books collection. One was generated in July 2009, the second in July 2012. Google will update these datasets as book scanning continues.

Examples

```r
caption
library(ggplot2)
ggram(c("hacker", "programmer"), year_start = 1950)

# Changing the geom.
ggram(c("cancer", "fumer", "cigarette"),
  year_start = 1900,
  corpus = "fr-2012",
  smoothing = 0,
  geom = "step")

# Passing more options.
ggram(c("cancer", "smoking", "tobacco"),
  year_start = 1900,
  corpus = "en-fiction-2012",
  geom = "point",
  smoothing = 0,
  geom_options = list(alpha = .5)) +
  stat_smooth(method="loess", se = FALSE, formula = y ~ x)

# Setting the layers manually.
ggram(c("cancer", "smoking", "tobacco"),
  year_start = 1900,
  corpus = "en-fiction-2012",
  smoothing = 0,
  geom = NULL) +
  stat_smooth(method="loess", se=FALSE, span = 0.3, formula = y ~ x)

# Setting the legend placement on a long query and using the Google theme.
# Example taken from a post by Ben Zimmer at Language Log.
p <- c("((The United States is + The United States has) / The United States)",
  "((The United States are + The United States have) / The United States")")
ggram(p, year_start = 1800, google_theme = TRUE) +
  theme(legend.direction="vertical")

# Pass ngram data rather than phrases

ggram(hacker) + facet_wrap(~ Corpus)
```

```
Sample n-gram data

Description

Frequency data for the phrases "hacker", "programmer", from 1950 to 2008.

Usage

hacker

Format

a 236 x 4 ngram data frame

ngram

Get n-gram frequencies

Description

ngram downloads data from the Google Ngram Viewer website and returns it in a tibble.

Usage

ngram(
  phrases,  # vector of phrases, with a maximum of 12 items
  corpus = "en-2019",  # Google corpus to search (see Details for possible values)
  year_start = 1800,  # start year, default is 1800. Data available back to 1500.
  year_end = 2020,  # end year, default is 2008
  smoothing = 3,  #
  case_ins = FALSE,  #
  aggregate = FALSE,  #
  count = FALSE,  #
  drop_corpus = FALSE,  #
  drop_parent = FALSE,  #
  drop_all = FALSE,  #
  type = FALSE  #
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phrases</td>
<td>vector of phrases, with a maximum of 12 items</td>
</tr>
<tr>
<td>corpus</td>
<td>Google corpus to search (see Details for possible values)</td>
</tr>
<tr>
<td>year_start</td>
<td>start year, default is 1800. Data available back to 1500.</td>
</tr>
<tr>
<td>year_end</td>
<td>end year, default is 2008</td>
</tr>
</tbody>
</table>
smoothing smoothing parameter, default is 3

case_ins Logical indicating whether to force a case insensitive search. Default is FALSE.

aggregate Sum up the frequencies for ngrams associated with wildcard or case insensitive searches. Default is FALSE.

count Default is FALSE.

drop_corpus When a corpus is specified directly with the ngram (e.g dog:eng_fiction_2012) specifies whether the corpus be used retained in the phrase column of the results. Note that that this method requires that the old corpus codes (eng_fiction_2012 not en-fiction-2012) are used. Default is FALSE.

drop_parent Drop the parent phrase associated with a wildcard or case-insensitive search. Default is FALSE.

drop_all Delete the suffix "(All)" from aggregated case-insensitive searches. Default is FALSE.

type Include the Google return type (e.g. NGRAM, NGRAM_COLLECTION, EXPANSION) from result set. Default is FALSE.

Details

Google generated two datasets drawn from digitised books in the Google Books collection. One was generated in July 2009, the second in July 2012 and the third in 2019. Google is expected to update these datasets as book scanning continues.

This function provides the annual frequency of words or phrases, known as n-grams, in a sub-collection or "corpus" taken from the Google Books collection. The search across the corpus is case-sensitive.

If the function is unable to retrieve data from the Google Ngram Viewer site (either because of access issues or if the format of Google’s site has changed) a NULL result is returned and messages are printed to the console but no errors or warnings are raised (this is to align with CRAN package policies).

Below is a list of available corpora. Note that the data for the 2012 corpuses only extends to 2009.

<table>
<thead>
<tr>
<th>Corpus</th>
<th>Corpus Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>en-GB-2019</td>
<td>British English 2019</td>
</tr>
<tr>
<td>en-GB-2012</td>
<td>British English 2012</td>
</tr>
<tr>
<td>en-GB-2009</td>
<td>British English 2009</td>
</tr>
<tr>
<td>en-2019</td>
<td>English 2019</td>
</tr>
<tr>
<td>en-2012</td>
<td>English 2012</td>
</tr>
<tr>
<td>en-2009</td>
<td>English 2009</td>
</tr>
<tr>
<td>en-fiction-2019</td>
<td>English Fiction 2019</td>
</tr>
<tr>
<td>en-fiction-2012</td>
<td>English Fiction 2012</td>
</tr>
<tr>
<td>en-fiction-2009</td>
<td>English Fiction 2009</td>
</tr>
</tbody>
</table>
The Google Million is a sub-collection of Google Books. All are in English with dates ranging from 1500 to 2008. No more than about 6,000 books were chosen from any one year, which means that all of the scanned books from early years are present, and books from later years are randomly sampled. The random samplings reflect the subject distributions for the year (so there are more computer books in 2000 than 1980).


**Value**

ngram returns an object of class "ngram", which is a tidyverse tibble enriched with attributes reflecting some of the parameters used in the Ngram Viewer query.

**Examples**

```r
ngram(c("mouse", "rat"), year_start = 1950)
ngram(c("blue_ADJ", "red_ADJ"))
ngram(c("_START_ President Roosevelt", "_START_ President Truman"), year_start = 1920)
```

**ngrami**

Get n-gram frequencies (case insensitive version)

**Description**

This function is a simple wrapper of ngram for case insensitive searches.
Usage

`ngrami(phrases, aggregate = TRUE, ...)`

Arguments

- `phrases`: vector of phrases
- `aggregate`: sum up each of the terms
- `...`: remaining parameters passed to ngram

---

`ngramw`      
Get n-gram frequencies ("wide" format)

Description

Get n-gram frequencies ("wide" format)

Usage

`ngramw(phrases, ignore_case = FALSE, ...)`

Arguments

- `phrases`: vector of phrases
- `ignore_case`: ignore case of phrases (i.e. call ngrami rather than ngram). Default value is FALSE.
- `...`: remaining parameters passed to ngram

---

`print.ngram`     
Print n-gram contents

Description

Print n-gram contents

Usage

```r
## S3 method for class 'ngram'
print(x, rows = 6, ...)
```

Arguments

- `x`: ngram object as returned by link{ngram}
- `rows`: number of rows to print. Default is 6.
- `...`: additional parameters passed to default print method.
theme_google

Examples

```r
x <- ngram(c("hacker", "programmer"), year_start = 1950)
print(x)
```

theme_google

Google Ngram theme for ggplot2

Description

Google Ngram theme for ggplot2

Usage

```r
theme_google(...)
```

Arguments

... additional parameters to pass to theme

Details

Use a Google Ngram-style plot theme.
**Index**

* **datasets**
  - corpuses, 3
  - hacker, 5

chunk, 2

corpuses, 3

ggram, 3

hacker, 5

ngram, 3, 5
ngrami, 3, 7
ngramw, 8

print.ngram, 8

theme_google, 9