Package: sylcount (via r-universe)

November 2, 2026

Type Package

Title Syllable Counting and Readability Measurements

Version 0.2-6

Description An English language syllable counter, plus readability score measure-er. For readability, we support 'Flesch' Reading Ease and 'Flesch-Kincaid' Grade Level ('Kincaid' 'et al'. 1975) <https://stars.library.ucf.edu/cgi/viewcontent.cgi?article=1055&context=istlibrary>, Automated Readability Index ('Senter' and Smith 1967) <https://apps.dtic.mil/sti/citations/AD0667273>, Simple Measure of Gobbledygook (McLaughlin 1969), and 'Coleman-Liau' (Coleman and 'Liau' 1975) <doi:10.1037/h0076540>. The package has been carefully optimized and should be very efficient, both in terms of run time performance and memory consumption. The main methods are 'vectorized' by document, and scores for multiple documents are computed in parallel via 'OpenMP'.

License BSD 2-clause License + file LICENSE

Depends R (>= 3.0.0)

LazyLoad yes

NeedsCompilation yes

ByteCompile yes

Maintainer Drew Schmidt <wrathematics@gmail.com>

URL https://github.com/wrathematics/sylcount

BugReports https://github.com/wrathematics/sylcount/issues

RoxygenNote 7.1.2

Author Drew Schmidt [aut, cre]

Repository https://cran.r-universe.dev

Date/Publication 2023-12-11 02:41:20 UTC

RemoteUrl https://github.com/cran/sylcount

RemoteRef HEAD

RemoteSha 29940e9c7bc7ea531d7ecdc822b942050631b7cc

Contents

sylcount-package 2
doc_counts 2
readability 3
sylcount 5
sylcount.nthreads 6

Index **7**

`sylcount-package` *sylcount: Syllable Counting and Readability Measurements*

Description

An English language syllable counter, plus readability score measure-er. For readability, we support Flesch Reading Ease and Flesch-Kincaid Grade Level (Kincaid et al. 1975), Automated Readability Index (Senter and Smith 1967), Simple Measure of Gobbledygook (McLaughlin 1969), and Coleman-Liau (Coleman and Liau 1975). The package has been carefully optimized and should be very efficient, both in terms of run time performance and memory consumption. The main methods are vectorized by document, and scores for multiple documents are computed in parallel via OpenMP.

Author(s)

Drew Schmidt

`doc_counts` *doc_counts*

Description

Computes some basic document counts (see the 'Value' section below for details).

The function is vectorized by document, and scores are computed in parallel via OpenMP. You can control the number of threads used with the `nthreads` parameter.

Usage

```
doc_counts(s, nthreads = sylcount.nthreads())
```

Arguments

`s`	A character vector (vector of strings).
`nthreads`	Number of threads to use. By default it will use the total number of cores + hyperthreads.

Details

The function is essentially just `readability()` without the readability scores.

Value

A dataframe containing:

`chars`	the total numberof characters
`wordchars`	the number of alphanumeric characters
`words`	text tokens that are probably English language words
`nonwords`	text tokens that are probably not English language words
`sents`	the number of sentences recognized in the text
`sylls`	the total number of syllables (ignores all non-words)
`polys`	the total number of "polysyllables", or words with 3+ syllables

See Also

`readability`

Examples

```
library(sylcount)
a <- "I am the very model of a modern major general."
b <- "I have information vegetable, animal, and mineral."

doc_counts(c(a, b), nthreads=1)
```

`readability` *readability*

Description

Computes some basic "readability" measurements, includeing Flesch Reading Ease, Flesch-Kincaid grade level, Automatic Readability Index, and the Simple Measure of Gobbledygook. The function is vectorized by document, and scores are computed in parallel via OpenMP. You can control the number of threads used with the `nthreads` parameter.

The function will have some difficulty on poorly processed and cleaned data. For example, if all punctuation is stripped out, then the number of sentences detected will always be zero. However, we do recommend removing quotes (single and double), as contractions can confuse the parser.

Usage

```
readability(s, nthreads = sylcount.nthreads())
```

Arguments

s	A character vector (vector of strings).
nthreads	Number of threads to use. By default it will use the total number of cores + hyperthreads.

Details

The return is split into words and non-words. A non-word is some block of text more than 64 characters long with no spaces or sentence-ending punctuation inbetween. The number of non-words is returned mostly for error-checking/debugging purposes. If you have a lot of non-words, you probably didn't clean your text properly. The word/non-word division is made in an attempt to improve run-time and memory performance.

For implementation details, see the Details section of `?sylcount`.

Value

A dataframe containing:

chars	the total numberof characters
wordchars	the number of alphanumeric characters
words	text tokens that are probably English language words
nonwords	text tokens that are probably not English language words
sents	the number of sentences recognized in the text
sylls	the total number of syllables (ignores all non-words)
polys	the total number of "polysyllables", or words with 3+ syllables
re	Flesch reading ease score
gl	Flesch-Kincaid grade level score
ari	Automatic Readability Index score
smog	Simple Measure of Gobbledygook (SMOG) score
cl	the Coleman-Liau Index score

References

Kincaid, J. Peter, et al. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. No. RBR-8-75. Naval Technical Training Command Millington TN Research Branch, 1975.

Senter, R. J., and Edgar A. Smith. Automated readability index. CINCINNATI UNIV OH, 1967.

McLaughlin, G. Harry. "SMOG grading-a new readability formula." Journal of reading 12.8 (1969): 639-646.

Coleman, Meri, and Ta Lin Liau. "A computer readability formula designed for machine scoring." Journal of Applied Psychology 60.2 (1975): 283.

See Also

`doc_counts`

Examples

```
library(sylcount)
a <- "I am the very model of a modern major general."
b <- "I have information vegetable, animal, and mineral."

# One or the other
readability(a, nthreads=1)
readability(b, nthreads=1)

# Bot at once as separate documents.
readability(c(a, b), nthreads=1)
# And as a single document.
readability(paste0(a, b, collapse=" "), nthreads=1)
```

`sylcount` *sylcount*

Description

A vectorized syllable counter for English language text.

Because of the R memory allocations required, the operation is not thread safe. It is evaluated in serial.

Usage

```
sylcount(s, counts.only = TRUE)
```

Arguments

`s`	A character vector (vector of strings).
`counts.only`	Should only counts be returned, or words + counts?

Details

The maximum supported word length is 64 characters. For any token having more than 64 characters, the returned syllable count will be `NA`.

The syllable counter uses a hash table of known, mostly "irregular" (with respect to syllable counting) words. If the word is not known to us (i.e., not in the hash table), then we try to "approximate" the number of syllables by counting the number of non-consecutive vowels in a word.

So for example, using this scheme, each of "to", "too", and "tool" would be classified as having one syllable. However, "tune" would be classified as having 2. Fortunately, "tune" is in our table, listed as having 1 syllable.

The hash table uses a perfect hash generated by gperf.

Value

A list of dataframes.

See Also

readability

Examples

```
library(sylcount)
a <- "I am the very model of a modern major general."
b <- "I have information vegetable, animal, and mineral."

sylcount(c(a, b))
sylcount(c(a, b), counts.only=FALSE)
```

---

sylcount.nthreads *sylcount.nthreads*

Description

Returns the number of cores + hyperthreads on the system. The function respects the environment variable OMP_NUM_THREADS.

Usage

```
sylcount.nthreads()
```

Value

An integer; the number of threads.

Index

∗ **Package**
sylcount-package, 2

doc_counts, 2, *4*

readability, 3, *3*, *6*

sylcount, 5
sylcount-package, 2
sylcount.nthreads, 6